Linux Admin-guide Documentation

Release 6.8.0

The kernel development community

Jan 16, 2026

CONTENTS

1 Linux kernel release 6.x <http://kernel.org/> 3

2 The kernel's command-line parameters 11

3 Linux allocated devices (4.x+ version) 167

4 Documentation for /proc/sys 235

5 Linux ABI description 303

6 Feature status on all architectures 1383

7 Hardware vulnerabilities 1397

8 Reporting issues 1451

9 Reporting regressions 1479

10 How to quickly build a trimmed Linux kernel 1487

11 Bug hunting 1505

12 Bisecting a bug 1513

13 Tainted kernels 1515

14 Ramoops oops/panic logger 1519

15 Dynamic debug 1523

16 Explaining the ”No working init found.” boot hang message 1531

17 Documentation for Kdump - The kexec-based Crash Dumping Solution 1533

18 Performance monitor support 1561

19 pstore block oops/panic logger 1589

20 Rules on how to access information in sysfs 1601

21 Discovering Linux kernel subsystems used by a workload 1605

22 ACPI Support 1617

23 ATA over Ethernet (AoE) 1627
24 Auxiliary Display Support 1633
25 A block layer cache (bcache) 1637
26 The Android binderfs Filesystem 1649
27 Kernel Support for miscellaneous Binary Formats (binfmt_misc) 1651
28 Block Devices 1655
29 Boot Configuration 1681
30 Linux Braille Console 1695
31 btmrvl driver 1697
32 Control Groups version 1 1701
33 Control Group v2 1769
34 CIFS 1817
35 Clearing WARN_ONCE 1839
36 CPU load 1841
37 How CPU topology info is exported via sysfs 1845
38 Dell Remote BIOS Update driver (dell_rbu) 1847
39 Device Mapper 1851
40 EDID 1957
41 The EFI Boot Stub 1959
42 ext4 General Information 1961
43 File system Monitoring with fanotify 1973
44 NFS 1975
45 gpio 1991
46 Notes on the change from 16-bit UIDs to 32-bit UIDs 1999
47 Hardware random number generators 2001
48 Using the initial RAM disk (initrd) 2003
49 I/O statistics fields 2011
50 Java(tm) Binary Kernel Support for Linux v1.03 2015
51 IBM's Journaled File System (JFS) for Linux 2023

52 Reducing OS jitter due to per-cpu kthreads 2025
53 Laptop Drivers 2031
54 Parallel port LCD/Keypad Panel support 2091
55 LDM - Logical Disk Manager (Dynamic Disks) 2093
56 Softlockup detector and hardlockup detector (aka nmi_watchdog) 2097
57 Linux Security Module Usage 2099
58 RAID arrays 2121
59 Media subsystem admin and user guide 2133
60 Memory Management 2379
61 Kernel module signing facility 2485
62 Mono(tm) Binary Kernel Support for Linux 2491
63 Namespaces 2493
64 Numa policy hit/miss statistics 2495
65 Parport 2497
66 Perf events and tool security 2503
67 Power Management 2509
68 Set udev rules for PMF Smart PC Builder 2601
69 Linux Plug and Play Documentation 2603
70 RapidIO Subsystem Guide 2609
71 Reliability, Availability and Serviceability 2657
72 Real Time Clock (RTC) Drivers for Linux 2679
73 Linux Serial Console 2683
74 Video Mode Selection Support 2.13 2687
75 Syscall User Dispatch 2693
76 Linux Magic System Request Key Hacks 2695
77 Thermal Subsystem 2701
78 USB4 and Thunderbolt 2707
79 Using UFS 2713
80 Unicode support 2715

81 Software cursor for VGA 2719

82 Video Output Switcher Control 2721

83 The SGI XFS Filesystem 2723

Index 2733

The following is a collection of user-oriented documents that have been added to the kernel over time. There is, as yet, little overall order or organization here — this material was not written to be a single, coherent document! With luck things will improve quickly over time.

This initial section contains overall information, including the README file describing the kernel as a whole, documentation on kernel parameters, etc.

CHAPTER

ONE

LINUX KERNEL RELEASE 6.X <HTTP://KERNEL.ORG/>

These are the release notes for Linux version 6. Read them carefully, as they tell you what this is all about, explain how to install the kernel, and what to do if something goes wrong.

1.1 What is Linux?

Linux is a clone of the operating system Unix, written from scratch by Linus Torvalds with assistance from a loosely-knit team of hackers across the Net. It aims towards POSIX and Single UNIX Specification compliance.

It has all the features you would expect in a modern fully-fledged Unix, including true multitasking, virtual memory, shared libraries, demand loading, shared copy-on-write executables, proper memory management, and multistack networking including IPv4 and IPv6.

It is distributed under the GNU General Public License v2 - see the accompanying COPYING file for more details.

1.2 On what hardware does it run?

Although originally developed first for 32-bit x86-based PCs (386 or higher), today Linux also runs on (at least) the Compaq Alpha AXP, Sun SPARC and UltraSPARC, Motorola 68000, PowerPC, PowerPC64, ARM, Hitachi SuperH, Cell, IBM S/390, MIPS, HP PA-RISC, Intel IA-64, DEC VAX, AMD x86-64 Xtensa, and ARC architectures.

Linux is easily portable to most general-purpose 32- or 64-bit architectures as long as they have a paged memory management unit (PMMU) and a port of the GNU C compiler (gcc) (part of The GNU Compiler Collection, GCC). Linux has also been ported to a number of architectures without a PMMU, although functionality is then obviously somewhat limited. Linux has also been ported to itself. You can now run the kernel as a userspace application - this is called UserMode Linux (UML).

1.3 Documentation

- There is a lot of documentation available both in electronic form on the Internet and in books, both Linux-specific and pertaining to general UNIX questions. I'd recommend looking into the documentation subdirectories on any Linux FTP site for the LDP (Linux Documentation Project) books. This README is not meant to be documentation on the system: there are much better sources available.
- There are various README files in the Documentation/ subdirectory: these typically contain kernel-specific installation notes for some drivers for example. Please read the Documentation/process/changes.rst file, as it contains information about the problems, which may result by upgrading your kernel.

1.4 Installing the kernel source

- If you install the full sources, put the kernel tarball in a directory where you have permissions (e.g. your home directory) and unpack it:

  ```
  xz -cd linux-6.x.tar.xz | tar xvf -
  ```

  Replace "X" with the version number of the latest kernel.

  Do NOT use the /usr/src/linux area! This area has a (usually incomplete) set of kernel headers that are used by the library header files. They should match the library, and not get messed up by whatever the kernel-du-jour happens to be.

- You can also upgrade between 6.x releases by patching. Patches are distributed in the xz format. To install by patching, get all the newer patch files, enter the top level directory of the kernel source (linux-6.x) and execute:

  ```
  xz -cd ../patch-6.x.xz | patch -p1
  ```

  Replace "x" for all versions bigger than the version "x" of your current source tree, **in_order**, and you should be ok. You may want to remove the backup files (some-file-name~ or some-file-name.orig), and make sure that there are no failed patches (some-file-name# or some-file-name.rej). If there are, either you or I have made a mistake.

  Unlike patches for the 6.x kernels, patches for the 6.x.y kernels (also known as the -stable kernels) are not incremental but instead apply directly to the base 6.x kernel. For example, if your base kernel is 6.0 and you want to apply the 6.0.3 patch, you must not first apply the 6.0.1 and 6.0.2 patches. Similarly, if you are running kernel version 6.0.2 and want to jump to 6.0.3, you must first reverse the 6.0.2 patch (that is, patch -R) **before** applying the 6.0.3 patch. You can read more on this in Documentation/process/applying-patches.rst.

  Alternatively, the script patch-kernel can be used to automate this process. It determines the current kernel version and applies any patches found:

  ```
  linux/scripts/patch-kernel linux
  ```

  The first argument in the command above is the location of the kernel source. Patches are applied from the current directory, but an alternative directory can be specified as the second argument.

- Make sure you have no stale .o files and dependencies lying around:

```
cd linux
make mrproper
```

You should now have the sources correctly installed.

1.5 Software requirements

Compiling and running the 6.x kernels requires up-to-date versions of various software packages. Consult Documentation/process/changes.rst for the minimum version numbers required and how to get updates for these packages. Beware that using excessively old versions of these packages can cause indirect errors that are very difficult to track down, so don't assume that you can just update packages when obvious problems arise during build or operation.

1.6 Build directory for the kernel

When compiling the kernel, all output files will per default be stored together with the kernel source code. Using the option `make O=output/dir` allows you to specify an alternate place for the output files (including .config). Example:

```
kernel source code: /usr/src/linux-6.x
build directory:    /home/name/build/kernel
```

To configure and build the kernel, use:

```
cd /usr/src/linux-6.x
make O=/home/name/build/kernel menuconfig
make O=/home/name/build/kernel
sudo make O=/home/name/build/kernel modules_install install
```

Please note: If the `O=output/dir` option is used, then it must be used for all invocations of make.

1.7 Configuring the kernel

Do not skip this step even if you are only upgrading one minor version. New configuration options are added in each release, and odd problems will turn up if the configuration files are not set up as expected. If you want to carry your existing configuration to a new version with minimal work, use `make oldconfig`, which will only ask you for the answers to new questions.

- Alternative configuration commands are:

```
"make config"      Plain text interface.

"make menuconfig"  Text based color menus, radiolists & dialogs.
```

```
"make nconfig"     Enhanced text based color menus.

"make xconfig"     Qt based configuration tool.

"make gconfig"     GTK+ based configuration tool.

"make oldconfig"   Default all questions based on the contents of
                   your existing ./.config file and asking about
                   new config symbols.

"make olddefconfig"
                   Like above, but sets new symbols to their␣
↪default
                   values without prompting.

"make defconfig"   Create a ./.config file by using the default
                   symbol values from either arch/$ARCH/defconfig
                   or arch/$ARCH/configs/${PLATFORM}_defconfig,
                   depending on the architecture.

"make ${PLATFORM}_defconfig"
                   Create a ./.config file by using the default
                   symbol values from
                   arch/$ARCH/configs/${PLATFORM}_defconfig.
                   Use "make help" to get a list of all available
                   platforms of your architecture.

"make allyesconfig"
                   Create a ./.config file by setting symbol
                   values to 'y' as much as possible.

"make allmodconfig"
                   Create a ./.config file by setting symbol
                   values to 'm' as much as possible.

"make allnoconfig" Create a ./.config file by setting symbol
                   values to 'n' as much as possible.

"make randconfig"  Create a ./.config file by setting symbol
                   values to random values.

"make localmodconfig" Create a config based on current config and
                      loaded modules (lsmod). Disables any module
                      option that is not needed for the loaded␣
↪modules.

                      To create a localmodconfig for another␣
↪machine,
                      store the lsmod of that machine into a file
```

```
                        and pass it in as a LSMOD parameter.

                        Also, you can preserve modules in certain␣
↪folders
                        or kconfig files by specifying their paths in
                        parameter LMC_KEEP.

               target$ lsmod > /tmp/mylsmod
               target$ scp /tmp/mylsmod host:/tmp

               host$ make LSMOD=/tmp/mylsmod \
                        LMC_KEEP="drivers/usb:drivers/gpu:fs" \
                        localmodconfig

                        The above also works when cross compiling.

"make localyesconfig" Similar to localmodconfig, except it will␣
↪convert
                        all module options to built in (=y) options.␣
↪You can
                        also preserve modules by LMC_KEEP.

"make kvm_guest.config"   Enable additional options for kvm guest␣
↪kernel
                            support.

"make xen.config"   Enable additional options for xen dom0 guest␣
↪kernel
                    support.

"make tinyconfig"  Configure the tiniest possible kernel.
```

You can find more information on using the Linux kernel config tools in Documentation/kbuild/kconfig.rst.

- NOTES on `make config`:
  - Having unnecessary drivers will make the kernel bigger, and can under some circumstances lead to problems: probing for a nonexistent controller card may confuse your other controllers.
  - A kernel with math-emulation compiled in will still use the coprocessor if one is present: the math emulation will just never get used in that case. The kernel will be slightly larger, but will work on different machines regardless of whether they have a math coprocessor or not.
  - The "kernel hacking" configuration details usually result in a bigger or slower kernel (or both), and can even make the kernel less stable by configuring some routines to actively try to break bad code to find kernel problems (kmalloc()). Thus you should probably answer 'n' to the questions for "development", "experimental", or "debugging" features.

1.8 Compiling the kernel

- Make sure you have at least gcc 5.1 available. For more information, refer to Documentation/process/changes.rst.
- Do a `make` to create a compressed kernel image. It is also possible to do `make install` if you have lilo installed to suit the kernel makefiles, but you may want to check your particular lilo setup first.

  To do the actual install, you have to be root, but none of the normal build should require that. Don't take the name of root in vain.
- If you configured any of the parts of the kernel as `modules`, you will also have to do `make modules_install`.
- Verbose kernel compile/build output:

  Normally, the kernel build system runs in a fairly quiet mode (but not totally silent). However, sometimes you or other kernel developers need to see compile, link, or other commands exactly as they are executed. For this, use "verbose" build mode. This is done by passing `V=1` to the `make` command, e.g.:

  ```
  make V=1 all
  ```

  To have the build system also tell the reason for the rebuild of each target, use `V=2`. The default is `V=0`.
- Keep a backup kernel handy in case something goes wrong. This is especially true for the development releases, since each new release contains new code which has not been debugged. Make sure you keep a backup of the modules corresponding to that kernel, as well. If you are installing a new kernel with the same version number as your working kernel, make a backup of your modules directory before you do a `make modules_install`.

  Alternatively, before compiling, use the kernel config option "LOCALVERSION" to append a unique suffix to the regular kernel version. LOCALVERSION can be set in the "General Setup" menu.
- In order to boot your new kernel, you'll need to copy the kernel image (e.g. .../linux/arch/x86/boot/bzImage after compilation) to the place where your regular bootable kernel is found.
- Booting a kernel directly from a floppy without the assistance of a bootloader such as LILO, is no longer supported.

  If you boot Linux from the hard drive, chances are you use LILO, which uses the kernel image as specified in the file /etc/lilo.conf. The kernel image file is usually /vmlinuz, /boot/vmlinuz, /bzImage or /boot/bzImage. To use the new kernel, save a copy of the old image and copy the new image over the old one. Then, you MUST RERUN LILO to update the loading map! If you don't, you won't be able to boot the new kernel image.

  Reinstalling LILO is usually a matter of running /sbin/lilo. You may wish to edit /etc/lilo.conf to specify an entry for your old kernel image (say, /vmlinux.old) in case the new one does not work. See the LILO docs for more information.

  After reinstalling LILO, you should be all set. Shutdown the system, reboot, and enjoy!

If you ever need to change the default root device, video mode, etc. in the kernel image, use your bootloader's boot options where appropriate. No need to recompile the kernel to change these parameters.

- Reboot with the new kernel and enjoy.

1.9 If something goes wrong

If you have problems that seem to be due to kernel bugs, please follow the instructions at '*Reporting issues*'.

Hints on understanding kernel bug reports are in '*Bug hunting*'. More on debugging the kernel with gdb is in 'Documentation/dev-tools/gdb-kernel-debugging.rst' and 'Documentation/dev-tools/kgdb.rst'.

CHAPTER

TWO

THE KERNEL'S COMMAND-LINE PARAMETERS

The following is a consolidated list of the kernel parameters as implemented by the __setup(), early_param(), core_param() and module_param() macros and sorted into English Dictionary order (defined as ignoring all punctuation and sorting digits before letters in a case insensitive manner), and with descriptions where known.

The kernel parses parameters from the kernel command line up to "--"; if it doesn't recognize a parameter and it doesn't contain a '.', the parameter gets passed to init: parameters with '=' go into init's environment, others are passed as command line arguments to init. Everything after "--" is passed as an argument to init.

Module parameters can be specified in two ways: via the kernel command line with a module name prefix, or via modprobe, e.g.:

```
(kernel command line) usbcore.blinkenlights=1
(modprobe command line) modprobe usbcore blinkenlights=1
```

Parameters for modules which are built into the kernel need to be specified on the kernel command line. modprobe looks through the kernel command line (/proc/cmdline) and collects module parameters when it loads a module, so the kernel command line can be used for loadable modules too.

Hyphens (dashes) and underscores are equivalent in parameter names, so:

```
log_buf_len=1M print-fatal-signals=1
```

can also be entered as:

```
log-buf-len=1M print_fatal_signals=1
```

Double-quotes can be used to protect spaces in values, e.g.:

```
param="spaces in here"
```

2.1 cpu lists:

Some kernel parameters take a list of CPUs as a value, e.g. isolcpus, nohz_full, irqaffinity, rcu_nocbs. The format of this list is:

<cpu number>,...,<cpu number>

or

<cpu number>-<cpu number> (must be a positive range in ascending order)

or a mixture

<cpu number>,...,<cpu number>-<cpu number>

Note that for the special case of a range one can split the range into equal sized groups and for each group use some amount from the beginning of that group:

<cpu number>-<cpu number>:<used size>/<group size>

For example one can add to the command line following parameter:

isolcpus=1,2,10-20,100-2000:2/25

where the final item represents CPUs 100,101,125,126,150,151,...

The value "N" can be used to represent the numerically last CPU on the system, i.e "foo_cpus=16-N" would be equivalent to "16-31" on a 32 core system.

Keep in mind that "N" is dynamic, so if system changes cause the bitmap width to change, such as less cores in the CPU list, then N and any ranges using N will also change. Use the same on a small 4 core system, and "16-N" becomes "16-3" and now the same boot input will be flagged as invalid (start > end).

The special case-tolerant group name "all" has a meaning of selecting all CPUs, so that "nohz_full=all" is the equivalent of "nohz_full=0-N".

The semantics of "N" and "all" is supported on a level of bitmaps and holds for all users of bitmap_parselist().

This document may not be entirely up to date and comprehensive. The command "modinfo -p ${modulename}" shows a current list of all parameters of a loadable module. Loadable modules, after being loaded into the running kernel, also reveal their parameters in /sys/module/${modulename}/parameters/. Some of these parameters may be changed at runtime by the command `echo -n ${value} > /sys/module/${modulename}/parameters/${parm}`.

The parameters listed below are only valid if certain kernel build options were enabled and if respective hardware is present. This list should be kept in alphabetical order. The text in square brackets at the beginning of each description states the restrictions within which a parameter is applicable:

```
ACPI    ACPI support is enabled.
AGP     AGP (Accelerated Graphics Port) is enabled.
ALSA    ALSA sound support is enabled.
APIC    APIC support is enabled.
APM     Advanced Power Management support is enabled.
APPARMOR AppArmor support is enabled.
ARM     ARM architecture is enabled.
```

```
ARM64   ARM64 architecture is enabled.
AX25    Appropriate AX.25 support is enabled.
CLK     Common clock infrastructure is enabled.
CMA     Contiguous Memory Area support is enabled.
DRM     Direct Rendering Management support is enabled.
DYNAMIC_DEBUG Build in debug messages and enable them at runtime
EDD     BIOS Enhanced Disk Drive Services (EDD) is enabled
EFI     EFI Partitioning (GPT) is enabled
EVM     Extended Verification Module
FB      The frame buffer device is enabled.
FTRACE  Function tracing enabled.
GCOV    GCOV profiling is enabled.
HIBERNATION HIBERNATION is enabled.
HW      Appropriate hardware is enabled.
HYPER_V HYPERV support is enabled.
IA-64   IA-64 architecture is enabled.
IMA     Integrity measurement architecture is enabled.
IP_PNP  IP DHCP, BOOTP, or RARP is enabled.
IPV6    IPv6 support is enabled.
ISAPNP  ISA PnP code is enabled.
ISDN    Appropriate ISDN support is enabled.
ISOL    CPU Isolation is enabled.
JOY     Appropriate joystick support is enabled.
KGDB    Kernel debugger support is enabled.
KVM     Kernel Virtual Machine support is enabled.
LIBATA  Libata driver is enabled
LOONGARCH LoongArch architecture is enabled.
LOOP    Loopback device support is enabled.
LP      Printer support is enabled.
M68k    M68k architecture is enabled.
                These options have more detailed description inside of
                Documentation/arch/m68k/kernel-options.rst.
MDA     MDA console support is enabled.
MIPS    MIPS architecture is enabled.
MOUSE   Appropriate mouse support is enabled.
MSI     Message Signaled Interrupts (PCI).
MTD     MTD (Memory Technology Device) support is enabled.
NET     Appropriate network support is enabled.
NFS     Appropriate NFS support is enabled.
NUMA    NUMA support is enabled.
OF      Devicetree is enabled.
PARISC  The PA-RISC architecture is enabled.
PCI     PCI bus support is enabled.
PCIE    PCI Express support is enabled.
PCMCIA  The PCMCIA subsystem is enabled.
PNP     Plug & Play support is enabled.
PPC     PowerPC architecture is enabled.
PPT     Parallel port support is enabled.
PS2     Appropriate PS/2 support is enabled.
PV_OPS  A paravirtualized kernel is enabled.
```

```
RAM     RAM disk support is enabled.
RDT     Intel Resource Director Technology.
RISCV   RISCV architecture is enabled.
S390    S390 architecture is enabled.
SCSI    Appropriate SCSI support is enabled.
                A lot of drivers have their options described inside
                the Documentation/scsi/ sub-directory.
SECURITY Different security models are enabled.
SELINUX SELinux support is enabled.
SERIAL  Serial support is enabled.
SH      SuperH architecture is enabled.
SMP     The kernel is an SMP kernel.
SPARC   Sparc architecture is enabled.
SUSPEND System suspend states are enabled.
SWSUSP  Software suspend (hibernation) is enabled.
TPM     TPM drivers are enabled.
UMS     USB Mass Storage support is enabled.
USB     USB support is enabled.
USBHID  USB Human Interface Device support is enabled.
V4L     Video For Linux support is enabled.
VGA     The VGA console has been enabled.
VMMIO   Driver for memory mapped virtio devices is enabled.
VT      Virtual terminal support is enabled.
WDT     Watchdog support is enabled.
X86-32  X86-32, aka i386 architecture is enabled.
X86-64  X86-64 architecture is enabled.
                More X86-64 boot options can be found in
                Documentation/arch/x86/x86_64/boot-options.rst.
X86     Either 32-bit or 64-bit x86 (same as X86-32+X86-64)
X86_UV  SGI UV support is enabled.
XEN     Xen support is enabled
XTENSA  xtensa architecture is enabled.
```

In addition, the following text indicates that the option:

```
BOOT    Is a boot loader parameter.
BUGS=   Relates to possible processor bugs on the said processor.
KNL     Is a kernel start-up parameter.
```

Parameters denoted with BOOT are actually interpreted by the boot loader, and have no meaning to the kernel directly. Do not modify the syntax of boot loader parameters without extreme need or coordination with <Documentation/arch/x86/boot.rst>.

There are also arch-specific kernel-parameters not documented here. See for example <Documentation/arch/x86/x86_64/boot-options.rst>.

Note that ALL kernel parameters listed below are CASE SENSITIVE, and that a trailing = on the name of any parameter states that that parameter will be entered as an environment variable, whereas its absence indicates that it will appear as a kernel argument readable via /proc/cmdline by programs running once the system is up.

The number of kernel parameters is not limited, but the length of the complete command line

(parameters including spaces etc.) is limited to a fixed number of characters. This limit depends on the architecture and is between 256 and 4096 characters. It is defined in the file ./include/uapi/asm-generic/setup.h as COMMAND_LINE_SIZE.

Finally, the [KMG] suffix is commonly described after a number of kernel parameter values. These 'K', 'M', and 'G' letters represent the _binary_ multipliers 'Kilo', 'Mega', and 'Giga', equaling 2^10, 2^20, and 2^30 bytes respectively. Such letter suffixes can also be entirely omitted:

```
        accept_memory=  [MM]
                        Format: { eager | lazy }
                        default: lazy
                        By default, unaccepted memory is accepted lazily to
                        avoid prolonged boot times. The lazy option will add
                        some runtime overhead until all memory is eventually
                        accepted. In most cases the overhead is negligible.
                        For some workloads or for debugging purposes
                        accept_memory=eager can be used to accept all memory
                        at once during boot.

        acpi=           [HW,ACPI,X86,ARM64,RISCV64]
                        Advanced Configuration and Power Interface
                        Format: { force | on | off | strict | noirq | rsdt |
                                  copy_dsdt }
                        force -- enable ACPI if default was off
                        on -- enable ACPI but allow fallback to DT [arm64,
↪riscv64]
                        off -- disable ACPI if default was on
                        noirq -- do not use ACPI for IRQ routing
                        strict -- Be less tolerant of platforms that are not
                                strictly ACPI specification compliant.
                        rsdt -- prefer RSDT over (default) XSDT
                        copy_dsdt -- copy DSDT to memory
                        For ARM64 and RISCV64, ONLY "acpi=off", "acpi=on" or
                        "acpi=force" are available

                        See also Documentation/power/runtime_pm.rst, pci=noacpi

        acpi_apic_instance=     [ACPI, IOAPIC]
                        Format: <int>
                        2: use 2nd APIC table, if available
                        1,0: use 1st APIC table
                        default: 0

        acpi_backlight= [HW,ACPI]
                        { vendor | video | native | none }
                        If set to vendor, prefer vendor-specific driver
                        (e.g. thinkpad_acpi, sony_acpi, etc.) instead
                        of the ACPI video.ko driver.
                        If set to video, use the ACPI video.ko driver.
                        If set to native, use the device's native backlight␣
↪mode.
                        If set to none, disable the ACPI backlight interface.
```

```
        acpi_force_32bit_fadt_addr
                        force FADT to use 32 bit addresses rather than the
                        64 bit X_* addresses. Some firmware have broken 64
                        bit addresses for force ACPI ignore these and use
                        the older legacy 32 bit addresses.

        acpica_no_return_repair [HW, ACPI]
                        Disable AML predefined validation mechanism
                        This mechanism can repair the evaluation result to make
                        the return objects more ACPI specification compliant.
                        This option is useful for developers to identify the
                        root cause of an AML interpreter issue when the issue
                        has something to do with the repair mechanism.

        acpi.debug_layer=       [HW,ACPI,ACPI_DEBUG]
        acpi.debug_level=       [HW,ACPI,ACPI_DEBUG]
                        Format: <int>
                        CONFIG_ACPI_DEBUG must be enabled to produce any ACPI
                        debug output.  Bits in debug_layer correspond to a
                        _COMPONENT in an ACPI source file, e.g.,
                            #define _COMPONENT ACPI_EVENTS
                        Bits in debug_level correspond to a level in
                        ACPI_DEBUG_PRINT statements, e.g.,
                            ACPI_DEBUG_PRINT((ACPI_DB_INFO, ...
                        The debug_level mask defaults to "info".  See
                        Documentation/firmware-guide/acpi/debug.rst for more␣
↪information about
                        debug layers and levels.

                        Enable processor driver info messages:
                            acpi.debug_layer=0x20000000
                        Enable AML "Debug" output, i.e., stores to the Debug
                        object while interpreting AML:
                            acpi.debug_layer=0xffffffff acpi.debug_level=0x2
                        Enable all messages related to ACPI hardware:
                            acpi.debug_layer=0x2 acpi.debug_level=0xffffffff

                        Some values produce so much output that the system is
                        unusable.  The "log_buf_len" parameter may be useful
                        if you need to capture more output.

        acpi_enforce_resources= [ACPI]
                        { strict | lax | no }
                        Check for resource conflicts between native drivers
                        and ACPI OperationRegions (SystemIO and SystemMemory
                        only). IO ports and memory declared in ACPI might be
                        used by the ACPI subsystem in arbitrary AML code and
                        can interfere with legacy drivers.
                        strict (default): access to resources claimed by ACPI
                        is denied; legacy drivers trying to access reserved
```

```
                resources will fail to bind to device using them.
                lax: access to resources claimed by ACPI is allowed;
                legacy drivers trying to access reserved resources
                will bind successfully but a warning message is logged.
                no: ACPI OperationRegions are not marked as reserved,
                no further checks are performed.

acpi_force_table_verification   [HW,ACPI]
                Enable table checksum verification during early stage.
                By default, this is disabled due to x86 early mapping
                size limitation.

acpi_irq_balance [HW,ACPI]
                ACPI will balance active IRQs
                default in APIC mode

acpi_irq_nobalance [HW,ACPI]
                ACPI will not move active IRQs (default)
                default in PIC mode

acpi_irq_isa=   [HW,ACPI] If irq_balance, mark listed IRQs used by ISA
                Format: <irq>,<irq>...

acpi_irq_pci=   [HW,ACPI] If irq_balance, clear listed IRQs for
                use by PCI
                Format: <irq>,<irq>...

acpi_mask_gpe=  [HW,ACPI]
                Due to the existence of _Lxx/_Exx, some GPEs triggered
                by unsupported hardware/firmware features can result in
                GPE floodings that cannot be automatically disabled by
                the GPE dispatcher.
                This facility can be used to prevent such uncontrolled
                GPE floodings.
                Format: <byte> or <bitmap-list>

acpi_no_auto_serialize  [HW,ACPI]
                Disable auto-serialization of AML methods
                AML control methods that contain the opcodes to create
                named objects will be marked as "Serialized" by the
                auto-serialization feature.
                This feature is enabled by default.
                This option allows to turn off the feature.

acpi_no_memhotplug [ACPI] Disable memory hotplug.  Useful for kdump
                   kernels.

acpi_no_static_ssdt     [HW,ACPI]
                Disable installation of static SSDTs at early boot time
                By default, SSDTs contained in the RSDT/XSDT will be
                installed automatically and they will appear under
```

```
                        /sys/firmware/acpi/tables.
                        This option turns off this feature.
                        Note that specifying this option does not affect
                        dynamic table installation which will install SSDT
                        tables to /sys/firmware/acpi/tables/dynamic.

        acpi_no_watchdog        [HW,ACPI,WDT]
                        Ignore the ACPI-based watchdog interface (WDAT) and let
                        a native driver control the watchdog device instead.

        acpi_rsdp=      [ACPI,EFI,KEXEC]
                        Pass the RSDP address to the kernel, mostly used
                        on machines running EFI runtime service to boot the
                        second kernel for kdump.

        acpi_os_name=   [HW,ACPI] Tell ACPI BIOS the name of the OS
                        Format: To spoof as Windows 98: ="Microsoft Windows"

        acpi_rev_override [ACPI] Override the _REV object to return 5 (instead
                        of 2 which is mandated by ACPI 6) as the supported ACPI
                        specification revision (when using this switch, it may
                        be necessary to carry out a cold reboot _twice_ in a
                        row to make it take effect on the platform firmware).

        acpi_osi=       [HW,ACPI] Modify list of supported OS interface strings
                        acpi_osi="string1"      # add string1
                        acpi_osi="!string2"     # remove string2
                        acpi_osi=!*             # remove all strings
                        acpi_osi=!              # disable all built-in OS␣
↪vendor
                                                  strings
                        acpi_osi=!!             # enable all built-in OS vendor
                                                  strings
                        acpi_osi=               # disable all strings

                        'acpi_osi=!' can be used in combination with single or
                        multiple 'acpi_osi="string1"' to support specific OS
                        vendor string(s).  Note that such command can only
                        affect the default state of the OS vendor strings, thus
                        it cannot affect the default state of the feature group
                        strings and the current state of the OS vendor strings,
                        specifying it multiple times through kernel command␣
↪line
                        is meaningless.  This command is useful when one do not
                        care about the state of the feature group strings which
                        should be controlled by the OSPM.
                        Examples:
                          1. 'acpi_osi=! acpi_osi="Windows 2000"' is equivalent
                             to 'acpi_osi="Windows 2000" acpi_osi=!', they all
                             can make '_OSI("Windows 2000")' TRUE.
```

```
                        'acpi_osi=' cannot be used in combination with other
                        'acpi_osi=' command lines, the _OSI method will not
                        exist in the ACPI namespace.  NOTE that such command␣
↪can
                        only affect the _OSI support state, thus specifying it
                        multiple times through kernel command line is also
                        meaningless.
                        Examples:
                          1. 'acpi_osi=' can make 'CondRefOf(_OSI, Local1)'
                             FALSE.

                        'acpi_osi=!*' can be used in combination with single or
                        multiple 'acpi_osi="string1"' to support specific
                        string(s).  Note that such command can affect the
                        current state of both the OS vendor strings and the
                        feature group strings, thus specifying it multiple␣
↪times
                        through kernel command line is meaningful.  But it may
                        still not able to affect the final state of a string if
                        there are quirks related to this string.  This command
                        is useful when one want to control the state of the
                        feature group strings to debug BIOS issues related to
                        the OSPM features.
                        Examples:
                          1. 'acpi_osi="Module Device" acpi_osi=!*' can make
                             '_OSI("Module Device")' FALSE.
                          2. 'acpi_osi=!* acpi_osi="Module Device"' can make
                             '_OSI("Module Device")' TRUE.
                          3. 'acpi_osi=! acpi_osi=!* acpi_osi="Windows 2000"'␣
↪is
                             equivalent to
                             'acpi_osi=!* acpi_osi=! acpi_osi="Windows 2000"'
                             and
                             'acpi_osi=!* acpi_osi="Windows 2000" acpi_osi=!',
                             they all will make '_OSI("Windows 2000")' TRUE.

        acpi_pm_good    [X86]
                        Override the pmtimer bug detection: force the kernel
                        to assume that this machine's pmtimer latches its value
                        and always returns good values.

        acpi_sci=       [HW,ACPI] ACPI System Control Interrupt trigger mode
                        Format: { level | edge | high | low }

        acpi_skip_timer_override [HW,ACPI]
                        Recognize and ignore IRQ0/pin2 Interrupt Override.
                        For broken nForce2 BIOS resulting in XT-PIC timer.

        acpi_sleep=     [HW,ACPI] Sleep options
                        Format: { s3_bios, s3_mode, s3_beep, s4_hwsig,
                                  s4_nohwsig, old_ordering, nonvs,
```

```
                                sci_force_enable, nobl }
                        See Documentation/power/video.rst for information on
                        s3_bios and s3_mode.
                        s3_beep is for debugging; it makes the PC's speaker␣
↪beep
                        as soon as the kernel's real-mode entry point is␣
↪called.
                        s4_hwsig causes the kernel to check the ACPI hardware
                        signature during resume from hibernation, and␣
↪gracefully
                        refuse to resume if it has changed. This complies with
                        the ACPI specification but not with reality, since
                        Windows does not do this and many laptops do change it
                        on docking. So the default behaviour is to allow resume
                        and simply warn when the signature changes, unless the
                        s4_hwsig option is enabled.
                        s4_nohwsig prevents ACPI hardware signature from being
                        used (or even warned about) during resume.
                        old_ordering causes the ACPI 1.0 ordering of the _PTS
                        control method, with respect to putting devices into
                        low power states, to be enforced (the ACPI 2.0 ordering
                        of _PTS is used by default).
                        nonvs prevents the kernel from saving/restoring the
                        ACPI NVS memory during suspend/hibernation and resume.
                        sci_force_enable causes the kernel to set SCI_EN␣
↪directly
                        on resume from S1/S3 (which is against the ACPI spec,
                        but some broken systems don't work without it).
                        nobl causes the internal blacklist of systems known to
                        behave incorrectly in some ways with respect to system
                        suspend and resume to be ignored (use wisely).

        acpi_use_timer_override [HW,ACPI]
                        Use timer override. For some broken Nvidia NF5 boards
                        that require a timer override, but don't have HPET

        add_efi_memmap  [EFI; X86] Include EFI memory map in
                        kernel's map of available physical RAM.

        agp=            [AGP]
                        { off | try_unsupported }
                        off: disable AGP support
                        try_unsupported: try to drive unsupported chipsets
                                (may crash computer or cause data corruption)

        ALSA            [HW,ALSA]
                        See Documentation/sound/alsa-configuration.rst

        alignment=      [KNL,ARM]
                        Allow the default userspace alignment fault handler
                        behaviour to be specified.  Bit 0 enables warnings,
```

```
                        bit 1 enables fixups, and bit 2 sends a segfault.

        align_va_addr=  [X86-64]
                        Align virtual addresses by clearing slice [14:12] when
                        allocating a VMA at process creation time. This option
                        gives you up to 3% performance improvement on AMD F15h
                        machines (where it is enabled by default) for a
                        CPU-intensive style benchmark, and it can vary highly␣
↪in
                        a microbenchmark depending on workload and compiler.

                        32: only for 32-bit processes
                        64: only for 64-bit processes
                        on: enable for both 32- and 64-bit processes
                        off: disable for both 32- and 64-bit processes

        alloc_snapshot  [FTRACE]
                        Allocate the ftrace snapshot buffer on boot up when the
                        main buffer is allocated. This is handy if debugging
                        and you need to use tracing_snapshot() on boot up, and
                        do not want to use tracing_snapshot_alloc() as it needs
                        to be done where GFP_KERNEL allocations are allowed.

        allow_mismatched_32bit_el0 [ARM64]
                        Allow execve() of 32-bit applications and setting of␣
↪the
                        PER_LINUX32 personality on systems where only a strict
                        subset of the CPUs support 32-bit EL0. When this
                        parameter is present, the set of CPUs supporting 32-bit
                        EL0 is indicated by /sys/devices/system/cpu/aarch32_el0
                        and hot-unplug operations may be restricted.

                        See Documentation/arch/arm64/asymmetric-32bit.rst for␣
↪more
                        information.

        amd_iommu=      [HW,X86-64]
                        Pass parameters to the AMD IOMMU driver in the system.
                        Possible values are:
                        fullflush - Deprecated, equivalent to iommu.strict=1
                        off       - do not initialize any AMD IOMMU found in
                                    the system
                        force_isolation - Force device isolation for all
                                          devices. The IOMMU driver is not
                                          allowed anymore to lift isolation
                                          requirements as needed. This option
                                          does not override iommu=pt
                        force_enable - Force enable the IOMMU on platforms␣
↪known
                                       to be buggy with IOMMU enabled. Use this
                                       option with care.
```

```
                        pgtbl_v1     - Use v1 page table for DMA-API (Default).
                        pgtbl_v2     - Use v2 page table for DMA-API.
                        irtcachedis  - Disable Interrupt Remapping Table (IRT)␣
↪caching.

        amd_iommu_dump= [HW,X86-64]
                        Enable AMD IOMMU driver option to dump the ACPI table
                        for AMD IOMMU. With this option enabled, AMD IOMMU
                        driver will print ACPI tables for AMD IOMMU during
                        IOMMU initialization.

        amd_iommu_intr= [HW,X86-64]
                        Specifies one of the following AMD IOMMU interrupt
                        remapping modes:
                        legacy     - Use legacy interrupt remapping mode.
                        vapic      - Use virtual APIC mode, which allows IOMMU
                                     to inject interrupts directly into guest.
                                     This mode requires kvm-amd.avic=1.
                                     (Default when IOMMU HW support is present.
↪)

        amd_pstate=     [X86]
                        disable
                          Do not enable amd_pstate as the default
                          scaling driver for the supported processors
                        passive
                          Use amd_pstate with passive mode as a scaling driver.
                          In this mode autonomous selection is disabled.
                          Driver requests a desired performance level and␣
↪platform
                          tries to match the same performance level if it is
                          satisfied by guaranteed performance level.
                        active
                          Use amd_pstate_epp driver instance as the scaling␣
↪driver,
                          driver provides a hint to the hardware if software␣
↪wants
                          to bias toward performance (0x0) or energy␣
↪efficiency (0xff)
                          to the CPPC firmware. then CPPC power algorithm will
                          calculate the runtime workload and adjust the␣
↪realtime cores
                          frequency.
                        guided
                          Activate guided autonomous mode. Driver requests␣
↪minimum and
                          maximum performance level and the platform␣
↪autonomously
                          selects a performance level in this range and␣
↪appropriate
                          to the current workload.
```

```
        amijoy.map=     [HW,JOY] Amiga joystick support
                        Map of devices attached to JOY0DAT and JOY1DAT
                        Format: <a>,<b>
                        See also Documentation/input/joydev/joystick.rst

        analog.map=     [HW,JOY] Analog joystick and gamepad support
                        Specifies type or capabilities of an analog joystick
                        connected to one of 16 gameports
                        Format: <type1>,<type2>,..<type16>

        apc=            [HW,SPARC]
                        Power management functions (SPARCstation-4/5 + deriv.)
                        Format: noidle
                        Disable APC CPU standby support. SPARCstation-Fox does
                        not play well with APC CPU idle - disable it if you␣
↪have
                        APC and your system crashes randomly.

        apic=           [APIC,X86] Advanced Programmable Interrupt Controller
                        Change the output verbosity while booting
                        Format: { quiet (default) | verbose | debug }
                        Change the amount of debugging information output
                        when initialising the APIC and IO-APIC components.
                        For X86-32, this can also be used to specify an APIC
                        driver name.
                        Format: apic=driver_name
                        Examples: apic=bigsmp

        apic_extnmi=    [APIC,X86] External NMI delivery setting
                        Format: { bsp (default) | all | none }
                        bsp:  External NMI is delivered only to CPU 0
                        all:  External NMIs are broadcast to all CPUs as a
                              backup of CPU 0
                        none: External NMI is masked for all CPUs. This is
                              useful so that a dump capture kernel won't be
                              shot down by NMI

        autoconf=       [IPV6]
                        See Documentation/networking/ipv6.rst.

        apm=            [APM] Advanced Power Management
                        See header of arch/x86/kernel/apm_32.c.

        apparmor=       [APPARMOR] Disable or enable AppArmor at boot time
                        Format: { "0" | "1" }
                        See security/apparmor/Kconfig help text
                        0 -- disable.
                        1 -- enable.
                        Default value is set via kernel config option.
```

```
        arcrimi=        [HW,NET] ARCnet - "RIM I" (entirely mem-mapped) cards
                        Format: <io>,<irq>,<nodeID>

        arm64.nobti     [ARM64] Unconditionally disable Branch Target
                        Identification support

        arm64.nomops    [ARM64] Unconditionally disable Memory Copy and Memory
                        Set instructions support

        arm64.nomte     [ARM64] Unconditionally disable Memory Tagging␣
↪Extension
                        support

        arm64.nopauth   [ARM64] Unconditionally disable Pointer Authentication
                        support

        arm64.nosme     [ARM64] Unconditionally disable Scalable Matrix
                        Extension support

        arm64.nosve     [ARM64] Unconditionally disable Scalable Vector
                        Extension support

        ataflop=        [HW,M68k]

        atarimouse=     [HW,MOUSE] Atari Mouse

        atkbd.extra=    [HW] Enable extra LEDs and keys on IBM RapidAccess,
                        EzKey and similar keyboards

        atkbd.reset=    [HW] Reset keyboard during initialization

        atkbd.set=      [HW] Select keyboard code set
                        Format: <int> (2 = AT (default), 3 = PS/2)

        atkbd.scroll=   [HW] Enable scroll wheel on MS Office and similar
                        keyboards

        atkbd.softraw=  [HW] Choose between synthetic and real raw mode
                        Format: <bool> (0 = real, 1 = synthetic (default))

        atkbd.softrepeat= [HW]
                        Use software keyboard repeat

        audit=          [KNL] Enable the audit sub-system
                        Format: { "0" | "1" | "off" | "on" }
                        0 | off - kernel audit is disabled and can not be
                            enabled until the next reboot
                        unset - kernel audit is initialized but disabled and
                            will be fully enabled by the userspace auditd.
                        1 | on - kernel audit is initialized and partially
                            enabled, storing at most audit_backlog_limit
```

```
                    messages in RAM until it is fully enabled by the
                    userspace auditd.
                Default: unset

audit_backlog_limit= [KNL] Set the audit queue size limit.
                Format: <int> (must be >=0)
                Default: 64

bau=            [X86_UV] Enable the BAU on SGI UV.  The default
                behavior is to disable the BAU (i.e. bau=0).
                Format: { "0" | "1" }
                0 - Disable the BAU.
                1 - Enable the BAU.
                unset - Disable the BAU.

baycom_epp=     [HW,AX25]
                Format: <io>,<mode>

baycom_par=     [HW,AX25] BayCom Parallel Port AX.25 Modem
                Format: <io>,<mode>
                See header of drivers/net/hamradio/baycom_par.c.

baycom_ser_fdx= [HW,AX25]
                BayCom Serial Port AX.25 Modem (Full Duplex Mode)
                Format: <io>,<irq>,<mode>[,<baud>]
                See header of drivers/net/hamradio/baycom_ser_fdx.c.

baycom_ser_hdx= [HW,AX25]
                BayCom Serial Port AX.25 Modem (Half Duplex Mode)
                Format: <io>,<irq>,<mode>
                See header of drivers/net/hamradio/baycom_ser_hdx.c.

bert_disable    [ACPI]
                Disable BERT OS support on buggy BIOSes.

bgrt_disable    [ACPI][X86]
                Disable BGRT to avoid flickering OEM logo.

blkdevparts=    Manual partition parsing of block device(s) for
                embedded devices based on command line input.
                See Documentation/block/cmdline-partition.rst

boot_delay=     Milliseconds to delay each printk during boot.
                Only works if CONFIG_BOOT_PRINTK_DELAY is enabled,
                and you may also have to specify "lpj=".  Boot_delay
                values larger than 10 seconds (10000) are assumed
                erroneous and ignored.
                Format: integer

bootconfig      [KNL]
                Extended command line options can be added to an initrd
```

```
                        and this will cause the kernel to look for it.

                        See Documentation/admin-guide/bootconfig.rst

        bttv.card=      [HW,V4L] bttv (bt848 + bt878 based grabber cards)
        bttv.radio=     Most important insmod options are available as
                        kernel args too.
        bttv.pll=       See Documentation/admin-guide/media/bttv.rst
        bttv.tuner=

        bulk_remove=off [PPC]  This parameter disables the use of the pSeries
                        firmware feature for flushing multiple hpte entries
                        at a time.

        c101=           [NET] Moxa C101 synchronous serial card

        cachesize=      [BUGS=X86-32] Override level 2 CPU cache size␣
↪detection.
                        Sometimes CPU hardware bugs make them report the cache
                        size incorrectly. The kernel will attempt work arounds
                        to fix known problems, but for some CPUs it is not
                        possible to determine what the correct size should be.
                        This option provides an override for these situations.

        carrier_timeout=
                        [NET] Specifies amount of time (in seconds) that
                        the kernel should wait for a network carrier. By␣
↪default
                        it waits 120 seconds.

        ca_keys=        [KEYS] This parameter identifies a specific key(s) on
                        the system trusted keyring to be used for certificate
                        trust validation.
                        format: { id:<keyid> | builtin }

        cca=            [MIPS] Override the kernel pages' cache coherency
                        algorithm.  Accepted values range from 0 to 7
                        inclusive. See arch/mips/include/asm/pgtable-bits.h
                        for platform specific values (SB1, Loongson3 and
                        others).

        ccw_timeout_log [S390]
                        See Documentation/arch/s390/common_io.rst for details.

        cgroup_disable= [KNL] Disable a particular controller or optional␣
↪feature
                        Format: {name of the controller(s) or feature(s) to␣
↪disable}
                        The effects of cgroup_disable=foo are:
                        - foo isn't auto-mounted if you mount all cgroups in
                          a single hierarchy
```

```
                        - foo isn't visible as an individually mountable
                          subsystem
                        - if foo is an optional feature then the feature is
                          disabled and corresponding cgroup files are not
                          created
                        {Currently only "memory" controller deal with this and
                        cut the overhead, others just disable the usage. So
                        only cgroup_disable=memory is actually worthy}
                        Specifying "pressure" disables per-cgroup pressure
                        stall information accounting feature

        cgroup_no_v1=   [KNL] Disable cgroup controllers and named hierarchies␣
↪in v1
                        Format: { { controller | "all" | "named" }
                                  [,{ controller | "all" | "named" }...] }
                        Like cgroup_disable, but only applies to cgroup v1;
                        the blacklisted controllers remain available in␣
↪cgroup2.
                        "all" blacklists all controllers and "named" disables
                        named mounts. Specifying both "all" and "named"␣
↪disables
                        all v1 hierarchies.

        cgroup_favordynmods= [KNL] Enable or Disable favordynmods.
                        Format: { "true" | "false" }
                        Defaults to the value of CONFIG_CGROUP_FAVOR_DYNMODS.

        cgroup.memory=  [KNL] Pass options to the cgroup memory controller.
                        Format: <string>
                        nosocket -- Disable socket memory accounting.
                        nokmem -- Disable kernel memory accounting.
                        nobpf -- Disable BPF memory accounting.

        checkreqprot=   [SELINUX] Set initial checkreqprot flag value.
                        Format: { "0" | "1" }
                        See security/selinux/Kconfig help text.
                        0 -- check protection applied by kernel (includes
                                any implied execute protection).
                        1 -- check protection requested by application.
                        Default value is set via a kernel config option.
                        Value can be changed at runtime via
                                /sys/fs/selinux/checkreqprot.
                        Setting checkreqprot to 1 is deprecated.

        cio_ignore=     [S390]
                        See Documentation/arch/s390/common_io.rst for details.

        clearcpuid=X[,X...] [X86]
                        Disable CPUID feature X for the kernel. See
                        arch/x86/include/asm/cpufeatures.h for the valid bit
                        numbers X. Note the Linux-specific bits are not␣
```

```
↪necessarily
                        stable over kernel options, but the vendor-specific
                        ones should be.
                        X can also be a string as appearing in the flags: line
                        in /proc/cpuinfo which does not have the above
                        instability issue. However, not all features have names
                        in /proc/cpuinfo.
                        Note that using this option will taint your kernel.
                        Also note that user programs calling CPUID directly
                        or using the feature without checking anything
                        will still see it. This just prevents it from
                        being used by the kernel or shown in /proc/cpuinfo.
                        Also note the kernel might malfunction if you disable
                        some critical bits.

        clk_ignore_unused
                        [CLK]
                        Prevents the clock framework from automatically gating
                        clocks that have not been explicitly enabled by a Linux
                        device driver but are enabled in hardware at reset or
                        by the bootloader/firmware. Note that this does not
                        force such clocks to be always-on nor does it reserve
                        those clocks in any way. This parameter is useful for
                        debug and development, but should not be needed on a
                        platform with proper driver support.  For more
                        information, see Documentation/driver-api/clk.rst.

        clock=          [BUGS=X86-32, HW] gettimeofday clocksource override.
                        [Deprecated]
                        Forces specified clocksource (if available) to be used
                        when calculating gettimeofday(). If specified
                        clocksource is not available, it defaults to PIT.
                        Format: { pit | tsc | cyclone | pmtmr }

        clocksource=    Override the default clocksource
                        Format: <string>
                        Override the default clocksource and use the␣
↪clocksource
                        with the name specified.
                        Some clocksource names to choose from, depending on
                        the platform:
                        [all] jiffies (this is the base, fallback clocksource)
                        [ACPI] acpi_pm
                        [ARM] imx_timer1,OSTS,netx_timer,mpu_timer2,
                                pxa_timer,timer3,32k_counter,timer0_1
                        [X86-32] pit,hpet,tsc;
                                scx200_hrt on Geode; cyclone on IBM x440
                        [MIPS] MIPS
                        [PARISC] cr16
                        [S390] tod
                        [SH] SuperH
```

```
                [SPARC64] tick
                [X86-64] hpet,tsc

clocksource.arm_arch_timer.evtstrm=
                [ARM,ARM64]
                Format: <bool>
                Enable/disable the eventstream feature of the ARM
                architected timer so that code using WFE-based polling
                loops can be debugged more effectively on production
                systems.

clocksource.max_cswd_read_retries= [KNL]
                Number of clocksource_watchdog() retries due to
                external delays before the clock will be marked
                unstable.  Defaults to two retries, that is,
                three attempts to read the clock under test.

clocksource.verify_n_cpus= [KNL]
                Limit the number of CPUs checked for clocksources
                marked with CLOCK_SOURCE_VERIFY_PERCPU that
                are marked unstable due to excessive skew.
                A negative value says to check all CPUs, while
                zero says not to check any.  Values larger than
                nr_cpu_ids are silently truncated to nr_cpu_ids.
                The actual CPUs are chosen randomly, with
                no replacement if the same CPU is chosen twice.

clocksource-wdtest.holdoff= [KNL]
                Set the time in seconds that the clocksource
                watchdog test waits before commencing its tests.
                Defaults to zero when built as a module and to
                10 seconds when built into the kernel.

cma=nn[MG]@[start[MG][-end[MG]]]
                [KNL,CMA]
                Sets the size of kernel global memory area for
                contiguous memory allocations and optionally the
                placement constraint by the physical address range of
                memory allocations. A value of 0 disables CMA
                altogether. For more information, see
                kernel/dma/contiguous.c

cma_pernuma=nn[MG]
                [KNL,CMA]
                Sets the size of kernel per-numa memory area for
                contiguous memory allocations. A value of 0 disables
                per-numa CMA altogether. And If this option is not
                specified, the default value is 0.
                With per-numa CMA enabled, DMA users on node nid will
                first try to allocate buffer from the pernuma area
                which is located in node nid, if the allocation fails,
```

```
                        they will fallback to the global default memory area.

        numa_cma=<node>:nn[MG][,<node>:nn[MG]]
                        [KNL,CMA]
                        Sets the size of kernel numa memory area for
                        contiguous memory allocations. It will reserve CMA
                        area for the specified node.

                        With numa CMA enabled, DMA users on node nid will
                        first try to allocate buffer from the numa area
                        which is located in node nid, if the allocation fails,
                        they will fallback to the global default memory area.

        cmo_free_hint=  [PPC] Format: { yes | no }
                        Specify whether pages are marked as being inactive
                        when they are freed.  This is used in CMO environments
                        to determine OS memory pressure for page stealing by
                        a hypervisor.
                        Default: yes

        coherent_pool=nn[KMG]   [ARM,KNL]
                        Sets the size of memory pool for coherent, atomic dma
                        allocations, by default set to 256K.

        com20020=       [HW,NET] ARCnet - COM20020 chipset
                        Format:
                        <io>[,<irq>[,<nodeID>[,<backplane>[,<ckp>[,<timeout>
↪]]]]]

        com90io=        [HW,NET] ARCnet - COM90xx chipset (IO-mapped buffers)
                        Format: <io>[,<irq>]

        com90xx=        [HW,NET]
                        ARCnet - COM90xx chipset (memory-mapped buffers)
                        Format: <io>[,<irq>[,<memstart>]]

        condev=         [HW,S390] console device
        conmode=

        con3215_drop=   [S390] 3215 console drop mode.
                        Format: y|n|Y|N|1|0
                        When set to true, drop data on the 3215 console when
                        the console buffer is full. In this case the
                        operator using a 3270 terminal emulator (for example
                        x3270) does not have to enter the clear key for the
                        console output to advance and the kernel to continue.
                        This leads to a much faster boot time when a 3270
                        terminal emulator is active. If no 3270 terminal
                        emulator is used, this parameter has no effect.

        console=        [KNL] Output console device and options.
```

```
                tty<n>  Use the virtual console device <n>.

                ttyS<n>[,options]
                ttyUSB0[,options]
                        Use the specified serial port.  The options are of
                        the form "bbbbpnf", where "bbbb" is the baud rate,
                        "p" is parity ("n", "o", or "e"), "n" is number of
                        bits, and "f" is flow control ("r" for RTS or
                        omit it).  Default is "9600n8".

                        See Documentation/admin-guide/serial-console.rst for␣
↪more
                        information.  See
                        Documentation/networking/netconsole.rst for an
                        alternative.

                uart[8250],io,<addr>[,options]
                uart[8250],mmio,<addr>[,options]
                uart[8250],mmio16,<addr>[,options]
                uart[8250],mmio32,<addr>[,options]
                uart[8250],0x<addr>[,options]
                        Start an early, polled-mode console on the 8250/16550
                        UART at the specified I/O port or MMIO address,
                        switching to the matching ttyS device later.
                        MMIO inter-register address stride is either 8-bit
                        (mmio), 16-bit (mmio16), or 32-bit (mmio32).
                        If none of [io|mmio|mmio16|mmio32], <addr> is assumed
                        to be equivalent to 'mmio'. 'options' are specified in
                        the same format described for ttyS above; if␣
↪unspecified,
                        the h/w is not re-initialized.

                hvc<n>  Use the hypervisor console device <n>. This is for
                        both Xen and PowerPC hypervisors.

                { null | "" }
                        Use to disable console output, i.e., to have kernel
                        console messages discarded.
                        This must be the only console= parameter used on the
                        kernel command line.

                If the device connected to the port is not a TTY but a braille
                device, prepend "brl," before the device type, for instance
                        console=brl,ttyS0
                For now, only VisioBraille is supported.

        console_msg_format=
                        [KNL] Change console messages format
                default
                        By default we print messages on consoles in
```

```
                        "[time stamp] text\n" format (time stamp may not be
                        printed, depending on CONFIG_PRINTK_TIME or
                        `printk_time' param).
                syslog
                        Switch to syslog format: "<%u>[time stamp] text\n"
                        IOW, each message will have a facility and loglevel
                        prefix. The format is similar to one used by syslog()
                        syscall, or to executing "dmesg -S --raw" or to reading
                        from /proc/kmsg.

        consoleblank=   [KNL] The console blank (screen saver) timeout in
                        seconds. A value of 0 disables the blank timer.
                        Defaults to 0.

        coredump_filter=
                        [KNL] Change the default value for
                        /proc/<pid>/coredump_filter.
                        See also Documentation/filesystems/proc.rst.

        coresight_cpu_debug.enable
                        [ARM,ARM64]
                        Format: <bool>
                        Enable/disable the CPU sampling based debugging.
                        0: default value, disable debugging
                        1: enable debugging at boot time

        cpcihp_generic= [HW,PCI] Generic port I/O CompactPCI driver
                        Format:
                        <first_slot>,<last_slot>,<port>,<enum_bit>[,<debug>]

        cpuidle.off=1   [CPU_IDLE]
                        disable the cpuidle sub-system

        cpuidle.governor=
                        [CPU_IDLE] Name of the cpuidle governor to use.

        cpufreq.off=1   [CPU_FREQ]
                        disable the cpufreq sub-system

        cpufreq.default_governor=
                        [CPU_FREQ] Name of the default cpufreq governor or
                        policy to use. This governor must be registered in the
                        kernel before the cpufreq driver probes.

        cpu_init_udelay=N
                        [X86] Delay for N microsec between assert and de-assert
                        of APIC INIT to start processors.  This delay occurs
                        on every CPU online, such as boot, and resume from␣
↪suspend.
                        Default: 10000
```

```
        cpuhp.parallel=
                        [SMP] Enable/disable parallel bringup of secondary CPUs
                        Format: <bool>
                        Default is enabled if CONFIG_HOTPLUG_PARALLEL=y.␣
↪Otherwise
                        the parameter has no effect.

        crash_kexec_post_notifiers
                        Run kdump after running panic-notifiers and dumping
                        kmsg. This only for the users who doubt kdump always
                        succeeds in any situation.
                        Note that this also increases risks of kdump failure,
                        because some panic notifiers can make the crashed
                        kernel more unstable.

        crashkernel=size[KMG][@offset[KMG]]
                        [KNL] Using kexec, Linux can switch to a 'crash kernel'
                        upon panic. This parameter reserves the physical
                        memory region [offset, offset + size] for that kernel
                        image. If '@offset' is omitted, then a suitable offset
                        is selected automatically.
                        [KNL, X86-64, ARM64, RISCV, LoongArch] Select a region
                        under 4G first, and fall back to reserve region above
                        4G when '@offset' hasn't been specified.
                        See Documentation/admin-guide/kdump/kdump.rst for␣
↪further details.

        crashkernel=range1:size1[,range2:size2,...][@offset]
                        [KNL] Same as above, but depends on the memory
                        in the running system. The syntax of range is
                        start-[end] where start and end are both
                        a memory unit (amount[KMG]). See also
                        Documentation/admin-guide/kdump/kdump.rst for an␣
↪example.

        crashkernel=size[KMG],high
                        [KNL, X86-64, ARM64, RISCV, LoongArch] range could be
                        above 4G.
                        Allow kernel to allocate physical memory region from␣
↪top,
                        so could be above 4G if system have more than 4G ram
                        installed. Otherwise memory region will be allocated
                        below 4G, if available.
                        It will be ignored if crashkernel=X is specified.
        crashkernel=size[KMG],low
                        [KNL, X86-64, ARM64, RISCV, LoongArch] range under 4G.
                        When crashkernel=X,high is passed, kernel could␣
↪allocate
                        physical memory region above 4G, that cause second␣
↪kernel
                        crash on system that require some amount of low memory,
```

```
                        e.g. swiotlb requires at least 64M+32K low memory, also
                        enough extra low memory is needed to make sure DMA␣
↪buffers
                        for 32-bit devices won't run out. Kernel would try to␣
↪allocate
                        default size of memory below 4G automatically. The␣
↪default
                        size is platform dependent.
                          --> x86: max(swiotlb_size_or_default() + 8MiB,␣
↪256MiB)
                          --> arm64: 128MiB
                          --> riscv: 128MiB
                          --> loongarch: 128MiB
                        This one lets the user specify own low range under 4G
                        for second kernel instead.
                        0: to disable low allocation.
                        It will be ignored when crashkernel=X,high is not used
                        or memory reserved is below 4G.

        cryptomgr.notests
                        [KNL] Disable crypto self-tests

        cs89x0_dma=     [HW,NET]
                        Format: <dma>

        cs89x0_media=   [HW,NET]
                        Format: { rj45 | aui | bnc }

        csdlock_debug=  [KNL] Enable or disable debug add-ons of cross-CPU
                        function call handling. When switched on,
                        additional debug data is printed to the console
                        in case a hanging CPU is detected, and that
                        CPU is pinged again in order to try to resolve
                        the hang situation.  The default value of this
                        option depends on the CSD_LOCK_WAIT_DEBUG_DEFAULT
                        Kconfig option.

        dasd=           [HW,NET]
                        See header of drivers/s390/block/dasd_devmap.c.

        db9.dev[2|3]=   [HW,JOY] Multisystem joystick support via parallel port
                        (one device per port)
                        Format: <port#>,<type>
                        See also Documentation/input/devices/joystick-parport.
↪rst

        debug           [KNL] Enable kernel debugging (events log level).

        debug_boot_weak_hash
                        [KNL] Enable printing [hashed] pointers early in the
                        boot sequence.  If enabled, we use a weak hash instead
```

```
                        of siphash to hash pointers.  Use this option if you␣
↪are
                        seeing instances of '(___ptrval___)') and need to see a
                        value (hashed pointer) instead. Cryptographically
                        insecure, please do not use on production kernels.

        debug_locks_verbose=
                        [KNL] verbose locking self-tests
                        Format: <int>
                        Print debugging info while doing the locking API
                        self-tests.
                        Bitmask for the various LOCKTYPE_ tests. Defaults to 0
                        (no extra messages), setting it to -1 (all bits set)
                        will print _a_lot_ more information - normally only
                        useful to lockdep developers.

        debug_objects   [KNL] Enable object debugging

        debug_guardpage_minorder=
                        [KNL] When CONFIG_DEBUG_PAGEALLOC is set, this
                        parameter allows control of the order of pages that␣
↪will
                        be intentionally kept free (and hence protected) by the
                        buddy allocator. Bigger value increase the probability
                        of catching random memory corruption, but reduce the
                        amount of memory for normal system use. The maximum
                        possible value is MAX_PAGE_ORDER/2.  Setting this
                        parameter to 1 or 2 should be enough to identify most
                        random memory corruption problems caused by bugs in
                        kernel or driver code when a CPU writes to (or reads
                        from) a random memory location. Note that there exists
                        a class of memory corruptions problems caused by buggy
                        H/W or F/W or by drivers badly programming DMA
                        (basically when memory is written at bus level and the
                        CPU MMU is bypassed) which are not detectable by
                        CONFIG_DEBUG_PAGEALLOC, hence this option will not
                        help tracking down these problems.

        debug_pagealloc=
                        [KNL] When CONFIG_DEBUG_PAGEALLOC is set, this␣
↪parameter
                        enables the feature at boot time. By default, it is
                        disabled and the system will work mostly the same as a
                        kernel built without CONFIG_DEBUG_PAGEALLOC.
                        Note: to get most of debug_pagealloc error reports,␣
↪it's
                        useful to also enable the page_owner functionality.
                        on: enable the feature

        debugfs=        [KNL] This parameter enables what is exposed to␣
↪userspace
```

```
                        and debugfs internal clients.
                        Format: { on, no-mount, off }
                        on:     All functions are enabled.
                        no-mount:
                                Filesystem is not registered but kernel␣
↪clients can
                                access APIs and a crashkernel can be used to␣
↪read
                                its content. There is nothing to mount.
                        off:    Filesystem is not registered and clients
                                get a -EPERM as result when trying to register␣
↪files
                                or directories within debugfs.
                                This is equivalent of the runtime␣
↪functionality if
                                debugfs was not enabled in the kernel at all.
                        Default value is set in build-time with a kernel␣
↪configuration.

        debugpat        [X86] Enable PAT debugging

        default_hugepagesz=
                        [HW] The size of the default HugeTLB page. This is
                        the size represented by the legacy /proc/ hugepages
                        APIs.  In addition, this is the default hugetlb size
                        used for shmget(), mmap() and mounting hugetlbfs
                        filesystems.  If not specified, defaults to the
                        architecture's default huge page size.  Huge page
                        sizes are architecture dependent.  See also
                        Documentation/admin-guide/mm/hugetlbpage.rst.
                        Format: size[KMG]

        deferred_probe_timeout=
                        [KNL] Debugging option to set a timeout in seconds for
                        deferred probe to give up waiting on dependencies to
                        probe. Only specific dependencies (subsystems or
                        drivers) that have opted in will be ignored. A timeout
                        of 0 will timeout at the end of initcalls. If the time
                        out hasn't expired, it'll be restarted by each
                        successful driver registration. This option will also
                        dump out devices still on the deferred probe list after
                        retrying.

        delayacct       [KNL] Enable per-task delay accounting

        dell_smm_hwmon.ignore_dmi=
                        [HW] Continue probing hardware even if DMI data
                        indicates that the driver is running on unsupported
                        hardware.

        dell_smm_hwmon.force=
```

```
                        [HW] Activate driver even if SMM BIOS signature does
                        not match list of supported models and enable otherwise
                        blacklisted features.

        dell_smm_hwmon.power_status=
                        [HW] Report power status in /proc/i8k
                        (disabled by default).

        dell_smm_hwmon.restricted=
                        [HW] Allow controlling fans only if SYS_ADMIN
                        capability is set.

        dell_smm_hwmon.fan_mult=
                        [HW] Factor to multiply fan speed with.

        dell_smm_hwmon.fan_max=
                        [HW] Maximum configurable fan speed.

        dfltcc=         [HW,S390]
                        Format: { on | off | def_only | inf_only | always }
                        on:       s390 zlib hardware support for compression on
                                  level 1 and decompression (default)
                        off:      No s390 zlib hardware support
                        def_only: s390 zlib hardware support for deflate
                                  only (compression on level 1)
                        inf_only: s390 zlib hardware support for inflate
                                  only (decompression)
                        always:   Same as 'on' but ignores the selected␣
↪compression
                                  level always using hardware support (used␣
↪for debugging)

        dhash_entries=  [KNL]
                        Set number of hash buckets for dentry cache.

        disable_1tb_segments [PPC]
                        Disables the use of 1TB hash page table segments. This
                        causes the kernel to fall back to 256MB segments which
                        can be useful when debugging issues that require an SLB
                        miss to occur.

        disable=        [IPV6]
                        See Documentation/networking/ipv6.rst.

        disable_radix   [PPC]
                        Disable RADIX MMU mode on POWER9

        disable_tlbie   [PPC]
                        Disable TLBIE instruction. Currently does not work
                        with KVM, with HASH MMU, or with coherent accelerators.
```

```
disable_cpu_apicid= [X86,APIC,SMP]
                Format: <int>
                The number of initial APIC ID for the
                corresponding CPU to be disabled at boot,
                mostly used for the kdump 2nd kernel to
                disable BSP to wake up multiple CPUs without
                causing system reset or hang due to sending
                INIT from AP to BSP.

disable_ddw     [PPC/PSERIES]
                Disable Dynamic DMA Window support. Use this
                to workaround buggy firmware.

disable_ipv6=   [IPV6]
                See Documentation/networking/ipv6.rst.

disable_mtrr_cleanup [X86]
                The kernel tries to adjust MTRR layout from continuous
                to discrete, to make X server driver able to add WB
                entry later. This parameter disables that.

disable_mtrr_trim [X86, Intel and AMD only]
                By default the kernel will trim any uncacheable
                memory out of your available memory pool based on
                MTRR settings.  This parameter disables that behavior,
                possibly causing your machine to run very slowly.

disable_timer_pin_1 [X86]
                Disable PIN 1 of APIC timer
                Can be useful to work around chipset bugs.

dis_ucode_ldr   [X86] Disable the microcode loader.

dma_debug=off   If the kernel is compiled with DMA_API_DEBUG support,
                this option disables the debugging code at boot.

dma_debug_entries=<number>
                This option allows to tune the number of preallocated
                entries for DMA-API debugging code. One entry is
                required per DMA-API allocation. Use this if the
                DMA-API debugging code disables itself because the
                architectural default is too low.

dma_debug_driver=<driver_name>
                With this option the DMA-API debugging driver
                filter feature can be enabled at boot time. Just
                pass the driver to filter for as the parameter.
                The filter can be disabled or changed to another
                driver later using sysfs.

driver_async_probe=  [KNL]
```

```
                        List of driver names to be probed asynchronously. *
                        matches with all driver names. If * is specified, the
                        rest of the listed driver names are those that will NOT
                        match the *.
                        Format: <driver_name1>,<driver_name2>...

        drm.edid_firmware=[<connector>:]<file>[,[<connector>:]<file>]
                        Broken monitors, graphic adapters, KVMs and EDIDless
                        panels may send no or incorrect EDID data sets.
                        This parameter allows to specify an EDID data sets
                        in the /lib/firmware directory that are used instead.
                        Generic built-in EDID data sets are used, if one of
                        edid/1024x768.bin, edid/1280x1024.bin,
                        edid/1680x1050.bin, or edid/1920x1080.bin is given
                        and no file with the same name exists. Details and
                        instructions how to build your own EDID data are
                        available in Documentation/admin-guide/edid.rst. An␣
↪EDID
                        data set will only be used for a particular connector,
                        if its name and a colon are prepended to the EDID
                        name. Each connector may use a unique EDID data
                        set by separating the files with a comma.  An EDID
                        data set with no connector name will be used for
                        any connectors not explicitly specified.

        dscc4.setup=    [NET]

        dt_cpu_ftrs=    [PPC]
                        Format: {"off" | "known"}
                        Control how the dt_cpu_ftrs device-tree binding is
                        used for CPU feature discovery and setup (if it
                        exists).
                        off: Do not use it, fall back to legacy cpu table.
                        known: Do not pass through unknown features to guests
                        or userspace, only those that the kernel is aware of.

        dump_apple_properties   [X86]
                        Dump name and content of EFI device properties on
                        x86 Macs.  Useful for driver authors to determine
                        what data is available or for reverse-engineering.

        dyndbg[="val"]          [KNL,DYNAMIC_DEBUG]
        <module>.dyndbg[="val"]
                        Enable debug messages at boot time.  See
                        Documentation/admin-guide/dynamic-debug-howto.rst
                        for details.

        early_ioremap_debug [KNL]
                        Enable debug messages in early_ioremap support. This
                        is useful for tracking down temporary early mappings
                        which are not unmapped.
```

```
earlycon=       [KNL] Output early console device and options.

                When used with no options, the early console is
                determined by stdout-path property in device tree's
                chosen node or the ACPI SPCR table if supported by
                the platform.

        cdns,<addr>[,options]
                Start an early, polled-mode console on a Cadence
                (xuartps) serial port at the specified address. Only
                supported option is baud rate. If baud rate is not
                specified, the serial port must already be setup and
                configured.

        uart[8250],io,<addr>[,options[,uartclk]]
        uart[8250],mmio,<addr>[,options[,uartclk]]
        uart[8250],mmio32,<addr>[,options[,uartclk]]
        uart[8250],mmio32be,<addr>[,options[,uartclk]]
        uart[8250],0x<addr>[,options]
                Start an early, polled-mode console on the 8250/16550
                UART at the specified I/O port or MMIO address.
                MMIO inter-register address stride is either 8-bit
                (mmio) or 32-bit (mmio32 or mmio32be).
                If none of [io|mmio|mmio32|mmio32be], <addr> is assumed
                to be equivalent to 'mmio'. 'options' are specified
                in the same format described for "console=ttyS<n>"; if
                unspecified, the h/w is not initialized. 'uartclk' is
                the uart clock frequency; if unspecified, it is set
                to 'BASE_BAUD' * 16.

        pl011,<addr>
        pl011,mmio32,<addr>
                Start an early, polled-mode console on a pl011 serial
                port at the specified address. The pl011 serial port
                must already be setup and configured. Options are not
                yet supported.  If 'mmio32' is specified, then only
                the driver will use only 32-bit accessors to read/write
                the device registers.

        liteuart,<addr>
                Start an early console on a litex serial port at the
                specified address. The serial port must already be
                setup and configured. Options are not yet supported.

        meson,<addr>
                Start an early, polled-mode console on a meson serial
                port at the specified address. The serial port must
                already be setup and configured. Options are not yet
                supported.
```

```
msm_serial,<addr>
        Start an early, polled-mode console on an msm serial
        port at the specified address. The serial port
        must already be setup and configured. Options are not
        yet supported.

msm_serial_dm,<addr>
        Start an early, polled-mode console on an msm serial
        dm port at the specified address. The serial port
        must already be setup and configured. Options are not
        yet supported.

owl,<addr>
        Start an early, polled-mode console on a serial port
        of an Actions Semi SoC, such as S500 or S900, at the
        specified address. The serial port must already be
        setup and configured. Options are not yet supported.

rda,<addr>
        Start an early, polled-mode console on a serial port
        of an RDA Micro SoC, such as RDA8810PL, at the
        specified address. The serial port must already be
        setup and configured. Options are not yet supported.

sbi
        Use RISC-V SBI (Supervisor Binary Interface) for early
        console.

smh     Use ARM semihosting calls for early console.

s3c2410,<addr>
s3c2412,<addr>
s3c2440,<addr>
s3c6400,<addr>
s5pv210,<addr>
exynos4210,<addr>
        Use early console provided by serial driver available
        on Samsung SoCs, requires selecting proper type and
        a correct base address of the selected UART port. The
        serial port must already be setup and configured.
        Options are not yet supported.

lantiq,<addr>
        Start an early, polled-mode console on a lantiq serial
        (lqasc) port at the specified address. The serial port
        must already be setup and configured. Options are not
        yet supported.

lpuart,<addr>
lpuart32,<addr>
        Use early console provided by Freescale LP UART driver
```

```
                        found on Freescale Vybrid and QorIQ LS1021A processors.
                        A valid base address must be provided, and the serial
                        port must already be setup and configured.

                ec_imx21,<addr>
                ec_imx6q,<addr>
                        Start an early, polled-mode, output-only console on the
                        Freescale i.MX UART at the specified address. The UART
                        must already be setup and configured.

                ar3700_uart,<addr>
                        Start an early, polled-mode console on the
                        Armada 3700 serial port at the specified
                        address. The serial port must already be setup
                        and configured. Options are not yet supported.

                qcom_geni,<addr>
                        Start an early, polled-mode console on a Qualcomm
                        Generic Interface (GENI) based serial port at the
                        specified address. The serial port must already be
                        setup and configured. Options are not yet supported.

                efifb,[options]
                        Start an early, unaccelerated console on the EFI
                        memory mapped framebuffer (if available). On cache
                        coherent non-x86 systems that use system memory for
                        the framebuffer, pass the 'ram' option so that it is
                        mapped with the correct attributes.

                linflex,<addr>
                        Use early console provided by Freescale LINFlexD UART
                        serial driver for NXP S32V234 SoCs. A valid base
                        address must be provided, and the serial port must
                        already be setup and configured.

        earlyprintk=    [X86,SH,ARM,M68k,S390]
                        earlyprintk=vga
                        earlyprintk=sclp
                        earlyprintk=xen
                        earlyprintk=serial[,ttySn[,baudrate]]
                        earlyprintk=serial[,0x...[,baudrate]]
                        earlyprintk=ttySn[,baudrate]
                        earlyprintk=dbgp[debugController#]
                        earlyprintk=pciserial[,force],bus:device.function[,
↪baudrate]
                        earlyprintk=xdbc[xhciController#]
                        earlyprintk=bios

                        earlyprintk is useful when the kernel crashes before
                        the normal console is initialized. It is not enabled by
                        default because it has some cosmetic problems.
```

```
                    Append ",keep" to not disable it when the real console
                    takes over.

                    Only one of vga, serial, or usb debug port can
                    be used at a time.

                    Currently only ttyS0 and ttyS1 may be specified by
                    name.  Other I/O ports may be explicitly specified
                    on some architectures (x86 and arm at least) by
                    replacing ttySn with an I/O port address, like this:
                            earlyprintk=serial,0x1008,115200
                    You can find the port for a given device in
                    /proc/tty/driver/serial:
                            2: uart:ST16650V2 port:00001008 irq:18 ...

                    Interaction with the standard serial driver is not
                    very good.

                    The VGA output is eventually overwritten by
                    the real console.

                    The xen option can only be used in Xen domains.

                    The sclp output can only be used on s390.

                    The bios output can only be used on SuperH.

                    The optional "force" to "pciserial" enables use of a
                    PCI device even when its classcode is not of the
                    UART class.

    edac_report=    [HW,EDAC] Control how to report EDAC event
                    Format: {"on" | "off" | "force"}
                    on: enable EDAC to report H/W event. May be overridden
                    by other higher priority error reporting module.
                    off: disable H/W event reporting through EDAC.
                    force: enforce the use of EDAC to report H/W event.
                    default: on.

    edd=            [EDD]
                    Format: {"off" | "on" | "skip[mbr]"}

    efi=            [EFI]
                    Format: { "debug", "disable_early_pci_dma",
                              "nochunk", "noruntime", "nosoftreserve",
                              "novamap", "no_disable_early_pci_dma" }
                    debug: enable misc debug output.
                    disable_early_pci_dma: disable the busmaster bit on all
                    PCI bridges while in the EFI boot stub.
                    nochunk: disable reading files in "chunks" in the EFI
```

```
                        boot stub, as chunking can cause problems with some
                        firmware implementations.
                        noruntime : disable EFI runtime services support
                        nosoftreserve: The EFI_MEMORY_SP (Specific Purpose)
                        attribute may cause the kernel to reserve the
                        memory range for a memory mapping driver to
                        claim. Specify efi=nosoftreserve to disable this
                        reservation and treat the memory by its base type
                        (i.e. EFI_CONVENTIONAL_MEMORY / "System RAM").
                        novamap: do not call SetVirtualAddressMap().
                        no_disable_early_pci_dma: Leave the busmaster bit set
                        on all PCI bridges while in the EFI boot stub

        efi_no_storage_paranoia [EFI; X86]
                        Using this parameter you can use more than 50% of
                        your efi variable storage. Use this parameter only if
                        you are really sure that your UEFI does sane gc and
                        fulfills the spec otherwise your board may brick.

        efi_fake_mem=   nn[KMG]@ss[KMG]:aa[,nn[KMG]@ss[KMG]:aa,..] [EFI; X86]
                        Add arbitrary attribute to specific memory range by
                        updating original EFI memory map.
                        Region of memory which aa attribute is added to is
                        from ss to ss+nn.

                        If efi_fake_mem=2G@4G:0x10000,2G@0x10a0000000:0x10000
                        is specified, EFI_MEMORY_MORE_RELIABLE(0x10000)
                        attribute is added to range 0x100000000-0x180000000 and
                        0x10a0000000-0x1120000000.

                        If efi_fake_mem=8G@9G:0x40000 is specified, the
                        EFI_MEMORY_SP(0x40000) attribute is added to
                        range 0x240000000-0x43fffffff.

                        Using this parameter you can do debugging of EFI memmap
                        related features. For example, you can do debugging of
                        Address Range Mirroring feature even if your box
                        doesn't support it, or mark specific memory as
                        "soft reserved".

        efivar_ssdt=    [EFI; X86] Name of an EFI variable that contains an␣
↪SSDT
                        that is to be dynamically loaded by Linux. If there are
                        multiple variables with the same name but with␣
↪different
                        vendor GUIDs, all of them will be loaded. See
                        Documentation/admin-guide/acpi/ssdt-overlays.rst for␣
↪details.


        eisa_irq_edge=  [PARISC,HW]
```

```
                        See header of drivers/parisc/eisa.c.

        ekgdboc=        [X86,KGDB] Allow early kernel console debugging
                        Format: ekgdboc=kbd

                        This is designed to be used in conjunction with
                        the boot argument: earlyprintk=vga

                        This parameter works in place of the kgdboc parameter
                        but can only be used if the backing tty is available
                        very early in the boot process. For early debugging
                        via a serial port see kgdboc_earlycon instead.

        elanfreq=       [X86-32]
                        See comment before function elanfreq_setup() in
                        arch/x86/kernel/cpu/cpufreq/elanfreq.c.

        elfcorehdr=[size[KMG]@]offset[KMG] [PPC,SH,X86,S390]
                        Specifies physical address of start of kernel core
                        image elf header and optionally the size. Generally
                        kexec loader will pass this option to capture kernel.
                        See Documentation/admin-guide/kdump/kdump.rst for␣
↪details.

        enable_mtrr_cleanup [X86]
                        The kernel tries to adjust MTRR layout from continuous
                        to discrete, to make X server driver able to add WB
                        entry later. This parameter enables that.

        enable_timer_pin_1 [X86]
                        Enable PIN 1 of APIC timer
                        Can be useful to work around chipset bugs
                        (in particular on some ATI chipsets).
                        The kernel tries to set a reasonable default.

        enforcing=      [SELINUX] Set initial enforcing status.
                        Format: {"0" | "1"}
                        See security/selinux/Kconfig help text.
                        0 -- permissive (log only, no denials).
                        1 -- enforcing (deny and log).
                        Default value is 0.
                        Value can be changed at runtime via
                        /sys/fs/selinux/enforce.

        erst_disable    [ACPI]
                        Disable Error Record Serialization Table (ERST)
                        support.

        ether=          [HW,NET] Ethernet cards parameters
                        This option is obsoleted by the "netdev=" option, which
                        has equivalent usage. See its documentation for␣
```

```
↪details.

        evm=            [EVM]
                        Format: { "fix" }
                        Permit 'security.evm' to be updated regardless of
                        current integrity status.

        early_page_ext [KNL] Enforces page_ext initialization to earlier
                        stages so cover more early boot allocations.
                        Please note that as side effect some optimizations
                        might be disabled to achieve that (e.g. parallelized
                        memory initialization is disabled) so the boot process
                        might take longer, especially on systems with a lot of
                        memory. Available with CONFIG_PAGE_EXTENSION=y.

        failslab=
        fail_usercopy=
        fail_page_alloc=
        fail_make_request=[KNL]
                        General fault injection mechanism.
                        Format: <interval>,<probability>,<space>,<times>
                        See also Documentation/fault-injection/.

        fb_tunnels=     [NET]
                        Format: { initns | none }
                        See Documentation/admin-guide/sysctl/net.rst for
                        fb_tunnels_only_for_init_ns

        floppy=         [HW]
                        See Documentation/admin-guide/blockdev/floppy.rst.

        forcepae        [X86-32]
                        Forcefully enable Physical Address Extension (PAE).
                        Many Pentium M systems disable PAE but may have a
                        functionally usable PAE implementation.
                        Warning: use of this parameter will taint the kernel
                        and may cause unknown problems.

        ftrace=[tracer]
                        [FTRACE] will set and start the specified tracer
                        as early as possible in order to facilitate early
                        boot debugging.

        ftrace_boot_snapshot
                        [FTRACE] On boot up, a snapshot will be taken of the
                        ftrace ring buffer that can be read at:
                        /sys/kernel/tracing/snapshot.
                        This is useful if you need tracing information from␣
↪kernel
                        boot up that is likely to be overridden by user space
                        start up functionality.
```

```
                        Optionally, the snapshot can also be defined for a␣
↪tracing
                        instance that was created by the trace_instance=␣
↪command
                        line parameter.

                        trace_instance=foo,sched_switch ftrace_boot_
↪snapshot=foo

                        The above will cause the "foo" tracing instance to␣
↪trigger
                        a snapshot at the end of boot up.

        ftrace_dump_on_oops[=orig_cpu]
                        [FTRACE] will dump the trace buffers on oops.
                        If no parameter is passed, ftrace will dump
                        buffers of all CPUs, but if you pass orig_cpu, it will
                        dump only the buffer of the CPU that triggered the
                        oops.

        ftrace_filter=[function-list]
                        [FTRACE] Limit the functions traced by the function
                        tracer at boot up. function-list is a comma-separated
                        list of functions. This list can be changed at run
                        time by the set_ftrace_filter file in the debugfs
                        tracing directory.

        ftrace_notrace=[function-list]
                        [FTRACE] Do not trace the functions specified in
                        function-list. This list can be changed at run time
                        by the set_ftrace_notrace file in the debugfs
                        tracing directory.

        ftrace_graph_filter=[function-list]
                        [FTRACE] Limit the top level callers functions traced
                        by the function graph tracer at boot up.
                        function-list is a comma-separated list of functions
                        that can be changed at run time by the
                        set_graph_function file in the debugfs tracing␣
↪directory.

        ftrace_graph_notrace=[function-list]
                        [FTRACE] Do not trace from the functions specified in
                        function-list.  This list is a comma-separated list of
                        functions that can be changed at run time by the
                        set_graph_notrace file in the debugfs tracing␣
↪directory.

        ftrace_graph_max_depth=<uint>
                        [FTRACE] Used with the function graph tracer. This is
```

```
                        the max depth it will trace into a function. This value
                        can be changed at run time by the max_graph_depth file
                        in the tracefs tracing directory. default: 0 (no limit)

        fw_devlink=     [KNL] Create device links between consumer and supplier
                        devices by scanning the firmware to infer the
                        consumer/supplier relationships. This feature is
                        especially useful when drivers are loaded as modules as
                        it ensures proper ordering of tasks like device probing
                        (suppliers first, then consumers), supplier boot state
                        clean up (only after all consumers have probed),
                        suspend/resume & runtime PM (consumers first, then
                        suppliers).
                        Format: { off | permissive | on | rpm }
                        off --  Don't create device links from firmware info.
                        permissive -- Create device links from firmware info
                                but use it only for ordering boot state clean
                                up (sync_state() calls).
                        on --   Create device links from firmware info and use␣
↪it
                                to enforce probe and suspend/resume ordering.
                        rpm --  Like "on", but also use to order runtime PM.

        fw_devlink.strict=<bool>
                        [KNL] Treat all inferred dependencies as mandatory
                        dependencies. This only applies for fw_devlink=on|rpm.
                        Format: <bool>

        fw_devlink.sync_state =
                        [KNL] When all devices that could probe have finished
                        probing, this parameter controls what to do with
                        devices that haven't yet received their sync_state()
                        calls.
                        Format: { strict | timeout }
                        strict -- Default. Continue waiting on consumers to
                                probe successfully.
                        timeout -- Give up waiting on consumers and call
                                sync_state() on any devices that haven't yet
                                received their sync_state() calls after
                                deferred_probe_timeout has expired or by
                                late_initcall() if !CONFIG_MODULES.

        gamecon.map[2|3]=
                        [HW,JOY] Multisystem joystick and NES/SNES/PSX pad
                        support via parallel port (up to 5 devices per port)
                        Format: <port#>,<pad1>,<pad2>,<pad3>,<pad4>,<pad5>
                        See also Documentation/input/devices/joystick-parport.
↪rst

        gamma=          [HW,DRM]
```

```
        gart_fix_e820=  [X86-64] disable the fix e820 for K8 GART
                        Format: off | on
                        default: on

        gather_data_sampling=
                        [X86,INTEL] Control the Gather Data Sampling (GDS)
                        mitigation.

                        Gather Data Sampling is a hardware vulnerability which
                        allows unprivileged speculative access to data which␣
↪was
                        previously stored in vector registers.

                        This issue is mitigated by default in updated␣
↪microcode.
                        The mitigation may have a performance impact but can be
                        disabled. On systems without the microcode mitigation
                        disabling AVX serves as a mitigation.

                        force:  Disable AVX to mitigate systems without
                                microcode mitigation. No effect if the␣
↪microcode
                                mitigation is present. Known to cause crashes␣
↪in
                                userspace with buggy AVX enumeration.

                        off:    Disable GDS mitigation.

        gcov_persist=   [GCOV] When non-zero (default), profiling data for
                        kernel modules is saved and remains accessible via
                        debugfs, even when the module is unloaded/reloaded.
                        When zero, profiling data is discarded and associated
                        debugfs files are removed at module unload time.

        goldfish        [X86] Enable the goldfish android emulator platform.
                        Don't use this when you are not running on the
                        android emulator

        gpio-mockup.gpio_mockup_ranges
                        [HW] Sets the ranges of gpiochip of for this device.
                        Format: <start1>,<end1>,<start2>,<end2>...
        gpio-mockup.gpio_mockup_named_lines
                        [HW] Let the driver know GPIO lines should be named.

        gpt             [EFI] Forces disk with valid GPT signature but
                        invalid Protective MBR to be treated as GPT. If the
                        primary GPT is corrupted, it enables the backup/
↪alternate
                        GPT to be used instead.

        grcan.enable0=  [HW] Configuration of physical interface 0. Determines
```

```
                        the "Enable 0" bit of the configuration register.
                        Format: 0 | 1
                        Default: 0
        grcan.enable1=  [HW] Configuration of physical interface 1. Determines
                        the "Enable 0" bit of the configuration register.
                        Format: 0 | 1
                        Default: 0
        grcan.select=   [HW] Select which physical interface to use.
                        Format: 0 | 1
                        Default: 0
        grcan.txsize=   [HW] Sets the size of the tx buffer.
                        Format: <unsigned int> such that (txsize & ~0x1fffc0)␣
↪== 0.
                        Default: 1024
        grcan.rxsize=   [HW] Sets the size of the rx buffer.
                        Format: <unsigned int> such that (rxsize & ~0x1fffc0)␣
↪== 0.
                        Default: 1024

        hardened_usercopy=
                        [KNL] Under CONFIG_HARDENED_USERCOPY, whether
                        hardening is enabled for this boot. Hardened
                        usercopy checking is used to protect the kernel
                        from reading or writing beyond known memory
                        allocation boundaries as a proactive defense
                        against bounds-checking flaws in the kernel's
                        copy_to_user()/copy_from_user() interface.
                on      Perform hardened usercopy checks (default).
                off     Disable hardened usercopy checks.

        hardlockup_all_cpu_backtrace=
                        [KNL] Should the hard-lockup detector generate
                        backtraces on all cpus.
                        Format: 0 | 1

        hashdist=       [KNL,NUMA] Large hashes allocated during boot
                        are distributed across NUMA nodes.  Defaults on
                        for 64-bit NUMA, off otherwise.
                        Format: 0 | 1 (for off | on)

        hcl=            [IA-64] SGI's Hardware Graph compatibility layer

        hd=             [EIDE] (E)IDE hard drive subsystem geometry
                        Format: <cyl>,<head>,<sect>

        hest_disable    [ACPI]
                        Disable Hardware Error Source Table (HEST) support;
                        corresponding firmware-first mode error processing
                        logic will be disabled.

        hibernate=      [HIBERNATION]
```

```
        noresume        Don't check if there's a hibernation image
                        present during boot.
        nocompress      Don't compress/decompress hibernation images.
        no              Disable hibernation and resume.
        protect_image   Turn on image protection during restoration
                        (that will set all pages holding image data
                        during restoration read-only).

highmem=nn[KMG] [KNL,BOOT] forces the highmem zone to have an exact
                size of <nn>. This works even on boxes that have no
                highmem otherwise. This also works to reduce highmem
                size on bigger boxes.

highres=        [KNL] Enable/disable high resolution timer mode.
                Valid parameters: "on", "off"
                Default: "on"

hlt             [BUGS=ARM,SH]

hostname=       [KNL] Set the hostname (aka UTS nodename).
                Format: <string>
                This allows setting the system's hostname during early
                startup. This sets the name returned by gethostname.
                Using this parameter to set the hostname makes it
                possible to ensure the hostname is correctly set before
                any userspace processes run, avoiding the possibility
                that a process may call gethostname before the hostname
                has been explicitly set, resulting in the calling
                process getting an incorrect result. The string must
                not exceed the maximum allowed hostname length (usually
                64 characters) and will be truncated otherwise.

hpet=           [X86-32,HPET] option to control HPET usage
                Format: { enable (default) | disable | force |
                        verbose }
                disable: disable HPET and use PIT instead
                force: allow force enabled of undocumented chips (ICH4,
                        VIA, nVidia)
                verbose: show contents of HPET registers during setup

hpet_mmap=      [X86, HPET_MMAP] Allow userspace to mmap HPET
                registers.  Default set by CONFIG_HPET_MMAP_DEFAULT.

hugepages=      [HW] Number of HugeTLB pages to allocate at boot.
                If this follows hugepagesz (below), it specifies
                the number of pages of hugepagesz to be allocated.
                If this is the first HugeTLB parameter on the command
                line, it specifies the number of pages to allocate for
                the default huge page size. If using node format, the
                number of pages to allocate per-node can be specified.
                See also Documentation/admin-guide/mm/hugetlbpage.rst.
```

```
                        Format: <integer> or (node format)
                                <node>:<integer>[,<node>:<integer>]

        hugepagesz=
                        [HW] The size of the HugeTLB pages.  This is used in
                        conjunction with hugepages (above) to allocate huge
                        pages of a specific size at boot.  The pair
                        hugepagesz=X hugepages=Y can be specified once for
                        each supported huge page size. Huge page sizes are
                        architecture dependent.  See also
                        Documentation/admin-guide/mm/hugetlbpage.rst.
                        Format: size[KMG]

        hugetlb_cma=    [HW,CMA] The size of a CMA area used for allocation
                        of gigantic hugepages. Or using node format, the size
                        of a CMA area per node can be specified.
                        Format: nn[KMGTPE] or (node format)
                                <node>:nn[KMGTPE][,<node>:nn[KMGTPE]]

                        Reserve a CMA area of given size and allocate gigantic
                        hugepages using the CMA allocator. If enabled, the
                        boot-time allocation of gigantic hugepages is skipped.

        hugetlb_free_vmemmap=
                        [KNL] Requires CONFIG_HUGETLB_PAGE_OPTIMIZE_VMEMMAP
                        enabled.
                        Control if HugeTLB Vmemmap Optimization (HVO) is␣
↪enabled.
                        Allows heavy hugetlb users to free up some more
                        memory (7 * PAGE_SIZE for each 2MB hugetlb page).
                        Format: { on | off (default) }

                        on: enable HVO
                        off: disable HVO

                        Built with CONFIG_HUGETLB_PAGE_OPTIMIZE_VMEMMAP_
↪DEFAULT_ON=y,
                        the default is on.

                        Note that the vmemmap pages may be allocated from the␣
↪added
                        memory block itself when memory_hotplug.memmap_on_
↪memory is
                        enabled, those vmemmap pages cannot be optimized even␣
↪if this
                        feature is enabled.  Other vmemmap pages not allocated␣
↪from
                        the added memory block itself do not be affected.

        hung_task_panic=
                        [KNL] Should the hung task detector generate panics.
```

```
                        Format: 0 | 1

                        A value of 1 instructs the kernel to panic when a
                        hung task is detected. The default value is controlled
                        by the CONFIG_BOOTPARAM_HUNG_TASK_PANIC build-time
                        option. The value selected by this boot parameter can
                        be changed later by the kernel.hung_task_panic sysctl.

        hvc_iucv=       [S390]  Number of z/VM IUCV hypervisor console (HVC)
                                terminal devices. Valid values: 0..8
        hvc_iucv_allow= [S390]  Comma-separated list of z/VM user IDs.
                                If specified, z/VM IUCV HVC accepts connections
                                from listed z/VM user IDs only.

        hv_nopvspin     [X86,HYPER_V] Disables the paravirt spinlock␣
↪optimizations
                                      which allow the hypervisor to 'idle' the
                                      guest on lock contention.

        i2c_bus=        [HW]    Override the default board specific I2C bus␣
↪speed
                                or register an additional I2C bus that is not
                                registered from board initialization code.
                                Format:
                                <bus_id>,<clkrate>

        i8042.debug     [HW] Toggle i8042 debug mode
        i8042.unmask_kbd_data
                        [HW] Enable printing of interrupt data from the KBD␣
↪port
                             (disabled by default, and as a pre-condition
                             requires that i8042.debug=1 be enabled)
        i8042.direct    [HW] Put keyboard port into non-translated mode
        i8042.dumbkbd   [HW] Pretend that controller can only read data from
                             keyboard and cannot control its state
                             (Don't attempt to blink the leds)
        i8042.noaux     [HW] Don't check for auxiliary (== mouse) port
        i8042.nokbd     [HW] Don't check/create keyboard port
        i8042.noloop    [HW] Disable the AUX Loopback command while probing
                             for the AUX port
        i8042.nomux     [HW] Don't check presence of an active multiplexing
                             controller
        i8042.nopnp     [HW] Don't use ACPIPnP / PnPBIOS to discover KBD/AUX
                             controllers
        i8042.notimeout [HW] Ignore timeout condition signalled by controller
        i8042.reset     [HW] Reset the controller during init, cleanup and
                             suspend-to-ram transitions, only during s2r
                             transitions, or never reset
                        Format: { 1 | Y | y | 0 | N | n }
                        1, Y, y: always reset controller
                        0, N, n: don't ever reset controller
```

```
                        Default: only on s2r transitions on x86; most other
                        architectures force reset to be always executed
        i8042.unlock    [HW] Unlock (ignore) the keylock
        i8042.kbdreset  [HW] Reset device connected to KBD port
        i8042.probe_defer
                        [HW] Allow deferred probing upon i8042 probe errors

        i810=           [HW,DRM]

        i915.invert_brightness=
                        [DRM] Invert the sense of the variable that is used to
                        set the brightness of the panel backlight. Normally a
                        brightness value of 0 indicates backlight switched off,
                        and the maximum of the brightness value sets the␣
↪backlight
                        to maximum brightness. If this parameter is set to 0
                        (default) and the machine requires it, or this␣
↪parameter
                        is set to 1, a brightness value of 0 sets the backlight
                        to maximum brightness, and the maximum of the␣
↪brightness
                        value switches the backlight off.
                        -1 -- never invert brightness
                         0 -- machine default
                         1 -- force brightness inversion

        ia32_emulation= [X86-64]
                        Format: <bool>
                        When true, allows loading 32-bit programs and␣
↪executing 32-bit
                        syscalls, essentially overriding IA32_EMULATION_
↪DEFAULT_DISABLED at
                        boot time. When false, unconditionally disables IA32␣
↪emulation.

        icn=            [HW,ISDN]
                        Format: <io>[,<membase>[,<icn_id>[,<icn_id2>]]]


        idle=           [X86]
                        Format: idle=poll, idle=halt, idle=nomwait
                        Poll forces a polling idle loop that can slightly
                        improve the performance of waking up a idle CPU, but
                        will use a lot of power and make the system run hot.
                        Not recommended.
                        idle=halt: Halt is forced to be used for CPU idle.
                        In such case C2/C3 won't be used again.
                        idle=nomwait: Disable mwait for CPU C-states

        idxd.sva=       [HW]
                        Format: <bool>
```

```
                Allow force disabling of Shared Virtual Memory (SVA)
                support for the idxd driver. By default it is set to
                true (1).

idxd.tc_override= [HW]
                Format: <bool>
                Allow override of default traffic class configuration
                for the device. By default it is set to false (0).

ieee754=        [MIPS] Select IEEE Std 754 conformance mode
                Format: { strict | legacy | 2008 | relaxed }
                Default: strict

                Choose which programs will be accepted for execution
                based on the IEEE 754 NaN encoding(s) supported by
                the FPU and the NaN encoding requested with the value
                of an ELF file header flag individually set by each
                binary.  Hardware implementations are permitted to
                support either or both of the legacy and the 2008 NaN
                encoding mode.

                Available settings are as follows:
                strict  accept binaries that request a NaN encoding
                        supported by the FPU
                legacy  only accept legacy-NaN binaries, if supported
                        by the FPU
                2008    only accept 2008-NaN binaries, if supported
                        by the FPU
                relaxed accept any binaries regardless of whether
                        supported by the FPU

                The FPU emulator is always able to support both NaN
                encodings, so if no FPU hardware is present or it has
                been disabled with 'nofpu', then the settings of
                'legacy' and '2008' strap the emulator accordingly,
                'relaxed' straps the emulator for both legacy-NaN and
                2008-NaN, whereas 'strict' enables legacy-NaN only on
                legacy processors and both NaN encodings on MIPS32 or
                MIPS64 CPUs.

                The setting for ABS.fmt/NEG.fmt instruction execution
                mode generally follows that for the NaN encoding,
                except where unsupported by hardware.

ignore_loglevel [KNL]
                Ignore loglevel setting - this will print /all/
                kernel messages to the console. Useful for debugging.
                We also add it as printk module parameter, so users
                could change it dynamically, usually by
                /sys/module/printk/parameters/ignore_loglevel.
```

```
        ignore_rlimit_data
                        Ignore RLIMIT_DATA setting for data mappings,
                        print warning at first misuse.  Can be changed via
                        /sys/module/kernel/parameters/ignore_rlimit_data.

        ihash_entries=  [KNL]
                        Set number of hash buckets for inode cache.

        ima_appraise=   [IMA] appraise integrity measurements
                        Format: { "off" | "enforce" | "fix" | "log" }
                        default: "enforce"

        ima_appraise_tcb [IMA] Deprecated.  Use ima_policy= instead.
                        The builtin appraise policy appraises all files
                        owned by uid=0.

        ima_canonical_fmt [IMA]
                        Use the canonical format for the binary runtime
                        measurements, instead of host native format.

        ima_hash=       [IMA]
                        Format: { md5 | sha1 | rmd160 | sha256 | sha384
                                   | sha512 | ... }
                        default: "sha1"

                        The list of supported hash algorithms is defined
                        in crypto/hash_info.h.

        ima_policy=     [IMA]
                        The builtin policies to load during IMA setup.
                        Format: "tcb | appraise_tcb | secure_boot |
                                 fail_securely | critical_data"

                        The "tcb" policy measures all programs exec'd, files
                        mmap'd for exec, and all files opened with the read
                        mode bit set by either the effective uid (euid=0) or
                        uid=0.

                        The "appraise_tcb" policy appraises the integrity of
                        all files owned by root.

                        The "secure_boot" policy appraises the integrity
                        of files (eg. kexec kernel image, kernel modules,
                        firmware, policy, etc) based on file signatures.

                        The "fail_securely" policy forces file signature
                        verification failure also on privileged mounted
                        filesystems with the SB_I_UNVERIFIABLE_SIGNATURE
                        flag.

                        The "critical_data" policy measures kernel integrity
```

```
                        critical data.

        ima_tcb         [IMA] Deprecated.  Use ima_policy= instead.
                        Load a policy which meets the needs of the Trusted
                        Computing Base.  This means IMA will measure all
                        programs exec'd, files mmap'd for exec, and all files
                        opened for read by uid=0.

        ima_template=   [IMA]
                        Select one of defined IMA measurements template␣
↪formats.
                        Formats: { "ima" | "ima-ng" | "ima-ngv2" | "ima-sig" |
                                   "ima-sigv2" }
                        Default: "ima-ng"

        ima_template_fmt=
                        [IMA] Define a custom template format.
                        Format: { "field1|...|fieldN" }

        ima.ahash_minsize= [IMA] Minimum file size for asynchronous hash usage
                        Format: <min_file_size>
                        Set the minimal file size for using asynchronous hash.
                        If left unspecified, ahash usage is disabled.

                        ahash performance varies for different data sizes on
                        different crypto accelerators. This option can be used
                        to achieve the best performance for a particular HW.

        ima.ahash_bufsize= [IMA] Asynchronous hash buffer size
                        Format: <bufsize>
                        Set hashing buffer size. Default: 4k.

                        ahash performance varies for different chunk sizes on
                        different crypto accelerators. This option can be used
                        to achieve best performance for particular HW.

        init=           [KNL]
                        Format: <full_path>
                        Run specified binary instead of /sbin/init as init
                        process.

        initcall_debug  [KNL] Trace initcalls as they are executed.  Useful
                        for working out where the kernel is dying during
                        startup.

        initcall_blacklist=  [KNL] Do not execute a comma-separated list of
                        initcall functions.  Useful for debugging built-in
                        modules and initcalls.

        initramfs_async= [KNL]
                        Format: <bool>
```

```
                        Default: 1
                        This parameter controls whether the initramfs
                        image is unpacked asynchronously, concurrently
                        with devices being probed and
                        initialized. This should normally just work,
                        but as a debugging aid, one can get the
                        historical behaviour of the initramfs
                        unpacking being completed before device_ and
                        late_ initcalls.

        initrd=         [BOOT] Specify the location of the initial ramdisk

        initrdmem=      [KNL] Specify a physical address and size from which to
                        load the initrd. If an initrd is compiled in or
                        specified in the bootparams, it takes priority over␣
↪this
                        setting.
                        Format: ss[KMG],nn[KMG]
                        Default is 0, 0

        init_on_alloc=  [MM] Fill newly allocated pages and heap objects with
                        zeroes.
                        Format: 0 | 1
                        Default set by CONFIG_INIT_ON_ALLOC_DEFAULT_ON.

        init_on_free=   [MM] Fill freed pages and heap objects with zeroes.
                        Format: 0 | 1
                        Default set by CONFIG_INIT_ON_FREE_DEFAULT_ON.

        init_pkru=      [X86] Specify the default memory protection keys rights
                        register contents for all processes.  0x55555554 by
                        default (disallow access to all but pkey 0).  Can
                        override in debugfs after boot.

        inport.irq=     [HW] Inport (ATI XL and Microsoft) busmouse driver
                        Format: <irq>

        int_pln_enable  [X86] Enable power limit notification interrupt

        integrity_audit=[IMA]
                        Format: { "0" | "1" }
                        0 -- basic integrity auditing messages. (Default)
                        1 -- additional integrity auditing messages.

        intel_iommu=    [DMAR] Intel IOMMU driver (DMAR) option
                on
                        Enable intel iommu driver.
                off
                        Disable intel iommu driver.
                igfx_off [Default Off]
                        By default, gfx is mapped as normal device. If a gfx
```

```
                        device has a dedicated DMAR unit, the DMAR unit is
                        bypassed by not enabling DMAR with this option. In
                        this case, gfx device will use physical address for
                        DMA.
                strict [Default Off]
                        Deprecated, equivalent to iommu.strict=1.
                sp_off [Default Off]
                        By default, super page will be supported if Intel IOMMU
                        has the capability. With this option, super page will
                        not be supported.
                sm_on
                        Enable the Intel IOMMU scalable mode if the hardware
                        advertises that it has support for the scalable mode
                        translation.
                sm_off
                        Disallow use of the Intel IOMMU scalable mode.
                tboot_noforce [Default Off]
                        Do not force the Intel IOMMU enabled under tboot.
                        By default, tboot will force Intel IOMMU on, which
                        could harm performance of some high-throughput
                        devices like 40GBit network cards, even if identity
                        mapping is enabled.
                        Note that using this option lowers the security
                        provided by tboot because it makes the system
                        vulnerable to DMA attacks.

        intel_idle.max_cstate=  [KNL,HW,ACPI,X86]
                        0       disables intel_idle and fall back on acpi_idle.
                        1 to 9  specify maximum depth of C-state.

        intel_pstate=   [X86]
                        disable
                          Do not enable intel_pstate as the default
                          scaling driver for the supported processors
                        active
                          Use intel_pstate driver to bypass the scaling
                          governors layer of cpufreq and provides it own
                          algorithms for p-state selection. There are two
                          P-state selection algorithms provided by
                          intel_pstate in the active mode: powersave and
                          performance.  The way they both operate depends
                          on whether or not the hardware managed P-states
                          (HWP) feature has been enabled in the processor
                          and possibly on the processor model.
                        passive
                          Use intel_pstate as a scaling driver, but configure␣
↪it
                          to work with generic cpufreq governors (instead of
                          enabling its internal governor).  This mode cannot be
                          used along with the hardware-managed P-states (HWP)
                          feature.
```

```
                        force
                          Enable intel_pstate on systems that prohibit it by␣
↪default
                          in favor of acpi-cpufreq. Forcing the intel_pstate␣
↪driver
                          instead of acpi-cpufreq may disable platform␣
↪features, such
                          as thermal controls and power capping, that rely on␣
↪ACPI
                          P-States information being indicated to OSPM and␣
↪therefore
                          should be used with caution. This option does not␣
↪work with
                          processors that aren't supported by the intel_pstate␣
↪driver
                          or on platforms that use pcc-cpufreq instead of␣
↪acpi-cpufreq.
                        no_hwp
                          Do not enable hardware P state control (HWP)
                          if available.
                        hwp_only
                          Only load intel_pstate on systems which support
                          hardware P state control (HWP) if available.
                        support_acpi_ppc
                          Enforce ACPI _PPC performance limits. If the Fixed␣
↪ACPI
                          Description Table, specifies preferred power␣
↪management
                          profile as "Enterprise Server" or "Performance␣
↪Server",
                          then this feature is turned on by default.
                        per_cpu_perf_limits
                          Allow per-logical-CPU P-State performance control␣
↪limits using
                          cpufreq sysfs interface

        intremap=       [X86-64, Intel-IOMMU]
                        on      enable Interrupt Remapping (default)
                        off     disable Interrupt Remapping
                        nosid   disable Source ID checking
                        no_x2apic_optout
                                BIOS x2APIC opt-out request will be ignored
                        nopost  disable Interrupt Posting

        iomem=          Disable strict checking of access to MMIO memory
                strict  regions from userspace.
                relaxed

        iommu=          [X86]
                off
                force
```

```
                noforce
                biomerge
                panic
                nopanic
                merge
                nomerge
                soft
                pt              [X86]
                nopt            [X86]
                nobypass        [PPC/POWERNV]
                        Disable IOMMU bypass, using IOMMU for PCI devices.

        iommu.forcedac= [ARM64, X86] Control IOVA allocation for PCI devices.
                        Format: { "0" | "1" }
                        0 - Try to allocate a 32-bit DMA address first, before
                          falling back to the full range if needed.
                        1 - Allocate directly from the full usable range,
                          forcing Dual Address Cycle for PCI cards supporting
                          greater than 32-bit addressing.

        iommu.strict=   [ARM64, X86, S390] Configure TLB invalidation behaviour
                        Format: { "0" | "1" }
                        0 - Lazy mode.
                          Request that DMA unmap operations use deferred
                          invalidation of hardware TLBs, for increased
                          throughput at the cost of reduced device isolation.
                          Will fall back to strict mode if not supported by
                          the relevant IOMMU driver.
                        1 - Strict mode.
                          DMA unmap operations invalidate IOMMU hardware TLBs
                          synchronously.
                        unset - Use value of CONFIG_IOMMU_DEFAULT_DMA_{LAZY,
↪STRICT}.
                        Note: on x86, strict mode specified via one of the
                        legacy driver-specific options takes precedence.

        iommu.passthrough=
                        [ARM64, X86] Configure DMA to bypass the IOMMU by␣
↪default.
                        Format: { "0" | "1" }
                        0 - Use IOMMU translation for DMA.
                        1 - Bypass the IOMMU for DMA.
                        unset - Use value of CONFIG_IOMMU_DEFAULT_PASSTHROUGH.

        io7=            [HW] IO7 for Marvel-based Alpha systems
                        See comment before marvel_specify_io7 in
                        arch/alpha/kernel/core_marvel.c.

        io_delay=       [X86] I/O delay method
                0x80
                        Standard port 0x80 based delay
```

```
                0xed
                        Alternate port 0xed based delay (needed on some␣
↪systems)
                udelay
                        Simple two microseconds delay
                none
                        No delay

        ip=             [IP_PNP]
                        See Documentation/admin-guide/nfs/nfsroot.rst.

        ipcmni_extend   [KNL] Extend the maximum number of unique System V
                        IPC identifiers from 32,768 to 16,777,216.

        irqaffinity=    [SMP] Set the default irq affinity mask
                        The argument is a cpu list, as described above.

        irqchip.gicv2_force_probe=
                        [ARM, ARM64]
                        Format: <bool>
                        Force the kernel to look for the second 4kB page
                        of a GICv2 controller even if the memory range
                        exposed by the device tree is too small.

        irqchip.gicv3_nolpi=
                        [ARM, ARM64]
                        Force the kernel to ignore the availability of
                        LPIs (and by consequence ITSs). Intended for system
                        that use the kernel as a bootloader, and thus want
                        to let secondary kernels in charge of setting up
                        LPIs.

        irqchip.gicv3_pseudo_nmi= [ARM64]
                        Enables support for pseudo-NMIs in the kernel. This
                        requires the kernel to be built with
                        CONFIG_ARM64_PSEUDO_NMI.

        irqfixup        [HW]
                        When an interrupt is not handled search all handlers
                        for it. Intended to get systems with badly broken
                        firmware running.

        irqpoll         [HW]
                        When an interrupt is not handled search all handlers
                        for it. Also check all handlers each timer
                        interrupt. Intended to get systems with badly broken
                        firmware running.

        isapnp=         [ISAPNP]
                        Format: <RDP>,<reset>,<pci_scan>,<verbosity>
```

```
        isolcpus=       [KNL,SMP,ISOL] Isolate a given set of CPUs from␣
↪disturbance.
                        [Deprecated - use cpusets instead]
                        Format: [flag-list,]<cpu-list>

                        Specify one or more CPUs to isolate from disturbances
                        specified in the flag list (default: domain):

                        nohz
                          Disable the tick when a single task runs.

                          A residual 1Hz tick is offloaded to workqueues,␣
↪which you
                          need to affine to housekeeping through the global
                          workqueue's affinity configured via the
                          /sys/devices/virtual/workqueue/cpumask sysfs file, or
                          by using the 'domain' flag described below.

                          NOTE: by default the global workqueue runs on all␣
↪CPUs,
                          so to protect individual CPUs the 'cpumask' file has␣
↪to
                          be configured manually after bootup.

                        domain
                          Isolate from the general SMP balancing and scheduling
                          algorithms. Note that performing domain isolation␣
↪this way
                          is irreversible: it's not possible to bring back a␣
↪CPU to
                          the domains once isolated through isolcpus. It's␣
↪strongly
                          advised to use cpusets instead to disable scheduler␣
↪load
                          balancing through the "cpuset.sched_load_balance"␣
↪file.
                          It offers a much more flexible interface where CPUs␣
↪can
                          move in and out of an isolated set anytime.

                          You can move a process onto or off an "isolated" CPU␣
↪via
                          the CPU affinity syscalls or cpuset.
                          <cpu number> begins at 0 and the maximum value is
                          "number of CPUs in system - 1".

                        managed_irq

                          Isolate from being targeted by managed interrupts
                          which have an interrupt mask containing isolated
                          CPUs. The affinity of managed interrupts is
```

```
                    handled by the kernel and cannot be changed via
                    the /proc/irq/* interfaces.

                    This isolation is best effort and only effective
                    if the automatically assigned interrupt mask of a
                    device queue contains isolated and housekeeping
                    CPUs. If housekeeping CPUs are online then such
                    interrupts are directed to the housekeeping CPU
                    so that IO submitted on the housekeeping CPU
                    cannot disturb the isolated CPU.

                    If a queue's affinity mask contains only isolated
                    CPUs then this parameter has no effect on the
                    interrupt routing decision, though interrupts are
                    only delivered when tasks running on those
                    isolated CPUs submit IO. IO submitted on
                    housekeeping CPUs has no influence on those
                    queues.

                  The format of <cpu-list> is described above.

iucv=             [HW,NET]

ivrs_ioapic       [HW,X86-64]
                  Provide an override to the IOAPIC-ID<->DEVICE-ID
                  mapping provided in the IVRS ACPI table.
                  By default, PCI segment is 0, and can be omitted.

                  For example, to map IOAPIC-ID decimal 10 to
                  PCI segment 0x1 and PCI device 00:14.0,
                  write the parameter as:
                          ivrs_ioapic=10@0001:00:14.0

                  Deprecated formats:
                  * To map IOAPIC-ID decimal 10 to PCI device 00:14.0
                    write the parameter as:
                          ivrs_ioapic[10]=00:14.0
                  * To map IOAPIC-ID decimal 10 to PCI segment 0x1 and
                    PCI device 00:14.0 write the parameter as:
                          ivrs_ioapic[10]=0001:00:14.0

ivrs_hpet         [HW,X86-64]
                  Provide an override to the HPET-ID<->DEVICE-ID
                  mapping provided in the IVRS ACPI table.
                  By default, PCI segment is 0, and can be omitted.

                  For example, to map HPET-ID decimal 10 to
                  PCI segment 0x1 and PCI device 00:14.0,
                  write the parameter as:
                          ivrs_hpet=10@0001:00:14.0
```

```
                        Deprecated formats:
                        * To map HPET-ID decimal 0 to PCI device 00:14.0
                          write the parameter as:
                                ivrs_hpet[0]=00:14.0
                        * To map HPET-ID decimal 10 to PCI segment 0x1 and
                          PCI device 00:14.0 write the parameter as:
                                ivrs_ioapic[10]=0001:00:14.0

        ivrs_acpihid    [HW,X86-64]
                        Provide an override to the ACPI-HID:UID<->DEVICE-ID
                        mapping provided in the IVRS ACPI table.
                        By default, PCI segment is 0, and can be omitted.

                        For example, to map UART-HID:UID AMD0020:0 to
                        PCI segment 0x1 and PCI device ID 00:14.5,
                        write the parameter as:
                                ivrs_acpihid=AMD0020:0@0001:00:14.5

                        Deprecated formats:
                        * To map UART-HID:UID AMD0020:0 to PCI segment is 0,
                          PCI device ID 00:14.5, write the parameter as:
                                ivrs_acpihid[00:14.5]=AMD0020:0
                        * To map UART-HID:UID AMD0020:0 to PCI segment 0x1 and
                          PCI device ID 00:14.5, write the parameter as:
                                ivrs_acpihid[0001:00:14.5]=AMD0020:0

        js=             [HW,JOY] Analog joystick
                        See Documentation/input/joydev/joystick.rst.

        kasan_multi_shot
                        [KNL] Enforce KASAN (Kernel Address Sanitizer) to print
                        report on every invalid memory access. Without this
                        parameter KASAN will print report only for the first
                        invalid access.

        keep_bootcon    [KNL]
                        Do not unregister boot console at start. This is only
                        useful for debugging when something happens in the␣
↪window
                        between unregistering the boot console and initializing
                        the real console.

        keepinitrd      [HW,ARM] See retain_initrd.

        kernelcore=     [KNL,X86,IA-64,PPC]
                        Format: nn[KMGTPE] | nn% | "mirror"
                        This parameter specifies the amount of memory usable by
                        the kernel for non-movable allocations.  The requested
                        amount is spread evenly throughout all nodes in the
                        system as ZONE_NORMAL.  The remaining memory is used␣
↪for
```

```
                        movable memory in its own zone, ZONE_MOVABLE.  In the
                        event, a node is too small to have both ZONE_NORMAL and
                        ZONE_MOVABLE, kernelcore memory will take priority and
                        other nodes will have a larger ZONE_MOVABLE.

                        ZONE_MOVABLE is used for the allocation of pages that
                        may be reclaimed or moved by the page migration
                        subsystem.  Note that allocations like␣
↪PTEs-from-HighMem
                        still use the HighMem zone if it exists, and the Normal
                        zone if it does not.

                        It is possible to specify the exact amount of memory in
                        the form of "nn[KMGTPE]", a percentage of total system
                        memory in the form of "nn%", or "mirror".  If "mirror"
                        option is specified, mirrored (reliable) memory is used
                        for non-movable allocations and remaining memory is␣
↪used
                        for Movable pages.  "nn[KMGTPE]", "nn%", and "mirror"
                        are exclusive, so you cannot specify multiple forms.

        kgdbdbgp=       [KGDB,HW] kgdb over EHCI usb debug port.
                        Format: <Controller#>[,poll interval]
                        The controller # is the number of the ehci usb debug
                        port as it is probed via PCI.  The poll interval is
                        optional and is the number seconds in between
                        each poll cycle to the debug port in case you need
                        the functionality for interrupting the kernel with
                        gdb or control-c on the dbgp connection.  When
                        not using this parameter you use sysrq-g to break into
                        the kernel debugger.

        kgdboc=         [KGDB,HW] kgdb over consoles.
                        Requires a tty driver that supports console polling,
                        or a supported polling keyboard driver (non-usb).
                         Serial only format: <serial_device>[,baud]
                         keyboard only format: kbd
                         keyboard and serial format: kbd,<serial_device>[,baud]
                        Optional Kernel mode setting:
                         kms, kbd format: kms,kbd
                         kms, kbd and serial format: kms,kbd,<ser_dev>[,baud]

        kgdboc_earlycon=        [KGDB,HW]
                        If the boot console provides the ability to read
                        characters and can work in polling mode, you can use
                        this parameter to tell kgdb to use it as a backend
                        until the normal console is registered. Intended to
                        be used together with the kgdboc parameter which
                        specifies the normal console to transition to.

                        The name of the early console should be specified
```

```
                  as the value of this parameter. Note that the name of
                  the early console might be different than the tty
                  name passed to kgdboc. It's OK to leave the value
                  blank and the first boot console that implements
                  read() will be picked.

kgdbwait          [KGDB] Stop kernel execution and enter the
                  kernel debugger at the earliest opportunity.

kmac=             [MIPS] Korina ethernet MAC address.
                  Configure the RouterBoard 532 series on-chip
                  Ethernet adapter MAC address.

kmemleak=         [KNL] Boot-time kmemleak enable/disable
                  Valid arguments: on, off
                  Default: on
                  Built with CONFIG_DEBUG_KMEMLEAK_DEFAULT_OFF=y,
                  the default is off.

kprobe_event=[probe-list]
                  [FTRACE] Add kprobe events and enable at boot time.
                  The probe-list is a semicolon delimited list of probe
                  definitions. Each definition is same as kprobe_events
                  interface, but the parameters are comma delimited.
                  For example, to add a kprobe event on vfs_read with
                  arg1 and arg2, add to the command line;

                        kprobe_event=p,vfs_read,$arg1,$arg2

                  See also Documentation/trace/kprobetrace.rst "Kernel
                  Boot Parameter" section.

kpti=             [ARM64] Control page table isolation of user
                  and kernel address spaces.
                  Default: enabled on cores which need mitigation.
                  0: force disabled
                  1: force enabled

kunit.enable=     [KUNIT] Enable executing KUnit tests. Requires
                  CONFIG_KUNIT to be set to be fully enabled. The
                  default value can be overridden via
                  KUNIT_DEFAULT_ENABLED.
                  Default is 1 (enabled)

kvm.ignore_msrs=[KVM] Ignore guest accesses to unhandled MSRs.
                  Default is 0 (don't ignore, but inject #GP)

kvm.eager_page_split=
                  [KVM,X86] Controls whether or not KVM will try to
                  proactively split all huge pages during dirty logging.
                  Eager page splitting reduces interruptions to vCPU
```

```
                        execution by eliminating the write-protection faults
                        and MMU lock contention that would otherwise be
                        required to split huge pages lazily.

                        VM workloads that rarely perform writes or that write
                        only to a small region of VM memory may benefit from
                        disabling eager page splitting to allow huge pages to
                        still be used for reads.

                        The behavior of eager page splitting depends on whether
                        KVM_DIRTY_LOG_INITIALLY_SET is enabled or disabled. If
                        disabled, all huge pages in a memslot will be eagerly
                        split when dirty logging is enabled on that memslot. If
                        enabled, eager page splitting will be performed during
                        the KVM_CLEAR_DIRTY ioctl, and only for the pages being
                        cleared.

                        Eager page splitting is only supported when kvm.tdp_
↪mmu=Y.

                        Default is Y (on).

        kvm.enable_vmware_backdoor=[KVM] Support VMware backdoor PV interface.
                                   Default is false (don't support).

        kvm.nx_huge_pages=
                        [KVM] Controls the software workaround for the
                        X86_BUG_ITLB_MULTIHIT bug.
                        force   : Always deploy workaround.
                        off     : Never deploy workaround.
                        auto    : Deploy workaround based on the presence of
                                  X86_BUG_ITLB_MULTIHIT.

                        Default is 'auto'.

                        If the software workaround is enabled for the host,
                        guests do need not to enable it for nested guests.

        kvm.nx_huge_pages_recovery_ratio=
                        [KVM] Controls how many 4KiB pages are periodically␣
↪zapped
                        back to huge pages.  0 disables the recovery,␣
↪otherwise if
                        the value is N KVM will zap 1/Nth of the 4KiB pages␣
↪every
                        period (see below).  The default is 60.

        kvm.nx_huge_pages_recovery_period_ms=
                        [KVM] Controls the time period at which KVM zaps 4KiB␣
↪pages
                        back to huge pages. If the value is a non-zero N, KVM␣
```

```
↪will
                        zap a portion (see ratio above) of the pages every N␣
↪msecs.
                        If the value is 0 (the default), KVM will pick a␣
↪period based
                        on the ratio, such that a page is zapped after 1 hour␣
↪on average.

        kvm-amd.nested= [KVM,AMD] Control nested virtualization feature in
                        KVM/SVM. Default is 1 (enabled).

        kvm-amd.npt=    [KVM,AMD] Control KVM's use of Nested Page Tables,
                        a.k.a. Two-Dimensional Page Tables. Default is 1
                        (enabled). Disable by KVM if hardware lacks support
                        for NPT.

        kvm-arm.mode=
                        [KVM,ARM] Select one of KVM/arm64's modes of operation.

                        none: Forcefully disable KVM.

                        nvhe: Standard nVHE-based mode, without support for
                              protected guests.

                        protected: nVHE-based mode with support for guests␣
↪whose
                                   state is kept private from the host.

                        nested: VHE-based mode with support for nested
                                virtualization. Requires at least ARMv8.3
                                hardware.

                        Defaults to VHE/nVHE based on hardware support. Setting
                        mode to "protected" will disable kexec and hibernation
                        for the host. "nested" is experimental and should be
                        used with extreme caution.

        kvm-arm.vgic_v3_group0_trap=
                        [KVM,ARM] Trap guest accesses to GICv3 group-0
                        system registers

        kvm-arm.vgic_v3_group1_trap=
                        [KVM,ARM] Trap guest accesses to GICv3 group-1
                        system registers

        kvm-arm.vgic_v3_common_trap=
                        [KVM,ARM] Trap guest accesses to GICv3 common
                        system registers

        kvm-arm.vgic_v4_enable=
                        [KVM,ARM] Allow use of GICv4 for direct injection of
```

```
                        LPIs.

        kvm_cma_resv_ratio=n [PPC]
                        Reserves given percentage from system memory area for
                        contiguous memory allocation for KVM hash pagetable
                        allocation.
                        By default it reserves 5% of total system memory.
                        Format: <integer>
                        Default: 5

        kvm-intel.ept=  [KVM,Intel] Control KVM's use of Extended Page Tables,
                        a.k.a. Two-Dimensional Page Tables.  Default is 1
                        (enabled). Disable by KVM if hardware lacks support
                        for EPT.

        kvm-intel.emulate_invalid_guest_state=
                        [KVM,Intel] Control whether to emulate invalid guest
                        state. Ignored if kvm-intel.enable_unrestricted_
↪guest=1,
                        as guest state is never invalid for unrestricted
                        guests. This param doesn't apply to nested guests (L2),
                        as KVM never emulates invalid L2 guest state.
                        Default is 1 (enabled).

        kvm-intel.flexpriority=
                        [KVM,Intel] Control KVM's use of FlexPriority feature
                        (TPR shadow). Default is 1 (enabled). Disable by KVM if
                        hardware lacks support for it.

        kvm-intel.nested=
                        [KVM,Intel] Control nested virtualization feature in
                        KVM/VMX. Default is 1 (enabled).

        kvm-intel.unrestricted_guest=
                        [KVM,Intel] Control KVM's use of unrestricted guest
                        feature (virtualized real and unpaged mode). Default
                        is 1 (enabled). Disable by KVM if EPT is disabled or
                        hardware lacks support for it.

        kvm-intel.vmentry_l1d_flush=[KVM,Intel] Mitigation for L1 Terminal␣
↪Fault
                        CVE-2018-3620.

                        Valid arguments: never, cond, always

                        always: L1D cache flush on every VMENTER.
                        cond:   Flush L1D on VMENTER only when the code between
                                VMEXIT and VMENTER can leak host memory.
                        never:  Disables the mitigation

                        Default is cond (do L1 cache flush in specific␣
```

```
↪instances)

        kvm-intel.vpid= [KVM,Intel] Control KVM's use of Virtual Processor
                        Identification feature (tagged TLBs). Default is 1
                        (enabled). Disable by KVM if hardware lacks support
                        for it.

        l1d_flush=      [X86,INTEL]
                        Control mitigation for L1D based snooping␣
↪vulnerability.

                        Certain CPUs are vulnerable to an exploit against CPU
                        internal buffers which can forward information to a
                        disclosure gadget under certain conditions.

                        In vulnerable processors, the speculatively
                        forwarded data can be used in a cache side channel
                        attack, to access data to which the attacker does
                        not have direct access.

                        This parameter controls the mitigation. The
                        options are:

                        on         - enable the interface for the mitigation

        l1tf=           [X86] Control mitigation of the L1TF vulnerability on
                              affected CPUs

                        The kernel PTE inversion protection is unconditionally
                        enabled and cannot be disabled.

                        full
                                Provides all available mitigations for the
                                L1TF vulnerability. Disables SMT and
                                enables all mitigations in the
                                hypervisors, i.e. unconditional L1D flush.

                                SMT control and L1D flush control via the
                                sysfs interface is still possible after
                                boot.  Hypervisors will issue a warning
                                when the first VM is started in a
                                potentially insecure configuration,
                                i.e. SMT enabled or L1D flush disabled.

                        full,force
                                Same as 'full', but disables SMT and L1D
                                flush runtime control. Implies the
                                'nosmt=force' command line option.
                                (i.e. sysfs control of SMT is disabled.)

                        flush
```

```
                                Leaves SMT enabled and enables the default
                                hypervisor mitigation, i.e. conditional
                                L1D flush.

                                SMT control and L1D flush control via the
                                sysfs interface is still possible after
                                boot.  Hypervisors will issue a warning
                                when the first VM is started in a
                                potentially insecure configuration,
                                i.e. SMT enabled or L1D flush disabled.

                        flush,nosmt

                                Disables SMT and enables the default
                                hypervisor mitigation.

                                SMT control and L1D flush control via the
                                sysfs interface is still possible after
                                boot.  Hypervisors will issue a warning
                                when the first VM is started in a
                                potentially insecure configuration,
                                i.e. SMT enabled or L1D flush disabled.

                        flush,nowarn
                                Same as 'flush', but hypervisors will not
                                warn when a VM is started in a potentially
                                insecure configuration.

                        off
                                Disables hypervisor mitigations and doesn't
                                emit any warnings.
                                It also drops the swap size and available
                                RAM limit restriction on both hypervisor and
                                bare metal.

                        Default is 'flush'.

                        For details see: Documentation/admin-guide/hw-vuln/
↪l1tf.rst

        l2cr=           [PPC]

        l3cr=           [PPC]

        lapic           [X86-32,APIC] Enable the local APIC even if BIOS
                        disabled it.

        lapic=          [X86,APIC] Do not use TSC deadline
                        value for LAPIC timer one-shot implementation. Default
                        back to the programmable timer unit in the LAPIC.
                        Format: notscdeadline
```

```
        lapic_timer_c2_ok       [X86,APIC] trust the local apic timer
                                in C2 power state.

        libata.dma=     [LIBATA] DMA control
                        libata.dma=0      Disable all PATA and SATA DMA
                        libata.dma=1      PATA and SATA Disk DMA only
                        libata.dma=2      ATAPI (CDROM) DMA only
                        libata.dma=4      Compact Flash DMA only
                        Combinations also work, so libata.dma=3 enables DMA
                        for disks and CDROMs, but not CFs.

        libata.ignore_hpa=      [LIBATA] Ignore HPA limit
                        libata.ignore_hpa=0       keep BIOS limits (default)
                        libata.ignore_hpa=1       ignore limits, using full␣
↪disk

        libata.noacpi   [LIBATA] Disables use of ACPI in libata suspend/resume
                        when set.
                        Format: <int>

        libata.force=   [LIBATA] Force configurations.  The format is a comma-
                        separated list of "[ID:]VAL" where ID is PORT[.DEVICE].
                        PORT and DEVICE are decimal numbers matching port, link
                        or device.  Basically, it matches the ATA ID string
                        printed on console by libata.  If the whole ID part is
                        omitted, the last PORT and DEVICE values are used.  If
                        ID hasn't been specified yet, the configuration applies
                        to all ports, links and devices.

                        If only DEVICE is omitted, the parameter applies to
                        the port and all links and devices behind it.  DEVICE
                        number of 0 either selects the first device or the
                        first fan-out link behind PMP device.  It does not
                        select the host link.  DEVICE number of 15 selects the
                        host link and device attached to it.

                        The VAL specifies the configuration to force.  As long
                        as there is no ambiguity, shortcut notation is allowed.
                        For example, both 1.5 and 1.5G would work for 1.5Gbps.
                        The following configurations can be forced.

                        * Cable type: 40c, 80c, short40c, unk, ign or sata.
                          Any ID with matching PORT is used.

                        * SATA link speed limit: 1.5Gbps or 3.0Gbps.

                        * Transfer mode: pio[0-7], mwdma[0-4] and udma[0-7].
                          udma[/][16,25,33,44,66,100,133] notation is also
                          allowed.
```

```
                        * nohrst, nosrst, norst: suppress hard, soft and both
                          resets.

                        * rstonce: only attempt one reset during hot-unplug
                          link recovery.

                        * [no]dbdelay: Enable or disable the extra 200ms delay
                          before debouncing a link PHY and device presence
                          detection.

                        * [no]ncq: Turn on or off NCQ.

                        * [no]ncqtrim: Enable or disable queued DSM TRIM.

                        * [no]ncqati: Enable or disable NCQ trim on ATI␣
↪chipset.

                        * [no]trim: Enable or disable (unqueued) TRIM.

                        * trim_zero: Indicate that TRIM command zeroes data.

                        * max_trim_128m: Set 128M maximum trim size limit.

                        * [no]dma: Turn on or off DMA transfers.

                        * atapi_dmadir: Enable ATAPI DMADIR bridge support.

                        * atapi_mod16_dma: Enable the use of ATAPI DMA for
                          commands that are not a multiple of 16 bytes.

                        * [no]dmalog: Enable or disable the use of the
                          READ LOG DMA EXT command to access logs.

                        * [no]iddevlog: Enable or disable access to the
                          identify device data log.

                        * [no]logdir: Enable or disable access to the general
                          purpose log directory.

                        * max_sec_128: Set transfer size limit to 128 sectors.

                        * max_sec_1024: Set or clear transfer size limit to
                          1024 sectors.

                        * max_sec_lba48: Set or clear transfer size limit to
                          65535 sectors.

                        * [no]lpm: Enable or disable link power management.

                        * [no]setxfer: Indicate if transfer speed mode setting
                          should be skipped.
```

```
                  * [no]fua: Disable or enable FUA (Force Unit Access)
                    support for devices supporting this feature.

                  * dump_id: Dump IDENTIFY data.

                  * disable: Disable this device.

                  If there are multiple matching configurations changing
                  the same attribute, the last one is used.

    load_ramdisk=   [RAM] [Deprecated]

    lockd.nlm_grace_period=P  [NFS] Assign grace period.
                  Format: <integer>

    lockd.nlm_tcpport=N     [NFS] Assign TCP port.
                  Format: <integer>

    lockd.nlm_timeout=T     [NFS] Assign timeout value.
                  Format: <integer>

    lockd.nlm_udpport=M     [NFS] Assign UDP port.
                  Format: <integer>

    lockdown=       [SECURITY]
                  { integrity | confidentiality }
                  Enable the kernel lockdown feature. If set to
                  integrity, kernel features that allow userland to
                  modify the running kernel are disabled. If set to
                  confidentiality, kernel features that allow userland
                  to extract confidential information from the kernel
                  are also disabled.

    locktorture.acq_writer_lim= [KNL]
                  Set the time limit in jiffies for a lock
                  acquisition.  Acquisitions exceeding this limit
                  will result in a splat once they do complete.

    locktorture.bind_readers= [KNL]
                  Specify the list of CPUs to which the readers are
                  to be bound.

    locktorture.bind_writers= [KNL]
                  Specify the list of CPUs to which the writers are
                  to be bound.

    locktorture.call_rcu_chains= [KNL]
                  Specify the number of self-propagating call_rcu()
                  chains to set up.  These are used to ensure that
                  there is a high probability of an RCU grace period
```

```
                in progress at any given time.  Defaults to 0,
                which disables these call_rcu() chains.

        locktorture.long_hold= [KNL]
                Specify the duration in milliseconds for the
                occasional long-duration lock hold time.  Defaults
                to 100 milliseconds.  Select 0 to disable.

        locktorture.nested_locks= [KNL]
                Specify the maximum lock nesting depth that
                locktorture is to exercise, up to a limit of 8
                (MAX_NESTED_LOCKS).  Specify zero to disable.
                Note that this parameter is ineffective on types
                of locks that do not support nested acquisition.

        locktorture.nreaders_stress= [KNL]
                Set the number of locking read-acquisition kthreads.
                Defaults to being automatically set based on the
                number of online CPUs.

        locktorture.nwriters_stress= [KNL]
                Set the number of locking write-acquisition kthreads.

        locktorture.onoff_holdoff= [KNL]
                Set time (s) after boot for CPU-hotplug testing.

        locktorture.onoff_interval= [KNL]
                Set time (s) between CPU-hotplug operations, or
                zero to disable CPU-hotplug testing.

        locktorture.rt_boost= [KNL]
                Do periodic testing of real-time lock priority
                boosting.  Select 0 to disable, 1 to boost
                only rt_mutex, and 2 to boost unconditionally.
                Defaults to 2, which might seem to be an
                odd choice, but which should be harmless for
                non-real-time spinlocks, due to their disabling
                of preemption.  Note that non-realtime mutexes
                disable boosting.

        locktorture.rt_boost_factor= [KNL]
                Number that determines how often and for how
                long priority boosting is exercised.  This is
                scaled down by the number of writers, so that the
                number of boosts per unit time remains roughly
                constant as the number of writers increases.
                On the other hand, the duration of each boost
                increases with the number of writers.

        locktorture.shuffle_interval= [KNL]
                Set task-shuffle interval (jiffies).  Shuffling
```

```
                        tasks allows some CPUs to go into dyntick-idle
                        mode during the locktorture test.

        locktorture.shutdown_secs= [KNL]
                        Set time (s) after boot system shutdown.  This
                        is useful for hands-off automated testing.

        locktorture.stat_interval= [KNL]
                        Time (s) between statistics printk()s.

        locktorture.stutter= [KNL]
                        Time (s) to stutter testing, for example,
                        specifying five seconds causes the test to run for
                        five seconds, wait for five seconds, and so on.
                        This tests the locking primitive's ability to
                        transition abruptly to and from idle.

        locktorture.torture_type= [KNL]
                        Specify the locking implementation to test.

        locktorture.verbose= [KNL]
                        Enable additional printk() statements.

        locktorture.writer_fifo= [KNL]
                        Run the write-side locktorture kthreads at
                        sched_set_fifo() real-time priority.

        logibm.irq=     [HW,MOUSE] Logitech Bus Mouse Driver
                        Format: <irq>

        loglevel=       All Kernel Messages with a loglevel smaller than the
                        console loglevel will be printed to the console. It can
                        also be changed with klogd or other programs. The
                        loglevels are defined as follows:

                        0 (KERN_EMERG)          system is unusable
                        1 (KERN_ALERT)          action must be taken␣
↪immediately
                        2 (KERN_CRIT)           critical conditions
                        3 (KERN_ERR)            error conditions
                        4 (KERN_WARNING)        warning conditions
                        5 (KERN_NOTICE)         normal but significant␣
↪condition
                        6 (KERN_INFO)           informational
                        7 (KERN_DEBUG)          debug-level messages

        log_buf_len=n[KMG]      Sets the size of the printk ring buffer,
                        in bytes.  n must be a power of two and greater
                        than the minimal size. The minimal size is defined
                        by LOG_BUF_SHIFT kernel config parameter. There is
                        also CONFIG_LOG_CPU_MAX_BUF_SHIFT config parameter
```

```
                        that allows to increase the default size depending on
                        the number of CPUs. See init/Kconfig for more details.

        logo.nologo     [FB] Disables display of the built-in Linux logo.
                        This may be used to provide more screen space for
                        kernel log messages and is useful when debugging
                        kernel boot problems.

        lp=0            [LP]    Specify parallel ports to use, e.g,
        lp=port[,port...]       lp=none,parport0 (lp0 not configured, lp1 uses
        lp=reset                first parallel port). 'lp=0' disables the
        lp=auto                 printer driver. 'lp=reset' (which can be
                                specified in addition to the ports) causes
                                attached printers to be reset. Using
                                lp=port1,port2,... specifies the parallel ports
                                to associate lp devices with, starting with
                                lp0. A port specification may be 'none' to skip
                                that lp device, or a parport name such as
                                'parport0'. Specifying 'lp=auto' instead of a
                                port specification list means that device IDs
                                from each port should be examined, to see if
                                an IEEE 1284-compliant printer is attached; if
                                so, the driver will manage that printer.
                                See also header of drivers/char/lp.c.

        lpj=n           [KNL]
                        Sets loops_per_jiffy to given constant, thus avoiding
                        time-consuming boot-time autodetection (up to 250 ms␣
↪per
                        CPU). 0 enables autodetection (default). To determine
                        the correct value for your kernel, boot with normal
                        autodetection and see what value is printed. Note that
                        on SMP systems the preset will be applied to all CPUs,
                        which is likely to cause problems if your CPUs need
                        significantly divergent settings. An incorrect value
                        will cause delays in the kernel to be wrong, leading to
                        unpredictable I/O errors and other breakage. Although
                        unlikely, in the extreme case this might damage your
                        hardware.

        ltpc=           [NET]
                        Format: <io>,<irq>,<dma>

        lsm.debug       [SECURITY] Enable LSM initialization debugging output.

        lsm=lsm1,...,lsmN
                        [SECURITY] Choose order of LSM initialization. This
                        overrides CONFIG_LSM, and the "security=" parameter.

        machvec=        [IA-64] Force the use of a particular machine-vector
                        (machvec) in a generic kernel.
```

```
                        Example: machvec=hpzx1

        machtype=       [Loongson] Share the same kernel image file between
                        different yeeloong laptops.
                        Example: machtype=lemote-yeeloong-2f-7inch

        max_addr=nn[KMG]        [KNL,BOOT,IA-64] All physical memory greater
                        than or equal to this physical address is ignored.

        maxcpus=        [SMP] Maximum number of processors that an SMP kernel
                        will bring up during bootup.  maxcpus=n : n >= 0 limits
                        the kernel to bring up 'n' processors. Surely after
                        bootup you can bring up the other plugged cpu by␣
↪executing
                        "echo 1 > /sys/devices/system/cpu/cpuX/online". So␣
↪maxcpus
                        only takes effect during system bootup.
                        While n=0 is a special case, it is equivalent to␣
↪"nosmp",
                        which also disables the IO APIC.

        max_loop=       [LOOP] The number of loop block devices that get
        (loop.max_loop) unconditionally pre-created at init time. The default
                        number is configured by BLK_DEV_LOOP_MIN_COUNT. Instead
                        of statically allocating a predefined number, loop
                        devices can be requested on-demand with the
                        /dev/loop-control interface.

        mce             [X86-32] Machine Check Exception

        mce=option      [X86-64] See Documentation/arch/x86/x86_64/
↪boot-options.rst

        md=             [HW] RAID subsystems devices and level
                        See Documentation/admin-guide/md.rst.

        mdacon=         [MDA]
                        Format: <first>,<last>
                        Specifies range of consoles to be captured by the MDA.

        mds=            [X86,INTEL]
                        Control mitigation for the Micro-architectural Data
                        Sampling (MDS) vulnerability.

                        Certain CPUs are vulnerable to an exploit against CPU
                        internal buffers which can forward information to a
                        disclosure gadget under certain conditions.

                        In vulnerable processors, the speculatively
                        forwarded data can be used in a cache side channel
                        attack, to access data to which the attacker does
```

```
                        not have direct access.

                        This parameter controls the MDS mitigation. The
                        options are:

                        full       - Enable MDS mitigation on vulnerable CPUs
                        full,nosmt - Enable MDS mitigation and disable
                                     SMT on vulnerable CPUs
                        off        - Unconditionally disable MDS mitigation

                        On TAA-affected machines, mds=off can be prevented by
                        an active TAA mitigation as both vulnerabilities are
                        mitigated with the same mechanism so in order to␣
↪disable
                        this mitigation, you need to specify tsx_async_
↪abort=off
                        too.

                        Not specifying this option is equivalent to
                        mds=full.

                        For details see: Documentation/admin-guide/hw-vuln/mds.
↪rst

       mem=nn[KMG]      [HEXAGON] Set the memory size.
                        Must be specified, otherwise memory size will be 0.

       mem=nn[KMG]      [KNL,BOOT] Force usage of a specific amount of memory
                        Amount of memory to be used in cases as follows:

                        1 for test;
                        2 when the kernel is not able to see the whole system␣
↪memory;
                        3 memory that lies after 'mem=' boundary is excluded␣
↪from
                         the hypervisor, then assigned to KVM guests.
                        4 to limit the memory available for kdump kernel.

                        [ARC,MICROBLAZE] - the limit applies only to low␣
↪memory,
                        high memory is not affected.

                        [ARM64] - only limits memory covered by the linear
                        mapping. The NOMAP regions are not affected.

                        [X86] Work as limiting max address. Use together
                        with memmap= to avoid physical address space␣
↪collisions.
                        Without memmap= PCI devices could be placed at␣
↪addresses
                        belonging to unused RAM.
```

```
                        Note that this only takes effects during boot time␣
↪since
                        in above case 3, memory may need be hot added after␣
↪boot
                        if system memory of hypervisor is not sufficient.

        mem=nn[KMG]@ss[KMG]
                        [ARM,MIPS] - override the memory layout reported by
                        firmware.
                        Define a memory region of size nn[KMG] starting at
                        ss[KMG].
                        Multiple different regions can be specified with
                        multiple mem= parameters on the command line.

        mem=nopentium   [BUGS=X86-32] Disable usage of 4MB pages for kernel
                        memory.

        memblock=debug  [KNL] Enable memblock debug messages.

        memchunk=nn[KMG]
                        [KNL,SH] Allow user to override the default size for
                        per-device physically contiguous DMA buffers.

        memhp_default_state=online/offline/online_kernel/online_movable
                        [KNL] Set the initial state for the memory hotplug
                        onlining policy. If not specified, the default value is
                        set according to the
                        CONFIG_MEMORY_HOTPLUG_DEFAULT_ONLINE kernel config
                        option.
                        See Documentation/admin-guide/mm/memory-hotplug.rst.

        memmap=exactmap [KNL,X86] Enable setting of an exact
                        E820 memory map, as specified by the user.
                        Such memmap=exactmap lines can be constructed based on
                        BIOS output or other requirements. See the memmap=nn@ss
                        option description.

        memmap=nn[KMG]@ss[KMG]
                        [KNL, X86, MIPS, XTENSA] Force usage of a specific␣
↪region of memory.
                        Region of memory to be used is from ss to ss+nn.
                        If @ss[KMG] is omitted, it is equivalent to␣
↪mem=nn[KMG],
                        which limits max address to nn[KMG].
                        Multiple different regions can be specified,
                        comma delimited.
                        Example:
                                memmap=100M@2G,100M#3G,1G!1024G

        memmap=nn[KMG]#ss[KMG]
```

```
                        [KNL,ACPI] Mark specific memory as ACPI data.
                        Region of memory to be marked is from ss to ss+nn.

        memmap=nn[KMG]$ss[KMG]
                        [KNL,ACPI] Mark specific memory as reserved.
                        Region of memory to be reserved is from ss to ss+nn.
                        Example: Exclude memory from 0x18690000-0x1869ffff
                                 memmap=64K$0x18690000
                                 or
                                 memmap=0x10000$0x18690000
                        Some bootloaders may need an escape character before '
↪$',
                        like Grub2, otherwise '$' and the following number
                        will be eaten.

        memmap=nn[KMG]!ss[KMG]
                        [KNL,X86] Mark specific memory as protected.
                        Region of memory to be used, from ss to ss+nn.
                        The memory region may be marked as e820 type 12 (0xc)
                        and is NVDIMM or ADR memory.

        memmap=<size>%<offset>-<oldtype>+<newtype>
                        [KNL,ACPI] Convert memory within the specified region
                        from <oldtype> to <newtype>. If "-<oldtype>" is left
                        out, the whole region will be marked as <newtype>,
                        even if previously unavailable. If "+<newtype>" is left
                        out, matching memory will be removed. Types are
                        specified as e820 types, e.g., 1 = RAM, 2 = reserved,
                        3 = ACPI, 12 = PRAM.

        memory_corruption_check=0/1 [X86]
                        Some BIOSes seem to corrupt the first 64k of
                        memory when doing things like suspend/resume.
                        Setting this option will scan the memory
                        looking for corruption.  Enabling this will
                        both detect corruption and prevent the kernel
                        from using the memory being corrupted.
                        However, its intended as a diagnostic tool; if
                        repeatable BIOS-originated corruption always
                        affects the same memory, you can use memmap=
                        to prevent the kernel from using that memory.

        memory_corruption_check_size=size [X86]
                        By default it checks for corruption in the low
                        64k, making this memory unavailable for normal
                        use.  Use this parameter to scan for
                        corruption in more or less memory.

        memory_corruption_check_period=seconds [X86]
                        By default it checks for corruption every 60
                        seconds.  Use this parameter to check at some
```

```
                        other rate.  0 disables periodic checking.

        memory_hotplug.memmap_on_memory
                        [KNL,X86,ARM] Boolean flag to enable this feature.
                        Format: {on | off (default)}
                        When enabled, runtime hotplugged memory will
                        allocate its internal metadata (struct pages,
                        those vmemmap pages cannot be optimized even
                        if hugetlb_free_vmemmap is enabled) from the
                        hotadded memory which will allow to hotadd a
                        lot of memory without requiring additional
                        memory to do so.
                        This feature is disabled by default because it
                        has some implication on large (e.g. GB)
                        allocations in some configurations (e.g. small
                        memory blocks).
                        The state of the flag can be read in
                        /sys/module/memory_hotplug/parameters/memmap_on_memory.
                        Note that even when enabled, there are a few cases␣
↪where
                        the feature is not effective.

        memtest=        [KNL,X86,ARM,M68K,PPC,RISCV] Enable memtest
                        Format: <integer>
                        default : 0 <disable>
                        Specifies the number of memtest passes to be
                        performed. Each pass selects another test
                        pattern from a given set of patterns. Memtest
                        fills the memory with this pattern, validates
                        memory contents and reserves bad memory
                        regions that are detected.

        mem_encrypt=    [X86-64] AMD Secure Memory Encryption (SME) control
                        Valid arguments: on, off
                        Default (depends on kernel configuration option):
                          on  (CONFIG_AMD_MEM_ENCRYPT_ACTIVE_BY_DEFAULT=y)
                          off (CONFIG_AMD_MEM_ENCRYPT_ACTIVE_BY_DEFAULT=n)
                        mem_encrypt=on:         Activate SME
                        mem_encrypt=off:        Do not activate SME

                        Refer to Documentation/virt/kvm/x86/
↪amd-memory-encryption.rst
                        for details on when memory encryption can be activated.

        mem_sleep_default=      [SUSPEND] Default system suspend mode:
                        s2idle  - Suspend-To-Idle
                        shallow - Power-On Suspend or equivalent (if supported)
                        deep    - Suspend-To-RAM or equivalent (if supported)
                        See Documentation/admin-guide/pm/sleep-states.rst.

        mfgpt_irq=      [IA-32] Specify the IRQ to use for the
```

```
                Multi-Function General Purpose Timers on AMD Geode
                platforms.

mfgptfix        [X86-32] Fix MFGPT timers on AMD Geode platforms when
                the BIOS has incorrectly applied a workaround. TinyBIOS
                version 0.98 is known to be affected, 0.99 fixes the
                problem by letting the user disable the workaround.

mga=            [HW,DRM]

microcode.force_minrev= [X86]
                Format: <bool>
                Enable or disable the microcode minimal revision
                enforcement for the runtime microcode loader.

min_addr=nn[KMG]        [KNL,BOOT,IA-64] All physical memory below this
                physical address is ignored.

mini2440=       [ARM,HW,KNL]
                Format:[0..2][b][c][t]
                Default: "0tb"
                MINI2440 configuration specification:
                0 - The attached screen is the 3.5" TFT
                1 - The attached screen is the 7" TFT
                2 - The VGA Shield is attached (1024x768)
                Leaving out the screen size parameter will not load
                the TFT driver, and the framebuffer will be left
                unconfigured.
                b - Enable backlight. The TFT backlight pin will be
                linked to the kernel VESA blanking code and a GPIO
                LED. This parameter is not necessary when using the
                VGA shield.
                c - Enable the s3c camera interface.
                t - Reserved for enabling touchscreen support. The
                touchscreen support is not enabled in the mainstream
                kernel as of 2.6.30, a preliminary port can be found
                in the "bleeding edge" mini2440 support kernel at
                https://repo.or.cz/w/linux-2.6/mini2440.git

mitigations=
                [X86,PPC,S390,ARM64] Control optional mitigations for
                CPU vulnerabilities.  This is a set of curated,
                arch-independent options, each of which is an
                aggregation of existing arch-specific options.

                off
                        Disable all optional CPU mitigations.  This
                        improves system performance, but it may also
                        expose users to several CPU vulnerabilities.
                        Equivalent to: if nokaslr then kpti=0 [ARM64]
                                       gather_data_sampling=off [X86]
```

```
                                                kvm.nx_huge_pages=off [X86]
                                                l1tf=off [X86]
                                                mds=off [X86]
                                                mmio_stale_data=off [X86]
                                                no_entry_flush [PPC]
                                                no_uaccess_flush [PPC]
                                                nobp=0 [S390]
                                                nopti [X86,PPC]
                                                nospectre_bhb [ARM64]
                                                nospectre_v1 [X86,PPC]
                                                nospectre_v2 [X86,PPC,S390,
↪ARM64]
                                                retbleed=off [X86]
                                                spec_store_bypass_disable=off␣
↪[X86,PPC]
                                                spectre_v2_user=off [X86]
                                                srbds=off [X86,INTEL]
                                                ssbd=force-off [ARM64]
                                                tsx_async_abort=off [X86]

                                        Exceptions:
                                                This does not have any effect on
                                                kvm.nx_huge_pages when
                                                kvm.nx_huge_pages=force.

                                auto (default)
                                        Mitigate all CPU vulnerabilities, but leave SMT
                                        enabled, even if it's vulnerable.  This is for
                                        users who don't want to be surprised by SMT
                                        getting disabled across kernel upgrades, or who
                                        have other ways of avoiding SMT-based attacks.
                                        Equivalent to: (default behavior)

                                auto,nosmt
                                        Mitigate all CPU vulnerabilities, disabling SMT
                                        if needed.  This is for users who always want␣
↪to
                                        be fully mitigated, even if it means losing␣
↪SMT.
                                        Equivalent to: l1tf=flush,nosmt [X86]
                                                mds=full,nosmt [X86]
                                                tsx_async_abort=full,nosmt [X86]
                                                mmio_stale_data=full,nosmt [X86]
                                                retbleed=auto,nosmt [X86]

        mminit_loglevel=
                                [KNL] When CONFIG_DEBUG_MEMORY_INIT is set, this
                                parameter allows control of the logging verbosity for
                                the additional memory initialisation checks. A value
                                of 0 disables mminit logging and a level of 4 will
                                log everything. Information is printed at KERN_DEBUG
```

```
                        so loglevel=8 may also need to be specified.

        mmio_stale_data=
                        [X86,INTEL] Control mitigation for the Processor
                        MMIO Stale Data vulnerabilities.

                        Processor MMIO Stale Data is a class of
                        vulnerabilities that may expose data after an MMIO
                        operation. Exposed data could originate or end in
                        the same CPU buffers as affected by MDS and TAA.
                        Therefore, similar to MDS and TAA, the mitigation
                        is to clear the affected CPU buffers.

                        This parameter controls the mitigation. The
                        options are:

                        full       - Enable mitigation on vulnerable CPUs

                        full,nosmt - Enable mitigation and disable SMT on
                                     vulnerable CPUs.

                        off        - Unconditionally disable mitigation

                        On MDS or TAA affected machines,
                        mmio_stale_data=off can be prevented by an active
                        MDS or TAA mitigation as these vulnerabilities are
                        mitigated with the same mechanism so in order to
                        disable this mitigation, you need to specify
                        mds=off and tsx_async_abort=off too.

                        Not specifying this option is equivalent to
                        mmio_stale_data=full.

                        For details see:
                        Documentation/admin-guide/hw-vuln/processor_mmio_stale_
↪data.rst

        <module>.async_probe[=<bool>] [KNL]
                        If no <bool> value is specified or if the value
                        specified is not a valid <bool>, enable asynchronous
                        probe on this module.  Otherwise, enable/disable
                        asynchronous probe on this module as indicated by the
                        <bool> value. See also: module.async_probe

        module.async_probe=<bool>
                        [KNL] When set to true, modules will use async probing
                        by default. To enable/disable async probing for a
                        specific module, use the module specific control that
                        is documented under <module>.async_probe. When both
                        module.async_probe and <module>.async_probe are
                        specified, <module>.async_probe takes precedence for
```

```
                        the specific module.

        module.enable_dups_trace
                        [KNL] When CONFIG_MODULE_DEBUG_AUTOLOAD_DUPS is set,
                        this means that duplicate request_module() calls will
                        trigger a WARN_ON() instead of a pr_warn(). Note that
                        if MODULE_DEBUG_AUTOLOAD_DUPS_TRACE is set, WARN_ON()s
                        will always be issued and this option does nothing.
        module.sig_enforce
                        [KNL] When CONFIG_MODULE_SIG is set, this means that
                        modules without (valid) signatures will fail to load.
                        Note that if CONFIG_MODULE_SIG_FORCE is set, that
                        is always true, so this option does nothing.

        module_blacklist=  [KNL] Do not load a comma-separated list of
                        modules.  Useful for debugging problem modules.

        mousedev.tap_time=
                        [MOUSE] Maximum time between finger touching and
                        leaving touchpad surface for touch to be considered
                        a tap and be reported as a left button click (for
                        touchpads working in absolute mode only).
                        Format: <msecs>
        mousedev.xres=  [MOUSE] Horizontal screen resolution, used for devices
                        reporting absolute coordinates, such as tablets
        mousedev.yres=  [MOUSE] Vertical screen resolution, used for devices
                        reporting absolute coordinates, such as tablets

        movablecore=    [KNL,X86,IA-64,PPC]
                        Format: nn[KMGTPE] | nn%
                        This parameter is the complement to kernelcore=, it
                        specifies the amount of memory used for migratable
                        allocations.  If both kernelcore and movablecore is
                        specified, then kernelcore will be at *least* the
                        specified value but may be more.  If movablecore on its
                        own is specified, the administrator must be careful
                        that the amount of memory usable for all allocations
                        is not too small.

        movable_node    [KNL] Boot-time switch to make hotplugable memory
                        NUMA nodes to be movable. This means that the memory
                        of such nodes will be usable only for movable
                        allocations which rules out almost all kernel
                        allocations. Use with caution!

        MTD_Partition=  [MTD]
                        Format: <name>,<region-number>,<size>,<offset>

        MTD_Region=     [MTD] Format:
                        <name>,<region-number>[,<base>,<size>,<buswidth>,
↪<altbuswidth>]
```

```
mtdparts=       [MTD]
                See drivers/mtd/parsers/cmdlinepart.c

mtdset=         [ARM]
                ARM/S3C2412 JIVE boot control

                See arch/arm/mach-s3c/mach-jive.c

mtouchusb.raw_coordinates=
                [HW] Make the MicroTouch USB driver use raw coordinates
                ('y', default) or cooked coordinates ('n')

mtrr=debug      [X86]
                Enable printing debug information related to MTRR
                registers at boot time.

mtrr_chunk_size=nn[KMG] [X86]
                used for mtrr cleanup. It is largest continuous chunk
                that could hold holes aka. UC entries.

mtrr_gran_size=nn[KMG] [X86]
                Used for mtrr cleanup. It is granularity of mtrr block.
                Default is 1.
                Large value could prevent small alignment from
                using up MTRRs.

mtrr_spare_reg_nr=n [X86]
                Format: <integer>
                Range: 0,7 : spare reg number
                Default : 1
                Used for mtrr cleanup. It is spare mtrr entries number.
                Set to 2 or more if your graphical card needs more.

multitce=off    [PPC]  This parameter disables the use of the pSeries
                firmware feature for updating multiple TCE entries
                at a time.

n2=             [NET] SDL Inc. RISCom/N2 synchronous serial card

netdev=         [NET] Network devices parameters
                Format: <irq>,<io>,<mem_start>,<mem_end>,<name>
                Note that mem_start is often overloaded to mean
                something different and driver-specific.
                This usage is only documented in each driver source
                file if at all.

netpoll.carrier_timeout=
                [NET] Specifies amount of time (in seconds) that
                netpoll should wait for a carrier. By default netpoll
                waits 4 seconds.
```

```
nf_conntrack.acct=
                [NETFILTER] Enable connection tracking flow accounting
                0 to disable accounting
                1 to enable accounting
                Default value is 0.

nfs.cache_getent=
                [NFS] sets the pathname to the program which is used
                to update the NFS client cache entries.

nfs.cache_getent_timeout=
                [NFS] sets the timeout after which an attempt to
                update a cache entry is deemed to have failed.

nfs.callback_nr_threads=
                [NFSv4] set the total number of threads that the
                NFS client will assign to service NFSv4 callback
                requests.

nfs.callback_tcpport=
                [NFS] set the TCP port on which the NFSv4 callback
                channel should listen.

nfs.delay_retrans=
                [NFS] specifies the number of times the NFSv4 client
                retries the request before returning an EAGAIN error,
                after a reply of NFS4ERR_DELAY from the server.
                Only applies if the softerr mount option is enabled,
                and the specified value is >= 0.

nfs.enable_ino64=
                [NFS] enable 64-bit inode numbers.
                If zero, the NFS client will fake up a 32-bit inode
                number for the readdir() and stat() syscalls instead
                of returning the full 64-bit number.
                The default is to return 64-bit inode numbers.

nfs.idmap_cache_timeout=
                [NFS] set the maximum lifetime for idmapper cache
                entries.

nfs.max_session_cb_slots=
                [NFSv4.1] Sets the maximum number of session
                slots the client will assign to the callback
                channel. This determines the maximum number of
                callbacks the client will process in parallel for
                a particular server.

nfs.max_session_slots=
                [NFSv4.1] Sets the maximum number of session slots
```

```
                        the client will attempt to negotiate with the server.
                        This limits the number of simultaneous RPC requests
                        that the client can send to the NFSv4.1 server.
                        Note that there is little point in setting this
                        value higher than the max_tcp_slot_table_limit.

        nfs.nfs4_disable_idmapping=
                        [NFSv4] When set to the default of '1', this option
                        ensures that both the RPC level authentication
                        scheme and the NFS level operations agree to use
                        numeric uids/gids if the mount is using the
                        'sec=sys' security flavour. In effect it is
                        disabling idmapping, which can make migration from
                        legacy NFSv2/v3 systems to NFSv4 easier.
                        Servers that do not support this mode of operation
                        will be autodetected by the client, and it will fall
                        back to using the idmapper.
                        To turn off this behaviour, set the value to '0'.

        nfs.nfs4_unique_id=
                        [NFS4] Specify an additional fixed unique ident-
                        ification string that NFSv4 clients can insert into
                        their nfs_client_id4 string.  This is typically a
                        UUID that is generated at system install time.

        nfs.recover_lost_locks=
                        [NFSv4] Attempt to recover locks that were lost due
                        to a lease timeout on the server. Please note that
                        doing this risks data corruption, since there are
                        no guarantees that the file will remain unchanged
                        after the locks are lost.
                        If you want to enable the kernel legacy behaviour of
                        attempting to recover these locks, then set this
                        parameter to '1'.
                        The default parameter value of '0' causes the kernel
                        not to attempt recovery of lost locks.

        nfs.send_implementation_id=
                        [NFSv4.1] Send client implementation identification
                        information in exchange_id requests.
                        If zero, no implementation identification information
                        will be sent.
                        The default is to send the implementation␣
↪identification
                        information.

        nfs4.layoutstats_timer=
                        [NFSv4.2] Change the rate at which the kernel sends
                        layoutstats to the pNFS metadata server.

                        Setting this to value to 0 causes the kernel to use
```

```
                whatever value is the default set by the layout
                driver. A non-zero value sets the minimum interval
                in seconds between layoutstats transmissions.

nfsd.inter_copy_offload_enable=
                [NFSv4.2] When set to 1, the server will support
                server-to-server copies for which this server is
                the destination of the copy.

nfsd.nfs4_disable_idmapping=
                [NFSv4] When set to the default of '1', the NFSv4
                server will return only numeric uids and gids to
                clients using auth_sys, and will accept numeric uids
                and gids from such clients.  This is intended to ease
                migration from NFSv2/v3.

nfsd.nfsd4_ssc_umount_timeout=
                [NFSv4.2] When used as the destination of a
                server-to-server copy, knfsd temporarily mounts
                the source server.  It caches the mount in case
                it will be needed again, and discards it if not
                used for the number of milliseconds specified by
                this parameter.

nfsaddrs=       [NFS] Deprecated.  Use ip= instead.
                See Documentation/admin-guide/nfs/nfsroot.rst.

nfsroot=        [NFS] nfs root filesystem for disk-less boxes.
                See Documentation/admin-guide/nfs/nfsroot.rst.

nfsrootdebug    [NFS] enable nfsroot debugging messages.
                See Documentation/admin-guide/nfs/nfsroot.rst.

nmi_backtrace.backtrace_idle [KNL]
                Dump stacks even of idle CPUs in response to an
                NMI stack-backtrace request.

nmi_debug=      [KNL,SH] Specify one or more actions to take
                when a NMI is triggered.
                Format: [state][,regs][,debounce][,die]

nmi_watchdog=   [KNL,BUGS=X86] Debugging features for SMP kernels
                Format: [panic,][nopanic,][num]
                Valid num: 0 or 1
                0 - turn hardlockup detector in nmi_watchdog off
                1 - turn hardlockup detector in nmi_watchdog on
                When panic is specified, panic when an NMI watchdog
                timeout occurs (or 'nopanic' to not panic on an NMI
                watchdog, if CONFIG_BOOTPARAM_HARDLOCKUP_PANIC is set)
                To disable both hard and soft lockup detectors,
                please see 'nowatchdog'.
```

```
                        This is useful when you use a panic=... timeout and
                        need the box quickly up again.

                        These settings can be accessed at runtime via
                        the nmi_watchdog and hardlockup_panic sysctls.

        no387           [BUGS=X86-32] Tells the kernel to use the 387 maths
                        emulation library even if a 387 maths coprocessor
                        is present.

        no4lvl          [RISCV] Disable 4-level and 5-level paging modes.␣
↪Forces
                        kernel to use 3-level paging instead.

        no5lvl          [X86-64,RISCV] Disable 5-level paging mode. Forces
                        kernel to use 4-level paging instead.

        noaliencache    [MM, NUMA, SLAB] Disables the allocation of alien
                        caches in the slab allocator.  Saves per-node memory,
                        but will impact performance.

        noalign         [KNL,ARM]

        noaltinstr      [S390] Disables alternative instructions patching
                        (CPU alternatives feature).

        noapic          [SMP,APIC] Tells the kernel to not make use of any
                        IOAPICs that may be present in the system.

        noautogroup     Disable scheduler automatic task group creation.

        nocache         [ARM]

        no_console_suspend
                        [HW] Never suspend the console
                        Disable suspending of consoles during suspend and
                        hibernate operations.  Once disabled, debugging
                        messages can reach various consoles while the rest
                        of the system is being put to sleep (ie, while
                        debugging driver suspend/resume hooks).  This may
                        not work reliably with all consoles, but is known
                        to work with serial and VGA consoles.
                        To facilitate more flexible debugging, we also add
                        console_suspend, a printk module parameter to control
                        it. Users could use console_suspend (usually
                        /sys/module/printk/parameters/console_suspend) to
                        turn on/off it dynamically.

        no_debug_objects
                        [KNL] Disable object debugging
```

```
        nodsp           [SH] Disable hardware DSP at boot time.

        noefi           Disable EFI runtime services support.

        no_entry_flush  [PPC] Don't flush the L1-D cache when entering the␣
↪kernel.

        noexec          [IA-64]

        noexec32        [X86-64]
                        This affects only 32-bit executables.
                        noexec32=on: enable non-executable mappings (default)
                                read doesn't imply executable mappings
                        noexec32=off: disable non-executable mappings
                                read implies executable mappings

        no_file_caps    Tells the kernel not to honor file capabilities.  The
                        only way then for a file to be executed with privilege
                        is to be setuid root or executed by root.

        nofpu           [MIPS,SH] Disable hardware FPU at boot time.

        nofsgsbase      [X86] Disables FSGSBASE instructions.

        nofxsr          [BUGS=X86-32] Disables x86 floating point extended
                        register save and restore. The kernel will only save
                        legacy floating-point registers on task switch.

        nohalt          [IA-64] Tells the kernel not to use the power saving
                        function PAL_HALT_LIGHT when idle. This increases
                        power-consumption. On the positive side, it reduces
                        interrupt wake-up latency, which may improve␣
↪performance
                        in certain environments such as networked servers or
                        real-time systems.

        no_hash_pointers
                        Force pointers printed to the console or buffers to be
                        unhashed.  By default, when a pointer is printed via %p
                        format string, that pointer is "hashed", i.e. obscured
                        by hashing the pointer value.  This is a security␣
↪feature
                        that hides actual kernel addresses from unprivileged
                        users, but it also makes debugging the kernel more
                        difficult since unequal pointers can no longer be
                        compared.  However, if this command-line option is
                        specified, then all normal pointers will have their␣
↪true
                        value printed. This option should only be specified␣
↪when
                        debugging the kernel.  Please do not use on production
```

```
                        kernels.

        nohibernate     [HIBERNATION] Disable hibernation and resume.

        nohlt           [ARM,ARM64,MICROBLAZE,MIPS,PPC,SH] Forces the kernel to
                        busy wait in do_idle() and not use the arch_cpu_idle()
                        implementation; requires CONFIG_GENERIC_IDLE_POLL_SETUP
                        to be effective. This is useful on platforms where the
                        sleep(SH) or wfi(ARM,ARM64) instructions do not work
                        correctly or when doing power measurements to evaluate
                        the impact of the sleep instructions. This is also
                        useful when using JTAG debugger.

        nohugeiomap     [KNL,X86,PPC,ARM64] Disable kernel huge I/O mappings.

        nohugevmalloc   [KNL,X86,PPC,ARM64] Disable kernel huge vmalloc␣
↪mappings.

        nohz=           [KNL] Boottime enable/disable dynamic ticks
                        Valid arguments: on, off
                        Default: on

        nohz_full=      [KNL,BOOT,SMP,ISOL]
                        The argument is a cpu list, as described above.
                        In kernels built with CONFIG_NO_HZ_FULL=y, set
                        the specified list of CPUs whose tick will be stopped
                        whenever possible. The boot CPU will be forced outside
                        the range to maintain the timekeeping.  Any CPUs
                        in this list will have their RCU callbacks offloaded,
                        just as if they had also been called out in the
                        rcu_nocbs= boot parameter.

                        Note that this argument takes precedence over
                        the CONFIG_RCU_NOCB_CPU_DEFAULT_ALL option.

        noinitrd        [RAM] Tells the kernel not to load any configured
                        initial RAM disk.

        nointremap      [X86-64, Intel-IOMMU] Do not enable interrupt
                        remapping.
                        [Deprecated - use intremap=off]

        nointroute      [IA-64]

        noinvpcid       [X86] Disable the INVPCID cpu feature.

        noiotrap        [SH] Disables trapped I/O port accesses.

        noirqdebug      [X86-32] Disables the code which attempts to detect and
                        disable unhandled interrupt sources.
```

```
noisapnp        [ISAPNP] Disables ISA PnP code.

nojitter        [IA-64] Disables jitter checking for ITC timers.

nokaslr         [KNL]
                When CONFIG_RANDOMIZE_BASE is set, this disables
                kernel and module base offset ASLR (Address Space
                Layout Randomization).

no-kvmapf       [X86,KVM] Disable paravirtualized asynchronous page
                fault handling.

no-kvmclock     [X86,KVM] Disable paravirtualized KVM clock driver

nolapic         [X86-32,APIC] Do not enable or use the local APIC.

nolapic_timer   [X86-32,APIC] Do not use the local APIC timer.

nomca           [IA-64] Disable machine check abort handling

nomce           [X86-32] Disable Machine Check Exception

nomfgpt         [X86-32] Disable Multi-Function General Purpose
                Timer usage (for AMD Geode machines).

nomodeset       Disable kernel modesetting. Most systems' firmware
                sets up a display mode and provides framebuffer memory
                for output. With nomodeset, DRM and fbdev drivers will
                not load if they could possibly displace the pre-
                initialized output. Only the system framebuffer will
                be available for use. The respective drivers will not
                perform display-mode changes or accelerated rendering.

                Useful as error fallback, or for testing and debugging.

nomodule        Disable module load

nonmi_ipi       [X86] Disable using NMI IPIs during panic/reboot to
                shutdown the other cpus.  Instead use the REBOOT_VECTOR
                irq.

nopat           [X86] Disable PAT (page attribute table extension of
                pagetables) support.

nopcid          [X86-64] Disable the PCID cpu feature.

nopku           [X86] Disable Memory Protection Keys CPU feature found
                in some Intel CPUs.

nopti           [X86-64]
                Equivalent to pti=off
```

```
        nopv=           [X86,XEN,KVM,HYPER_V,VMWARE]
                        Disables the PV optimizations forcing the guest to run
                        as generic guest with no PV drivers. Currently support
                        XEN HVM, KVM, HYPER_V and VMWARE guest.

        nopvspin        [X86,XEN,KVM]
                        Disables the qspinlock slow path using PV optimizations
                        which allow the hypervisor to 'idle' the guest on lock
                        contention.

        norandmaps      Don't use address space randomization.  Equivalent to
                        echo 0 > /proc/sys/kernel/randomize_va_space

        noreplace-smp   [X86-32,SMP] Don't replace SMP instructions
                        with UP alternatives

        noresume        [SWSUSP] Disables resume and restores original swap
                        space.

        nosbagart       [IA-64]

        no-scroll       [VGA] Disables scrollback.
                        This is required for the Braillex ib80-piezo Braille
                        reader made by F.H. Papenmeier (Germany).

        nosgx           [X86-64,SGX] Disables Intel SGX kernel support.

        nosmap          [PPC]
                        Disable SMAP (Supervisor Mode Access Prevention)
                        even if it is supported by processor.

        nosmep          [PPC64s]
                        Disable SMEP (Supervisor Mode Execution Prevention)
                        even if it is supported by processor.

        nosmp           [SMP] Tells an SMP kernel to act as a UP kernel,
                        and disable the IO APIC.  legacy for "maxcpus=0".

        nosmt           [KNL,MIPS,PPC,S390] Disable symmetric multithreading␣
↪(SMT).
                        Equivalent to smt=1.

                        [KNL,X86,PPC] Disable symmetric multithreading (SMT).
                        nosmt=force: Force disable SMT, cannot be undone
                                     via the sysfs control file.

        nosoftlockup    [KNL] Disable the soft-lockup detector.

        nospec_store_bypass_disable
                        [HW] Disable all mitigations for the Speculative Store␣
```

```
↪Bypass vulnerability

        nospectre_bhb   [ARM64] Disable all mitigations for Spectre-BHB (branch
                        history injection) vulnerability. System may allow␣
↪data leaks
                        with this option.

        nospectre_v1    [X86,PPC] Disable mitigations for Spectre Variant 1
                        (bounds check bypass). With this option data leaks are
                        possible in the system.

        nospectre_v2    [X86,PPC_E500,ARM64] Disable all mitigations for
                        the Spectre variant 2 (indirect branch prediction)
                        vulnerability. System may allow data leaks with this
                        option.

        no-steal-acc    [X86,PV_OPS,ARM64,PPC/PSERIES,RISCV] Disable
                        paravirtualized steal time accounting. steal time is
                        computed, but won't influence scheduler behaviour

        nosync          [HW,M68K] Disables sync negotiation for all devices.

        no_timer_check  [X86,APIC] Disables the code which tests for
                        broken timer IRQ sources.

        no_uaccess_flush
                        [PPC] Don't flush the L1-D cache after accessing user␣
↪data.

        novmcoredd      [KNL,KDUMP]
                        Disable device dump. Device dump allows drivers to
                        append dump data to vmcore so you can collect driver
                        specified debug info.  Drivers can append the data
                        without any limit and this data is stored in memory,
                        so this may cause significant memory stress.  Disabling
                        device dump can help save memory but the driver debug
                        data will be no longer available.  This parameter
                        is only available when CONFIG_PROC_VMCORE_DEVICE_DUMP
                        is set.

        no-vmw-sched-clock
                        [X86,PV_OPS] Disable paravirtualized VMware scheduler
                        clock and use the default one.

        nowatchdog      [KNL] Disable both lockup detectors, i.e.
                        soft-lockup and NMI watchdog (hard-lockup).

        nowb            [ARM]

        nox2apic        [X86-64,APIC] Do not enable x2APIC mode.
```

```
                        NOTE: this parameter will be ignored on systems with␣
↪the
                        LEGACY_XAPIC_DISABLED bit set in the
                        IA32_XAPIC_DISABLE_STATUS MSR.

        noxsave         [BUGS=X86] Disables x86 extended register state save
                        and restore using xsave. The kernel will fallback to
                        enabling legacy floating-point and sse state.

        noxsaveopt      [X86] Disables xsaveopt used in saving x86 extended
                        register states. The kernel will fall back to use
                        xsave to save the states. By using this parameter,
                        performance of saving the states is degraded because
                        xsave doesn't support modified optimization while
                        xsaveopt supports it on xsaveopt enabled systems.

        noxsaves        [X86] Disables xsaves and xrstors used in saving and
                        restoring x86 extended register state in compacted
                        form of xsave area. The kernel will fall back to use
                        xsaveopt and xrstor to save and restore the states
                        in standard form of xsave area. By using this
                        parameter, xsave area per process might occupy more
                        memory on xsaves enabled systems.

        nps_mtm_hs_ctr= [KNL,ARC]
                        This parameter sets the maximum duration, in
                        cycles, each HW thread of the CTOP can run
                        without interruptions, before HW switches it.
                        The actual maximum duration is 16 times this
                        parameter's value.
                        Format: integer between 1 and 255
                        Default: 255

        nptcg=          [IA-64] Override max number of concurrent global TLB
                        purges which is reported from either PAL_VM_SUMMARY or
                        SAL PALO.

        nr_cpus=        [SMP] Maximum number of processors that an SMP kernel
                        could support.  nr_cpus=n : n >= 1 limits the kernel to
                        support 'n' processors. It could be larger than the
                        number of already plugged CPU during bootup, later in
                        runtime you can physically add extra cpu until it␣
↪reaches
                        n. So during boot up some boot time memory for per-cpu
                        variables need be pre-allocated for later physical cpu
                        hot plugging.

        nr_uarts=       [SERIAL] maximum number of UARTs to be registered.

        numa=off        [KNL, ARM64, PPC, RISCV, SPARC, X86] Disable NUMA, Only
                        set up a single NUMA node spanning all memory.
```

```
        numa_balancing= [KNL,ARM64,PPC,RISCV,S390,X86] Enable or disable␣
↪automatic
                        NUMA balancing.
                        Allowed values are enable and disable

        numa_zonelist_order= [KNL, BOOT] Select zonelist order for NUMA.
                        'node', 'default' can be specified
                        This can be set from sysctl after boot.
                        See Documentation/admin-guide/sysctl/vm.rst for␣
↪details.

        ohci1394_dma=early      [HW] enable debugging via the ohci1394 driver.
                        See Documentation/core-api/debugging-via-ohci1394.rst␣
↪for more
                        info.

        olpc_ec_timeout= [OLPC] ms delay when issuing EC commands
                        Rather than timing out after 20 ms if an EC
                        command is not properly ACKed, override the length
                        of the timeout.  We have interrupts disabled while
                        waiting for the ACK, so if this is set too high
                        interrupts *may* be lost!

        omap_mux=       [OMAP] Override bootloader pin multiplexing.
                        Format: <mux_mode0.mode_name=value>...
                        For example, to override I2C bus2:
                        omap_mux=i2c2_scl.i2c2_scl=0x100,i2c2_sda.i2c2_
↪sda=0x100

        onenand.bdry=   [HW,MTD] Flex-OneNAND Boundary Configuration

                        Format: [die0_boundary][,die0_lock][,die1_boundary][,
↪die1_lock]

                        boundary - index of last SLC block on Flex-OneNAND.
                                   The remaining blocks are configured as MLC␣
↪blocks.
                        lock     - Configure if Flex-OneNAND boundary should␣
↪be locked.
                                   Once locked, the boundary cannot be changed.
                                   1 indicates lock status, 0 indicates unlock␣
↪status.

        oops=panic      Always panic on oopses. Default is to just kill the
                        process, but there is a small probability of
                        deadlocking the machine.
                        This will also cause panics on machine check␣
↪exceptions.
                        Useful together with panic=30 to trigger a reboot.
```

```
        page_alloc.shuffle=
                        [KNL] Boolean flag to control whether the page␣
↪allocator
                        should randomize its free lists. The randomization may
                        be automatically enabled if the kernel detects it is
                        running on a platform with a direct-mapped memory-side
                        cache, and this parameter can be used to
                        override/disable that behavior. The state of the flag
                        can be read from sysfs at:
                        /sys/module/page_alloc/parameters/shuffle.

        page_owner=     [KNL] Boot-time page_owner enabling option.
                        Storage of the information about who allocated
                        each page is disabled in default. With this switch,
                        we can turn it on.
                        on: enable the feature

        page_poison=    [KNL] Boot-time parameter changing the state of
                        poisoning on the buddy allocator, available with
                        CONFIG_PAGE_POISONING=y.
                        off: turn off poisoning (default)
                        on: turn on poisoning

        page_reporting.page_reporting_order=
                        [KNL] Minimal page reporting order
                        Format: <integer>
                        Adjust the minimal page reporting order. The page
                        reporting is disabled when it exceeds MAX_PAGE_ORDER.

        panic=          [KNL] Kernel behaviour on panic: delay <timeout>
                        timeout > 0: seconds before rebooting
                        timeout = 0: wait forever
                        timeout < 0: reboot immediately
                        Format: <timeout>

        panic_on_taint= Bitmask for conditionally calling panic() in add_
↪taint()
                        Format: <hex>[,nousertaint]
                        Hexadecimal bitmask representing the set of TAINT flags
                        that will cause the kernel to panic when add_taint() is
                        called with any of the flags in this set.
                        The optional switch "nousertaint" can be utilized to
                        prevent userspace forced crashes by writing to sysctl
                        /proc/sys/kernel/tainted any flagset matching with the
                        bitmask set on panic_on_taint.
                        See Documentation/admin-guide/tainted-kernels.rst for
                        extra details on the taint flags that users can pick
                        to compose the bitmask to assign to panic_on_taint.

        panic_on_warn=1 panic() instead of WARN().  Useful to cause kdump
                        on a WARN().
```

```
        panic_print=    Bitmask for printing system info when panic happens.
                        User can chose combination of the following bits:
                        bit 0: print all tasks info
                        bit 1: print system memory info
                        bit 2: print timer info
                        bit 3: print locks info if CONFIG_LOCKDEP is on
                        bit 4: print ftrace buffer
                        bit 5: print all printk messages in buffer
                        bit 6: print all CPUs backtrace (if available in the␣
↪arch)
                        *Be aware* that this option may print a _lot_ of lines,
                        so there are risks of losing older messages in the log.
                        Use this option carefully, maybe worth to setup a
                        bigger log buffer with "log_buf_len" along with this.

        parkbd.port=    [HW] Parallel port number the keyboard adapter is
                        connected to, default is 0.
                        Format: <parport#>
        parkbd.mode=    [HW] Parallel port keyboard adapter mode of operation,
                        0 for XT, 1 for AT (default is AT).
                        Format: <mode>

        parport=        [HW,PPT] Specify parallel ports. 0 disables.
                        Format: { 0 | auto | 0xBBB[,IRQ[,DMA]] }
                        Use 'auto' to force the driver to use any
                        IRQ/DMA settings detected (the default is to
                        ignore detected IRQ/DMA settings because of
                        possible conflicts). You can specify the base
                        address, IRQ, and DMA settings; IRQ and DMA
                        should be numbers, or 'auto' (for using detected
                        settings on that particular port), or 'nofifo'
                        (to avoid using a FIFO even if it is detected).
                        Parallel ports are assigned in the order they
                        are specified on the command line, starting
                        with parport0.

        parport_init_mode=      [HW,PPT]
                        Configure VIA parallel port to operate in
                        a specific mode. This is necessary on Pegasos
                        computer where firmware has no options for setting
                        up parallel port mode and sets it to spp.
                        Currently this function knows 686a and 8231 chips.
                        Format: [spp|ps2|epp|ecp|ecpepp]

        pata_legacy.all=        [HW,LIBATA]
                        Format: <int>
                        Set to non-zero to probe primary and secondary ISA
                        port ranges on PCI systems where no PCI PATA device
                        has been found at either range.  Disabled by default.
```

```
pata_legacy.autospeed=  [HW,LIBATA]
                Format: <int>
                Set to non-zero if a chip is present that snoops speed
                changes.  Disabled by default.

pata_legacy.ht6560a=    [HW,LIBATA]
                Format: <int>
                Set to 1, 2, or 3 for HT 6560A on the primary channel,
                the secondary channel, or both channels respectively.
                Disabled by default.

pata_legacy.ht6560b=    [HW,LIBATA]
                Format: <int>
                Set to 1, 2, or 3 for HT 6560B on the primary channel,
                the secondary channel, or both channels respectively.
                Disabled by default.

pata_legacy.iordy_mask= [HW,LIBATA]
                Format: <int>
                IORDY enable mask.  Set individual bits to allow IORDY
                for the respective channel.  Bit 0 is for the first
                legacy channel handled by this driver, bit 1 is for
                the second channel, and so on.  The sequence will often
                correspond to the primary legacy channel, the secondary
                legacy channel, and so on, but the handling of a PCI
                bus and the use of other driver options may interfere
                with the sequence.  By default IORDY is allowed across
                all channels.

pata_legacy.opti82c46x= [HW,LIBATA]
                Format: <int>
                Set to 1, 2, or 3 for Opti 82c611A on the primary
                channel, the secondary channel, or both channels
                respectively.  Disabled by default.

pata_legacy.opti82c611a=        [HW,LIBATA]
                Format: <int>
                Set to 1, 2, or 3 for Opti 82c465MV on the primary
                channel, the secondary channel, or both channels
                respectively.  Disabled by default.

pata_legacy.pio_mask=   [HW,LIBATA]
                Format: <int>
                PIO mode mask for autospeed devices.  Set individual
                bits to allow the use of the respective PIO modes.
                Bit 0 is for mode 0, bit 1 is for mode 1, and so on.
                All modes allowed by default.

pata_legacy.probe_all=  [HW,LIBATA]
                Format: <int>
                Set to non-zero to probe tertiary and further ISA
```

```
                        port ranges on PCI systems.  Disabled by default.

        pata_legacy.probe_mask= [HW,LIBATA]
                        Format: <int>
                        Probe mask for legacy ISA PATA ports.  Depending on
                        platform configuration and the use of other driver
                        options up to 6 legacy ports are supported: 0x1f0,
                        0x170, 0x1e8, 0x168, 0x1e0, 0x160, however probing
                        of individual ports can be disabled by setting the
                        corresponding bits in the mask to 1.  Bit 0 is for
                        the first port in the list above (0x1f0), and so on.
                        By default all supported ports are probed.

        pata_legacy.qdi=        [HW,LIBATA]
                        Format: <int>
                        Set to non-zero to probe QDI controllers.  By default
                        set to 1 if CONFIG_PATA_QDI_MODULE, 0 otherwise.

        pata_legacy.winbond=    [HW,LIBATA]
                        Format: <int>
                        Set to non-zero to probe Winbond controllers.  Use
                        the standard I/O port (0x130) if 1, otherwise the
                        value given is the I/O port to use (typically 0x1b0).
                        By default set to 1 if CONFIG_PATA_WINBOND_VLB_MODULE,
                        0 otherwise.

        pata_platform.pio_mask= [HW,LIBATA]
                        Format: <int>
                        Supported PIO mode mask.  Set individual bits to allow
                        the use of the respective PIO modes.  Bit 0 is for
                        mode 0, bit 1 is for mode 1, and so on.  Mode 0 only
                        allowed by default.

        pause_on_oops=<int>
                        Halt all CPUs after the first oops has been printed for
                        the specified number of seconds.  This is to be used if
                        your oopses keep scrolling off the screen.

        pcbit=          [HW,ISDN]

        pci=option[,option...]  [PCI] various PCI subsystem options.

                                Some options herein operate on a specific␣
↪device
                                or a set of devices (<pci_dev>). These are
                                specified in one of the following formats:

                                [<domain>:]<bus>:<dev>.<func>[/<dev>.<func>]*
                                pci:<vendor>:<device>[:<subvendor>:<subdevice>]

                                Note: the first format specifies a PCI
```

```
                                bus/device/function address which may change
                                if new hardware is inserted, if motherboard
                                firmware changes, or due to changes caused
                                by other kernel parameters. If the
                                domain is left unspecified, it is
                                taken to be zero. Optionally, a path
                                to a device through multiple device/function
                                addresses can be specified after the base
                                address (this is more robust against
                                renumbering issues).  The second format
                                selects devices using IDs from the
                                configuration space which may match multiple
                                devices in the system.

                earlydump       dump PCI config space before the kernel
                                changes anything
                off             [X86] don't probe for the PCI bus
                bios            [X86-32] force use of PCI BIOS, don't access
                                the hardware directly. Use this if your machine
                                has a non-standard PCI host bridge.
                nobios          [X86-32] disallow use of PCI BIOS, only direct
                                hardware access methods are allowed. Use this
                                if you experience crashes upon bootup and you
                                suspect they are caused by the BIOS.
                conf1           [X86] Force use of PCI Configuration Access
                                Mechanism 1 (config address in IO port 0xCF8,
                                data in IO port 0xCFC, both 32-bit).
                conf2           [X86] Force use of PCI Configuration Access
                                Mechanism 2 (IO port 0xCF8 is an 8-bit port for
                                the function, IO port 0xCFA, also 8-bit, sets
                                bus number. The config space is then accessed
                                through ports 0xC000-0xCFFF).
                                See http://wiki.osdev.org/PCI for more info
                                on the configuration access mechanisms.
                noaer           [PCIE] If the PCIEAER kernel config parameter␣
↪is
                                enabled, this kernel boot option can be used to
                                disable the use of PCIE advanced error␣
↪reporting.
                nodomains       [PCI] Disable support for multiple PCI
                                root domains (aka PCI segments, in ACPI-speak).
                nommconf        [X86] Disable use of MMCONFIG for PCI
                                Configuration
                check_enable_amd_mmconf [X86] check for and enable
                                properly configured MMIO access to PCI
                                config space on AMD family 10h CPU
                nomsi           [MSI] If the PCI_MSI kernel config parameter is
                                enabled, this kernel boot option can be used to
                                disable the use of MSI interrupts system-wide.
                noioapicquirk   [APIC] Disable all boot interrupt quirks.
                                Safety option to keep boot IRQs enabled. This
```

```
                                should never be necessary.
                ioapicreroute   [APIC] Enable rerouting of boot IRQs to the
                                primary IO-APIC for bridges that cannot disable
                                boot IRQs. This fixes a source of spurious IRQs
                                when the system masks IRQs.
                noioapicreroute [APIC] Disable workaround that uses the
                                boot IRQ equivalent of an IRQ that connects to
                                a chipset where boot IRQs cannot be disabled.
                                The opposite of ioapicreroute.
                biosirq         [X86-32] Use PCI BIOS calls to get the␣
↪interrupt
                                routing table. These calls are known to be␣
↪buggy
                                on several machines and they hang the machine
                                when used, but on other computers it's the only
                                way to get the interrupt routing table. Try
                                this option if the kernel is unable to allocate
                                IRQs or discover secondary PCI buses on your
                                motherboard.
                rom             [X86] Assign address space to expansion ROMs.
                                Use with caution as certain devices share
                                address decoders between ROMs and other
                                resources.
                norom           [X86] Do not assign address space to
                                expansion ROMs that do not already have
                                BIOS assigned address ranges.
                nobar           [X86] Do not assign address space to the
                                BARs that weren't assigned by the BIOS.
                irqmask=0xMMMM  [X86] Set a bit mask of IRQs allowed to be
                                assigned automatically to PCI devices. You can
                                make the kernel exclude IRQs of your ISA cards
                                this way.
                pirqaddr=0xAAAAA        [X86] Specify the physical address
                                of the PIRQ table (normally generated
                                by the BIOS) if it is outside the
                                F0000h-100000h range.
                lastbus=N       [X86] Scan all buses thru bus #N. Can be
                                useful if the kernel is unable to find your
                                secondary buses and you want to tell it
                                explicitly which ones they are.
                assign-busses   [X86] Always assign all PCI bus
                                numbers ourselves, overriding
                                whatever the firmware may have done.
                usepirqmask     [X86] Honor the possible IRQ mask stored
                                in the BIOS $PIR table. This is needed on
                                some systems with broken BIOSes, notably
                                some HP Pavilion N5400 and Omnibook XE3
                                notebooks. This will have no effect if ACPI
                                IRQ routing is enabled.
                noacpi          [X86] Do not use ACPI for IRQ routing
                                or for PCI scanning.
```

```
                use_crs         [X86] Use PCI host bridge window information
                                from ACPI.  On BIOSes from 2008 or later, this
                                is enabled by default.  If you need to use␣
↪this,
                                please report a bug.
                nocrs           [X86] Ignore PCI host bridge windows from ACPI.
                                If you need to use this, please report a bug.
                use_e820        [X86] Use E820 reservations to exclude parts of
                                PCI host bridge windows. This is a workaround
                                for BIOS defects in host bridge _CRS methods.
                                If you need to use this, please report a bug to
                                <linux-pci@vger.kernel.org>.
                no_e820         [X86] Ignore E820 reservations for PCI host
                                bridge windows. This is the default on modern
                                hardware. If you need to use this, please␣
↪report
                                a bug to <linux-pci@vger.kernel.org>.
                routeirq        Do IRQ routing for all PCI devices.
                                This is normally done in pci_enable_device(),
                                so this option is a temporary workaround
                                for broken drivers that don't call it.
                skip_isa_align  [X86] do not align io start addr, so can
                                handle more pci cards
                noearly         [X86] Don't do any early type 1 scanning.
                                This might help on some broken boards which
                                machine check when some devices' config space
                                is read. But various workarounds are disabled
                                and some IOMMU drivers will not work.
                bfsort          Sort PCI devices into breadth-first order.
                                This sorting is done to get a device
                                order compatible with older (<= 2.4) kernels.
                nobfsort        Don't sort PCI devices into breadth-first␣
↪order.
                pcie_bus_tune_off       Disable PCIe MPS (Max Payload Size)
                                tuning and use the BIOS-configured MPS␣
↪defaults.
                pcie_bus_safe   Set every device's MPS to the largest value
                                supported by all devices below the root␣
↪complex.
                pcie_bus_perf   Set device MPS to the largest allowable MPS
                                based on its parent bus. Also set MRRS (Max
                                Read Request Size) to the largest supported
                                value (no larger than the MPS that the device
                                or bus can support) for best performance.
                pcie_bus_peer2peer      Set every device's MPS to 128B, which
                                every device is guaranteed to support. This
                                configuration allows peer-to-peer DMA between
                                any pair of devices, possibly at the cost of
                                reduced performance.  This also guarantees
                                that hot-added devices will work.
                cbiosize=nn[KMG]        The fixed amount of bus space which is
```

```
                                reserved for the CardBus bridge's IO window.
                                The default value is 256 bytes.
                cbmemsize=nn[KMG]       The fixed amount of bus space which is
                                reserved for the CardBus bridge's memory
                                window. The default value is 64 megabytes.
                resource_alignment=
                                Format:
                                [<order of align>@]<pci_dev>[; ...]
                                Specifies alignment and device to reassign
                                aligned memory resources. How to
                                specify the device is described above.
                                If <order of align> is not specified,
                                PAGE_SIZE is used as alignment.
                                A PCI-PCI bridge can be specified if resource
                                windows need to be expanded.
                                To specify the alignment for several
                                instances of a device, the PCI vendor,
                                device, subvendor, and subdevice may be
                                specified, e.g., 12@pci:8086:9c22:103c:198f
                                for 4096-byte alignment.
                ecrc=           Enable/disable PCIe ECRC (transaction layer
                                end-to-end CRC checking). Only effective if
                                OS has native AER control (either granted by
                                ACPI _OSC or forced via "pcie_ports=native")
                                bios: Use BIOS/firmware settings. This is the
                                the default.
                                off: Turn ECRC off
                                on: Turn ECRC on.
                hpiosize=nn[KMG]        The fixed amount of bus space which is
                                reserved for hotplug bridge's IO window.
                                Default size is 256 bytes.
                hpmmiosize=nn[KMG]      The fixed amount of bus space which is
                                reserved for hotplug bridge's MMIO window.
                                Default size is 2 megabytes.
                hpmmioprefsize=nn[KMG]  The fixed amount of bus space which is
                                reserved for hotplug bridge's MMIO_PREF window.
                                Default size is 2 megabytes.
                hpmemsize=nn[KMG]       The fixed amount of bus space which is
                                reserved for hotplug bridge's MMIO and
                                MMIO_PREF window.
                                Default size is 2 megabytes.
                hpbussize=nn    The minimum amount of additional bus numbers
                                reserved for buses below a hotplug bridge.
                                Default is 1.
                realloc=        Enable/disable reallocating PCI bridge␣
↪resources
                                if allocations done by BIOS are too small to
                                accommodate resources required by all child
                                devices.
                                off: Turn realloc off
                                on: Turn realloc on
```

```
                realloc         same as realloc=on
                noari           do not use PCIe ARI.
                noats           [PCIE, Intel-IOMMU, AMD-IOMMU]
                                do not use PCIe ATS (and IOMMU device IOTLB).
                pcie_scan_all   Scan all possible PCIe devices.  Otherwise we
                                only look for one device below a PCIe␣
↪downstream
                                port.
                big_root_window Try to add a big 64bit memory window to the␣
↪PCIe
                                root complex on AMD CPUs. Some GFX hardware
                                can resize a BAR to allow access to all VRAM.
                                Adding the window is slightly risky (it may
                                conflict with unreported devices), so this
                                taints the kernel.
                disable_acs_redir=<pci_dev>[; ...]
                                Specify one or more PCI devices (in the format
                                specified above) separated by semicolons.
                                Each device specified will have the PCI ACS
                                redirect capabilities forced off which will
                                allow P2P traffic between devices through
                                bridges without forcing it upstream. Note:
                                this removes isolation between devices and
                                may put more devices in an IOMMU group.
                force_floating  [S390] Force usage of floating interrupts.
                nomio           [S390] Do not use MIO instructions.
                norid           [S390] ignore the RID field and force use of
                                one PCI domain per PCI function

        pcie_aspm=      [PCIE] Forcibly enable or disable PCIe Active State␣
↪Power
                        Management.
                off     Disable ASPM.
                force   Enable ASPM even on devices that claim not to support␣
↪it.
                        WARNING: Forcing ASPM on may cause system lockups.

        pcie_ports=     [PCIE] PCIe port services handling:
                native  Use native PCIe services (PME, AER, DPC, PCIe hotplug)
                        even if the platform doesn't give the OS permission to
                        use them.  This may cause conflicts if the platform
                        also tries to use these services.
                dpc-native      Use native PCIe service for DPC only.  May
                                cause conflicts if firmware uses AER or DPC.
                compat  Disable native PCIe services (PME, AER, DPC, PCIe
                        hotplug).

        pcie_port_pm=   [PCIE] PCIe port power management handling:
                off     Disable power management of all PCIe ports
                force   Forcibly enable power management of all PCIe ports
```

```
        pcie_pme=       [PCIE,PM] Native PCIe PME signaling options:
                nomsi   Do not use MSI for native PCIe PME signaling (this␣
↪makes
                        all PCIe root ports use INTx for all services).

        pcmv=           [HW,PCMCIA] BadgePAD 4

        pd_ignore_unused
                        [PM]
                        Keep all power-domains already enabled by bootloader␣
↪on,
                        even if no driver has claimed them. This is useful
                        for debug and development, but should not be
                        needed on a platform with proper driver support.

        pdcchassis=     [PARISC,HW] Disable/Enable PDC Chassis Status codes at
                        boot time.
                        Format: { 0 | 1 }
                        See arch/parisc/kernel/pdc_chassis.c

        percpu_alloc=   Select which percpu first chunk allocator to use.
                        Currently supported values are "embed" and "page".
                        Archs may support subset or none of the selections.
                        See comments in mm/percpu.c for details on each
                        allocator.  This parameter is primarily for debugging
                        and performance comparison.

        pirq=           [SMP,APIC] Manual mp-table setup
                        See Documentation/arch/x86/i386/IO-APIC.rst.

        plip=           [PPT,NET] Parallel port network link
                        Format: { parport<nr> | timid | 0 }
                        See also Documentation/admin-guide/parport.rst.

        pmtmr=          [X86] Manual setup of pmtmr I/O Port.
                        Override pmtimer IOPort with a hex value.
                        e.g. pmtmr=0x508

        pmu_override=   [PPC] Override the PMU.
                        This option takes over the PMU facility, so it is no
                        longer usable by perf. Setting this option starts the
                        PMU counters by setting MMCR0 to 0 (the FC bit is
                        cleared). If a number is given, then MMCR1 is set to
                        that number, otherwise (e.g., 'pmu_override=on'), MMCR1
                        remains 0.

        pm_debug_messages       [SUSPEND,KNL]
                        Enable suspend/resume debug messages during boot up.

        pnp.debug=1     [PNP]
                        Enable PNP debug messages (depends on the
```

```
                        CONFIG_PNP_DEBUG_MESSAGES option).  Change at run-time
                        via /sys/module/pnp/parameters/debug.  We always show
                        current resource usage; turning this on also shows
                        possible settings and some assignment information.

        pnpacpi=        [ACPI]
                        { off }

        pnpbios=        [ISAPNP]
                        { on | off | curr | res | no-curr | no-res }

        pnp_reserve_irq=
                        [ISAPNP] Exclude IRQs for the autoconfiguration

        pnp_reserve_dma=
                        [ISAPNP] Exclude DMAs for the autoconfiguration

        pnp_reserve_io= [ISAPNP] Exclude I/O ports for the autoconfiguration
                        Ranges are in pairs (I/O port base and size).

        pnp_reserve_mem=
                        [ISAPNP] Exclude memory regions for the
                        autoconfiguration.
                        Ranges are in pairs (memory base and size).

        ports=          [IP_VS_FTP] IPVS ftp helper module
                        Default is 21.
                        Up to 8 (IP_VS_APP_MAX_PORTS) ports
                        may be specified.
                        Format: <port>,<port>....

        powersave=off   [PPC] This option disables power saving features.
                        It specifically disables cpuidle and sets the
                        platform machine description specific power_save
                        function to NULL. On Idle the CPU just reduces
                        execution priority.

        ppc_strict_facility_enable
                        [PPC] This option catches any kernel floating point,
                        Altivec, VSX and SPE outside of regions specifically
                        allowed (eg kernel_enable_fpu()/kernel_disable_fpu()).
                        There is some performance impact when enabling this.

        ppc_tm=         [PPC]
                        Format: {"off"}
                        Disable Hardware Transactional Memory

        preempt=        [KNL]
                        Select preemption mode if you have CONFIG_PREEMPT_
↪DYNAMIC
                        none - Limited to cond_resched() calls
```

```
                        voluntary - Limited to cond_resched() and might_
↪sleep() calls
                        full - Any section that isn't explicitly preempt␣
↪disabled
                               can be preempted anytime.

        print-fatal-signals=
                        [KNL] debug: print fatal signals

                        If enabled, warn about various signal handling
                        related application anomalies: too many signals,
                        too many POSIX.1 timers, fatal signals causing a
                        coredump - etc.

                        If you hit the warning due to signal overflow,
                        you might want to try "ulimit -i unlimited".

                        default: off.

        printk.always_kmsg_dump=
                        Trigger kmsg_dump for cases other than kernel oops or
                        panics
                        Format: <bool>  (1/Y/y=enable, 0/N/n=disable)
                        default: disabled

        printk.console_no_auto_verbose=
                        Disable console loglevel raise on oops, panic
                        or lockdep-detected issues (only if lock debug is on).
                        With an exception to setups with low baudrate on
                        serial console, keeping this 0 is a good choice
                        in order to provide more debug information.
                        Format: <bool>
                        default: 0 (auto_verbose is enabled)

        printk.devkmsg={on,off,ratelimit}
                        Control writing to /dev/kmsg.
                        on - unlimited logging to /dev/kmsg from userspace
                        off - logging to /dev/kmsg disabled
                        ratelimit - ratelimit the logging
                        Default: ratelimit

        printk.time=    Show timing data prefixed to each printk message line
                        Format: <bool>  (1/Y/y=enable, 0/N/n=disable)

        processor.max_cstate=   [HW,ACPI]
                        Limit processor to maximum C-state
                        max_cstate=9 overrides any DMI blacklist limit.

        processor.nocst [HW,ACPI]
                        Ignore the _CST method to determine C-states,
                        instead using the legacy FADT method
```

```
profile=        [KNL] Enable kernel profiling via /proc/profile
                Format: [<profiletype>,]<number>
                Param: <profiletype>: "schedule", "sleep", or "kvm"
                        [defaults to kernel profiling]
                Param: "schedule" - profile schedule points.
                Param: "sleep" - profile D-state sleeping (millisecs).
                        Requires CONFIG_SCHEDSTATS
                Param: "kvm" - profile VM exits.
                Param: <number> - step/bucket size as a power of 2 for
                        statistical time based profiling.

prompt_ramdisk= [RAM] [Deprecated]

prot_virt=      [S390] enable hosting protected virtual machines
                isolated from the hypervisor (if hardware supports
                that).
                Format: <bool>

psi=            [KNL] Enable or disable pressure stall information
                tracking.
                Format: <bool>

psmouse.proto=  [HW,MOUSE] Highest PS2 mouse protocol extension to
                probe for; one of (bare|imps|exps|lifebook|any).
psmouse.rate=   [HW,MOUSE] Set desired mouse report rate, in reports
                per second.
psmouse.resetafter=     [HW,MOUSE]
                Try to reset the device after so many bad packets
                (0 = never).
psmouse.resolution=
                [HW,MOUSE] Set desired mouse resolution, in dpi.
psmouse.smartscroll=
                [HW,MOUSE] Controls Logitech smartscroll autorepeat.
                0 = disabled, 1 = enabled (default).

pstore.backend= Specify the name of the pstore backend to use

pti=            [X86-64] Control Page Table Isolation of user and
                kernel address spaces.  Disabling this feature
                removes hardening, but improves performance of
                system calls and interrupts.

                on   - unconditionally enable
                off  - unconditionally disable
                auto - kernel detects whether your CPU model is
                       vulnerable to issues that PTI mitigates

                Not specifying this option is equivalent to pti=auto.

pty.legacy_count=
```

```
                [KNL] Number of legacy pty's. Overwrites compiled-in
                default number.

quiet           [KNL] Disable most log messages

r128=           [HW,DRM]

radix_hcall_invalidate=on  [PPC/PSERIES]
                Disable RADIX GTSE feature and use hcall for TLB
                invalidate.

raid=           [HW,RAID]
                See Documentation/admin-guide/md.rst.

ramdisk_size=   [RAM] Sizes of RAM disks in kilobytes
                See Documentation/admin-guide/blockdev/ramdisk.rst.

ramdisk_start=  [RAM] RAM disk image start address

random.trust_cpu=off
                [KNL] Disable trusting the use of the CPU's
                random number generator (if available) to
                initialize the kernel's RNG.

random.trust_bootloader=off
                [KNL] Disable trusting the use of the a seed
                passed by the bootloader (if available) to
                initialize the kernel's RNG.

randomize_kstack_offset=
                [KNL] Enable or disable kernel stack offset
                randomization, which provides roughly 5 bits of
                entropy, frustrating memory corruption attacks
                that depend on stack address determinism or
                cross-syscall address exposures. This is only
                available on architectures that have defined
                CONFIG_HAVE_ARCH_RANDOMIZE_KSTACK_OFFSET.
                Format: <bool>  (1/Y/y=enable, 0/N/n=disable)
                Default is CONFIG_RANDOMIZE_KSTACK_OFFSET_DEFAULT.

ras=option[,option,...] [KNL] RAS-specific options

        cec_disable     [X86]
                        Disable the Correctable Errors Collector,
                        see CONFIG_RAS_CEC help text.

rcu_nocbs[=cpu-list]
                [KNL] The optional argument is a cpu list,
                as described above.

                In kernels built with CONFIG_RCU_NOCB_CPU=y,
```

```
                enable the no-callback CPU mode, which prevents
                such CPUs' callbacks from being invoked in
                softirq context.  Invocation of such CPUs' RCU
                callbacks will instead be offloaded to "rcuox/N"
                kthreads created for that purpose, where "x" is
                "p" for RCU-preempt, "s" for RCU-sched, and "g"
                for the kthreads that mediate grace periods; and
                "N" is the CPU number. This reduces OS jitter on
                the offloaded CPUs, which can be useful for HPC
                and real-time workloads.  It can also improve
                energy efficiency for asymmetric multiprocessors.

                If a cpulist is passed as an argument, the specified
                list of CPUs is set to no-callback mode from boot.

                Otherwise, if the '=' sign and the cpulist
                arguments are omitted, no CPU will be set to
                no-callback mode from boot but the mode may be
                toggled at runtime via cpusets.

                Note that this argument takes precedence over
                the CONFIG_RCU_NOCB_CPU_DEFAULT_ALL option.

        rcu_nocb_poll   [KNL]
                Rather than requiring that offloaded CPUs
                (specified by rcu_nocbs= above) explicitly
                awaken the corresponding "rcuoN" kthreads,
                make these kthreads poll for callbacks.
                This improves the real-time response for the
                offloaded CPUs by relieving them of the need to
                wake up the corresponding kthread, but degrades
                energy efficiency by requiring that the kthreads
                periodically wake up to do the polling.

        rcutree.blimit= [KNL]
                Set maximum number of finished RCU callbacks to
                process in one batch.

        rcutree.do_rcu_barrier= [KNL]
                Request a call to rcu_barrier().  This is
                throttled so that userspace tests can safely
                hammer on the sysfs variable if they so choose.
                If triggered before the RCU grace-period machinery
                is fully active, this will error out with EAGAIN.

        rcutree.dump_tree=      [KNL]
                Dump the structure of the rcu_node combining tree
                out at early boot.  This is used for diagnostic
                purposes, to verify correct tree setup.

        rcutree.gp_cleanup_delay=       [KNL]
```

```
                Set the number of jiffies to delay each step of
                RCU grace-period cleanup.

rcutree.gp_init_delay=  [KNL]
                Set the number of jiffies to delay each step of
                RCU grace-period initialization.

rcutree.gp_preinit_delay=       [KNL]
                Set the number of jiffies to delay each step of
                RCU grace-period pre-initialization, that is,
                the propagation of recent CPU-hotplug changes up
                the rcu_node combining tree.

rcutree.jiffies_till_first_fqs= [KNL]
                Set delay from grace-period initialization to
                first attempt to force quiescent states.
                Units are jiffies, minimum value is zero,
                and maximum value is HZ.

rcutree.jiffies_till_next_fqs= [KNL]
                Set delay between subsequent attempts to force
                quiescent states.  Units are jiffies, minimum
                value is one, and maximum value is HZ.

rcutree.jiffies_till_sched_qs= [KNL]
                Set required age in jiffies for a
                given grace period before RCU starts
                soliciting quiescent-state help from
                rcu_note_context_switch() and cond_resched().
                If not specified, the kernel will calculate
                a value based on the most recent settings
                of rcutree.jiffies_till_first_fqs
                and rcutree.jiffies_till_next_fqs.
                This calculated value may be viewed in
                rcutree.jiffies_to_sched_qs.  Any attempt to set
                rcutree.jiffies_to_sched_qs will be cheerfully
                overwritten.

rcutree.kthread_prio=    [KNL,BOOT]
                Set the SCHED_FIFO priority of the RCU per-CPU
                kthreads (rcuc/N). This value is also used for
                the priority of the RCU boost threads (rcub/N)
                and for the RCU grace-period kthreads (rcu_bh,
                rcu_preempt, and rcu_sched). If RCU_BOOST is
                set, valid values are 1-99 and the default is 1
                (the least-favored priority).  Otherwise, when
                RCU_BOOST is not set, valid values are 0-99 and
                the default is zero (non-realtime operation).
                When RCU_NOCB_CPU is set, also adjust the
                priority of NOCB callback kthreads.
```

```
rcutree.nocb_nobypass_lim_per_jiffy= [KNL]
                On callback-offloaded (rcu_nocbs) CPUs,
                RCU reduces the lock contention that would
                otherwise be caused by callback floods through
                use of the ->nocb_bypass list.  However, in the
                common non-flooded case, RCU queues directly to
                the main ->cblist in order to avoid the extra
                overhead of the ->nocb_bypass list and its lock.
                But if there are too many callbacks queued during
                a single jiffy, RCU pre-queues the callbacks into
                the ->nocb_bypass queue.  The definition of "too
                many" is supplied by this kernel boot parameter.

rcutree.qhimark= [KNL]
                Set threshold of queued RCU callbacks beyond which
                batch limiting is disabled.

rcutree.qlowmark= [KNL]
                Set threshold of queued RCU callbacks below which
                batch limiting is re-enabled.

rcutree.qovld= [KNL]
                Set threshold of queued RCU callbacks beyond which
                RCU's force-quiescent-state scan will aggressively
                enlist help from cond_resched() and sched IPIs to
                help CPUs more quickly reach quiescent states.
                Set to less than zero to make this be set based
                on rcutree.qhimark at boot time and to zero to
                disable more aggressive help enlistment.

rcutree.rcu_delay_page_cache_fill_msec= [KNL]
                Set the page-cache refill delay (in milliseconds)
                in response to low-memory conditions.  The range
                of permitted values is in the range 0:100000.

rcutree.rcu_divisor= [KNL]
                Set the shift-right count to use to compute
                the callback-invocation batch limit bl from
                the number of callbacks queued on this CPU.
                The result will be bounded below by the value of
                the rcutree.blimit kernel parameter.  Every bl
                callbacks, the softirq handler will exit in
                order to allow the CPU to do other work.

                Please note that this callback-invocation batch
                limit applies only to non-offloaded callback
                invocation.  Offloaded callbacks are instead
                invoked in the context of an rcuoc kthread, which
                scheduler will preempt as it does any other task.

rcutree.rcu_fanout_exact= [KNL]
```

```
                Disable autobalancing of the rcu_node combining
                tree.  This is used by rcutorture, and might
                possibly be useful for architectures having high
                cache-to-cache transfer latencies.

rcutree.rcu_fanout_leaf= [KNL]
                Change the number of CPUs assigned to each
                leaf rcu_node structure.  Useful for very
                large systems, which will choose the value 64,
                and for NUMA systems with large remote-access
                latencies, which will choose a value aligned
                with the appropriate hardware boundaries.

rcutree.rcu_min_cached_objs= [KNL]
                Minimum number of objects which are cached and
                maintained per one CPU. Object size is equal
                to PAGE_SIZE. The cache allows to reduce the
                pressure to page allocator, also it makes the
                whole algorithm to behave better in low memory
                condition.

rcutree.rcu_nocb_gp_stride= [KNL]
                Set the number of NOCB callback kthreads in
                each group, which defaults to the square root
                of the number of CPUs.  Larger numbers reduce
                the wakeup overhead on the global grace-period
                kthread, but increases that same overhead on
                each group's NOCB grace-period kthread.

rcutree.rcu_kick_kthreads= [KNL]
                Cause the grace-period kthread to get an extra
                wake_up() if it sleeps three times longer than
                it should at force-quiescent-state time.
                This wake_up() will be accompanied by a
                WARN_ONCE() splat and an ftrace_dump().

rcutree.rcu_resched_ns= [KNL]
                Limit the time spend invoking a batch of RCU
                callbacks to the specified number of nanoseconds.
                By default, this limit is checked only once
                every 32 callbacks in order to limit the pain
                inflicted by local_clock() overhead.

rcutree.rcu_unlock_delay= [KNL]
                In CONFIG_RCU_STRICT_GRACE_PERIOD=y kernels,
                this specifies an rcu_read_unlock()-time delay
                in microseconds.  This defaults to zero.
                Larger delays increase the probability of
                catching RCU pointer leaks, that is, buggy use
                of RCU-protected pointers after the relevant
                rcu_read_unlock() has completed.
```

```
rcutree.sysrq_rcu= [KNL]
                Commandeer a sysrq key to dump out Tree RCU's
                rcu_node tree with an eye towards determining
                why a new grace period has not yet started.

rcutree.use_softirq=    [KNL]
                If set to zero, move all RCU_SOFTIRQ processing to
                per-CPU rcuc kthreads.  Defaults to a non-zero
                value, meaning that RCU_SOFTIRQ is used by default.
                Specify rcutree.use_softirq=0 to use rcuc kthreads.

                But note that CONFIG_PREEMPT_RT=y kernels disable
                this kernel boot parameter, forcibly setting it
                to zero.

rcuscale.gp_async= [KNL]
                Measure performance of asynchronous
                grace-period primitives such as call_rcu().

rcuscale.gp_async_max= [KNL]
                Specify the maximum number of outstanding
                callbacks per writer thread.  When a writer
                thread exceeds this limit, it invokes the
                corresponding flavor of rcu_barrier() to allow
                previously posted callbacks to drain.

rcuscale.gp_exp= [KNL]
                Measure performance of expedited synchronous
                grace-period primitives.

rcuscale.holdoff= [KNL]
                Set test-start holdoff period.  The purpose of
                this parameter is to delay the start of the
                test until boot completes in order to avoid
                interference.

rcuscale.kfree_by_call_rcu= [KNL]
                In kernels built with CONFIG_RCU_LAZY=y, test
                call_rcu() instead of kfree_rcu().

rcuscale.kfree_mult= [KNL]
                Instead of allocating an object of size kfree_obj,
                allocate one of kfree_mult * sizeof(kfree_obj).
                Defaults to 1.

rcuscale.kfree_rcu_test= [KNL]
                Set to measure performance of kfree_rcu() flooding.

rcuscale.kfree_rcu_test_double= [KNL]
                Test the double-argument variant of kfree_rcu().
```

```
                If this parameter has the same value as
                rcuscale.kfree_rcu_test_single, both the single-
                and double-argument variants are tested.

rcuscale.kfree_rcu_test_single= [KNL]
                Test the single-argument variant of kfree_rcu().
                If this parameter has the same value as
                rcuscale.kfree_rcu_test_double, both the single-
                and double-argument variants are tested.

rcuscale.kfree_nthreads= [KNL]
                The number of threads running loops of kfree_rcu().

rcuscale.kfree_alloc_num= [KNL]
                Number of allocations and frees done in an iteration.

rcuscale.kfree_loops= [KNL]
                Number of loops doing rcuscale.kfree_alloc_num number
                of allocations and frees.

rcuscale.minruntime= [KNL]
                Set the minimum test run time in seconds.  This
                does not affect the data-collection interval,
                but instead allows better measurement of things
                like CPU consumption.

rcuscale.nreaders= [KNL]
                Set number of RCU readers.  The value -1 selects
                N, where N is the number of CPUs.  A value
                "n" less than -1 selects N-n+1, where N is again
                the number of CPUs.  For example, -2 selects N
                (the number of CPUs), -3 selects N+1, and so on.
                A value of "n" less than or equal to -N selects
                a single reader.

rcuscale.nwriters= [KNL]
                Set number of RCU writers.  The values operate
                the same as for rcuscale.nreaders.
                N, where N is the number of CPUs

rcuscale.scale_type= [KNL]
                Specify the RCU implementation to test.

rcuscale.shutdown= [KNL]
                Shut the system down after performance tests
                complete.  This is useful for hands-off automated
                testing.

rcuscale.verbose= [KNL]
                Enable additional printk() statements.
```

```
rcuscale.writer_holdoff= [KNL]
                Write-side holdoff between grace periods,
                in microseconds.  The default of zero says
                no holdoff.

rcuscale.writer_holdoff_jiffies= [KNL]
                Additional write-side holdoff between grace
                periods, but in jiffies.  The default of zero
                says no holdoff.

rcutorture.fqs_duration= [KNL]
                Set duration of force_quiescent_state bursts
                in microseconds.

rcutorture.fqs_holdoff= [KNL]
                Set holdoff time within force_quiescent_state bursts
                in microseconds.

rcutorture.fqs_stutter= [KNL]
                Set wait time between force_quiescent_state bursts
                in seconds.

rcutorture.fwd_progress= [KNL]
                Specifies the number of kthreads to be used
                for  RCU grace-period forward-progress testing
                for the types of RCU supporting this notion.
                Defaults to 1 kthread, values less than zero or
                greater than the number of CPUs cause the number
                of CPUs to be used.

rcutorture.fwd_progress_div= [KNL]
                Specify the fraction of a CPU-stall-warning
                period to do tight-loop forward-progress testing.

rcutorture.fwd_progress_holdoff= [KNL]
                Number of seconds to wait between successive
                forward-progress tests.

rcutorture.fwd_progress_need_resched= [KNL]
                Enclose cond_resched() calls within checks for
                need_resched() during tight-loop forward-progress
                testing.

rcutorture.gp_cond= [KNL]
                Use conditional/asynchronous update-side
                primitives, if available.

rcutorture.gp_exp= [KNL]
                Use expedited update-side primitives, if available.

rcutorture.gp_normal= [KNL]
```

```
                Use normal (non-expedited) asynchronous
                update-side primitives, if available.

rcutorture.gp_sync= [KNL]
                Use normal (non-expedited) synchronous
                update-side primitives, if available.  If all
                of rcutorture.gp_cond=, rcutorture.gp_exp=,
                rcutorture.gp_normal=, and rcutorture.gp_sync=
                are zero, rcutorture acts as if is interpreted
                they are all non-zero.

rcutorture.irqreader= [KNL]
                Run RCU readers from irq handlers, or, more
                accurately, from a timer handler.  Not all RCU
                flavors take kindly to this sort of thing.

rcutorture.leakpointer= [KNL]
                Leak an RCU-protected pointer out of the reader.
                This can of course result in splats, and is
                intended to test the ability of things like
                CONFIG_RCU_STRICT_GRACE_PERIOD=y to detect
                such leaks.

rcutorture.n_barrier_cbs= [KNL]
                Set callbacks/threads for rcu_barrier() testing.

rcutorture.nfakewriters= [KNL]
                Set number of concurrent RCU writers.  These just
                stress RCU, they don't participate in the actual
                test, hence the "fake".

rcutorture.nocbs_nthreads= [KNL]
                Set number of RCU callback-offload togglers.
                Zero (the default) disables toggling.

rcutorture.nocbs_toggle= [KNL]
                Set the delay in milliseconds between successive
                callback-offload toggling attempts.

rcutorture.nreaders= [KNL]
                Set number of RCU readers.  The value -1 selects
                N-1, where N is the number of CPUs.  A value
                "n" less than -1 selects N-n-2, where N is again
                the number of CPUs.  For example, -2 selects N
                (the number of CPUs), -3 selects N+1, and so on.

rcutorture.object_debug= [KNL]
                Enable debug-object double-call_rcu() testing.

rcutorture.onoff_holdoff= [KNL]
                Set time (s) after boot for CPU-hotplug testing.
```

```
rcutorture.onoff_interval= [KNL]
                Set time (jiffies) between CPU-hotplug operations,
                or zero to disable CPU-hotplug testing.

rcutorture.read_exit= [KNL]
                Set the number of read-then-exit kthreads used
                to test the interaction of RCU updaters and
                task-exit processing.

rcutorture.read_exit_burst= [KNL]
                The number of times in a given read-then-exit
                episode that a set of read-then-exit kthreads
                is spawned.

rcutorture.read_exit_delay= [KNL]
                The delay, in seconds, between successive
                read-then-exit testing episodes.

rcutorture.shuffle_interval= [KNL]
                Set task-shuffle interval (s).  Shuffling tasks
                allows some CPUs to go into dyntick-idle mode
                during the rcutorture test.

rcutorture.shutdown_secs= [KNL]
                Set time (s) after boot system shutdown.  This
                is useful for hands-off automated testing.

rcutorture.stall_cpu= [KNL]
                Duration of CPU stall (s) to test RCU CPU stall
                warnings, zero to disable.

rcutorture.stall_cpu_block= [KNL]
                Sleep while stalling if set.  This will result
                in warnings from preemptible RCU in addition to
                any other stall-related activity.  Note that
                in kernels built with CONFIG_PREEMPTION=n and
                CONFIG_PREEMPT_COUNT=y, this parameter will
                cause the CPU to pass through a quiescent state.
                Given CONFIG_PREEMPTION=n, this will suppress
                RCU CPU stall warnings, but will instead result
                in scheduling-while-atomic splats.

                Use of this module parameter results in splats.


rcutorture.stall_cpu_holdoff= [KNL]
                Time to wait (s) after boot before inducing stall.

rcutorture.stall_cpu_irqsoff= [KNL]
                Disable interrupts while stalling if set.
```

```
rcutorture.stall_gp_kthread= [KNL]
                Duration (s) of forced sleep within RCU
                grace-period kthread to test RCU CPU stall
                warnings, zero to disable.  If both stall_cpu
                and stall_gp_kthread are specified, the
                kthread is starved first, then the CPU.

rcutorture.stat_interval= [KNL]
                Time (s) between statistics printk()s.

rcutorture.stutter= [KNL]
                Time (s) to stutter testing, for example, specifying
                five seconds causes the test to run for five seconds,
                wait for five seconds, and so on.  This tests RCU's
                ability to transition abruptly to and from idle.

rcutorture.test_boost= [KNL]
                Test RCU priority boosting?  0=no, 1=maybe, 2=yes.
                "Maybe" means test if the RCU implementation
                under test support RCU priority boosting.

rcutorture.test_boost_duration= [KNL]
                Duration (s) of each individual boost test.

rcutorture.test_boost_interval= [KNL]
                Interval (s) between each boost test.

rcutorture.test_no_idle_hz= [KNL]
                Test RCU's dyntick-idle handling.  See also the
                rcutorture.shuffle_interval parameter.

rcutorture.torture_type= [KNL]
                Specify the RCU implementation to test.

rcutorture.verbose= [KNL]
                Enable additional printk() statements.

rcupdate.rcu_cpu_stall_ftrace_dump= [KNL]
                Dump ftrace buffer after reporting RCU CPU
                stall warning.

rcupdate.rcu_cpu_stall_notifiers= [KNL]
                Provide RCU CPU stall notifiers, but see the
                warnings in the RCU_CPU_STALL_NOTIFIER Kconfig
                option's help text.  TL;DR:  You almost certainly
                do not want rcupdate.rcu_cpu_stall_notifiers.

rcupdate.rcu_cpu_stall_suppress= [KNL]
                Suppress RCU CPU stall warning messages.
```

```
rcupdate.rcu_cpu_stall_suppress_at_boot= [KNL]
                Suppress RCU CPU stall warning messages and
                rcutorture writer stall warnings that occur
                during early boot, that is, during the time
                before the init task is spawned.

rcupdate.rcu_cpu_stall_timeout= [KNL]
                Set timeout for RCU CPU stall warning messages.
                The value is in seconds and the maximum allowed
                value is 300 seconds.

rcupdate.rcu_exp_cpu_stall_timeout= [KNL]
                Set timeout for expedited RCU CPU stall warning
                messages.  The value is in milliseconds
                and the maximum allowed value is 21000
                milliseconds. Please note that this value is
                adjusted to an arch timer tick resolution.
                Setting this to zero causes the value from
                rcupdate.rcu_cpu_stall_timeout to be used (after
                conversion from seconds to milliseconds).

rcupdate.rcu_cpu_stall_cputime= [KNL]
                Provide statistics on the cputime and count of
                interrupts and tasks during the sampling period. For
                multiple continuous RCU stalls, all sampling periods
                begin at half of the first RCU stall timeout.

rcupdate.rcu_exp_stall_task_details= [KNL]
                Print stack dumps of any tasks blocking the
                current expedited RCU grace period during an
                expedited RCU CPU stall warning.

rcupdate.rcu_expedited= [KNL]
                Use expedited grace-period primitives, for
                example, synchronize_rcu_expedited() instead
                of synchronize_rcu().  This reduces latency,
                but can increase CPU utilization, degrade
                real-time latency, and degrade energy efficiency.
                No effect on CONFIG_TINY_RCU kernels.

rcupdate.rcu_normal= [KNL]
                Use only normal grace-period primitives,
                for example, synchronize_rcu() instead of
                synchronize_rcu_expedited().  This improves
                real-time latency, CPU utilization, and
                energy efficiency, but can expose users to
                increased grace-period latency.  This parameter
                overrides rcupdate.rcu_expedited.  No effect on
                CONFIG_TINY_RCU kernels.

rcupdate.rcu_normal_after_boot= [KNL]
```

```
                Once boot has completed (that is, after
                rcu_end_inkernel_boot() has been invoked), use
                only normal grace-period primitives.  No effect
                on CONFIG_TINY_RCU kernels.

                But note that CONFIG_PREEMPT_RT=y kernels enables
                this kernel boot parameter, forcibly setting
                it to the value one, that is, converting any
                post-boot attempt at an expedited RCU grace
                period to instead use normal non-expedited
                grace-period processing.

        rcupdate.rcu_task_collapse_lim= [KNL]
                Set the maximum number of callbacks present
                at the beginning of a grace period that allows
                the RCU Tasks flavors to collapse back to using
                a single callback queue.  This switching only
                occurs when rcupdate.rcu_task_enqueue_lim is
                set to the default value of -1.

        rcupdate.rcu_task_contend_lim= [KNL]
                Set the minimum number of callback-queuing-time
                lock-contention events per jiffy required to
                cause the RCU Tasks flavors to switch to per-CPU
                callback queuing.  This switching only occurs
                when rcupdate.rcu_task_enqueue_lim is set to
                the default value of -1.

        rcupdate.rcu_task_enqueue_lim= [KNL]
                Set the number of callback queues to use for the
                RCU Tasks family of RCU flavors.  The default
                of -1 allows this to be automatically (and
                dynamically) adjusted.  This parameter is intended
                for use in testing.

        rcupdate.rcu_task_ipi_delay= [KNL]
                Set time in jiffies during which RCU tasks will
                avoid sending IPIs, starting with the beginning
                of a given grace period.  Setting a large
                number avoids disturbing real-time workloads,
                but lengthens grace periods.

        rcupdate.rcu_task_lazy_lim= [KNL]
                Number of callbacks on a given CPU that will
                cancel laziness on that CPU.  Use -1 to disable
                cancellation of laziness, but be advised that
                doing so increases the danger of OOM due to
                callback flooding.

        rcupdate.rcu_task_stall_info= [KNL]
                Set initial timeout in jiffies for RCU task stall
```

```
                 informational messages, which give some indication
                 of the problem for those not patient enough to
                 wait for ten minutes.  Informational messages are
                 only printed prior to the stall-warning message
                 for a given grace period. Disable with a value
                 less than or equal to zero.  Defaults to ten
                 seconds.  A change in value does not take effect
                 until the beginning of the next grace period.

rcupdate.rcu_task_stall_info_mult= [KNL]
                 Multiplier for time interval between successive
                 RCU task stall informational messages for a given
                 RCU tasks grace period.  This value is clamped
                 to one through ten, inclusive.  It defaults to
                 the value three, so that the first informational
                 message is printed 10 seconds into the grace
                 period, the second at 40 seconds, the third at
                 160 seconds, and then the stall warning at 600
                 seconds would prevent a fourth at 640 seconds.

rcupdate.rcu_task_stall_timeout= [KNL]
                 Set timeout in jiffies for RCU task stall
                 warning messages.  Disable with a value less
                 than or equal to zero.  Defaults to ten minutes.
                 A change in value does not take effect until
                 the beginning of the next grace period.

rcupdate.rcu_tasks_lazy_ms= [KNL]
                 Set timeout in milliseconds RCU Tasks asynchronous
                 callback batching for call_rcu_tasks().
                 A negative value will take the default.  A value
                 of zero will disable batching.  Batching is
                 always disabled for synchronize_rcu_tasks().

rcupdate.rcu_tasks_rude_lazy_ms= [KNL]
                 Set timeout in milliseconds RCU Tasks
                 Rude asynchronous callback batching for
                 call_rcu_tasks_rude().  A negative value
                 will take the default.  A value of zero will
                 disable batching.  Batching is always disabled
                 for synchronize_rcu_tasks_rude().

rcupdate.rcu_tasks_trace_lazy_ms= [KNL]
                 Set timeout in milliseconds RCU Tasks
                 Trace asynchronous callback batching for
                 call_rcu_tasks_trace().  A negative value
                 will take the default.  A value of zero will
                 disable batching.  Batching is always disabled
                 for synchronize_rcu_tasks_trace().

rcupdate.rcu_self_test= [KNL]
```

```
                        Run the RCU early boot self tests

        rdinit=         [KNL]
                        Format: <full_path>
                        Run specified binary instead of /init from the ramdisk,
                        used for early userspace startup. See initrd.

        rdrand=         [X86]
                        force - Override the decision by the kernel to hide the
                                advertisement of RDRAND support (this affects
                                certain AMD processors because of buggy BIOS
                                support, specifically around the suspend/resume
                                path).

        rdt=            [HW,X86,RDT]
                        Turn on/off individual RDT features. List is:
                        cmt, mbmtotal, mbmlocal, l3cat, l3cdp, l2cat, l2cdp,
                        mba, smba, bmec.
                        E.g. to turn on cmt and turn off mba use:
                                rdt=cmt,!mba

        reboot=         [KNL]
                        Format (x86 or x86_64):
                                [w[arm] | c[old] | h[ard] | s[oft] | g[pio]] |␣
↪d[efault] \
                                [[,]s[mp]#### \
                                [[,]b[ios] | a[cpi] | k[bd] | t[riple] | e[fi]␣
↪| p[ci]] \
                                [[,]f[orce]
                        Where reboot_mode is one of warm (soft) or cold (hard)␣
↪or gpio
                                        (prefix with 'panic_' to set mode for␣
↪panic
                                        reboot only),
                              reboot_type is one of bios, acpi, kbd, triple,␣
↪efi, or pci,
                              reboot_force is either force or not specified,
                              reboot_cpu is s[mp]#### with #### being the␣
↪processor
                                        to be used for rebooting.

        refscale.holdoff= [KNL]
                        Set test-start holdoff period.  The purpose of
                        this parameter is to delay the start of the
                        test until boot completes in order to avoid
                        interference.

        refscale.lookup_instances= [KNL]
                        Number of data elements to use for the forms of
                        SLAB_TYPESAFE_BY_RCU testing.  A negative number
                        is negated and multiplied by nr_cpu_ids, while
```

```
                        zero specifies nr_cpu_ids.

        refscale.loops= [KNL]
                        Set the number of loops over the synchronization
                        primitive under test.  Increasing this number
                        reduces noise due to loop start/end overhead,
                        but the default has already reduced the per-pass
                        noise to a handful of picoseconds on ca. 2020
                        x86 laptops.

        refscale.nreaders= [KNL]
                        Set number of readers.  The default value of -1
                        selects N, where N is roughly 75% of the number
                        of CPUs.  A value of zero is an interesting choice.

        refscale.nruns= [KNL]
                        Set number of runs, each of which is dumped onto
                        the console log.

        refscale.readdelay= [KNL]
                        Set the read-side critical-section duration,
                        measured in microseconds.

        refscale.scale_type= [KNL]
                        Specify the read-protection implementation to test.

        refscale.shutdown= [KNL]
                        Shut down the system at the end of the performance
                        test.  This defaults to 1 (shut it down) when
                        refscale is built into the kernel and to 0 (leave
                        it running) when refscale is built as a module.

        refscale.verbose= [KNL]
                        Enable additional printk() statements.

        refscale.verbose_batched= [KNL]
                        Batch the additional printk() statements.  If zero
                        (the default) or negative, print everything. ␣
↪Otherwise,
                        print every Nth verbose statement, where N is the value
                        specified.

        regulator_ignore_unused
                        [REGULATOR]
                        Prevents regulator framework from disabling regulators
                        that are unused, due no driver claiming them. This may
                        be useful for debug and development, but should not be
                        needed on a platform with proper driver support.

        relax_domain_level=
                        [KNL, SMP] Set scheduler's default relax_domain_level.
```

```
                        See Documentation/admin-guide/cgroup-v1/cpusets.rst.

        reserve=        [KNL,BUGS] Force kernel to ignore I/O ports or memory
                        Format: <base1>,<size1>[,<base2>,<size2>,...]
                        Reserve I/O ports or memory so the kernel won't use
                        them.  If <base> is less than 0x10000, the region
                        is assumed to be I/O ports; otherwise it is memory.

        reservetop=     [X86-32]
                        Format: nn[KMG]
                        Reserves a hole at the top of the kernel virtual
                        address space.

        reset_devices   [KNL] Force drivers to reset the underlying device
                        during initialization.

        resume=         [SWSUSP]
                        Specify the partition device for software suspend
                        Format:
                        {/dev/<dev> | PARTUUID=<uuid> | <int>:<int> | <hex>}

        resume_offset=  [SWSUSP]
                        Specify the offset from the beginning of the partition
                        given by "resume=" at which the swap header is located,
                        in <PAGE_SIZE> units (needed only for swap files).
                        See  Documentation/power/swsusp-and-swap-files.rst

        resumedelay=    [HIBERNATION] Delay (in seconds) to pause before␣
↪attempting to
                        read the resume files

        resumewait      [HIBERNATION] Wait (indefinitely) for resume device to␣
↪show up.
                        Useful for devices that are detected asynchronously
                        (e.g. USB and MMC devices).

        retain_initrd   [RAM] Keep initrd memory after extraction. After boot,␣
↪it will
                        be accessible via /sys/firmware/initrd.

        retbleed=       [X86] Control mitigation of RETBleed (Arbitrary
                        Speculative Code Execution with Return Instructions)
                        vulnerability.

                        AMD-based UNRET and IBPB mitigations alone do not stop
                        sibling threads from influencing the predictions of␣
↪other
                        sibling threads. For that reason, STIBP is used on pro-
                        cessors that support it, and mitigate SMT on processors
                        that don't.
```

```
                        off          - no mitigation
                        auto         - automatically select a migitation
                        auto,nosmt   - automatically select a mitigation,
                                       disabling SMT if necessary for
                                       the full mitigation (only on Zen1
                                       and older without STIBP).
                        ibpb         - On AMD, mitigate short speculation
                                       windows on basic block boundaries too.
                                       Safe, highest perf impact. It also
                                       enables STIBP if present. Not suitable
                                       on Intel.
                        ibpb,nosmt   - Like "ibpb" above but will disable SMT
                                       when STIBP is not available. This is
                                       the alternative for systems which do not
                                       have STIBP.
                        unret        - Force enable untrained return thunks,
                                       only effective on AMD f15h-f17h based
                                       systems.
                        unret,nosmt  - Like unret, but will disable SMT when␣
↪STIBP
                                       is not available. This is the␣
↪alternative for
                                       systems which do not have STIBP.

                        Selecting 'auto' will choose a mitigation method at run
                        time according to the CPU.

                        Not specifying this option is equivalent to␣
↪retbleed=auto.

        rfkill.default_state=
                0       "airplane mode".  All wifi, bluetooth, wimax, gps, fm,
                        etc. communication is blocked by default.
                1       Unblocked.

        rfkill.master_switch_mode=
                0       The "airplane mode" button does nothing.
                1       The "airplane mode" button toggles between everything
                        blocked and the previous configuration.
                2       The "airplane mode" button toggles between everything
                        blocked and everything unblocked.

        rhash_entries=  [KNL,NET]
                        Set number of hash buckets for route cache

        ring3mwait=disable
                        [KNL] Disable ring 3 MONITOR/MWAIT feature on supported
                        CPUs.

        riscv_isa_fallback [RISCV]
                        When CONFIG_RISCV_ISA_FALLBACK is not enabled, permit
```

```
                falling back to detecting extension support by parsing
                "riscv,isa" property on devicetree systems when the
                replacement properties are not found. See the Kconfig
                entry for RISCV_ISA_FALLBACK.

ro              [KNL] Mount root device read-only on boot

rodata=         [KNL]
        on      Mark read-only kernel memory as read-only (default).
        off     Leave read-only kernel memory writable for debugging.
        full    Mark read-only kernel memory and aliases as read-only
                [arm64]

rockchip.usb_uart
                Enable the uart passthrough on the designated usb port
                on Rockchip SoCs. When active, the signals of the
                debug-uart get routed to the D+ and D- pins of the usb
                port and the regular usb controller gets disabled.

root=           [KNL] Root filesystem
                Usually this a a block device specifier of some kind,
                see the early_lookup_bdev comment in
                block/early-lookup.c for details.
                Alternatively this can be "ram" for the legacy initial
                ramdisk, "nfs" and "cifs" for root on a network file
                system, or "mtd" and "ubi" for mounting from raw flash.

rootdelay=      [KNL] Delay (in seconds) to pause before attempting to
                mount the root filesystem

rootflags=      [KNL] Set root filesystem mount option string

rootfstype=     [KNL] Set root filesystem type

rootwait        [KNL] Wait (indefinitely) for root device to show up.
                Useful for devices that are detected asynchronously
                (e.g. USB and MMC devices).

rootwait=       [KNL] Maximum time (in seconds) to wait for root device
                to show up before attempting to mount the root
                filesystem.

rproc_mem=nn[KMG][@address]
                [KNL,ARM,CMA] Remoteproc physical memory block.
                Memory area to be used by remote processor image,
                managed by CMA.

rw              [KNL] Mount root device read-write on boot

S               [KNL] Run init in single mode
```

```
s390_iommu=     [HW,S390]
                Set s390 IOTLB flushing mode
        strict
                With strict flushing every unmap operation will result
                in an IOTLB flush. Default is lazy flushing before
                reuse, which is faster. Deprecated, equivalent to
                iommu.strict=1.

s390_iommu_aperture=    [KNL,S390]
                Specifies the size of the per device DMA address space
                accessible through the DMA and IOMMU APIs as a decimal
                factor of the size of main memory.
                The default is 1 meaning that one can concurrently use
                as many DMA addresses as physical memory is installed,
                if supported by hardware, and thus map all of memory
                once. With a value of 2 one can map all of memory twice
                and so on. As a special case a factor of 0 imposes no
                restrictions other than those given by hardware at the
                cost of significant additional memory use for tables.

sa1100ir        [NET]
                See drivers/net/irda/sa1100_ir.c.

sched_verbose   [KNL] Enables verbose scheduler debug messages.

schedstats=     [KNL,X86] Enable or disable scheduled statistics.
                Allowed values are enable and disable. This feature
                incurs a small amount of overhead in the scheduler
                but is useful for debugging and performance tuning.

sched_thermal_decay_shift=
                [KNL, SMP] Set a decay shift for scheduler thermal
                pressure signal. Thermal pressure signal follows the
                default decay period of other scheduler pelt
                signals(usually 32 ms but configurable). Setting
                sched_thermal_decay_shift will left shift the decay
                period for the thermal pressure signal by the shift
                value.
                i.e. with the default pelt decay period of 32 ms
                sched_thermal_decay_shift   thermal pressure decay pr
                        1                       64 ms
                        2                       128 ms
                and so on.
                Format: integer between 0 and 10
                Default is 0.

scftorture.holdoff= [KNL]
                Number of seconds to hold off before starting
                test.  Defaults to zero for module insertion and
                to 10 seconds for built-in smp_call_function()
                tests.
```

```
scftorture.longwait= [KNL]
                Request ridiculously long waits randomly selected
                up to the chosen limit in seconds.  Zero (the
                default) disables this feature.  Please note
                that requesting even small non-zero numbers of
                seconds can result in RCU CPU stall warnings,
                softlockup complaints, and so on.

scftorture.nthreads= [KNL]
                Number of kthreads to spawn to invoke the
                smp_call_function() family of functions.
                The default of -1 specifies a number of kthreads
                equal to the number of CPUs.

scftorture.onoff_holdoff= [KNL]
                Number seconds to wait after the start of the
                test before initiating CPU-hotplug operations.

scftorture.onoff_interval= [KNL]
                Number seconds to wait between successive
                CPU-hotplug operations.  Specifying zero (which
                is the default) disables CPU-hotplug operations.

scftorture.shutdown_secs= [KNL]
                The number of seconds following the start of the
                test after which to shut down the system.  The
                default of zero avoids shutting down the system.
                Non-zero values are useful for automated tests.

scftorture.stat_interval= [KNL]
                The number of seconds between outputting the
                current test statistics to the console.  A value
                of zero disables statistics output.

scftorture.stutter_cpus= [KNL]
                The number of jiffies to wait between each change
                to the set of CPUs under test.

scftorture.use_cpus_read_lock= [KNL]
                Use use_cpus_read_lock() instead of the default
                preempt_disable() to disable CPU hotplug
                while invoking one of the smp_call_function*()
                functions.

scftorture.verbose= [KNL]
                Enable additional printk() statements.

scftorture.weight_single= [KNL]
                The probability weighting to use for the
                smp_call_function_single() function with a zero
```

```
                        "wait" parameter.  A value of -1 selects the
                        default if all other weights are -1.  However,
                        if at least one weight has some other value, a
                        value of -1 will instead select a weight of zero.

        scftorture.weight_single_wait= [KNL]
                        The probability weighting to use for the
                        smp_call_function_single() function with a
                        non-zero "wait" parameter.  See weight_single.

        scftorture.weight_many= [KNL]
                        The probability weighting to use for the
                        smp_call_function_many() function with a zero
                        "wait" parameter.  See weight_single.
                        Note well that setting a high probability for
                        this weighting can place serious IPI load
                        on the system.

        scftorture.weight_many_wait= [KNL]
                        The probability weighting to use for the
                        smp_call_function_many() function with a
                        non-zero "wait" parameter.  See weight_single
                        and weight_many.

        scftorture.weight_all= [KNL]
                        The probability weighting to use for the
                        smp_call_function_all() function with a zero
                        "wait" parameter.  See weight_single and
                        weight_many.

        scftorture.weight_all_wait= [KNL]
                        The probability weighting to use for the
                        smp_call_function_all() function with a
                        non-zero "wait" parameter.  See weight_single
                        and weight_many.

        skew_tick=      [KNL] Offset the periodic timer tick per cpu to␣
↪mitigate
                        xtime_lock contention on larger systems, and/or RCU␣
↪lock
                        contention on all systems with CONFIG_MAXSMP set.
                        Format: { "0" | "1" }
                        0 -- disable. (may be 1 via CONFIG_CMDLINE="skew_
↪tick=1"
                        1 -- enable.
                        Note: increases power consumption, thus should only be
                        enabled if running jitter sensitive (HPC/RT) workloads.

        security=       [SECURITY] Choose a legacy "major" security module to
                        enable at boot. This has been deprecated by the
                        "lsm=" parameter.
```

```
        selinux=        [SELINUX] Disable or enable SELinux at boot time.
                        Format: { "0" | "1" }
                        See security/selinux/Kconfig help text.
                        0 -- disable.
                        1 -- enable.
                        Default value is 1.

        serialnumber    [BUGS=X86-32]

        sev=option[,option...] [X86-64] See Documentation/arch/x86/x86_64/
↪boot-options.rst

        shapers=        [NET]
                        Maximal number of shapers.

        show_lapic=     [APIC,X86] Advanced Programmable Interrupt Controller
                        Limit apic dumping. The parameter defines the maximal
                        number of local apics being dumped. Also it is possible
                        to set it to "all" by meaning -- no limit here.
                        Format: { 1 (default) | 2 | ... | all }.
                        The parameter valid if only apic=debug or
                        apic=verbose is specified.
                        Example: apic=debug show_lapic=all

        simeth=         [IA-64]
        simscsi=

        slram=          [HW,MTD]

        slab_merge      [MM]
                        Enable merging of slabs with similar size when the
                        kernel is built without CONFIG_SLAB_MERGE_DEFAULT.

        slab_nomerge    [MM]
                        Disable merging of slabs with similar size. May be
                        necessary if there is some reason to distinguish
                        allocs to different slabs, especially in hardened
                        environments where the risk of heap overflows and
                        layout control by attackers can usually be
                        frustrated by disabling merging. This will reduce
                        most of the exposure of a heap attack to a single
                        cache (risks via metadata attacks are mostly
                        unchanged). Debug options disable merging on their
                        own.
                        For more information see Documentation/mm/slub.rst.

        slab_max_order= [MM, SLAB]
                        Determines the maximum allowed order for slabs.
                        A high setting may cause OOMs due to memory
                        fragmentation.  Defaults to 1 for systems with
```

```
                more than 32MB of RAM, 0 otherwise.

slub_debug[=options[,slabs][;[options[,slabs]]...]      [MM, SLUB]
                Enabling slub_debug allows one to determine the
                culprit if slab objects become corrupted. Enabling
                slub_debug can create guard zones around objects and
                may poison objects when not in use. Also tracks the
                last alloc / free. For more information see
                Documentation/mm/slub.rst.

slub_max_order= [MM, SLUB]
                Determines the maximum allowed order for slabs.
                A high setting may cause OOMs due to memory
                fragmentation. For more information see
                Documentation/mm/slub.rst.

slub_min_objects=       [MM, SLUB]
                The minimum number of objects per slab. SLUB will
                increase the slab order up to slub_max_order to
                generate a sufficiently large slab able to contain
                the number of objects indicated. The higher the number
                of objects the smaller the overhead of tracking slabs
                and the less frequently locks need to be acquired.
                For more information see Documentation/mm/slub.rst.

slub_min_order= [MM, SLUB]
                Determines the minimum page order for slabs. Must be
                lower than slub_max_order.
                For more information see Documentation/mm/slub.rst.

slub_merge      [MM, SLUB]
                Same with slab_merge.

slub_nomerge    [MM, SLUB]
                Same with slab_nomerge. This is supported for legacy.
                See slab_nomerge for more information.

smart2=         [HW]
                Format: <io1>[,<io2>[,...,<io8>]]

smp.csd_lock_timeout= [KNL]
                Specify the period of time in milliseconds
                that smp_call_function() and friends will wait
                for a CPU to release the CSD lock.  This is
                useful when diagnosing bugs involving CPUs
                disabling interrupts for extended periods
                of time.  Defaults to 5,000 milliseconds, and
                setting a value of zero disables this feature.
                This feature may be more efficiently disabled
                using the csdlock_debug- kernel parameter.
```

```
        smp.panic_on_ipistall= [KNL]
                        If a csd_lock_timeout extends for more than
                        the specified number of milliseconds, panic the
                        system.  By default, let CSD-lock acquisition
                        take as long as they take.  Specifying 300,000
                        for this value provides a 5-minute timeout.

        smsc-ircc2.nopnp        [HW] Don't use PNP to discover SMC devices
        smsc-ircc2.ircc_cfg=    [HW] Device configuration I/O port
        smsc-ircc2.ircc_sir=    [HW] SIR base I/O port
        smsc-ircc2.ircc_fir=    [HW] FIR base I/O port
        smsc-ircc2.ircc_irq=    [HW] IRQ line
        smsc-ircc2.ircc_dma=    [HW] DMA channel
        smsc-ircc2.ircc_transceiver= [HW] Transceiver type:
                                0: Toshiba Satellite 1800 (GP data pin select)
                                1: Fast pin select (default)
                                2: ATC IRMode

        smt=            [KNL,MIPS,S390] Set the maximum number of threads␣
↪(logical
                        CPUs) to use per physical CPU on systems capable of
                        symmetric multithreading (SMT). Will be capped to the
                        actual hardware limit.
                        Format: <integer>
                        Default: -1 (no limit)

        softlockup_panic=
                        [KNL] Should the soft-lockup detector generate panics.
                        Format: 0 | 1

                        A value of 1 instructs the soft-lockup detector
                        to panic the machine when a soft-lockup occurs. It is
                        also controlled by the kernel.softlockup_panic sysctl
                        and CONFIG_BOOTPARAM_SOFTLOCKUP_PANIC, which is the
                        respective build-time switch to that functionality.

        softlockup_all_cpu_backtrace=
                        [KNL] Should the soft-lockup detector generate
                        backtraces on all cpus.
                        Format: 0 | 1

        sonypi.*=       [HW] Sony Programmable I/O Control Device driver
                        See Documentation/admin-guide/laptops/sonypi.rst

        spectre_v2=     [X86] Control mitigation of Spectre variant 2
                        (indirect branch speculation) vulnerability.
                        The default operation protects the kernel from
                        user space attacks.

                        on   - unconditionally enable, implies
                               spectre_v2_user=on
```

```
                    off  - unconditionally disable, implies
                           spectre_v2_user=off
                    auto - kernel detects whether your CPU model is
                           vulnerable

                    Selecting 'on' will, and 'auto' may, choose a
                    mitigation method at run time according to the
                    CPU, the available microcode, the setting of the
                    CONFIG_RETPOLINE configuration option, and the
                    compiler with which the kernel was built.

                    Selecting 'on' will also enable the mitigation
                    against user space to user space task attacks.

                    Selecting 'off' will disable both the kernel and
                    the user space protections.

                    Specific mitigations can also be selected manually:

                    retpoline         - replace indirect branches
                    retpoline,generic - Retpolines
                    retpoline,lfence  - LFENCE; indirect branch
                    retpoline,amd     - alias for retpoline,lfence
                    eibrs             - Enhanced/Auto IBRS
                    eibrs,retpoline   - Enhanced/Auto IBRS + Retpolines
                    eibrs,lfence      - Enhanced/Auto IBRS + LFENCE
                    ibrs              - use IBRS to protect kernel

                    Not specifying this option is equivalent to
                    spectre_v2=auto.

spectre_v2_user=
                    [X86] Control mitigation of Spectre variant 2
                    (indirect branch speculation) vulnerability between
                    user space tasks

                    on      - Unconditionally enable mitigations. Is
                              enforced by spectre_v2=on

                    off     - Unconditionally disable mitigations. Is
                              enforced by spectre_v2=off

                    prctl   - Indirect branch speculation is enabled,
                              but mitigation can be enabled via prctl
                              per thread.  The mitigation control state
                              is inherited on fork.

                    prctl,ibpb
                            - Like "prctl" above, but only STIBP is
                              controlled per thread. IBPB is issued
                              always when switching between different user
```

```
                                  space processes.

                        seccomp
                                - Same as "prctl" above, but all seccomp
                                  threads will enable the mitigation unless
                                  they explicitly opt out.

                        seccomp,ibpb
                                - Like "seccomp" above, but only STIBP is
                                  controlled per thread. IBPB is issued
                                  always when switching between different
                                  user space processes.

                        auto    - Kernel selects the mitigation depending on
                                  the available CPU features and vulnerability.

                        Default mitigation: "prctl"

                        Not specifying this option is equivalent to
                        spectre_v2_user=auto.

        spec_rstack_overflow=
                        [X86] Control RAS overflow mitigation on AMD Zen CPUs

                        off             - Disable mitigation
                        microcode       - Enable microcode mitigation only
                        safe-ret        - Enable sw-only safe RET mitigation␣
↪(default)
                        ibpb            - Enable mitigation by issuing IBPB on
                                          kernel entry
                        ibpb-vmexit     - Issue IBPB only on VMEXIT
                                          (cloud-specific mitigation)

        spec_store_bypass_disable=
                        [HW] Control Speculative Store Bypass (SSB) Disable␣
↪mitigation
                        (Speculative Store Bypass vulnerability)

                        Certain CPUs are vulnerable to an exploit against a
                        a common industry wide performance optimization known
                        as "Speculative Store Bypass" in which recent stores
                        to the same memory location may not be observed by
                        later loads during speculative execution. The idea
                        is that such stores are unlikely and that they can
                        be detected prior to instruction retirement at the
                        end of a particular speculation execution window.

                        In vulnerable processors, the speculatively forwarded
                        store can be used in a cache side channel attack, for
                        example to read memory to which the attacker does not
                        directly have access (e.g. inside sandboxed code).
```

```
                        This parameter controls whether the Speculative Store
                        Bypass optimization is used.

                        On x86 the options are:

                        on      - Unconditionally disable Speculative Store␣
↪Bypass
                        off     - Unconditionally enable Speculative Store␣
↪Bypass
                        auto    - Kernel detects whether the CPU model␣
↪contains an
                                  implementation of Speculative Store Bypass␣
↪and
                                  picks the most appropriate mitigation. If the
                                  CPU is not vulnerable, "off" is selected. If␣
↪the
                                  CPU is vulnerable the default mitigation is
                                  architecture and Kconfig dependent. See␣
↪below.
                        prctl   - Control Speculative Store Bypass per thread
                                  via prctl. Speculative Store Bypass is␣
↪enabled
                                  for a process by default. The state of the␣
↪control
                                  is inherited on fork.
                        seccomp - Same as "prctl" above, but all seccomp␣
↪threads
                                  will disable SSB unless they explicitly opt␣
↪out.

                        Default mitigations:
                        X86:    "prctl"

                        On powerpc the options are:

                        on,auto - On Power8 and Power9 insert a␣
↪store-forwarding
                                  barrier on kernel entry and exit. On Power7
                                  perform a software flush on kernel entry and
                                  exit.
                        off     - No action.

                        Not specifying this option is equivalent to
                        spec_store_bypass_disable=auto.

        spia_io_base=   [HW,MTD]
        spia_fio_base=
        spia_pedr=
        spia_peddr=
```

```
        split_lock_detect=
                        [X86] Enable split lock detection or bus lock detection

                        When enabled (and if hardware support is present),␣
↪atomic
                        instructions that access data across cache line
                        boundaries will result in an alignment check exception
                        for split lock detection or a debug exception for
                        bus lock detection.

                        off     - not enabled

                        warn    - the kernel will emit rate-limited warnings
                                  about applications triggering the #AC
                                  exception or the #DB exception. This mode is
                                  the default on CPUs that support split lock
                                  detection or bus lock detection. Default
                                  behavior is by #AC if both features are
                                  enabled in hardware.

                        fatal   - the kernel will send SIGBUS to applications
                                  that trigger the #AC exception or the #DB
                                  exception. Default behavior is by #AC if
                                  both features are enabled in hardware.

                        ratelimit:N -
                                  Set system wide rate limit to N bus locks
                                  per second for bus lock detection.
                                  0 < N <= 1000.

                                  N/A for split lock detection.


                        If an #AC exception is hit in the kernel or in
                        firmware (i.e. not while executing in user mode)
                        the kernel will oops in either "warn" or "fatal"
                        mode.

                        #DB exception for bus lock is triggered only when
                        CPL > 0.

        srbds=          [X86,INTEL]
                        Control the Special Register Buffer Data Sampling
                        (SRBDS) mitigation.

                        Certain CPUs are vulnerable to an MDS-like
                        exploit which can leak bits from the random
                        number generator.

                        By default, this issue is mitigated by
                        microcode.  However, the microcode fix can cause
```

```
                the RDRAND and RDSEED instructions to become
                much slower.  Among other effects, this will
                result in reduced throughput from /dev/urandom.

                The microcode mitigation can be disabled with
                the following option:

                off:    Disable mitigation and remove
                        performance impact to RDRAND and RDSEED

srcutree.big_cpu_lim [KNL]
                Specifies the number of CPUs constituting a
                large system, such that srcu_struct structures
                should immediately allocate an srcu_node array.
                This kernel-boot parameter defaults to 128,
                but takes effect only when the low-order four
                bits of srcutree.convert_to_big is equal to 3
                (decide at boot).

srcutree.convert_to_big [KNL]
                Specifies under what conditions an SRCU tree
                srcu_struct structure will be converted to big
                form, that is, with an rcu_node tree:

                           0:  Never.
                           1:  At init_srcu_struct() time.
                           2:  When rcutorture decides to.
                           3:  Decide at boot time (default).
                        0x1X:  Above plus if high contention.

                Either way, the srcu_node tree will be sized based
                on the actual runtime number of CPUs (nr_cpu_ids)
                instead of the compile-time CONFIG_NR_CPUS.

srcutree.counter_wrap_check [KNL]
                Specifies how frequently to check for
                grace-period sequence counter wrap for the
                srcu_data structure's ->srcu_gp_seq_needed field.
                The greater the number of bits set in this kernel
                parameter, the less frequently counter wrap will
                be checked for.  Note that the bottom two bits
                are ignored.

srcutree.exp_holdoff [KNL]
                Specifies how many nanoseconds must elapse
                since the end of the last SRCU grace period for
                a given srcu_struct until the next normal SRCU
                grace period will be considered for automatic
                expediting.  Set to zero to disable automatic
                expediting.
```

```
srcutree.srcu_max_nodelay [KNL]
                Specifies the number of no-delay instances
                per jiffy for which the SRCU grace period
                worker thread will be rescheduled with zero
                delay. Beyond this limit, worker thread will
                be rescheduled with a sleep delay of one jiffy.

srcutree.srcu_max_nodelay_phase [KNL]
                Specifies the per-grace-period phase, number of
                non-sleeping polls of readers. Beyond this limit,
                grace period worker thread will be rescheduled
                with a sleep delay of one jiffy, between each
                rescan of the readers, for a grace period phase.

srcutree.srcu_retry_check_delay [KNL]
                Specifies number of microseconds of non-sleeping
                delay between each non-sleeping poll of readers.

srcutree.small_contention_lim [KNL]
                Specifies the number of update-side contention
                events per jiffy will be tolerated before
                initiating a conversion of an srcu_struct
                structure to big form.  Note that the value of
                srcutree.convert_to_big must have the 0x10 bit
                set for contention-based conversions to occur.

ssbd=           [ARM64,HW]
                Speculative Store Bypass Disable control

                On CPUs that are vulnerable to the Speculative
                Store Bypass vulnerability and offer a
                firmware based mitigation, this parameter
                indicates how the mitigation should be used:

                force-on:  Unconditionally enable mitigation for
                           for both kernel and userspace
                force-off: Unconditionally disable mitigation for
                           for both kernel and userspace
                kernel:    Always enable mitigation in the
                           kernel, and offer a prctl interface
                           to allow userspace to register its
                           interest in being mitigated too.

stack_guard_gap=        [MM]
                override the default stack gap protection. The value
                is in page units and it defines how many pages prior
                to (for stacks growing down) resp. after (for stacks
                growing up) the main stack are reserved for no other
                mapping. Default value is 256 pages.

stack_depot_disable= [KNL]
```

```
                        Setting this to true through kernel command line will
                        disable the stack depot thereby saving the static␣
↪memory
                        consumed by the stack hash table. By default this is␣
↪set
                        to false.

        stacktrace      [FTRACE]
                        Enabled the stack tracer on boot up.

        stacktrace_filter=[function-list]
                        [FTRACE] Limit the functions that the stack tracer
                        will trace at boot up. function-list is a␣
↪comma-separated
                        list of functions. This list can be changed at run
                        time by the stack_trace_filter file in the debugfs
                        tracing directory. Note, this enables stack tracing
                        and the stacktrace above is not needed.

        sti=            [PARISC,HW]
                        Format: <num>
                        Set the STI (builtin display/keyboard on the HP-PARISC
                        machines) console (graphic card) which should be used
                        as the initial boot-console.
                        See also comment in drivers/video/console/sticore.c.

        sti_font=       [HW]
                        See comment in drivers/video/console/sticore.c.

        stifb=          [HW]
                        Format: bpp:<bpp1>[:<bpp2>[:<bpp3>...]]

        strict_sas_size=
                        [X86]
                        Format: <bool>
                        Enable or disable strict sigaltstack size checks
                        against the required signal frame size which
                        depends on the supported FPU features. This can
                        be used to filter out binaries which have
                        not yet been made aware of AT_MINSIGSTKSZ.

        stress_hpt      [PPC]
                        Limits the number of kernel HPT entries in the hash
                        page table to increase the rate of hash page table
                        faults on kernel addresses.

        stress_slb      [PPC]
                        Limits the number of kernel SLB entries, and flushes
                        them frequently to increase the rate of SLB faults
                        on kernel addresses.
```

```
        sunrpc.min_resvport=
        sunrpc.max_resvport=
                        [NFS,SUNRPC]
                        SunRPC servers often require that client requests
                        originate from a privileged port (i.e. a port in the
                        range 0 < portnr < 1024).
                        An administrator who wishes to reserve some of these
                        ports for other uses may adjust the range that the
                        kernel's sunrpc client considers to be privileged
                        using these two parameters to set the minimum and
                        maximum port values.

        sunrpc.svc_rpc_per_connection_limit=
                        [NFS,SUNRPC]
                        Limit the number of requests that the server will
                        process in parallel from a single connection.
                        The default value is 0 (no limit).

        sunrpc.pool_mode=
                        [NFS]
                        Control how the NFS server code allocates CPUs to
                        service thread pools.  Depending on how many NICs
                        you have and where their interrupts are bound, this
                        option will affect which CPUs will do NFS serving.
                        Note: this parameter cannot be changed while the
                        NFS server is running.

                        auto        the server chooses an appropriate mode
                                    automatically using heuristics
                        global      a single global pool contains all CPUs
                        percpu      one pool for each CPU
                        pernode     one pool for each NUMA node (equivalent
                                    to global on non-NUMA machines)

        sunrpc.tcp_slot_table_entries=
        sunrpc.udp_slot_table_entries=
                        [NFS,SUNRPC]
                        Sets the upper limit on the number of simultaneous
                        RPC calls that can be sent from the client to a
                        server. Increasing these values may allow you to
                        improve throughput, but will also increase the
                        amount of memory reserved for use by the client.

        suspend.pm_test_delay=
                        [SUSPEND]
                        Sets the number of seconds to remain in a suspend test
                        mode before resuming the system (see
                        /sys/power/pm_test). Only available when CONFIG_PM_
↪DEBUG
                        is set. Default value is 5.
```

```
svm=            [PPC]
                Format: { on | off | y | n | 1 | 0 }
                This parameter controls use of the Protected
                Execution Facility on pSeries.

swiotlb=        [ARM,IA-64,PPC,MIPS,X86]
                Format: { <int> [,<int>] | force | noforce }
                <int> -- Number of I/O TLB slabs
                <int> -- Second integer after comma. Number of swiotlb
                         areas with their own lock. Will be rounded up
                         to a power of 2.
                force -- force using of bounce buffers even if they
                         wouldn't be automatically used by the kernel
                noforce -- Never use bounce buffers (for debugging)

switches=       [HW,M68k]

sysctl.*=       [KNL]
                Set a sysctl parameter, right before loading the init
                process, as if the value was written to the respective
                /proc/sys/... file. Both '.' and '/' are recognized as
                separators. Unrecognized parameters and invalid values
                are reported in the kernel log. Sysctls registered
                later by a loaded module cannot be set this way.
                Example: sysctl.vm.swappiness=40

sysrq_always_enabled
                [KNL]
                Ignore sysrq setting - this boot parameter will
                neutralize any effect of /proc/sys/kernel/sysrq.
                Useful for debugging.

tcpmhash_entries= [KNL,NET]
                Set the number of tcp_metrics_hash slots.
                Default value is 8192 or 16384 depending on total
                ram pages. This is used to specify the TCP metrics
                cache size. See Documentation/networking/ip-sysctl.rst
                "tcp_no_metrics_save" section for more details.

tdfx=           [HW,DRM]

test_suspend=   [SUSPEND]
                Format: { "mem" | "standby" | "freeze" }[,N]
                Specify "mem" (for Suspend-to-RAM) or "standby" (for
                standby suspend) or "freeze" (for suspend type freeze)
                as the system sleep state during system startup with
                the optional capability to repeat N number of times.
                The system is woken from this state using a
                wakeup-capable RTC alarm.

thash_entries=  [KNL,NET]
```

```
                        Set number of hash buckets for TCP connection

        thermal.act=    [HW,ACPI]
                        -1: disable all active trip points in all thermal zones
                        <degrees C>: override all lowest active trip points

        thermal.crt=    [HW,ACPI]
                        -1: disable all critical trip points in all thermal␣
↪zones
                        <degrees C>: override all critical trip points

        thermal.off=    [HW,ACPI]
                        1: disable ACPI thermal control

        thermal.psv=    [HW,ACPI]
                        -1: disable all passive trip points
                        <degrees C>: override all passive trip points to this
                        value

        thermal.tzp=    [HW,ACPI]
                        Specify global default ACPI thermal zone polling rate
                        <deci-seconds>: poll all this frequency
                        0: no polling (default)

        threadirqs      [KNL]
                        Force threading of all interrupt handlers except those
                        marked explicitly IRQF_NO_THREAD.

        topology=       [S390]
                        Format: {off | on}
                        Specify if the kernel should make use of the cpu
                        topology information if the hardware supports this.
                        The scheduler will make use of this information and
                        e.g. base its process migration decisions on it.
                        Default is on.

        topology_updates= [KNL, PPC, NUMA]
                        Format: {off}
                        Specify if the kernel should ignore (off)
                        topology updates sent by the hypervisor to this
                        LPAR.

        torture.disable_onoff_at_boot= [KNL]
                        Prevent the CPU-hotplug component of torturing
                        until after init has spawned.

        torture.ftrace_dump_at_shutdown= [KNL]
                        Dump the ftrace buffer at torture-test shutdown,
                        even if there were no errors.  This can be a
                        very costly operation when many torture tests
                        are running concurrently, especially on systems
```

```
                        with rotating-rust storage.

        torture.verbose_sleep_frequency= [KNL]
                        Specifies how many verbose printk()s should be
                        emitted between each sleep.  The default of zero
                        disables verbose-printk() sleeping.

        torture.verbose_sleep_duration= [KNL]
                        Duration of each verbose-printk() sleep in jiffies.

        tp720=          [HW,PS2]

        tpm_suspend_pcr=[HW,TPM]
                        Format: integer pcr id
                        Specify that at suspend time, the tpm driver
                        should extend the specified pcr with zeros,
                        as a workaround for some chips which fail to
                        flush the last written pcr on TPM_SaveState.
                        This will guarantee that all the other pcrs
                        are saved.

        tpm_tis.interrupts= [HW,TPM]
                        Enable interrupts for the MMIO based physical layer
                        for the FIFO interface. By default it is set to false
                        (0). For more information about TPM hardware interfaces
                        defined by Trusted Computing Group (TCG) see
                        https://trustedcomputinggroup.org/resource/
↪pc-client-platform-tpm-profile-ptp-specification/

        tp_printk       [FTRACE]
                        Have the tracepoints sent to printk as well as the
                        tracing ring buffer. This is useful for early boot up
                        where the system hangs or reboots and does not give the
                        option for reading the tracing buffer or performing a
                        ftrace_dump_on_oops.

                        To turn off having tracepoints sent to printk,
                         echo 0 > /proc/sys/kernel/tracepoint_printk
                        Note, echoing 1 into this file without the
                        tracepoint_printk kernel cmdline option has no effect.

                        The tp_printk_stop_on_boot (see below) can also be used
                        to stop the printing of events to console at
                        late_initcall_sync.

                        ** CAUTION **

                        Having tracepoints sent to printk() and activating high
                        frequency tracepoints such as irq or sched, can cause
                        the system to live lock.
```

```
        tp_printk_stop_on_boot [FTRACE]
                        When tp_printk (above) is set, it can cause a lot of␣
↪noise
                        on the console. It may be useful to only include the
                        printing of events during boot up, as user space may
                        make the system inoperable.

                        This command line option will stop the printing of␣
↪events
                        to console at the late_initcall_sync() time frame.

        trace_buf_size=nn[KMG]
                        [FTRACE] will set tracing buffer size on each cpu.

        trace_clock=    [FTRACE] Set the clock used for tracing events
                        at boot up.
                        local - Use the per CPU time stamp counter
                                (converted into nanoseconds). Fast, but
                                depending on the architecture, may not be
                                in sync between CPUs.
                        global - Event time stamps are synchronize across
                                CPUs. May be slower than the local clock,
                                but better for some race conditions.
                        counter - Simple counting of events (1, 2, ..)
                                note, some counts may be skipped due to the
                                infrastructure grabbing the clock more than
                                once per event.
                        uptime - Use jiffies as the time stamp.
                        perf - Use the same clock that perf uses.
                        mono - Use ktime_get_mono_fast_ns() for time stamps.
                        mono_raw - Use ktime_get_raw_fast_ns() for time
                                stamps.
                        boot - Use ktime_get_boot_fast_ns() for time stamps.
                        Architectures may add more clocks. See
                        Documentation/trace/ftrace.rst for more details.

        trace_event=[event-list]
                        [FTRACE] Set and start specified trace events in order
                        to facilitate early boot debugging. The event-list is a
                        comma-separated list of trace events to enable. See
                        also Documentation/trace/events.rst

        trace_instance=[instance-info]
                        [FTRACE] Create a ring buffer instance early in boot␣
↪up.
                        This will be listed in:

                                /sys/kernel/tracing/instances

                        Events can be enabled at the time the instance is␣
↪created
```

```
                        via:

                                trace_instance=<name>,<system1>:<event1>,
↪<system2>:<event2>

                        Note, the "<system*>:" portion is optional if the␣
↪event is
                        unique.

                                trace_instance=foo,sched:sched_switch,irq_
↪handler_entry,initcall

                        will enable the "sched_switch" event (note, the␣
↪"sched:" is optional, and
                        the same thing would happen if it was left off). The␣
↪irq_handler_entry
                        event, and all events under the "initcall" system.

        trace_options=[option-list]
                        [FTRACE] Enable or disable tracer options at boot.
                        The option-list is a comma delimited list of options
                        that can be enabled or disabled just as if you were
                        to echo the option name into

                            /sys/kernel/tracing/trace_options

                        For example, to enable stacktrace option (to dump the
                        stack trace of each event), add to the command line:

                              trace_options=stacktrace

                        See also Documentation/trace/ftrace.rst "trace options"
                        section.

        trace_trigger=[trigger-list]
                        [FTRACE] Add a event trigger on specific events.
                        Set a trigger on top of a specific event, with an␣
↪optional
                        filter.

                        The format is is "trace_trigger=<event>.<trigger>[ if
↪<filter>],..."
                        Where more than one trigger may be specified that are␣
↪comma deliminated.

                        For example:

                          trace_trigger="sched_switch.stacktrace if prev_state␣
↪== 2"

                        The above will enable the "stacktrace" trigger on the␣
```

```
↪"sched_switch"
                        event but only trigger it if the "prev_state" of the␣
↪"sched_switch"
                        event is "2" (TASK_UNINTERUPTIBLE).

                        See also "Event triggers" in Documentation/trace/
↪events.rst


        traceoff_on_warning
                        [FTRACE] enable this option to disable tracing when a
                        warning is hit. This turns off "tracing_on". Tracing␣
↪can
                        be enabled again by echoing '1' into the "tracing_on"
                        file located in /sys/kernel/tracing/

                        This option is useful, as it disables the trace before
                        the WARNING dump is called, which prevents the trace to
                        be filled with content caused by the warning output.

                        This option can also be set at run time via the sysctl
                        option:  kernel/traceoff_on_warning

        transparent_hugepage=
                        [KNL]
                        Format: [always|madvise|never]
                        Can be used to control the default behavior of the␣
↪system
                        with respect to transparent hugepages.
                        See Documentation/admin-guide/mm/transhuge.rst
                        for more details.

        trusted.source= [KEYS]
                        Format: <string>
                        This parameter identifies the trust source as a backend
                        for trusted keys implementation. Supported trust
                        sources:
                        - "tpm"
                        - "tee"
                        - "caam"
                        If not specified then it defaults to iterating through
                        the trust source list starting with TPM and assigns the
                        first trust source as a backend which is initialized
                        successfully during iteration.

        trusted.rng=    [KEYS]
                        Format: <string>
                        The RNG used to generate key material for trusted keys.
                        Can be one of:
                        - "kernel"
                        - the same value as trusted.source: "tpm" or "tee"
```

```
                        - "default"
                        If not specified, "default" is used. In this case,
                        the RNG's choice is left to each individual trust␣
↪source.

        tsc=            Disable clocksource stability checks for TSC.
                        Format: <string>
                        [x86] reliable: mark tsc clocksource as reliable, this
                        disables clocksource verification at runtime, as well
                        as the stability checks done at bootup. Used to enable
                        high-resolution timer mode on older hardware, and in
                        virtualized environment.
                        [x86] noirqtime: Do not use TSC to do irq accounting.
                        Used to run time disable IRQ_TIME_ACCOUNTING on any
                        platforms where RDTSC is slow and this accounting
                        can add overhead.
                        [x86] unstable: mark the TSC clocksource as unstable,␣
↪this
                        marks the TSC unconditionally unstable at bootup and
                        avoids any further wobbles once the TSC watchdog␣
↪notices.
                        [x86] nowatchdog: disable clocksource watchdog. Used
                        in situations with strict latency requirements (where
                        interruptions from clocksource watchdog are not
                        acceptable).
                        [x86] recalibrate: force recalibration against a HW␣
↪timer
                        (HPET or PM timer) on systems whose TSC frequency was
                        obtained from HW or FW using either an MSR or␣
↪CPUID(0x15).
                        Warn if the difference is more than 500 ppm.
                        [x86] watchdog: Use TSC as the watchdog clocksource␣
↪with
                        which to check other HW timers (HPET or PM timer), but
                        only on systems where TSC has been deemed trustworthy.
                        This will be suppressed by an earlier tsc=nowatchdog␣
↪and
                        can be overridden by a later tsc=nowatchdog.  A console
                        message will flag any such suppression or overriding.

        tsc_early_khz=  [X86] Skip early TSC calibration and use the given
                        value instead. Useful when the early TSC frequency␣
↪discovery
                        procedure is not reliable, such as on overclocked␣
↪systems
                        with CPUID.16h support and partial CPUID.15h support.
                        Format: <unsigned int>

        tsx=            [X86] Control Transactional Synchronization
                        Extensions (TSX) feature in Intel processors that
                        support TSX control.
```

```
                        This parameter controls the TSX feature. The options␣
↪are:

                        on      - Enable TSX on the system. Although there are
                                  mitigations for all known security␣
↪vulnerabilities,
                                  TSX has been known to be an accelerator for
                                  several previous speculation-related CVEs, and
                                  so there may be unknown security risks␣
↪associated
                                  with leaving it enabled.

                        off     - Disable TSX on the system. (Note that this
                                  option takes effect only on newer CPUs which␣
↪are
                                  not vulnerable to MDS, i.e., have
                                  MSR_IA32_ARCH_CAPABILITIES.MDS_NO=1 and which␣
↪get
                                  the new IA32_TSX_CTRL MSR through a microcode
                                  update. This new MSR allows for the reliable
                                  deactivation of the TSX functionality.)

                        auto    - Disable TSX if X86_BUG_TAA is present,
                                  otherwise enable TSX on the system.

                        Not specifying this option is equivalent to tsx=off.

                        See Documentation/admin-guide/hw-vuln/tsx_async_abort.
↪rst
                        for more details.

       tsx_async_abort= [X86,INTEL] Control mitigation for the TSX Async
                        Abort (TAA) vulnerability.

                        Similar to Micro-architectural Data Sampling (MDS)
                        certain CPUs that support Transactional
                        Synchronization Extensions (TSX) are vulnerable to an
                        exploit against CPU internal buffers which can forward
                        information to a disclosure gadget under certain
                        conditions.

                        In vulnerable processors, the speculatively forwarded
                        data can be used in a cache side channel attack, to
                        access data to which the attacker does not have direct
                        access.

                        This parameter controls the TAA mitigation.  The
                        options are:

                        full       - Enable TAA mitigation on vulnerable CPUs
```

```
                                    if TSX is enabled.

                        full,nosmt - Enable TAA mitigation and disable SMT on
                                     vulnerable CPUs. If TSX is disabled, SMT
                                     is not disabled because CPU is not
                                     vulnerable to cross-thread TAA attacks.
                        off        - Unconditionally disable TAA mitigation

                        On MDS-affected machines, tsx_async_abort=off can be
                        prevented by an active MDS mitigation as both␣
↪vulnerabilities
                        are mitigated with the same mechanism so in order to␣
↪disable
                        this mitigation, you need to specify mds=off too.

                        Not specifying this option is equivalent to
                        tsx_async_abort=full.  On CPUs which are MDS affected
                        and deploy MDS mitigation, TAA mitigation is not
                        required and doesn't provide any additional
                        mitigation.

                        For details see:
                        Documentation/admin-guide/hw-vuln/tsx_async_abort.rst

        turbografx.map[2|3]=    [HW,JOY]
                        TurboGraFX parallel port interface
                        Format:
                        <port#>,<js1>,<js2>,<js3>,<js4>,<js5>,<js6>,<js7>
                        See also Documentation/input/devices/joystick-parport.
↪rst

        udbg-immortal   [PPC] When debugging early kernel crashes that
                        happen after console_init() and before a proper
                        console driver takes over, this boot options might
                        help "seeing" what's going on.

        uhash_entries=  [KNL,NET]
                        Set number of hash buckets for UDP/UDP-Lite connections

        uhci-hcd.ignore_oc=
                        [USB] Ignore overcurrent events (default N).
                        Some badly-designed motherboards generate lots of
                        bogus events, for ports that aren't wired to
                        anything.  Set this parameter to avoid log spamming.
                        Note that genuine overcurrent events won't be
                        reported either.

        unknown_nmi_panic
                        [X86] Cause panic on unknown NMI.

        unwind_debug    [X86-64]
```

```
                Enable unwinder debug output.  This can be
                useful for debugging certain unwinder error
                conditions, including corrupt stacks and
                bad/missing unwinder metadata.

usbcore.authorized_default=
                [USB] Default USB device authorization:
                (default -1 = authorized (same as 1),
                0 = not authorized, 1 = authorized, 2 = authorized
                if device connected to internal port)

usbcore.autosuspend=
                [USB] The autosuspend time delay (in seconds) used
                for newly-detected USB devices (default 2).  This
                is the time required before an idle device will be
                autosuspended.  Devices for which the delay is set
                to a negative value won't be autosuspended at all.

usbcore.usbfs_snoop=
                [USB] Set to log all usbfs traffic (default 0 = off).

usbcore.usbfs_snoop_max=
                [USB] Maximum number of bytes to snoop in each URB
                (default = 65536).

usbcore.blinkenlights=
                [USB] Set to cycle leds on hubs (default 0 = off).

usbcore.old_scheme_first=
                [USB] Start with the old device initialization
                scheme (default 0 = off).

usbcore.usbfs_memory_mb=
                [USB] Memory limit (in MB) for buffers allocated by
                usbfs (default = 16, 0 = max = 2047).

usbcore.use_both_schemes=
                [USB] Try the other device initialization scheme
                if the first one fails (default 1 = enabled).

usbcore.initial_descriptor_timeout=
                [USB] Specifies timeout for the initial 64-byte
                USB_REQ_GET_DESCRIPTOR request in milliseconds
                (default 5000 = 5.0 seconds).

usbcore.nousb   [USB] Disable the USB subsystem

usbcore.quirks=
                [USB] A list of quirk entries to augment the built-in
                usb core quirk list. List entries are separated by
                commas. Each entry has the form
```

```
                        VendorID:ProductID:Flags. The IDs are 4-digit hex
                        numbers and Flags is a set of letters. Each letter
                        will change the built-in quirk; setting it if it is
                        clear and clearing it if it is set. The letters have
                        the following meanings:
                                a = USB_QUIRK_STRING_FETCH_255 (string
                                        descriptors must not be fetched using
                                        a 255-byte read);
                                b = USB_QUIRK_RESET_RESUME (device can't resume
                                        correctly so reset it instead);
                                c = USB_QUIRK_NO_SET_INTF (device can't handle
                                        Set-Interface requests);
                                d = USB_QUIRK_CONFIG_INTF_STRINGS (device can't
                                        handle its Configuration or Interface
                                        strings);
                                e = USB_QUIRK_RESET (device can't be reset
                                        (e.g morph devices), don't use reset);
                                f = USB_QUIRK_HONOR_BNUMINTERFACES (device has
                                        more interface descriptions than the
                                        bNumInterfaces count, and can't handle
                                        talking to these interfaces);
                                g = USB_QUIRK_DELAY_INIT (device needs a pause
                                        during initialization, after we read
                                        the device descriptor);
                                h = USB_QUIRK_LINEAR_UFRAME_INTR_BINTERVAL (For
                                        high speed and super speed interrupt
                                        endpoints, the USB 2.0 and USB 3.0 spec
                                        require the interval in microframes (1
                                        microframe = 125 microseconds) to be
                                        calculated as interval = 2 ^
                                        (bInterval-1).
                                        Devices with this quirk report their
                                        bInterval as the result of this
                                        calculation instead of the exponent
                                        variable used in the calculation);
                                i = USB_QUIRK_DEVICE_QUALIFIER (device can't
                                        handle device_qualifier descriptor
                                        requests);
                                j = USB_QUIRK_IGNORE_REMOTE_WAKEUP (device
                                        generates spurious wakeup, ignore
                                        remote wakeup capability);
                                k = USB_QUIRK_NO_LPM (device can't handle Link
                                        Power Management);
                                l = USB_QUIRK_LINEAR_FRAME_INTR_BINTERVAL
                                        (Device reports its bInterval as linear
                                        frames instead of the USB 2.0
                                        calculation);
                                m = USB_QUIRK_DISCONNECT_SUSPEND (Device needs
                                        to be disconnected before suspend to
                                        prevent spurious wakeup);
                                n = USB_QUIRK_DELAY_CTRL_MSG (Device needs a
```

```
                                        pause after every control message);
                                o = USB_QUIRK_HUB_SLOW_RESET (Hub needs extra
                                        delay after resetting its port);
                                p = USB_QUIRK_SHORT_SET_ADDRESS_REQ_TIMEOUT
                                        (Reduce timeout of the SET_ADDRESS
                                        request from 5000 ms to 500 ms);
                        Example: quirks=0781:5580:bk,0a5c:5834:gij

        usbhid.mousepoll=
                        [USBHID] The interval which mice are to be polled at.

        usbhid.jspoll=
                        [USBHID] The interval which joysticks are to be polled␣
↪at.

        usbhid.kbpoll=
                        [USBHID] The interval which keyboards are to be polled␣
↪at.

        usb-storage.delay_use=
                        [UMS] The delay in seconds before a new device is
                        scanned for Logical Units (default 1).

        usb-storage.quirks=
                        [UMS] A list of quirks entries to supplement or
                        override the built-in unusual_devs list.  List
                        entries are separated by commas.  Each entry has
                        the form VID:PID:Flags where VID and PID are Vendor
                        and Product ID values (4-digit hex numbers) and
                        Flags is a set of characters, each corresponding
                        to a common usb-storage quirk flag as follows:
                                a = SANE_SENSE (collect more than 18 bytes
                                        of sense data, not on uas);
                                b = BAD_SENSE (don't collect more than 18
                                        bytes of sense data, not on uas);
                                c = FIX_CAPACITY (decrease the reported
                                        device capacity by one sector);
                                d = NO_READ_DISC_INFO (don't use
                                        READ_DISC_INFO command, not on uas);
                                e = NO_READ_CAPACITY_16 (don't use
                                        READ_CAPACITY_16 command);
                                f = NO_REPORT_OPCODES (don't use report opcodes
                                        command, uas only);
                                g = MAX_SECTORS_240 (don't transfer more than
                                        240 sectors at a time, uas only);
                                h = CAPACITY_HEURISTICS (decrease the
                                        reported device capacity by one
                                        sector if the number is odd);
                                i = IGNORE_DEVICE (don't bind to this
                                        device);
                                j = NO_REPORT_LUNS (don't use report luns
```

```
                                        command, uas only);
                                k = NO_SAME (do not use WRITE_SAME, uas only)
                                l = NOT_LOCKABLE (don't try to lock and
                                        unlock ejectable media, not on uas);
                                m = MAX_SECTORS_64 (don't transfer more
                                        than 64 sectors = 32 KB at a time,
                                        not on uas);
                                n = INITIAL_READ10 (force a retry of the
                                        initial READ(10) command, not on uas);
                                o = CAPACITY_OK (accept the capacity
                                        reported by the device, not on uas);
                                p = WRITE_CACHE (the device cache is ON
                                        by default, not on uas);
                                r = IGNORE_RESIDUE (the device reports
                                        bogus residue values, not on uas);
                                s = SINGLE_LUN (the device has only one
                                        Logical Unit);
                                t = NO_ATA_1X (don't allow ATA(12) and ATA(16)
                                        commands, uas only);
                                u = IGNORE_UAS (don't bind to the uas driver);
                                w = NO_WP_DETECT (don't test whether the
                                        medium is write-protected).
                                y = ALWAYS_SYNC (issue a SYNCHRONIZE_CACHE
                                        even if the device claims no cache,
                                        not on uas)
                        Example: quirks=0419:aaf5:rl,0421:0433:rc

        user_debug=     [KNL,ARM]
                        Format: <int>
                        See arch/arm/Kconfig.debug help text.
                                 1 - undefined instruction events
                                 2 - system calls
                                 4 - invalid data aborts
                                 8 - SIGSEGV faults
                                16 - SIGBUS faults
                        Example: user_debug=31

        userpte=
                        [X86] Flags controlling user PTE allocations.

                                nohigh = do not allocate PTE pages in
                                        HIGHMEM regardless of setting
                                        of CONFIG_HIGHPTE.

        vdso=           [X86,SH,SPARC]
                        On X86_32, this is an alias for vdso32=.  Otherwise:

                        vdso=1: enable VDSO (the default)
                        vdso=0: disable VDSO mapping

        vdso32=         [X86] Control the 32-bit vDSO
```

```
                        vdso32=1: enable 32-bit VDSO
                        vdso32=0 or vdso32=2: disable 32-bit VDSO

                        See the help text for CONFIG_COMPAT_VDSO for more
                        details.  If CONFIG_COMPAT_VDSO is set, the default is
                        vdso32=0; otherwise, the default is vdso32=1.

                        For compatibility with older kernels, vdso32=2 is an
                        alias for vdso32=0.

                        Try vdso32=0 if you encounter an error that says:
                        dl_main: Assertion `(void *) ph->p_vaddr == _rtld_
↪local._dl_sysinfo_dso' failed!

       vector=          [IA-64,SMP]
                        vector=percpu: enable percpu vector domain

       video=           [FB] Frame buffer configuration
                        See Documentation/fb/modedb.rst.

       video.brightness_switch_enabled= [ACPI]
                        Format: [0|1]
                        If set to 1, on receiving an ACPI notify event
                        generated by hotkey, video driver will adjust␣
↪brightness
                        level and then send out the event to user space through
                        the allocated input device. If set to 0, video driver
                        will only send out the event without touching backlight
                        brightness level.
                        default: 1

       virtio_mmio.device=
                        [VMMIO] Memory mapped virtio (platform) device.

                                <size>@<baseaddr>:<irq>[:<id>]
                        where:
                                <size>     := size (can use standard suffixes
                                                like K, M and G)
                                <baseaddr> := physical base address
                                <irq>      := interrupt number (as passed to
                                                request_irq())
                                <id>       := (optional) platform device id
                        example:
                                virtio_mmio.device=1K@0x100b0000:48:7

                        Can be used multiple times for multiple devices.

       vga=             [BOOT,X86-32] Select a particular video mode
                        See Documentation/arch/x86/boot.rst and
                        Documentation/admin-guide/svga.rst.
                        Use vga=ask for menu.
```

```
                This is actually a boot loader parameter; the value is
                passed to the kernel using a special protocol.

vm_debug[=options]      [KNL] Available with CONFIG_DEBUG_VM=y.
                May slow down system boot speed, especially when
                enabled on systems with a large amount of memory.
                All options are enabled by default, and this
                interface is meant to allow for selectively
                enabling or disabling specific virtual memory
                debugging features.

                Available options are:
                  P     Enable page structure init time poisoning
                  -     Disable all of the above options

vmalloc=nn[KMG] [KNL,BOOT] Forces the vmalloc area to have an exact
                size of <nn>. This can be used to increase the
                minimum size (128MB on x86). It can also be used to
                decrease the size and leave more room for directly
                mapped kernel RAM.

vmcp_cma=nn[MG] [KNL,S390]
                Sets the memory size reserved for contiguous memory
                allocations for the vmcp device driver.

vmhalt=         [KNL,S390] Perform z/VM CP command after system halt.
                Format: <command>

vmpanic=        [KNL,S390] Perform z/VM CP command after kernel panic.
                Format: <command>

vmpoff=         [KNL,S390] Perform z/VM CP command after power off.
                Format: <command>

vsyscall=       [X86-64]
                Controls the behavior of vsyscalls (i.e. calls to
                fixed addresses of 0xffffffffff600x00 from legacy
                code).  Most statically-linked binaries and older
                versions of glibc use these calls.  Because these
                functions are at fixed addresses, they make nice
                targets for exploits that can control RIP.

                emulate     Vsyscalls turn into traps and are emulated
                            reasonably safely.  The vsyscall page is
                            readable.

                xonly       [default] Vsyscalls turn into traps and are
                            emulated reasonably safely.  The vsyscall
                            page is not readable.

                none        Vsyscalls don't work at all.  This makes
```

```
                                them quite hard to use for exploits but
                                might break your system.

        vt.color=       [VT] Default text color.
                        Format: 0xYX, X = foreground, Y = background.
                        Default: 0x07 = light gray on black.

        vt.cur_default= [VT] Default cursor shape.
                        Format: 0xCCBBAA, where AA, BB, and CC are the same as
                        the parameters of the <Esc>[?A;B;Cc escape sequence;
                        see VGA-softcursor.txt. Default: 2 = underline.

        vt.default_blu= [VT]
                        Format: <blue0>,<blue1>,<blue2>,...,<blue15>
                        Change the default blue palette of the console.
                        This is a 16-member array composed of values
                        ranging from 0-255.

        vt.default_grn= [VT]
                        Format: <green0>,<green1>,<green2>,...,<green15>
                        Change the default green palette of the console.
                        This is a 16-member array composed of values
                        ranging from 0-255.

        vt.default_red= [VT]
                        Format: <red0>,<red1>,<red2>,...,<red15>
                        Change the default red palette of the console.
                        This is a 16-member array composed of values
                        ranging from 0-255.

        vt.default_utf8=
                        [VT]
                        Format=<0|1>
                        Set system-wide default UTF-8 mode for all tty's.
                        Default is 1, i.e. UTF-8 mode is enabled for all
                        newly opened terminals.

        vt.global_cursor_default=
                        [VT]
                        Format=<-1|0|1>
                        Set system-wide default for whether a cursor
                        is shown on new VTs. Default is -1,
                        i.e. cursors will be created by default unless
                        overridden by individual drivers. 0 will hide
                        cursors, 1 will display them.

        vt.italic=      [VT] Default color for italic text; 0-15.
                        Default: 2 = green.

        vt.underline=   [VT] Default color for underlined text; 0-15.
                        Default: 3 = cyan.
```

```
        watchdog timers [HW,WDT] For information on watchdog timers,
                        see Documentation/watchdog/watchdog-parameters.rst
                        or other driver-specific files in the
                        Documentation/watchdog/ directory.

        watchdog_thresh=
                        [KNL]
                        Set the hard lockup detector stall duration
                        threshold in seconds. The soft lockup detector
                        threshold is set to twice the value. A value of 0
                        disables both lockup detectors. Default is 10
                        seconds.

        workqueue.unbound_cpus=
                        [KNL,SMP] Specify to constrain one or some CPUs
                        to use in unbound workqueues.
                        Format: <cpu-list>
                        By default, all online CPUs are available for
                        unbound workqueues.

        workqueue.watchdog_thresh=
                        If CONFIG_WQ_WATCHDOG is configured, workqueue can
                        warn stall conditions and dump internal state to
                        help debugging.  0 disables workqueue stall
                        detection; otherwise, it's the stall threshold
                        duration in seconds.  The default value is 30 and
                        it can be updated at runtime by writing to the
                        corresponding sysfs file.

        workqueue.cpu_intensive_thresh_us=
                        Per-cpu work items which run for longer than this
                        threshold are automatically considered CPU intensive
                        and excluded from concurrency management to prevent
                        them from noticeably delaying other per-cpu work
                        items. Default is 10000 (10ms).

                        If CONFIG_WQ_CPU_INTENSIVE_REPORT is set, the kernel
                        will report the work functions which violate this
                        threshold repeatedly. They are likely good
                        candidates for using WQ_UNBOUND workqueues instead.

        workqueue.power_efficient
                        Per-cpu workqueues are generally preferred because
                        they show better performance thanks to cache
                        locality; unfortunately, per-cpu workqueues tend to
                        be more power hungry than unbound workqueues.

                        Enabling this makes the per-cpu workqueues which
                        were observed to contribute significantly to power
                        consumption unbound, leading to measurably lower
```

```
                power usage at the cost of small performance
                overhead.

                The default value of this parameter is determined by
                the config option CONFIG_WQ_POWER_EFFICIENT_DEFAULT.

        workqueue.default_affinity_scope=
                Select the default affinity scope to use for unbound
                workqueues. Can be one of "cpu", "smt", "cache",
                "numa" and "system". Default is "cache". For more
                information, see the Affinity Scopes section in
                Documentation/core-api/workqueue.rst.

                This can be changed after boot by writing to the
                matching /sys/module/workqueue/parameters file. All
                workqueues with the "default" affinity scope will be
                updated accordignly.

        workqueue.debug_force_rr_cpu
                Workqueue used to implicitly guarantee that work
                items queued without explicit CPU specified are put
                on the local CPU.  This guarantee is no longer true
                and while local CPU is still preferred work items
                may be put on foreign CPUs.  This debug option
                forces round-robin CPU selection to flush out
                usages which depend on the now broken guarantee.
                When enabled, memory and cache locality will be
                impacted.

        writecombine=   [LOONGARCH] Control the MAT (Memory Access Type) of
                ioremap_wc().

                on   - Enable writecombine, use WUC for ioremap_wc()
                off  - Disable writecombine, use SUC for ioremap_wc()

        x2apic_phys     [X86-64,APIC] Use x2apic physical mode instead of
                default x2apic cluster mode on platforms
                supporting x2apic.

        xen_512gb_limit         [KNL,X86-64,XEN]
                Restricts the kernel running paravirtualized under Xen
                to use only up to 512 GB of RAM. The reason to do so is
                crash analysis tools and Xen tools for doing domain
                save/restore/migration must be enabled to handle larger
                domains.

        xen_emul_unplug=                [HW,X86,XEN]
                Unplug Xen emulated devices
                Format: [unplug0,][unplug1]
                ide-disks -- unplug primary master IDE devices
                aux-ide-disks -- unplug non-primary-master IDE devices
```

```
                        nics -- unplug network devices
                        all -- unplug all emulated devices (NICs and IDE disks)
                        unnecessary -- unplugging emulated devices is
                                unnecessary even if the host did not respond to
                                the unplug protocol
                        never -- do not unplug even if version check succeeds

        xen_legacy_crash        [X86,XEN]
                        Crash from Xen panic notifier, without executing late
                        panic() code such as dumping handler.

        xen_msr_safe=   [X86,XEN]
                        Format: <bool>
                        Select whether to always use non-faulting (safe) MSR
                        access functions when running as Xen PV guest. The
                        default value is controlled by CONFIG_XEN_PV_MSR_SAFE.

        xen_nopvspin    [X86,XEN]
                        Disables the qspinlock slowpath using Xen PV␣
↪optimizations.
                        This parameter is obsoleted by "nopvspin" parameter,␣
↪which
                        has equivalent effect for XEN platform.

        xen_nopv        [X86]
                        Disables the PV optimizations forcing the HVM guest to
                        run as generic HVM guest with no PV drivers.
                        This option is obsoleted by the "nopv" option, which
                        has equivalent effect for XEN platform.

        xen_no_vector_callback
                        [KNL,X86,XEN] Disable the vector callback for Xen
                        event channel interrupts.

        xen_scrub_pages=        [XEN]
                        Boolean option to control scrubbing pages before␣
↪giving them back
                        to Xen, for use by other domains. Can be also changed␣
↪at runtime
                        with /sys/devices/system/xen_memory/xen_memory0/scrub_
↪pages.
                        Default value controlled with CONFIG_XEN_SCRUB_PAGES_
↪DEFAULT.

        xen_timer_slop= [X86-64,XEN]
                        Set the timer slop (in nanoseconds) for the virtual Xen
                        timers (default is 100000). This adjusts the minimum
                        delta of virtualized Xen timers, where lower values
                        improve timer resolution at the expense of processing
                        more timer interrupts.
```

```
        xen.balloon_boot_timeout= [XEN]
                        The time (in seconds) to wait before giving up to boot
                        in case initial ballooning fails to free enough memory.
                        Applies only when running as HVM or PVH guest and
                        started with less memory configured than allowed at
                        max. Default is 180.

        xen.event_eoi_delay=    [XEN]
                        How long to delay EOI handling in case of event
                        storms (jiffies). Default is 10.

        xen.event_loop_timeout= [XEN]
                        After which time (jiffies) the event handling loop
                        should start to delay EOI handling. Default is 2.

        xen.fifo_events=        [XEN]
                        Boolean parameter to disable using fifo event handling
                        even if available. Normally fifo event handling is
                        preferred over the 2-level event handling, as it is
                        fairer and the number of possible event channels is
                        much higher. Default is on (use fifo events).

        xirc2ps_cs=     [NET,PCMCIA]
                        Format:
                        <irq>,<irq_mask>,<io>,<full_duplex>,<do_sound>,<lockup_
↪hack>[,<irq2>[,<irq3>[,<irq4>]]]

        xive=           [PPC]
                        By default on POWER9 and above, the kernel will
                        natively use the XIVE interrupt controller. This option
                        allows the fallback firmware mode to be used:

                        off       Fallback to firmware control of XIVE␣
↪interrupt
                                  controller on both pseries and powernv
                                  platforms. Only useful on POWER9 and above.

        xive.store-eoi=off      [PPC]
                        By default on POWER10 and above, the kernel will use
                        stores for EOI handling when the XIVE interrupt mode
                        is active. This option allows the XIVE driver to use
                        loads instead, as on POWER9.

        xhci-hcd.quirks         [USB,KNL]
                        A hex value specifying bitmask with supplemental xhci
                        host controller quirks. Meaning of each bit can be
                        consulted in header drivers/usb/host/xhci.h.

        xmon            [PPC]
                        Format: { early | on | rw | ro | off }
                        Controls if xmon debugger is enabled. Default is off.
```

```
                        Passing only "xmon" is equivalent to "xmon=early".
                        early   Call xmon as early as possible on boot; xmon
                                debugger is called from setup_arch().
                        on      xmon debugger hooks will be installed so xmon
                                is only called on a kernel crash. Default mode,
                                i.e. either "ro" or "rw" mode, is controlled
                                with CONFIG_XMON_DEFAULT_RO_MODE.
                        rw      xmon debugger hooks will be installed so xmon
                                is called only on a kernel crash, mode is␣
↪write,
                                meaning SPR registers, memory and, other data
                                can be written using xmon commands.
                        ro      same as "rw" option above but SPR registers,
                                memory, and other data can't be written using
                                xmon commands.
                        off     xmon is disabled.
```

CHAPTER

THREE

LINUX ALLOCATED DEVICES (4.X+ VERSION)

This list is the Linux Device List, the official registry of allocated device numbers and `/dev` directory nodes for the Linux operating system.

The version of this document at lanana.org is no longer maintained. This version in the mainline Linux kernel is the master document. Updates shall be sent as patches to the kernel maintainers (see the Documentation/process/submitting-patches.rst document). Specifically explore the sections titled "CHAR and MISC DRIVERS", and "BLOCK LAYER" in the MAINTAINERS file to find the right maintainers to involve for character and block devices.

This document is included by reference into the Filesystem Hierarchy Standard (FHS). The FHS is available from https://www.pathname.com/fhs/.

Allocations marked (68k/Amiga) apply to Linux/68k on the Amiga platform only. Allocations marked (68k/Atari) apply to Linux/68k on the Atari platform only.

This document is in the public domain. The authors requests, however, that semantically altered versions are not distributed without permission of the authors, assuming the authors can be contacted without an unreasonable effort.

Attention: DEVICE DRIVERS AUTHORS PLEASE READ THIS

Linux now has extensive support for dynamic allocation of device numbering and can use `sysfs` and `udev` (`systemd`) to handle the naming needs. There are still some exceptions in the serial and boot device area. Before asking for a device number make sure you actually need one.

To have a major number allocated, or a minor number in situations where that applies (e.g. busmice), please submit a patch and send to the authors as indicated above.

Keep the description of the device *in the same format as this list*. The reason for this is that it is the only way we have found to ensure we have all the requisite information to publish your device and avoid conflicts.

Finally, sometimes we have to play "namespace police." Please don't be offended. We often get submissions for `/dev` names that would be bound to cause conflicts down the road. We are trying to avoid getting in a situation where we would have to suffer an incompatible forward change. Therefore, please consult with us **before** you make your device names and numbers in any way public, at least to the point where it would be at all difficult to get them changed.

Your cooperation is appreciated.

```
0             Unnamed devices (e.g. non-device mounts)
```

```
                  0 = reserved as null device number
                See block major 144, 145, 146 for expansion areas.

  1 char        Memory devices
                  1 = /dev/mem          Physical memory access
                  2 = /dev/kmem         OBSOLETE - replaced by /proc/kcore
                  3 = /dev/null         Null device
                  4 = /dev/port         I/O port access
                  5 = /dev/zero         Null byte source
                  6 = /dev/core         OBSOLETE - replaced by /proc/kcore
                  7 = /dev/full         Returns ENOSPC on write
                  8 = /dev/random       Nondeterministic random number gen.
                  9 = /dev/urandom      Faster, less secure random number gen.
                 10 = /dev/aio          Asynchronous I/O notification interface
                 11 = /dev/kmsg         Writes to this come out as printk's,␣
↪reads
                                        export the buffered printk records.
                 12 = /dev/oldmem       OBSOLETE - replaced by /proc/vmcore

  1 block       RAM disk
                  0 = /dev/ram0         First RAM disk
                  1 = /dev/ram1         Second RAM disk
                    ...
                250 = /dev/initrd       Initial RAM disk

                Older kernels had /dev/ramdisk (1, 1) here.
                /dev/initrd refers to a RAM disk which was preloaded
                by the boot loader; newer kernels use /dev/ram0 for
                the initrd.

  2 char        Pseudo-TTY masters
                  0 = /dev/ptyp0        First PTY master
                  1 = /dev/ptyp1        Second PTY master
                    ...
                255 = /dev/ptyef        256th PTY master

                Pseudo-tty's are named as follows:
                * Masters are "pty", slaves are "tty";
                * the fourth letter is one of pqrstuvwxyzabcde indicating
                  the 1st through 16th series of 16 pseudo-ttys each, and
                * the fifth letter is one of 0123456789abcdef indicating
                  the position within the series.

                These are the old-style (BSD) PTY devices; Unix98
                devices are on major 128 and above and use the PTY
                master multiplex (/dev/ptmx) to acquire a PTY on
                demand.

  2 block       Floppy disks
                  0 = /dev/fd0          Controller 0, drive 0, autodetect
                  1 = /dev/fd1          Controller 0, drive 1, autodetect
```

```
  2 = /dev/fd2          Controller 0, drive 2, autodetect
  3 = /dev/fd3          Controller 0, drive 3, autodetect
128 = /dev/fd4          Controller 1, drive 0, autodetect
129 = /dev/fd5          Controller 1, drive 1, autodetect
130 = /dev/fd6          Controller 1, drive 2, autodetect
131 = /dev/fd7          Controller 1, drive 3, autodetect

To specify format, add to the autodetect device number:
  0 = /dev/fd?          Autodetect format
  4 = /dev/fd?d360      5.25"  360K in a 360K  drive(1)
 20 = /dev/fd?h360      5.25"  360K in a 1200K drive(1)
 48 = /dev/fd?h410      5.25"  410K in a 1200K drive
 64 = /dev/fd?h420      5.25"  420K in a 1200K drive
 24 = /dev/fd?h720      5.25"  720K in a 1200K drive
 80 = /dev/fd?h880      5.25"  880K in a 1200K drive(1)
  8 = /dev/fd?h1200     5.25" 1200K in a 1200K drive(1)
 40 = /dev/fd?h1440     5.25" 1440K in a 1200K drive(1)
 56 = /dev/fd?h1476     5.25" 1476K in a 1200K drive
 72 = /dev/fd?h1494     5.25" 1494K in a 1200K drive
 92 = /dev/fd?h1600     5.25" 1600K in a 1200K drive(1)

 12 = /dev/fd?u360      3.5"   360K Double Density(2)
 16 = /dev/fd?u720      3.5"   720K Double Density(1)
120 = /dev/fd?u800      3.5"   800K Double Density(2)
 52 = /dev/fd?u820      3.5"   820K Double Density
 68 = /dev/fd?u830      3.5"   830K Double Density
 84 = /dev/fd?u1040     3.5"  1040K Double Density(1)
 88 = /dev/fd?u1120     3.5"  1120K Double Density(1)
 28 = /dev/fd?u1440     3.5"  1440K High Density(1)
124 = /dev/fd?u1600     3.5"  1600K High Density(1)
 44 = /dev/fd?u1680     3.5"  1680K High Density(3)
 60 = /dev/fd?u1722     3.5"  1722K High Density
 76 = /dev/fd?u1743     3.5"  1743K High Density
 96 = /dev/fd?u1760     3.5"  1760K High Density
116 = /dev/fd?u1840     3.5"  1840K High Density(3)
100 = /dev/fd?u1920     3.5"  1920K High Density(1)
 32 = /dev/fd?u2880     3.5"  2880K Extra Density(1)
104 = /dev/fd?u3200     3.5"  3200K Extra Density
108 = /dev/fd?u3520     3.5"  3520K Extra Density
112 = /dev/fd?u3840     3.5"  3840K Extra Density(1)

 36 = /dev/fd?CompaQ    Compaq 2880K drive; obsolete?

(1) Autodetectable format
(2) Autodetectable format in a Double Density (720K) drive only
(3) Autodetectable format in a High Density (1440K) drive only

NOTE: The letter in the device name (d, q, h or u)
signifies the type of drive: 5.25" Double Density (d),
5.25" Quad Density (q), 5.25" High Density (h) or 3.5"
(any model, u).  The use of the capital letters D, H
```

```
                and E for the 3.5" models have been deprecated, since
                the drive type is insignificant for these devices.

  3 char        Pseudo-TTY slaves
                  0 = /dev/ttyp0        First PTY slave
                  1 = /dev/ttyp1        Second PTY slave
                    ...
                255 = /dev/ttyef        256th PTY slave

                These are the old-style (BSD) PTY devices; Unix98
                devices are on major 136 and above.

  3 block       First MFM, RLL and IDE hard disk/CD-ROM interface
                  0 = /dev/hda          Master: whole disk (or CD-ROM)
                 64 = /dev/hdb          Slave: whole disk (or CD-ROM)

                For partitions, add to the whole disk device number:
                  0 = /dev/hd?          Whole disk
                  1 = /dev/hd?1         First partition
                  2 = /dev/hd?2         Second partition
                    ...
                 63 = /dev/hd?63        63rd partition

                For Linux/i386, partitions 1-4 are the primary
                partitions, and 5 and above are logical partitions.
                Other versions of Linux use partitioning schemes
                appropriate to their respective architectures.

  4 char        TTY devices
                  0 = /dev/tty0         Current virtual console

                  1 = /dev/tty1         First virtual console
                    ...
                 63 = /dev/tty63        63rd virtual console
                 64 = /dev/ttyS0        First UART serial port
                    ...
                255 = /dev/ttyS191      192nd UART serial port

                UART serial ports refer to 8250/16450/16550 series devices.

                Older versions of the Linux kernel used this major
                number for BSD PTY devices.  As of Linux 2.1.115, this
                is no longer supported.  Use major numbers 2 and 3.

  4 block       Aliases for dynamically allocated major devices to be used
                when its not possible to create the real device nodes
                because the root filesystem is mounted read-only.

                   0 = /dev/root

  5 char        Alternate TTY devices
```

```
                  0 = /dev/tty          Current TTY device
                  1 = /dev/console      System console
                  2 = /dev/ptmx         PTY master multiplex
                  3 = /dev/ttyprintk    User messages via printk TTY device
                 64 = /dev/cua0         Callout device for ttyS0
                    ...
                255 = /dev/cua191       Callout device for ttyS191

                (5,1) is /dev/console starting with Linux 2.1.71.  See
                the section on terminal devices for more information
                on /dev/console.

  6 char        Parallel printer devices
                  0 = /dev/lp0          Parallel printer on parport0
                  1 = /dev/lp1          Parallel printer on parport1
                    ...

                Current Linux kernels no longer have a fixed mapping
                between parallel ports and I/O addresses.  Instead,
                they are redirected through the parport multiplex layer.

  7 char        Virtual console capture devices
                  0 = /dev/vcs          Current vc text (glyph) contents
                  1 = /dev/vcs1         tty1 text (glyph) contents
                    ...
                 63 = /dev/vcs63        tty63 text (glyph) contents
                 64 = /dev/vcsu         Current vc text (unicode) contents
                65 = /dev/vcsu1         tty1 text (unicode) contents
                    ...
                127 = /dev/vcsu63       tty63 text (unicode) contents
                128 = /dev/vcsa         Current vc text/attribute (glyph)␣
↪contents
                129 = /dev/vcsa1        tty1 text/attribute (glyph) contents
                    ...
                191 = /dev/vcsa63       tty63 text/attribute (glyph) contents

                NOTE: These devices permit both read and write access.

  7 block       Loopback devices
                  0 = /dev/loop0        First loop device
                  1 = /dev/loop1        Second loop device
                    ...

                The loop devices are used to mount filesystems not
                associated with block devices.  The binding to the
                loop devices is handled by mount(8) or losetup(8).

  8 block       SCSI disk devices (0-15)
                  0 = /dev/sda          First SCSI disk whole disk
                 16 = /dev/sdb          Second SCSI disk whole disk
                 32 = /dev/sdc          Third SCSI disk whole disk
```

```
                ...
              240 = /dev/sdp          Sixteenth SCSI disk whole disk

              Partitions are handled in the same way as for IDE
              disks (see major number 3) except that the limit on
              partitions is 15.

  9 char      SCSI tape devices
                0 = /dev/st0          First SCSI tape, mode 0
                1 = /dev/st1          Second SCSI tape, mode 0
                ...
               32 = /dev/st0l         First SCSI tape, mode 1
               33 = /dev/st1l         Second SCSI tape, mode 1
                ...
               64 = /dev/st0m         First SCSI tape, mode 2
               65 = /dev/st1m         Second SCSI tape, mode 2
                ...
               96 = /dev/st0a         First SCSI tape, mode 3
               97 = /dev/st1a         Second SCSI tape, mode 3
                  ...
              128 = /dev/nst0         First SCSI tape, mode 0, no rewind
              129 = /dev/nst1         Second SCSI tape, mode 0, no rewind
                ...
              160 = /dev/nst0l        First SCSI tape, mode 1, no rewind
              161 = /dev/nst1l        Second SCSI tape, mode 1, no rewind
                ...
              192 = /dev/nst0m        First SCSI tape, mode 2, no rewind
              193 = /dev/nst1m        Second SCSI tape, mode 2, no rewind
                ...
              224 = /dev/nst0a        First SCSI tape, mode 3, no rewind
              225 = /dev/nst1a        Second SCSI tape, mode 3, no rewind
                ...

              "No rewind" refers to the omission of the default
              automatic rewind on device close.  The MTREW or MTOFFL
              ioctl()'s can be used to rewind the tape regardless of
              the device used to access it.

  9 block     Metadisk (RAID) devices
                0 = /dev/md0          First metadisk group
                1 = /dev/md1          Second metadisk group
                ...

              The metadisk driver is used to span a
              filesystem across multiple physical disks.

 10 char      Non-serial mice, misc features
                0 = /dev/logibm       Logitech bus mouse
                1 = /dev/psaux        PS/2-style mouse port
                2 = /dev/inportbm     Microsoft Inport bus mouse
                3 = /dev/atibm        ATI XL bus mouse
```

```
  4 = /dev/jbm          J-mouse
  4 = /dev/amigamouse   Amiga mouse (68k/Amiga)
  5 = /dev/atarimouse   Atari mouse
  6 = /dev/sunmouse     Sun mouse
  7 = /dev/amigamouse1  Second Amiga mouse
  8 = /dev/smouse       Simple serial mouse driver
  9 = /dev/pc110pad     IBM PC-110 digitizer pad
 10 = /dev/adbmouse     Apple Desktop Bus mouse
 11 = /dev/vrtpanel     Vr41xx embedded touch panel
 13 = /dev/vpcmouse     Connectix Virtual PC Mouse
 14 = /dev/touchscreen/ucb1x00  UCB 1x00 touchscreen
 15 = /dev/touchscreen/mk712    MK712 touchscreen
128 = /dev/beep         Fancy beep device
129 =
130 = /dev/watchdog     Watchdog timer port
131 = /dev/temperature  Machine internal temperature
132 = /dev/hwtrap       Hardware fault trap
133 = /dev/exttrp       External device trap
134 = /dev/apm_bios     Advanced Power Management BIOS
135 = /dev/rtc          Real Time Clock
137 = /dev/vhci         Bluetooth virtual HCI driver
139 = /dev/openprom     SPARC OpenBoot PROM
140 = /dev/relay8       Berkshire Products Octal relay card
141 = /dev/relay16      Berkshire Products ISO-16 relay card
142 =
143 = /dev/pciconf      PCI configuration space
144 = /dev/nvram        Non-volatile configuration RAM
145 = /dev/hfmodem      Soundcard shortwave modem control
146 = /dev/graphics     Linux/SGI graphics device
147 = /dev/opengl       Linux/SGI OpenGL pipe
148 = /dev/gfx          Linux/SGI graphics effects device
149 = /dev/input/mouse  Linux/SGI Irix emulation mouse
150 = /dev/input/keyboard Linux/SGI Irix emulation keyboard
151 = /dev/led          Front panel LEDs
152 = /dev/kpoll        Kernel Poll Driver
153 = /dev/mergemem     Memory merge device
154 = /dev/pmu          Macintosh PowerBook power manager
155 =
156 = /dev/lcd          Front panel LCD display
157 = /dev/ac           Applicom Intl Profibus card
158 = /dev/nwbutton     Netwinder external button
159 = /dev/nwdebug      Netwinder debug interface
160 = /dev/nwflash      Netwinder flash memory
161 = /dev/userdma      User-space DMA access
162 = /dev/smbus        System Management Bus
163 = /dev/lik          Logitech Internet Keyboard
164 = /dev/ipmo         Intel Intelligent Platform Management
165 = /dev/vmmon        VMware virtual machine monitor
166 = /dev/i2o/ctl      I2O configuration manager
167 = /dev/specialix_sxctl Specialix serial control
168 = /dev/tcldrv       Technology Concepts serial control
```

```
                169 = /dev/specialix_rioctl Specialix RIO serial control
                170 = /dev/thinkpad/thinkpad    IBM Thinkpad devices
                171 = /dev/srripc       QNX4 API IPC manager
                172 = /dev/usemaclone   Semaphore clone device
                173 = /dev/ipmikcs      Intelligent Platform Management
                174 = /dev/uctrl        SPARCbook 3 microcontroller
                175 = /dev/agpgart      AGP Graphics Address Remapping Table
                176 = /dev/gtrsc        Gorgy Timing radio clock
                177 = /dev/cbm          Serial CBM bus
                178 = /dev/jsflash      JavaStation OS flash SIMM
                179 = /dev/xsvc         High-speed shared-mem/semaphore service
                180 = /dev/vrbuttons    Vr41xx button input device
                181 = /dev/toshiba      Toshiba laptop SMM support
                182 = /dev/perfctr      Performance-monitoring counters
                183 = /dev/hwrng        Generic random number generator
                184 = /dev/cpu/microcode CPU microcode update interface
                186 = /dev/atomicps     Atomic snapshot of process state data
                187 = /dev/irnet        IrNET device
                188 = /dev/smbusbios    SMBus BIOS
                189 = /dev/ussp_ctl     User space serial port control
                190 = /dev/crash        Mission Critical Linux crash dump␣
↪facility
                191 = /dev/pcl181       <information missing>
                192 = /dev/nas_xbus     NAS xbus LCD/buttons access
                193 = /dev/d7s          SPARC 7-segment display
                194 = /dev/zkshim       Zero-Knowledge network shim control
                195 = /dev/elographics/e2201    Elographics touchscreen␣
↪E271-2201
                196 = /dev/vfio/vfio    VFIO userspace driver interface
                197 = /dev/pxa3xx-gcu   PXA3xx graphics controller unit driver
                198 = /dev/sexec        Signed executable interface
                199 = /dev/scanners/cuecat :CueCat barcode scanner
                200 = /dev/net/tun      TAP/TUN network device
                201 = /dev/button/gulpb Transmeta GULP-B buttons
                202 = /dev/emd/ctl      Enhanced Metadisk RAID (EMD) control
                203 = /dev/cuse         Cuse (character device in user-space)
                204 = /dev/video/em8300         EM8300 DVD decoder control
                205 = /dev/video/em8300_mv      EM8300 DVD decoder video
                206 = /dev/video/em8300_ma      EM8300 DVD decoder audio
                207 = /dev/video/em8300_sp      EM8300 DVD decoder subpicture
                208 = /dev/compaq/cpqphpc       Compaq PCI Hot Plug Controller
                209 = /dev/compaq/cpqrid        Compaq Remote Insight Driver
                210 = /dev/impi/bt      IMPI coprocessor block transfer
                211 = /dev/impi/smic    IMPI coprocessor stream interface
                212 = /dev/watchdogs/0  First watchdog device
                213 = /dev/watchdogs/1  Second watchdog device
                214 = /dev/watchdogs/2  Third watchdog device
                215 = /dev/watchdogs/3  Fourth watchdog device
                216 = /dev/fujitsu/apanel       Fujitsu/Siemens application␣
↪panel
                217 = /dev/ni/natmotn           National Instruments Motion
```

```
                218 = /dev/kchuid       Inter-process chuid control
                219 = /dev/modems/mwave MWave modem firmware upload
                220 = /dev/mptctl       Message passing technology (MPT)␣
↪control
                221 = /dev/mvista/hssdsi        Montavista PICMG hot swap␣
↪system driver
                222 = /dev/mvista/hasi          Montavista PICMG high␣
↪availability
                223 = /dev/input/uinput         User level driver support for␣
↪input
                224 = /dev/tpm          TCPA TPM driver
                225 = /dev/pps          Pulse Per Second driver
                226 = /dev/systrace     Systrace device
                227 = /dev/mcelog       X86_64 Machine Check Exception driver
                228 = /dev/hpet         HPET driver
                229 = /dev/fuse         Fuse (virtual filesystem in user-space)
                230 = /dev/midishare    MidiShare driver
                231 = /dev/snapshot     System memory snapshot device
                232 = /dev/kvm          Kernel-based virtual machine (hardware␣
↪virtualization extensions)
                233 = /dev/kmview       View-OS A process with a view
                234 = /dev/btrfs-control        Btrfs control device
                235 = /dev/autofs       Autofs control device
                236 = /dev/mapper/control       Device-Mapper control device
                237 = /dev/loop-control Loopback control device
                238 = /dev/vhost-net    Host kernel accelerator for virtio net
                239 = /dev/uhid         User-space I/O driver support for HID␣
↪subsystem
                240 = /dev/userio       Serio driver testing device
                241 = /dev/vhost-vsock  Host kernel driver for virtio vsock
                242 = /dev/rfkill       Turning off radio transmissions␣
↪(rfkill)

                243-254                 Reserved for local use
                255                     Reserved for MISC_DYNAMIC_MINOR

 11 char        Raw keyboard device     (Linux/SPARC only)
                  0 = /dev/kbd          Raw keyboard device

 11 char        Serial Mux device       (Linux/PA-RISC only)
                  0 = /dev/ttyB0        First mux port
                  1 = /dev/ttyB1        Second mux port
                    ...

 11 block       SCSI CD-ROM devices
                  0 = /dev/scd0         First SCSI CD-ROM
                  1 = /dev/scd1         Second SCSI CD-ROM
                    ...

                The prefix /dev/sr (instead of /dev/scd) has been deprecated.
```

```
12 char       QIC-02 tape
                2 = /dev/ntpqic11     QIC-11, no rewind-on-close
                3 = /dev/tpqic11      QIC-11, rewind-on-close
                4 = /dev/ntpqic24     QIC-24, no rewind-on-close
                5 = /dev/tpqic24      QIC-24, rewind-on-close
                6 = /dev/ntpqic120    QIC-120, no rewind-on-close
                7 = /dev/tpqic120     QIC-120, rewind-on-close
                8 = /dev/ntpqic150    QIC-150, no rewind-on-close
                9 = /dev/tpqic150     QIC-150, rewind-on-close

              The device names specified are proposed -- if there
              are "standard" names for these devices, please let me know.

12 block

13 char       Input core
                0 = /dev/input/js0    First joystick
                1 = /dev/input/js1    Second joystick
                   ...
               32 = /dev/input/mouse0 First mouse
               33 = /dev/input/mouse1 Second mouse
                   ...
               63 = /dev/input/mice   Unified mouse
               64 = /dev/input/event0 First event queue
               65 = /dev/input/event1 Second event queue
                   ...

              Each device type has 5 bits (32 minors).

13 block      Previously used for the XT disk (/dev/xdN)
              Deleted in kernel v3.9.

14 char       Open Sound System (OSS)
                0 = /dev/mixer        Mixer control
                1 = /dev/sequencer    Audio sequencer
                2 = /dev/midi00       First MIDI port
                3 = /dev/dsp          Digital audio
                4 = /dev/audio        Sun-compatible digital audio
                6 =
                7 = /dev/audioctl     SPARC audio control device
                8 = /dev/sequencer2   Sequencer -- alternate device
               16 = /dev/mixer1       Second soundcard mixer control
               17 = /dev/patmgr0      Sequencer patch manager
               18 = /dev/midi01       Second MIDI port
               19 = /dev/dsp1         Second soundcard digital audio
               20 = /dev/audio1       Second soundcard Sun digital audio
               33 = /dev/patmgr1      Sequencer patch manager
               34 = /dev/midi02       Third MIDI port
               50 = /dev/midi03       Fourth MIDI port

14 block
```

```
15 char       Joystick
                0 = /dev/js0          First analog joystick
                1 = /dev/js1          Second analog joystick
                  ...
              128 = /dev/djs0         First digital joystick
              129 = /dev/djs1         Second digital joystick
                  ...
15 block      Sony CDU-31A/CDU-33A CD-ROM
                0 = /dev/sonycd       Sony CDU-31a CD-ROM

16 char       Non-SCSI scanners
                0 = /dev/gs4500       Genius 4500 handheld scanner

16 block      GoldStar CD-ROM
                0 = /dev/gscd         GoldStar CD-ROM

17 char       OBSOLETE (was Chase serial card)
                0 = /dev/ttyH0        First Chase port
                1 = /dev/ttyH1        Second Chase port
                  ...
17 block      Optics Storage CD-ROM
                0 = /dev/optcd        Optics Storage CD-ROM

18 char       OBSOLETE (was Chase serial card - alternate devices)
                0 = /dev/cuh0         Callout device for ttyH0
                1 = /dev/cuh1         Callout device for ttyH1
                  ...
18 block      Sanyo CD-ROM
                0 = /dev/sjcd         Sanyo CD-ROM

19 block      "Double" compressed disk
                0 = /dev/double0      First compressed disk
                  ...
                7 = /dev/double7      Eighth compressed disk
              128 = /dev/cdouble0     Mirror of first compressed disk
                  ...
              135 = /dev/cdouble7     Mirror of eighth compressed disk

              See the Double documentation for the meaning of the
              mirror devices.

20 block      Hitachi CD-ROM (under development)
                0 = /dev/hitcd        Hitachi CD-ROM

21 char       Generic SCSI access
                0 = /dev/sg0          First generic SCSI device
                1 = /dev/sg1          Second generic SCSI device
                  ...

              Most distributions name these /dev/sga, /dev/sgb...;
```

```
                this sets an unnecessary limit of 26 SCSI devices in
                the system and is counter to standard Linux
                device-naming practice.

  21 block      Acorn MFM hard drive interface
                  0 = /dev/mfma         First MFM drive whole disk
                 64 = /dev/mfmb         Second MFM drive whole disk

                This device is used on the ARM-based Acorn RiscPC.
                Partitions are handled the same way as for IDE disks
                (see major number 3).

  22 char       Digiboard serial card
                  0 = /dev/ttyD0        First Digiboard port
                  1 = /dev/ttyD1        Second Digiboard port
                      ...
  22 block      Second IDE hard disk/CD-ROM interface
                  0 = /dev/hdc          Master: whole disk (or CD-ROM)
                 64 = /dev/hdd          Slave: whole disk (or CD-ROM)

                Partitions are handled the same way as for the first
                interface (see major number 3).

  23 char       Digiboard serial card - alternate devices
                  0 = /dev/cud0         Callout device for ttyD0
                  1 = /dev/cud1         Callout device for ttyD1
                        ...
  23 block      Mitsumi proprietary CD-ROM
                  0 = /dev/mcd          Mitsumi CD-ROM

  24 char       Stallion serial card
                  0 = /dev/ttyE0        Stallion port 0 card 0
                  1 = /dev/ttyE1        Stallion port 1 card 0
                      ...
                 64 = /dev/ttyE64       Stallion port 0 card 1
                 65 = /dev/ttyE65       Stallion port 1 card 1
                        ...
                128 = /dev/ttyE128      Stallion port 0 card 2
                129 = /dev/ttyE129      Stallion port 1 card 2
                      ...
                192 = /dev/ttyE192      Stallion port 0 card 3
                193 = /dev/ttyE193      Stallion port 1 card 3
                      ...
  24 block      Sony CDU-535 CD-ROM
                  0 = /dev/cdu535       Sony CDU-535 CD-ROM

  25 char       Stallion serial card - alternate devices
                  0 = /dev/cue0         Callout device for ttyE0
                  1 = /dev/cue1         Callout device for ttyE1
                      ...
                 64 = /dev/cue64        Callout device for ttyE64
```

```
                 65 = /dev/cue65        Callout device for ttyE65
                    ...
                128 = /dev/cue128       Callout device for ttyE128
                129 = /dev/cue129       Callout device for ttyE129
                    ...
                192 = /dev/cue192       Callout device for ttyE192
                193 = /dev/cue193       Callout device for ttyE193
                      ...
 25 block       First Matsushita (Panasonic/SoundBlaster) CD-ROM
                  0 = /dev/sbpcd0       Panasonic CD-ROM controller 0 unit 0
                  1 = /dev/sbpcd1       Panasonic CD-ROM controller 0 unit 1
                  2 = /dev/sbpcd2       Panasonic CD-ROM controller 0 unit 2
                  3 = /dev/sbpcd3       Panasonic CD-ROM controller 0 unit 3

 26 char

 26 block       Second Matsushita (Panasonic/SoundBlaster) CD-ROM
                  0 = /dev/sbpcd4       Panasonic CD-ROM controller 1 unit 0
                  1 = /dev/sbpcd5       Panasonic CD-ROM controller 1 unit 1
                  2 = /dev/sbpcd6       Panasonic CD-ROM controller 1 unit 2
                  3 = /dev/sbpcd7       Panasonic CD-ROM controller 1 unit 3

 27 char        QIC-117 tape
                  0 = /dev/qft0         Unit 0, rewind-on-close
                  1 = /dev/qft1         Unit 1, rewind-on-close
                  2 = /dev/qft2         Unit 2, rewind-on-close
                  3 = /dev/qft3         Unit 3, rewind-on-close
                  4 = /dev/nqft0        Unit 0, no rewind-on-close
                  5 = /dev/nqft1        Unit 1, no rewind-on-close
                  6 = /dev/nqft2        Unit 2, no rewind-on-close
                  7 = /dev/nqft3        Unit 3, no rewind-on-close
                 16 = /dev/zqft0        Unit 0, rewind-on-close, compression
                 17 = /dev/zqft1        Unit 1, rewind-on-close, compression
                 18 = /dev/zqft2        Unit 2, rewind-on-close, compression
                 19 = /dev/zqft3        Unit 3, rewind-on-close, compression
                 20 = /dev/nzqft0       Unit 0, no rewind-on-close, compression
                 21 = /dev/nzqft1       Unit 1, no rewind-on-close, compression
                 22 = /dev/nzqft2       Unit 2, no rewind-on-close, compression
                 23 = /dev/nzqft3       Unit 3, no rewind-on-close, compression
                 32 = /dev/rawqft0      Unit 0, rewind-on-close, no file marks
                 33 = /dev/rawqft1      Unit 1, rewind-on-close, no file marks
                 34 = /dev/rawqft2      Unit 2, rewind-on-close, no file marks
                 35 = /dev/rawqft3      Unit 3, rewind-on-close, no file marks
                 36 = /dev/nrawqft0     Unit 0, no rewind-on-close, no file␣
↪marks
                 37 = /dev/nrawqft1     Unit 1, no rewind-on-close, no file␣
↪marks
                 38 = /dev/nrawqft2     Unit 2, no rewind-on-close, no file␣
↪marks
                 39 = /dev/nrawqft3     Unit 3, no rewind-on-close, no file␣
↪marks
```

```
27 block      Third Matsushita (Panasonic/SoundBlaster) CD-ROM
                0 = /dev/sbpcd8       Panasonic CD-ROM controller 2 unit 0
                1 = /dev/sbpcd9       Panasonic CD-ROM controller 2 unit 1
                2 = /dev/sbpcd10      Panasonic CD-ROM controller 2 unit 2
                3 = /dev/sbpcd11      Panasonic CD-ROM controller 2 unit 3

28 char       Stallion serial card - card programming
                0 = /dev/staliomem0   First Stallion card I/O memory
                1 = /dev/staliomem1   Second Stallion card I/O memory
                2 = /dev/staliomem2   Third Stallion card I/O memory
                3 = /dev/staliomem3   Fourth Stallion card I/O memory

28 char       Atari SLM ACSI laser printer (68k/Atari)
                0 = /dev/slm0         First SLM laser printer
                1 = /dev/slm1         Second SLM laser printer
                    ...
28 block      Fourth Matsushita (Panasonic/SoundBlaster) CD-ROM
                0 = /dev/sbpcd12      Panasonic CD-ROM controller 3 unit 0
                1 = /dev/sbpcd13      Panasonic CD-ROM controller 3 unit 1
                2 = /dev/sbpcd14      Panasonic CD-ROM controller 3 unit 2
                3 = /dev/sbpcd15      Panasonic CD-ROM controller 3 unit 3

28 block      ACSI disk (68k/Atari)
                0 = /dev/ada          First ACSI disk whole disk
               16 = /dev/adb          Second ACSI disk whole disk
               32 = /dev/adc          Third ACSI disk whole disk
                    ...
              240 = /dev/adp          16th ACSI disk whole disk

              Partitions are handled in the same way as for IDE
              disks (see major number 3) except that the limit on
              partitions is 15, like SCSI.

29 char       Universal frame buffer
                0 = /dev/fb0          First frame buffer
                1 = /dev/fb1          Second frame buffer
                    ...
               31 = /dev/fb31         32nd frame buffer

29 block      Aztech/Orchid/Okano/Wearnes CD-ROM
                0 = /dev/aztcd        Aztech CD-ROM

30 char       iBCS-2 compatibility devices
                0 = /dev/socksys      Socket access
                1 = /dev/spx          SVR3 local X interface
               32 = /dev/inet/ip      Network access
               33 = /dev/inet/icmp
               34 = /dev/inet/ggp
               35 = /dev/inet/ipip
               36 = /dev/inet/tcp
```

```
                 37 = /dev/inet/egp
                 38 = /dev/inet/pup
                 39 = /dev/inet/udp
                 40 = /dev/inet/idp
                 41 = /dev/inet/rawip

                Additionally, iBCS-2 requires the following links:

                /dev/ip -> /dev/inet/ip
                /dev/icmp -> /dev/inet/icmp
                /dev/ggp -> /dev/inet/ggp
                /dev/ipip -> /dev/inet/ipip
                /dev/tcp -> /dev/inet/tcp
                /dev/egp -> /dev/inet/egp
                /dev/pup -> /dev/inet/pup
                /dev/udp -> /dev/inet/udp
                /dev/idp -> /dev/inet/idp
                /dev/rawip -> /dev/inet/rawip
                /dev/inet/arp -> /dev/inet/udp
                /dev/inet/rip -> /dev/inet/udp
                /dev/nfsd -> /dev/socksys
                /dev/X0R -> /dev/null (? apparently not required ?)

30 block        Philips LMS CM-205 CD-ROM
                  0 = /dev/cm205cd      Philips LMS CM-205 CD-ROM

                /dev/lmscd is an older name for this device.  This
                driver does not work with the CM-205MS CD-ROM.

31 char         MPU-401 MIDI
                  0 = /dev/mpu401data   MPU-401 data port
                  1 = /dev/mpu401stat   MPU-401 status port

31 block        ROM/flash memory card
                  0 = /dev/rom0         First ROM card (rw)
                      ...
                  7 = /dev/rom7         Eighth ROM card (rw)
                  8 = /dev/rrom0        First ROM card (ro)
                     ...
                 15 = /dev/rrom7        Eighth ROM card (ro)
                 16 = /dev/flash0       First flash memory card (rw)
                     ...
                 23 = /dev/flash7       Eighth flash memory card (rw)
                 24 = /dev/rflash0      First flash memory card (ro)
                     ...
                 31 = /dev/rflash7      Eighth flash memory card (ro)

                The read-write (rw) devices support back-caching
                written data in RAM, as well as writing to flash RAM
                devices.  The read-only devices (ro) support reading
                only.
```

```
32 char       Specialix serial card
                0 = /dev/ttyX0        First Specialix port
                1 = /dev/ttyX1        Second Specialix port
                  ...
32 block      Philips LMS CM-206 CD-ROM
                0 = /dev/cm206cd      Philips LMS CM-206 CD-ROM

33 char       Specialix serial card - alternate devices
                0 = /dev/cux0         Callout device for ttyX0
                1 = /dev/cux1         Callout device for ttyX1
                  ...
33 block      Third IDE hard disk/CD-ROM interface
                0 = /dev/hde          Master: whole disk (or CD-ROM)
               64 = /dev/hdf          Slave: whole disk (or CD-ROM)

              Partitions are handled the same way as for the first
              interface (see major number 3).

34 char       Z8530 HDLC driver
                0 = /dev/scc0         First Z8530, first port
                1 = /dev/scc1         First Z8530, second port
                2 = /dev/scc2         Second Z8530, first port
                3 = /dev/scc3         Second Z8530, second port
                  ...

              In a previous version these devices were named
              /dev/sc1 for /dev/scc0, /dev/sc2 for /dev/scc1, and so
              on.

34 block      Fourth IDE hard disk/CD-ROM interface
                0 = /dev/hdg          Master: whole disk (or CD-ROM)
               64 = /dev/hdh          Slave: whole disk (or CD-ROM)

              Partitions are handled the same way as for the first
              interface (see major number 3).

35 char       tclmidi MIDI driver
                0 = /dev/midi0        First MIDI port, kernel timed
                1 = /dev/midi1        Second MIDI port, kernel timed
                2 = /dev/midi2        Third MIDI port, kernel timed
                3 = /dev/midi3        Fourth MIDI port, kernel timed
               64 = /dev/rmidi0       First MIDI port, untimed
               65 = /dev/rmidi1       Second MIDI port, untimed
               66 = /dev/rmidi2       Third MIDI port, untimed
               67 = /dev/rmidi3       Fourth MIDI port, untimed
              128 = /dev/smpte0       First MIDI port, SMPTE timed
              129 = /dev/smpte1       Second MIDI port, SMPTE timed
              130 = /dev/smpte2       Third MIDI port, SMPTE timed
              131 = /dev/smpte3       Fourth MIDI port, SMPTE timed
```

```
35 block      Slow memory ramdisk
                0 = /dev/slram        Slow memory ramdisk

36 char       Netlink support
                0 = /dev/route        Routing, device updates, kernel to user
                1 = /dev/skip         enSKIP security cache control
                3 = /dev/fwmonitor    Firewall packet copies
               16 = /dev/tap0         First Ethertap device
                  ...
               31 = /dev/tap15        16th Ethertap device

36 block      OBSOLETE (was MCA ESDI hard disk)

37 char       IDE tape
                0 = /dev/ht0          First IDE tape
                1 = /dev/ht1          Second IDE tape
                  ...
              128 = /dev/nht0         First IDE tape, no rewind-on-close
              129 = /dev/nht1         Second IDE tape, no rewind-on-close
                  ...

              Currently, only one IDE tape drive is supported.

37 block      Zorro II ramdisk
                0 = /dev/z2ram        Zorro II ramdisk

38 char       Myricom PCI Myrinet board
                0 = /dev/mlanai0      First Myrinet board
                1 = /dev/mlanai1      Second Myrinet board
                  ...

              This device is used for status query, board control
              and "user level packet I/O."  This board is also
              accessible as a standard networking "eth" device.

38 block      OBSOLETE (was Linux/AP+)

39 char       ML-16P experimental I/O board
                0 = /dev/ml16pa-a0    First card, first analog channel
                1 = /dev/ml16pa-a1    First card, second analog channel
                  ...
               15 = /dev/ml16pa-a15   First card, 16th analog channel
               16 = /dev/ml16pa-d     First card, digital lines
               17 = /dev/ml16pa-c0    First card, first counter/timer
               18 = /dev/ml16pa-c1    First card, second counter/timer
               19 = /dev/ml16pa-c2    First card, third counter/timer
               32 = /dev/ml16pb-a0    Second card, first analog channel
               33 = /dev/ml16pb-a1    Second card, second analog channel
                  ...
               47 = /dev/ml16pb-a15   Second card, 16th analog channel
               48 = /dev/ml16pb-d     Second card, digital lines
```

```
                 49 = /dev/ml16pb-c0    Second card, first counter/timer
                 50 = /dev/ml16pb-c1    Second card, second counter/timer
                 51 = /dev/ml16pb-c2    Second card, third counter/timer
                      ...
39 block

40 char

40 block

41 char       Yet Another Micro Monitor
                 0 = /dev/yamm         Yet Another Micro Monitor

41 block

42 char       Demo/sample use

42 block      Demo/sample use

              This number is intended for use in sample code, as
              well as a general "example" device number.  It
              should never be used for a device driver that is being
              distributed; either obtain an official number or use
              the local/experimental range.  The sudden addition or
              removal of a driver with this number should not cause
              ill effects to the system (bugs excepted.)

              IN PARTICULAR, ANY DISTRIBUTION WHICH CONTAINS A
              DEVICE DRIVER USING MAJOR NUMBER 42 IS NONCOMPLIANT.

43 char       isdn4linux virtual modem
                 0 = /dev/ttyI0        First virtual modem
                    ...
                63 = /dev/ttyI63       64th virtual modem

43 block      Network block devices
                 0 = /dev/nb0          First network block device
                 1 = /dev/nb1          Second network block device
                    ...

              Network Block Device is somehow similar to loopback
              devices: If you read from it, it sends packet across
              network asking server for data. If you write to it, it
              sends packet telling server to write. It could be used
              to mounting filesystems over the net, swapping over
              the net, implementing block device in userland etc.

44 char       isdn4linux virtual modem - alternate devices
                 0 = /dev/cui0         Callout device for ttyI0
                    ...
                63 = /dev/cui63        Callout device for ttyI63
```

```
44 block      Flash Translation Layer (FTL) filesystems
                0 = /dev/ftla         FTL on first Memory Technology Device
               16 = /dev/ftlb         FTL on second Memory Technology Device
               32 = /dev/ftlc         FTL on third Memory Technology Device
                  ...
              240 = /dev/ftlp         FTL on 16th Memory Technology Device

              Partitions are handled in the same way as for IDE
              disks (see major number 3) except that the partition
              limit is 15 rather than 63 per disk (same as SCSI.)

45 char       isdn4linux ISDN BRI driver
                0 = /dev/isdn0        First virtual B channel raw data
                  ...
               63 = /dev/isdn63       64th virtual B channel raw data
               64 = /dev/isdnctrl0    First channel control/debug
                  ...
              127 = /dev/isdnctrl63   64th channel control/debug

              128 = /dev/ippp0        First SyncPPP device
                  ...
              191 = /dev/ippp63       64th SyncPPP device

              255 = /dev/isdninfo     ISDN monitor interface

45 block      Parallel port IDE disk devices
                0 = /dev/pda          First parallel port IDE disk
               16 = /dev/pdb          Second parallel port IDE disk
               32 = /dev/pdc          Third parallel port IDE disk
               48 = /dev/pdd          Fourth parallel port IDE disk

              Partitions are handled in the same way as for IDE
              disks (see major number 3) except that the partition
              limit is 15 rather than 63 per disk.

46 char       Comtrol Rocketport serial card
                0 = /dev/ttyR0        First Rocketport port
                1 = /dev/ttyR1        Second Rocketport port
                  ...
46 block      Parallel port ATAPI CD-ROM devices
                0 = /dev/pcd0         First parallel port ATAPI CD-ROM
                1 = /dev/pcd1         Second parallel port ATAPI CD-ROM
                2 = /dev/pcd2         Third parallel port ATAPI CD-ROM
                3 = /dev/pcd3         Fourth parallel port ATAPI CD-ROM

47 char       Comtrol Rocketport serial card - alternate devices
                0 = /dev/cur0         Callout device for ttyR0
                1 = /dev/cur1         Callout device for ttyR1
                  ...
47 block      Parallel port ATAPI disk devices
```

```
                0 = /dev/pf0          First parallel port ATAPI disk
                1 = /dev/pf1          Second parallel port ATAPI disk
                2 = /dev/pf2          Third parallel port ATAPI disk
                3 = /dev/pf3          Fourth parallel port ATAPI disk

              This driver is intended for floppy disks and similar
              devices and hence does not support partitioning.

48 char       SDL RISCom serial card
                0 = /dev/ttyL0        First RISCom port
                1 = /dev/ttyL1        Second RISCom port
                  ...
48 block      Mylex DAC960 PCI RAID controller; first controller
                0 = /dev/rd/c0d0      First disk, whole disk
                8 = /dev/rd/c0d1      Second disk, whole disk
                  ...
              248 = /dev/rd/c0d31     32nd disk, whole disk

              For partitions add:
                0 = /dev/rd/c?d?      Whole disk
                1 = /dev/rd/c?d?p1    First partition
                  ...
                7 = /dev/rd/c?d?p7    Seventh partition

49 char       SDL RISCom serial card - alternate devices
                0 = /dev/cul0         Callout device for ttyL0
                1 = /dev/cul1         Callout device for ttyL1
                  ...
49 block      Mylex DAC960 PCI RAID controller; second controller
                0 = /dev/rd/c1d0      First disk, whole disk
                8 = /dev/rd/c1d1      Second disk, whole disk
                  ...
              248 = /dev/rd/c1d31     32nd disk, whole disk

              Partitions are handled as for major 48.

50 char       Reserved for GLINT

50 block      Mylex DAC960 PCI RAID controller; third controller
                0 = /dev/rd/c2d0      First disk, whole disk
                8 = /dev/rd/c2d1      Second disk, whole disk
                  ...
              248 = /dev/rd/c2d31     32nd disk, whole disk

51 char       Baycom radio modem OR Radio Tech BIM-XXX-RS232 radio modem
                0 = /dev/bc0          First Baycom radio modem
                1 = /dev/bc1          Second Baycom radio modem
                  ...
51 block      Mylex DAC960 PCI RAID controller; fourth controller
                0 = /dev/rd/c3d0      First disk, whole disk
                8 = /dev/rd/c3d1      Second disk, whole disk
```

```
                ...
              248 = /dev/rd/c3d31     32nd disk, whole disk

              Partitions are handled as for major 48.

52 char       Spellcaster DataComm/BRI ISDN card
                0 = /dev/dcbri0       First DataComm card
                1 = /dev/dcbri1       Second DataComm card
                2 = /dev/dcbri2       Third DataComm card
                3 = /dev/dcbri3       Fourth DataComm card

52 block      Mylex DAC960 PCI RAID controller; fifth controller
                0 = /dev/rd/c4d0      First disk, whole disk
                8 = /dev/rd/c4d1      Second disk, whole disk
                ...
              248 = /dev/rd/c4d31     32nd disk, whole disk

              Partitions are handled as for major 48.

53 char       BDM interface for remote debugging MC683xx microcontrollers
                0 = /dev/pd_bdm0      PD BDM interface on lp0
                1 = /dev/pd_bdm1      PD BDM interface on lp1
                2 = /dev/pd_bdm2      PD BDM interface on lp2
                4 = /dev/icd_bdm0     ICD BDM interface on lp0
                5 = /dev/icd_bdm1     ICD BDM interface on lp1
                6 = /dev/icd_bdm2     ICD BDM interface on lp2

              This device is used for the interfacing to the MC683xx
              microcontrollers via Background Debug Mode by use of a
              Parallel Port interface. PD is the Motorola Public
              Domain Interface and ICD is the commercial interface
              by P&E.

53 block      Mylex DAC960 PCI RAID controller; sixth controller
                0 = /dev/rd/c5d0      First disk, whole disk
                8 = /dev/rd/c5d1      Second disk, whole disk
                ...
              248 = /dev/rd/c5d31     32nd disk, whole disk

              Partitions are handled as for major 48.

54 char       Electrocardiognosis Holter serial card
                0 = /dev/holter0      First Holter port
                1 = /dev/holter1      Second Holter port
                2 = /dev/holter2      Third Holter port

              A custom serial card used by Electrocardiognosis SRL
              <mseritan@ottonel.pub.ro> to transfer data from Holter
              24-hour heart monitoring equipment.

54 block      Mylex DAC960 PCI RAID controller; seventh controller
```

```
                 0 = /dev/rd/c6d0      First disk, whole disk
                 8 = /dev/rd/c6d1      Second disk, whole disk
                   ...
               248 = /dev/rd/c6d31     32nd disk, whole disk

               Partitions are handled as for major 48.

 55 char       DSP56001 digital signal processor
                 0 = /dev/dsp56k       First DSP56001

 55 block      Mylex DAC960 PCI RAID controller; eighth controller
                 0 = /dev/rd/c7d0      First disk, whole disk
                 8 = /dev/rd/c7d1      Second disk, whole disk
                   ...
               248 = /dev/rd/c7d31     32nd disk, whole disk

               Partitions are handled as for major 48.

 56 char       Apple Desktop Bus
                 0 = /dev/adb          ADB bus control

               Additional devices will be added to this number, all
               starting with /dev/adb.

 56 block      Fifth IDE hard disk/CD-ROM interface
                 0 = /dev/hdi          Master: whole disk (or CD-ROM)
                64 = /dev/hdj          Slave: whole disk (or CD-ROM)

               Partitions are handled the same way as for the first
               interface (see major number 3).

 57 char       Hayes ESP serial card
                 0 = /dev/ttyP0        First ESP port
                 1 = /dev/ttyP1        Second ESP port
                   ...

 57 block      Sixth IDE hard disk/CD-ROM interface
                 0 = /dev/hdk          Master: whole disk (or CD-ROM)
                64 = /dev/hdl          Slave: whole disk (or CD-ROM)

               Partitions are handled the same way as for the first
               interface (see major number 3).

 58 char       Hayes ESP serial card - alternate devices
                 0 = /dev/cup0         Callout device for ttyP0
                 1 = /dev/cup1         Callout device for ttyP1
                   ...

 58 block      Reserved for logical volume manager

 59 char       sf firewall package
```

```
                0 = /dev/firewall     Communication with sf kernel module

59 block      Generic PDA filesystem device
                0 = /dev/pda0         First PDA device
                1 = /dev/pda1         Second PDA device
                  ...

              The pda devices are used to mount filesystems on
              remote pda's (basically slow handheld machines with
              proprietary OS's and limited memory and storage
              running small fs translation drivers) through serial /
              IRDA / parallel links.

              NAMING CONFLICT -- PROPOSED REVISED NAME /dev/rpda0 etc

60-63 char    LOCAL/EXPERIMENTAL USE

60-63 block   LOCAL/EXPERIMENTAL USE
              Allocated for local/experimental use.  For devices not
              assigned official numbers, these ranges should be
              used in order to avoid conflicting with future assignments.

64 char       ENskip kernel encryption package
                0 = /dev/enskip       Communication with ENskip kernel module

64 block      Scramdisk/DriveCrypt encrypted devices
                0 = /dev/scramdisk/master    Master node for ioctls
                1 = /dev/scramdisk/1         First encrypted device
                2 = /dev/scramdisk/2         Second encrypted device
                ...
              255 = /dev/scramdisk/255       255th encrypted device

              The filename of the encrypted container and the passwords
              are sent via ioctls (using the sdmount tool) to the master
              node which then activates them via one of the
              /dev/scramdisk/x nodes for loop mounting (all handled
              through the sdmount tool).

              Requested by: andy@scramdisklinux.org

65 char       Sundance "plink" Transputer boards (obsolete, unused)
                0 = /dev/plink0       First plink device
                1 = /dev/plink1       Second plink device
                2 = /dev/plink2       Third plink device
                3 = /dev/plink3       Fourth plink device
               64 = /dev/rplink0      First plink device, raw
               65 = /dev/rplink1      Second plink device, raw
               66 = /dev/rplink2      Third plink device, raw
               67 = /dev/rplink3      Fourth plink device, raw
              128 = /dev/plink0d      First plink device, debug
              129 = /dev/plink1d      Second plink device, debug
```

```
                130 = /dev/plink2d      Third plink device, debug
                131 = /dev/plink3d      Fourth plink device, debug
                192 = /dev/rplink0d     First plink device, raw, debug
                193 = /dev/rplink1d     Second plink device, raw, debug
                194 = /dev/rplink2d     Third plink device, raw, debug
                195 = /dev/rplink3d     Fourth plink device, raw, debug

                This is a commercial driver; contact James Howes
                <jth@prosig.demon.co.uk> for information.

  65 block      SCSI disk devices (16-31)
                  0 = /dev/sdq          17th SCSI disk whole disk
                 16 = /dev/sdr          18th SCSI disk whole disk
                 32 = /dev/sds          19th SCSI disk whole disk
                    ...
                240 = /dev/sdaf         32nd SCSI disk whole disk

                Partitions are handled in the same way as for IDE
                disks (see major number 3) except that the limit on
                partitions is 15.

  66 char       YARC PowerPC PCI coprocessor card
                  0 = /dev/yppcpci0     First YARC card
                  1 = /dev/yppcpci1     Second YARC card
                    ...

  66 block      SCSI disk devices (32-47)
                  0 = /dev/sdag         33th SCSI disk whole disk
                 16 = /dev/sdah         34th SCSI disk whole disk
                 32 = /dev/sdai         35th SCSI disk whole disk
                    ...
                240 = /dev/sdav         48nd SCSI disk whole disk

                Partitions are handled in the same way as for IDE
                disks (see major number 3) except that the limit on
                partitions is 15.

  67 char       Coda network file system
                  0 = /dev/cfs0         Coda cache manager

                See http://www.coda.cs.cmu.edu for information about Coda.

  67 block      SCSI disk devices (48-63)
                  0 = /dev/sdaw         49th SCSI disk whole disk
                 16 = /dev/sdax         50th SCSI disk whole disk
                 32 = /dev/sday         51st SCSI disk whole disk
                    ...
                240 = /dev/sdbl         64th SCSI disk whole disk

                Partitions are handled in the same way as for IDE
                disks (see major number 3) except that the limit on
```

```
              partitions is 15.

68 char       CAPI 2.0 interface
                0 = /dev/capi20       Control device
                1 = /dev/capi20.00    First CAPI 2.0 application
                2 = /dev/capi20.01    Second CAPI 2.0 application
                  ...
               20 = /dev/capi20.19    19th CAPI 2.0 application

              ISDN CAPI 2.0 driver for use with CAPI 2.0
              applications; currently supports the AVM B1 card.

68 block      SCSI disk devices (64-79)
                0 = /dev/sdbm         65th SCSI disk whole disk
               16 = /dev/sdbn         66th SCSI disk whole disk
               32 = /dev/sdbo         67th SCSI disk whole disk
                  ...
              240 = /dev/sdcb         80th SCSI disk whole disk

              Partitions are handled in the same way as for IDE
              disks (see major number 3) except that the limit on
              partitions is 15.

69 char       MA16 numeric accelerator card
                0 = /dev/ma16         Board memory access

69 block      SCSI disk devices (80-95)
                0 = /dev/sdcc         81st SCSI disk whole disk
               16 = /dev/sdcd         82nd SCSI disk whole disk
               32 = /dev/sdce         83th SCSI disk whole disk
                  ...
              240 = /dev/sdcr         96th SCSI disk whole disk

              Partitions are handled in the same way as for IDE
              disks (see major number 3) except that the limit on
              partitions is 15.

70 char       SpellCaster Protocol Services Interface
                0 = /dev/apscfg       Configuration interface
                1 = /dev/apsauth      Authentication interface
                2 = /dev/apslog       Logging interface
                3 = /dev/apsdbg       Debugging interface
               64 = /dev/apsisdn      ISDN command interface
               65 = /dev/apsasync     Async command interface
              128 = /dev/apsmon       Monitor interface

70 block      SCSI disk devices (96-111)
                0 = /dev/sdcs         97th SCSI disk whole disk
               16 = /dev/sdct         98th SCSI disk whole disk
               32 = /dev/sdcu         99th SCSI disk whole disk
                  ...
```

```
                240 = /dev/sddh         112nd SCSI disk whole disk

                Partitions are handled in the same way as for IDE
                disks (see major number 3) except that the limit on
                partitions is 15.

  71 char       Computone IntelliPort II serial card
                  0 = /dev/ttyF0        IntelliPort II board 0, port 0
                  1 = /dev/ttyF1        IntelliPort II board 0, port 1
                      ...
                 63 = /dev/ttyF63       IntelliPort II board 0, port 63
                 64 = /dev/ttyF64       IntelliPort II board 1, port 0
                 65 = /dev/ttyF65       IntelliPort II board 1, port 1
                      ...
                127 = /dev/ttyF127      IntelliPort II board 1, port 63
                128 = /dev/ttyF128      IntelliPort II board 2, port 0
                129 = /dev/ttyF129      IntelliPort II board 2, port 1
                      ...
                191 = /dev/ttyF191      IntelliPort II board 2, port 63
                192 = /dev/ttyF192      IntelliPort II board 3, port 0
                193 = /dev/ttyF193      IntelliPort II board 3, port 1
                      ...
                255 = /dev/ttyF255      IntelliPort II board 3, port 63

  71 block      SCSI disk devices (112-127)
                  0 = /dev/sddi         113th SCSI disk whole disk
                 16 = /dev/sddj         114th SCSI disk whole disk
                 32 = /dev/sddk         115th SCSI disk whole disk
                      ...
                240 = /dev/sddx         128th SCSI disk whole disk

                Partitions are handled in the same way as for IDE
                disks (see major number 3) except that the limit on
                partitions is 15.

  72 char       Computone IntelliPort II serial card - alternate devices
                  0 = /dev/cuf0         Callout device for ttyF0
                  1 = /dev/cuf1         Callout device for ttyF1
                      ...
                 63 = /dev/cuf63        Callout device for ttyF63
                 64 = /dev/cuf64        Callout device for ttyF64
                 65 = /dev/cuf65        Callout device for ttyF65
                      ...
                127 = /dev/cuf127       Callout device for ttyF127
                128 = /dev/cuf128       Callout device for ttyF128
                129 = /dev/cuf129       Callout device for ttyF129
                      ...
                191 = /dev/cuf191       Callout device for ttyF191
                192 = /dev/cuf192       Callout device for ttyF192
                193 = /dev/cuf193       Callout device for ttyF193
                      ...
```

```
                255 = /dev/cuf255       Callout device for ttyF255

72 block        Compaq Intelligent Drive Array, first controller
                  0 = /dev/ida/c0d0     First logical drive whole disk
                 16 = /dev/ida/c0d1     Second logical drive whole disk
                    ...
                240 = /dev/ida/c0d15    16th logical drive whole disk

                Partitions are handled the same way as for Mylex
                DAC960 (see major number 48) except that the limit on
                partitions is 15.

73 char         Computone IntelliPort II serial card - control devices
                  0 = /dev/ip2ipl0      Loadware device for board 0
                  1 = /dev/ip2stat0     Status device for board 0
                  4 = /dev/ip2ipl1      Loadware device for board 1
                  5 = /dev/ip2stat1     Status device for board 1
                  8 = /dev/ip2ipl2      Loadware device for board 2
                  9 = /dev/ip2stat2     Status device for board 2
                 12 = /dev/ip2ipl3      Loadware device for board 3
                 13 = /dev/ip2stat3     Status device for board 3

73 block        Compaq Intelligent Drive Array, second controller
                  0 = /dev/ida/c1d0     First logical drive whole disk
                 16 = /dev/ida/c1d1     Second logical drive whole disk
                    ...
                240 = /dev/ida/c1d15    16th logical drive whole disk

                Partitions are handled the same way as for Mylex
                DAC960 (see major number 48) except that the limit on
                partitions is 15.

74 char         SCI bridge
                  0 = /dev/SCI/0        SCI device 0
                  1 = /dev/SCI/1        SCI device 1
                    ...

                Currently for Dolphin Interconnect Solutions' PCI-SCI
                bridge.

74 block        Compaq Intelligent Drive Array, third controller
                  0 = /dev/ida/c2d0     First logical drive whole disk
                 16 = /dev/ida/c2d1     Second logical drive whole disk
                    ...
                240 = /dev/ida/c2d15    16th logical drive whole disk

                Partitions are handled the same way as for Mylex
                DAC960 (see major number 48) except that the limit on
                partitions is 15.

75 char         Specialix IO8+ serial card
```

```
                  0 = /dev/ttyW0        First IO8+ port, first card
                  1 = /dev/ttyW1        Second IO8+ port, first card
                    ...
                  8 = /dev/ttyW8        First IO8+ port, second card
                    ...

75 block        Compaq Intelligent Drive Array, fourth controller
                  0 = /dev/ida/c3d0     First logical drive whole disk
                 16 = /dev/ida/c3d1     Second logical drive whole disk
                    ...
                240 = /dev/ida/c3d15    16th logical drive whole disk

                Partitions are handled the same way as for Mylex
                DAC960 (see major number 48) except that the limit on
                partitions is 15.

76 char         Specialix IO8+ serial card - alternate devices
                  0 = /dev/cuw0         Callout device for ttyW0
                  1 = /dev/cuw1         Callout device for ttyW1
                    ...
                  8 = /dev/cuw8         Callout device for ttyW8
                    ...

76 block        Compaq Intelligent Drive Array, fifth controller
                  0 = /dev/ida/c4d0     First logical drive whole disk
                 16 = /dev/ida/c4d1     Second logical drive whole disk
                    ...
                240 = /dev/ida/c4d15    16th logical drive whole disk

                Partitions are handled the same way as for Mylex
                DAC960 (see major number 48) except that the limit on
                partitions is 15.


77 char         ComScire Quantum Noise Generator
                  0 = /dev/qng          ComScire Quantum Noise Generator

77 block        Compaq Intelligent Drive Array, sixth controller
                  0 = /dev/ida/c5d0     First logical drive whole disk
                 16 = /dev/ida/c5d1     Second logical drive whole disk
                    ...
                240 = /dev/ida/c5d15    16th logical drive whole disk

                Partitions are handled the same way as for Mylex
                DAC960 (see major number 48) except that the limit on
                partitions is 15.

78 char         PAM Software's multimodem boards
                  0 = /dev/ttyM0        First PAM modem
                  1 = /dev/ttyM1        Second PAM modem
                    ...
```

```
78 block      Compaq Intelligent Drive Array, seventh controller
                0 = /dev/ida/c6d0     First logical drive whole disk
               16 = /dev/ida/c6d1     Second logical drive whole disk
                  ...
              240 = /dev/ida/c6d15    16th logical drive whole disk

              Partitions are handled the same way as for Mylex
              DAC960 (see major number 48) except that the limit on
              partitions is 15.

79 char       PAM Software's multimodem boards - alternate devices
                0 = /dev/cum0         Callout device for ttyM0
                1 = /dev/cum1         Callout device for ttyM1
                  ...

79 block      Compaq Intelligent Drive Array, eighth controller
                0 = /dev/ida/c7d0     First logical drive whole disk
               16 = /dev/ida/c7d1     Second logical drive whole disk
                  ...
              240 = /dev/ida/c715     16th logical drive whole disk

              Partitions are handled the same way as for Mylex
              DAC960 (see major number 48) except that the limit on
              partitions is 15.

80 char       Photometrics AT200 CCD camera
                0 = /dev/at200        Photometrics AT200 CCD camera

80 block      I2O hard disk
                0 = /dev/i2o/hda      First I2O hard disk, whole disk
               16 = /dev/i2o/hdb      Second I2O hard disk, whole disk
                  ...
              240 = /dev/i2o/hdp      16th I2O hard disk, whole disk

              Partitions are handled in the same way as for IDE
              disks (see major number 3) except that the limit on
              partitions is 15.

81 char       video4linux
                0 = /dev/video0       Video capture/overlay device
                  ...
               63 = /dev/video63      Video capture/overlay device
               64 = /dev/radio0       Radio device
                  ...
              127 = /dev/radio63      Radio device
              128 = /dev/swradio0     Software Defined Radio device
                  ...
              191 = /dev/swradio63    Software Defined Radio device
              224 = /dev/vbi0         Vertical blank interrupt
                  ...
```

```
                255 = /dev/vbi31        Vertical blank interrupt

                Minor numbers are allocated dynamically unless
                CONFIG_VIDEO_FIXED_MINOR_RANGES (default n)
                configuration option is set.

  81 block      I2O hard disk
                  0 = /dev/i2o/hdq      17th I2O hard disk, whole disk
                 16 = /dev/i2o/hdr      18th I2O hard disk, whole disk
                    ...
                240 = /dev/i2o/hdaf     32nd I2O hard disk, whole disk

                Partitions are handled in the same way as for IDE
                disks (see major number 3) except that the limit on
                partitions is 15.

  82 char       WiNRADiO communications receiver card
                  0 = /dev/winradio0    First WiNRADiO card
                  1 = /dev/winradio1    Second WiNRADiO card
                    ...

                The driver and documentation may be obtained from
                https://www.winradio.com/

  82 block      I2O hard disk
                  0 = /dev/i2o/hdag     33rd I2O hard disk, whole disk
                 16 = /dev/i2o/hdah     34th I2O hard disk, whole disk
                    ...
                240 = /dev/i2o/hdav     48th I2O hard disk, whole disk

                Partitions are handled in the same way as for IDE
                disks (see major number 3) except that the limit on
                partitions is 15.

  83 char       Matrox mga_vid video driver
                 0 = /dev/mga_vid0      1st video card
                 1 = /dev/mga_vid1      2nd video card
                 2 = /dev/mga_vid2      3rd video card
                  ...
                15 = /dev/mga_vid15     16th video card

  83 block      I2O hard disk
                  0 = /dev/i2o/hdaw     49th I2O hard disk, whole disk
                 16 = /dev/i2o/hdax     50th I2O hard disk, whole disk
                    ...
                240 = /dev/i2o/hdbl     64th I2O hard disk, whole disk

                Partitions are handled in the same way as for IDE
                disks (see major number 3) except that the limit on
                partitions is 15.
```

```
84 char       Ikon 1011[57] Versatec Greensheet Interface
                0 = /dev/ihcp0        First Greensheet port
                1 = /dev/ihcp1        Second Greensheet port

84 block      I2O hard disk
                0 = /dev/i2o/hdbm     65th I2O hard disk, whole disk
               16 = /dev/i2o/hdbn     66th I2O hard disk, whole disk
                  ...
              240 = /dev/i2o/hdcb     80th I2O hard disk, whole disk

              Partitions are handled in the same way as for IDE
              disks (see major number 3) except that the limit on
              partitions is 15.

85 char       Linux/SGI shared memory input queue
                0 = /dev/shmiq        Master shared input queue
                1 = /dev/qcntl0       First device pushed
                2 = /dev/qcntl1       Second device pushed
                  ...

85 block      I2O hard disk
                0 = /dev/i2o/hdcc     81st I2O hard disk, whole disk
               16 = /dev/i2o/hdcd     82nd I2O hard disk, whole disk
                  ...
              240 = /dev/i2o/hdcr     96th I2O hard disk, whole disk

              Partitions are handled in the same way as for IDE
              disks (see major number 3) except that the limit on
              partitions is 15.

86 char       SCSI media changer
                0 = /dev/sch0         First SCSI media changer
                1 = /dev/sch1         Second SCSI media changer
                  ...

86 block      I2O hard disk
                0 = /dev/i2o/hdcs     97th I2O hard disk, whole disk
               16 = /dev/i2o/hdct     98th I2O hard disk, whole disk
                  ...
              240 = /dev/i2o/hddh     112th I2O hard disk, whole disk

              Partitions are handled in the same way as for IDE
              disks (see major number 3) except that the limit on
              partitions is 15.

87 char       Sony Control-A1 stereo control bus
                0 = /dev/controla0    First device on chain
                1 = /dev/controla1    Second device on chain
                  ...

87 block      I2O hard disk
```

```
                 0 = /dev/i2o/hddi     113rd I2O hard disk, whole disk
                16 = /dev/i2o/hddj     114th I2O hard disk, whole disk
                   ...
               240 = /dev/i2o/hddx     128th I2O hard disk, whole disk

               Partitions are handled in the same way as for IDE
               disks (see major number 3) except that the limit on
               partitions is 15.

 88 char       COMX synchronous serial card
                 0 = /dev/comx0        COMX channel 0
                 1 = /dev/comx1        COMX channel 1
                   ...

 88 block      Seventh IDE hard disk/CD-ROM interface
                 0 = /dev/hdm          Master: whole disk (or CD-ROM)
                64 = /dev/hdn          Slave: whole disk (or CD-ROM)

               Partitions are handled the same way as for the first
               interface (see major number 3).

 89 char       I2C bus interface
                 0 = /dev/i2c-0        First I2C adapter
                 1 = /dev/i2c-1        Second I2C adapter
                   ...

 89 block      Eighth IDE hard disk/CD-ROM interface
                 0 = /dev/hdo          Master: whole disk (or CD-ROM)
                64 = /dev/hdp          Slave: whole disk (or CD-ROM)

               Partitions are handled the same way as for the first
               interface (see major number 3).

 90 char       Memory Technology Device (RAM, ROM, Flash)
                 0 = /dev/mtd0         First MTD (rw)
                 1 = /dev/mtdr0        First MTD (ro)
                   ...
                30 = /dev/mtd15        16th MTD (rw)
                31 = /dev/mtdr15       16th MTD (ro)

 90 block      Ninth IDE hard disk/CD-ROM interface
                 0 = /dev/hdq          Master: whole disk (or CD-ROM)
                64 = /dev/hdr          Slave: whole disk (or CD-ROM)

               Partitions are handled the same way as for the first
               interface (see major number 3).

 91 char       CAN-Bus devices
                 0 = /dev/can0         First CAN-Bus controller
                 1 = /dev/can1         Second CAN-Bus controller
                   ...
```

```
91 block      Tenth IDE hard disk/CD-ROM interface
                0 = /dev/hds          Master: whole disk (or CD-ROM)
               64 = /dev/hdt          Slave: whole disk (or CD-ROM)

              Partitions are handled the same way as for the first
              interface (see major number 3).

92 char       Reserved for ith Kommunikationstechnik MIC ISDN card

92 block      PPDD encrypted disk driver
                0 = /dev/ppdd0        First encrypted disk
                1 = /dev/ppdd1        Second encrypted disk
                  ...

              Partitions are handled in the same way as for IDE
              disks (see major number 3) except that the limit on
              partitions is 15.

93 char

93 block      NAND Flash Translation Layer filesystem
                0 = /dev/nftla        First NFTL layer
               16 = /dev/nftlb        Second NFTL layer
                  ...
              240 = /dev/nftlp        16th NTFL layer

94 char

94 block      IBM S/390 DASD block storage
                0 = /dev/dasda First DASD device, major
                1 = /dev/dasda1 First DASD device, block 1
                2 = /dev/dasda2 First DASD device, block 2
                3 = /dev/dasda3 First DASD device, block 3
                4 = /dev/dasdb Second DASD device, major
                5 = /dev/dasdb1 Second DASD device, block 1
                6 = /dev/dasdb2 Second DASD device, block 2
                7 = /dev/dasdb3 Second DASD device, block 3
                  ...

95 char       IP filter
                0 = /dev/ipl          Filter control device/log file
                1 = /dev/ipnat        NAT control device/log file
                2 = /dev/ipstate      State information log file
                3 = /dev/ipauth       Authentication control device/log file
                  ...

96 char       Parallel port ATAPI tape devices
                0 = /dev/pt0          First parallel port ATAPI tape
                1 = /dev/pt1          Second parallel port ATAPI tape
                  ...
```

```
                128 = /dev/npt0         First p.p. ATAPI tape, no rewind
                129 = /dev/npt1         Second p.p. ATAPI tape, no rewind
                    ...

96 block        Inverse NAND Flash Translation Layer
                  0 = /dev/inftla First INFTL layer
                 16 = /dev/inftlb Second INFTL layer
                    ...
                240 = /dev/inftlp       16th INTFL layer

97 char         Parallel port generic ATAPI interface
                  0 = /dev/pg0          First parallel port ATAPI device
                  1 = /dev/pg1          Second parallel port ATAPI device
                  2 = /dev/pg2          Third parallel port ATAPI device
                  3 = /dev/pg3          Fourth parallel port ATAPI device

                These devices support the same API as the generic SCSI
                devices.

98 char         Control and Measurement Device (comedi)
                  0 = /dev/comedi0      First comedi device
                  1 = /dev/comedi1      Second comedi device
                    ...
                 47 = /dev/comedi47     48th comedi device

                Minors 48 to 255 are reserved for comedi subdevices with
                pathnames of the form "/dev/comediX_subdY", where "X" is the
                minor number of the associated comedi device and "Y" is the
                subdevice number.  These subdevice minors are assigned
                dynamically, so there is no fixed mapping from subdevice
                pathnames to minor numbers.

                See https://www.comedi.org/ for information about the Comedi
                project.

98 block        User-mode virtual block device
                  0 = /dev/ubda         First user-mode block device
                 16 = /dev/ubdb         Second user-mode block device
                    ...

                Partitions are handled in the same way as for IDE
                disks (see major number 3) except that the limit on
                partitions is 15.

                This device is used by the user-mode virtual kernel port.

99 char         Raw parallel ports
                  0 = /dev/parport0     First parallel port
                  1 = /dev/parport1     Second parallel port
                    ...
```

```
 99 block      JavaStation flash disk
                 0 = /dev/jsfd         JavaStation flash disk

100 char       Telephony for Linux
                 0 = /dev/phone0       First telephony device
                 1 = /dev/phone1       Second telephony device
                   ...

101 char       Motorola DSP 56xxx board
                 0 = /dev/mdspstat     Status information
                 1 = /dev/mdsp1        First DSP board I/O controls
                   ...
                16 = /dev/mdsp16       16th DSP board I/O controls

101 block      AMI HyperDisk RAID controller
                 0 = /dev/amiraid/ar0  First array whole disk
                16 = /dev/amiraid/ar1  Second array whole disk
                   ...
               240 = /dev/amiraid/ar15 16th array whole disk

               For each device, partitions are added as:
                 0 = /dev/amiraid/ar?    Whole disk
                 1 = /dev/amiraid/ar?p1  First partition
                 2 = /dev/amiraid/ar?p2  Second partition
                   ...
                15 = /dev/amiraid/ar?p15 15th partition

102 char

102 block      Compressed block device
                 0 = /dev/cbd/a        First compressed block device, whole␣
↪device
                16 = /dev/cbd/b        Second compressed block device, whole␣
↪device
                   ...
               240 = /dev/cbd/p        16th compressed block device, whole␣
↪device

               Partitions are handled in the same way as for IDE
               disks (see major number 3) except that the limit on
               partitions is 15.

103 char       Arla network file system
                 0 = /dev/nnpfs0       First NNPFS device
                 1 = /dev/nnpfs1       Second NNPFS device

               Arla is a free clone of the Andrew File System, AFS.
               The NNPFS device gives user mode filesystem
               implementations a kernel presence for caching and easy
               mounting.  For more information about the project,
               write to <arla-drinkers@stacken.kth.se> or see
```

```
                https://www.stacken.kth.se/project/arla/

103 block       Audit device
                  0 = /dev/audit        Audit device

104 char        Flash BIOS support

104 block       Compaq Next Generation Drive Array, first controller
                  0 = /dev/cciss/c0d0   First logical drive, whole disk
                 16 = /dev/cciss/c0d1   Second logical drive, whole disk
                    ...
                240 = /dev/cciss/c0d15  16th logical drive, whole disk

                Partitions are handled the same way as for Mylex
                DAC960 (see major number 48) except that the limit on
                partitions is 15.

105 char        Comtrol VS-1000 serial controller
                  0 = /dev/ttyV0        First VS-1000 port
                  1 = /dev/ttyV1        Second VS-1000 port
                    ...

105 block       Compaq Next Generation Drive Array, second controller
                  0 = /dev/cciss/c1d0   First logical drive, whole disk
                 16 = /dev/cciss/c1d1   Second logical drive, whole disk
                    ...
                240 = /dev/cciss/c1d15  16th logical drive, whole disk

                Partitions are handled the same way as for Mylex
                DAC960 (see major number 48) except that the limit on
                partitions is 15.

106 char        Comtrol VS-1000 serial controller - alternate devices
                  0 = /dev/cuv0         First VS-1000 port
                  1 = /dev/cuv1         Second VS-1000 port
                    ...

106 block       Compaq Next Generation Drive Array, third controller
                  0 = /dev/cciss/c2d0   First logical drive, whole disk
                 16 = /dev/cciss/c2d1   Second logical drive, whole disk
                    ...
                240 = /dev/cciss/c2d15  16th logical drive, whole disk

                Partitions are handled the same way as for Mylex
                DAC960 (see major number 48) except that the limit on
                partitions is 15.

107 char        3Dfx Voodoo Graphics device
                  0 = /dev/3dfx         Primary 3Dfx graphics device

107 block       Compaq Next Generation Drive Array, fourth controller
```

```
                  0 = /dev/cciss/c3d0   First logical drive, whole disk
                 16 = /dev/cciss/c3d1   Second logical drive, whole disk
                    ...
                240 = /dev/cciss/c3d15  16th logical drive, whole disk

                Partitions are handled the same way as for Mylex
                DAC960 (see major number 48) except that the limit on
                partitions is 15.

108 char        Device independent PPP interface
                  0 = /dev/ppp          Device independent PPP interface

108 block       Compaq Next Generation Drive Array, fifth controller
                  0 = /dev/cciss/c4d0   First logical drive, whole disk
                 16 = /dev/cciss/c4d1   Second logical drive, whole disk
                    ...
                240 = /dev/cciss/c4d15  16th logical drive, whole disk

                Partitions are handled the same way as for Mylex
                DAC960 (see major number 48) except that the limit on
                partitions is 15.

109 char        Reserved for logical volume manager

109 block       Compaq Next Generation Drive Array, sixth controller
                  0 = /dev/cciss/c5d0   First logical drive, whole disk
                 16 = /dev/cciss/c5d1   Second logical drive, whole disk
                    ...
                240 = /dev/cciss/c5d15  16th logical drive, whole disk

                Partitions are handled the same way as for Mylex
                DAC960 (see major number 48) except that the limit on
                partitions is 15.

110 char        miroMEDIA Surround board
                  0 = /dev/srnd0        First miroMEDIA Surround board
                  1 = /dev/srnd1        Second miroMEDIA Surround board
                    ...

110 block       Compaq Next Generation Drive Array, seventh controller
                  0 = /dev/cciss/c6d0   First logical drive, whole disk
                 16 = /dev/cciss/c6d1   Second logical drive, whole disk
                    ...
                240 = /dev/cciss/c6d15  16th logical drive, whole disk

                Partitions are handled the same way as for Mylex
                DAC960 (see major number 48) except that the limit on
                partitions is 15.

111 char
```

```
111 block      Compaq Next Generation Drive Array, eighth controller
                 0 = /dev/cciss/c7d0   First logical drive, whole disk
                16 = /dev/cciss/c7d1   Second logical drive, whole disk
                   ...
               240 = /dev/cciss/c7d15  16th logical drive, whole disk

               Partitions are handled the same way as for Mylex
               DAC960 (see major number 48) except that the limit on
               partitions is 15.

112 char       ISI serial card
                 0 = /dev/ttyM0        First ISI port
                 1 = /dev/ttyM1        Second ISI port
                   ...

               There is currently a device-naming conflict between
               these and PAM multimodems (major 78).

112 block      IBM iSeries virtual disk
                 0 = /dev/iseries/vda  First virtual disk, whole disk
                 8 = /dev/iseries/vdb  Second virtual disk, whole disk
                   ...
               200 = /dev/iseries/vdz  26th virtual disk, whole disk
               208 = /dev/iseries/vdaa 27th virtual disk, whole disk
                   ...
               248 = /dev/iseries/vdaf 32nd virtual disk, whole disk

               Partitions are handled in the same way as for IDE
               disks (see major number 3) except that the limit on
               partitions is 7.

113 char       ISI serial card - alternate devices
                 0 = /dev/cum0         Callout device for ttyM0
                 1 = /dev/cum1         Callout device for ttyM1
                   ...

113 block      IBM iSeries virtual CD-ROM
                 0 = /dev/iseries/vcda First virtual CD-ROM
                 1 = /dev/iseries/vcdb Second virtual CD-ROM
                   ...

114 char       Picture Elements ISE board
                 0 = /dev/ise0         First ISE board
                 1 = /dev/ise1         Second ISE board
                   ...
               128 = /dev/isex0        Control node for first ISE board
               129 = /dev/isex1        Control node for second ISE board
                   ...

               The ISE board is an embedded computer, optimized for
               image processing. The /dev/iseN nodes are the general
```

```
                I/O access to the board, the /dev/isex0 nodes command
                nodes used to control the board.

114 block        IDE BIOS powered software RAID interfaces such as the
                Promise Fastrak

                   0 = /dev/ataraid/d0
                   1 = /dev/ataraid/d0p1
                   2 = /dev/ataraid/d0p2
                  ...
                  16 = /dev/ataraid/d1
                  17 = /dev/ataraid/d1p1
                  18 = /dev/ataraid/d1p2
                  ...
                 255 = /dev/ataraid/d15p15

                Partitions are handled in the same way as for IDE
                disks (see major number 3) except that the limit on
                partitions is 15.

115 char        TI link cable devices (115 was formerly the console driver␣
↪speaker)
                  0 = /dev/tipar0    Parallel cable on first parallel port
                  ...
                  7 = /dev/tipar7    Parallel cable on seventh parallel port

                  8 = /dev/tiser0    Serial cable on first serial port
                  ...
                 15 = /dev/tiser7    Serial cable on seventh serial port

                 16 = /dev/tiusb0    First USB cable
                  ...
                 47 = /dev/tiusb31   32nd USB cable

115 block        NetWare (NWFS) Devices (0-255)

                The NWFS (NetWare) devices are used to present a
                collection of NetWare Mirror Groups or NetWare
                Partitions as a logical storage segment for
                use in mounting NetWare volumes.  A maximum of
                 256 NetWare volumes can be supported in a single
                machine.

                http://cgfa.telepac.pt/ftp2/kernel.org/linux/kernel/people/
↪jmerkey/nwfs/

                 0 = /dev/nwfs/v0    First NetWare (NWFS) Logical Volume
                 1 = /dev/nwfs/v1    Second NetWare (NWFS) Logical Volume
                 2 = /dev/nwfs/v2    Third NetWare (NWFS) Logical Volume
                      ...
                 255 = /dev/nwfs/v255    Last NetWare (NWFS) Logical Volume
```

```
116 char       Advanced Linux Sound Driver (ALSA)

116 block       MicroMemory battery backed RAM adapter (NVRAM)
               Supports 16 boards, 15 partitions each.
               Requested by neilb at cse.unsw.edu.au.

                0 = /dev/umem/d0      Whole of first board
                1 = /dev/umem/d0p1    First partition of first board
                2 = /dev/umem/d0p2    Second partition of first board
               15 = /dev/umem/d0p15   15th partition of first board

               16 = /dev/umem/d1      Whole of second board
               17 = /dev/umem/d1p1    First partition of second board
                   ...
               255= /dev/umem/d15p15  15th partition of 16th board.

117 char       [REMOVED] COSA/SRP synchronous serial card
                 0 = /dev/cosa0c0      1st board, 1st channel
                 1 = /dev/cosa0c1      1st board, 2nd channel
                   ...
                16 = /dev/cosa1c0      2nd board, 1st channel
                17 = /dev/cosa1c1      2nd board, 2nd channel
                   ...

117 block       Enterprise Volume Management System (EVMS)

               The EVMS driver uses a layered, plug-in model to provide
               unparalleled flexibility and extensibility in managing
               storage.  This allows for easy expansion or customization
               of various levels of volume management.  Requested by
               Mark Peloquin (peloquin at us.ibm.com).

               Note: EVMS populates and manages all the devnodes in
               /dev/evms.

               http://sf.net/projects/evms

                  0 = /dev/evms/block_device   EVMS block device
                  1 = /dev/evms/legacyname1    First EVMS legacy device
                  2 = /dev/evms/legacyname2    Second EVMS legacy device
                   ...
                   Both ranges can grow (down or up) until they meet.
                   ...
                254 = /dev/evms/EVMSname2      Second EVMS native device
                255 = /dev/evms/EVMSname1      First EVMS native device

               Note: legacyname(s) are derived from the normal legacy
               device names.  For example, /dev/hda5 would become
               /dev/evms/hda5.
```

```
118 char       IBM Cryptographic Accelerator
                 0 = /dev/ica  Virtual interface to all IBM Crypto␣
↪Accelerators
                 1 = /dev/ica0 IBMCA Device 0
                 2 = /dev/ica1 IBMCA Device 1
                   ...

119 char       VMware virtual network control
                 0 = /dev/vnet0        1st virtual network
                 1 = /dev/vnet1        2nd virtual network
                   ...

120-127 char   LOCAL/EXPERIMENTAL USE

120-127 block  LOCAL/EXPERIMENTAL USE
               Allocated for local/experimental use.  For devices not
               assigned official numbers, these ranges should be
               used in order to avoid conflicting with future assignments.

128-135 char   Unix98 PTY masters

               These devices should not have corresponding device
               nodes; instead they should be accessed through the
               /dev/ptmx cloning interface.

128 block       SCSI disk devices (128-143)
                  0 = /dev/sddy         129th SCSI disk whole disk
                 16 = /dev/sddz         130th SCSI disk whole disk
                 32 = /dev/sdea         131th SCSI disk whole disk
                   ...
                240 = /dev/sden         144th SCSI disk whole disk

               Partitions are handled in the same way as for IDE
               disks (see major number 3) except that the limit on
               partitions is 15.

129 block       SCSI disk devices (144-159)
                  0 = /dev/sdeo         145th SCSI disk whole disk
                 16 = /dev/sdep         146th SCSI disk whole disk
                 32 = /dev/sdeq         147th SCSI disk whole disk
                   ...
                240 = /dev/sdfd         160th SCSI disk whole disk

               Partitions are handled in the same way as for IDE
               disks (see major number 3) except that the limit on
               partitions is 15.

130 char       (Misc devices)

130 block       SCSI disk devices (160-175)
                  0 = /dev/sdfe         161st SCSI disk whole disk
```

```
                 16 = /dev/sdff         162nd SCSI disk whole disk
                 32 = /dev/sdfg         163rd SCSI disk whole disk
                    ...
                240 = /dev/sdft         176th SCSI disk whole disk

               Partitions are handled in the same way as for IDE
               disks (see major number 3) except that the limit on
               partitions is 15.

131 block       SCSI disk devices (176-191)
                  0 = /dev/sdfu         177th SCSI disk whole disk
                 16 = /dev/sdfv         178th SCSI disk whole disk
                 32 = /dev/sdfw         179th SCSI disk whole disk
                    ...
                240 = /dev/sdgj         192nd SCSI disk whole disk

               Partitions are handled in the same way as for IDE
               disks (see major number 3) except that the limit on
               partitions is 15.

132 block       SCSI disk devices (192-207)
                  0 = /dev/sdgk         193rd SCSI disk whole disk
                 16 = /dev/sdgl         194th SCSI disk whole disk
                 32 = /dev/sdgm         195th SCSI disk whole disk
                    ...
                240 = /dev/sdgz         208th SCSI disk whole disk

               Partitions are handled in the same way as for IDE
               disks (see major number 3) except that the limit on
               partitions is 15.

133 block       SCSI disk devices (208-223)
                  0 = /dev/sdha         209th SCSI disk whole disk
                 16 = /dev/sdhb         210th SCSI disk whole disk
                 32 = /dev/sdhc         211th SCSI disk whole disk
                    ...
                240 = /dev/sdhp         224th SCSI disk whole disk

               Partitions are handled in the same way as for IDE
               disks (see major number 3) except that the limit on
               partitions is 15.

134 block       SCSI disk devices (224-239)
                  0 = /dev/sdhq         225th SCSI disk whole disk
                 16 = /dev/sdhr         226th SCSI disk whole disk
                 32 = /dev/sdhs         227th SCSI disk whole disk
                    ...
                240 = /dev/sdif         240th SCSI disk whole disk

               Partitions are handled in the same way as for IDE
               disks (see major number 3) except that the limit on
```

```
                partitions is 15.

135 block        SCSI disk devices (240-255)
                   0 = /dev/sdig         241st SCSI disk whole disk
                  16 = /dev/sdih         242nd SCSI disk whole disk
                  32 = /dev/sdih         243rd SCSI disk whole disk
                     ...
                 240 = /dev/sdiv         256th SCSI disk whole disk

                Partitions are handled in the same way as for IDE
                disks (see major number 3) except that the limit on
                partitions is 15.

136-143 char    Unix98 PTY slaves
                  0 = /dev/pts/0        First Unix98 pseudo-TTY
                  1 = /dev/pts/1        Second Unix98 pseudo-TTY
                     ...

                These device nodes are automatically generated with
                the proper permissions and modes by mounting the
                devpts filesystem onto /dev/pts with the appropriate
                mount options (distribution dependent, however, on
                *most* distributions the appropriate options are
                "mode=0620,gid=<gid of the "tty" group>".)

136 block       Mylex DAC960 PCI RAID controller; ninth controller
                  0 = /dev/rd/c8d0      First disk, whole disk
                  8 = /dev/rd/c8d1      Second disk, whole disk
                     ...
                248 = /dev/rd/c8d31     32nd disk, whole disk

                Partitions are handled as for major 48.

137 block       Mylex DAC960 PCI RAID controller; tenth controller
                  0 = /dev/rd/c9d0      First disk, whole disk
                  8 = /dev/rd/c9d1      Second disk, whole disk
                     ...
                248 = /dev/rd/c9d31     32nd disk, whole disk

                Partitions are handled as for major 48.

138 block       Mylex DAC960 PCI RAID controller; eleventh controller
                  0 = /dev/rd/c10d0     First disk, whole disk
                  8 = /dev/rd/c10d1     Second disk, whole disk
                     ...
                248 = /dev/rd/c10d31    32nd disk, whole disk

                Partitions are handled as for major 48.

139 block       Mylex DAC960 PCI RAID controller; twelfth controller
                  0 = /dev/rd/c11d0     First disk, whole disk
```

```
                  8 = /dev/rd/c11d1     Second disk, whole disk
                    ...
                248 = /dev/rd/c11d31    32nd disk, whole disk

                Partitions are handled as for major 48.

140 block       Mylex DAC960 PCI RAID controller; thirteenth controller
                  0 = /dev/rd/c12d0     First disk, whole disk
                  8 = /dev/rd/c12d1     Second disk, whole disk
                    ...
                248 = /dev/rd/c12d31    32nd disk, whole disk

                Partitions are handled as for major 48.

141 block       Mylex DAC960 PCI RAID controller; fourteenth controller
                  0 = /dev/rd/c13d0     First disk, whole disk
                  8 = /dev/rd/c13d1     Second disk, whole disk
                    ...
                248 = /dev/rd/c13d31    32nd disk, whole disk

                Partitions are handled as for major 48.

142 block       Mylex DAC960 PCI RAID controller; fifteenth controller
                  0 = /dev/rd/c14d0     First disk, whole disk
                  8 = /dev/rd/c14d1     Second disk, whole disk
                    ...
                248 = /dev/rd/c14d31    32nd disk, whole disk

                Partitions are handled as for major 48.

143 block       Mylex DAC960 PCI RAID controller; sixteenth controller
                  0 = /dev/rd/c15d0     First disk, whole disk
                  8 = /dev/rd/c15d1     Second disk, whole disk
                    ...
                248 = /dev/rd/c15d31    32nd disk, whole disk

                Partitions are handled as for major 48.

144 char        Encapsulated PPP
                  0 = /dev/pppox0       First PPP over Ethernet
                    ...
                 63 = /dev/pppox63      64th PPP over Ethernet

                This is primarily used for ADSL.

                The SST 5136-DN DeviceNet interface driver has been
                relocated to major 183 due to an unfortunate conflict.

144 block       Expansion Area #1 for more non-device (e.g. NFS) mounts
                  0 = mounted device 256
                255 = mounted device 511
```

```
145 char       SAM9407-based soundcard
                 0 = /dev/sam0_mixer
                 1 = /dev/sam0_sequencer
                 2 = /dev/sam0_midi00
                 3 = /dev/sam0_dsp
                 4 = /dev/sam0_audio
                 6 = /dev/sam0_sndstat
                18 = /dev/sam0_midi01
                34 = /dev/sam0_midi02
                50 = /dev/sam0_midi03
                64 = /dev/sam1_mixer
                   ...
               128 = /dev/sam2_mixer
                   ...
               192 = /dev/sam3_mixer
                   ...

               Device functions match OSS, but offer a number of
               addons, which are sam9407 specific.  OSS can be
               operated simultaneously, taking care of the codec.

145 block      Expansion Area #2 for more non-device (e.g. NFS) mounts
                 0 = mounted device 512
               255 = mounted device 767

146 char       SYSTRAM SCRAMNet mirrored-memory network
                 0 = /dev/scramnet0    First SCRAMNet device
                 1 = /dev/scramnet1    Second SCRAMNet device
                   ...

146 block      Expansion Area #3 for more non-device (e.g. NFS) mounts
                 0 = mounted device 768
               255 = mounted device 1023

147 char       Aureal Semiconductor Vortex Audio device
                 0 = /dev/aureal0      First Aureal Vortex
                 1 = /dev/aureal1      Second Aureal Vortex
                   ...

147 block      Distributed Replicated Block Device (DRBD)
                 0 = /dev/drbd0        First DRBD device
                 1 = /dev/drbd1        Second DRBD device
                   ...

148 char       Technology Concepts serial card
                 0 = /dev/ttyT0        First TCL port
                 1 = /dev/ttyT1        Second TCL port
                   ...

149 char       Technology Concepts serial card - alternate devices
```

```
                  0 = /dev/cut0         Callout device for ttyT0
                  1 = /dev/cut0         Callout device for ttyT1
                    ...

150 char        Real-Time Linux FIFOs
                  0 = /dev/rtf0         First RTLinux FIFO
                  1 = /dev/rtf1         Second RTLinux FIFO
                    ...

151 char        DPT I2O SmartRaid V controller
                  0 = /dev/dpti0        First DPT I2O adapter
                  1 = /dev/dpti1        Second DPT I2O adapter
                    ...

152 char        EtherDrive Control Device
                  0 = /dev/etherd/ctl   Connect/Disconnect an EtherDrive
                  1 = /dev/etherd/err   Monitor errors
                  2 = /dev/etherd/raw   Raw AoE packet monitor

152 block       EtherDrive Block Devices
                  0 = /dev/etherd/0     EtherDrive 0
                    ...
                255 = /dev/etherd/255   EtherDrive 255

153 char        SPI Bus Interface (sometimes referred to as MicroWire)
                  0 = /dev/spi0         First SPI device on the bus
                  1 = /dev/spi1         Second SPI device on the bus
                    ...
                 15 = /dev/spi15        Sixteenth SPI device on the bus

153 block       Enhanced Metadisk RAID (EMD) storage units
                  0 = /dev/emd/0        First unit
                  1 = /dev/emd/0p1      Partition 1 on First unit
                  2 = /dev/emd/0p2      Partition 2 on First unit
                    ...
                 15 = /dev/emd/0p15     Partition 15 on First unit

                 16 = /dev/emd/1        Second unit
                 32 = /dev/emd/2        Third unit
                    ...
                240 = /dev/emd/15       Sixteenth unit

                Partitions are handled in the same way as for IDE
                disks (see major number 3) except that the limit on
                partitions is 15.

154 char        Specialix RIO serial card
                  0 = /dev/ttySR0       First RIO port
                    ...
                255 = /dev/ttySR255     256th RIO port
```

```
155 char       Specialix RIO serial card - alternate devices
                 0 = /dev/cusr0        Callout device for ttySR0
                   ...
               255 = /dev/cusr255      Callout device for ttySR255

156 char       Specialix RIO serial card
                 0 = /dev/ttySR256     257th RIO port
                   ...
               255 = /dev/ttySR511     512th RIO port

157 char       Specialix RIO serial card - alternate devices
                 0 = /dev/cusr256      Callout device for ttySR256
                   ...
               255 = /dev/cusr511      Callout device for ttySR511

158 char       Dialogic GammaLink fax driver
                 0 = /dev/gfax0        GammaLink channel 0
                 1 = /dev/gfax1        GammaLink channel 1
                   ...

159 char       RESERVED

159 block      RESERVED

160 char       General Purpose Instrument Bus (GPIB)
                 0 = /dev/gpib0        First GPIB bus
                 1 = /dev/gpib1        Second GPIB bus
                   ...

160 block       Carmel 8-port SATA Disks on First Controller
                 0 = /dev/carmel/0     SATA disk 0 whole disk
                 1 = /dev/carmel/0p1   SATA disk 0 partition 1
                   ...
                31 = /dev/carmel/0p31  SATA disk 0 partition 31

                32 = /dev/carmel/1     SATA disk 1 whole disk
                64 = /dev/carmel/2     SATA disk 2 whole disk
                   ...
               224 = /dev/carmel/7     SATA disk 7 whole disk

               Partitions are handled in the same way as for IDE
               disks (see major number 3) except that the limit on
               partitions is 31.

161 char       IrCOMM devices (IrDA serial/parallel emulation)
                 0 = /dev/ircomm0      First IrCOMM device
                 1 = /dev/ircomm1      Second IrCOMM device
                   ...
                16 = /dev/irlpt0       First IrLPT device
                17 = /dev/irlpt1       Second IrLPT device
                   ...
```

```
161 block       Carmel 8-port SATA Disks on Second Controller
                 0 = /dev/carmel/8     SATA disk 8 whole disk
                 1 = /dev/carmel/8p1   SATA disk 8 partition 1
                    ...
                31 = /dev/carmel/8p31  SATA disk 8 partition 31

                32 = /dev/carmel/9     SATA disk 9 whole disk
                64 = /dev/carmel/10    SATA disk 10 whole disk
                    ...
               224 = /dev/carmel/15    SATA disk 15 whole disk

               Partitions are handled in the same way as for IDE
               disks (see major number 3) except that the limit on
               partitions is 31.

162 char       Used for (now removed) raw block device interface

163 char

164 char       Chase Research AT/PCI-Fast serial card
                 0 = /dev/ttyCH0       AT/PCI-Fast board 0, port 0
                    ...
                15 = /dev/ttyCH15      AT/PCI-Fast board 0, port 15
                16 = /dev/ttyCH16      AT/PCI-Fast board 1, port 0
                    ...
                31 = /dev/ttyCH31      AT/PCI-Fast board 1, port 15
                32 = /dev/ttyCH32      AT/PCI-Fast board 2, port 0
                    ...
                47 = /dev/ttyCH47      AT/PCI-Fast board 2, port 15
                48 = /dev/ttyCH48      AT/PCI-Fast board 3, port 0
                    ...
                63 = /dev/ttyCH63      AT/PCI-Fast board 3, port 15

165 char       Chase Research AT/PCI-Fast serial card - alternate devices
                 0 = /dev/cuch0        Callout device for ttyCH0
                    ...
                63 = /dev/cuch63       Callout device for ttyCH63

166 char       ACM USB modems
                 0 = /dev/ttyACM0      First ACM modem
                 1 = /dev/ttyACM1      Second ACM modem
                    ...

167 char       ACM USB modems - alternate devices
                 0 = /dev/cuacm0       Callout device for ttyACM0
                 1 = /dev/cuacm1       Callout device for ttyACM1
                    ...

168 char       Eracom CSA7000 PCI encryption adaptor
                 0 = /dev/ecsa0        First CSA7000
```

```
                  1 = /dev/ecsa1        Second CSA7000
                    ...

169 char        Eracom CSA8000 PCI encryption adaptor
                  0 = /dev/ecsa8-0      First CSA8000
                  1 = /dev/ecsa8-1      Second CSA8000
                    ...

170 char        AMI MegaRAC remote access controller
                  0 = /dev/megarac0     First MegaRAC card
                  1 = /dev/megarac1     Second MegaRAC card
                    ...

171 char        Reserved for IEEE 1394 (Firewire)

172 char        Moxa Intellio serial card
                  0 = /dev/ttyMX0       First Moxa port
                  1 = /dev/ttyMX1       Second Moxa port
                    ...
                127 = /dev/ttyMX127     128th Moxa port
                128 = /dev/moxactl      Moxa control port

173 char        Moxa Intellio serial card - alternate devices
                  0 = /dev/cumx0        Callout device for ttyMX0
                  1 = /dev/cumx1        Callout device for ttyMX1
                    ...
                127 = /dev/cumx127      Callout device for ttyMX127

174 char        SmartIO serial card
                  0 = /dev/ttySI0       First SmartIO port
                  1 = /dev/ttySI1       Second SmartIO port
                    ...

175 char        SmartIO serial card - alternate devices
                  0 = /dev/cusi0        Callout device for ttySI0
                  1 = /dev/cusi1        Callout device for ttySI1
                    ...

176 char        nCipher nFast PCI crypto accelerator
                  0 = /dev/nfastpci0    First nFast PCI device
                  1 = /dev/nfastpci1    First nFast PCI device
                    ...

177 char        TI PCILynx memory spaces
                  0 = /dev/pcilynx/aux0  AUX space of first PCILynx card
                    ...
                 15 = /dev/pcilynx/aux15 AUX space of 16th PCILynx card
                 16 = /dev/pcilynx/rom0  ROM space of first PCILynx card
                    ...
                 31 = /dev/pcilynx/rom15 ROM space of 16th PCILynx card
                 32 = /dev/pcilynx/ram0  RAM space of first PCILynx card
```

```
                  ...
                 47 = /dev/pcilynx/ram15 RAM space of 16th PCILynx card

178 char        Giganet cLAN1xxx virtual interface adapter
                  0 = /dev/clanvi0      First cLAN adapter
                  1 = /dev/clanvi1      Second cLAN adapter
                  ...

179 block        MMC block devices
                  0 = /dev/mmcblk0      First SD/MMC card
                  1 = /dev/mmcblk0p1    First partition on first MMC card
                  8 = /dev/mmcblk1      Second SD/MMC card
                  ...

                The start of next SD/MMC card can be configured with
                CONFIG_MMC_BLOCK_MINORS, or overridden at boot/modprobe
                time using the mmcblk.perdev_minors option. That would
                bump the offset between each card to be the configured
                value instead of the default 8.

179 char        CCube DVXChip-based PCI products
                  0 = /dev/dvxirq0      First DVX device
                  1 = /dev/dvxirq1      Second DVX device
                  ...

180 char        USB devices
                  0 = /dev/usb/lp0      First USB printer
                  ...
                 15 = /dev/usb/lp15     16th USB printer
                 48 = /dev/usb/scanner0 First USB scanner
                  ...
                 63 = /dev/usb/scanner15 16th USB scanner
                 64 = /dev/usb/rio500   Diamond Rio 500
                 65 = /dev/usb/usblcd   USBLCD Interface (info@usblcd.de)
                 66 = /dev/usb/cpad0    Synaptics cPad (mouse/LCD)
                 96 = /dev/usb/hiddev0  1st USB HID device
                  ...
                111 = /dev/usb/hiddev15 16th USB HID device
                112 = /dev/usb/auer0    1st auerswald ISDN device
                  ...
                127 = /dev/usb/auer15   16th auerswald ISDN device
                128 = /dev/usb/brlvgr0  First Braille Voyager device
                  ...
                131 = /dev/usb/brlvgr3  Fourth Braille Voyager device
                132 = /dev/usb/idmouse  ID Mouse (fingerprint scanner) device
                133 = /dev/usb/sisusbvga1       First SiSUSB VGA device
                  ...
                140 = /dev/usb/sisusbvga8       Eighth SISUSB VGA device
                144 = /dev/usb/lcd      USB LCD device
                160 = /dev/usb/legousbtower0    1st USB Legotower device
                  ...
```

```
                175 = /dev/usb/legousbtower15   16th USB Legotower device
                176 = /dev/usb/usbtmc1  First USB TMC device
                   ...
                191 = /dev/usb/usbtmc16 16th USB TMC device
                192 = /dev/usb/yurex1   First USB Yurex device
                   ...
                209 = /dev/usb/yurex16  16th USB Yurex device

180 block       USB block devices
                  0 = /dev/uba          First USB block device
                  8 = /dev/ubb          Second USB block device
                 16 = /dev/ubc          Third USB block device
                    ...

181 char        Conrad Electronic parallel port radio clocks
                  0 = /dev/pcfclock0    First Conrad radio clock
                  1 = /dev/pcfclock1    Second Conrad radio clock
                    ...

182 char        Picture Elements THR2 binarizer
                  0 = /dev/pethr0       First THR2 board
                  1 = /dev/pethr1       Second THR2 board
                    ...

183 char        SST 5136-DN DeviceNet interface
                  0 = /dev/ss5136dn0    First DeviceNet interface
                  1 = /dev/ss5136dn1    Second DeviceNet interface
                    ...

                This device used to be assigned to major number 144.
                It had to be moved due to an unfortunate conflict.

184 char        Picture Elements' video simulator/sender
                  0 = /dev/pevss0       First sender board
                  1 = /dev/pevss1       Second sender board
                    ...

185 char        InterMezzo high availability file system
                  0 = /dev/intermezzo0  First cache manager
                  1 = /dev/intermezzo1  Second cache manager
                    ...

                See http://web.archive.org/web/20080115195241/
                http://inter-mezzo.org/index.html

186 char        Object-based storage control device
                  0 = /dev/obd0         First obd control device
                  1 = /dev/obd1         Second obd control device
                    ...

                See ftp://ftp.lustre.org/pub/obd for code and information.
```

```
187 char       DESkey hardware encryption device
                 0 = /dev/deskey0      First DES key
                 1 = /dev/deskey1      Second DES key
                   ...

188 char       USB serial converters
                 0 = /dev/ttyUSB0      First USB serial converter
                 1 = /dev/ttyUSB1      Second USB serial converter
                   ...

189 char       USB serial converters - alternate devices
                 0 = /dev/cuusb0       Callout device for ttyUSB0
                 1 = /dev/cuusb1       Callout device for ttyUSB1
                   ...

190 char       Kansas City tracker/tuner card
                 0 = /dev/kctt0        First KCT/T card
                 1 = /dev/kctt1        Second KCT/T card
                   ...

191 char       Reserved for PCMCIA

192 char       Kernel profiling interface
                 0 = /dev/profile      Profiling control device
                 1 = /dev/profile0     Profiling device for CPU 0
                 2 = /dev/profile1     Profiling device for CPU 1
                   ...

193 char       Kernel event-tracing interface
                 0 = /dev/trace        Tracing control device
                 1 = /dev/trace0       Tracing device for CPU 0
                 2 = /dev/trace1       Tracing device for CPU 1
                   ...

194 char       linVideoStreams (LINVS)
                 0 = /dev/mvideo/status0       Video compression status
                 1 = /dev/mvideo/stream0       Video stream
                 2 = /dev/mvideo/frame0        Single compressed frame
                 3 = /dev/mvideo/rawframe0     Raw uncompressed frame
                 4 = /dev/mvideo/codec0        Direct codec access
                 5 = /dev/mvideo/video4linux0  Video4Linux compatibility

                16 = /dev/mvideo/status1       Second device
                   ...
                32 = /dev/mvideo/status2       Third device
                   ...
                   ...
               240 = /dev/mvideo/status15      16th device
                   ...
```

```
195 char       Nvidia graphics devices
                 0 = /dev/nvidia0              First Nvidia card
                 1 = /dev/nvidia1              Second Nvidia card
                   ...
               255 = /dev/nvidiactl            Nvidia card control device

196 char       Tormenta T1 card
                 0 = /dev/tor/0                Master control channel for all␣
↪cards
                 1 = /dev/tor/1                First DS0
                 2 = /dev/tor/2                Second DS0
                   ...
                48 = /dev/tor/48               48th DS0
                49 = /dev/tor/49               First pseudo-channel
                50 = /dev/tor/50               Second pseudo-channel
                   ...

197 char       OpenTNF tracing facility
                 0 = /dev/tnf/t0               Trace 0 data extraction
                 1 = /dev/tnf/t1               Trace 1 data extraction
                   ...
               128 = /dev/tnf/status           Tracing facility status
               130 = /dev/tnf/trace            Tracing device

198 char       Total Impact TPMP2 quad coprocessor PCI card
                 0 = /dev/tpmp2/0              First card
                 1 = /dev/tpmp2/1              Second card
                   ...

199 char       Veritas volume manager (VxVM) volumes
                 0 = /dev/vx/rdsk/*/*          First volume
                 1 = /dev/vx/rdsk/*/*          Second volume
                   ...

199 block      Veritas volume manager (VxVM) volumes
                 0 = /dev/vx/dsk/*/*           First volume
                 1 = /dev/vx/dsk/*/*           Second volume
                   ...

               The namespace in these directories is maintained by
               the user space VxVM software.

200 char       Veritas VxVM configuration interface
                  0 = /dev/vx/config           Configuration access node
                  1 = /dev/vx/trace            Volume i/o trace access node
                  2 = /dev/vx/iod              Volume i/o daemon access node
                  3 = /dev/vx/info             Volume information access node
                  4 = /dev/vx/task             Volume tasks access node
                  5 = /dev/vx/taskmon          Volume tasks monitor daemon

201 char       Veritas VxVM dynamic multipathing driver
```

```
                  0 = /dev/vx/rdmp/*            First multipath device
                  1 = /dev/vx/rdmp/*            Second multipath device
                    ...
201 block       Veritas VxVM dynamic multipathing driver
                  0 = /dev/vx/dmp/*             First multipath device
                  1 = /dev/vx/dmp/*             Second multipath device
                    ...

                The namespace in these directories is maintained by
                the user space VxVM software.

202 char        CPU model-specific registers
                  0 = /dev/cpu/0/msr            MSRs on CPU 0
                  1 = /dev/cpu/1/msr            MSRs on CPU 1
                    ...

202 block       Xen Virtual Block Device
                  0 = /dev/xvda       First Xen VBD whole disk
                  16 = /dev/xvdb      Second Xen VBD whole disk
                  32 = /dev/xvdc      Third Xen VBD whole disk
                    ...
                  240 = /dev/xvdp     Sixteenth Xen VBD whole disk

                Partitions are handled in the same way as for IDE
                disks (see major number 3) except that the limit on
                partitions is 15.

203 char        CPU CPUID information
                  0 = /dev/cpu/0/cpuid          CPUID on CPU 0
                  1 = /dev/cpu/1/cpuid          CPUID on CPU 1
                    ...

204 char        Low-density serial ports
                  0 = /dev/ttyLU0               LinkUp Systems L72xx UART -␣
↪port 0
                  1 = /dev/ttyLU1               LinkUp Systems L72xx UART -␣
↪port 1
                  2 = /dev/ttyLU2               LinkUp Systems L72xx UART -␣
↪port 2
                  3 = /dev/ttyLU3               LinkUp Systems L72xx UART -␣
↪port 3
                  4 = /dev/ttyFB0               Intel Footbridge (ARM)
                  5 = /dev/ttySA0               StrongARM builtin serial port 0
                  6 = /dev/ttySA1               StrongARM builtin serial port 1
                  7 = /dev/ttySA2               StrongARM builtin serial port 2
                  8 = /dev/ttySC0               SCI serial port (SuperH) -␣
↪port 0
                  9 = /dev/ttySC1               SCI serial port (SuperH) -␣
↪port 1
                 10 = /dev/ttySC2               SCI serial port (SuperH) -␣
↪port 2
```

```
                 11 = /dev/ttySC3               SCI serial port (SuperH) -␣
↪port 3
                 12 = /dev/ttyFW0               Firmware console - port 0
                 13 = /dev/ttyFW1               Firmware console - port 1
                 14 = /dev/ttyFW2               Firmware console - port 2
                 15 = /dev/ttyFW3               Firmware console - port 3
                 16 = /dev/ttyAM0               ARM "AMBA" serial port 0
                    ...
                 31 = /dev/ttyAM15              ARM "AMBA" serial port 15
                 32 = /dev/ttyDB0               DataBooster serial port 0
                    ...
                 39 = /dev/ttyDB7               DataBooster serial port 7
                 40 = /dev/ttySG0               SGI Altix console port
                 41 = /dev/ttySMX0              Motorola i.MX - port 0
                 42 = /dev/ttySMX1              Motorola i.MX - port 1
                 43 = /dev/ttySMX2              Motorola i.MX - port 2
                 44 = /dev/ttyMM0               Marvell MPSC - port 0␣
↪(obsolete unused)
                 45 = /dev/ttyMM1               Marvell MPSC - port 1␣
↪(obsolete unused)
                 46 = /dev/ttyCPM0              PPC CPM (SCC or SMC) - port 0
                    ...
                 51 = /dev/ttyCPM5              PPC CPM (SCC or SMC) - port 5
                 82 = /dev/ttyVR0               NEC VR4100 series SIU
                 83 = /dev/ttyVR1               NEC VR4100 series DSIU
                 148 = /dev/ttyPSC0             PPC PSC - port 0
                    ...
                 153 = /dev/ttyPSC5             PPC PSC - port 5
                 154 = /dev/ttyAT0              ATMEL serial port 0
                    ...
                 169 = /dev/ttyAT15             ATMEL serial port 15
                 170 = /dev/ttyNX0              Hilscher netX serial port 0
                    ...
                 185 = /dev/ttyNX15             Hilscher netX serial port 15
                 186 = /dev/ttyJ0               JTAG1 DCC protocol based␣
↪serial port emulation

                 If maximum number of uartlite serial ports is more than 4,␣
↪then the driver
                 uses dynamic allocation instead of static allocation for␣
↪major number.
                 187 = /dev/ttyUL0              Xilinx uartlite - port 0
                    ...
                 190 = /dev/ttyUL3              Xilinx uartlite - port 3
                 191 = /dev/xvc0                Xen virtual console - port 0
                 192 = /dev/ttyPZ0              pmac_zilog - port 0
                    ...
                 195 = /dev/ttyPZ3              pmac_zilog - port 3
                 196 = /dev/ttyTX0              TX39/49 serial port 0
                    ...
                 204 = /dev/ttyTX7              TX39/49 serial port 7
```

```
                 205 = /dev/ttySC0              SC26xx serial port 0
                 206 = /dev/ttySC1              SC26xx serial port 1
                 207 = /dev/ttySC2              SC26xx serial port 2
                 208 = /dev/ttySC3              SC26xx serial port 3
                 209 = /dev/ttyMAX0             MAX3100 serial port 0
                 210 = /dev/ttyMAX1             MAX3100 serial port 1
                 211 = /dev/ttyMAX2             MAX3100 serial port 2
                 212 = /dev/ttyMAX3             MAX3100 serial port 3

205 char        Low-density serial ports (alternate device)
                  0 = /dev/culu0                Callout device for ttyLU0
                  1 = /dev/culu1                Callout device for ttyLU1
                  2 = /dev/culu2                Callout device for ttyLU2
                  3 = /dev/culu3                Callout device for ttyLU3
                  4 = /dev/cufb0                Callout device for ttyFB0
                  5 = /dev/cusa0                Callout device for ttySA0
                  6 = /dev/cusa1                Callout device for ttySA1
                  7 = /dev/cusa2                Callout device for ttySA2
                  8 = /dev/cusc0                Callout device for ttySC0
                  9 = /dev/cusc1                Callout device for ttySC1
                 10 = /dev/cusc2                Callout device for ttySC2
                 11 = /dev/cusc3                Callout device for ttySC3
                 12 = /dev/cufw0                Callout device for ttyFW0
                 13 = /dev/cufw1                Callout device for ttyFW1
                 14 = /dev/cufw2                Callout device for ttyFW2
                 15 = /dev/cufw3                Callout device for ttyFW3
                 16 = /dev/cuam0                Callout device for ttyAM0
                    ...
                 31 = /dev/cuam15               Callout device for ttyAM15
                 32 = /dev/cudb0                Callout device for ttyDB0
                    ...
                 39 = /dev/cudb7                Callout device for ttyDB7
                 40 = /dev/cusg0                Callout device for ttySG0
                 41 = /dev/ttycusmx0            Callout device for ttySMX0
                 42 = /dev/ttycusmx1            Callout device for ttySMX1
                 43 = /dev/ttycusmx2            Callout device for ttySMX2
                 46 = /dev/cucpm0               Callout device for ttyCPM0
                    ...
                 51 = /dev/cucpm5               Callout device for ttyCPM5
                 82 = /dev/cuvr0                Callout device for ttyVR0
                 83 = /dev/cuvr1                Callout device for ttyVR1

206 char        OnStream SC-x0 tape devices
                  0 = /dev/osst0                First OnStream SCSI tape, mode␣
↪0
                  1 = /dev/osst1                Second OnStream SCSI tape,␣
↪mode 0
                    ...
                 32 = /dev/osst0l               First OnStream SCSI tape, mode␣
↪1
                 33 = /dev/osst1l               Second OnStream SCSI tape,␣
```

```
↪mode 1
                    ...
                 64 = /dev/osst0m               First OnStream SCSI tape, mode␣
↪2
                 65 = /dev/osst1m               Second OnStream SCSI tape,␣
↪mode 2
                    ...
                 96 = /dev/osst0a               First OnStream SCSI tape, mode␣
↪3
                 97 = /dev/osst1a               Second OnStream SCSI tape,␣
↪mode 3
                    ...
                128 = /dev/nosst0               No rewind version of /dev/osst0
                129 = /dev/nosst1               No rewind version of /dev/osst1
                    ...
                160 = /dev/nosst0l              No rewind version of /dev/
↪osst0l
                161 = /dev/nosst1l              No rewind version of /dev/
↪osst1l
                    ...
                192 = /dev/nosst0m              No rewind version of /dev/
↪osst0m
                193 = /dev/nosst1m              No rewind version of /dev/
↪osst1m
                    ...
                224 = /dev/nosst0a              No rewind version of /dev/
↪osst0a
                225 = /dev/nosst1a              No rewind version of /dev/
↪osst1a
                    ...

                The OnStream SC-x0 SCSI tapes do not support the
                standard SCSI SASD command set and therefore need
                their own driver "osst". Note that the IDE, USB (and
                maybe ParPort) versions may be driven via ide-scsi or
                usb-storage SCSI emulation and this osst device and
                driver as well.  The ADR-x0 drives are QIC-157
                compliant and don't need osst.

207 char        Compaq ProLiant health feature indicate
                  0 = /dev/cpqhealth/cpqw       Redirector interface
                  1 = /dev/cpqhealth/crom       EISA CROM
                  2 = /dev/cpqhealth/cdt        Data Table
                  3 = /dev/cpqhealth/cevt       Event Log
                  4 = /dev/cpqhealth/casr       Automatic Server Recovery
                  5 = /dev/cpqhealth/cecc       ECC Memory
                  6 = /dev/cpqhealth/cmca       Machine Check Architecture
                  7 = /dev/cpqhealth/ccsm       Deprecated CDT
                  8 = /dev/cpqhealth/cnmi       NMI Handling
                  9 = /dev/cpqhealth/css        Sideshow Management
                 10 = /dev/cpqhealth/cram       CMOS interface
```

```
                 11 = /dev/cpqhealth/cpci       PCI IRQ interface

208 char        User space serial ports
                  0 = /dev/ttyU0                First user space serial port
                  1 = /dev/ttyU1                Second user space serial port
                    ...

209 char        User space serial ports (alternate devices)
                  0 = /dev/cuu0                 Callout device for ttyU0
                  1 = /dev/cuu1                 Callout device for ttyU1
                    ...

210 char        SBE, Inc. sync/async serial card
                  0 = /dev/sbei/wxcfg0          Configuration device for board␣
↪0
                  1 = /dev/sbei/dld0            Download device for board 0
                  2 = /dev/sbei/wan00           WAN device, port 0, board 0
                  3 = /dev/sbei/wan01           WAN device, port 1, board 0
                  4 = /dev/sbei/wan02           WAN device, port 2, board 0
                  5 = /dev/sbei/wan03           WAN device, port 3, board 0
                  6 = /dev/sbei/wanc00          WAN clone device, port 0,␣
↪board 0
                  7 = /dev/sbei/wanc01          WAN clone device, port 1,␣
↪board 0
                  8 = /dev/sbei/wanc02          WAN clone device, port 2,␣
↪board 0
                  9 = /dev/sbei/wanc03          WAN clone device, port 3,␣
↪board 0
                 10 = /dev/sbei/wxcfg1          Configuration device for board␣
↪1
                 11 = /dev/sbei/dld1            Download device for board 1
                 12 = /dev/sbei/wan10           WAN device, port 0, board 1
                 13 = /dev/sbei/wan11           WAN device, port 1, board 1
                 14 = /dev/sbei/wan12           WAN device, port 2, board 1
                 15 = /dev/sbei/wan13           WAN device, port 3, board 1
                 16 = /dev/sbei/wanc10          WAN clone device, port 0,␣
↪board 1
                 17 = /dev/sbei/wanc11          WAN clone device, port 1,␣
↪board 1
                 18 = /dev/sbei/wanc12          WAN clone device, port 2,␣
↪board 1
                 19 = /dev/sbei/wanc13          WAN clone device, port 3,␣
↪board 1
                    ...

                Yes, each board is really spaced 10 (decimal) apart.

211 char        Addinum CPCI1500 digital I/O card
                  0 = /dev/addinum/cpci1500/0   First CPCI1500 card
                  1 = /dev/addinum/cpci1500/1   Second CPCI1500 card
                    ...
```

```
212 char       LinuxTV.org DVB driver subsystem
                 0 = /dev/dvb/adapter0/video0    first video decoder of first␣
↪card
                 1 = /dev/dvb/adapter0/audio0    first audio decoder of first␣
↪card
                 2 = /dev/dvb/adapter0/sec0      (obsolete/unused)
                 3 = /dev/dvb/adapter0/frontend0 first frontend device of␣
↪first card
                 4 = /dev/dvb/adapter0/demux0    first demux device of first␣
↪card
                 5 = /dev/dvb/adapter0/dvr0      first digital video recoder␣
↪device of first card
                 6 = /dev/dvb/adapter0/ca0       first common access port of␣
↪first card
                 7 = /dev/dvb/adapter0/net0      first network device of␣
↪first card
                 8 = /dev/dvb/adapter0/osd0      first on-screen-display␣
↪device of first card
                 9 = /dev/dvb/adapter0/video1    second video decoder of␣
↪first card
                   ...
                64 = /dev/dvb/adapter1/video0    first video decoder of␣
↪second card
                   ...
               128 = /dev/dvb/adapter2/video0    first video decoder of third␣
↪card
                   ...
               196 = /dev/dvb/adapter3/video0    first video decoder of␣
↪fourth card

216 char       Bluetooth RFCOMM TTY devices
                 0 = /dev/rfcomm0              First Bluetooth RFCOMM TTY␣
↪device
                 1 = /dev/rfcomm1              Second Bluetooth RFCOMM TTY␣
↪device
                   ...

217 char       Bluetooth RFCOMM TTY devices (alternate devices)
                 0 = /dev/curf0                Callout device for rfcomm0
                 1 = /dev/curf1                Callout device for rfcomm1
                   ...

218 char       The Logical Company bus Unibus/Qbus adapters
                 0 = /dev/logicalco/bci/0      First bus adapter
                 1 = /dev/logicalco/bci/1      First bus adapter
                   ...

219 char       The Logical Company DCI-1300 digital I/O card
                 0 = /dev/logicalco/dci1300/0  First DCI-1300 card
                 1 = /dev/logicalco/dci1300/1  Second DCI-1300 card
```

```
                  ...

220 char       Myricom Myrinet "GM" board
                 0 = /dev/myricom/gm0          First Myrinet GM board
                 1 = /dev/myricom/gmp0         First board "root access"
                 2 = /dev/myricom/gm1          Second Myrinet GM board
                 3 = /dev/myricom/gmp1         Second board "root access"
                   ...

221 char       VME bus
                 0 = /dev/bus/vme/m0           First master image
                 1 = /dev/bus/vme/m1           Second master image
                 2 = /dev/bus/vme/m2           Third master image
                 3 = /dev/bus/vme/m3           Fourth master image
                 4 = /dev/bus/vme/s0           First slave image
                 5 = /dev/bus/vme/s1           Second slave image
                 6 = /dev/bus/vme/s2           Third slave image
                 7 = /dev/bus/vme/s3           Fourth slave image
                 8 = /dev/bus/vme/ctl          Control

               It is expected that all VME bus drivers will use the
               same interface.  For interface documentation see
               http://www.vmelinux.org/.

224 char       A2232 serial card
                 0 = /dev/ttyY0                First A2232 port
                 1 = /dev/ttyY1                Second A2232 port
                   ...

225 char       A2232 serial card (alternate devices)
                 0 = /dev/cuy0                 Callout device for ttyY0
                 1 = /dev/cuy1                 Callout device for ttyY1
                   ...

226 char       Direct Rendering Infrastructure (DRI)
                 0 = /dev/dri/card0            First graphics card
                 1 = /dev/dri/card1            Second graphics card
                   ...

227 char       IBM 3270 terminal Unix tty access
                 1 = /dev/3270/tty1            First 3270 terminal
                 2 = /dev/3270/tty2            Seconds 3270 terminal
                   ...

228 char       IBM 3270 terminal block-mode access
                 0 = /dev/3270/tub             Controlling interface
                 1 = /dev/3270/tub1            First 3270 terminal
                 2 = /dev/3270/tub2            Second 3270 terminal
                   ...

229 char       IBM iSeries/pSeries virtual console
```

```
                  0 = /dev/hvc0                  First console port
                  1 = /dev/hvc1                  Second console port
                    ...

230 char        IBM iSeries virtual tape
                  0 = /dev/iseries/vt0           First virtual tape, mode 0
                  1 = /dev/iseries/vt1           Second virtual tape, mode 0
                    ...
                 32 = /dev/iseries/vt0l          First virtual tape, mode 1
                 33 = /dev/iseries/vt1l          Second virtual tape, mode 1
                    ...
                 64 = /dev/iseries/vt0m          First virtual tape, mode 2
                 65 = /dev/iseries/vt1m          Second virtual tape, mode 2
                    ...
                 96 = /dev/iseries/vt0a          First virtual tape, mode 3
                 97 = /dev/iseries/vt1a          Second virtual tape, mode 3
                      ...
                128 = /dev/iseries/nvt0          First virtual tape, mode 0, no␣
↪rewind
                129 = /dev/iseries/nvt1          Second virtual tape, mode 0,␣
↪no rewind
                    ...
                160 = /dev/iseries/nvt0l         First virtual tape, mode 1, no␣
↪rewind
                161 = /dev/iseries/nvt1l         Second virtual tape, mode 1,␣
↪no rewind
                    ...
                192 = /dev/iseries/nvt0m         First virtual tape, mode 2, no␣
↪rewind
                193 = /dev/iseries/nvt1m         Second virtual tape, mode 2,␣
↪no rewind
                    ...
                224 = /dev/iseries/nvt0a         First virtual tape, mode 3, no␣
↪rewind
                225 = /dev/iseries/nvt1a         Second virtual tape, mode 3,␣
↪no rewind
                    ...

                "No rewind" refers to the omission of the default
                automatic rewind on device close.  The MTREW or MTOFFL
                ioctl()'s can be used to rewind the tape regardless of
                the device used to access it.

231 char        InfiniBand
                0 = /dev/infiniband/umad0
                1 = /dev/infiniband/umad1
                  ...
                63 = /dev/infiniband/umad63    63rd InfiniBandMad device
                64 = /dev/infiniband/issm0     First InfiniBand IsSM device
                65 = /dev/infiniband/issm1     Second InfiniBand IsSM device
                  ...
```

```
                127 = /dev/infiniband/issm63    63rd InfiniBand IsSM device
                192 = /dev/infiniband/uverbs0   First InfiniBand verbs device
                193 = /dev/infiniband/uverbs1   Second InfiniBand verbs device
                  ...
                223 = /dev/infiniband/uverbs31  31st InfiniBand verbs device

232 char        Biometric Devices
                0 = /dev/biometric/sensor0/fingerprint  first fingerprint␣
↪sensor on first device
                1 = /dev/biometric/sensor0/iris         first iris sensor on␣
↪first device
                2 = /dev/biometric/sensor0/retina       first retina sensor on␣
↪first device
                3 = /dev/biometric/sensor0/voiceprint   first voiceprint␣
↪sensor on first device
                4 = /dev/biometric/sensor0/facial       first facial sensor on␣
↪first device
                5 = /dev/biometric/sensor0/hand         first hand sensor on␣
↪first device
                  ...
                10 = /dev/biometric/sensor1/fingerprint first fingerprint␣
↪sensor on second device
                  ...
                20 = /dev/biometric/sensor2/fingerprint first fingerprint␣
↪sensor on third device
                  ...

233 char        PathScale InfiniPath interconnect
                0 = /dev/ipath        Primary device for programs (any unit)
                1 = /dev/ipath0       Access specifically to unit 0
                2 = /dev/ipath1       Access specifically to unit 1
                  ...
                4 = /dev/ipath3       Access specifically to unit 3
                129 = /dev/ipath_sma    Device used by Subnet Management Agent
                130 = /dev/ipath_diag   Device used by diagnostics programs

234-254         char    RESERVED FOR DYNAMIC ASSIGNMENT
                Character devices that request a dynamic allocation of major␣
↪number will
                take numbers starting from 254 and downward.

240-254 block   LOCAL/EXPERIMENTAL USE
                Allocated for local/experimental use.  For devices not
                assigned official numbers, these ranges should be
                used in order to avoid conflicting with future assignments.

255 char        RESERVED

255 block       RESERVED

                This major is reserved to assist the expansion to a
```

```
                larger number space.  No device nodes with this major
                should ever be created on the filesystem.
                (This is probably not true anymore, but I'll leave it
                for now /Torben)

---LARGE MAJORS!!!!!---

256 char        Equinox SST multi-port serial boards
                   0 = /dev/ttyEQ0      First serial port on first Equinox SST␣
↪board
                 127 = /dev/ttyEQ127    Last serial port on first Equinox SST␣
↪board
                 128 = /dev/ttyEQ128    First serial port on second Equinox␣
↪SST board
                  ...
                1027 = /dev/ttyEQ1027   Last serial port on eighth Equinox SST␣
↪board

256 block       Resident Flash Disk Flash Translation Layer
                  0 = /dev/rfda         First RFD FTL layer
                 16 = /dev/rfdb         Second RFD FTL layer
                  ...
                240 = /dev/rfdp         16th RFD FTL layer

257 char        Phoenix Technologies Cryptographic Services Driver
                  0 = /dev/ptlsec       Crypto Services Driver

257 block       SSFDC Flash Translation Layer filesystem
                  0 = /dev/ssfdca       First SSFDC layer
                  8 = /dev/ssfdcb       Second SSFDC layer
                 16 = /dev/ssfdcc       Third SSFDC layer
                 24 = /dev/ssfdcd       4th SSFDC layer
                 32 = /dev/ssfdce       5th SSFDC layer
                 40 = /dev/ssfdcf       6th SSFDC layer
                 48 = /dev/ssfdcg       7th SSFDC layer
                 56 = /dev/ssfdch       8th SSFDC layer

258 block       ROM/Flash read-only translation layer
                  0 = /dev/blockrom0    First ROM card's translation layer␣
↪interface
                  1 = /dev/blockrom1    Second ROM card's translation layer␣
↪interface
                  ...

259 block       Block Extended Major
                  Used dynamically to hold additional partition minor
                  numbers and allow large numbers of partitions per device

259 char        FPGA configuration interfaces
                  0 = /dev/icap0        First Xilinx internal configuration
                  1 = /dev/icap1        Second Xilinx internal configuration
```

```
260 char       OSD (Object-based-device) SCSI Device
                 0 = /dev/osd0         First OSD Device
                 1 = /dev/osd1         Second OSD Device
                 ...
                 255 = /dev/osd255     256th OSD Device

261 char       Compute Acceleration Devices
                 0 = /dev/accel/accel0 First acceleration device
                 1 = /dev/accel/accel1 Second acceleration device
                   ...

384-511 char   RESERVED FOR DYNAMIC ASSIGNMENT
               Character devices that request a dynamic allocation of major
               number will take numbers starting from 511 and downward,
               once the 234-254 range is full.
```

3.1 Additional /dev/ directory entries

This section details additional entries that should or may exist in the /dev directory. It is preferred that symbolic links use the same form (absolute or relative) as is indicated here. Links are classified as ”hard” or ”symbolic” depending on the preferred type of link; if possible, the indicated type of link should be used.

3.1.1 Compulsory links

These links should exist on all systems:

/dev/fd	/proc/self/fd	symbolic	File descriptors
/dev/stdin	fd/0	symbolic	stdin file descriptor
/dev/stdout	fd/1	symbolic	stdout file descriptor
/dev/stderr	fd/2	symbolic	stderr file descriptor
/dev/nfsd	socksys	symbolic	Required by iBCS-2
/dev/X0R	null	symbolic	Required by iBCS-2

Note: `/dev/X0R` is <letter X>-<digit 0>-<letter R>.

3.1.2 Recommended links

It is recommended that these links exist on all systems:

/dev/core	/proc/kcore	symbolic	Backward compatibility
/dev/ramdisk	ram0	symbolic	Backward compatibility
/dev/ftape	qft0	symbolic	Backward compatibility
/dev/bttv0	video0	symbolic	Backward compatibility
/dev/radio	radio0	symbolic	Backward compatibility
/dev/i2o*	/dev/i2o/*	symbolic	Backward compatibility
/dev/scd?	sr?	hard	Alternate SCSI CD-ROM name

3.1.3 Locally defined links

The following links may be established locally to conform to the configuration of the system. This is merely a tabulation of existing practice, and does not constitute a recommendation. However, if they exist, they should have the following uses.

/dev/mouse	mouse port	symbolic	Current mouse device
/dev/tape	tape device	symbolic	Current tape device
/dev/cdrom	CD-ROM device	symbolic	Current CD-ROM device
/dev/cdwriter	CD-writer	symbolic	Current CD-writer device
/dev/scanner	scanner	symbolic	Current scanner device
/dev/modem	modem port	symbolic	Current dialout device
/dev/root	root device	symbolic	Current root filesystem
/dev/swap	swap device	symbolic	Current swap device

`/dev/modem` should not be used for a modem which supports dialin as well as dialout, as it tends to cause lock file problems. If it exists, `/dev/modem` should point to the appropriate primary TTY device (the use of the alternate callout devices is deprecated).

For SCSI devices, `/dev/tape` and `/dev/cdrom` should point to the *cooked* devices (`/dev/st*` and `/dev/sr*`, respectively), whereas `/dev/cdwriter` and /dev/scanner should point to the appropriate generic SCSI devices (/dev/sg*).

`/dev/mouse` may point to a primary serial TTY device, a hardware mouse device, or a socket for a mouse driver program (e.g. `/dev/gpmdata`).

3.1.4 Sockets and pipes

Non-transient sockets and named pipes may exist in /dev. Common entries are:

/dev/printer	socket	lpd local socket
/dev/log	socket	syslog local socket
/dev/gpmdata	socket	gpm mouse multiplexer

3.1.5 Mount points

The following names are reserved for mounting special filesystems under /dev. These special filesystems provide kernel interfaces that cannot be provided with standard device nodes.

/dev/pts	devpts	PTY slave filesystem
/dev/shm	tmpfs	POSIX shared memory maintenance access

3.2 Terminal devices

Terminal, or TTY devices are a special class of character devices. A terminal device is any device that could act as a controlling terminal for a session; this includes virtual consoles, serial ports, and pseudoterminals (PTYs).

All terminal devices share a common set of capabilities known as line disciplines; these include the common terminal line discipline as well as SLIP and PPP modes.

All terminal devices are named similarly; this section explains the naming and use of the various types of TTYs. Note that the naming conventions include several historical warts; some of these are Linux-specific, some were inherited from other systems, and some reflect Linux outgrowing a borrowed convention.

A hash mark (#) in a device name is used here to indicate a decimal number without leading zeroes.

3.2.1 Virtual consoles and the console device

Virtual consoles are full-screen terminal displays on the system video monitor. Virtual consoles are named `/dev/tty#`, with numbering starting at `/dev/tty1`; `/dev/tty0` is the current virtual console. `/dev/tty0` is the device that should be used to access the system video card on those architectures for which the frame buffer devices (`/dev/fb*`) are not applicable. Do not use `/dev/console` for this purpose.

The console device, `/dev/console`, is the device to which system messages should be sent, and on which logins should be permitted in single-user mode. Starting with Linux 2.1.71, `/dev/console` is managed by the kernel; for previous versions it should be a symbolic link to either `/dev/tty0`, a specific virtual console such as `/dev/tty1`, or to a serial port primary (`tty*`, not `cu*`) device, depending on the configuration of the system.

3.2.2 Serial ports

Serial ports are RS-232 serial ports and any device which simulates one, either in hardware (such as internal modems) or in software (such as the ISDN driver.) Under Linux, each serial ports has two device names, the primary or callin device and the alternate or callout one. Each kind of device is indicated by a different letter. For any letter X, the names of the devices are `/dev/ttyX#` and `/dev/cux#`, respectively; for historical reasons, `/dev/ttyS#` and `/dev/ttyC#` correspond to `/dev/cua#` and `/dev/cub#`. In the future, it should be expected that multiple letters will be used; all letters will be upper case for the ”tty” device (e.g. `/dev/ttyDP#`) and lower case for the ”cu” device (e.g. `/dev/cudp#`).

The names /dev/ttyQ# and /dev/cuq# are reserved for local use.

The alternate devices provide for kernel-based exclusion and somewhat different defaults than the primary devices. Their main purpose is to allow the use of serial ports with programs with no inherent or broken support for serial ports. Their use is deprecated, and they may be removed from a future version of Linux.

Arbitration of serial ports is provided by the use of lock files with the names /var/lock/LCK..ttyX#. The contents of the lock file should be the PID of the locking process as an ASCII number.

It is common practice to install links such as /dev/modem which point to serial ports. In order to ensure proper locking in the presence of these links, it is recommended that software chase symlinks and lock all possible names; additionally, it is recommended that a lock file be installed with the corresponding alternate device. In order to avoid deadlocks, it is recommended that the locks are acquired in the following order, and released in the reverse:

1. The symbolic link name, if any (/var/lock/LCK..modem)
2. The "tty" name (/var/lock/LCK..ttyS2)
3. The alternate device name (/var/lock/LCK..cua2)

In the case of nested symbolic links, the lock files should be installed in the order the symlinks are resolved.

Under no circumstances should an application hold a lock while waiting for another to be released. In addition, applications which attempt to create lock files for the corresponding alternate device names should take into account the possibility of being used on a non-serial port TTY, for which no alternate device would exist.

3.2.3 Pseudoterminals (PTYs)

Pseudoterminals, or PTYs, are used to create login sessions or provide other capabilities requiring a TTY line discipline (including SLIP or PPP capability) to arbitrary data-generation processes. Each PTY has a master side, named /dev/pty[p-za-e][0-9a-f], and a slave side, named /dev/tty[p-za-e][0-9a-f]. The kernel arbitrates the use of PTYs by allowing each master side to be opened only once.

Once the master side has been opened, the corresponding slave device can be used in the same manner as any TTY device. The master and slave devices are connected by the kernel, generating the equivalent of a bidirectional pipe with TTY capabilities.

Recent versions of the Linux kernels and GNU libc contain support for the System V/Unix98 naming scheme for PTYs, which assigns a common device, /dev/ptmx, to all the masters (opening it will automatically give you a previously unassigned PTY) and a subdirectory, /dev/pts, for the slaves; the slaves are named with decimal integers (/dev/pts/# in our notation). This removes the problem of exhausting the namespace and enables the kernel to automatically create the device nodes for the slaves on demand using the "devpts" filesystem.

CHAPTER

FOUR

DOCUMENTATION FOR /PROC/SYS

Copyright (c) 1998, 1999, Rik van Riel <riel@nl.linux.org>

'Why', I hear you ask, 'would anyone even _want_ documentation for them sysctl files? If anybody really needs it, it's all in the source...'

Well, this documentation is written because some people either don't know they need to tweak something, or because they don't have the time or knowledge to read the source code.

Furthermore, the programmers who built sysctl have built it to be actually used, not just for the fun of programming it :-)

Legal blurb:

As usual, there are two main things to consider:

1. you get what you pay for
2. it's free

The consequences are that I won't guarantee the correctness of this document, and if you come to me complaining about how you screwed up your system because of wrong documentation, I won't feel sorry for you. I might even laugh at you...

But of course, if you _do_ manage to screw up your system using only the sysctl options used in this file, I'd like to hear of it. Not only to have a great laugh, but also to make sure that you're the last RTFMing person to screw up.

In short, e-mail your suggestions, corrections and / or horror stories to: <riel@nl.linux.org>

Rik van Riel.

4.1 Introduction

Sysctl is a means of configuring certain aspects of the kernel at run-time, and the /proc/sys/ directory is there so that you don't even need special tools to do it! In fact, there are only four things needed to use these config facilities:

- a running Linux system
- root access

- common sense (this is especially hard to come by these days)
- knowledge of what all those values mean

As a quick 'ls /proc/sys' will show, the directory consists of several (arch-dependent?) subdirs. Each subdir is mainly about one part of the kernel, so you can do configuration on a piece by piece basis, or just some 'thematic frobbing'.

This documentation is about:

abi/	execution domains & personalities
debug/	<empty>
dev/	device specific information (eg dev/cdrom/info)
fs/	specific filesystems filehandle, inode, dentry and quota tuning binfmt_misc <*Kernel Support for miscellaneous Binary Formats (binfmt_misc)*>
kernel/	global kernel info / tuning miscellaneous stuff
net/	networking stuff, for documentation look in: <Documentation/networking/>
proc/	<empty>
sunrpc/	SUN Remote Procedure Call (NFS)
vm/	memory management tuning buffer and cache management
user/	Per user per user namespace limits

These are the subdirs I have on my system. There might be more or other subdirs in another setup. If you see another dir, I'd really like to hear about it :-)

4.1.1 Documentation for /proc/sys/abi/

Copyright (c) 2020, Stephen Kitt

For general info, see *Documentation for /proc/sys*.

The files in `/proc/sys/abi` can be used to see and modify ABI-related settings.

Currently, these files might (depending on your configuration) show up in `/proc/sys/kernel`:

- *vsyscall32 (x86)*

vsyscall32 (x86)

Determines whether the kernels maps a vDSO page into 32-bit processes; can be set to 1 to enable, or 0 to disable. Defaults to enabled if `CONFIG_COMPAT_VDSO` is set, disabled otherwise.

This controls the same setting as the `vdso32` kernel boot parameter.

4.1.2 Documentation for /proc/sys/fs/

Copyright (c) 1998, 1999, Rik van Riel <riel@nl.linux.org>

Copyright (c) 2009, Shen Feng<shen@cn.fujitsu.com>

For general info and legal blurb, please look in intro.rst.

This file contains documentation for the sysctl files and directories in `/proc/sys/fs/`.

The files in this directory can be used to tune and monitor miscellaneous and general things in the operation of the Linux kernel. Since some of the files *can* be used to screw up your system, it is advisable to read both documentation and source before actually making adjustments.

1. /proc/sys/fs

Currently, these files might (depending on your configuration) show up in `/proc/sys/fs`:

- *aio-nr & aio-max-nr*
- *dentry-state*
- *file-max & file-nr*
- *inode-nr & inode-state*
- *mount-max*
- *nr_open*
- *overflowgid & overflowuid*
- *pipe-user-pages-hard*
- *pipe-user-pages-soft*
- *protected_fifos*
- *protected_hardlinks*
- *protected_regular*
- *protected_symlinks*
- *suid_dumpable*

aio-nr & aio-max-nr

`aio-nr` shows the current system-wide number of asynchronous io requests. `aio-max-nr` allows you to change the maximum value `aio-nr` can grow to. If `aio-nr` reaches `aio-nr-max` then `io_setup` will fail with `EAGAIN`. Note that raising `aio-max-nr` does not result in the pre-allocation or re-sizing of any kernel data structures.

dentry-state

This file shows the values in `struct dentry_stat_t`, as defined in `fs/dcache.c`:

```
struct dentry_stat_t dentry_stat {
      long nr_dentry;
      long nr_unused;
      long age_limit;         /* age in seconds */
      long want_pages;        /* pages requested by system */
      long nr_negative;       /* # of unused negative dentries */
      long dummy;             /* Reserved for future use */
};
```

Dentries are dynamically allocated and deallocated.

`nr_dentry` shows the total number of dentries allocated (active + unused). `nr_unused shows` the number of dentries that are not actively used, but are saved in the LRU list for future reuse.

`age_limit` is the age in seconds after which dcache entries can be reclaimed when memory is short and `want_pages` is nonzero when `shrink_dcache_pages()` has been called and the dcache isn't pruned yet.

`nr_negative` shows the number of unused dentries that are also negative dentries which do not map to any files. Instead, they help speeding up rejection of non-existing files provided by the users.

file-max & file-nr

The value in `file-max` denotes the maximum number of file- handles that the Linux kernel will allocate. When you get lots of error messages about running out of file handles, you might want to increase this limit.

Historically,the kernel was able to allocate file handles dynamically, but not to free them again. The three values in `file-nr` denote the number of allocated file handles, the number of allocated but unused file handles, and the maximum number of file handles. Linux 2.6 and later always reports 0 as the number of free file handles -- this is not an error, it just means that the number of allocated file handles exactly matches the number of used file handles.

Attempts to allocate more file descriptors than `file-max` are reported with `printk`, look for:

```
VFS: file-max limit <number> reached
```

in the kernel logs.

inode-nr & inode-state

As with file handles, the kernel allocates the inode structures dynamically, but can't free them yet.

The file `inode-nr` contains the first two items from `inode-state`, so we'll skip to that file...

`inode-state` contains three actual numbers and four dummies. The actual numbers are, in order of appearance, `nr_inodes`, `nr_free_inodes` and `preshrink`.

`nr_inodes` stands for the number of inodes the system has allocated.

`nr_free_inodes` represents the number of free inodes (?) and preshrink is nonzero when the system needs to prune the inode list instead of allocating more.

mount-max

This denotes the maximum number of mounts that may exist in a mount namespace.

nr_open

This denotes the maximum number of file-handles a process can allocate. Default value is 1024*1024 (1048576) which should be enough for most machines. Actual limit depends on `RLIMIT_NOFILE` resource limit.

overflowgid & overflowuid

Some filesystems only support 16-bit UIDs and GIDs, although in Linux UIDs and GIDs are 32 bits. When one of these filesystems is mounted with writes enabled, any UID or GID that would exceed 65535 is translated to a fixed value before being written to disk.

These sysctls allow you to change the value of the fixed UID and GID. The default is 65534.

pipe-user-pages-hard

Maximum total number of pages a non-privileged user may allocate for pipes. Once this limit is reached, no new pipes may be allocated until usage goes below the limit again. When set to 0, no limit is applied, which is the default setting.

pipe-user-pages-soft

Maximum total number of pages a non-privileged user may allocate for pipes before the pipe size gets limited to a single page. Once this limit is reached, new pipes will be limited to a single page in size for this user in order to limit total memory usage, and trying to increase them using `fcntl()` will be denied until usage goes below the limit again. The default value allows to allocate up to 1024 pipes at their default size. When set to 0, no limit is applied.

protected_fifos

The intent of this protection is to avoid unintentional writes to an attacker-controlled FIFO, where a program expected to create a regular file.

When set to "0", writing to FIFOs is unrestricted.

When set to "1" don't allow `O_CREAT` open on FIFOs that we don't own in world writable sticky directories, unless they are owned by the owner of the directory.

When set to "2" it also applies to group writable sticky directories.

This protection is based on the restrictions in Openwall.

protected_hardlinks

A long-standing class of security issues is the hardlink-based time-of-check-time-of-use race, most commonly seen in world-writable directories like `/tmp`. The common method of exploitation of this flaw is to cross privilege boundaries when following a given hardlink (i.e. a root process follows a hardlink created by another user). Additionally, on systems without separated partitions, this stops unauthorized users from "pinning" vulnerable setuid/setgid files against being upgraded by the administrator, or linking to special files.

When set to "0", hardlink creation behavior is unrestricted.

When set to "1" hardlinks cannot be created by users if they do not already own the source file, or do not have read/write access to it.

This protection is based on the restrictions in Openwall and grsecurity.

protected_regular

This protection is similar to *protected_fifos*, but it avoids writes to an attacker-controlled regular file, where a program expected to create one.

When set to "0", writing to regular files is unrestricted.

When set to "1" don't allow `O_CREAT` open on regular files that we don't own in world writable sticky directories, unless they are owned by the owner of the directory.

When set to "2" it also applies to group writable sticky directories.

protected_symlinks

A long-standing class of security issues is the symlink-based time-of-check-time-of-use race, most commonly seen in world-writable directories like `/tmp`. The common method of exploitation of this flaw is to cross privilege boundaries when following a given symlink (i.e. a root process follows a symlink belonging to another user). For a likely incomplete list of hundreds of examples across the years, please see: https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=/tmp

When set to "0", symlink following behavior is unrestricted.

When set to "1" symlinks are permitted to be followed only when outside a sticky world-writable directory, or when the uid of the symlink and follower match, or when the directory owner matches the symlink's owner.

This protection is based on the restrictions in Openwall and grsecurity.

suid_dumpable

This value can be used to query and set the core dump mode for setuid or otherwise protected/tainted binaries. The modes are

0	(default)	Traditional behaviour. Any process which has changed privilege levels or is execute only will not be dumped.
1	(debug)	All processes dump core when possible. The core dump is owned by the current user and no security is applied. This is intended for system debugging situations only. Ptrace is unchecked. This is insecure as it allows regular users to examine the memory contents of privileged processes.
2	(suidsafe)	Any binary which normally would not be dumped is dumped anyway, but only if the `core_pattern` kernel sysctl (see *Documentation/admin-guide/sysctl/kernel.rst*) is set to either a pipe handler or a fully qualified path. (For more details on this limitation, see CVE-2006-2451.) This mode is appropriate when administrators are attempting to debug problems in a normal environment, and either have a core dump pipe handler that knows to treat privileged core dumps with care, or specific directory defined for catching core dumps. If a core dump happens without a pipe handler or fully qualified path, a message will be emitted to syslog warning about the lack of a correct setting.

2. /proc/sys/fs/binfmt_misc

Documentation for the files in `/proc/sys/fs/binfmt_misc` is in *Kernel Support for miscellaneous Binary Formats (binfmt_misc)*.

3. /proc/sys/fs/mqueue - POSIX message queues filesystem

The "mqueue" filesystem provides the necessary kernel features to enable the creation of a user space library that implements the POSIX message queues API (as noted by the MSG tag in the POSIX 1003.1-2001 version of the System Interfaces specification.)

The "mqueue" filesystem contains values for determining/setting the amount of resources used by the file system.

`/proc/sys/fs/mqueue/queues_max` is a read/write file for setting/getting the maximum number of message queues allowed on the system.

`/proc/sys/fs/mqueue/msg_max` is a read/write file for setting/getting the maximum number of messages in a queue value. In fact it is the limiting value for another (user) limit which is set in `mq_open` invocation. This attribute of a queue must be less than or equal to `msg_max`.

`/proc/sys/fs/mqueue/msgsize_max` is a read/write file for setting/getting the maximum message size value (it is an attribute of every message queue, set during its creation).

`/proc/sys/fs/mqueue/msg_default` is a read/write file for setting/getting the default number of messages in a queue value if the `attr` parameter of `mq_open(2)` is `NULL`. If it exceeds `msg_max`, the default value is initialized to `msg_max`.

`/proc/sys/fs/mqueue/msgsize_default` is a read/write file for setting/getting the default message size value if the `attr` parameter of `mq_open(2)` is `NULL`. If it exceeds `msgsize_max`, the default value is initialized to `msgsize_max`.

4. /proc/sys/fs/epoll - Configuration options for the epoll interface

This directory contains configuration options for the epoll(7) interface.

max_user_watches

Every epoll file descriptor can store a number of files to be monitored for event readiness. Each one of these monitored files constitutes a "watch". This configuration option sets the maximum number of "watches" that are allowed for each user. Each "watch" costs roughly 90 bytes on a 32-bit kernel, and roughly 160 bytes on a 64-bit one. The current default value for `max_user_watches` is 4% of the available low memory, divided by the "watch" cost in bytes.

4.1.3 Documentation for /proc/sys/kernel/

Copyright (c) 1998, 1999, Rik van Riel <riel@nl.linux.org>

Copyright (c) 2009, Shen Feng<shen@cn.fujitsu.com>

For general info and legal blurb, please look in *Documentation for /proc/sys*.

This file contains documentation for the sysctl files in `/proc/sys/kernel/`.

The files in this directory can be used to tune and monitor miscellaneous and general things in the operation of the Linux kernel. Since some of the files *can* be used to screw up your system, it is advisable to read both documentation and source before actually making adjustments.

Currently, these files might (depending on your configuration) show up in `/proc/sys/kernel`:

- *acct*
- *acpi_video_flags*
- *arch*
- *auto_msgmni*
- *bootloader_type (x86 only)*
- *bootloader_version (x86 only)*
- *bpf_stats_enabled*
- *cad_pid*
- *cap_last_cap*

- *core_pattern*
- *core_pipe_limit*
- *core_uses_pid*
- *ctrl-alt-del*
- *dmesg_restrict*
- *domainname & hostname*
- *firmware_config*
- *ftrace_dump_on_oops*
- *ftrace_enabled, stack_tracer_enabled*
- *hardlockup_all_cpu_backtrace*
- *hardlockup_panic*
- *hotplug*
- *hung_task_all_cpu_backtrace*
- *hung_task_panic*
- *hung_task_check_count*
- *hung_task_timeout_secs*
- *hung_task_check_interval_secs*
- *hung_task_warnings*
- *hyperv_record_panic_msg*
- *ignore-unaligned-usertrap*
- *io_uring_disabled*
- *io_uring_group*
- *kexec_load_disabled*
- *kexec_load_limit_panic*
- *kexec_load_limit_reboot*
- *kptr_restrict*
- *modprobe*
- *modules_disabled*
- *msgmax, msgmnb, and msgmni*
- *msg_next_id, sem_next_id, and shm_next_id (System V IPC)*
- *ngroups_max*
- *nmi_watchdog*
- *nmi_wd_lpm_factor (PPC only)*
- *numa_balancing*

- *numa_balancing_promote_rate_limit_MBps*
- *oops_all_cpu_backtrace*
- *oops_limit*
- *osrelease, ostype & version*
- *overflowgid & overflowuid*
- *panic*
- *panic_on_io_nmi*
- *panic_on_oops*
- *panic_on_stackoverflow*
- *panic_on_unrecovered_nmi*
- *panic_on_warn*
- *panic_print*
- *panic_on_rcu_stall*
- *max_rcu_stall_to_panic*
- *perf_cpu_time_max_percent*
- *perf_event_paranoid*
- *perf_event_max_stack*
- *perf_event_mlock_kb*
- *perf_event_max_contexts_per_stack*
- *perf_user_access (arm64 and riscv only)*
- *arm64*
- *riscv*
- *pid_max*
- *ns_last_pid*
- *powersave-nap (PPC only)*
- *printk*
- *printk_delay*
- *printk_ratelimit*
- *printk_ratelimit_burst*
- *printk_devkmsg*
- *pty*
- *random*
- *randomize_va_space*
- *real-root-dev*

- *reboot-cmd (SPARC only)*
- *sched_energy_aware*
- *task_delayacct*
- *sched_schedstats*
- *sched_util_clamp_min*
- *sched_util_clamp_max*
- *sched_util_clamp_min_rt_default*
- *seccomp*
- *sg-big-buff*
- *shmall*
- *shmmax*
- *shmmni*
- *shm_rmid_forced*
- *sysctl_writes_strict*
- *softlockup_all_cpu_backtrace*
- *softlockup_panic*
- *soft_watchdog*
- *split_lock_mitigate (x86 only)*
- *stack_erasing*
- *stop-a (SPARC only)*
- *sysrq*
- *tainted*
- *threads-max*
- *traceoff_on_warning*
- *tracepoint_printk*
- *unaligned-trap*
- *unknown_nmi_panic*
- *unprivileged_bpf_disabled*
- *warn_limit*
- *watchdog*
- *watchdog_cpumask*
- *watchdog_thresh*

acct

```
highwater lowwater frequency
```

If BSD-style process accounting is enabled these values control its behaviour. If free space on filesystem where the log lives goes below `lowwater`% accounting suspends. If free space gets above `highwater`% accounting resumes. `frequency` determines how often do we check the amount of free space (value is in seconds). Default:

```
4 2 30
```

That is, suspend accounting if free space drops below 2%; resume it if it increases to at least 4%; consider information about amount of free space valid for 30 seconds.

acpi_video_flags

See Documentation/power/video.rst. This allows the video resume mode to be set, in a similar fashion to the `acpi_sleep` kernel parameter, by combining the following values:

1	s3_bios
2	s3_mode
4	s3_beep

arch

The machine hardware name, the same output as `uname -m` (e.g. `x86_64` or `aarch64`).

auto_msgmni

This variable has no effect and may be removed in future kernel releases. Reading it always returns 0. Up to Linux 3.17, it enabled/disabled automatic recomputing of *msgmni* upon memory add/remove or upon IPC namespace creation/removal. Echoing "1" into this file enabled msgmni automatic recomputing. Echoing "0" turned it off. The default value was 1.

bootloader_type (x86 only)

This gives the bootloader type number as indicated by the bootloader, shifted left by 4, and OR'd with the low four bits of the bootloader version. The reason for this encoding is that this used to match the `type_of_loader` field in the kernel header; the encoding is kept for backwards compatibility. That is, if the full bootloader type number is 0x15 and the full version number is 0x234, this file will contain the value 340 = 0x154.

See the `type_of_loader` and `ext_loader_type` fields in Documentation/arch/x86/boot.rst for additional information.

bootloader_version (x86 only)

The complete bootloader version number. In the example above, this file will contain the value 564 = 0x234.

See the `type_of_loader` and `ext_loader_ver` fields in Documentation/arch/x86/boot.rst for additional information.

bpf_stats_enabled

Controls whether the kernel should collect statistics on BPF programs (total time spent running, number of times run...). Enabling statistics causes a slight reduction in performance on each program run. The statistics can be seen using `bpftool`.

0	Don't collect statistics (default).
1	Collect statistics.

cad_pid

This is the pid which will be signalled on reboot (notably, by Ctrl-Alt-Delete). Writing a value to this file which doesn't correspond to a running process will result in `-ESRCH`.

See also *ctrl-alt-del*.

cap_last_cap

Highest valid capability of the running kernel. Exports `CAP_LAST_CAP` from the kernel.

core_pattern

`core_pattern` is used to specify a core dumpfile pattern name.

- max length 127 characters; default value is "core"
- `core_pattern` is used as a pattern template for the output filename; certain string patterns (beginning with '%') are substituted with their actual values.
- backward compatibility with `core_uses_pid`:

  If `core_pattern` does not include "%p" (default does not) and `core_uses_pid` is set, then .PID will be appended to the filename.
- corename format specifiers

%<NUL>	'%' is dropped
%%	output one '%'
%p	pid
%P	global pid (init PID namespace)
%i	tid
%I	global tid (init PID namespace)
%u	uid (in initial user namespace)
%g	gid (in initial user namespace)
%d	dump mode, matches `PR_SET_DUMPABLE` and `/proc/sys/fs/suid_dumpable`
%s	signal number
%t	UNIX time of dump
%h	hostname
%e	executable filename (may be shortened, could be changed by prctl etc)
%f	executable filename
%E	executable path
%c	maximum size of core file by resource limit RLIMIT_CORE
%C	CPU the task ran on
%<OTHER>	both are dropped

- If the first character of the pattern is a '|', the kernel will treat the rest of the pattern as a command to run. The core dump will be written to the standard input of that program instead of to a file.

core_pipe_limit

This sysctl is only applicable when *core_pattern* is configured to pipe core files to a user space helper (when the first character of `core_pattern` is a '|', see above). When collecting cores via a pipe to an application, it is occasionally useful for the collecting application to gather data about the crashing process from its `/proc/pid` directory. In order to do this safely, the kernel must wait for the collecting process to exit, so as not to remove the crashing processes proc files prematurely. This in turn creates the possibility that a misbehaving userspace collecting process can block the reaping of a crashed process simply by never exiting. This sysctl defends against that. It defines how many concurrent crashing processes may be piped to user space applications in parallel. If this value is exceeded, then those crashing processes above that value are noted via the kernel log and their cores are skipped. 0 is a special value, indicating that unlimited processes may be captured in parallel, but that no waiting will take place (i.e. the collecting process is not guaranteed access to `/proc/<crashing pid>/`). This value defaults to 0.

core_uses_pid

The default coredump filename is "core". By setting `core_uses_pid` to 1, the coredump filename becomes core.PID. If *core_pattern* does not include "%p" (default does not) and `core_uses_pid` is set, then .PID will be appended to the filename.

ctrl-alt-del

When the value in this file is 0, ctrl-alt-del is trapped and sent to the `init(1)` program to handle a graceful restart. When, however, the value is > 0, Linux's reaction to a Vulcan Nerve Pinch (tm) will be an immediate reboot, without even syncing its dirty buffers.

Note:
when a program (like dosemu) has the keyboard in 'raw' mode, the ctrl-alt-del is intercepted by the program before it ever reaches the kernel tty layer, and it's up to the program to decide what to do with it.

dmesg_restrict

This toggle indicates whether unprivileged users are prevented from using `dmesg(8)` to view messages from the kernel's log buffer. When `dmesg_restrict` is set to 0 there are no restrictions. When `dmesg_restrict` is set to 1, users must have `CAP_SYSLOG` to use `dmesg(8)`.

The kernel config option `CONFIG_SECURITY_DMESG_RESTRICT` sets the default value of `dmesg_restrict`.

domainname & hostname

These files can be used to set the NIS/YP domainname and the hostname of your box in exactly the same way as the commands domainname and hostname, i.e.:

```
# echo "darkstar" > /proc/sys/kernel/hostname
# echo "mydomain" > /proc/sys/kernel/domainname
```

has the same effect as:

```
# hostname "darkstar"
# domainname "mydomain"
```

Note, however, that the classic darkstar.frop.org has the hostname "darkstar" and DNS (Internet Domain Name Server) domainname "frop.org", not to be confused with the NIS (Network Information Service) or YP (Yellow Pages) domainname. These two domain names are in general different. For a detailed discussion see the `hostname(1)` man page.

firmware_config

See Documentation/driver-api/firmware/fallback-mechanisms.rst.

The entries in this directory allow the firmware loader helper fallback to be controlled:

- `force_sysfs_fallback`, when set to 1, forces the use of the fallback;
- `ignore_sysfs_fallback`, when set to 1, ignores any fallback.

ftrace_dump_on_oops

Determines whether `ftrace_dump()` should be called on an oops (or kernel panic). This will output the contents of the ftrace buffers to the console. This is very useful for capturing traces that lead to crashes and outputting them to a serial console.

0	Disabled (default).
1	Dump buffers of all CPUs.
2	Dump the buffer of the CPU that triggered the oops.

ftrace_enabled, stack_tracer_enabled

See Documentation/trace/ftrace.rst.

hardlockup_all_cpu_backtrace

This value controls the hard lockup detector behavior when a hard lockup condition is detected as to whether or not to gather further debug information. If enabled, arch-specific all-CPU stack dumping will be initiated.

0	Do nothing. This is the default behavior.
1	On detection capture more debug information.

hardlockup_panic

This parameter can be used to control whether the kernel panics when a hard lockup is detected.

0	Don't panic on hard lockup.
1	Panic on hard lockup.

See *Softlockup detector and hardlockup detector (aka nmi_watchdog)* for more information. This can also be set using the nmi_watchdog kernel parameter.

hotplug

Path for the hotplug policy agent. Default value is `CONFIG_UEVENT_HELPER_PATH`, which in turn defaults to the empty string.

This file only exists when `CONFIG_UEVENT_HELPER` is enabled. Most modern systems rely exclusively on the netlink-based uevent source and don't need this.

hung_task_all_cpu_backtrace

If this option is set, the kernel will send an NMI to all CPUs to dump their backtraces when a hung task is detected. This file shows up if CONFIG_DETECT_HUNG_TASK and CONFIG_SMP are enabled.

0: Won't show all CPUs backtraces when a hung task is detected. This is the default behavior.

1: Will non-maskably interrupt all CPUs and dump their backtraces when a hung task is detected.

hung_task_panic

Controls the kernel's behavior when a hung task is detected. This file shows up if `CONFIG_DETECT_HUNG_TASK` is enabled.

0	Continue operation. This is the default behavior.
1	Panic immediately.

hung_task_check_count

The upper bound on the number of tasks that are checked. This file shows up if `CONFIG_DETECT_HUNG_TASK` is enabled.

hung_task_timeout_secs

When a task in D state did not get scheduled for more than this value report a warning. This file shows up if `CONFIG_DETECT_HUNG_TASK` is enabled.

0 means infinite timeout, no checking is done.

Possible values to set are in range {0:`LONG_MAX/HZ`}.

hung_task_check_interval_secs

Hung task check interval. If hung task checking is enabled (see *hung_task_timeout_secs*), the check is done every `hung_task_check_interval_secs` seconds. This file shows up if `CONFIG_DETECT_HUNG_TASK` is enabled.

0 (default) means use `hung_task_timeout_secs` as checking interval.

Possible values to set are in range {0:`LONG_MAX/HZ`}.

hung_task_warnings

The maximum number of warnings to report. During a check interval if a hung task is detected, this value is decreased by 1. When this value reaches 0, no more warnings will be reported. This file shows up if `CONFIG_DETECT_HUNG_TASK` is enabled.

-1: report an infinite number of warnings.

hyperv_record_panic_msg

Controls whether the panic kmsg data should be reported to Hyper-V.

0	Do not report panic kmsg data.
1	Report the panic kmsg data. This is the default behavior.

ignore-unaligned-usertrap

On architectures where unaligned accesses cause traps, and where this feature is supported (`CONFIG_SYSCTL_ARCH_UNALIGN_NO_WARN`; currently, `arc` and `loongarch`), controls whether all unaligned traps are logged.

0	Log all unaligned accesses.
1	Only warn the first time a process traps. This is the default setting.

See also *unaligned-trap*.

io_uring_disabled

Prevents all processes from creating new io_uring instances. Enabling this shrinks the kernel's attack surface.

0	All processes can create io_uring instances as normal. This is the default setting.
1	io_uring creation is disabled (io_uring_setup() will fail with -EPERM) for unprivileged processes not in the io_uring_group group. Existing io_uring instances can still be used. See the documentation for io_uring_group for more information.
2	io_uring creation is disabled for all processes. io_uring_setup() always fails with -EPERM. Existing io_uring instances can still be used.

io_uring_group

When io_uring_disabled is set to 1, a process must either be privileged (CAP_SYS_ADMIN) or be in the io_uring_group group in order to create an io_uring instance. If io_uring_group is set to -1 (the default), only processes with the CAP_SYS_ADMIN capability may create io_uring instances.

kexec_load_disabled

A toggle indicating if the syscalls `kexec_load` and `kexec_file_load` have been disabled. This value defaults to 0 (false: `kexec_*load` enabled), but can be set to 1 (true: `kexec_*load` disabled). Once true, kexec can no longer be used, and the toggle cannot be set back to false. This allows a kexec image to be loaded before disabling the syscall, allowing a system to set up (and later use) an image without it being altered. Generally used together with the *modules_disabled* sysctl.

kexec_load_limit_panic

This parameter specifies a limit to the number of times the syscalls `kexec_load` and `kexec_file_load` can be called with a crash image. It can only be set with a more restrictive value than the current one.

-1	Unlimited calls to kexec. This is the default setting.
N	Number of calls left.

kexec_load_limit_reboot

Similar functionality as `kexec_load_limit_panic`, but for a normal image.

kptr_restrict

This toggle indicates whether restrictions are placed on exposing kernel addresses via `/proc` and other interfaces.

When `kptr_restrict` is set to 0 (the default) the address is hashed before printing. (This is the equivalent to %p.)

When `kptr_restrict` is set to 1, kernel pointers printed using the %pK format specifier will be replaced with 0s unless the user has `CAP_SYSLOG` and effective user and group ids are equal to the real ids. This is because %pK checks are done at read() time rather than open() time, so if permissions are elevated between the open() and the read() (e.g via a setuid binary) then %pK will not leak kernel pointers to unprivileged users. Note, this is a temporary solution only. The correct long-term solution is to do the permission checks at open() time. Consider removing world read permissions from files that use %pK, and using *dmesg_restrict* to protect against uses of %pK in `dmesg(8)` if leaking kernel pointer values to unprivileged users is a concern.

When `kptr_restrict` is set to 2, kernel pointers printed using %pK will be replaced with 0s regardless of privileges.

modprobe

The full path to the usermode helper for autoloading kernel modules, by default `CONFIG_MODPROBE_PATH`, which in turn defaults to "/sbin/modprobe". This binary is executed when the kernel requests a module. For example, if userspace passes an unknown filesystem type to mount(), then the kernel will automatically request the corresponding filesystem module by executing this usermode helper. This usermode helper should insert the needed module into the kernel.

This sysctl only affects module autoloading. It has no effect on the ability to explicitly insert modules.

This sysctl can be used to debug module loading requests:

```
echo '#! /bin/sh' > /tmp/modprobe
echo 'echo "$@" >> /tmp/modprobe.log' >> /tmp/modprobe
echo 'exec /sbin/modprobe "$@"' >> /tmp/modprobe
chmod a+x /tmp/modprobe
echo /tmp/modprobe > /proc/sys/kernel/modprobe
```

Alternatively, if this sysctl is set to the empty string, then module autoloading is completely disabled. The kernel will not try to execute a usermode helper at all, nor will it call the kernel_module_request LSM hook.

If CONFIG_STATIC_USERMODEHELPER=y is set in the kernel configuration, then the configured static usermode helper overrides this sysctl, except that the empty string is still accepted to completely disable module autoloading as described above.

modules_disabled

A toggle value indicating if modules are allowed to be loaded in an otherwise modular kernel. This toggle defaults to off (0), but can be set true (1). Once true, modules can be neither loaded nor unloaded, and the toggle cannot be set back to false. Generally used with the *kexec_load_disabled* toggle.

msgmax, msgmnb, and msgmni

`msgmax` is the maximum size of an IPC message, in bytes. 8192 by default (`MSGMAX`).

`msgmnb` is the maximum size of an IPC queue, in bytes. 16384 by default (`MSGMNB`).

`msgmni` is the maximum number of IPC queues. 32000 by default (`MSGMNI`).

msg_next_id, sem_next_id, and shm_next_id (System V IPC)

These three toggles allows to specify desired id for next allocated IPC object: message, semaphore or shared memory respectively.

By default they are equal to -1, which means generic allocation logic. Possible values to set are in range {0:INT_MAX}.

Notes:

1) kernel doesn't guarantee, that new object will have desired id. So, it's up to userspace, how to handle an object with "wrong" id.
2) Toggle with non-default value will be set back to -1 by kernel after successful IPC object allocation. If an IPC object allocation syscall fails, it is undefined if the value remains unmodified or is reset to -1.

ngroups_max

Maximum number of supplementary groups, _i.e._ the maximum size which setgroups will accept. Exports NGROUPS_MAX from the kernel.

nmi_watchdog

This parameter can be used to control the NMI watchdog (i.e. the hard lockup detector) on x86 systems.

0	Disable the hard lockup detector.
1	Enable the hard lockup detector.

The hard lockup detector monitors each CPU for its ability to respond to timer interrupts. The mechanism utilizes CPU performance counter registers that are programmed to generate Non-Maskable Interrupts (NMIs) periodically while a CPU is busy. Hence, the alternative name 'NMI watchdog'.

The NMI watchdog is disabled by default if the kernel is running as a guest in a KVM virtual machine. This default can be overridden by adding:

```
nmi_watchdog=1
```

to the guest kernel command line (see *The kernel's command-line parameters*).

nmi_wd_lpm_factor (PPC only)

Factor to apply to the NMI watchdog timeout (only when nmi_watchdog is set to 1). This factor represents the percentage added to watchdog_thresh when calculating the NMI watchdog timeout during an LPM. The soft lockup timeout is not impacted.

A value of 0 means no change. The default value is 200 meaning the NMI watchdog is set to 30s (based on watchdog_thresh equal to 10).

numa_balancing

Enables/disables and configures automatic page fault based NUMA memory balancing. Memory is moved automatically to nodes that access it often. The value to set can be the result of ORing the following:

0	NUMA_BALANCING_DISABLED
1	NUMA_BALANCING_NORMAL
2	NUMA_BALANCING_MEMORY_TIERING

Or NUMA_BALANCING_NORMAL to optimize page placement among different NUMA nodes to reduce remote accessing. On NUMA machines, there is a performance penalty if remote memory is accessed by a CPU. When this feature is enabled the kernel samples what task thread is accessing memory by periodically unmapping pages and later trapping a page fault. At the time of the page fault, it is determined if the data being accessed should be migrated to a local memory node.

The unmapping of pages and trapping faults incur additional overhead that ideally is offset by improved memory locality but there is no universal guarantee. If the target workload is already bound to NUMA nodes then this feature should be disabled.

Or NUMA_BALANCING_MEMORY_TIERING to optimize page placement among different types of memory (represented as different NUMA nodes) to place the hot pages in the fast memory. This is implemented based on unmapping and page fault too.

numa_balancing_promote_rate_limit_MBps

Too high promotion/demotion throughput between different memory types may hurt application latency. This can be used to rate limit the promotion throughput. The per-node max promotion throughput in MB/s will be limited to be no more than the set value.

A rule of thumb is to set this to less than 1/10 of the PMEM node write bandwidth.

oops_all_cpu_backtrace

If this option is set, the kernel will send an NMI to all CPUs to dump their backtraces when an oops event occurs. It should be used as a last resort in case a panic cannot be triggered (to protect VMs running, for example) or kdump can't be collected. This file shows up if CONFIG_SMP is enabled.

0: Won't show all CPUs backtraces when an oops is detected. This is the default behavior.

1: Will non-maskably interrupt all CPUs and dump their backtraces when an oops event is detected.

oops_limit

Number of kernel oopses after which the kernel should panic when `panic_on_oops` is not set. Setting this to 0 disables checking the count. Setting this to 1 has the same effect as setting `panic_on_oops=1`. The default value is 10000.

osrelease, ostype & version

```
# cat osrelease
2.1.88
# cat ostype
Linux
# cat version
#5 Wed Feb 25 21:49:24 MET 1998
```

The files `osrelease` and `ostype` should be clear enough. `version` needs a little more clarification however. The '#5' means that this is the fifth kernel built from this source base and the date behind it indicates the time the kernel was built. The only way to tune these values is to rebuild the kernel :-)

overflowgid & overflowuid

if your architecture did not always support 32-bit UIDs (i.e. arm, i386, m68k, sh, and sparc32), a fixed UID and GID will be returned to applications that use the old 16-bit UID/GID system calls, if the actual UID or GID would exceed 65535.

These sysctls allow you to change the value of the fixed UID and GID. The default is 65534.

panic

The value in this file determines the behaviour of the kernel on a panic:

- if zero, the kernel will loop forever;
- if negative, the kernel will reboot immediately;
- if positive, the kernel will reboot after the corresponding number of seconds.

When you use the software watchdog, the recommended setting is 60.

panic_on_io_nmi

Controls the kernel's behavior when a CPU receives an NMI caused by an IO error.

0	Try to continue operation (default).
1	Panic immediately. The IO error triggered an NMI. This indicates a serious system condition which could result in IO data corruption. Rather than continuing, panicking might be a better choice. Some servers issue this sort of NMI when the dump button is pushed, and you can use this option to take a crash dump.

panic_on_oops

Controls the kernel's behaviour when an oops or BUG is encountered.

0	Try to continue operation.
1	Panic immediately. If the *panic* sysctl is also non-zero then the machine will be rebooted.

panic_on_stackoverflow

Controls the kernel's behavior when detecting the overflows of kernel, IRQ and exception stacks except a user stack. This file shows up if `CONFIG_DEBUG_STACKOVERFLOW` is enabled.

0	Try to continue operation.
1	Panic immediately.

panic_on_unrecovered_nmi

The default Linux behaviour on an NMI of either memory or unknown is to continue operation. For many environments such as scientific computing it is preferable that the box is taken out and the error dealt with than an uncorrected parity/ECC error get propagated.

A small number of systems do generate NMIs for bizarre random reasons such as power management so the default is off. That sysctl works like the existing panic controls already in that directory.

panic_on_warn

Calls panic() in the WARN() path when set to 1. This is useful to avoid a kernel rebuild when attempting to kdump at the location of a WARN().

0	Only WARN(), default behaviour.
1	Call panic() after printing out WARN() location.

panic_print

Bitmask for printing system info when panic happens. User can chose combination of the following bits:

bit 0	print all tasks info
bit 1	print system memory info
bit 2	print timer info
bit 3	print locks info if `CONFIG_LOCKDEP` is on
bit 4	print ftrace buffer
bit 5	print all printk messages in buffer
bit 6	print all CPUs backtrace (if available in the arch)

So for example to print tasks and memory info on panic, user can:

```
echo 3 > /proc/sys/kernel/panic_print
```

panic_on_rcu_stall

When set to 1, calls panic() after RCU stall detection messages. This is useful to define the root cause of RCU stalls using a vmcore.

0	Do not panic() when RCU stall takes place, default behavior.
1	panic() after printing RCU stall messages.

max_rcu_stall_to_panic

When `panic_on_rcu_stall` is set to 1, this value determines the number of times that RCU can stall before panic() is called.

When `panic_on_rcu_stall` is set to 0, this value is has no effect.

perf_cpu_time_max_percent

Hints to the kernel how much CPU time it should be allowed to use to handle perf sampling events. If the perf subsystem is informed that its samples are exceeding this limit, it will drop its sampling frequency to attempt to reduce its CPU usage.

Some perf sampling happens in NMIs. If these samples unexpectedly take too long to execute, the NMIs can become stacked up next to each other so much that nothing else is allowed to execute.

0	Disable the mechanism. Do not monitor or correct perf's sampling rate no matter how CPU time it takes.
1-100	Attempt to throttle perf's sample rate to this percentage of CPU. Note: the kernel calculates an ”expected” length of each sample event. 100 here means 100% of that expected length. Even if this is set to 100, you may still see sample throttling if this length is exceeded. Set to 0 if you truly do not care how much CPU is consumed.

perf_event_paranoid

Controls use of the performance events system by unprivileged users (without CAP_PERFMON). The default value is 2.

For backward compatibility reasons access to system performance monitoring and observability remains open for CAP_SYS_ADMIN privileged processes but CAP_SYS_ADMIN usage for secure system performance monitoring and observability operations is discouraged with respect to CAP_PERFMON use cases.

-1	Allow use of (almost) all events by all users. Ignore mlock limit after perf_event_mlock_kb without `CAP_IPC_LOCK`.
>=0	Disallow ftrace function tracepoint by users without `CAP_PERFMON`. Disallow raw tracepoint access by users without `CAP_PERFMON`.
>=1	Disallow CPU event access by users without `CAP_PERFMON`.
>=2	Disallow kernel profiling by users without `CAP_PERFMON`.

perf_event_max_stack

Controls maximum number of stack frames to copy for (`attr.sample_type` & `PERF_SAMPLE_CALLCHAIN`) configured events, for instance, when using '`perf record -g`' or '`perf trace --call-graph fp`'.

This can only be done when no events are in use that have callchains enabled, otherwise writing to this file will return `-EBUSY`.

The default value is 127.

perf_event_mlock_kb

Control size of per-cpu ring buffer not counted against mlock limit.

The default value is 512 + 1 page

perf_event_max_contexts_per_stack

Controls maximum number of stack frame context entries for (`attr.sample_type` & `PERF_SAMPLE_CALLCHAIN`) configured events, for instance, when using '`perf record -g`' or '`perf trace --call-graph fp`'.

This can only be done when no events are in use that have callchains enabled, otherwise writing to this file will return `-EBUSY`.

The default value is 8.

perf_user_access (arm64 and riscv only)

Controls user space access for reading perf event counters.

arm64

The default value is 0 (access disabled).

When set to 1, user space can read performance monitor counter registers directly.

See Documentation/arch/arm64/perf.rst for more information.

riscv

When set to 0, user space access is disabled.

The default value is 1, user space can read performance monitor counter registers through perf, any direct access without perf intervention will trigger an illegal instruction.

When set to 2, which enables legacy mode (user space has direct access to cycle and insret CSRs only). Note that this legacy value is deprecated and will be removed once all user space applications are fixed.

Note that the time CSR is always directly accessible to all modes.

pid_max

PID allocation wrap value. When the kernel's next PID value reaches this value, it wraps back to a minimum PID value. PIDs of value `pid_max` or larger are not allocated.

ns_last_pid

The last pid allocated in the current (the one task using this sysctl lives in) pid namespace. When selecting a pid for a next task on fork kernel tries to allocate a number starting from this one.

powersave-nap (PPC only)

If set, Linux-PPC will use the 'nap' mode of powersaving, otherwise the 'doze' mode will be used.

---

printk

The four values in printk denote: `console_loglevel`, `default_message_loglevel`, `minimum_console_loglevel` and `default_console_loglevel` respectively.

These values influence printk() behavior when printing or logging error messages. See '`man 2 syslog`' for more info on the different loglevels.

console_loglevel	messages with a higher priority than this will be printed to the console
default_message_loglevel	messages without an explicit priority will be printed with this priority
minimum_console_loglevel	minimum (highest) value to which console_loglevel can be set
default_console_loglevel	default value for console_loglevel

printk_delay

Delay each printk message in `printk_delay` milliseconds

Value from 0 - 10000 is allowed.

printk_ratelimit

Some warning messages are rate limited. `printk_ratelimit` specifies the minimum length of time between these messages (in seconds). The default value is 5 seconds.

A value of 0 will disable rate limiting.

printk_ratelimit_burst

While long term we enforce one message per *printk_ratelimit* seconds, we do allow a burst of messages to pass through. `printk_ratelimit_burst` specifies the number of messages we can send before ratelimiting kicks in.

The default value is 10 messages.

printk_devkmsg

Control the logging to `/dev/kmsg` from userspace:

ratelimit	default, ratelimited
on	unlimited logging to /dev/kmsg from userspace
off	logging to /dev/kmsg disabled

The kernel command line parameter `printk.devkmsg=` overrides this and is a one-time setting until next reboot: once set, it cannot be changed by this sysctl interface anymore.

---

pty

See Documentation/filesystems/devpts.rst.

random

This is a directory, with the following entries:

- `boot_id`: a UUID generated the first time this is retrieved, and unvarying after that;
- `uuid`: a UUID generated every time this is retrieved (this can thus be used to generate UUIDs at will);
- `entropy_avail`: the pool's entropy count, in bits;
- `poolsize`: the entropy pool size, in bits;

- `urandom_min_reseed_secs`: obsolete (used to determine the minimum number of seconds between urandom pool reseeding). This file is writable for compatibility purposes, but writing to it has no effect on any RNG behavior;
- `write_wakeup_threshold`: when the entropy count drops below this (as a number of bits), processes waiting to write to `/dev/random` are woken up. This file is writable for compatibility purposes, but writing to it has no effect on any RNG behavior.

randomize_va_space

This option can be used to select the type of process address space randomization that is used in the system, for architectures that support this feature.

0	Turn the process address space randomization off. This is the default for architectures that do not support this feature anyways, and kernels that are booted with the "norandmaps" parameter.
1	Make the addresses of mmap base, stack and VDSO page randomized. This, among other things, implies that shared libraries will be loaded to random addresses. Also for PIE-linked binaries, the location of code start is randomized. This is the default if the `CONFIG_COMPAT_BRK` option is enabled.
2	Additionally enable heap randomization. This is the default if `CONFIG_COMPAT_BRK` is disabled. There are a few legacy applications out there (such as some ancient versions of libc.so.5 from 1996) that assume that brk area starts just after the end of the code+bss. These applications break when start of the brk area is randomized. There are however no known non-legacy applications that would be broken this way, so for most systems it is safe to choose full randomization. Systems with ancient and/or broken binaries should be configured with `CONFIG_COMPAT_BRK` enabled, which excludes the heap from process address space randomization.

real-root-dev

See *Using the initial RAM disk (initrd)*.

reboot-cmd (SPARC only)

??? This seems to be a way to give an argument to the Sparc ROM/Flash boot loader. Maybe to tell it what to do after rebooting. ???

sched_energy_aware

Enables/disables Energy Aware Scheduling (EAS). EAS starts automatically on platforms where it can run (that is, platforms with asymmetric CPU topologies and having an Energy Model available). If your platform happens to meet the requirements for EAS but you do not want to use it, change this value to 0. On Non-EAS platforms, write operation fails and read doesn't return anything.

task_delayacct

Enables/disables task delay accounting (see Documentation/accounting/delay-accounting.rst. Enabling this feature incurs a small amount of overhead in the scheduler but is useful for debugging and performance tuning. It is required by some tools such as iotop.

sched_schedstats

Enables/disables scheduler statistics. Enabling this feature incurs a small amount of overhead in the scheduler but is useful for debugging and performance tuning.

sched_util_clamp_min

Max allowed *minimum* utilization.

Default value is 1024, which is the maximum possible value.

It means that any requested uclamp.min value cannot be greater than sched_util_clamp_min, i.e., it is restricted to the range [0:sched_util_clamp_min].

sched_util_clamp_max

Max allowed *maximum* utilization.

Default value is 1024, which is the maximum possible value.

It means that any requested uclamp.max value cannot be greater than sched_util_clamp_max, i.e., it is restricted to the range [0:sched_util_clamp_max].

sched_util_clamp_min_rt_default

By default Linux is tuned for performance. Which means that RT tasks always run at the highest frequency and most capable (highest capacity) CPU (in heterogeneous systems).

Uclamp achieves this by setting the requested uclamp.min of all RT tasks to 1024 by default, which effectively boosts the tasks to run at the highest frequency and biases them to run on the biggest CPU.

This knob allows admins to change the default behavior when uclamp is being used. In battery powered devices particularly, running at the maximum capacity and frequency will increase energy consumption and shorten the battery life.

This knob is only effective for RT tasks which the user hasn't modified their requested uclamp.min value via sched_setattr() syscall.

This knob will not escape the range constraint imposed by sched_util_clamp_min defined above.

For example if

> sched_util_clamp_min_rt_default = 800 sched_util_clamp_min = 600

Then the boost will be clamped to 600 because 800 is outside of the permissible range of [0:600]. This could happen for instance if a powersave mode will restrict all boosts temporarily by modifying sched_util_clamp_min. As soon as this restriction is lifted, the requested sched_util_clamp_min_rt_default will take effect.

seccomp

See Documentation/userspace-api/seccomp_filter.rst.

sg-big-buff

This file shows the size of the generic SCSI (sg) buffer. You can't tune it just yet, but you could change it on compile time by editing `include/scsi/sg.h` and changing the value of `SG_BIG_BUFF`.

There shouldn't be any reason to change this value. If you can come up with one, you probably know what you are doing anyway :)

shmall

This parameter sets the total amount of shared memory pages that can be used system wide. Hence, `shmall` should always be at least `ceil(shmmax/PAGE_SIZE)`.

If you are not sure what the default `PAGE_SIZE` is on your Linux system, you can run the following command:

```
# getconf PAGE_SIZE
```

shmmax

This value can be used to query and set the run time limit on the maximum shared memory segment size that can be created. Shared memory segments up to 1Gb are now supported in the kernel. This value defaults to `SHMMAX`.

shmmni

This value determines the maximum number of shared memory segments. 4096 by default (`SHMMNI`).

shm_rmid_forced

Linux lets you set resource limits, including how much memory one process can consume, via `setrlimit(2)`. Unfortunately, shared memory segments are allowed to exist without association with any process, and thus might not be counted against any resource limits. If enabled, shared memory segments are automatically destroyed when their attach count becomes zero after a detach or a process termination. It will also destroy segments that were created, but never attached to, on exit from the process. The only use left for `IPC_RMID` is to immediately destroy an unattached segment. Of course, this breaks the way things are defined, so some applications might stop working. Note that this feature will do you no good unless you also configure your resource limits (in particular, `RLIMIT_AS` and `RLIMIT_NPROC`). Most systems don't need this.

Note that if you change this from 0 to 1, already created segments without users and with a dead originative process will be destroyed.

sysctl_writes_strict

Control how file position affects the behavior of updating sysctl values via the `/proc/sys` interface:

-1	Legacy per-write sysctl value handling, with no printk warnings. Each write syscall must fully contain the sysctl value to be written, and multiple writes on the same sysctl file descriptor will rewrite the sysctl value, regardless of file position.
0	Same behavior as above, but warn about processes that perform writes to a sysctl file descriptor when the file position is not 0.
1	(default) Respect file position when writing sysctl strings. Multiple writes will append to the sysctl value buffer. Anything past the max length of the sysctl value buffer will be ignored. Writes to numeric sysctl entries must always be at file position 0 and the value must be fully contained in the buffer sent in the write syscall.

softlockup_all_cpu_backtrace

This value controls the soft lockup detector thread's behavior when a soft lockup condition is detected as to whether or not to gather further debug information. If enabled, each cpu will be issued an NMI and instructed to capture stack trace.

This feature is only applicable for architectures which support NMI.

0	Do nothing. This is the default behavior.
1	On detection capture more debug information.

softlockup_panic

This parameter can be used to control whether the kernel panics when a soft lockup is detected.

0	Don't panic on soft lockup.
1	Panic on soft lockup.

This can also be set using the softlockup_panic kernel parameter.

soft_watchdog

This parameter can be used to control the soft lockup detector.

0	Disable the soft lockup detector.
1	Enable the soft lockup detector.

The soft lockup detector monitors CPUs for threads that are hogging the CPUs without rescheduling voluntarily, and thus prevent the 'migration/N' threads from running, causing the watchdog work fail to execute. The mechanism depends on the CPUs ability to respond to timer interrupts which are needed for the watchdog work to be queued by the watchdog timer function, otherwise the NMI watchdog — if enabled — can detect a hard lockup condition.

split_lock_mitigate (x86 only)

On x86, each ”split lock” imposes a system-wide performance penalty. On larger systems, large numbers of split locks from unprivileged users can result in denials of service to well-behaved and potentially more important users.

The kernel mitigates these bad users by detecting split locks and imposing penalties: forcing them to wait and only allowing one core to execute split locks at a time.

These mitigations can make those bad applications unbearably slow. Setting split_lock_mitigate=0 may restore some application performance, but will also increase system exposure to denial of service attacks from split lock users.

0	Disable the mitigation mode - just warns the split lock on kernel log and exposes the system to denials of service from the split lockers.
1	Enable the mitigation mode (this is the default) - penalizes the split lockers with intentional performance degradation.

stack_erasing

This parameter can be used to control kernel stack erasing at the end of syscalls for kernels built with `CONFIG_GCC_PLUGIN_STACKLEAK`.

That erasing reduces the information which kernel stack leak bugs can reveal and blocks some uninitialized stack variable attacks. The tradeoff is the performance impact: on a single CPU system kernel compilation sees a 1% slowdown, other systems and workloads may vary.

0	Kernel stack erasing is disabled, STACKLEAK_METRICS are not updated.
1	Kernel stack erasing is enabled (default), it is performed before returning to the userspace at the end of syscalls.

stop-a (SPARC only)

Controls Stop-A:

0	Stop-A has no effect.
1	Stop-A breaks to the PROM (default).

Stop-A is always enabled on a panic, so that the user can return to the boot PROM.

sysrq

See *Linux Magic System Request Key Hacks*.

tainted

Non-zero if the kernel has been tainted. Numeric values, which can be ORed together. The letters are seen in "Tainted" line of Oops reports.

1	*(P)*	proprietary module was loaded
2	*(F)*	module was force loaded
4	*(S)*	kernel running on an out of specification system
8	*(R)*	module was force unloaded
16	*(M)*	processor reported a Machine Check Exception (MCE)
32	*(B)*	bad page referenced or some unexpected page flags
64	*(U)*	taint requested by userspace application
128	*(D)*	kernel died recently, i.e. there was an OOPS or BUG
256	*(A)*	an ACPI table was overridden by user
512	*(W)*	kernel issued warning
1024	*(C)*	staging driver was loaded
2048	*(I)*	workaround for bug in platform firmware applied
4096	*(O)*	externally-built (”out-of-tree”) module was loaded
8192	*(E)*	unsigned module was loaded
16384	*(L)*	soft lockup occurred
32768	*(K)*	kernel has been live patched
65536	*(X)*	Auxiliary taint, defined and used by for distros
131072	*(T)*	The kernel was built with the struct randomization plugin

See *Tainted kernels* for more information.

Note:

writes to this sysctl interface will fail with `EINVAL` if the kernel is booted with the command line option `panic_on_taint=<bitmask>,nousertaint` and any of the ORed together values being written to `tainted` match with the bitmask declared on panic_on_taint. See *The kernel's command-line parameters* for more details on that particular kernel command line option and its optional `nousertaint` switch.

threads-max

This value controls the maximum number of threads that can be created using `fork()`.

During initialization the kernel sets this value such that even if the maximum number of threads is created, the thread structures occupy only a part (1/8th) of the available RAM pages.

The minimum value that can be written to `threads-max` is 1.

The maximum value that can be written to `threads-max` is given by the constant `FUTEX_TID_MASK` (0x3fffffff).

If a value outside of this range is written to `threads-max` an `EINVAL` error occurs.

traceoff_on_warning

When set, disables tracing (see Documentation/trace/ftrace.rst) when a `WARN()` is hit.

tracepoint_printk

When tracepoints are sent to printk() (enabled by the `tp_printk` boot parameter), this entry provides runtime control:

```
echo 0 > /proc/sys/kernel/tracepoint_printk
```

will stop tracepoints from being sent to printk(), and:

```
echo 1 > /proc/sys/kernel/tracepoint_printk
```

will send them to printk() again.

This only works if the kernel was booted with `tp_printk` enabled.

See *The kernel's command-line parameters* and Documentation/trace/boottime-trace.rst.

unaligned-trap

On architectures where unaligned accesses cause traps, and where this feature is supported (`CONFIG_SYSCTL_ARCH_UNALIGN_ALLOW`; currently, `arc`, `parisc` and `loongarch`), controls whether unaligned traps are caught and emulated (instead of failing).

0	Do not emulate unaligned accesses.
1	Emulate unaligned accesses. This is the default setting.

See also *ignore-unaligned-usertrap*.

unknown_nmi_panic

The value in this file affects behavior of handling NMI. When the value is non-zero, unknown NMI is trapped and then panic occurs. At that time, kernel debugging information is displayed on console.

NMI switch that most IA32 servers have fires unknown NMI up, for example. If a system hangs up, try pressing the NMI switch.

unprivileged_bpf_disabled

Writing 1 to this entry will disable unprivileged calls to `bpf()`; once disabled, calling `bpf()` without `CAP_SYS_ADMIN` or `CAP_BPF` will return `-EPERM`. Once set to 1, this can't be cleared from the running kernel anymore.

Writing 2 to this entry will also disable unprivileged calls to `bpf()`, however, an admin can still change this setting later on, if needed, by writing 0 or 1 to this entry.

If BPF_UNPRIV_DEFAULT_OFF is enabled in the kernel config, then this entry will default to 2 instead of 0.

0	Unprivileged calls to bpf() are enabled
1	Unprivileged calls to bpf() are disabled without recovery
2	Unprivileged calls to bpf() are disabled

warn_limit

Number of kernel warnings after which the kernel should panic when panic_on_warn is not set. Setting this to 0 disables checking the warning count. Setting this to 1 has the same effect as setting panic_on_warn=1. The default value is 0.

watchdog

This parameter can be used to disable or enable the soft lockup detector *and* the NMI watchdog (i.e. the hard lockup detector) at the same time.

0	Disable both lockup detectors.
1	Enable both lockup detectors.

The soft lockup detector and the NMI watchdog can also be disabled or enabled individually, using the soft_watchdog and nmi_watchdog parameters. If the watchdog parameter is read, for example by executing:

```
cat /proc/sys/kernel/watchdog
```

the output of this command (0 or 1) shows the logical OR of soft_watchdog and nmi_watchdog.

watchdog_cpumask

This value can be used to control on which cpus the watchdog may run. The default cpumask is all possible cores, but if NO_HZ_FULL is enabled in the kernel config, and cores are specified with the nohz_full= boot argument, those cores are excluded by default. Offline cores can be included in this mask, and if the core is later brought online, the watchdog will be started based on the mask value.

Typically this value would only be touched in the nohz_full case to re-enable cores that by default were not running the watchdog, if a kernel lockup was suspected on those cores.

The argument value is the standard cpulist format for cpumasks, so for example to enable the watchdog on cores 0, 2, 3, and 4 you might say:

```
echo 0,2-4 > /proc/sys/kernel/watchdog_cpumask
```

watchdog_thresh

This value can be used to control the frequency of hrtimer and NMI events and the soft and hard lockup thresholds. The default threshold is 10 seconds.

The softlockup threshold is (`2 * watchdog_thresh`). Setting this tunable to zero will disable lockup detection altogether.

4.1.4 Documentation for /proc/sys/net/

Copyright

Copyright (c) 1999

- Terrehon Bowden <terrehon@pacbell.net>
- Bodo Bauer <bb@ricochet.net>

Copyright (c) 2000

- Jorge Nerin <comandante@zaralinux.com>

Copyright (c) 2009

- Shen Feng <shen@cn.fujitsu.com>

For general info and legal blurb, please look in *The Linux kernel user's and administrator's guide*.

This file contains the documentation for the sysctl files in /proc/sys/net

The interface to the networking parts of the kernel is located in /proc/sys/net. The following table shows all possible subdirectories. You may see only some of them, depending on your kernel's configuration.

Table : Subdirectories in /proc/sys/net

Directory	Content		Directory	Content
802	E802 protocol		mptcp	Multipath TCP
appletalk	Appletalk protocol		netfilter	Network Filter
ax25	AX25		netrom	NET/ROM
bridge	Bridging		rose	X.25 PLP layer
core	General parameter		tipc	TIPC
ethernet	Ethernet protocol		unix	Unix domain sockets
ipv4	IP version 4		x25	X.25 protocol
ipv6	IP version 6			

1. /proc/sys/net/core - Network core options

bpf_jit_enable

This enables the BPF Just in Time (JIT) compiler. BPF is a flexible and efficient infrastructure allowing to execute bytecode at various hook points. It is used in a number of Linux kernel subsystems such as networking (e.g. XDP, tc), tracing (e.g. kprobes, uprobes, tracepoints) and security (e.g. seccomp). LLVM has a BPF back end that can compile restricted C into a sequence of BPF instructions. After program load through bpf(2) and passing a verifier in the kernel, a JIT will then translate these BPF proglets into native CPU instructions. There are two flavors of JITs, the newer eBPF JIT currently supported on:

- x86_64
- x86_32
- arm64
- arm32
- ppc64
- ppc32
- sparc64
- mips64
- s390x
- riscv64
- riscv32
- loongarch64

And the older cBPF JIT supported on the following archs:

- mips
- sparc

eBPF JITs are a superset of cBPF JITs, meaning the kernel will migrate cBPF instructions into eBPF instructions and then JIT compile them transparently. Older cBPF JITs can only translate tcpdump filters, seccomp rules, etc, but not mentioned eBPF programs loaded through bpf(2).

Values:

- 0 - disable the JIT (default value)
- 1 - enable the JIT
- 2 - enable the JIT and ask the compiler to emit traces on kernel log.

bpf_jit_harden

This enables hardening for the BPF JIT compiler. Supported are eBPF JIT backends. Enabling hardening trades off performance, but can mitigate JIT spraying.

Values:

- 0 - disable JIT hardening (default value)
- 1 - enable JIT hardening for unprivileged users only
- 2 - enable JIT hardening for all users

where ”privileged user” in this context means a process having CAP_BPF or CAP_SYS_ADMIN in the root user name space.

bpf_jit_kallsyms

When BPF JIT compiler is enabled, then compiled images are unknown addresses to the kernel, meaning they neither show up in traces nor in /proc/kallsyms. This enables export of these addresses, which can be used for debugging/tracing. If bpf_jit_harden is enabled, this feature is disabled.

Values :

- 0 - disable JIT kallsyms export (default value)
- 1 - enable JIT kallsyms export for privileged users only

bpf_jit_limit

This enforces a global limit for memory allocations to the BPF JIT compiler in order to reject unprivileged JIT requests once it has been surpassed. bpf_jit_limit contains the value of the global limit in bytes.

dev_weight

The maximum number of packets that kernel can handle on a NAPI interrupt, it's a Per-CPU variable. For drivers that support LRO or GRO_HW, a hardware aggregated packet is counted as one packet in this context.

Default: 64

dev_weight_rx_bias

RPS (e.g. RFS, aRFS) processing is competing with the registered NAPI poll function of the driver for the per softirq cycle netdev_budget. This parameter influences the proportion of the configured netdev_budget that is spent on RPS based packet processing during RX softirq cycles. It is further meant for making current dev_weight adaptable for asymmetric CPU needs on RX/TX side of the network stack. (see dev_weight_tx_bias) It is effective on a per CPU basis. Determination is based on dev_weight and is calculated multiplicative (dev_weight * dev_weight_rx_bias).

Default: 1

dev_weight_tx_bias

Scales the maximum number of packets that can be processed during a TX softirq cycle. Effective on a per CPU basis. Allows scaling of current dev_weight for asymmetric net stack processing needs. Be careful to avoid making TX softirq processing a CPU hog.

Calculation is based on dev_weight (dev_weight * dev_weight_tx_bias).

Default: 1

default_qdisc

The default queuing discipline to use for network devices. This allows overriding the default of pfifo_fast with an alternative. Since the default queuing discipline is created without additional parameters so is best suited to queuing disciplines that work well without configuration like stochastic fair queue (sfq), CoDel (codel) or fair queue CoDel (fq_codel). Don't use queuing disciplines like Hierarchical Token Bucket or Deficit Round Robin which require setting up classes and bandwidths. Note that physical multiqueue interfaces still use mq as root qdisc, which in turn uses this default for its leaves. Virtual devices (like e.g. lo or veth) ignore this setting and instead default to noqueue.

Default: pfifo_fast

busy_read

Low latency busy poll timeout for socket reads. (needs CONFIG_NET_RX_BUSY_POLL) Approximate time in us to busy loop waiting for packets on the device queue. This sets the default value of the SO_BUSY_POLL socket option. Can be set or overridden per socket by setting socket option SO_BUSY_POLL, which is the preferred method of enabling. If you need to enable the feature globally via sysctl, a value of 50 is recommended.

Will increase power usage.

Default: 0 (off)

busy_poll

Low latency busy poll timeout for poll and select. (needs CONFIG_NET_RX_BUSY_POLL) Approximate time in us to busy loop waiting for events. Recommended value depends on the number of sockets you poll on. For several sockets 50, for several hundreds 100. For more than that you probably want to use epoll. Note that only sockets with SO_BUSY_POLL set will be busy polled, so you want to either selectively set SO_BUSY_POLL on those sockets or set sysctl.net.busy_read globally.

Will increase power usage.

Default: 0 (off)

rmem_default

The default setting of the socket receive buffer in bytes.

rmem_max

The maximum receive socket buffer size in bytes.

rps_default_mask

The default RPS CPU mask used on newly created network devices. An empty mask means RPS disabled by default.

tstamp_allow_data

Allow processes to receive tx timestamps looped together with the original packet contents. If disabled, transmit timestamp requests from unprivileged processes are dropped unless socket option SOF_TIMESTAMPING_OPT_TSONLY is set.

Default: 1 (on)

wmem_default

The default setting (in bytes) of the socket send buffer.

wmem_max

The maximum send socket buffer size in bytes.

message_burst and message_cost

These parameters are used to limit the warning messages written to the kernel log from the networking code. They enforce a rate limit to make a denial-of-service attack impossible. A higher message_cost factor, results in fewer messages that will be written. Message_burst controls when messages will be dropped. The default settings limit warning messages to one every five seconds.

warnings

This sysctl is now unused.

This was used to control console messages from the networking stack that occur because of problems on the network like duplicate address or bad checksums.

These messages are now emitted at KERN_DEBUG and can generally be enabled and controlled by the dynamic_debug facility.

netdev_budget

Maximum number of packets taken from all interfaces in one polling cycle (NAPI poll). In one polling cycle interfaces which are registered to polling are probed in a round-robin manner. Also, a polling cycle may not exceed netdev_budget_usecs microseconds, even if netdev_budget has not been exhausted.

netdev_budget_usecs

Maximum number of microseconds in one NAPI polling cycle. Polling will exit when either netdev_budget_usecs have elapsed during the poll cycle or the number of packets processed reaches netdev_budget.

netdev_max_backlog

Maximum number of packets, queued on the INPUT side, when the interface receives packets faster than kernel can process them.

netdev_rss_key

RSS (Receive Side Scaling) enabled drivers use a 40 bytes host key that is randomly generated. Some user space might need to gather its content even if drivers do not provide ethtool -x support yet.

```
myhost:~# cat /proc/sys/net/core/netdev_rss_key
84:50:f4:00:a8:15:d1:a7:e9:7f:1d:60:35:c7:47:25:42:97:74:ca:56:bb:b6:a1:d8: ...
↪ (52 bytes total)
```

File contains nul bytes if no driver ever called netdev_rss_key_fill() function.

Note:
/proc/sys/net/core/netdev_rss_key contains 52 bytes of key, but most drivers only use 40 bytes of it.

```
myhost:~# ethtool -x eth0
RX flow hash indirection table for eth0 with 8 RX ring(s):
    0:    0     1     2     3     4     5     6     7
RSS hash key:
84:50:f4:00:a8:15:d1:a7:e9:7f:1d:60:35:c7:47:25:42:97:74:ca:56:bb:b6:a1:d8:43:e3:c9:0
```

netdev_tstamp_prequeue

If set to 0, RX packet timestamps can be sampled after RPS processing, when the target CPU processes packets. It might give some delay on timestamps, but permit to distribute the load on several cpus.

If set to 1 (default), timestamps are sampled as soon as possible, before queueing.

netdev_unregister_timeout_secs

Unregister network device timeout in seconds. This option controls the timeout (in seconds) used to issue a warning while waiting for a network device refcount to drop to 0 during device unregistration. A lower value may be useful during bisection to detect a leaked reference faster. A larger value may be useful to prevent false warnings on slow/loaded systems. Default value is 10, minimum 1, maximum 3600.

skb_defer_max

Max size (in skbs) of the per-cpu list of skbs being freed by the cpu which allocated them. Used by TCP stack so far.

Default: 64

optmem_max

Maximum ancillary buffer size allowed per socket. Ancillary data is a sequence of struct cmsghdr structures with appended data. TCP tx zerocopy also uses optmem_max as a limit for its internal structures.

Default : 128 KB

fb_tunnels_only_for_init_net

Controls if fallback tunnels (like tunl0, gre0, gretap0, erspan0, sit0, ip6tnl0, ip6gre0) are automatically created. There are 3 possibilities (a) value = 0; respective fallback tunnels are created when module is loaded in every net namespaces (backward compatible behavior). (b) value = 1; [kcmd value: initns] respective fallback tunnels are created only in init net namespace and every other net namespace will not have them. (c) value = 2; [kcmd value: none] fallback tunnels are not created when a module is loaded in any of the net namespace. Setting value to "2" is pointless after boot if these modules are built-in, so there is a kernel command-line option that can change this default. Please refer to *The kernel's command-line parameters* for additional details.

Not creating fallback tunnels gives control to userspace to create whatever is needed only and avoid creating devices which are redundant.

Default : 0 (for compatibility reasons)

devconf_inherit_init_net

Controls if a new network namespace should inherit all current settings under /proc/sys/net/{ipv4,ipv6}/conf/{all,default}/. By default, we keep the current behavior: for IPv4 we inherit all current settings from init_net and for IPv6 we reset all settings to default.

If set to 1, both IPv4 and IPv6 settings are forced to inherit from current ones in init_net. If set to 2, both IPv4 and IPv6 settings are forced to reset to their default values. If set to 3, both IPv4 and IPv6 settings are forced to inherit from current ones in the netns where this new netns has been created.

Default : 0 (for compatibility reasons)

txrehash

Controls default hash rethink behaviour on socket when SO_TXREHASH option is set to SOCK_TXREHASH_DEFAULT (i. e. not overridden by setsockopt).

If set to 1 (default), hash rethink is performed on listening socket. If set to 0, hash rethink is not performed.

gro_normal_batch

Maximum number of the segments to batch up on output of GRO. When a packet exits GRO, either as a coalesced superframe or as an original packet which GRO has decided not to coalesce, it is placed on a per-NAPI list. This list is then passed to the stack when the number of segments reaches the gro_normal_batch limit.

high_order_alloc_disable

By default the allocator for page frags tries to use high order pages (order-3 on x86). While the default behavior gives good results in most cases, some users might have hit a contention in page allocations/freeing. This was especially true on older kernels (< 5.14) when high-order pages were not stored on per-cpu lists. This allows to opt-in for order-0 allocation instead but is now mostly of historical importance.

Default: 0

2. /proc/sys/net/unix - Parameters for Unix domain sockets

There is only one file in this directory. unix_dgram_qlen limits the max number of datagrams queued in Unix domain socket's buffer. It will not take effect unless PF_UNIX flag is specified.

3. /proc/sys/net/ipv4 - IPV4 settings

Please see: Documentation/networking/ip-sysctl.rst and *Documentation for /proc/sys/net/* for descriptions of these entries.

4. Appletalk

The /proc/sys/net/appletalk directory holds the Appletalk configuration data when Appletalk is loaded. The configurable parameters are:

aarp-expiry-time

The amount of time we keep an ARP entry before expiring it. Used to age out old hosts.

aarp-resolve-time

The amount of time we will spend trying to resolve an Appletalk address.

aarp-retransmit-limit

The number of times we will retransmit a query before giving up.

aarp-tick-time

Controls the rate at which expires are checked.

The directory /proc/net/appletalk holds the list of active Appletalk sockets on a machine.

The fields indicate the DDP type, the local address (in network:node format) the remote address, the size of the transmit pending queue, the size of the received queue (bytes waiting for applications to read) the state and the uid owning the socket.

/proc/net/atalk_iface lists all the interfaces configured for appletalk.It shows the name of the interface, its Appletalk address, the network range on that address (or network number for phase 1 networks), and the status of the interface.

/proc/net/atalk_route lists each known network route. It lists the target (network) that the route leads to, the router (may be directly connected), the route flags, and the device the route is using.

5. TIPC

tipc_rmem

The TIPC protocol now has a tunable for the receive memory, similar to the tcp_rmem - i.e. a vector of 3 INTEGERs: (min, default, max)

```
# cat /proc/sys/net/tipc/tipc_rmem
4252725 34021800        68043600
#
```

The max value is set to CONN_OVERLOAD_LIMIT, and the default and min values are scaled (shifted) versions of that same value. Note that the min value is not at this point in time used in any meaningful way, but the triplet is preserved in order to be consistent with things like tcp_rmem.

named_timeout

TIPC name table updates are distributed asynchronously in a cluster, without any form of transaction handling. This means that different race scenarios are possible. One such is that a name withdrawal sent out by one node and received by another node may arrive after a second, overlapping name publication already has been accepted from a third node, although the conflicting updates originally may have been issued in the correct sequential order. If named_timeout is nonzero, failed topology updates will be placed on a defer queue until another event arrives that clears the error, or until the timeout expires. Value is in milliseconds.

4.1.5 Documentation for /proc/sys/sunrpc/

kernel version 2.2.10

Copyright (c) 1998, 1999, Rik van Riel <riel@nl.linux.org>

For general info and legal blurb, please look in *The Linux kernel user's and administrator's guide*.

This file contains the documentation for the sysctl files in /proc/sys/sunrpc and is valid for Linux kernel version 2.2.

The files in this directory can be used to (re)set the debug flags of the SUN Remote Procedure Call (RPC) subsystem in the Linux kernel. This stuff is used for NFS, KNFSD and maybe a few other things as well.

The files in there are used to control the debugging flags: rpc_debug, nfs_debug, nfsd_debug and nlm_debug.

These flags are for kernel hackers only. You should read the source code in net/sunrpc/ for more information.

4.1.6 Documentation for /proc/sys/user/

kernel version 4.9.0

Copyright (c) 2016 Eric Biederman <ebiederm@xmission.com>

This file contains the documentation for the sysctl files in /proc/sys/user.

The files in this directory can be used to override the default limits on the number of namespaces and other objects that have per user per user namespace limits.

The primary purpose of these limits is to stop programs that malfunction and attempt to create a ridiculous number of objects, before the malfunction becomes a system wide problem. It is the intention that the defaults of these limits are set high enough that no program in normal operation should run into these limits.

The creation of per user per user namespace objects are charged to the user in the user namespace who created the object and verified to be below the per user limit in that user namespace.

The creation of objects is also charged to all of the users who created user namespaces the creation of the object happens in (user namespaces can be nested) and verified to be below the per user limits in the user namespaces of those users.

This recursive counting of created objects ensures that creating a user namespace does not allow a user to escape their current limits.

Currently, these files are in /proc/sys/user:

max_cgroup_namespaces

The maximum number of cgroup namespaces that any user in the current user namespace may create.

max_ipc_namespaces

The maximum number of ipc namespaces that any user in the current user namespace may create.

max_mnt_namespaces

The maximum number of mount namespaces that any user in the current user namespace may create.

max_net_namespaces

The maximum number of network namespaces that any user in the current user namespace may create.

max_pid_namespaces

The maximum number of pid namespaces that any user in the current user namespace may create.

max_time_namespaces

The maximum number of time namespaces that any user in the current user namespace may create.

max_user_namespaces

The maximum number of user namespaces that any user in the current user namespace may create.

max_uts_namespaces

The maximum number of user namespaces that any user in the current user namespace may create.

4.1.7 Documentation for /proc/sys/vm/

kernel version 2.6.29

Copyright (c) 1998, 1999, Rik van Riel <riel@nl.linux.org>

Copyright (c) 2008 Peter W. Morreale <pmorreale@novell.com>

For general info and legal blurb, please look in *The Linux kernel user's and administrator's guide*.

This file contains the documentation for the sysctl files in /proc/sys/vm and is valid for Linux kernel version 2.6.29.

The files in this directory can be used to tune the operation of the virtual memory (VM) subsystem of the Linux kernel and the writeout of dirty data to disk.

Default values and initialization routines for most of these files can be found in mm/swap.c.

Currently, these files are in /proc/sys/vm:

- admin_reserve_kbytes
- compact_memory
- compaction_proactiveness
- compact_unevictable_allowed
- dirty_background_bytes
- dirty_background_ratio
- dirty_bytes
- dirty_expire_centisecs
- dirty_ratio
- dirtytime_expire_seconds
- dirty_writeback_centisecs
- drop_caches
- extfrag_threshold
- highmem_is_dirtyable
- hugetlb_shm_group
- laptop_mode
- legacy_va_layout
- lowmem_reserve_ratio
- max_map_count
- memory_failure_early_kill
- memory_failure_recovery
- min_free_kbytes

- min_slab_ratio
- min_unmapped_ratio
- mmap_min_addr
- mmap_rnd_bits
- mmap_rnd_compat_bits
- nr_hugepages
- nr_hugepages_mempolicy
- nr_overcommit_hugepages
- nr_trim_pages (only if CONFIG_MMU=n)
- numa_zonelist_order
- oom_dump_tasks
- oom_kill_allocating_task
- overcommit_kbytes
- overcommit_memory
- overcommit_ratio
- page-cluster
- page_lock_unfairness
- panic_on_oom
- percpu_pagelist_high_fraction
- stat_interval
- stat_refresh
- numa_stat
- swappiness
- unprivileged_userfaultfd
- user_reserve_kbytes
- vfs_cache_pressure
- watermark_boost_factor
- watermark_scale_factor
- zone_reclaim_mode

admin_reserve_kbytes

The amount of free memory in the system that should be reserved for users with the capability cap_sys_admin.

admin_reserve_kbytes defaults to min(3% of free pages, 8MB)

That should provide enough for the admin to log in and kill a process, if necessary, under the default overcommit 'guess' mode.

Systems running under overcommit 'never' should increase this to account for the full Virtual Memory Size of programs used to recover. Otherwise, root may not be able to log in to recover the system.

How do you calculate a minimum useful reserve?

sshd or login + bash (or some other shell) + top (or ps, kill, etc.)

For overcommit 'guess', we can sum resident set sizes (RSS). On x86_64 this is about 8MB.

For overcommit 'never', we can take the max of their virtual sizes (VSZ) and add the sum of their RSS. On x86_64 this is about 128MB.

Changing this takes effect whenever an application requests memory.

compact_memory

Available only when CONFIG_COMPACTION is set. When 1 is written to the file, all zones are compacted such that free memory is available in contiguous blocks where possible. This can be important for example in the allocation of huge pages although processes will also directly compact memory as required.

compaction_proactiveness

This tunable takes a value in the range [0, 100] with a default value of 20. This tunable determines how aggressively compaction is done in the background. Write of a non zero value to this tunable will immediately trigger the proactive compaction. Setting it to 0 disables proactive compaction.

Note that compaction has a non-trivial system-wide impact as pages belonging to different processes are moved around, which could also lead to latency spikes in unsuspecting applications. The kernel employs various heuristics to avoid wasting CPU cycles if it detects that proactive compaction is not being effective.

Be careful when setting it to extreme values like 100, as that may cause excessive background compaction activity.

compact_unevictable_allowed

Available only when CONFIG_COMPACTION is set. When set to 1, compaction is allowed to examine the unevictable lru (mlocked pages) for pages to compact. This should be used on systems where stalls for minor page faults are an acceptable trade for large contiguous free memory. Set to 0 to prevent compaction from moving pages that are unevictable. Default value is 1. On CONFIG_PREEMPT_RT the default value is 0 in order to avoid a page fault, due to compaction, which would block the task from becoming active until the fault is resolved.

dirty_background_bytes

Contains the amount of dirty memory at which the background kernel flusher threads will start writeback.

Note:

dirty_background_bytes is the counterpart of dirty_background_ratio. Only one of them may be specified at a time. When one sysctl is written it is immediately taken into account to evaluate the dirty memory limits and the other appears as 0 when read.

dirty_background_ratio

Contains, as a percentage of total available memory that contains free pages and reclaimable pages, the number of pages at which the background kernel flusher threads will start writing out dirty data.

The total available memory is not equal to total system memory.

dirty_bytes

Contains the amount of dirty memory at which a process generating disk writes will itself start writeback.

Note: dirty_bytes is the counterpart of dirty_ratio. Only one of them may be specified at a time. When one sysctl is written it is immediately taken into account to evaluate the dirty memory limits and the other appears as 0 when read.

Note: the minimum value allowed for dirty_bytes is two pages (in bytes); any value lower than this limit will be ignored and the old configuration will be retained.

dirty_expire_centisecs

This tunable is used to define when dirty data is old enough to be eligible for writeout by the kernel flusher threads. It is expressed in 100'ths of a second. Data which has been dirty in-memory for longer than this interval will be written out next time a flusher thread wakes up.

dirty_ratio

Contains, as a percentage of total available memory that contains free pages and reclaimable pages, the number of pages at which a process which is generating disk writes will itself start writing out dirty data.

The total available memory is not equal to total system memory.

dirtytime_expire_seconds

When a lazytime inode is constantly having its pages dirtied, the inode with an updated timestamp will never get chance to be written out. And, if the only thing that has happened on the file system is a dirtytime inode caused by an atime update, a worker will be scheduled to make sure that inode eventually gets pushed out to disk. This tunable is used to define when dirty inode is old enough to be eligible for writeback by the kernel flusher threads. And, it is also used as the interval to wakeup dirtytime_writeback thread.

dirty_writeback_centisecs

The kernel flusher threads will periodically wake up and write *old* data out to disk. This tunable expresses the interval between those wakeups, in 100'ths of a second.

Setting this to zero disables periodic writeback altogether.

drop_caches

Writing to this will cause the kernel to drop clean caches, as well as reclaimable slab objects like dentries and inodes. Once dropped, their memory becomes free.

To free pagecache:

```
echo 1 > /proc/sys/vm/drop_caches
```

To free reclaimable slab objects (includes dentries and inodes):

```
echo 2 > /proc/sys/vm/drop_caches
```

To free slab objects and pagecache:

```
echo 3 > /proc/sys/vm/drop_caches
```

This is a non-destructive operation and will not free any dirty objects. To increase the number of objects freed by this operation, the user may run *sync* prior to writing to /proc/sys/vm/drop_caches. This will minimize the number of dirty objects on the system and create more candidates to be dropped.

This file is not a means to control the growth of the various kernel caches (inodes, dentries, pagecache, etc...) These objects are automatically reclaimed by the kernel when memory is needed elsewhere on the system.

Use of this file can cause performance problems. Since it discards cached objects, it may cost a significant amount of I/O and CPU to recreate the dropped objects, especially if they were

under heavy use. Because of this, use outside of a testing or debugging environment is not recommended.

You may see informational messages in your kernel log when this file is used:

```
cat (1234): drop_caches: 3
```

These are informational only. They do not mean that anything is wrong with your system. To disable them, echo 4 (bit 2) into drop_caches.

extfrag_threshold

This parameter affects whether the kernel will compact memory or direct reclaim to satisfy a high-order allocation. The extfrag/extfrag_index file in debugfs shows what the fragmentation index for each order is in each zone in the system. Values tending towards 0 imply allocations would fail due to lack of memory, values towards 1000 imply failures are due to fragmentation and -1 implies that the allocation will succeed as long as watermarks are met.

The kernel will not compact memory in a zone if the fragmentation index is <= extfrag_threshold. The default value is 500.

highmem_is_dirtyable

Available only for systems with CONFIG_HIGHMEM enabled (32b systems).

This parameter controls whether the high memory is considered for dirty writers throttling. This is not the case by default which means that only the amount of memory directly visible/usable by the kernel can be dirtied. As a result, on systems with a large amount of memory and lowmem basically depleted writers might be throttled too early and streaming writes can get very slow.

Changing the value to non zero would allow more memory to be dirtied and thus allow writers to write more data which can be flushed to the storage more effectively. Note this also comes with a risk of pre-mature OOM killer because some writers (e.g. direct block device writes) can only use the low memory and they can fill it up with dirty data without any throttling.

hugetlb_shm_group

hugetlb_shm_group contains group id that is allowed to create SysV shared memory segment using hugetlb page.

laptop_mode

laptop_mode is a knob that controls ”laptop mode”. All the things that are controlled by this knob are discussed in *How to conserve battery power using laptop-mode*.

legacy_va_layout

If non-zero, this sysctl disables the new 32-bit mmap layout - the kernel will use the legacy (2.4) layout for all processes.

lowmem_reserve_ratio

For some specialised workloads on highmem machines it is dangerous for the kernel to allow process memory to be allocated from the "lowmem" zone. This is because that memory could then be pinned via the mlock() system call, or by unavailability of swapspace.

And on large highmem machines this lack of reclaimable lowmem memory can be fatal.

So the Linux page allocator has a mechanism which prevents allocations which *could* use highmem from using too much lowmem. This means that a certain amount of lowmem is defended from the possibility of being captured into pinned user memory.

(The same argument applies to the old 16 megabyte ISA DMA region. This mechanism will also defend that region from allocations which could use highmem or lowmem).

The *lowmem_reserve_ratio* tunable determines how aggressive the kernel is in defending these lower zones.

If you have a machine which uses highmem or ISA DMA and your applications are using mlock(), or if you are running with no swap then you probably should change the lowmem_reserve_ratio setting.

The lowmem_reserve_ratio is an array. You can see them by reading this file:

```
% cat /proc/sys/vm/lowmem_reserve_ratio
256     256     32
```

But, these values are not used directly. The kernel calculates # of protection pages for each zones from them. These are shown as array of protection pages in /proc/zoneinfo like the following. (This is an example of x86-64 box). Each zone has an array of protection pages like this:

```
Node 0, zone      DMA
  pages free     1355
        min      3
        low      3
        high     4
      :
      :
    numa_other   0
        protection: (0, 2004, 2004, 2004)
      ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  pagesets
    cpu: 0 pcp: 0
        :
```

These protections are added to score to judge whether this zone should be used for page allocation or should be reclaimed.

In this example, if normal pages (index=2) are required to this DMA zone and watermark[WMARK_HIGH] is used for watermark, the kernel judges this zone should not be used because pages_free(1355) is smaller than watermark + protection[2] (4 + 2004 = 2008). If this protection value is 0, this zone would be used for normal page requirement. If requirement is DMA zone(index=0), protection[0] (=0) is used.

zone[i]'s protection[j] is calculated by following expression:

```
(i < j):
  zone[i]->protection[j]
  = (total sums of managed_pages from zone[i+1] to zone[j] on the node)
    / lowmem_reserve_ratio[i];
(i = j):
   (should not be protected. = 0;
(i > j):
   (not necessary, but looks 0)
```

The default values of lowmem_reserve_ratio[i] are

256	(if zone[i] means DMA or DMA32 zone)
32	(others)

As above expression, they are reciprocal number of ratio. 256 means 1/256. # of protection pages becomes about "0.39%" of total managed pages of higher zones on the node.

If you would like to protect more pages, smaller values are effective. The minimum value is 1 (1/1 -> 100%). The value less than 1 completely disables protection of the pages.

max_map_count:

This file contains the maximum number of memory map areas a process may have. Memory map areas are used as a side-effect of calling malloc, directly by mmap, mprotect, and madvise, and also when loading shared libraries.

While most applications need less than a thousand maps, certain programs, particularly malloc debuggers, may consume lots of them, e.g., up to one or two maps per allocation.

The default value is 65530.

memory_failure_early_kill:

Control how to kill processes when uncorrected memory error (typically a 2bit error in a memory module) is detected in the background by hardware that cannot be handled by the kernel. In some cases (like the page still having a valid copy on disk) the kernel will handle the failure transparently without affecting any applications. But if there is no other up-to-date copy of the data it will kill to prevent any data corruptions from propagating.

1: Kill all processes that have the corrupted and not reloadable page mapped as soon as the corruption is detected. Note this is not supported for a few types of pages, like kernel internally allocated data or the swap cache, but works for the majority of user pages.

0: Only unmap the corrupted page from all processes and only kill a process who tries to access it.

The kill is done using a catchable SIGBUS with BUS_MCEERR_AO, so processes can handle this if they want to.

This is only active on architectures/platforms with advanced machine check handling and depends on the hardware capabilities.

Applications can override this setting individually with the PR_MCE_KILL prctl

memory_failure_recovery

Enable memory failure recovery (when supported by the platform)

1: Attempt recovery.

0: Always panic on a memory failure.

min_free_kbytes

This is used to force the Linux VM to keep a minimum number of kilobytes free. The VM uses this number to compute a watermark[WMARK_MIN] value for each lowmem zone in the system. Each lowmem zone gets a number of reserved free pages based proportionally on its size.

Some minimal amount of memory is needed to satisfy PF_MEMALLOC allocations; if you set this to lower than 1024KB, your system will become subtly broken, and prone to deadlock under high loads.

Setting this too high will OOM your machine instantly.

min_slab_ratio

This is available only on NUMA kernels.

A percentage of the total pages in each zone. On Zone reclaim (fallback from the local zone occurs) slabs will be reclaimed if more than this percentage of pages in a zone are reclaimable slab pages. This insures that the slab growth stays under control even in NUMA systems that rarely perform global reclaim.

The default is 5 percent.

Note that slab reclaim is triggered in a per zone / node fashion. The process of reclaiming slab memory is currently not node specific and may not be fast.

min_unmapped_ratio

This is available only on NUMA kernels.

This is a percentage of the total pages in each zone. Zone reclaim will only occur if more than this percentage of pages are in a state that zone_reclaim_mode allows to be reclaimed.

If zone_reclaim_mode has the value 4 OR'd, then the percentage is compared against all file-backed unmapped pages including swapcache pages and tmpfs files. Otherwise, only unmapped pages backed by normal files but not tmpfs files and similar are considered.

The default is 1 percent.

mmap_min_addr

This file indicates the amount of address space which a user process will be restricted from mmapping. Since kernel null dereference bugs could accidentally operate based on the information in the first couple of pages of memory userspace processes should not be allowed to write to them. By default this value is set to 0 and no protections will be enforced by the security module. Setting this value to something like 64k will allow the vast majority of applications to work correctly and provide defense in depth against future potential kernel bugs.

mmap_rnd_bits

This value can be used to select the number of bits to use to determine the random offset to the base address of vma regions resulting from mmap allocations on architectures which support tuning address space randomization. This value will be bounded by the architecture's minimum and maximum supported values.

This value can be changed after boot using the /proc/sys/vm/mmap_rnd_bits tunable

mmap_rnd_compat_bits

This value can be used to select the number of bits to use to determine the random offset to the base address of vma regions resulting from mmap allocations for applications run in compatibility mode on architectures which support tuning address space randomization. This value will be bounded by the architecture's minimum and maximum supported values.

This value can be changed after boot using the /proc/sys/vm/mmap_rnd_compat_bits tunable

nr_hugepages

Change the minimum size of the hugepage pool.

See *HugeTLB Pages*

hugetlb_optimize_vmemmap

This knob is not available when the size of 'struct page' (a structure defined in include/linux/mm_types.h) is not power of two (an unusual system config could result in this).

Enable (set to 1) or disable (set to 0) HugeTLB Vmemmap Optimization (HVO).

Once enabled, the vmemmap pages of subsequent allocation of HugeTLB pages from buddy allocator will be optimized (7 pages per 2MB HugeTLB page and 4095 pages per 1GB HugeTLB page), whereas already allocated HugeTLB pages will not be optimized. When those optimized HugeTLB pages are freed from the HugeTLB pool to the buddy allocator, the vmemmap pages representing that range needs to be remapped again and the vmemmap pages discarded earlier need to be rellocated again. If your use case is that HugeTLB pages are allocated 'on the fly' (e.g. never explicitly allocating HugeTLB pages with 'nr_hugepages' but only set 'nr_overcommit_hugepages', those overcommitted HugeTLB pages are allocated 'on the fly') instead of being pulled from the HugeTLB pool, you should weigh the benefits of memory savings against the more overhead (~2x slower than before) of allocation or freeing HugeTLB pages between the HugeTLB pool and the buddy allocator. Another behavior to note is that if

the system is under heavy memory pressure, it could prevent the user from freeing HugeTLB pages from the HugeTLB pool to the buddy allocator since the allocation of vmemmap pages could be failed, you have to retry later if your system encounter this situation.

Once disabled, the vmemmap pages of subsequent allocation of HugeTLB pages from buddy allocator will not be optimized meaning the extra overhead at allocation time from buddy allocator disappears, whereas already optimized HugeTLB pages will not be affected. If you want to make sure there are no optimized HugeTLB pages, you can set "nr_hugepages" to 0 first and then disable this. Note that writing 0 to nr_hugepages will make any "in use" HugeTLB pages become surplus pages. So, those surplus pages are still optimized until they are no longer in use. You would need to wait for those surplus pages to be released before there are no optimized pages in the system.

nr_hugepages_mempolicy

Change the size of the hugepage pool at run-time on a specific set of NUMA nodes.

See *HugeTLB Pages*

nr_overcommit_hugepages

Change the maximum size of the hugepage pool. The maximum is nr_hugepages + nr_overcommit_hugepages.

See *HugeTLB Pages*

nr_trim_pages

This is available only on NOMMU kernels.

This value adjusts the excess page trimming behaviour of power-of-2 aligned NOMMU mmap allocations.

A value of 0 disables trimming of allocations entirely, while a value of 1 trims excess pages aggressively. Any value >= 1 acts as the watermark where trimming of allocations is initiated.

The default value is 1.

See *No-MMU memory mapping support* for more information.

numa_zonelist_order

This sysctl is only for NUMA and it is deprecated. Anything but Node order will fail!

'where the memory is allocated from' is controlled by zonelists.

(This documentation ignores ZONE_HIGHMEM/ZONE_DMA32 for simple explanation. you may be able to read ZONE_DMA as ZONE_DMA32...)

In non-NUMA case, a zonelist for GFP_KERNEL is ordered as following. ZONE_NORMAL -> ZONE_DMA This means that a memory allocation request for GFP_KERNEL will get memory from ZONE_DMA only when ZONE_NORMAL is not available.

In NUMA case, you can think of following 2 types of order. Assume 2 node NUMA and below is zonelist of Node(0)'s GFP_KERNEL:

```
(A) Node(0) ZONE_NORMAL -> Node(0) ZONE_DMA -> Node(1) ZONE_NORMAL
(B) Node(0) ZONE_NORMAL -> Node(1) ZONE_NORMAL -> Node(0) ZONE_DMA.
```

Type(A) offers the best locality for processes on Node(0), but ZONE_DMA will be used before ZONE_NORMAL exhaustion. This increases possibility of out-of-memory(OOM) of ZONE_DMA because ZONE_DMA is tend to be small.

Type(B) cannot offer the best locality but is more robust against OOM of the DMA zone.

Type(A) is called as ”Node” order. Type (B) is ”Zone” order.

”Node order” orders the zonelists by node, then by zone within each node. Specify ”[Nn]ode” for node order

”Zone Order” orders the zonelists by zone type, then by node within each zone. Specify ”[Zz]one” for zone order.

Specify ”[Dd]efault” to request automatic configuration.

On 32-bit, the Normal zone needs to be preserved for allocations accessible by the kernel, so ”zone” order will be selected.

On 64-bit, devices that require DMA32/DMA are relatively rare, so ”node” order will be selected.

Default order is recommended unless this is causing problems for your system/application.

oom_dump_tasks

Enables a system-wide task dump (excluding kernel threads) to be produced when the kernel performs an OOM-killing and includes such information as pid, uid, tgid, vm size, rss, pgtables_bytes, swapents, oom_score_adj score, and name. This is helpful to determine why the OOM killer was invoked, to identify the rogue task that caused it, and to determine why the OOM killer chose the task it did to kill.

If this is set to zero, this information is suppressed. On very large systems with thousands of tasks it may not be feasible to dump the memory state information for each one. Such systems should not be forced to incur a performance penalty in OOM conditions when the information may not be desired.

If this is set to non-zero, this information is shown whenever the OOM killer actually kills a memory-hogging task.

The default value is 1 (enabled).

oom_kill_allocating_task

This enables or disables killing the OOM-triggering task in out-of-memory situations.

If this is set to zero, the OOM killer will scan through the entire tasklist and select a task based on heuristics to kill. This normally selects a rogue memory-hogging task that frees up a large amount of memory when killed.

If this is set to non-zero, the OOM killer simply kills the task that triggered the out-of-memory condition. This avoids the expensive tasklist scan.

If panic_on_oom is selected, it takes precedence over whatever value is used in oom_kill_allocating_task.

The default value is 0.

overcommit_kbytes

When overcommit_memory is set to 2, the committed address space is not permitted to exceed swap plus this amount of physical RAM. See below.

Note: overcommit_kbytes is the counterpart of overcommit_ratio. Only one of them may be specified at a time. Setting one disables the other (which then appears as 0 when read).

overcommit_memory

This value contains a flag that enables memory overcommitment.

When this flag is 0, the kernel compares the userspace memory request size against total memory plus swap and rejects obvious overcommits.

When this flag is 1, the kernel pretends there is always enough memory until it actually runs out.

When this flag is 2, the kernel uses a ”never overcommit” policy that attempts to prevent any overcommit of memory. Note that user_reserve_kbytes affects this policy.

This feature can be very useful because there are a lot of programs that malloc() huge amounts of memory ”just-in-case” and don't use much of it.

The default value is 0.

See Documentation/mm/overcommit-accounting.rst and mm/util.c::__vm_enough_memory() for more information.

overcommit_ratio

When overcommit_memory is set to 2, the committed address space is not permitted to exceed swap plus this percentage of physical RAM. See above.

page-cluster

page-cluster controls the number of pages up to which consecutive pages are read in from swap in a single attempt. This is the swap counterpart to page cache readahead. The mentioned consecutivity is not in terms of virtual/physical addresses, but consecutive on swap space - that means they were swapped out together.

It is a logarithmic value - setting it to zero means "1 page", setting it to 1 means "2 pages", setting it to 2 means "4 pages", etc. Zero disables swap readahead completely.

The default value is three (eight pages at a time). There may be some small benefits in tuning this to a different value if your workload is swap-intensive.

Lower values mean lower latencies for initial faults, but at the same time extra faults and I/O delays for following faults if they would have been part of that consecutive pages readahead would have brought in.

page_lock_unfairness

This value determines the number of times that the page lock can be stolen from under a waiter. After the lock is stolen the number of times specified in this file (default is 5), the "fair lock handoff" semantics will apply, and the waiter will only be awakened if the lock can be taken.

panic_on_oom

This enables or disables panic on out-of-memory feature.

If this is set to 0, the kernel will kill some rogue process, called oom_killer. Usually, oom_killer can kill rogue processes and system will survive.

If this is set to 1, the kernel panics when out-of-memory happens. However, if a process limits using nodes by mempolicy/cpusets, and those nodes become memory exhaustion status, one process may be killed by oom-killer. No panic occurs in this case. Because other nodes' memory may be free. This means system total status may be not fatal yet.

If this is set to 2, the kernel panics compulsorily even on the above-mentioned. Even oom happens under memory cgroup, the whole system panics.

The default value is 0.

1 and 2 are for failover of clustering. Please select either according to your policy of failover.

panic_on_oom=2+kdump gives you very strong tool to investigate why oom happens. You can get snapshot.

percpu_pagelist_high_fraction

This is the fraction of pages in each zone that are can be stored to per-cpu page lists. It is an upper boundary that is divided depending on the number of online CPUs. The min value for this is 8 which means that we do not allow more than 1/8th of pages in each zone to be stored on per-cpu page lists. This entry only changes the value of hot per-cpu page lists. A user can specify a number like 100 to allocate 1/100th of each zone between per-cpu lists.

The batch value of each per-cpu page list remains the same regardless of the value of the high fraction so allocation latencies are unaffected.

The initial value is zero. Kernel uses this value to set the high pcp->high mark based on the low watermark for the zone and the number of local online CPUs. If the user writes '0' to this sysctl, it will revert to this default behavior.

stat_interval

The time interval between which vm statistics are updated. The default is 1 second.

stat_refresh

Any read or write (by root only) flushes all the per-cpu vm statistics into their global totals, for more accurate reports when testing e.g. cat /proc/sys/vm/stat_refresh /proc/meminfo

As a side-effect, it also checks for negative totals (elsewhere reported as 0) and ”fails” with EINVAL if any are found, with a warning in dmesg. (At time of writing, a few stats are known sometimes to be found negative, with no ill effects: errors and warnings on these stats are suppressed.)

numa_stat

This interface allows runtime configuration of numa statistics.

When page allocation performance becomes a bottleneck and you can tolerate some possible tool breakage and decreased numa counter precision, you can do:

```
echo 0 > /proc/sys/vm/numa_stat
```

When page allocation performance is not a bottleneck and you want all tooling to work, you can do:

```
echo 1 > /proc/sys/vm/numa_stat
```

swappiness

This control is used to define the rough relative IO cost of swapping and filesystem paging, as a value between 0 and 200. At 100, the VM assumes equal IO cost and will thus apply memory pressure to the page cache and swap-backed pages equally; lower values signify more expensive swap IO, higher values indicates cheaper.

Keep in mind that filesystem IO patterns under memory pressure tend to be more efficient than swap's random IO. An optimal value will require experimentation and will also be workload-dependent.

The default value is 60.

For in-memory swap, like zram or zswap, as well as hybrid setups that have swap on faster devices than the filesystem, values beyond 100 can be considered. For example, if the random IO against the swap device is on average 2x faster than IO from the filesystem, swappiness should be 133 (x + 2x = 200, 2x = 133.33).

At 0, the kernel will not initiate swap until the amount of free and file-backed pages is less than the high watermark in a zone.

unprivileged_userfaultfd

This flag controls the mode in which unprivileged users can use the userfaultfd system calls. Set this to 0 to restrict unprivileged users to handle page faults in user mode only. In this case, users without SYS_CAP_PTRACE must pass UFFD_USER_MODE_ONLY in order for userfaultfd to succeed. Prohibiting use of userfaultfd for handling faults from kernel mode may make certain vulnerabilities more difficult to exploit.

Set this to 1 to allow unprivileged users to use the userfaultfd system calls without any restrictions.

The default value is 0.

Another way to control permissions for userfaultfd is to use /dev/userfaultfd instead of userfaultfd(2). See *Userfaultfd*.

user_reserve_kbytes

When overcommit_memory is set to 2, ”never overcommit” mode, reserve min(3% of current process size, user_reserve_kbytes) of free memory. This is intended to prevent a user from starting a single memory hogging process, such that they cannot recover (kill the hog).

user_reserve_kbytes defaults to min(3% of the current process size, 128MB).

If this is reduced to zero, then the user will be allowed to allocate all free memory with a single process, minus admin_reserve_kbytes. Any subsequent attempts to execute a command will result in ”fork: Cannot allocate memory”.

Changing this takes effect whenever an application requests memory.

vfs_cache_pressure

This percentage value controls the tendency of the kernel to reclaim the memory which is used for caching of directory and inode objects.

At the default value of vfs_cache_pressure=100 the kernel will attempt to reclaim dentries and inodes at a "fair" rate with respect to pagecache and swapcache reclaim. Decreasing vfs_cache_pressure causes the kernel to prefer to retain dentry and inode caches. When vfs_cache_pressure=0, the kernel will never reclaim dentries and inodes due to memory pressure and this can easily lead to out-of-memory conditions. Increasing vfs_cache_pressure beyond 100 causes the kernel to prefer to reclaim dentries and inodes.

Increasing vfs_cache_pressure significantly beyond 100 may have negative performance impact. Reclaim code needs to take various locks to find freeable directory and inode objects. With vfs_cache_pressure=1000, it will look for ten times more freeable objects than there are.

watermark_boost_factor

This factor controls the level of reclaim when memory is being fragmented. It defines the percentage of the high watermark of a zone that will be reclaimed if pages of different mobility are being mixed within pageblocks. The intent is that compaction has less work to do in the future and to increase the success rate of future high-order allocations such as SLUB allocations, THP and hugetlbfs pages.

To make it sensible with respect to the watermark_scale_factor parameter, the unit is in fractions of 10,000. The default value of 15,000 means that up to 150% of the high watermark will be reclaimed in the event of a pageblock being mixed due to fragmentation. The level of reclaim is determined by the number of fragmentation events that occurred in the recent past. If this value is smaller than a pageblock then a pageblocks worth of pages will be reclaimed (e.g. 2MB on 64-bit x86). A boost factor of 0 will disable the feature.

watermark_scale_factor

This factor controls the aggressiveness of kswapd. It defines the amount of memory left in a node/system before kswapd is woken up and how much memory needs to be free before kswapd goes back to sleep.

The unit is in fractions of 10,000. The default value of 10 means the distances between watermarks are 0.1% of the available memory in the node/system. The maximum value is 3000, or 30% of memory.

A high rate of threads entering direct reclaim (allocstall) or kswapd going to sleep prematurely (kswapd_low_wmark_hit_quickly) can indicate that the number of free pages kswapd maintains for latency reasons is too small for the allocation bursts occurring in the system. This knob can then be used to tune kswapd aggressiveness accordingly.

zone_reclaim_mode

Zone_reclaim_mode allows someone to set more or less aggressive approaches to reclaim memory when a zone runs out of memory. If it is set to zero then no zone reclaim occurs. Allocations will be satisfied from other zones / nodes in the system.

This is value OR'ed together of

1	Zone reclaim on
2	Zone reclaim writes dirty pages out
4	Zone reclaim swaps pages

zone_reclaim_mode is disabled by default. For file servers or workloads that benefit from having their data cached, zone_reclaim_mode should be left disabled as the caching effect is likely to be more important than data locality.

Consider enabling one or more zone_reclaim mode bits if it's known that the workload is partitioned such that each partition fits within a NUMA node and that accessing remote memory would cause a measurable performance reduction. The page allocator will take additional actions before allocating off node pages.

Allowing zone reclaim to write out pages stops processes that are writing large amounts of data from dirtying pages on other nodes. Zone reclaim will write out dirty pages if a zone fills up and so effectively throttle the process. This may decrease the performance of a single process since it cannot use all of system memory to buffer the outgoing writes anymore but it preserve the memory on other nodes so that the performance of other processes running on other nodes will not be affected.

Allowing regular swap effectively restricts allocations to the local node unless explicitly overridden by memory policies or cpuset configurations.

CHAPTER

FIVE

LINUX ABI DESCRIPTION

5.1 ABI stable symbols

Documents the interfaces that the developer has defined to be stable.

Userspace programs are free to use these interfaces with no restrictions, and backward compatibility for them will be guaranteed for at least 2 years.

Most interfaces (like syscalls) are expected to never change and always be available.

5.1.1 Symbols under /dev/fw

/dev/fw[0-9]+

Defined on file *firewire-cdev*

The character device files /dev/fw* are the interface between firewire-core and IEEE 1394 device drivers implemented in userspace. The ioctl(2)- and read(2)-based ABI is defined and documented in <linux/firewire-cdev.h>.

This ABI offers most of the features which firewire-core also exposes to kernelspace IEEE 1394 drivers.

Each /dev/fw* is associated with one IEEE 1394 node, which can be remote or local nodes. Operations on a /dev/fw* file have different scope:

- The 1394 node which is associated with the file:
  - Asynchronous request transmission
  - Get the Configuration ROM
  - Query node ID
  - Query maximum speed of the path between this node and local node
- The 1394 bus (i.e. "card") to which the node is attached to:
  - Isochronous stream transmission and reception
  - Asynchronous stream transmission and reception
  - Asynchronous broadcast request transmission
  - PHY packet transmission and reception

  - Allocate, reallocate, deallocate isochronous resources (channels, bandwidth) at the bus's IRM
  - Query node IDs of local node, root node, IRM, bus manager
  - Query cycle time
  - Bus reset initiation, bus reset event reception
- All 1394 buses:
  - Allocation of IEEE 1212 address ranges on the local link layers, reception of inbound requests to such an address range, asynchronous response transmission to inbound requests
  - Addition of descriptors or directories to the local nodes' Configuration ROM

Due to the different scope of operations and in order to let userland implement different access permission models, some operations are restricted to /dev/fw* files that are associated with a local node:

- Addition of descriptors or directories to the local nodes' Configuration ROM
- PHY packet transmission and reception

A /dev/fw* file remains associated with one particular node during its entire life time. Bus topology changes, and hence node ID changes, are tracked by firewire-core. ABI users do not need to be aware of topology.

The following file operations are supported:

open(2)
Currently the only useful flags are O_RDWR.

ioctl(2)
Initiate various actions. Some take immediate effect, others are performed asynchronously while or after the ioctl returns. See the inline documentation in <linux/firewire-cdev.h> for descriptions of all ioctls.

poll(2), select(2), epoll_wait(2) etc.
Watch for events to become available to be read.

read(2)
Receive various events. There are solicited events like outbound asynchronous transaction completion or isochronous buffer completion, and unsolicited events such as bus resets, request reception, or PHY packet reception. Always use a read buffer which is large enough to receive the largest event that could ever arrive. See <linux/firewire-cdev.h> for descriptions of all event types and for which ioctls affect reception of events.

mmap(2)
Allocate a DMA buffer for isochronous reception or transmission and map it into the process address space. The arguments should be used as follows: addr = NULL, length = the desired buffer size, i.e. number of packets times size of largest packet, prot = at least PROT_READ for reception and at least PROT_WRITE for transmission, flags = MAP_SHARED, fd = the handle to the /dev/fw*, offset = 0.

Isochronous reception works in packet-per-buffer fashion except for multichannel reception which works in buffer-fill mode.

munmap(2)
Unmap the isochronous I/O buffer from the process address space.

close(2)
Besides stopping and freeing I/O contexts that were associated with the file descriptor, back out any changes to the local nodes' Configuration ROM. Deallocate isochronous channels and bandwidth at the IRM that were marked for kernel-assisted re- and deallocation.

Users: libraw1394; libdc1394; libhinawa; tools like linux-firewire-utils, fwhack, ...

5.1.2 Symbols under /sys/accessibility

/sys/accessibility/speakup/<synth-name>/

Defined on file *sysfs-driver-speakup*

In */sys/accessibility/speakup* is a directory corresponding to the synthesizer driver currently in use (E.G) *soft* for the soft driver. This directory contains files which control the speech synthesizer itself, as opposed to controlling the speakup screen reader. The parameters in this directory have the same names and functions across all supported synthesizers. The range of values for freq, pitch, rate, and vol is the same for all supported synthesizers, with the given range being internally mapped by the driver to more or less fit the range of values supported for a given parameter by the individual synthesizer. Below is a description of values and parameters for soft synthesizer, which is currently the most commonly used.

/sys/accessibility/speakup/<synth-name>/caps_start

Defined on file *sysfs-driver-speakup*

This is the string that is sent to the synthesizer to cause it to start speaking uppercase letters. For the soft synthesizer and most others, this causes the pitch of the voice to rise above the currently set pitch.

/sys/accessibility/speakup/<synth-name>/caps_stop

Defined on file *sysfs-driver-speakup*

This is the string sent to the synthesizer to cause it to stop speaking uppercase letters. In the case of the soft synthesizer and most others, this returns the pitch of the voice down to the currently set pitch.

/sys/accessibility/speakup/<synth-name>/delay_time

Defined on file *sysfs-driver-speakup*

TODO:

/sys/accessibility/speakup/<synth-name>/direct

Defined on file *sysfs-driver-speakup*

Controls if punctuation is spoken by speakup, or by the synthesizer.

For example, speakup speaks ”>” as ”greater”, while the espeak synthesizer used by the soft driver speaks ”greater than”. Zero lets speakup speak the punctuation. One lets the synthesizer itself speak punctuation.

/sys/accessibility/speakup/<synth-name>/flush_time

Defined on file *sysfs-driver-speakup*

Gets or sets the timeout to wait for the synthesizer flush to complete. This can be used when the cable gets faulty and flush notifications are getting lost.

/sys/accessibility/speakup/<synth-name>/freq

Defined on file *sysfs-driver-speakup*

Gets or sets the frequency of the speech synthesizer. Range is 0-9.

/sys/accessibility/speakup/<synth-name>/full_time

Defined on file *sysfs-driver-speakup*

TODO:

/sys/accessibility/speakup/<synth-name>/inflection

Defined on file *sysfs-driver-speakup*

Gets or sets the inflection of the synthesizer, i.e. the pitch range. The range is 0-9.

/sys/accessibility/speakup/<synth-name>/jiffy_delta

Defined on file *sysfs-driver-speakup*

This controls how many jiffys the kernel gives to the synthesizer. Setting this too high can make a system unstable, or even crash it.

/sys/accessibility/speakup/<synth-name>/pitch

Defined on file *sysfs-driver-speakup*

Gets or sets the pitch of the synthesizer. The range is 0-9.

/sys/accessibility/speakup/<synth-name>/punct

Defined on file *sysfs-driver-speakup*

Gets or sets the amount of punctuation spoken by the synthesizer. The range for the soft driver seems to be 0-2. TODO: How is this related to speakup's punc_level, or reading_punc.

/sys/accessibility/speakup/<synth-name>/rate

Defined on file *sysfs-driver-speakup*

Gets or sets the rate of the synthesizer. Range is from zero slowest, to nine fastest.

/sys/accessibility/speakup/<synth-name>/tone

Defined on file *sysfs-driver-speakup*

Gets or sets the tone of the speech synthesizer. The range for the soft driver seems to be 0-2. This seems to make no difference if using espeak and the espeakup connector. TODO: does espeakup support different tonalities?

/sys/accessibility/speakup/<synth-name>/trigger_time

Defined on file *sysfs-driver-speakup*

TODO:

/sys/accessibility/speakup/<synth-name>/voice

Defined on file *sysfs-driver-speakup*

Gets or sets the voice used by the synthesizer if the synthesizer can speak in more than one voice. The range for the soft driver is 0-7. Note that while espeak supports multiple voices, this parameter will not set the voice when the espeakup connector is used between speakup and espeak.

/sys/accessibility/speakup/<synth-name>/vol

Defined on file *sysfs-driver-speakup*

Gets or sets the volume of the speech synthesizer. Range is 0-9, with zero being the softest, and nine being the loudest.

/sys/accessibility/speakup/attrib_bleep

Defined on file *sysfs-driver-speakup*

Beeps the PC speaker when there is an attribute change such as foreground or background color when using speakup review commands. One = on, zero = off.

/sys/accessibility/speakup/bell_pos

Defined on file *sysfs-driver-speakup*

This works much like a typewriter bell. If for example 72 is echoed to bell_pos, it will beep the PC speaker when typing on a line past character 72.

/sys/accessibility/speakup/bleep_time

Defined on file *sysfs-driver-speakup*

This controls the duration of the PC speaker beeps speakup produces. TODO: What are the units? Jiffies?

/sys/accessibility/speakup/bleeps

Defined on file *sysfs-driver-speakup*

This controls whether one hears beeps through the PC speaker when using speakup's review commands. TODO: what values does it accept?

/sys/accessibility/speakup/cur_phonetic

Defined on file *sysfs-driver-speakup*

This allows speakup to speak letters phoneticaly when arrowing through a word letter by letter. This doesn't affect the spelling when typing the characters. When cur_phonetic=1, speakup will speak characters phoneticaly when arrowing over a letter. When cur_phonetic=0, speakup will speak letters as normally.

/sys/accessibility/speakup/cursor_time

Defined on file *sysfs-driver-speakup*

This controls cursor delay when using arrow keys. When a connection is very slow, with the default setting, when moving with the arrows, or backspacing etc. speakup says the incorrect characters. Set this to a higher value to adjust for the delay and better synchronisation between cursor position and speech.

/sys/accessibility/speakup/delimiters

Defined on file *sysfs-driver-speakup*

Delimit a word from speakup. TODO: add more info

/sys/accessibility/speakup/ex_num

Defined on file *sysfs-driver-speakup*

TODO:

/sys/accessibility/speakup/i18n/announcements

Defined on file *sysfs-driver-speakup*

This file contains various general announcements, most of which cannot be categorized. You will find messages such as ”You killed Speakup”, ”I'm alive”, ”leaving help”, ”parked”, ”unparked”, and others. You will also find the names of the screen edges and cursor tracking modes here.

/sys/accessibility/speakup/i18n/characters

Defined on file *sysfs-driver-speakup*

Through this sys entry, Speakup gives you the ability to change how Speakup pronounces a given character. You could, for example, change how some punctuation characters are spoken. You can even change how Speakup will pronounce certain letters. For further details see '12. Changing the Pronunciation of Characters' in Speakup User's Guide (file spkguide.txt in source).

/sys/accessibility/speakup/i18n/chartab

Defined on file *sysfs-driver-speakup*

TODO

/sys/accessibility/speakup/i18n/colors

Defined on file *sysfs-driver-speakup*

When you use the "say attributes" function, Speakup says the name of the foreground and background colors. These names come from the i18n/colors file.

/sys/accessibility/speakup/i18n/ctl_keys

Defined on file *sysfs-driver-speakup*

Here, you will find names of control keys. These are used with Speakup's say_control feature.

/sys/accessibility/speakup/i18n/formatted

Defined on file *sysfs-driver-speakup*

This group of messages contains embedded formatting codes, to specify the type and width of displayed data. If you change these, you must preserve all of the formatting codes, and they must appear in the order used by the default messages.

/sys/accessibility/speakup/i18n/function_names

Defined on file *sysfs-driver-speakup*

Here, you will find a list of names for Speakup functions. These are used by the help system. For example, suppose that you have activated help mode, and you pressed keypad 3. Speakup says: "keypad 3 is character, say next." The message "character, say next" names a Speakup function, and it comes from this function_names file.

/sys/accessibility/speakup/i18n/key_names

Defined on file *sysfs-driver-speakup*

Again, key_names is used by Speakup's help system. In the previous example, Speakup said that you pressed "keypad 3." This name came from the key_names file.

/sys/accessibility/speakup/i18n/states

Defined on file *sysfs-driver-speakup*

This file contains names for key states. Again, these are part of the help system. For instance, if you had pressed speakup + keypad 3, you would hear: "speakup keypad 3 is go to bottom edge."

The speakup key is depressed, so the name of the key state is speakup.

This part of the message comes from the states collection.

/sys/accessibility/speakup/key_echo

Defined on file *sysfs-driver-speakup*

Controls if speakup speaks keys when they are typed. One = on, zero = off or don't echo keys.

/sys/accessibility/speakup/keymap

Defined on file *sysfs-driver-speakup*

Speakup keymap remaps keys to Speakup functions. It uses a binary format. A special program called genmap is needed to compile a textual keymap into the binary format which is then loaded into */sys/accessibility/speakup/keymap*.

/sys/accessibility/speakup/no_interrupt

Defined on file *sysfs-driver-speakup*

Controls if typing interrupts output from speakup. With no_interrupt set to zero, typing on the keyboard will interrupt speakup if for example the say screen command is used before the entire screen is read.

With no_interrupt set to one, if the say screen command is used, and one then types on the keyboard, speakup will continue to say the whole screen regardless until it finishes.

/sys/accessibility/speakup/punc_all

Defined on file *sysfs-driver-speakup*

This is a list of all the punctuation speakup should speak when punc_level is set to four.

/sys/accessibility/speakup/punc_level

Defined on file *sysfs-driver-speakup*

Controls the level of punctuation spoken as the screen is displayed, not reviewed. Levels range from zero no punctuation, to four, all punctuation. One corresponds to punc_some, two corresponds to punc_most, and three as well as four both correspond to punc_all. Some hardware synthesizers may have different levels each corresponding to three and four for punc_level. Also note that if punc_level is set to zero, and key_echo is set to one, typed punctuation is still spoken as it is typed.

/sys/accessibility/speakup/punc_most

Defined on file *sysfs-driver-speakup*

This is a list of all the punctuation speakup should speak when punc_level is set to two.

/sys/accessibility/speakup/punc_some

Defined on file *sysfs-driver-speakup*

This is a list of all the punctuation speakup should speak when punc_level is set to one.

/sys/accessibility/speakup/reading_punc

Defined on file *sysfs-driver-speakup*

Almost the same as punc_level, the differences being that reading_punc controls the level of punctuation when reviewing the screen with speakup's screen review commands. The other difference is that reading_punc set to three speaks punc_all, and reading_punc set to four speaks all punctuation, including spaces.

/sys/accessibility/speakup/repeats

Defined on file *sysfs-driver-speakup*

A list of characters speakup repeats. Normally, when there are more than three characters in a row, speakup just reads three of those characters. For example, "....." would be read as dot, dot, dot. If a . is added to the list of characters in repeats, "....." would be read as dot, dot, dot, times six.

/sys/accessibility/speakup/say_control

Defined on file *sysfs-driver-speakup*

If set to one, speakup speaks shift, alt and control when those keys are pressed. If say_control is set to zero, shift, ctrl, and alt are not spoken when they are pressed.

/sys/accessibility/speakup/say_word_ctl

Defined on file *sysfs-driver-speakup*

TODO:

/sys/accessibility/speakup/silent

Defined on file *sysfs-driver-speakup*

TODO:

/sys/accessibility/speakup/spell_delay

Defined on file *sysfs-driver-speakup*

This controls how fast a word is spelled when speakup's say word review command is pressed twice quickly to speak the current word being reviewed. Zero just speaks the letters one after

another, while values one through four seem to introduce more of a pause between the spelling of each letter by speakup.

/sys/accessibility/speakup/synth

Defined on file *sysfs-driver-speakup*

Gets or sets the synthesizer driver currently in use. Reading synth returns the synthesizer driver currently in use. Writing synth switches to the given synthesizer driver, provided it is either built into the kernel, or already loaded as a module.

/sys/accessibility/speakup/synth_direct

Defined on file *sysfs-driver-speakup*

Sends whatever is written to synth_direct directly to the speech synthesizer in use, bypassing speakup. This could be used to make the synthesizer speak a string, or to send control sequences to the synthesizer to change how the synthesizer behaves.

/sys/accessibility/speakup/version

Defined on file *sysfs-driver-speakup*

Reading version returns the version of speakup, and the version of the synthesizer driver currently in use.

5.1.3 Symbols under /sys/block

/sys/block/<disk>/<partition>/alignment_offset

Defined on file *sysfs-block*

Storage devices may report a physical block size that is bigger than the logical block size (for instance a drive with 4KB physical sectors exposing 512-byte logical blocks to the operating system). This parameter indicates how many bytes the beginning of the partition is offset from the disk's natural alignment.

/sys/block/<disk>/<partition>/discard_alignment

Defined on file *sysfs-block*

Devices that support discard functionality may internally allocate space in units that are bigger than the exported logical block size. The discard_alignment parameter indicates how many bytes the beginning of the partition is offset from the internal allocation unit's natural alignment.

/sys/block/<disk>/<partition>/stat

Defined on file *sysfs-block*

The */sys/block/<disk>/<partition>/stat* files display the I/O statistics of partition <partition>. The format is the same as the format of */sys/block/<disk>/stat*.

/sys/block/<disk>/alignment_offset

Defined on file *sysfs-block*

Storage devices may report a physical block size that is bigger than the logical block size (for instance a drive with 4KB physical sectors exposing 512-byte logical blocks to the operating system). This parameter indicates how many bytes the beginning of the device is offset from the disk's natural alignment.

/sys/block/<disk>/discard_alignment

Defined on file *sysfs-block*

Devices that support discard functionality may internally allocate space in units that are bigger than the exported logical block size. The discard_alignment parameter indicates how many bytes the beginning of the device is offset from the internal allocation unit's natural alignment.

/sys/block/<disk>/diskseq

Defined on file *sysfs-block*

The */sys/block/<disk>/diskseq* files reports the disk sequence number, which is a monotonically increasing number assigned to every drive. Some devices, like the loop device, refresh such number every time the backing file is changed. The value type is 64 bit unsigned.

/sys/block/<disk>/hidden

Defined on file *sysfs-block*

[RO] the block device is hidden. it doesn't produce events, and can't be opened from userspace or using blkdev_get*. Used for the underlying components of multipath devices.

/sys/block/<disk>/inflight

Defined on file *sysfs-block*

Reports the number of I/O requests currently in progress (pending / in flight) in a device driver. This can be less than the number of requests queued in the block device queue. The report contains 2 fields: one for read requests and one for write requests. The value type is unsigned int. Cf. /block/stat which contains a single value for requests in flight. This is related to */sys/block/<disk>/queue/nr_requests* and for SCSI device also its queue_depth.

/sys/block/<disk>/integrity/device_is_integrity_capable

Defined on file *sysfs-block*

Indicates whether a storage device is capable of storing integrity metadata. Set if the device is T10 PI-capable.

/sys/block/<disk>/integrity/format

Defined on file *sysfs-block*

Metadata format for integrity capable block device. E.g. T10-DIF-TYPE1-CRC.

/sys/block/<disk>/integrity/protection_interval_bytes

Defined on file *sysfs-block*

Describes the number of data bytes which are protected by one integrity tuple. Typically the device's logical block size.

/sys/block/<disk>/integrity/read_verify

Defined on file *sysfs-block*

Indicates whether the block layer should verify the integrity of read requests serviced by devices that support sending integrity metadata.

/sys/block/<disk>/integrity/tag_size

Defined on file *sysfs-block*

Number of bytes of integrity tag space available per 512 bytes of data.

/sys/block/<disk>/integrity/write_generate

Defined on file *sysfs-block*

Indicates whether the block layer should automatically generate checksums for write requests bound for devices that support receiving integrity metadata.

/sys/block/<disk>/queue/add_random

Defined on file *sysfs-block*

[RW] This file allows to turn off the disk entropy contribution. Default value of this file is '1'(on).

/sys/block/<disk>/queue/chunk_sectors

Defined on file *sysfs-block*

[RO] chunk_sectors has different meaning depending on the type of the disk. For a RAID device (dm-raid), chunk_sectors indicates the size in 512B sectors of the RAID volume stripe segment. For a zoned block device, either host-aware or host-managed, chunk_sectors indicates the size in 512B sectors of the zones of the device, with the eventual exception of the last zone of the device which may be smaller.

/sys/block/<disk>/queue/crypto/

Defined on file *sysfs-block*

The presence of this subdirectory of /sys/block/<disk>/queue/ indicates that the device supports inline encryption. This subdirectory contains files which describe the inline encryption capabilities of the device. For more information about inline encryption, refer to /block/inline-encryption.

/sys/block/<disk>/queue/crypto/max_dun_bits

Defined on file *sysfs-block*

[RO] This file shows the maximum length, in bits, of data unit numbers accepted by the device in inline encryption requests.

/sys/block/<disk>/queue/crypto/modes/<mode>

Defined on file *sysfs-block*

[RO] For each crypto mode (i.e., encryption/decryption algorithm) the device supports with inline encryption, a file will exist at this location. It will contain a hexadecimal number that is a bitmask of the supported data unit sizes, in bytes, for that crypto mode.

Currently, the crypto modes that may be supported are:

- AES-256-XTS
- AES-128-CBC-ESSIV
- Adiantum

For example, if a device supports AES-256-XTS inline encryption with data unit sizes of 512 and 4096 bytes, the file /sys/block/<disk>/queue/crypto/modes/AES-256-XTS will exist and will contain "0x1200".

/sys/block/<disk>/queue/crypto/num_keyslots

Defined on file *sysfs-block*

[RO] This file shows the number of keyslots the device has for use with inline encryption.

/sys/block/<disk>/queue/dax

Defined on file *sysfs-block*

[RO] This file indicates whether the device supports Direct Access (DAX), used by CPU-addressable storage to bypass the pagecache. It shows '1' if true, '0' if not.

/sys/block/<disk>/queue/discard_granularity

Defined on file *sysfs-block*

[RO] Devices that support discard functionality may internally allocate space using units that are bigger than the logical block size. The discard_granularity parameter indicates the size of the internal allocation unit in bytes if reported by the device. Otherwise the discard_granularity will be set to match the device's physical block size. A discard_granularity of 0 means that the device does not support discard functionality.

/sys/block/<disk>/queue/discard_max_bytes

Defined on file *sysfs-block*

[RW] While discard_max_hw_bytes is the hardware limit for the device, this setting is the software limit. Some devices exhibit large latencies when large discards are issued, setting this value lower will make Linux issue smaller discards and potentially help reduce latencies induced by large discard operations.

/sys/block/<disk>/queue/discard_max_hw_bytes

Defined on file *sysfs-block*

[RO] Devices that support discard functionality may have internal limits on the number of bytes that can be trimmed or unmapped in a single operation. The *discard_max_hw_bytes* parameter is set by the device driver to the maximum number of bytes that can be discarded in a single operation. Discard requests issued to the device must not exceed this limit. A *discard_max_hw_bytes* value of 0 means that the device does not support discard functionality.

/sys/block/<disk>/queue/discard_zeroes_data

Defined on file *sysfs-block*

[RO] Will always return 0. Don't rely on any specific behavior for discards, and don't read this file.

/sys/block/<disk>/queue/dma_alignment

Defined on file *sysfs-block*

Reports the alignment that user space addresses must have to be used for raw block device access with O_DIRECT and other driver specific passthrough mechanisms.

/sys/block/<disk>/queue/fua

Defined on file *sysfs-block*

[RO] Whether or not the block driver supports the FUA flag for write requests. FUA stands for Force Unit Access. If the FUA flag is set that means that write requests must bypass the volatile cache of the storage device.

/sys/block/<disk>/queue/hw_sector_size

Defined on file *sysfs-block*

[RO] This is the hardware sector size of the device, in bytes.

/sys/block/<disk>/queue/independent_access_ranges/

Defined on file *sysfs-block*

[RO] The presence of this sub-directory of the /sys/block/xxx/queue/ directory indicates that the device is capable of executing requests targeting different sector ranges in parallel. For instance, single LUN multi-actuator hard-disks will have an independent_access_ranges directory if the device correctly advertises the sector ranges of its actuators.

The independent_access_ranges directory contains one directory per access range, with each range described using the sector (RO) attribute file to indicate the first sector of the range and the nr_sectors (RO) attribute file to indicate the total number of sectors in the range starting from the first sector of the range. For example, a dual-actuator hard-disk will have the following independent_access_ranges entries.:

```
$ tree /sys/block/<disk>/queue/independent_access_ranges/
/sys/block/<disk>/queue/independent_access_ranges/
|-- 0
|   |-- nr_sectors
|   `-- sector
`-- 1
    |-- nr_sectors
    `-- sector
```

The sector and nr_sectors attributes use 512B sector unit, regardless of the actual block size of the device. Independent access ranges do not overlap and include all sectors within the device capacity. The access ranges are numbered in increasing order of the range start sector, that is, the sector attribute of range 0 always has the value 0.

/sys/block/<disk>/queue/io_poll

Defined on file *sysfs-block*

[RW] When read, this file shows whether polling is enabled (1) or disabled (0). Writing '0' to this file will disable polling for this device. Writing any non-zero value will enable this feature.

/sys/block/<disk>/queue/io_poll_delay

Defined on file *sysfs-block*

[RW] This was used to control what kind of polling will be performed. It is now fixed to -1, which is classic polling. In this mode, the CPU will repeatedly ask for completions without giving up any time. <deprecated>

/sys/block/<disk>/queue/io_timeout

Defined on file *sysfs-block*

[RW] io_timeout is the request timeout in milliseconds. If a request does not complete in this time then the block driver timeout handler is invoked. That timeout handler can decide to retry the request, to fail it or to start a device recovery strategy.

/sys/block/<disk>/queue/iostats

Defined on file *sysfs-block*

[RW] This file is used to control (on/off) the iostats accounting of the disk.

/sys/block/<disk>/queue/logical_block_size

Defined on file *sysfs-block*

[RO] This is the smallest unit the storage device can address. It is typically 512 bytes.

/sys/block/<disk>/queue/max_active_zones

Defined on file *sysfs-block*

[RO] For zoned block devices (zoned attribute indicating "host-managed" or "host-aware"), the sum of zones belonging to any of the zone states: EXPLICIT OPEN, IMPLICIT OPEN or CLOSED, is limited by this value. If this value is 0, there is no limit.

If the host attempts to exceed this limit, the driver should report this error with BLK_STS_ZONE_ACTIVE_RESOURCE, which user space may see as the EOVERFLOW errno.

/sys/block/<disk>/queue/max_discard_segments

Defined on file *sysfs-block*

[RO] The maximum number of DMA scatter/gather entries in a discard request.

/sys/block/<disk>/queue/max_hw_sectors_kb

Defined on file *sysfs-block*

[RO] This is the maximum number of kilobytes supported in a single data transfer.

/sys/block/<disk>/queue/max_integrity_segments

Defined on file *sysfs-block*

[RO] Maximum number of elements in a DMA scatter/gather list with integrity data that will be submitted by the block layer core to the associated block driver.

/sys/block/<disk>/queue/max_open_zones

Defined on file *sysfs-block*

[RO] For zoned block devices (zoned attribute indicating "host-managed" or "host-aware"), the sum of zones belonging to any of the zone states: EXPLICIT OPEN or IMPLICIT OPEN, is limited by this value. If this value is 0, there is no limit.

/sys/block/<disk>/queue/max_sectors_kb

Defined on file *sysfs-block*

[RW] This is the maximum number of kilobytes that the block layer will allow for a filesystem request. Must be smaller than or equal to the maximum size allowed by the hardware. Write 0 to use default kernel settings.

/sys/block/<disk>/queue/max_segment_size

Defined on file *sysfs-block*

[RO] Maximum size in bytes of a single element in a DMA scatter/gather list.

/sys/block/<disk>/queue/max_segments

Defined on file *sysfs-block*

[RO] Maximum number of elements in a DMA scatter/gather list that is submitted to the associated block driver.

/sys/block/<disk>/queue/minimum_io_size

Defined on file *sysfs-block*

[RO] Storage devices may report a granularity or preferred minimum I/O size which is the smallest request the device can perform without incurring a performance penalty. For disk drives this is often the physical block size. For RAID arrays it is often the stripe chunk size. A properly aligned multiple of minimum_io_size is the preferred request size for workloads where a high number of I/O operations is desired.

/sys/block/<disk>/queue/nomerges

Defined on file *sysfs-block*

[RW] Standard I/O elevator operations include attempts to merge contiguous I/Os. For known random I/O loads these attempts will always fail and result in extra cycles being spent in the kernel. This allows one to turn off this behavior on one of two ways: When set to 1, complex merge checks are disabled, but the simple one-shot merges with the previous I/O request are enabled. When set to 2, all merge tries are disabled. The default value is 0 - which enables all types of merge tries.

/sys/block/<disk>/queue/nr_requests

Defined on file *sysfs-block*

[RW] This controls how many requests may be allocated in the block layer for read or write requests. Note that the total allocated number may be twice this amount, since it applies only to reads or writes (not the accumulated sum).

To avoid priority inversion through request starvation, a request queue maintains a separate request pool per each cgroup when CONFIG_BLK_CGROUP is enabled, and this parameter applies to each such per-block-cgroup request pool. IOW, if there are N block cgroups, each request queue may have up to N request pools, each independently regulated by nr_requests.

/sys/block/<disk>/queue/nr_zones

Defined on file *sysfs-block*

[RO] nr_zones indicates the total number of zones of a zoned block device (”host-aware” or ”host-managed” zone model). For regular block devices, the value is always 0.

/sys/block/<disk>/queue/optimal_io_size

Defined on file *sysfs-block*

[RO] Storage devices may report an optimal I/O size, which is the device's preferred unit for sustained I/O. This is rarely reported for disk drives. For RAID arrays it is usually the stripe width or the internal track size. A properly aligned multiple of optimal_io_size is the preferred request size for workloads where sustained throughput is desired. If no optimal I/O size is reported this file contains 0.

/sys/block/<disk>/queue/physical_block_size

Defined on file *sysfs-block*

[RO] This is the smallest unit a physical storage device can write atomically. It is usually the same as the logical block size but may be bigger. One example is SATA drives with 4KB sectors that expose a 512-byte logical block size to the operating system. For stacked block devices the physical_block_size variable contains the maximum physical_block_size of the component devices.

/sys/block/<disk>/queue/read_ahead_kb

Defined on file *sysfs-block*

[RW] Maximum number of kilobytes to read-ahead for filesystems on this block device.

/sys/block/<disk>/queue/rotational

Defined on file *sysfs-block*

[RW] This file is used to stat if the device is of rotational type or non-rotational type.

/sys/block/<disk>/queue/rq_affinity

Defined on file *sysfs-block*

[RW] If this option is '1', the block layer will migrate request completions to the cpu ”group” that originally submitted the request. For some workloads this provides a significant reduction in CPU cycles due to caching effects.

For storage configurations that need to maximize distribution of completion processing setting this option to '2' forces the completion to run on the requesting cpu (bypassing the ”group” aggregation logic).

/sys/block/<disk>/queue/scheduler

Defined on file *sysfs-block*

[RW] When read, this file will display the current and available IO schedulers for this block device. The currently active IO scheduler will be enclosed in [] brackets. Writing an IO scheduler name to this file will switch control of this block device to that new IO scheduler. Note that writing an IO scheduler name to this file will attempt to load that IO scheduler module, if it isn't already present in the system.

/sys/block/<disk>/queue/stable_writes

Defined on file *sysfs-block*

[RW] This file will contain '1' if memory must not be modified while it is being used in a write request to this device. When this is the case and the kernel is performing writeback of a page, the kernel will wait for writeback to complete before allowing the page to be modified again, rather than allowing immediate modification as is normally the case. This restriction arises when the device accesses the memory multiple times where the same data must be seen every time -- for example, once to calculate a checksum and once to actually write the data. If no such restriction exists, this file will contain '0'. This file is writable for testing purposes.

/sys/block/<disk>/queue/throttle_sample_time

Defined on file *sysfs-block*

[RW] This is the time window that blk-throttle samples data, in millisecond. blk-throttle makes decision based on the samplings. Lower time means cgroups have more smooth throughput, but higher CPU overhead. This exists only when CONFIG_BLK_DEV_THROTTLING_LOW is enabled.

/sys/block/<disk>/queue/virt_boundary_mask

Defined on file *sysfs-block*

[RO] This file shows the I/O segment memory alignment mask for the block device. I/O requests to this device will be split between segments wherever either the memory address of the end of the previous segment or the memory address of the beginning of the current segment is not aligned to virt_boundary_mask + 1 bytes.

/sys/block/<disk>/queue/wbt_lat_usec

Defined on file *sysfs-block*

[RW] If the device is registered for writeback throttling, then this file shows the target minimum read latency. If this latency is exceeded in a given window of time (see wb_window_usec), then the writeback throttling will start scaling back writes. Writing a value of '0' to this file disables the feature. Writing a value of '-1' to this file resets the value to the default setting.

/sys/block/<disk>/queue/write_cache

Defined on file *sysfs-block*

[RW] When read, this file will display whether the device has write back caching enabled or not. It will return ”write back” for the former case, and ”write through” for the latter. Writing to this file can change the kernels view of the device, but it doesn't alter the device state. This

means that it might not be safe to toggle the setting from "write back" to "write through", since that will also eliminate cache flushes issued by the kernel.

/sys/block/<disk>/queue/write_same_max_bytes

Defined on file *sysfs-block*

[RO] Some devices support a write same operation in which a single data block can be written to a range of several contiguous blocks on storage. This can be used to wipe areas on disk or to initialize drives in a RAID configuration. write_same_max_bytes indicates how many bytes can be written in a single write same command. If write_same_max_bytes is 0, write same is not supported by the device.

/sys/block/<disk>/queue/write_zeroes_max_bytes

Defined on file *sysfs-block*

[RO] Devices that support write zeroes operation in which a single request can be issued to zero out the range of contiguous blocks on storage without having any payload in the request. This can be used to optimize writing zeroes to the devices. write_zeroes_max_bytes indicates how many bytes can be written in a single write zeroes command. If write_zeroes_max_bytes is 0, write zeroes is not supported by the device.

/sys/block/<disk>/queue/zone_append_max_bytes

Defined on file *sysfs-block*

[RO] This is the maximum number of bytes that can be written to a sequential zone of a zoned block device using a zone append write operation (REQ_OP_ZONE_APPEND). This value is always 0 for regular block devices.

/sys/block/<disk>/queue/zone_write_granularity

Defined on file *sysfs-block*

[RO] This indicates the alignment constraint, in bytes, for write operations in sequential zones of zoned block devices (devices with a zoned attributed that reports "host-managed" or "host-aware"). This value is always 0 for regular block devices.

/sys/block/<disk>/queue/zoned

Defined on file *sysfs-block*

[RO] zoned indicates if the device is a zoned block device and the zone model of the device if it is indeed zoned. The possible values indicated by zoned are "none" for regular block devices and "host-aware" or "host-managed" for zoned block devices. The characteristics of host-aware and host-managed zoned block devices are described in the ZBC (Zoned Block Commands) and ZAC (Zoned Device ATA Command Set) standards. These standards also define the "drive-managed" zone model. However, since drive-managed zoned block devices do not support zone commands, they will be treated as regular block devices and zoned will report "none".

/sys/block/<disk>/stat

Defined on file *sysfs-block*

The */sys/block/<disk>/stat* files displays the I/O statistics of disk <disk>. They contain 11 fields:

1	reads completed successfully
2	reads merged
3	sectors read
4	time spent reading (ms)
5	writes completed
6	writes merged
7	sectors written
8	time spent writing (ms)
9	I/Os currently in progress
10	time spent doing I/Os (ms)
11	weighted time spent doing I/Os (ms)
12	discards completed
13	discards merged
14	sectors discarded
15	time spent discarding (ms)
16	flush requests completed
17	time spent flushing (ms)

For more details refer *I/O statistics fields*

5.1.4 Symbols under /sys/bus

/sys/bus/dsa/devices/dsa<m>/cdev_major

Defined on file *sysfs-driver-dma-idxd*

The major number that the character device driver assigned to this device.

/sys/bus/dsa/devices/dsa<m>/cmd_status

Defined on file *sysfs-driver-dma-idxd*

The last executed device administrative command's status/error. Also last configuration error overloaded. Writing to it will clear the status.

/sys/bus/dsa/devices/dsa<m>/configurable

Defined on file *sysfs-driver-dma-idxd*

To indicate if this device is configurable or not.

/sys/bus/dsa/devices/dsa<m>/engine<m>.<n>

Defined on file *sysfs-driver-dma-idxd*

The assigned engine under this device.

/sys/bus/dsa/devices/dsa<m>/errors

Defined on file *sysfs-driver-dma-idxd*

The error information for this device.

/sys/bus/dsa/devices/dsa<m>/event_log_size

Defined on file *sysfs-driver-dma-idxd*

The event log size to be configured. Default is 64 entries and occupies 4k size if the evl entry is 64 bytes. It's visible only on platforms that support the capability.

/sys/bus/dsa/devices/dsa<m>/group<m>.<n>

Defined on file *sysfs-driver-dma-idxd*

The assigned group under this device.

/sys/bus/dsa/devices/dsa<m>/iaa_cap

Defined on file *sysfs-driver-dma-idxd*

IAA (IAX) capability mask. Exported to user space for application consumption. This attribute should only be visible on IAA devices that are version 2 or later.

/sys/bus/dsa/devices/dsa<m>/max_batch_size

Defined on file *sysfs-driver-dma-idxd*

The largest number of work descriptors in a batch. It's not visible when the device does not support batch.

/sys/bus/dsa/devices/dsa<m>/max_engines

Defined on file *sysfs-driver-dma-idxd*

The maximum number of engines supported by this device.

/sys/bus/dsa/devices/dsa<m>/max_groups

Defined on file *sysfs-driver-dma-idxd*

The maximum number of groups can be created under this device.

/sys/bus/dsa/devices/dsa<m>/max_read_buffers

Defined on file *sysfs-driver-dma-idxd*

The total number of read buffers supported by this device. The read buffers represent resources within the DSA implementation, and these resources are allocated by engines to support operations. See DSA spec v1.2 9.2.4 Total Read Buffers. It's not visible when the device does not support Read Buffer allocation control.

/sys/bus/dsa/devices/dsa<m>/max_transfer_size

Defined on file *sysfs-driver-dma-idxd*

The number of bytes to be read from the source address to perform the operation. The maximum transfer size is dependent on the workqueue the descriptor was submitted to.

/sys/bus/dsa/devices/dsa<m>/max_work_queues

Defined on file *sysfs-driver-dma-idxd*

The maximum work queue number that this device supports.

/sys/bus/dsa/devices/dsa<m>/max_work_queues_size

Defined on file *sysfs-driver-dma-idxd*

The maximum work queue size supported by this device.

/sys/bus/dsa/devices/dsa<m>/numa_node

Defined on file *sysfs-driver-dma-idxd*

The numa node number for this device.

/sys/bus/dsa/devices/dsa<m>/op_cap

Defined on file *sysfs-driver-dma-idxd*

The operation capability bit mask specify the operation types supported by the this device.

/sys/bus/dsa/devices/dsa<m>/pasid_enabled

Defined on file *sysfs-driver-dma-idxd*

To indicate if user PASID (process address space identifier) is enabled or not for this device.

/sys/bus/dsa/devices/dsa<m>/read_buffer_limit

Defined on file *sysfs-driver-dma-idxd*

The maximum number of read buffers that may be in use at one time by operations that access low bandwidth memory in the device. See DSA spec v1.2 9.2.8 GENCFG on Global Read Buffer Limit. It's not visible when the device does not support Read Buffer allocation control.

/sys/bus/dsa/devices/dsa<m>/state

Defined on file *sysfs-driver-dma-idxd*

The state information of this device. It can be either enabled or disabled.

/sys/bus/dsa/devices/dsa<m>/version

Defined on file *sysfs-driver-dma-idxd*

The hardware version number.

/sys/bus/dsa/devices/dsa<m>/wq<m>.<n>

Defined on file *sysfs-driver-dma-idxd*

The assigned work queue under this device.

/sys/bus/dsa/devices/engine<m>.<n>/group_id

Defined on file *sysfs-driver-dma-idxd*

The group that this engine belongs to.

/sys/bus/dsa/devices/group<m>.<n>/batch_progress_limit

Defined on file *sysfs-driver-dma-idxd*

Allows control of the number of batch descriptors that can be concurrently processed by an engine in the group as a fraction of the Maximum Batch Descriptors in Progress value specified in the ENGCAP register. The acceptable values are 0 (default), 1 (1/2 of max value), 2 (1/4 of the max value), and 3 (1/8 of the max value). It's visible only on platforms that support the capability.

/sys/bus/dsa/devices/group<m>.<n>/desc_progress_limit

Defined on file *sysfs-driver-dma-idxd*

Allows control of the number of work descriptors that can be concurrently processed by an engine in the group as a fraction of the Maximum Work Descriptors in Progress value specified in the ENGCAP register. The acceptable values are 0 (default), 1 (1/2 of max value), 2 (1/4 of the max value), and 3 (1/8 of the max value). It's visible only on platforms that support the capability.

/sys/bus/dsa/devices/group<m>.<n>/read_buffers_allowed

Defined on file *sysfs-driver-dma-idxd*

Indicates max number of read buffers that may be in use at one time by all engines in the group. See DSA spec v1.2 9.2.18 GRPCFG Read Buffers Allowed. It's not visible when the device does not support Read Buffer allocation control.

/sys/bus/dsa/devices/group<m>.<n>/read_buffers_reserved

Defined on file *sysfs-driver-dma-idxd*

Indicates the number of Read Buffers reserved for the use of engines in the group. See DSA spec v1.2 9.2.18 GRPCFG Read Buffers Reserved. It's not visible when the device does not support Read Buffer allocation control.

/sys/bus/dsa/devices/group<m>.<n>/use_read_buffer_limit

Defined on file *sysfs-driver-dma-idxd*

Enable the use of global read buffer limit for the group. See DSA spec v1.2 9.2.18 GRPCFG Use Global Read Buffer Limit. It's not visible when the device does not support Read Buffer allocation control.

/sys/bus/dsa/devices/wq<m>.<n>/ats_disable

Defined on file *sysfs-driver-dma-idxd*

Indicate whether ATS disable is turned on for the workqueue. 0 indicates ATS is on, and 1 indicates ATS is off for the workqueue.

/sys/bus/dsa/devices/wq<m>.<n>/block_on_fault

Defined on file *sysfs-driver-dma-idxd*

To indicate block on fault is allowed or not for the work queue to support on demand paging.

/sys/bus/dsa/devices/wq<m>.<n>/cdev_minor

Defined on file *sysfs-driver-dma-idxd*

The minor number assigned to this work queue by the character device driver.

/sys/bus/dsa/devices/wq<m>.<n>/driver_name

Defined on file *sysfs-driver-dma-idxd*

Name of driver to be bounded to the wq.

/sys/bus/dsa/devices/wq<m>.<n>/dsa<x>\!wq<m>.<n>/file<y>/cr_fault_failures

Defined on file *sysfs-driver-dma-idxd*

Show the number of Completion Record (CR) faults failures that this application has caused. The failure counter is incremented when the driver cannot fault in the address for the CR. Typically this is caused by a bad address programmed in the submitted descriptor or a malicious submitter is using bad CR address on purpose.

/sys/bus/dsa/devices/wq<m>.<n>/dsa<x>\!wq<m>.<n>/file<y>/cr_faults

Defined on file *sysfs-driver-dma-idxd*

Show the number of Completion Record (CR) faults this application has caused.

/sys/bus/dsa/devices/wq<m>.<n>/dsa<x>\!wq<m>.<n>/file<y>/pid

Defined on file *sysfs-driver-dma-idxd*

Show the process id of the application that opened the file. This is helpful information for a monitor daemon that wants to kill the application that opened the file.

/sys/bus/dsa/devices/wq<m>.<n>/enqcmds_retries

Defined on file *sysfs-driver-dma-idxd*

Indicate the number of retires for an enqcmds submission on a sharedwq. A max value to set attribute is capped at 64.

/sys/bus/dsa/devices/wq<m>.<n>/group_id

Defined on file *sysfs-driver-dma-idxd*

The group id that this work queue belongs to.

/sys/bus/dsa/devices/wq<m>.<n>/max_batch_size

Defined on file *sysfs-driver-dma-idxd*

The max batch size for this workqueue. Cannot exceed device max batch size. Configurable parameter. It's not visible when the device does not support batch.

/sys/bus/dsa/devices/wq<m>.<n>/max_transfer_size

Defined on file *sysfs-driver-dma-idxd*

The max transfer sized for this workqueue. Cannot exceed device max transfer size. Configurable parameter.

/sys/bus/dsa/devices/wq<m>.<n>/mode

Defined on file *sysfs-driver-dma-idxd*

The work queue mode type for this work queue.

/sys/bus/dsa/devices/wq<m>.<n>/occupancy

Defined on file *sysfs-driver-dma-idxd*

Show the current number of entries in this WQ if WQ Occupancy Support bit WQ capabilities is 1.

/sys/bus/dsa/devices/wq<m>.<n>/op_config

Defined on file *sysfs-driver-dma-idxd*

Shows the operation capability bits displayed in bitmap format presented by %*pb printk() output format specifier. The attribute can be configured when the WQ is disabled in order to configure the WQ to accept specific bits that correlates to the operations allowed. It's visible only on platforms that support the capability.

/sys/bus/dsa/devices/wq<m>.<n>/priority

Defined on file *sysfs-driver-dma-idxd*

The priority value of this work queue, it is a value relative to other work queue in the same group to control quality of service for dispatching work from multiple workqueues in the same group.

/sys/bus/dsa/devices/wq<m>.<n>/prs_disable

Defined on file *sysfs-driver-dma-idxd*

Controls whether PRS disable is turned on for the workqueue. 0 indicates PRS is on, and 1 indicates PRS is off for the workqueue. This option overrides block_on_fault attribute if set. It's visible only on platforms that support the capability.

/sys/bus/dsa/devices/wq<m>.<n>/size

Defined on file *sysfs-driver-dma-idxd*

The work queue size for this work queue.

/sys/bus/dsa/devices/wq<m>.<n>/state

Defined on file *sysfs-driver-dma-idxd*

The current state of the work queue.

/sys/bus/dsa/devices/wq<m>.<n>/threshold

Defined on file *sysfs-driver-dma-idxd*

The number of entries in this work queue that may be filled via a limited portal.

/sys/bus/dsa/devices/wq<m>.<n>/type

Defined on file *sysfs-driver-dma-idxd*

The type of this work queue, it can be ”kernel” type for work queue usages in the kernel space or ”user” type for work queue usages by applications in user space.

/sys/bus/firewire/devices/*/

Defined on file *sysfs-bus-firewire*

Attributes common to IEEE 1394 node devices and unit devices. Read-only. Mutable during the node device's lifetime. Immutable during the unit device's lifetime. See IEEE 1212 for semantic definitions.

These attributes are only created if the root directory of an IEEE 1394 node or the unit directory of an IEEE 1394 unit actually contains according entries.

hardware_version
Hexadecimal string representation of an u24.

hardware_version_name
Contents of a respective textual descriptor leaf.

model
Hexadecimal string representation of an u24.

model_name
Contents of a respective textual descriptor leaf.

specifier_id
Hexadecimal string representation of an u24. Mandatory in unit directories according to IEEE 1212.

vendor
Hexadecimal string representation of an u24. Mandatory in the root directory according to IEEE 1212.

vendor_name
Contents of a respective textual descriptor leaf.

version
Hexadecimal string representation of an u24. Mandatory in unit directories according to IEEE 1212.

/sys/bus/firewire/devices/fw[0-9]+/

Defined on file *sysfs-bus-firewire*

IEEE 1394 node device attributes. Read-only. Mutable during the node device's lifetime. See IEEE 1212 for semantic definitions.

config_rom
Contents of the Configuration ROM register. Binary attribute; an array of host-endian u32.

guid
The node's EUI-64 in the bus information block of Configuration ROM. Hexadecimal string representation of an u64.

/sys/bus/firewire/devices/fw[0-9]+/is_local

Defined on file *sysfs-bus-firewire*

IEEE 1394 node device attribute. Read-only and immutable. Values: 1: The sysfs entry represents a local node (a controller card).

0: The sysfs entry represents a remote node.

/sys/bus/firewire/devices/fw[0-9]+/units

Defined on file *sysfs-bus-firewire*

IEEE 1394 node device attribute. Read-only. Mutable during the node device's lifetime. See IEEE 1212 for semantic definitions.

units
Summary of all units present in an IEEE 1394 node. Contains space-separated tuples of specifier_id and version of each unit present in the node. Specifier_id and version are hexadecimal string representations of u24 of the respective unit directory entries. Specifier_id and version within each tuple are separated by a colon.

Users: udev rules to set ownership and access permissions or ACLs of /dev/fw[0-9]+ character device files

/sys/bus/firewire/devices/fw[0-9]+[.][0-9]+/

Defined on file *sysfs-bus-firewire*

IEEE 1394 unit device attributes. Read-only. Immutable during the unit device's lifetime. See IEEE 1212 for semantic definitions.

modalias
Same as MODALIAS in the uevent at device creation.

rom_index
Offset of the unit directory within the parent device's (node device's) Configuration ROM, in quadlets. Decimal string representation.

/sys/bus/firewire/drivers/sbp2/fw*/host*/target*/*:*:*:*/ieee1394_id

Defined on file *sysfs-bus-firewire*

SCSI target port identifier and logical unit identifier of a logical unit of an SBP-2 target. The identifiers are specified in SAM-2...SAM-4 annex A. They are persistent and world-wide unique properties the SBP-2 attached target.

Read-only attribute, immutable during the target's lifetime. Format, as exposed by firewire-sbp2 since 2.6.22, May 2007: Colon-separated hexadecimal string representations of

u64 EUI-64 : u24 directory_ID : u16 LUN

without 0x prefixes, without whitespace. The former sbp2 driver (removed in 2.6.37 after being superseded by firewire-sbp2) used a somewhat shorter format which was not as close to SAM.

Users: udev rules to create /dev/disk/by-id/ symlinks

/sys/bus/fsl-mc/autorescan

Defined on file *sysfs-bus-fsl-mc*

Writing a zero value to this attribute will disable the DPRC IRQs on which automatic rescan of the fsl-mc bus is performed. A non-zero value will enable the DPRC IRQs.

Users: Userspace drivers and management tools

/sys/bus/fsl-mc/rescan

Defined on file *sysfs-bus-fsl-mc*

Writing a non-zero value to this attribute will force a rescan of fsl-mc bus in the system and synchronize the objects under fsl-mc bus and the Management Complex firmware.

Users: Userspace drivers and management tools

/sys/bus/mhi/devices/.../oem_pk_hash

Defined on file *sysfs-bus-mhi*

The file holds the OEM PK Hash value of the endpoint device obtained using a BHI (Boot Host Interface) register read after at least one attempt to power up the device has been done. If read without having the device power on at least once, the file will read all 0's.

Users: Any userspace application or clients interested in device info.

/sys/bus/mhi/devices/.../serialnumber

Defined on file *sysfs-bus-mhi*

The file holds the serial number of the client device obtained using a BHI (Boot Host Interface) register read after at least one attempt to power up the device has been done. If read without having the device power on at least once, the file will read all 0's.

Users: Any userspace application or clients interested in device info.

/sys/bus/mhi/devices/.../soc_reset

Defined on file *sysfs-bus-mhi*

Initiates a SoC reset on the MHI controller. A SoC reset is a reset of last resort, and will require a complete re-init. This can be useful as a method of recovery if the device is non-responsive, or as a means of loading new firmware as a system administration task.

/sys/bus/nvmem/devices/.../nvmem

Defined on file *sysfs-bus-nvmem*

This file allows user to read/write the raw NVMEM contents. Permissions for write to this file depends on the nvmem provider configuration. Note: This file is only present if CONFIG_NVMEM_SYSFS is enabled

ex:

```
hexdump /sys/bus/nvmem/devices/qfprom0/nvmem

0000000 0000 0000 0000 0000 0000 0000 0000 0000
*
00000a0 db10 2240 0000 e000 0c00 0c00 0000 0c00
0000000 0000 0000 0000 0000 0000 0000 0000 0000
```

```
...
*
0001000
```

/sys/bus/pci/drivers/qla2xxx/.../devices/*

Defined on file *sysfs-driver-qla2xxx*

qla2xxx-udev.sh currently looks for uevent CHANGE events to signal a firmware-dump has been generated by the driver and is ready for retrieval.

Users: qla2xxx-udev.sh. Proposed changes should be mailed to linux-driver@qlogic.com

/sys/bus/platform/drivers/aspeed-vuart/*/lpc_address

Defined on file *sysfs-driver-aspeed-vuart*

Configures which IO port the host side of the UART will appear on the host <-> BMC LPC bus.

Users: OpenBMC. Proposed changes should be mailed to openbmc@lists.ozlabs.org

/sys/bus/platform/drivers/aspeed-vuart/*/sirq

Defined on file *sysfs-driver-aspeed-vuart*

Configures which interrupt number the host side of the UART will appear on the host <-> BMC LPC bus.

Users: OpenBMC. Proposed changes should be mailed to openbmc@lists.ozlabs.org

/sys/bus/platform/drivers/aspeed-vuart/*/sirq_polarity

Defined on file *sysfs-driver-aspeed-vuart*

Configures the polarity of the serial interrupt to the host via the BMC LPC bus. Set to 0 for active-low or 1 for active-high.

Users: OpenBMC. Proposed changes should be mailed to openbmc@lists.ozlabs.org

/sys/bus/usb/device/.../avoid_reset_quirk

Defined on file *sysfs-bus-usb*

Writing 1 to this file tells the kernel that this device will morph into another mode when it is reset. Drivers will not use reset for error handling for such devices.

Users:

usb_modeswitch

/sys/bus/usb/device/.../power/active_duration

Defined on file *sysfs-bus-usb*

If CONFIG_PM is enabled, then this file is present. When read, it returns the total time (in msec) that the USB device has been active, i.e. not in a suspended state. This file is read-only.

Tools can use this file and the connected_duration file to compute the percentage of time that a device has been active. For example:

```
echo $((100 * `cat active_duration` / `cat connected_duration`))
```

will give an integer percentage. Note that this does not account for counter wrap.

Users:

PowerTOP <powertop@lists.01.org> https://01.org/powertop/

/sys/bus/usb/device/.../power/connected_duration

Defined on file *sysfs-bus-usb*

If CONFIG_PM is enabled, then this file is present. When read, it returns the total time (in msec) that the USB device has been connected to the machine. This file is read-only.

Users:

PowerTOP <powertop@lists.01.org> https://01.org/powertop/

/sys/bus/usb/devices/.../bConfigurationValue

Defined on file *sysfs-bus-usb*

bConfigurationValue of the *active* configuration for the device. Writing 0 or -1 to bConfigurationValue will reset the active configuration (unconfigure the device). Writing another value will change the active configuration.

Note that some devices, in violation of the USB spec, have a configuration with a value equal to 0. Writing 0 to bConfigurationValue for these devices will install that configuration, rather then unconfigure the device.

Writing -1 will always unconfigure the device.

Users:

libusb

/sys/bus/usb/devices/.../busnum

Defined on file *sysfs-bus-usb*

Bus-number of the USB-bus the device is connected to.

Users:

libusb

/sys/bus/usb/devices/.../descriptors

Defined on file *sysfs-bus-usb*

Binary file containing cached descriptors of the device. The binary data consists of the device descriptor followed by the descriptors for each configuration of the device. Note that the wTotalLength of the config descriptors can not be trusted, as the device may have a smaller config descriptor than it advertises. The bLength field of each (sub) descriptor can be trusted, and can be used to seek forward one (sub) descriptor at a time until the next config descriptor is found. All descriptors read from this file are in bus-endian format

Users:

libusb

/sys/bus/usb/devices/.../devnum

Defined on file *sysfs-bus-usb*

Device address on the USB bus.

Users:

libusb

/sys/bus/usb/devices/.../power/autosuspend

Defined on file *sysfs-bus-usb*

Each USB device directory will contain a file named power/autosuspend. This file holds the time (in seconds) the device must be idle before it will be autosuspended. 0 means the device will be autosuspended as soon as possible. Negative values will prevent the device from being autosuspended at all, and writing a negative value will resume the device if it is already suspended.

The autosuspend delay for newly-created devices is set to the value of the usbcore.autosuspend module parameter.

/sys/bus/usb/devices/.../power/persist

Defined on file *sysfs-bus-usb*

USB device directories can contain a file named power/persist. The file holds a boolean value (0 or 1) indicating whether or not the "USB-Persist" facility is enabled for the device. For hubs this facility is always enabled and their device directories will not contain this file.

For more information, see /driver-api/usb/persist.

/sys/bus/usb/devices/.../speed

Defined on file *sysfs-bus-usb*

Speed the device is connected with to the usb-host in Mbit / second. IE one of 1.5 / 12 / 480 / 5000.

Users:

libusb

/sys/bus/usb/devices/<busnum>-<port[.port]>...:<config num>-<interface num>/supports_autosuspend

Defined on file *sysfs-bus-usb*

When read, this file returns 1 if the interface driver for this interface supports autosuspend. It also returns 1 if no driver has claimed this interface, as an unclaimed interface will not stop the device from being autosuspended if all other interface drivers are idle. The file returns 0 if autosuspend support has not been added to the driver.

Users:

USB PM tool git://git.moblin.org/users/sarah/usb-pm-tool/

/sys/bus/usb/drivers/usbtmc/*/interface_capabilities
/sys/bus/usb/drivers/usbtmc/*/device_capabilities

Defined on file *sysfs-driver-usb-usbtmc*

These files show the various USB TMC capabilities as described by the device itself. The full description of the bitfields can be found in the USB TMC documents from the USB-IF entitled "Universal Serial Bus Test and Measurement Class Specification (USBTMC) Revision 1.0" section 4.2.1.8.

The files are read only.

/sys/bus/usb/drivers/usbtmc/*/usb488_interface_capabilities
/sys/bus/usb/drivers/usbtmc/*/usb488_device_capabilities

Defined on file *sysfs-driver-usb-usbtmc*

These files show the various USB TMC capabilities as described by the device itself. The full description of the bitfields can be found in the USB TMC documents from the USB-IF entitled "Universal Serial Bus Test and Measurement Class, Subclass USB488 Specification (USBTMC-USB488) Revision 1.0" section 4.2.2.

The files are read only.

/sys/bus/vmbus/devices/<UUID>/channel_vp_mapping

Defined on file *sysfs-bus-vmbus*

The mapping of which primary/sub channels are bound to which Virtual Processors. Format: <channel's child_relid:the bound cpu's number>

Users: tools/hv/lsvmbus

/sys/bus/vmbus/devices/<UUID>/channels/<N>

Defined on file *sysfs-bus-vmbus*

Directory for per-channel information NN is the VMBUS relid associated with the channel.

/sys/bus/vmbus/devices/<UUID>/channels/<N>/cpu

Defined on file *sysfs-bus-vmbus*

VCPU (sub)channel is affinitized to

Users: tools/hv/lsvmbus and other debugging tools

/sys/bus/vmbus/devices/<UUID>/channels/<N>/events

Defined on file *sysfs-bus-vmbus*

Number of times we have signaled the host

Users: Debugging tools

/sys/bus/vmbus/devices/<UUID>/channels/<N>/in_mask

Defined on file *sysfs-bus-vmbus*

Host to guest channel interrupt mask

Users: Debugging tools

/sys/bus/vmbus/devices/<UUID>/channels/<N>/interrupts

Defined on file *sysfs-bus-vmbus*

Number of times we have taken an interrupt (incoming)

Users: Debugging tools

/sys/bus/vmbus/devices/<UUID>/channels/<N>/intr_in_full

Defined on file *sysfs-bus-vmbus*

Number of guest to host interrupts caused by the inbound ring buffer transitioning from full to not full while a packet is waiting for buffer space to become available

Users: Debugging tools

/sys/bus/vmbus/devices/<UUID>/channels/<N>/intr_out_empty

Defined on file *sysfs-bus-vmbus*

Number of guest to host interrupts caused by the outbound ring buffer transitioning from empty to not empty

Users: Debugging tools

/sys/bus/vmbus/devices/<UUID>/channels/<N>/latency

Defined on file *sysfs-bus-vmbus*

Channel signaling latency. This file is available only for performance critical channels (storage, network, etc.) that use the monitor page mechanism.

Users: Debugging tools

/sys/bus/vmbus/devices/<UUID>/channels/<N>/monitor_id

Defined on file *sysfs-bus-vmbus*

Monitor bit associated with channel. This file is available only for performance critical channels (storage, network, etc.) that use the monitor page mechanism.

Users: Debugging tools and userspace drivers

/sys/bus/vmbus/devices/<UUID>/channels/<N>/out_full_first

Defined on file *sysfs-bus-vmbus*

Number of write operations that were the first to encounter an outbound ring buffer full condition

Users: Debugging tools

/sys/bus/vmbus/devices/<UUID>/channels/<N>/out_full_total

Defined on file *sysfs-bus-vmbus*

Total number of write operations that encountered an outbound ring buffer full condition

Users: Debugging tools

/sys/bus/vmbus/devices/<UUID>/channels/<N>/out_mask

Defined on file *sysfs-bus-vmbus*

Guest to host channel interrupt mask

Users: Debugging tools

/sys/bus/vmbus/devices/<UUID>/channels/<N>/pending

Defined on file *sysfs-bus-vmbus*

Channel interrupt pending state. This file is available only for performance critical channels (storage, network, etc.) that use the monitor page mechanism.

Users: Debugging tools

/sys/bus/vmbus/devices/<UUID>/channels/<N>/read_avail

Defined on file *sysfs-bus-vmbus*

Bytes available to read

Users: Debugging tools

/sys/bus/vmbus/devices/<UUID>/channels/<N>/ring

Defined on file *sysfs-bus-vmbus*

Binary file created by uio_hv_generic for ring buffer

Users: Userspace drivers

/sys/bus/vmbus/devices/<UUID>/channels/<N>/subchannel_id

Defined on file *sysfs-bus-vmbus*

Subchannel ID associated with VMBUS channel

Users: Debugging tools and userspace drivers

/sys/bus/vmbus/devices/<UUID>/channels/<N>/write_avail

Defined on file *sysfs-bus-vmbus*

Bytes available to write

Users: Debugging tools

/sys/bus/vmbus/devices/<UUID>/class_id

Defined on file *sysfs-bus-vmbus*

The VMBus interface type GUID of the device

Users: tools/hv/lsvmbus

/sys/bus/vmbus/devices/<UUID>/device

Defined on file *sysfs-bus-vmbus*

The 16 bit device ID of the device

Users: tools/hv/lsvmbus and user level RDMA libraries

/sys/bus/vmbus/devices/<UUID>/device_id

Defined on file *sysfs-bus-vmbus*

The VMBus interface instance GUID of the device

Users: tools/hv/lsvmbus

/sys/bus/vmbus/devices/<UUID>/id

Defined on file *sysfs-bus-vmbus*

The VMBus child_relid of the device's primary channel

Users: tools/hv/lsvmbus

/sys/bus/vmbus/devices/<UUID>/numa_node

Defined on file *sysfs-bus-vmbus*

This NUMA node to which the VMBUS device is attached, or -1 if the node is unknown.

/sys/bus/vmbus/devices/<UUID>/vendor

Defined on file *sysfs-bus-vmbus*

The 16 bit vendor ID of the device

Users: tools/hv/lsvmbus and user level RDMA libraries

/sys/bus/vmbus/hibernation

Defined on file *sysfs-bus-vmbus*

Whether the host supports hibernation for the VM.

Users: Daemon that sets up swap partition/file for hibernation.

/sys/bus/w1/devices/.../eeprom

Defined on file *sysfs-driver-w1_ds28e04*

read/write the contents of the EEPROM memory of the DS28E04-100 see /w1/slaves/w1_ds28e04 for detailed information

Users: any user space application which wants to communicate with DS28E04-100

/sys/bus/w1/devices/.../offset

Defined on file *sysfs-driver-w1_ds2438*

write the contents to the offset register of the DS2438 see /w1/slaves/w1_ds2438 for detailed information

Users: any user space application which wants to communicate with DS2438

/sys/bus/w1/devices/.../page1

Defined on file *sysfs-driver-w1_ds2438*

read the contents of the page1 of the DS2438 see /w1/slaves/w1_ds2438 for detailed information

Users: any user space application which wants to communicate with DS2438

/sys/bus/w1/devices/.../pio

Defined on file *sysfs-driver-w1_ds28e04*

read/write the contents of the two PIO's of the DS28E04-100 see /w1/slaves/w1_ds28e04 for detailed information

Users: any user space application which wants to communicate with DS28E04-100

/sys/bus/w1/devices/.../w1_master_timeout_us

Defined on file *sysfs-bus-w1*

Bus scanning interval, microseconds component. Some of 1-Wire devices commonly associated with physical access control systems are attached/generate presence for as short as 100 ms - hence the tens-to-hundreds milliseconds scan intervals are required.

see /w1/w1-generic for detailed information.

Users: any user space application which wants to know bus scanning interval

/sys/bus/w1/devices/.../w1_seq

Defined on file *sysfs-driver-w1_ds28ea00*

Support for the DS28EA00 chain sequence function see /w1/slaves/w1_therm for detailed information

Users: any user space application which wants to communicate with DS28EA00

/sys/bus/wmi/devices/05901221-D566-11D1-B2F0-00A0C9062910[-X]/bmof

Defined on file *sysfs-platform-wmi-bmof*

Binary MOF metadata used to describe the details of available ACPI WMI interfaces.

See /wmi/devices/wmi-bmof for details.

/sys/bus/xen-backend/devices/*/devtype

Defined on file *sysfs-bus-xen-backend*

The type of the device. e.g., one of: 'vbd' (block), 'vif' (network), or 'vfb' (framebuffer).

/sys/bus/xen-backend/devices/*/nodename

Defined on file *sysfs-bus-xen-backend*

XenStore node (under /local/domain/NNN/) for this backend device.

/sys/bus/xen-backend/devices/*/state

Defined on file *sysfs-bus-xen-backend*

The state of the device. One of: 'Unknown', 'Initialising', 'Initialised', 'Connected', 'Closing', 'Closed', 'Reconfiguring', 'Reconfigured'.

/sys/bus/xen-backend/devices/vbd-*/mode

Defined on file *sysfs-bus-xen-backend*

Whether the block device is read-only ('r') or read-write ('w').

/sys/bus/xen-backend/devices/vbd-*/physical_device

Defined on file *sysfs-bus-xen-backend*

The major:minor number (in hexadecimal) of the physical device providing the storage for this backend block device.

/sys/bus/xen-backend/devices/vbd-*/statistics/f_req

Defined on file *sysfs-bus-xen-backend*

Number of flush requests from the frontend.

/sys/bus/xen-backend/devices/vbd-*/statistics/oo_req

Defined on file *sysfs-bus-xen-backend*

Number of requests delayed because the backend was too busy processing previous requests.

/sys/bus/xen-backend/devices/vbd-*/statistics/rd_req

Defined on file *sysfs-bus-xen-backend*

Number of read requests from the frontend.

/sys/bus/xen-backend/devices/vbd-*/statistics/rd_sect

Defined on file *sysfs-bus-xen-backend*

Number of sectors read by the frontend.

/sys/bus/xen-backend/devices/vbd-*/statistics/wr_req

Defined on file *sysfs-bus-xen-backend*

Number of write requests from the frontend.

/sys/bus/xen-backend/devices/vbd-*/statistics/wr_sect

Defined on file *sysfs-bus-xen-backend*

Number of sectors written by the frontend.

5.1.5 Symbols under /sys/class

/sys/class/backlight/<backlight>/actual_brightness

Defined on file *sysfs-class-backlight*

Show the actual brightness by querying the hardware.

Users: HAL

/sys/class/backlight/<backlight>/bl_power

Defined on file *sysfs-class-backlight*

Control BACKLIGHT power, values are FB_BLANK_* from fb.h

- FB_BLANK_UNBLANK (0) : power on.
- FB_BLANK_POWERDOWN (4) : power off

Users: HAL

/sys/class/backlight/<backlight>/brightness

Defined on file *sysfs-class-backlight*

Control the brightness for this <backlight>. Values are between 0 and max_brightness. This file will also show the brightness level stored in the driver, which may not be the actual brightness (see actual_brightness).

Users: HAL

/sys/class/backlight/<backlight>/max_brightness

Defined on file *sysfs-class-backlight*

Maximum brightness for <backlight>.

Users: HAL

/sys/class/backlight/<backlight>/type

Defined on file *sysfs-class-backlight*

The type of interface controlled by <backlight>. ”firmware”: The driver uses a standard firmware interface ”platform”: The driver uses a platform-specific interface ”raw”: The driver controls hardware registers directly

In the general case, when multiple backlight interfaces are available for a single device, firmware control should be preferred to platform control should be preferred to raw control.

Using a firmware interface reduces the probability of confusion with the hardware and the OS independently updating the backlight state. Platform interfaces are mostly a holdover from pre-standardisation of firmware interfaces.

/sys/class/infiniband/<device-name>/hw_counters/lifespan
/sys/class/infiniband/<device-name>/ports/<port-num>/hw_counters/lifespan

Defined on file *sysfs-class-infiniband*

The optional "hw_counters" subdirectory can be under either the parent device or the port subdirectories or both. If present, there are a list of counters provided by the hardware. They may match some of the counters in the counters directory, but they often include many other counters. In addition to the various counters, there will be a file named "lifespan" that configures how frequently the core should update the counters when they are being accessed (counters are not updated if they are not being accessed). The lifespan is in milliseconds and defaults to 10 unless set to something else by the driver. Users may echo a value between 0-10000 to the lifespan file to set the length of time between updates in milliseconds.

/sys/class/infiniband/<device>/fw_ver

Defined on file *sysfs-class-infiniband*

(RO) Display firmware version

/sys/class/infiniband/<device>/node_desc

Defined on file *sysfs-class-infiniband*

(RW) Update the node description with information such as the node's hostname, so that IB network management software can tie its view to the real world.

/sys/class/infiniband/<device>/node_type
/sys/class/infiniband/<device>/node_guid
/sys/class/infiniband/<device>/sys_image_guid

Defined on file *sysfs-class-infiniband*

node_type:	(RO) Node type (CA, RNIC, usNIC, usNIC UDP, switch or router)
node_guid:	(RO) Node GUID
sys_image_guid:	(RO) System image GUID

/sys/class/infiniband/<device>/ports/<port-num>/counters/symbol_error
/sys/class/infiniband/<device>/ports/<port-num>/counters/port_rcv_errors
/sys/class/infiniband/<device>/ports/<port-num>/counters/port_rcv_remote_physical_error
/sys/class/infiniband/<device>/ports/<port-num>/counters/port_rcv_switch_relay_errors
/sys/class/infiniband/<device>/ports/<port-num>/counters/link_error_recovery
/sys/class/infiniband/<device>/ports/<port-num>/counters/port_xmit_constraint_errors
/sys/class/infiniband/<device>/ports/<port-num>/counters/port_rcv_contraint_errors
/sys/class/infiniband/<device>/ports/<port-num>/counters/local_link_integrity_errors
/sys/class/infiniband/<device>/ports/<port-num>/counters/excessive_buffer_overrun_errors
/sys/class/infiniband/<device>/ports/<port-num>/counters/port_xmit_data
/sys/class/infiniband/<device>/ports/<port-num>/counters/port_rcv_data
/sys/class/infiniband/<device>/ports/<port-num>/counters/port_xmit_packets
/sys/class/infiniband/<device>/ports/<port-num>/counters/port_rcv_packets
/sys/class/infiniband/<device>/ports/<port-num>/counters/unicast_rcv_packets
/sys/class/infiniband/<device>/ports/<port-num>/counters/unicast_xmit_packets
/sys/class/infiniband/<device>/ports/<port-num>/counters/multicast_rcv_packets
/sys/class/infiniband/<device>/ports/<port-num>/counters/multicast_xmit_packets
/sys/class/infiniband/<device>/ports/<port-num>/counters/link_downed
/sys/class/infiniband/<device>/ports/<port-num>/counters/port_xmit_discards
/sys/class/infiniband/<device>/ports/<port-num>/counters/VL15_dropped
/sys/class/infiniband/<device>/ports/<port-num>/counters/port_xmit_wait

Defined on file *sysfs-class-infiniband*

Errors info:

symbol_error: (RO) Total number of minor link errors detected on one or more physical lanes.

port_rcv_errors : (RO) Total number of packets containing an error that were received on the port.

port_rcv_remote_physical_errors : (RO) Total number of packets marked with the EBP delimiter received on the port.

port_rcv_switch_relay_errors : (RO) Total number of packets received on the port that were discarded because they could not be forwarded by the switch relay.

link_error_recovery: (RO) Total number of times the Port Training state machine has successfully completed the link error recovery process.

port_xmit_constraint_errors: (RO) Total number of packets not transmitted from the switch physical port due to outbound raw filtering or failing outbound partition or IP version check.

port_rcv_constraint_errors: (RO) Total number of packets received on the switch physical port that are discarded due to inbound raw filtering or failing inbound partition or IP version check.

local_link_integrity_errors: (RO) The number of times that the count of local physical errors exceeded the threshold specified by LocalPhyErrors

excessive_buffer_overrun_errors: (RO) This counter, indicates an input buffer overrun. It indicates possible misconfiguration of a port, either by the Subnet Manager (SM) or by user intervention. It can also indicate hardware issues or extremely poor link signal integrity

Data info:

port_xmit_data: (RO) Total number of data octets, divided by 4 (lanes), transmitted on all VLs. This is 64 bit counter

port_rcv_data: (RO) Total number of data octets, divided by 4 (lanes), received on all VLs. This is 64 bit counter.

port_xmit_packets: (RO) Total number of packets transmitted on all VLs from this port. This may include packets with errors. This is 64 bit counter.

port_rcv_packets: (RO) Total number of packets (this may include packets containing Errors. This is 64 bit counter.

link_downed: (RO) Total number of times the Port Training state machine has failed the link error recovery process and downed the link.

unicast_rcv_packets: (RO) Total number of unicast packets, including unicast packets containing errors.

unicast_xmit_packets: (RO) Total number of unicast packets transmitted on all VLs from the port. This may include unicast packets with errors.

multicast_rcv_packets: (RO) Total number of multicast packets, including multicast packets containing errors.

multicast_xmit_packets: (RO) Total number of multicast packets transmitted on all VLs from the port. This may include multicast packets with errors.

Misc info:

port_xmit_discards: (RO) Total number of outbound packets discarded by the port because the port is down or congested.

VL15_dropped: (RO) Number of incoming VL15 packets dropped due to resource limitations (e.g., lack of buffers) of the port.

port_xmit_wait: (RO) The number of ticks during which the port had data to transmit but no data was sent during the entire tick (either because of insufficient credits or because of lack of arbitration).

Each of these files contains the corresponding value from the port's Performance Management PortCounters attribute, as described in the InfiniBand Architecture Specification.

/sys/class/infiniband/<device>/ports/<port-num>/lid
/sys/class/infiniband/<device>/ports/<port-num>/rate
/sys/class/infiniband/<device>/ports/<port-num>/lid_mask_count
/sys/class/infiniband/<device>/ports/<port-num>/sm_sl
/sys/class/infiniband/<device>/ports/<port-num>/sm_lid
/sys/class/infiniband/<device>/ports/<port-num>/state
/sys/class/infiniband/<device>/ports/<port-num>/phys_state
/sys/class/infiniband/<device>/ports/<port-num>/cap_mask

Defined on file *sysfs-class-infiniband*

lid:	(RO) Port LID
rate:	(RO) Port data rate (active width * active speed)
lid_mask_count:	(RO) Port LID mask count
sm_sl:	(RO) Subnet manager SL for port's subnet
sm_lid:	(RO) Subnet manager LID for port's subnet
state:	(RO) Port state (DOWN, INIT, ARMED, ACTIVE or ACTIVE_DEFER)
phys_state:	(RO) Port physical state (Sleep, Polling, LinkUp, etc)
cap_mask:	(RO) Port capability mask. 2 bits here are settable- IsCommunicationManagementSupported (set when CM module is loaded) and IsSM (set via open of issmN file).

/sys/class/infiniband/<device>/ports/<port-num>/link_layer

Defined on file *sysfs-class-infiniband*

(RO) Link layer type information (Infiniband or Ethernet type)

/sys/class/infiniband/<hca>/ports/<port-number>/gid_attrs/ndevs/<gid-index>

Defined on file *sysfs-class-infiniband*

The net-device's name associated with the GID resides at index <gid-index>.

/sys/class/infiniband/<hca>/ports/<port-number>/gid_attrs/types/<gid-index>

Defined on file *sysfs-class-infiniband*

The RoCE type of the associated GID resides at index <gid-index>. This could either be "IB/RoCE v1" for IB and RoCE v1 based GIDs or "RoCE v2" for RoCE v2 based GIDs.

/sys/class/infiniband/bnxt_reX/hw_rev
/sys/class/infiniband/bnxt_reX/hca_type

Defined on file *sysfs-class-infiniband*

hw_rev:	(RO)	Hardware revision number
hca_type:	(RO)	Host channel adapter type

/sys/class/infiniband/cxgb4_X/hw_rev
/sys/class/infiniband/cxgb4_X/hca_type
/sys/class/infiniband/cxgb4_X/board_id

Defined on file *sysfs-class-infiniband*

hw_rev:	(RO) Hardware revision number
hca_type:	(RO) Driver short name. Should normally match the name in its bus driver structure (e.g. pci_driver::name)
board_id:	(RO) Manufacturing board id. (Vendor + device information)

sysfs interface for Intel IB driver qib

/sys/class/infiniband/hfi1_X/hw_rev
/sys/class/infiniband/hfi1_X/board_id
/sys/class/infiniband/hfi1_X/nctxts
/sys/class/infiniband/hfi1_X/serial
/sys/class/infiniband/hfi1_X/chip_reset
/sys/class/infiniband/hfi1_X/boardversion
/sys/class/infiniband/hfi1_X/nfreectxts
/sys/class/infiniband/hfi1_X/tempsense

Defined on file *sysfs-class-infiniband*

hw_rev:	(RO) Hardware revision number
board_id:	(RO) Manufacturing board id
nctxts:	(RO) Total contexts available.
serial:	(RO) Board serial number
chip_reset:	(WO) Write ”reset” to this file to reset the chip if possible. Only allowed if no user contexts are open that use chip resources.
boardversion:	(RO) Human readable board info
nfreectxts:	(RO) The number of free user ports (contexts) available.
tempsense:	(RO) Thermal sense information

/sys/class/infiniband/hfi1_X/ports/<N>/CCMgtA/cc_settings_bin
/sys/class/infiniband/hfi1_X/ports/<N>/CCMgtA/cc_table_bin
/sys/class/infiniband/hfi1_X/ports/<N>/CCMgtA/cc_prescan

Defined on file *sysfs-class-infiniband*

Per-port congestion control.

cc_table_bin	(RO) CCA tables used by PSM2 Congestion control table size followed by table entries. Binary attribute.
cc_settings_bin	(RO) Congestion settings: port control, control map and an array of 16 entries for the congestion entries - increase, timer, event log trigger threshold and the minimum injection rate delay. Binary attribute.
cc_prescan	(RW) enable prescanning for faster BECN response. Write ”on” to enable and ”off” to disable.

/sys/class/infiniband/hfi1_X/ports/<N>/sc2vl/[0-31]
/sys/class/infiniband/hfi1_X/ports/<N>/sl2sc/[0-31]
/sys/class/infiniband/hfi1_X/ports/<N>/vl2mtu/[0-15]

Defined on file *sysfs-class-infiniband*

sc2vl/:	(RO) 32 files (0 - 31) used to translate sl->vl
sl2sc/:	(RO) 32 files (0 - 31) used to translate sl->sc
vl2mtu/:	(RO) 16 files (0 - 15) used to determine MTU for vl

/sys/class/infiniband/hfi1_X/sdma_<N>/cpu_list
/sys/class/infiniband/hfi1_X/sdma_<N>/vl

Defined on file *sysfs-class-infiniband*

sdma<N>/ contains one directory per sdma engine (0 - 15)

cpu_list:	(RW) List of cpus for user-process to sdma engine assignment.
vl:	(RO) Displays the virtual lane (vl) the sdma engine maps to.

This interface gives the user control on the affinity settings for the device. As an example, to set an sdma engine irq affinity and thread affinity of a user processes to use the sdma engine, which is "near" in terms of NUMA configuration, or physical cpu location, the user will do:

```
echo "3" > /proc/irq/<N>/smp_affinity_list
echo "4-7" > /sys/devices/.../sdma3/cpu_list
cat /sys/devices/.../sdma3/vl
0
echo "8" > /proc/irq/<M>/smp_affinity_list
echo "9-12" > /sys/devices/.../sdma4/cpu_list
cat /sys/devices/.../sdma4/vl
1
```

to make sure that when a process runs on cpus 4,5,6, or 7, and uses vl=0, then sdma engine 3 is selected by the driver, and also the interrupt of the sdma engine 3 is steered to cpu 3. Similarly, when a process runs on cpus 9,10,11, or 12 and sets vl=1, then engine 4 will be selected and the irq of the sdma engine 4 is steered to cpu 8. This assumes that in the above N is the irq number of "sdma3", and M is irq number of "sdma4" in the /proc/interrupts file.

sysfs interface for QLogic qedr NIC Driver

/sys/class/infiniband/mlx4_X/hw_rev
/sys/class/infiniband/mlx4_X/hca_type
/sys/class/infiniband/mlx4_X/board_id

Defined on file *sysfs-class-infiniband*

hw_rev:	(RO) Hardware revision number
hca_type:	(RO) Host channel adapter type
board_id:	(RO) Manufacturing board ID

/sys/class/infiniband/mlx4_X/iov/<pci-slot-num>/ports/<m>/smi_enabled
/sys/class/infiniband/mlx4_X/iov/<pci-slot-num>/ports/<m>/enable_smi_admin

Defined on file *sysfs-class-infiniband*

Enabling QP0 on VFs for selected VF/port. By default, no VFs are enabled for QP0 operation.

smi_enabled:	(RO)	Indicates whether smi is currently enabled for the indicated VF/port
enable_smi_admin:	(RW)	Used by the admin to request that smi capability be enabled or disabled for the indicated VF/port. 0 = disable, 1 = enable.

The requested enablement will occur at the next reset of the VF (e.g. driver restart on the VM which owns the VF).

sysfs interface for Chelsio T4/T5 RDMA driver (cxgb4)

/sys/class/infiniband/mlx4_X/iov/ports/<port-num>/gids/<n>
/sys/class/infiniband/mlx4_X/iov/ports/<port-num>/admin_guids/<n>
/sys/class/infiniband/mlx4_X/iov/ports/<port-num>/pkeys/<n>
/sys/class/infiniband/mlx4_X/iov/ports/<port-num>/mcgs/
/sys/class/infiniband/mlx4_X/iov/ports/<pci-slot-num>/ports/<m>/gid_idx/0
/sys/class/infiniband/mlx4_X/iov/ports/<pci-slot-num>/ports/<m>/pkey_idx/<n>

Defined on file *sysfs-class-infiniband*

The sysfs iov directory is used to manage and examine the port P_Key and guid paravirtualization. This directory is added only for the master -- slaves do not have it.

Under iov/ports, the administrator may examine the gid and P_Key tables as they are present in the device (and as are seen in the ”network view” presented to the SM).

The ”pkeys” and ”gids” subdirectories contain one file for each entry in the port's P_Key or GID table respectively. For example, ports/1/pkeys/10 contains the value at index 10 in port 1's P_Key table.

gids/<n>:	(RO) The physical port gids n = 0..127
admin_guids/<n>:	(RW) Allows examining or changing the administrative state of a given GUID n = 0..127
pkeys/<n>:	(RO) Displays the contents of the physical key table n = 0..126
mcgs/:	(RO) Multicast group table
<m>/gid_idx/0:	(RO) Display the GID mapping m = 1..2
<m>/pkey_idx/<n>:	(RW) Writable except for RoCE pkeys. m = 1..2, n = 0..126 Under the iov/<pci slot number> directories, the admin may map the index numbers in the physical tables (as under iov/ports) to the paravirtualized index numbers that guests see. For example, if the administrator, for port 1 on guest 2 maps physical pkey index 10 to virtual index 1, then that guest, whenever it uses its pkey index 1, will actually be using the real pkey index 10.

/sys/class/infiniband/mlx5_X/hw_rev
/sys/class/infiniband/mlx5_X/hca_type
/sys/class/infiniband/mlx5_X/reg_pages
/sys/class/infiniband/mlx5_X/fw_pages

Defined on file *sysfs-class-infiniband*

[to be documented]

sysfs interface for Cisco VIC (usNIC) Verbs Driver

/sys/class/infiniband/mthcaX/hw_rev
/sys/class/infiniband/mthcaX/hca_type
/sys/class/infiniband/mthcaX/board_id

Defined on file *sysfs-class-infiniband*

hw_rev:	(RO) Hardware revision number
hca_type:	(RO) Host Channel Adapter type: MT23108, MT25208 (MT23108 compat mode), MT25208 or MT25204
board_id:	(RO) Manufacturing board ID

sysfs interface for Mellanox ConnectX HCA IB driver (mlx4)

/sys/class/infiniband/ocrdmaX/hca_type

Defined on file *sysfs-class-infiniband*

hca_type: (RO) Display FW version

sysfs interface for Intel Omni-Path driver (HFI1)

/sys/class/infiniband/ocrdmaX/hw_rev

Defined on file *sysfs-class-infiniband*

hw_rev: (RO) Hardware revision number

/sys/class/infiniband/qedrX/hw_rev
/sys/class/infiniband/qedrX/hca_type

Defined on file *sysfs-class-infiniband*

hw_rev:	(RO)	Hardware revision number
hca_type:	(RO)	Display HCA type

sysfs interface for VMware Paravirtual RDMA driver

/sys/class/infiniband/qibX/ports/<N>/CCMgtA/cc_settings_bin
/sys/class/infiniband/qibX/ports/<N>/CCMgtA/cc_table_bin

Defined on file *sysfs-class-infiniband*

Per-port congestion control. Both are binary attributes.

cc_table_bin	(RO) Congestion control table size followed by table entries.
cc_settings_bin	(RO) Congestion settings: port control, control map and an array of 16 entries for the congestion entries - increase, timer, event log trigger threshold and the minimum injection rate delay.

/sys/class/infiniband/qibX/ports/<N>/diag_counters/rc_resends
/sys/class/infiniband/qibX/ports/<N>/diag_counters/seq_naks
/sys/class/infiniband/qibX/ports/<N>/diag_counters/rdma_seq
/sys/class/infiniband/qibX/ports/<N>/diag_counters/rnr_naks
/sys/class/infiniband/qibX/ports/<N>/diag_counters/other_naks
/sys/class/infiniband/qibX/ports/<N>/diag_counters/rc_timeouts
/sys/class/infiniband/qibX/ports/<N>/diag_counters/look_pkts
/sys/class/infiniband/qibX/ports/<N>/diag_counters/pkt_drops
/sys/class/infiniband/qibX/ports/<N>/diag_counters/dma_wait
/sys/class/infiniband/qibX/ports/<N>/diag_counters/unaligned

Defined on file *sysfs-class-infiniband*

[to be documented]

sysfs interface for Mellanox Connect-IB HCA driver mlx5

/sys/class/infiniband/qibX/ports/<N>/linkstate/loopback
/sys/class/infiniband/qibX/ports/<N>/linkstate/led_override
/sys/class/infiniband/qibX/ports/<N>/linkstate/hrtbt_enable
/sys/class/infiniband/qibX/ports/<N>/linkstate/status
/sys/class/infiniband/qibX/ports/<N>/linkstate/status_str

Defined on file *sysfs-class-infiniband*

[to be documented]

loopback:	(WO)
led_override:	(WO)
hrtbt_enable:	(RW)
status:	(RO)
status_str:	(RO) Displays information about the link state, possible cable/switch problems, and hardware errors. Possible states are- ”Initted”, ”Present”, ”IB_link_up”, ”IB_configured” or ”Fatal_Hardware_Error”.

/sys/class/infiniband/qibX/ports/<N>/sl2vl/[0-15]

Defined on file *sysfs-class-infiniband*

(RO) The directory contains 16 files numbered 0-15 that specify the Service Level (SL). Listing the SL files returns the Virtual Lane (VL) as programmed by the SL.

/sys/class/infiniband/qibX/version
/sys/class/infiniband/qibX/hw_rev
/sys/class/infiniband/qibX/hca_type
/sys/class/infiniband/qibX/board_id
/sys/class/infiniband/qibX/boardversion
/sys/class/infiniband/qibX/nctxts
/sys/class/infiniband/qibX/localbus_info
/sys/class/infiniband/qibX/tempsense
/sys/class/infiniband/qibX/serial
/sys/class/infiniband/qibX/nfreectxts
/sys/class/infiniband/qibX/chip_reset

Defined on file *sysfs-class-infiniband*

version:	(RO) Display version information of installed software and drivers.
hw_rev:	(RO) Hardware revision number
hca_type:	(RO) Host channel adapter type
board_id:	(RO) Manufacturing board id
boardversion:	(RO) Current version of the chip architecture
nctxts:	(RO) Return the number of user ports (contexts) available
localbus_info:	(RO) Human readable localbus info
tempsense:	(RO) Display temp sense registers in decimal
serial:	(RO) Serial number of the HCA
nfreectxts:	(RO) The number of free user ports (contexts) available.
chip_reset:	(WO) Reset the chip if possible by writing ”reset” to this file. Only allowed if no user contexts are open that use chip resources.

/sys/class/infiniband/usnic_X/board_id
/sys/class/infiniband/usnic_X/config
/sys/class/infiniband/usnic_X/qp_per_vf
/sys/class/infiniband/usnic_X/max_vf
/sys/class/infiniband/usnic_X/cq_per_vf
/sys/class/infiniband/usnic_X/iface

Defined on file *sysfs-class-infiniband*

board_id:	(RO) Manufacturing board id
config:	(RO) Report the configuration for this PF
qp_per_vf:	(RO) Queue pairs per virtual function.
max_vf:	(RO) Max virtual functions
cq_per_vf:	(RO) Completion queue per virtual function
iface:	(RO) Shows which network interface this usNIC entry is associated to (visible with ifconfig).

/sys/class/infiniband/usnic_X/qpn/summary
/sys/class/infiniband/usnic_X/qpn/context

Defined on file *sysfs-class-infiniband*

[to be documented]

sysfs interface for Emulex RoCE HCA Driver

/sys/class/infiniband/vmw_pvrdmaX/hw_rev
/sys/class/infiniband/vmw_pvrdmaX/hca_type
/sys/class/infiniband/vmw_pvrdmaX/board_id

Defined on file *sysfs-class-infiniband*

hw_rev:	(RO)	Hardware revision number
hca_type:	(RO)	Host channel adapter type
board_id:	(RO)	Display PVRDMA manufacturing board ID

sysfs interface for Broadcom NetXtreme-E RoCE driver

/sys/class/infiniband_mad/abi_version

Defined on file *sysfs-class-infiniband*

(RO) Value is incremented if any changes are made that break userspace ABI compatibility of umad & issm devices.

/sys/class/infiniband_mad/umad<N>/ibdev
/sys/class/infiniband_mad/umad<N>/port
/sys/class/infiniband_mad/issm<N>/ibdev
/sys/class/infiniband_mad/issm<N>/port

Defined on file *sysfs-class-infiniband*

Each port of each InfiniBand device has a "umad" device and an "issm" device attached. For example, a two-port HCA will have two umad devices and two issm devices, while a switch will have one device of each type (for switch port 0).

ibdev:	(RO) Show Infiniband (IB) device name
port:	(RO) Display port number

/sys/class/infiniband_srp/srp-<hca>-<port_number>/add_target

Defined on file *sysfs-driver-ib_srp*

Interface for making ib_srp connect to a new target. One can request ib_srp to connect to a new target by writing a comma-separated list of login parameters to this sysfs attribute. The supported parameters are:

- id_ext, a 16-digit hexadecimal number specifying the eight byte identifier extension in the 16-byte SRP target port identifier. The target port identifier is sent by ib_srp to the target in the SRP_LOGIN_REQ request.

- ioc_guid, a 16-digit hexadecimal number specifying the eight byte I/O controller GUID portion of the 16-byte target port identifier.
- dgid, a 32-digit hexadecimal number specifying the destination GID.
- pkey, a four-digit hexadecimal number specifying the InfiniBand partition key.
- service_id, a 16-digit hexadecimal number specifying the InfiniBand service ID used to establish communication with the SRP target. How to find out the value of the service ID is specified in the documentation of the SRP target.
- max_sect, a decimal number specifying the maximum number of 512-byte sectors to be transferred via a single SCSI command.
- max_cmd_per_lun, a decimal number specifying the maximum number of outstanding commands for a single LUN.
- io_class, a hexadecimal number specifying the SRP I/O class. Must be either 0xff00 (rev 10) or 0x0100 (rev 16a). The I/O class defines the format of the SRP initiator and target port identifiers.
- initiator_ext, a 16-digit hexadecimal number specifying the identifier extension portion of the SRP initiator port identifier. This data is sent by the initiator to the target in the SRP_LOGIN_REQ request.
- cmd_sg_entries, a number in the range 1..255 that specifies the maximum number of data buffer descriptors stored in the SRP_CMD information unit itself. With allow_ext_sg=0 the parameter cmd_sg_entries defines the maximum S/G list length for a single SRP_CMD, and commands whose S/G list length exceeds this limit after S/G list collapsing will fail.
- allow_ext_sg, whether ib_srp is allowed to include a partial memory descriptor list in an SRP_CMD instead of the entire list. If a partial memory descriptor list has been included in an SRP_CMD the remaining memory descriptors are communicated from initiator to target via an additional RDMA transfer. Setting allow_ext_sg to 1 increases the maximum amount of data that can be transferred between initiator and target via a single SCSI command. Since not all SRP target implementations support partial memory descriptor lists the default value for this option is 0.
- sg_tablesize, a number in the range 1..2048 specifying the maximum S/G list length the SCSI layer is allowed to pass to ib_srp. Specifying a value that exceeds cmd_sg_entries is only safe with partial memory descriptor list support enabled (allow_ext_sg=1).
- comp_vector, a number in the range 0..n-1 specifying the MSI-X completion vector of the first RDMA channel. Some HCA's allocate multiple (n) MSI-X vectors per HCA port. If the IRQ affinity masks of these interrupts have been configured such that each MSI-X interrupt is handled by a different CPU then the comp_vector parameter can be used to spread the SRP completion workload over multiple CPU's.
- tl_retry_count, a number in the range 2..7 specifying the IB RC retry count.
- queue_size, the maximum number of commands that the initiator is allowed to queue per SCSI host. The default value for this parameter is 62. The lowest supported value is 2.
- max_it_iu_size, a decimal number specifying the maximum initiator to target information unit length.

/sys/class/infiniband_srp/srp-<hca>-<port_number>/ibdev

Defined on file *sysfs-driver-ib_srp*

HCA name (<hca>).

/sys/class/infiniband_srp/srp-<hca>-<port_number>/port

Defined on file *sysfs-driver-ib_srp*

HCA port number (<port_number>).

/sys/class/infiniband_verbs/abi_version

Defined on file *sysfs-class-infiniband*

(RO) Value is incremented if any changes are made that break userspace ABI compatibility of uverbs devices.

sysfs interface for Mellanox IB HCA low-level driver (mthca)

/sys/class/infiniband_verbs/uverbs<N>/ibdev
/sys/class/infiniband_verbs/uverbs<N>/abi_version

Defined on file *sysfs-class-infiniband*

ibdev:	(RO) Display Infiniband (IB) device name
abi_version:	(RO) Show ABI version of IB device specific interfaces.

/sys/class/rfkill

Defined on file *sysfs-class-rfkill*

The rfkill class subsystem folder. Each registered rfkill driver is represented by an rfkillX subfolder (X being an integer >= 0).

/sys/class/rfkill/rfkill[0-9]+/hard

Defined on file *sysfs-class-rfkill*

Current hardblock state. This file is read only. Values: A numeric value.

0: inactive
The transmitter is (potentially) active.

1: active
The transmitter is forced off by something outside of the driver's control.

/sys/class/rfkill/rfkill[0-9]+/name

Defined on file *sysfs-class-rfkill*

Name assigned by driver to this key (interface or driver name). Values: arbitrary string.

/sys/class/rfkill/rfkill[0-9]+/persistent

Defined on file *sysfs-class-rfkill*

Whether the soft blocked state is initialised from non-volatile storage at startup. Values: A numeric value:

- 0: false
- 1: true

/sys/class/rfkill/rfkill[0-9]+/soft

Defined on file *sysfs-class-rfkill*

Current softblock state. This file is read and write. Values: A numeric value.

0: inactive
The transmitter is (potentially) active.

1: active
The transmitter is turned off by software.

/sys/class/rfkill/rfkill[0-9]+/state

Defined on file *sysfs-class-rfkill*

Current state of the transmitter. This file was scheduled to be removed in 2014, but due to its large number of users it will be sticking around for a bit longer. Despite it being marked as stable, the newer ”hard” and ”soft” interfaces should be preferred, since it is not possible to express the 'soft and hard block' state of the rfkill driver through this interface. There will likely be another attempt to remove it in the future. Values: A numeric value.

0: RFKILL_STATE_SOFT_BLOCKED
transmitter is turned off by software

1: RFKILL_STATE_UNBLOCKED
transmitter is (potentially) active

2: RFKILL_STATE_HARD_BLOCKED
transmitter is forced off by something outside of the driver's control.

/sys/class/rfkill/rfkill[0-9]+/type

Defined on file *sysfs-class-rfkill*

Driver type string (”wlan”, ”bluetooth”, etc). Values: See include/linux/rfkill.h.

/sys/class/scsi_host/host<n>/allow_ext_sg

Defined on file *sysfs-driver-ib_srp*

Whether ib_srp is allowed to include a partial memory descriptor list in an SRP_CMD when communicating with an SRP target.

/sys/class/scsi_host/host<n>/ch_count

Defined on file *sysfs-driver-ib_srp*

Number of RDMA channels used for communication with the SRP target.

/sys/class/scsi_host/host<n>/cmd_sg_entries

Defined on file *sysfs-driver-ib_srp*

Maximum number of data buffer descriptors that may be sent to the target in a single SRP_CMD request.

/sys/class/scsi_host/host<n>/comp_vector

Defined on file *sysfs-driver-ib_srp*

Completion vector used for the first RDMA channel.

/sys/class/scsi_host/host<n>/dgid

Defined on file *sysfs-driver-ib_srp*

InfiniBand destination GID used for communication with the SRP target. Differs from orig_dgid if port redirection has happened.

/sys/class/scsi_host/host<n>/id_ext

Defined on file *sysfs-driver-ib_srp*

Eight-byte identifier extension portion of the 16-byte target port identifier.

/sys/class/scsi_host/host<n>/ioc_guid

Defined on file *sysfs-driver-ib_srp*

Eight-byte I/O controller GUID portion of the 16-byte target port identifier.

/sys/class/scsi_host/host<n>/local_ib_device

Defined on file *sysfs-driver-ib_srp*

Name of the InfiniBand HCA used for communicating with the SRP target.

/sys/class/scsi_host/host<n>/local_ib_port

Defined on file *sysfs-driver-ib_srp*

Number of the HCA port used for communicating with the SRP target.

/sys/class/scsi_host/host<n>/orig_dgid

Defined on file *sysfs-driver-ib_srp*

InfiniBand destination GID specified in the parameters written to the add_target sysfs attribute.

/sys/class/scsi_host/host<n>/pkey

Defined on file *sysfs-driver-ib_srp*

A 16-bit number representing the InfiniBand partition key used for communication with the SRP target.

/sys/class/scsi_host/host<n>/req_lim

Defined on file *sysfs-driver-ib_srp*

Number of requests ib_srp can send to the target before it has to wait for more credits. For more information see also the SRP credit algorithm in the SRP specification.

/sys/class/scsi_host/host<n>/service_id

Defined on file *sysfs-driver-ib_srp*

InfiniBand service ID used for establishing communication with the SRP target.

/sys/class/scsi_host/host<n>/sgid

Defined on file *sysfs-driver-ib_srp*

InfiniBand GID of the source port used for communication with the SRP target.

/sys/class/scsi_host/host<n>/zero_req_lim

Defined on file *sysfs-driver-ib_srp*

Number of times the initiator had to wait before sending a request to the target because it ran out of credits. For more information see also the SRP credit algorithm in the SRP specification.

/sys/class/srp_remote_ports/port-<h>:<n>/delete

Defined on file *sysfs-transport-srp*

Instructs an SRP initiator to disconnect from a target and to remove all LUNs imported from that target.

/sys/class/srp_remote_ports/port-<h>:<n>/dev_loss_tmo

Defined on file *sysfs-transport-srp*

Number of seconds the SCSI layer will wait after a transport layer error has been observed before removing a target port. Zero means immediate removal. Setting this attribute to "off" will disable the dev_loss timer.

/sys/class/srp_remote_ports/port-<h>:<n>/fast_io_fail_tmo

Defined on file *sysfs-transport-srp*

Number of seconds the SCSI layer will wait after a transport layer error has been observed before failing I/O. Zero means failing I/O immediately. Setting this attribute to "off" will disable the fast_io_fail timer.

/sys/class/srp_remote_ports/port-<h>:<n>/port_id

Defined on file *sysfs-transport-srp*

16-byte local SRP port identifier in hexadecimal format. An example: 4c:49:4e:55:58:20:56:49:4f:00:00:00:00:00:00:00.

/sys/class/srp_remote_ports/port-<h>:<n>/reconnect_delay

Defined on file *sysfs-transport-srp*

Number of seconds the SCSI layer will wait after a reconnect attempt failed before retrying. Setting this attribute to "off" will disable time-based reconnecting.

/sys/class/srp_remote_ports/port-<h>:<n>/roles

Defined on file *sysfs-transport-srp*

Role of the remote port. Either "SRP Initiator" or "SRP Target".

/sys/class/srp_remote_ports/port-<h>:<n>/state

Defined on file *sysfs-transport-srp*

State of the transport layer used for communication with the remote port. "running" if the transport layer is operational; "blocked" if a transport layer error has been encountered but the fast_io_fail_tmo timer has not yet fired; "fail-fast" after the fast_io_fail_tmo timer has fired and before the "dev_loss_tmo" timer has fired; "lost" after the "dev_loss_tmo" timer has fired and before the port is finally removed.

/sys/class/tpm/tpmX/device/

Defined on file *sysfs-class-tpm*

The device/ directory under a specific TPM instance exposes the properties of that TPM chip

/sys/class/tpm/tpmX/device/active

Defined on file *sysfs-class-tpm*

The "active" property prints a '1' if the TPM chip is accepting commands. An inactive TPM chip still contains all the state of an active chip (Storage Root Key, NVRAM, etc), and can be visible

to the OS, but will only accept a restricted set of commands. See the TPM Main Specification part 2, Structures, section 17 for more information on which commands are available.

/sys/class/tpm/tpmX/device/cancel

Defined on file *sysfs-class-tpm*

The ”cancel” property allows you to cancel the currently pending TPM command. Writing any value to cancel will call the TPM vendor specific cancel operation.

/sys/class/tpm/tpmX/device/caps

Defined on file *sysfs-class-tpm*

The ”caps” property contains TPM manufacturer and version info.

Example output:

```
Manufacturer: 0x53544d20
TCG version: 1.2
Firmware version: 8.16
```

Manufacturer is a hex dump of the 4 byte manufacturer info space in a TPM. TCG version shows the TCG TPM spec level that the chip supports. Firmware version is that of the chip and is manufacturer specific.

/sys/class/tpm/tpmX/device/durations

Defined on file *sysfs-class-tpm*

The ”durations” property shows the 3 vendor-specific values used to wait for a short, medium and long TPM command. All TPM commands are categorized as short, medium or long in execution time, so that the driver doesn't have to wait any longer than necessary before starting to poll for a result.

Example output:

```
3015000 4508000 180995000 [original]
```

Here the short, medium and long durations are displayed in usecs. ”[original]” indicates that the values are displayed unmodified from when they were queried from the chip. Durations can be modified in the case where a buggy chip reports them in msec instead of usec and they need to be scaled to be displayed in usecs. In this case ”[adjusted]” will be displayed in place of ”[original]”.

/sys/class/tpm/tpmX/device/enabled

Defined on file *sysfs-class-tpm*

The ”enabled” property prints a '1' if the TPM chip is enabled, meaning that it should be visible to the OS. This property may be visible but produce a '0' after some operation that disables the TPM.

/sys/class/tpm/tpmX/device/owned

Defined on file *sysfs-class-tpm*

The "owned" property produces a '1' if the TPM_TakeOwnership ordinal has been executed successfully in the chip. A '0' indicates that ownership hasn't been taken.

/sys/class/tpm/tpmX/device/pcrs

Defined on file *sysfs-class-tpm*

The "pcrs" property will dump the current value of all Platform Configuration Registers in the TPM. Note that since these values may be constantly changing, the output is only valid for a snapshot in time.

Example output:

```
PCR-00: 3A 3F 78 0F 11 A4 B4 99 69 FC AA 80 CD 6E 39 57 C3 3B 22 75
PCR-01: 3A 3F 78 0F 11 A4 B4 99 69 FC AA 80 CD 6E 39 57 C3 3B 22 75
PCR-02: 3A 3F 78 0F 11 A4 B4 99 69 FC AA 80 CD 6E 39 57 C3 3B 22 75
PCR-03: 3A 3F 78 0F 11 A4 B4 99 69 FC AA 80 CD 6E 39 57 C3 3B 22 75
PCR-04: 3A 3F 78 0F 11 A4 B4 99 69 FC AA 80 CD 6E 39 57 C3 3B 22 75
...
```

The number of PCRs and hex bytes needed to represent a PCR value will vary depending on TPM chip version. For TPM 1.1 and 1.2 chips, PCRs represent SHA-1 hashes, which are 20 bytes long. Use the "caps" property to determine TPM version.

/sys/class/tpm/tpmX/device/pubek

Defined on file *sysfs-class-tpm*

The "pubek" property will return the TPM's public endorsement key if possible. If the TPM has had ownership established and is version 1.2, the pubek will not be available without the owner's authorization. Since the TPM driver doesn't store any secrets, it can't authorize its own request for the pubek, making it unaccessible. The public endorsement key is gener- ated at TPM manufacture time and exists for the life of the chip.

Example output:

```
Algorithm: 00 00 00 01
Encscheme: 00 03
Sigscheme: 00 01
Parameters: 00 00 08 00 00 00 00 02 00 00 00 00
Modulus length: 256
Modulus:
B4 76 41 82 C9 20 2C 10 18 40 BC 8B E5 44 4C 6C
3A B2 92 0C A4 9B 2A 83 EB 5C 12 85 04 48 A0 B6
1E E4 81 84 CE B2 F2 45 1C F0 85 99 61 02 4D EB
86 C4 F7 F3 29 60 52 93 6B B2 E5 AB 8B A9 09 E3
D7 0E 7D CA 41 BF 43 07 65 86 3C 8C 13 7A D0 8B
82 5E 96 0B F8 1F 5F 34 06 DA A2 52 C1 A9 D5 26
```

```
0F F4 04 4B D9 3F 2D F2 AC 2F 74 64 1F 8B CD 3E
1E 30 38 6C 70 63 69 AB E2 50 DF 49 05 2E E1 8D
6F 78 44 DA 57 43 69 EE 76 6C 38 8A E9 8E A3 F0
A7 1F 3C A8 D0 12 15 3E CA 0E BD FA 24 CD 33 C6
47 AE A4 18 83 8E 22 39 75 93 86 E6 FD 66 48 B6
10 AD 94 14 65 F9 6A 17 78 BD 16 53 84 30 BF 70
E0 DC 65 FD 3C C6 B0 1E BF B9 C1 B5 6C EF B1 3A
F8 28 05 83 62 26 11 DC B4 6B 5A 97 FF 32 26 B6
F7 02 71 CF 15 AE 16 DD D1 C1 8E A8 CF 9B 50 7B
C3 91 FF 44 1E CF 7C 39 FE 17 77 21 20 BD CE 9B
```

Possible values:

```
Algorithm:    TPM_ALG_RSA                     (1)
Encscheme:    TPM_ES_RSAESPKCSv15             (2)
              TPM_ES_RSAESOAEP_SHA1_MGF1      (3)
Sigscheme:    TPM_SS_NONE                     (1)
Parameters, a byte string of 3 u32 values:
      Key Length (bits):      00 00 08 00     (2048)
      Num primes:             00 00 00 02     (2)
      Exponent Size:          00 00 00 00     (0 means the
                                               default exp)
Modulus Length: 256 (bytes)
Modulus:      The 256 byte Endorsement Key modulus
```

/sys/class/tpm/tpmX/device/temp_deactivated

Defined on file *sysfs-class-tpm*

The ”temp_deactivated” property returns a '1' if the chip has been temporarily deactivated, usually until the next power cycle. Whether a warm boot (reboot) will clear a TPM chip from a temp_deactivated state is platform specific.

/sys/class/tpm/tpmX/device/timeouts

Defined on file *sysfs-class-tpm*

The ”timeouts” property shows the 4 vendor-specific values for the TPM's interface spec timeouts. The use of these timeouts is defined by the TPM interface spec that the chip conforms to.

Example output:

```
750000 750000 750000 750000 [original]
```

The four timeout values are shown in usecs, with a trailing ”[original]” or ”[adjusted]” depending on whether the values were scaled by the driver to be reported in usec from msecs.

/sys/class/tpm/tpmX/pcr-<H>/<N>

Defined on file *sysfs-class-tpm*

produces output in compact hex representation for PCR number N from hash bank H. N is the numeric value of the PCR number and H is the crypto string representation of the hash

Example output:

```
cat /sys/class/tpm/tpm0/pcr-sha256/7
2ED93F199692DC6788EFA6A1FE74514AB9760B2A6CEEAEF6C808C13E4ABB0D42
```

/sys/class/tpm/tpmX/tpm_version_major

Defined on file *sysfs-class-tpm*

The "tpm_version_major" property shows the TCG spec major version implemented by the TPM device.

Example output:

```
2
```

/sys/class/ubi/

Defined on file *sysfs-class-ubi*

The ubi/ class sub-directory belongs to the UBI subsystem and provides general UBI information, per-UBI device information and per-UBI volume information.

/sys/class/ubi/ubiX/avail_eraseblocks

Defined on file *sysfs-class-ubi*

Amount of available logical eraseblock. For example, one may create a new UBI volume which has this amount of logical eraseblocks.

/sys/class/ubi/ubiX/bad_peb_count

Defined on file *sysfs-class-ubi*

Count of bad physical eraseblocks on the underlying MTD device.

/sys/class/ubi/ubiX/bgt_enabled

Defined on file *sysfs-class-ubi*

Contains ASCII "0n" if the UBI background thread is disabled, and ASCII "1n" if it is enabled.

/sys/class/ubi/ubiX/dev

Defined on file *sysfs-class-ubi*

Major and minor numbers of the character device corresponding to this UBI device (in <major>:<minor> format).

/sys/class/ubi/ubiX/eraseblock_size

Defined on file *sysfs-class-ubi*

Maximum logical eraseblock size this UBI device may provide. UBI volumes may have smaller logical eraseblock size because of their alignment.

/sys/class/ubi/ubiX/max_ec

Defined on file *sysfs-class-ubi*

Maximum physical eraseblock erase counter value.

/sys/class/ubi/ubiX/max_vol_count

Defined on file *sysfs-class-ubi*

Maximum number of volumes which this UBI device may have.

/sys/class/ubi/ubiX/min_io_size

Defined on file *sysfs-class-ubi*

Minimum input/output unit size. All the I/O may only be done in fractions of the contained number.

/sys/class/ubi/ubiX/mtd_num

Defined on file *sysfs-class-ubi*

Number of the underlying MTD device.

/sys/class/ubi/ubiX/reserved_for_bad

Defined on file *sysfs-class-ubi*

Number of physical eraseblocks reserved for bad block handling.

/sys/class/ubi/ubiX/ro_mode

Defined on file *sysfs-class-ubi*

Contains ASCII "1n" if the read-only flag is set on this device, and "0n" if it is cleared. UBI devices mark themselves as read-only when they detect an unrecoverable error.

/sys/class/ubi/ubiX/total_eraseblocks

Defined on file *sysfs-class-ubi*

Total number of good (not marked as bad) physical eraseblocks on the underlying MTD device.

/sys/class/ubi/ubiX/ubiX_Y/

Defined on file *sysfs-class-ubi*

The /sys/class/ubi/ubiX/ubiX_0/, /sys/class/ubi/ubiX/ubiX_1/, etc directories describe UBI volumes on UBI device X (volumes 0, 1, etc).

/sys/class/ubi/ubiX/ubiX_Y/alignment

Defined on file *sysfs-class-ubi*

Volume alignment - the value the logical eraseblock size of this volume has to be aligned on. For example, 2048 means that logical eraseblock size is multiple of 2048. In other words, volume logical eraseblock size is UBI device logical eraseblock size aligned to the alignment value.

/sys/class/ubi/ubiX/ubiX_Y/corrupted

Defined on file *sysfs-class-ubi*

Contains ASCII "0n" if the UBI volume is OK, and ASCII "1n" if it is corrupted (e.g., due to an interrupted volume update).

/sys/class/ubi/ubiX/ubiX_Y/data_bytes

Defined on file *sysfs-class-ubi*

The amount of data this volume contains. This value makes sense only for static volumes, and for dynamic volume it equivalent to the total volume size in bytes.

/sys/class/ubi/ubiX/ubiX_Y/dev

Defined on file *sysfs-class-ubi*

Major and minor numbers of the character device corresponding to this UBI volume (in <major>:<minor> format).

/sys/class/ubi/ubiX/ubiX_Y/name

Defined on file *sysfs-class-ubi*

Volume name.

/sys/class/ubi/ubiX/ubiX_Y/reserved_ebs

Defined on file *sysfs-class-ubi*

Count of physical eraseblock reserved for this volume. Equivalent to the volume size in logical eraseblocks.

/sys/class/ubi/ubiX/ubiX_Y/type

Defined on file *sysfs-class-ubi*

Volume type. Contains ASCII ”dynamicn” for dynamic volumes and ”staticn” for static volumes.

/sys/class/ubi/ubiX/ubiX_Y/upd_marker

Defined on file *sysfs-class-ubi*

Contains ASCII ”0n” if the update marker is not set for this volume, and ”1n” if it is set. The update marker is set when volume update starts, and cleaned when it ends. So the presence of the update marker indicates that the volume is being updated at the moment of the update was interrupted. The later may be checked using the ”corrupted” sysfs file.

/sys/class/ubi/ubiX/ubiX_Y/usable_eb_size

Defined on file *sysfs-class-ubi*

Logical eraseblock size of this volume. Equivalent to logical eraseblock size of the device aligned on the volume alignment value.

/sys/class/ubi/ubiX/volumes_count

Defined on file *sysfs-class-ubi*

Count of volumes on this UBI device.

/sys/class/ubi/version

Defined on file *sysfs-class-ubi*

This file contains version of the latest supported UBI on-media format. Currently it is 1, and there is no plan to change this. However, if in the future UBI needs on-flash format changes which cannot be done in a compatible manner, a new format version will be added. So this is a mechanism for possible future backward-compatible (but forward-incompatible) improvements.

/sys/class/ubiX/

Defined on file *sysfs-class-ubi*

The /sys/class/ubi0, /sys/class/ubi1, etc directories describe UBI devices (UBI device 0, 1, etc). They contain general UBI device information and per UBI volume information (each UBI device may have many UBI volumes)

/sys/class/udc/<udc>/a_alt_hnp_support

Defined on file *sysfs-class-udc*

Indicates if an OTG A-Host supports HNP at an alternate port.

/sys/class/udc/<udc>/a_hnp_support

Defined on file *sysfs-class-udc*

Indicates if an OTG A-Host supports HNP at this port.

/sys/class/udc/<udc>/b_hnp_enable

Defined on file *sysfs-class-udc*

Indicates if an OTG A-Host enabled HNP support.

/sys/class/udc/<udc>/current_speed

Defined on file *sysfs-class-udc*

Indicates the current negotiated speed at this port.

/sys/class/udc/<udc>/function

Defined on file *sysfs-class-udc*

Prints out name of currently running USB Gadget Driver.

/sys/class/udc/<udc>/is_a_peripheral

Defined on file *sysfs-class-udc*

Indicates that this port is the default Host on an OTG session but HNP was used to switch roles.

/sys/class/udc/<udc>/is_otg

Defined on file *sysfs-class-udc*

Indicates that this port support OTG.

/sys/class/udc/<udc>/maximum_speed

Defined on file *sysfs-class-udc*

Indicates the maximum USB speed supported by this port.

/sys/class/udc/<udc>/soft_connect

Defined on file *sysfs-class-udc*

Allows users to disconnect data pullup resistors thus causing a logical disconnection from the USB Host.

/sys/class/udc/<udc>/srp

Defined on file *sysfs-class-udc*

Allows users to manually start Session Request Protocol.

/sys/class/udc/<udc>/state

Defined on file *sysfs-class-udc*

Indicates current state of the USB Device Controller. Valid states are: 'not-attached', 'attached', 'powered', 'reconnecting', 'unauthenticated', 'default', 'addressed', 'configured', and 'suspended'; however not all USB Device Controllers support reporting all states.

5.1.6 Symbols under /sys/devices

/sys/devices/*/dev

Defined on file *sysfs-devices*

Major and minor numbers of the character device corresponding to the device (in <major>:<minor> format).

/sys/devices/*/devspec

Defined on file *sysfs-devices*

If CONFIG_OF is enabled, then this file is present. When read, it returns full name of the device node.

/sys/devices/*/obppath

Defined on file *sysfs-devices*

If CONFIG_OF is enabled, then this file is present. When read, it returns full name of the device node.

/sys/devices/*/of_node

Defined on file *sysfs-devices*

Any device associated with a device-tree node will have an of_path symlink pointing to the corresponding device node in /sys/firmware/devicetree/

/sys/devices/pciXXXX:XX/0000:XX:XX.X/dma/dma<n>chan<n>/quickdata/cap

Defined on file *sysfs-driver-dma-ioatdma*

Capabilities the DMA supports.Currently there are DMA_PQ, DMA_PQ_VAL, DMA_XOR,DMA_XOR_VAL,DMA_INTERRUPT.

/sys/devices/pciXXXX:XX/0000:XX:XX.X/dma/dma<n>chan<n>/quickdata/intr_coalesce

Defined on file *sysfs-driver-dma-ioatdma*

Tune-able interrupt delay value per channel basis.

/sys/devices/pciXXXX:XX/0000:XX:XX.X/dma/dma<n>chan<n>/quickdata/ring_active

Defined on file *sysfs-driver-dma-ioatdma*

The number of descriptors active in the ring.

/sys/devices/pciXXXX:XX/0000:XX:XX.X/dma/dma<n>chan<n>/quickdata/ring_size

Defined on file *sysfs-driver-dma-ioatdma*

Descriptor ring size, total number of descriptors available.

/sys/devices/pciXXXX:XX/0000:XX:XX.X/dma/dma<n>chan<n>/quickdata/version

Defined on file *sysfs-driver-dma-ioatdma*

Version of ioatdma device.

/sys/devices/platform/firmware\:zynqmp-firmware/feature_config_id

Defined on file *sysfs-driver-firmware-zynqmp*

This sysfs interface allows user to configure features at runtime. The user can enable or disable features running at firmware as well as the user can configure the parameters of the features at runtime. The supported features are over temperature and external watchdog. Here, the external watchdog is completely different than the /dev/watchdog as the external watchdog is running on the firmware and it is used to monitor the health of firmware not APU(Linux). Also, the external watchdog is interfaced outside of the zynqmp soc.

The supported config ids are for the feature configuration is, 1. PM_FEATURE_OVERTEMP_STATUS = 1, the user can enable or disable the over temperature feature. 2. PM_FEATURE_OVERTEMP_VALUE = 2, the user can configure the over temperature limit in Degree Celsius. 3. PM_FEATURE_EXTWDT_STATUS = 3, the user can enable or disable the external watchdog feature. 4. PM_FEATURE_EXTWDT_VALUE = 4, the user can configure the external watchdog feature.

Usage:

Select over temperature config ID to enable/disable feature # echo 1 > /sys/devices/platform/firmware:zynqmp-firmware/feature_config_id

Check over temperature config ID is selected or not # cat /sys/devices/platform/firmware:zynqmp-firmware/feature_config_id The expected result is 1.

Select over temperature config ID to configure OT limit # echo 2 > /sys/devices/platform/firmware:zynqmp-firmware/feature_config_id

Check over temperature config ID is selected or not # cat /sys/devices/platform/firmware:zynqmp-firmware/feature_config_id The expected result is 2.

Select external watchdog config ID to enable/disable feature # echo 3 > /sys/devices/platform/firmware:zynqmp-firmware/feature_config_id

Check external watchdog config ID is selected or not # cat /sys/devices/platform/firmware:zynqmp-firmware/feature_config_id The expected result is 3.

Select external watchdog config ID to configure time interval # echo 4 > /sys/devices/platform/firmware:zynqmp-firmware/feature_config_id

Check external watchdog config ID is selected or not # cat /sys/devices/platform/firmware:zynqmp-firmware/feature_config_id The expected result is 4.

Users: Xilinx

/sys/devices/platform/firmware\:zynqmp-firmware/feature_config_value

Defined on file *sysfs-driver-firmware-zynqmp*

This sysfs interface allows to configure features at runtime. The user can enable or disable features running at firmware. Also, the user can configure the parameters of the features at runtime. The supported features are over temperature and external watchdog. Here, the external watchdog is completely different than the /dev/watchdog as the external watchdog is running on the firmware and it is used to monitor the health of firmware not APU(Linux). Also, the external watchdog is interfaced outside of the zynqmp soc.

By default the features are disabled in the firmware. The user can enable features by querying appropriate config id of the features.

The default limit for the over temperature is 90 Degree Celsius. The default timer interval for the external watchdog is 570ms.

The supported config ids are for the feature configuration is, 1. PM_FEATURE_OVERTEMP_STATUS = 1, the user can enable or disable the over temperature feature. 2. PM_FEATURE_OVERTEMP_VALUE = 2, the user can configure the over temperature limit in Degree Celsius. 3. PM_FEATURE_EXTWDT_STATUS = 3, the user can enable or disable the external watchdog feature. 4. PM_FEATURE_EXTWDT_VALUE = 4, the user can configure the external watchdog feature.

Usage:

Enable over temperature feature # echo 1 > /sys/devices/platform/firmware:zynqmp-firmware/feature_config_id # echo 1 > /sys/devices/platform/firmware:zynqmp-firmware/feature_config_value

Check whether the over temperature feature is enabled or not # cat /sys/devices/platform/firmware:zynqmp-firmware/feature_config_value The expected result is 1.

Disable over temperature feature # echo 1 > /sys/devices/platform/firmware:zynqmp-firmware/feature_config_id # echo 0 > /sys/devices/platform/firmware:zynqmp-firmware/feature_config_value

Check whether the over temperature feature is disabled or not # cat /sys/devices/platform/firmware:zynqmp-firmware/feature_config_value The expected result is 0.

Configure over temperature limit to 50 Degree Celsius # echo 2 > /sys/devices/platform/firmware:zynqmp-firmware/feature_config_id # echo 50 > /sys/devices/platform/firmware:zynqmp-firmware/feature_config_value

Check whether the over temperature limit is configured or not # cat /sys/devices/platform/firmware:zynqmp-firmware/feature_config_value The expected result

is 50.

Enable external watchdog feature # echo 3 > /sys/devices/platform/firmware:zynqmp-firmware/feature_config_id # echo 1 > /sys/devices/platform/firmware:zynqmp-firmware/feature_config_value

Check whether the external watchdog feature is enabled or not # cat /sys/devices/platform/firmware:zynqmp-firmware/feature_config_value The expected result is 1.

Disable external watchdog feature # echo 3 > /sys/devices/platform/firmware:zynqmp-firmware/feature_config_id # echo 0 > /sys/devices/platform/firmware:zynqmp-firmware/feature_config_value

Check whether the external watchdog feature is disabled or not # cat /sys/devices/platform/firmware:zynqmp-firmware/feature_config_value The expected result is 0.

Configure external watchdog timer interval to 500ms # echo 4 > /sys/devices/platform/firmware:zynqmp-firmware/feature_config_id # echo 500 > /sys/devices/platform/firmware:zynqmp-firmware/feature_config_value

Check whether the external watchdog timer interval is configured or not # cat /sys/devices/platform/firmware:zynqmp-firmware/feature_config_value The expected result is 500.

Users: Xilinx

/sys/devices/platform/firmware\:zynqmp-firmware/ggs*

Defined on file *sysfs-driver-firmware-zynqmp*

Read/Write PMU global general storage register value, GLOBAL_GEN_STORAGE{0:3}. Global general storage register that can be used by system to pass information between masters.

The register is reset during system or power-on resets. Three registers are used by the FSBL and other Xilinx software products: GLOBAL_GEN_STORAGE{4:6}.

Usage:

```
# cat /sys/devices/platform/firmware\:zynqmp-firmware/ggs0
# echo <value> > /sys/devices/platform/firmware\:zynqmp-firmware/ggs0
```

Example:

```
# cat /sys/devices/platform/firmware\:zynqmp-firmware/ggs0
# echo 0x1234ABCD > /sys/devices/platform/firmware\:zynqmp-firmware/ggs0
```

Users: Xilinx

/sys/devices/platform/firmware\:zynqmp-firmware/health_status

Defined on file *sysfs-driver-firmware-zynqmp*

This sysfs interface allows to set the health status. If PMUFW is compiled with CHECK_HEALTHY_BOOT, it will check the healthy bit on FPD WDT expiration. If healthy bit is

set by a user application running in Linux, PMUFW will do APU only restart. If healthy bit is not set during FPD WDT expiration, PMUFW will do system restart.

Usage:

Set healthy bit:

```
# echo 1 > /sys/devices/platform/firmware\:zynqmp-firmware/health_status
```

Unset healthy bit:

```
# echo 0 > /sys/devices/platform/firmware\:zynqmp-firmware/health_status
```

Users: Xilinx

/sys/devices/platform/firmware\:zynqmp-firmware/pggs*

Defined on file *sysfs-driver-firmware-zynqmp*

Read/Write PMU persistent global general storage register value, PERS_GLOB_GEN_STORAGE{0:3}. Persistent global general storage register that can be used by system to pass information between masters.

This register is only reset by the power-on reset and maintains its value through a system reset. Four registers are used by the FSBL and other Xilinx software products: PERS_GLOB_GEN_STORAGE{4:7}. Register is reset only by a POR reset.

Usage:

```
# cat /sys/devices/platform/firmware\:zynqmp-firmware/pggs0
# echo <value> > /sys/devices/platform/firmware\:zynqmp-firmware/pggs0
```

Example:

```
# cat /sys/devices/platform/firmware\:zynqmp-firmware/pggs0
# echo 0x1234ABCD > /sys/devices/platform/firmware\:zynqmp-firmware/pggs0
```

Users: Xilinx

/sys/devices/platform/firmware\:zynqmp-firmware/shutdown_scope

Defined on file *sysfs-driver-firmware-zynqmp*

This sysfs interface allows to set the shutdown scope for the next shutdown request. When the next shutdown is performed, the platform specific portion of PSCI-system_off can use the chosen shutdown scope.

Following are available shutdown scopes(subtypes):

subsystem:
Only the APU along with all of its peripherals not used by other processing units will be shut down. This may result in the FPD power domain being shut down provided that no other processing unit uses FPD peripherals or DRAM.

ps_only:
The complete PS will be shut down, including the RPU, PMU, etc. Only the PL domain (FPGA) remains untouched.

system:
The complete system/device is shut down.

Usage:

```
# cat /sys/devices/platform/firmware\:zynqmp-firmware/shutdown_scope
# echo <scope> > /sys/devices/platform/firmware\:zynqmp-firmware/shutdown_scope
```

Example:

```
# cat /sys/devices/platform/firmware\:zynqmp-firmware/shutdown_scope
# echo "subsystem" > /sys/devices/platform/firmware\:zynqmp-firmware/shutdown_
↪scope
```

Users: Xilinx

/sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/agb_spi_burn_en
/sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/fpga_spi_burn_en

Defined on file *sysfs-driver-mlxreg-io*

These files allow gearboxes and FPGA SPI flash burning. The attributes are set 1 to enable burning, 0 - to disable. If the system is in locked-down mode writing these files will not be allowed. The purpose of these files to allow line card Gearboxes and FPGA burning during production flow.

The file is read/write.

/sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/cpld1_pn
/sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/cpld1_version
/sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/cpld1_version_min

Defined on file *sysfs-driver-mlxreg-io*

These files show with which CPLD major and minor versions and part number has been burned CPLD device on line card.

The files are read only.

/sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/cpld_upgrade_en
/sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/fpga_upgrade_en

Defined on file *sysfs-driver-mlxreg-io*

These files allow CPLD and FPGA burning. Value 1 in file means burning is enabled, 0 - otherwise. If the system is in locked-down mode writing these files will not be allowed. The purpose of these files to allow line card CPLD and FPGA upgrade through the JTAG daisy-chain.

The files are read/write.

/sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/fpga1_pn
/sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/fpga1_version
/sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/fpga1_version_min

Defined on file *sysfs-driver-mlxreg-io*

These files show with which FPGA major and minor versions and part number has been burned FPGA device on line card.

The files are read only.

/sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/max_power
/sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/config

Defined on file *sysfs-driver-mlxreg-io*

These files provide the maximum powered required for line card feeding and line card configuration Id.

The files are read only.

/sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/qsfp_pwr_en
/sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/pwr_en

Defined on file *sysfs-driver-mlxreg-io*

These files allow to power on/off all QSFP ports and whole line card. The attributes are set 1 for power on, 0 - for power off.

The files are read/write.

/sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/reset_aux_pwr_or_ref
/sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/reset_dc_dc_pwr_fail
/sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/reset_fpga_not_done
/sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/reset_from_chassis
/sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/reset_line_card
/sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/reset_pwr_off_from_chassis

Defined on file *sysfs-driver-mlxreg-io*

These files show the line reset cause, as following: power auxiliary outage or power refresh, DC-to-DC power failure, FPGA reset failed, line card reset failed, power off from chassis. Value 1 in file means this is reset cause, 0 - otherwise. Only one of the above causes could be 1 at the same time, representing only last reset cause.

The files are read only.

/sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/vpd_wp

Defined on file *sysfs-driver-mlxreg-io*

This file allow to overwrite line card VPD hardware write protection mode. When attribute is set 1 - write protection is disabled, when 0 - enabled. Default is 0. If the system is in locked-down mode writing this file will not be allowed. The purpose if this file is to allow line card VPD burning during production flow.

The file is read/write.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/asic2_health

Defined on file *sysfs-driver-mlxreg-io*

This file shows 2-nd ASIC health status. The possible values are: 0 - health failed, 2 - health OK, 3 - ASIC in booting state.

The file is read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/asic_health

Defined on file *sysfs-driver-mlxreg-io*

This file shows ASIC health status. The possible values are: 0 - health failed, 2 - health OK, 3 - ASIC in booting state.

The files are read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/asic_pg_fail

Defined on file *sysfs-driver-mlxreg-io*

This file shows ASIC Power Good status. Value 1 in file means ASIC Power Good failed, 0 - otherwise.

The file is read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/asic_reset
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/asic2_reset

Defined on file *sysfs-driver-mlxreg-io*

These files allow to each of ASICs by writing 1.

The files are write only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/bios_active_image
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/bios_auth_fail
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/bios_upgrade_fail

Defined on file *sysfs-driver-mlxreg-io*

The files represent BIOS statuses:

bios_active_image: location of current active BIOS image: 0: Top, 1: Bottom. The reported value should correspond to value expected by OS in case of BIOS safe mode is 0. This bit is related to Intel top-swap feature of DualBios on the same flash.

bios_auth_fail: BIOS upgrade is failed because provided BIOS image is not signed correctly.

bios_upgrade_fail: BIOS upgrade is failed by some other reason not because authentication. For example due to physical SPI flash problem.

The files are read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/clk_brd1_boot_fail
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/clk_brd2_boot_fail
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/clk_brd_fail

Defined on file *sysfs-driver-mlxreg-io*

These files are related to clock boards status in system. - clk_brd1_boot_fail: warning about 1-st clock board failed to boot from CI. - clk_brd2_boot_fail: warning about 2-nd clock board failed to boot from CI. - clk_brd_fail: error about common clock board boot failure.

The files are read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/clk_brd_prog_en

Defined on file *sysfs-driver-mlxreg-io*

This file enables programming of clock boards. Default is 0 (programming disabled). If the system is in locked-down mode writing this file will not be allowed.

The file is read/write.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/comm_chnl_ready

Defined on file *sysfs-driver-mlxreg-io*

This file is used to indicate remote end (for example BMC) that system host CPU is ready for sending telemetry data to remote end. For indication the file should be written 1.

The file is write only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/config1
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/config2

Defined on file *sysfs-driver-mlxreg-io*

These files show system static topology identification like system's static I2C topology, number and type of FPGA devices within the system and so on.

The files are read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/config3

Defined on file *sysfs-driver-mlxreg-io*

The file indicates COME module hardware configuration. The value is pushed by hardware through GPIO pins. The purpose is to expose some minor BOM changes for the same system SKU.

The file is read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld1_pn
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld2_pn
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld3_pn
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld4_pn
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld1_version_min
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld2_version_min
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld3_version_min
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld4_version_min

Defined on file *sysfs-driver-mlxreg-io*

These files show with which CPLD part numbers and minor versions have been burned CPLD devices equipped on a system.

The files are read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld1_version
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld2_version

Defined on file *sysfs-driver-mlxreg-io*

These files show with which CPLD versions have been burned on carrier and switch boards.

The files are read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld3_version

Defined on file *sysfs-driver-mlxreg-io*

These files show with which CPLD versions have been burned on LED or Gearbox board.

The files are read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld4_version

Defined on file *sysfs-driver-mlxreg-io*

These files show with which CPLD versions have been burned on LED board.

The files are read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld5_pn
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld5_version
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld5_version_min

Defined on file *sysfs-driver-mlxreg-io*

These files show with which CPLD part numbers, version and minor versions have been burned the 5-th CPLD device equipped on a system.

The files are read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/erot1_ap_reset
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/erot2_ap_reset

Defined on file *sysfs-driver-mlxreg-io*

These files aim to monitor the status of the External Root of Trust (EROT) processor's RESET output to the Application Processor (AP). By reading this file, could be determined if the EROT has invalidated or revoked AP Firmware, at which point it will hold the AP in RESET until a valid firmware is loaded. This protects the AP from running an unauthorized firmware. In the normal flow, the AP reset should be released after the EROT validates the integrity of the FW, and it should be done so as quickly as possible so that the AP boots before the CPU starts to communicate to each ASIC.

The files are read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/erot1_recovery
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/erot2_recovery
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/erot1_reset
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/erot2_reset

Defined on file *sysfs-driver-mlxreg-io*

These files aim to perform External Root of Trust (EROT) recovery sequence after EROT device failure. These EROT devices protect ASICs from unauthorized access and in normal flow their reset should be released with system power – earliest power up stage, so that EROTs can begin boot and authentication process before CPU starts to communicate to ASICs. Issuing a reset

to the EROT while asserting the recovery signal will cause the EROT Application Processor to enter recovery mode so that the EROT FW can be updated/recovered. For reset/recovery the related file should be toggled by 1/0.

The files are read/write.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/erot1_wp
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/erot2_wp

Defined on file *sysfs-driver-mlxreg-io*

These files allow access to External Root of Trust (EROT) for reset and recovery sequence after EROT device failure. Default is 0 (programming disabled). If the system is in locked-down mode writing this file will not be allowed.

The files are read/write.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/fan_dir

Defined on file *sysfs-driver-mlxreg-io*

This file shows the system fans direction: forward direction - relevant bit is set 0; reversed direction - relevant bit is set 1.

The files are read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/jtag_cap

Defined on file *sysfs-driver-mlxreg-io*

This file indicates the available method of CPLD/FPGA devices field update through the JTAG chain:

b00 - field update through LPC bus register memory space. b01 - Reserved. b10 - Reserved. b11 - field update through CPU GPIOs bit-banging.

The file is read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/jtag_enable

Defined on file *sysfs-driver-mlxreg-io*

These files enable and disable the access to the JTAG domain. By default access to the JTAG domain is disabled.

The file is read/write.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc1_enable
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc2_enable
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc3_enable
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc4_enable
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc5_enable
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc6_enable
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc7_enable
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc8_enable

Defined on file *sysfs-driver-mlxreg-io*

These files allow line cards enable state control. Expected behavior: When lc{n}_enable is written 1, related line card is released from the reset state, when 0 - is hold in reset state.

The files are read/write.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc1_pwr
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc2_pwr
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc3_pwr
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc4_pwr
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc5_pwr
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc6_pwr
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc7_pwr
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc8_pwr

Defined on file *sysfs-driver-mlxreg-io*

These files switching line cards power on and off. Expected behavior: When lc{n}_pwr is written 1, related line card is powered on, when written 0 - powered off.

The files are read/write.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc1_rst_mask
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc2_rst_mask
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc3_rst_mask
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc4_rst_mask
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc5_rst_mask
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc6_rst_mask
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc7_rst_mask
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc8_rst_mask

Defined on file *sysfs-driver-mlxreg-io*

These files clear line card reset bit enforced by ASIC, when it sets it due to some abnormal ASIC behavior. Expected behavior: When lc{n}_rst_mask is written 1, related line card reset bit is cleared, when written 0 - no effect.

The files are write only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lid_open

Defined on file *sysfs-driver-mlxreg-io*

1 - indicates that system lid is opened, otherwise 0.

The file is read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/mac_reset

Defined on file *sysfs-driver-mlxreg-io*

This file allows to reset ASIC MT52132 when attribute is set 0 due to some abnormal ASIC behavior. Expected behavior: When mac_reset is written 1, the ASIC MT52132 is released from the reset state, when 0 - is hold in reset state.

The files are read/write.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/os_started

Defined on file *sysfs-driver-mlxreg-io*

This file, when written 1, indicates to programmable devices that OS is taking control over it.

The file is read/write.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/pcie_asic_reset_dis

Defined on file *sysfs-driver-mlxreg-io*

This file allows to retain ASIC up during PCIe root complex reset, when attribute is set 1.

The file is read/write.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/phy_reset

Defined on file *sysfs-driver-mlxreg-io*

This file allows to reset PHY 88E1548 when attribute is set 0 due to some abnormal PHY behavior. Expected behavior: When phy_reset is written 1, all PHY 88E1548 are released from the reset state, when 0 - are hold in reset state.

The files are read/write.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/pm_mgmt_en

Defined on file *sysfs-driver-mlxreg-io*

This file assigns power management control ownership. When power management control is provided by hardware, hardware will automatically power off one or more line previously powered line cards in case system power budget is getting insufficient. It could be in case when some of power units lost power good state. When pm_mgmt_en is written 1, power management control by software is enabled, 0 - power management control by hardware. Note that for any setting of pm_mgmt_en attribute hardware will not allow to power on any new line card in case system power budget is insufficient. Same in case software will try to power on several line cards at once - hardware will power line cards while system has enough power budget. Default is 0.

The file is read/write.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/psu1_on

Defined on file *sysfs-driver-mlxreg-io*

These files allow asserting system power cycling, switching power supply units on and off and system's main power domain shutdown. Expected behavior: When pwr_cycle is written 1: auxiliary power domain will go down and after short period (about 1 second) up. When psu1_on or psu2_on is written 1, related unit will be disconnected from the power source, when written 0 - connected. If both are written 1 - power supplies main power domain will go down. When pwr_down is written 1, system's main power domain will go down.

The files are write only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/psu3_on
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/psu4_on

Defined on file *sysfs-driver-mlxreg-io*

These files switching power supply units on and off. Expected behavior: When psu3_on or psu4_on is written 1, related unit will be disconnected from the power source, when written 0 - connected.

The files are write only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/pwr_converter_prog_en

Defined on file *sysfs-driver-mlxreg-io*

This file enables programming of power converters. Default is 0 (programming disabled). If the system is in locked-down mode writing this file will not be allowed.

The file is read/write.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/qsfp_pwr_good

Defined on file *sysfs-driver-mlxreg-io*

This file shows QSFP ports power status. The value is set to 0 when one of any QSFP ports is plugged. The value is set to 1 when there are no any QSFP ports are plugged. The possible values are: 0 - Power good, 1 - Not power good.

The files are read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_ac_ok_fail

Defined on file *sysfs-driver-mlxreg-io*

This file shows the system reset cause due to AC power failure. Value 1 in file means this is reset cause, 0 - otherwise.

The file is read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_ac_pwr_fail
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_platform
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_soc
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_sw_pwr_off

Defined on file *sysfs-driver-mlxreg-io*

These files show the system reset causes, as following: reset due to AC power failure, reset invoked from software by assertion reset signal through CPLD. reset caused by signal asserted by SOC through ACPI register, reset invoked from software by assertion power off signal through CPLD.

The files are read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_aux_pwr_or_ref
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_asic_thermal
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_hotswap_or_halt
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_hotswap_or_wd
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_fw_reset
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_long_pb
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_main_pwr_fail
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_short_pb
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_sw_reset

Defined on file *sysfs-driver-mlxreg-io*

These files show the system reset cause, as following: power auxiliary outage or power refresh, ASIC thermal shutdown, halt, hotswap, watchdog, firmware reset, long press power button, short press power button, software reset. Value 1 in file means this is reset cause, 0 - otherwise. Only one of the above causes could be 1 at the same time, representing only last reset cause.

The files are read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_comex_pwr_fail
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_from_comex
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_system
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_voltmon_upgrade_fail

Defined on file *sysfs-driver-mlxreg-io*

These files show the system reset cause, as following: ComEx power fail, reset from ComEx, system platform reset, reset due to voltage monitor devices upgrade failure, Value 1 in file means this is reset cause, 0 - otherwise. Only one bit could be 1 at the same time, representing only the last reset cause.

The files are read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_comex_thermal
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_comex_wd
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_from_asic
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_reload_bios
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_sff_wd
/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_swb_wd

Defined on file *sysfs-driver-mlxreg-io*

These files show the system reset cause, as following: COMEX thermal shutdown; wathchdog power off or reset was derived by one of the next components: COMEX, switch board or by Small Form Factor mezzanine, reset requested from ASIC, reset caused by BIOS reload. Value

1 in file means this is reset cause, 0 - otherwise. Only one of the above causes could be 1 at the same time, representing only last reset cause.

The files are read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_long_pwr_pb

Defined on file *sysfs-driver-mlxreg-io*

This file if set 1 indicates that system has been reset by long press of power button.

The file is read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_pwr_converter_fail

Defined on file *sysfs-driver-mlxreg-io*

This file shows the system reset cause due to power converter devices failure. Value 1 in file means this is reset cause, 0 - otherwise.

The file is read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_swb_dc_dc_pwr_fail

Defined on file *sysfs-driver-mlxreg-io*

This file shows 1 in case the system reset happened due to the failure of any DC-DC power converter devices equipped on the switch board.

The file is read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/select_iio

Defined on file *sysfs-driver-mlxreg-io*

This file allows iio devices selection.

Attribute select_iio can be written with 0 or with 1. It selects which one of iio devices can be accessed.

The file is read/write.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/shutdown_unlock

Defined on file *sysfs-driver-mlxreg-io*

This file allows to unlock ASIC after thermal shutdown event. When system thermal shutdown is enforced by ASIC, ASIC is getting locked and after system boot it will not be available. Software can decide to unlock it by setting this attribute to 1 and then perform system power cycle by setting pwr_cycle attribute to 1 (power cycle of main power domain). Before setting shutdown_unlock to 1 it is recommended to validate that system reboot cause is reset_asic_thermal or reset_thermal_spc_or_pciesw. In case shutdown_unlock is not set 1, the only way to release ASIC from locking - is full system power cycle through the external power distribution unit. Default is 1.

The file is read/write.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/spi_chnl_select

Defined on file *sysfs-driver-mlxreg-io*

This file allows SPI chip selection for External Root of Trust (EROT) device Out-of-Band recovery. File can be written with 0 or with 1. It selects which EROT can be accessed through SPI device.

The file is read/write.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/ufm_version

Defined on file *sysfs-driver-mlxreg-io*

This file exposes the firmware version of burnable voltage regulator devices.

The file is read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/voltreg_update_status

Defined on file *sysfs-driver-mlxreg-io*

This file exposes the configuration update status of burnable voltage regulator devices. The status values are as following: 0 - OK; 1 - CRC failure; 2 = I2C failure; 3 - in progress.

The file is read only.

/sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/vpd_wp

Defined on file *sysfs-driver-mlxreg-io*

This file allows to overwrite system VPD hardware write protection when attribute is set 1.

The file is read/write.

/sys/devices/system/cpu/cpuX/topology/book_id

Defined on file *sysfs-devices-system-cpu*

the book ID of cpuX. Typically it is the hardware platform's identifier (rather than the kernel's). The actual value is architecture and platform dependent. it's only used on s390. Values: integer

/sys/devices/system/cpu/cpuX/topology/book_siblings

Defined on file *sysfs-devices-system-cpu*

internal kernel map of cpuX's hardware threads within the same book_id. it's only used on s390. Values: hexadecimal bitmask.

/sys/devices/system/cpu/cpuX/topology/book_siblings_list

Defined on file *sysfs-devices-system-cpu*

human-readable list of cpuX's hardware threads within the same book_id. The format is like 0-3, 8-11, 14,17. it's only used on s390. Values: decimal list.

/sys/devices/system/cpu/cpuX/topology/cluster_cpus

Defined on file *sysfs-devices-system-cpu*

internal kernel map of CPUs within the same cluster. Values: hexadecimal bitmask.

/sys/devices/system/cpu/cpuX/topology/cluster_cpus_list

Defined on file *sysfs-devices-system-cpu*

human-readable list of CPUs within the same cluster. The format is like 0-3, 8-11, 14,17. Values: decimal list.

/sys/devices/system/cpu/cpuX/topology/cluster_id

Defined on file *sysfs-devices-system-cpu*

the cluster ID of cpuX. Typically it is the hardware platform's identifier (rather than the kernel's). The actual value is architecture and platform dependent. Values: integer

/sys/devices/system/cpu/cpuX/topology/core_cpus

Defined on file *sysfs-devices-system-cpu*

internal kernel map of CPUs within the same core. (deprecated name: "thread_siblings") Values: hexadecimal bitmask.

/sys/devices/system/cpu/cpuX/topology/core_cpus_list

Defined on file *sysfs-devices-system-cpu*

human-readable list of CPUs within the same core. The format is like 0-3, 8-11, 14,17. (deprecated name: "thread_siblings_list"). Values: decimal list.

/sys/devices/system/cpu/cpuX/topology/core_id

Defined on file *sysfs-devices-system-cpu*

the CPU core ID of cpuX. Typically it is the hardware platform's identifier (rather than the kernel's). The actual value is architecture and platform dependent. Values: integer

/sys/devices/system/cpu/cpuX/topology/die_cpus

Defined on file *sysfs-devices-system-cpu*

internal kernel map of CPUs within the same die. Values: hexadecimal bitmask.

/sys/devices/system/cpu/cpuX/topology/die_cpus_list

Defined on file *sysfs-devices-system-cpu*

human-readable list of CPUs within the same die. The format is like 0-3, 8-11, 14,17. Values: decimal list.

/sys/devices/system/cpu/cpuX/topology/die_id

Defined on file *sysfs-devices-system-cpu*

the CPU die ID of cpuX. Typically it is the hardware platform's identifier (rather than the kernel's). The actual value is architecture and platform dependent. Values: integer

/sys/devices/system/cpu/cpuX/topology/drawer_id

Defined on file *sysfs-devices-system-cpu*

the drawer ID of cpuX. Typically it is the hardware platform's identifier (rather than the kernel's). The actual value is architecture and platform dependent. it's only used on s390. Values: integer

/sys/devices/system/cpu/cpuX/topology/drawer_siblings

Defined on file *sysfs-devices-system-cpu*

internal kernel map of cpuX's hardware threads within the same drawer_id. it's only used on s390. Values: hexadecimal bitmask.

/sys/devices/system/cpu/cpuX/topology/drawer_siblings_list

Defined on file *sysfs-devices-system-cpu*

human-readable list of cpuX's hardware threads within the same drawer_id. The format is like 0-3, 8-11, 14,17. it's only used on s390. Values: decimal list.

/sys/devices/system/cpu/cpuX/topology/package_cpus

Defined on file *sysfs-devices-system-cpu*

internal kernel map of the CPUs sharing the same physical_package_id. (deprecated name: "core_siblings"). Values: hexadecimal bitmask.

/sys/devices/system/cpu/cpuX/topology/package_cpus_list

Defined on file *sysfs-devices-system-cpu*

human-readable list of CPUs sharing the same physical_package_id. The format is like 0-3, 8-11, 14,17. (deprecated name: "core_siblings_list") Values: decimal list.

/sys/devices/system/cpu/cpuX/topology/physical_package_id

Defined on file *sysfs-devices-system-cpu*

physical package id of cpuX. Typically corresponds to a physical socket number, but the actual value is architecture and platform dependent. Values: integer

/sys/devices/system/cpu/cpuX/topology/ppin

Defined on file *sysfs-devices-system-cpu*

per-socket protected processor inventory number Values: hexadecimal.

/sys/devices/system/cpu/cpu[0-9]+/dscr

Defined on file *sysfs-devices-system-cpu*

Default value for the Data Stream Control Register (DSCR) on a CPU. This default value is used when the kernel is executing and for any process that has not set the DSCR itself. If a process ever sets the DSCR (via direct access to the SPR) that value will be persisted for that process and used on any CPU where it executes (overriding the value described here). If set by a process it will be inherited by child processes. Values: 64 bit unsigned integer (bit field)

/sys/devices/system/cpu/dscr_default

Defined on file *sysfs-devices-system-cpu*

Writes are equivalent to writing to /sys/devices/system/cpu/cpuN/dscr on all CPUs. Reads return the last written value or 0. This value is not a global default: it is a way to set all per-CPU defaults at the same time. Values: 64 bit unsigned integer (bit field)

/sys/devices/system/node/has_cpu

Defined on file *sysfs-devices-node*

Nodes that have one or more CPUs.

/sys/devices/system/node/has_high_memory

Defined on file *sysfs-devices-node*

Nodes that have regular or high memory. Depends on CONFIG_HIGHMEM.

/sys/devices/system/node/has_normal_memory

Defined on file *sysfs-devices-node*

Nodes that have regular memory.

/sys/devices/system/node/nodeX

Defined on file *sysfs-devices-node*

When CONFIG_NUMA is enabled, this is a directory containing information on node X such as what CPUs are local to the node. Each file is detailed next.

/sys/devices/system/node/nodeX/accessY/

Defined on file *sysfs-devices-node*

The node's relationship to other nodes for access class "Y".

/sys/devices/system/node/nodeX/accessY/initiators/

Defined on file *sysfs-devices-node*

The directory containing symlinks to memory initiator nodes that have class "Y" access to this target node's memory. CPUs and other memory initiators in nodes not in the list accessing this node's memory may have different performance.

/sys/devices/system/node/nodeX/accessY/initiators/read_bandwidth

Defined on file *sysfs-devices-node*

This node's read bandwidth in MB/s when accessed from nodes found in this access class's linked initiators.

/sys/devices/system/node/nodeX/accessY/initiators/read_latency

Defined on file *sysfs-devices-node*

This node's read latency in nanoseconds when accessed from nodes found in this access class's linked initiators.

/sys/devices/system/node/nodeX/accessY/initiators/write_bandwidth

Defined on file *sysfs-devices-node*

This node's write bandwidth in MB/s when accessed from found in this access class's linked initiators.

/sys/devices/system/node/nodeX/accessY/initiators/write_latency

Defined on file *sysfs-devices-node*

This node's write latency in nanoseconds when access from nodes found in this class's linked initiators.

/sys/devices/system/node/nodeX/accessY/targets/

Defined on file *sysfs-devices-node*

The directory containing symlinks to memory targets that this initiator node has class "Y" access.

/sys/devices/system/node/nodeX/compact

Defined on file *sysfs-devices-node*

When this file is written to, all memory within that node will be compacted. When it completes, memory will be freed into blocks which have as many contiguous pages as possible

/sys/devices/system/node/nodeX/cpulist

Defined on file *sysfs-devices-node*

The CPUs associated to the node.

/sys/devices/system/node/nodeX/cpumap

Defined on file *sysfs-devices-node*

The node's cpumap.

/sys/devices/system/node/nodeX/distance

Defined on file *sysfs-devices-node*

Distance between the node and all the other nodes in the system.

/sys/devices/system/node/nodeX/hugepages/hugepages-<size>/

Defined on file *sysfs-devices-node*

The node's huge page size control/query attributes. See *HugeTLB Pages*

/sys/devices/system/node/nodeX/meminfo

Defined on file *sysfs-devices-node*

Provides information about the node's distribution and memory utilization. Similar to /proc/meminfo, see /filesystems/proc

/sys/devices/system/node/nodeX/memory_failure/delayed

Defined on file *sysfs-devices-node*

Of the raw poisoned pages on a NUMA node, how many pages are delayed by memory error recovery attempt. Delayed poisoned pages usually will be retried by kernel.

/sys/devices/system/node/nodeX/memory_failure/failed

Defined on file *sysfs-devices-node*

Of the raw poisoned pages on a NUMA node, how many pages are failed by memory error recovery attempt. This usually means a key recovery operation failed.

/sys/devices/system/node/nodeX/memory_failure/ignored

Defined on file *sysfs-devices-node*

Of the raw poisoned pages on a NUMA node, how many pages are ignored by memory error recovery attempt, usually because support for this type of pages is unavailable, and kernel gives up the recovery.

/sys/devices/system/node/nodeX/memory_failure/recovered

Defined on file *sysfs-devices-node*

Of the raw poisoned pages on a NUMA node, how many pages are recovered by memory error recovery attempt.

/sys/devices/system/node/nodeX/memory_failure/total

Defined on file *sysfs-devices-node*

The total number of raw poisoned pages (pages containing corrupted data due to memory errors) on a NUMA node.

/sys/devices/system/node/nodeX/memory_side_cache/indexY/

Defined on file *sysfs-devices-node*

The directory containing attributes for the memory-side cache level 'Y'.

/sys/devices/system/node/nodeX/memory_side_cache/indexY/indexing

Defined on file *sysfs-devices-node*

The caches associativity indexing: 0 for direct mapped, non-zero if indexed.

/sys/devices/system/node/nodeX/memory_side_cache/indexY/line_size

Defined on file *sysfs-devices-node*

The number of bytes accessed from the next cache level on a cache miss.

/sys/devices/system/node/nodeX/memory_side_cache/indexY/size

Defined on file *sysfs-devices-node*

The size of this memory side cache in bytes.

/sys/devices/system/node/nodeX/memory_side_cache/indexY/write_policy

Defined on file *sysfs-devices-node*

The cache write policy: 0 for write-back, 1 for write-through, other or unknown.

/sys/devices/system/node/nodeX/numastat

Defined on file *sysfs-devices-node*

The node's hit/miss statistics, in units of pages. See *Numa policy hit/miss statistics*

/sys/devices/system/node/nodeX/vmstat

Defined on file *sysfs-devices-node*

The node's zoned virtual memory statistics. This is a superset of numastat.

/sys/devices/system/node/nodeX/x86/sgx_total_bytes

Defined on file *sysfs-devices-node*

The total amount of SGX physical memory in bytes.

/sys/devices/system/node/online

Defined on file *sysfs-devices-node*

Nodes that are online.

/sys/devices/system/node/possible

Defined on file *sysfs-devices-node*

Nodes that could be possibly become online at some point.

/sys/devices/system/xen_memory/xen_memory0/info/current_kb

Defined on file *sysfs-devices-system-xen_memory*

Current size (in KiB) of this domain's memory reservation.

/sys/devices/system/xen_memory/xen_memory0/info/high_kb

Defined on file *sysfs-devices-system-xen_memory*

Amount (in KiB) of high memory in the balloon.

/sys/devices/system/xen_memory/xen_memory0/info/low_kb

Defined on file *sysfs-devices-system-xen_memory*

Amount (in KiB) of low (or normal) memory in the balloon.

/sys/devices/system/xen_memory/xen_memory0/max_retry_count

Defined on file *sysfs-devices-system-xen_memory*

The maximum number of times the balloon driver will attempt to increase the balloon before giving up. See also 'retry_count' below. A value of zero means retry forever and is the default one.

/sys/devices/system/xen_memory/xen_memory0/max_schedule_delay

Defined on file *sysfs-devices-system-xen_memory*

The limit that 'schedule_delay' (see below) will be increased to. The default value is 32 seconds.

/sys/devices/system/xen_memory/xen_memory0/retry_count

Defined on file *sysfs-devices-system-xen_memory*

The current number of times that the balloon driver has attempted to increase the size of the balloon. The default value is one. With max_retry_count being zero (unlimited), this means that the driver will attempt to retry with a 'schedule_delay' delay.

/sys/devices/system/xen_memory/xen_memory0/schedule_delay

Defined on file *sysfs-devices-system-xen_memory*

The time (in seconds) to wait between attempts to increase the balloon. Each time the balloon cannot be increased, 'schedule_delay' is increased (until 'max_schedule_delay' is reached at which point it will use the max value).

/sys/devices/system/xen_memory/xen_memory0/scrub_pages

Defined on file *sysfs-devices-system-xen_memory*

Control scrubbing pages before returning them to Xen for others domains use. Can be set with xen_scrub_pages cmdline parameter. Default value controlled with CONFIG_XEN_SCRUB_PAGES_DEFAULT.

/sys/devices/system/xen_memory/xen_memory0/target

Defined on file *sysfs-devices-system-xen_memory*

The target number of pages to adjust this domain's memory reservation to.

/sys/devices/system/xen_memory/xen_memory0/target_kb

Defined on file *sysfs-devices-system-xen_memory*

As target above, except the value is in KiB.

5.1.7 Symbols under /sys/firmware

/sys/firmware/acpi/pm_profile

Defined on file *sysfs-acpi-pmprofile*

The ACPI pm_profile sysfs interface exposes the preferred power management (and performance) profile of the platform as provided in the ACPI FADT Preferred_PM_Profile field.

The integer value is directly passed as retrieved from the FADT.

Values: For the possible values refer to the Preferred_PM_Profile field
definition in Table 5.9 ”FADT Format”, Section 5.2.9 ”Fixed ACPI Description Table (FADT)” of the ACPI specification.

As of ACPI 6.5, the following values are defined:

0	Unspecified
1	Desktop
2	Mobile
3	Workstation
4	Enterprise Server
5	SOHO Server
6	Appliance PC
7	Performance Server
8	Tablet
>8	Reserved

/sys/firmware/efi/vars

Defined on file *sysfs-firmware-efi-vars*

This directory exposes interfaces for interactive with EFI variables. For more information on EFI variables, see 'Variable Services' in the UEFI specification (section 7.2 in specification version 2.3 Errata D).

In summary, EFI variables are named, and are classified into separate namespaces through the use of a vendor GUID. They also have an arbitrary binary value associated with them.

The efivars module enumerates these variables and creates a separate directory for each one found. Each directory has a name of the form ”<key>-<vendor guid>” and contains the following files:

attributes:	A read-only text file enumerating the EFI variable flags. Potential values include: EFI_VARIABLE_NON_VOLATILE EFI_VARIABLE_BOOTSERVICE_ACCESS EFI_VARIABLE_RUNTIME_ACCESS EFI_VARIABLE_HARDWARE_ERROR_RECORD EFI_VARIABLE_AUTHENTICATED_WRITE_ACCESS See the EFI documentation for an explanation of each of these variables.
data:	A read-only binary file that can be read to attain the value of the EFI variable
guid:	The vendor GUID of the variable. This should always match the GUID in the variable's name.
raw_var:	A binary file that can be read to obtain a structure that contains everything there is to know about the variable. For structure definition see ”struct efi_variable” in the kernel sources. This file can also be written to in order to update the value of a variable. For this to work however, all fields of the ”struct efi_variable” passed must match byte for byte with the structure read out of the file, save for the value portion. **Note** the efi_variable structure read/written with this file contains a 'long' type that may change widths depending on your underlying architecture.
size:	As ASCII representation of the size of the variable's value.

In addition, two other magic binary files are provided in the top-level directory and are used for adding and removing variables:

new_var:	Takes a ”struct efi_variable” and instructs the EFI firmware to create a new variable.
del_var:	Takes a ”struct efi_variable” and instructs the EFI firmware to remove any variable that has a matching vendor GUID and variable key name.

/sys/firmware/opal/dump

Defined on file *sysfs-firmware-opal-dump*

This directory exposes interfaces for interacting with the FSP and platform dumps through OPAL firmware interface.

This is only for the powerpc/powernv platform.

initiate_dump:	When '1' is written to it, we will initiate a dump. Read this file for supported commands.
0xXX-0xYYYY:	A directory for dump of type 0xXX and id 0xYYYY (in hex). The name of this directory should not be relied upon to be in this format, only that it's unique among all dumps. For determining the type and ID of the dump, use the id and type files. Do not rely on any particular size of dump type or dump id.

Each dump has the following files:

id:	An ASCII representation of the dump ID in hex (e.g. '0x01')
type:	An ASCII representation of the type of dump in the format "0x%x %s" with the ID in hex and a description of the dump type (or 'unknown'). Type '0xffffffff unknown' is used when we could not get the type from firmware. e.g. '0x02 System/Platform Dump'
dump:	A binary file containing the dump. The size of the dump is the size of this file.
acknowledge:	When 'ack' is written to this, we will acknowledge that we've retrieved the dump to the service processor. It will then remove it, making the dump inaccessible. Reading this file will get a list of supported actions.

/sys/firmware/opal/elog

Defined on file *sysfs-firmware-opal-elog*

This directory exposes error log entries retrieved through the OPAL firmware interface.

Each error log is identified by a unique ID and will exist until explicitly acknowledged to firmware.

Each log entry has a directory in */sys/firmware/opal/elog*.

Log entries may be purged by the service processor before retrieved by firmware or retrieved/acknowledged by Linux if there is no room for more log entries.

In the event that Linux has retrieved the log entries but not explicitly acknowledged them to firmware and the service processor needs more room for log entries, the only remaining copy of a log message may be in Linux.

Typically, a user space daemon will monitor for new entries, read them out and acknowledge them.

The service processor may be able to store more log entries than firmware can, so after you acknowledge an event from Linux you may instantly get another one from the queue that was generated some time in the past.

The raw log format is a binary format. We currently do not parse this at all in kernel, leaving it up to user space to solve the problem. In future, we may do more parsing in kernel and add more files to make it easier for simple user space processes to extract more information.

For each log entry (directory), there are the following files:

id:	An ASCII representation of the ID of the error log, in hex - e.g. "0x01".
type:	An ASCII representation of the type id and description of the type of error log. Currently just "0x00 PEL" - platform error log. In the future there may be additional types.
raw:	A read-only binary file that can be read to get the raw log entry. These are <16kb, often just hundreds of bytes and "average" 2kb.
acknowledge:	Writing 'ack' to this file will acknowledge the error log to firmware (and in turn the service processor, if applicable). Shortly after acknowledging it, the log entry will be removed from sysfs. Reading this file will list the supported operations (currently just acknowledge).

5.1.8 Symbols under /sys/fs

/sys/fs/o2cb/

Defined on file *o2cb*

Ocfs2-tools looks at 'interface-revision' for versioning information. Each logmask/ file controls a set of debug prints and can be written into with the strings "allow", "deny", or "off". Reading the file returns the current state.

Users: ocfs2-tools. It's sufficient to mail proposed changes to ocfs2-devel@lists.linux.dev.

/sys/fs/orangefs/acache/*

Defined on file *sysfs-fs-orangefs*

Attribute cache configurable settings.

/sys/fs/orangefs/capcache/*

Defined on file *sysfs-fs-orangefs*

Capability cache configurable settings.

/sys/fs/orangefs/ccache/*

Defined on file *sysfs-fs-orangefs*

Credential cache configurable settings.

/sys/fs/orangefs/ncache/*

Defined on file *sysfs-fs-orangefs*

Name cache configurable settings.

/sys/fs/orangefs/op_timeout_secs

Defined on file *sysfs-fs-orangefs*

Service operation timeout in seconds.

/sys/fs/orangefs/perf_counter_reset

Defined on file *sysfs-fs-orangefs*

echo a 0 or a 1 into perf_counter_reset to reset all the counters in /sys/fs/orangefs/perf_counters except ones with PINT_PERF_PRESERVE set.

/sys/fs/orangefs/perf_counters/*

Defined on file *sysfs-fs-orangefs*

Counters and settings for various caches. Read only.

/sys/fs/orangefs/perf_history_size

Defined on file *sysfs-fs-orangefs*

The perf_counters cache statistics have N, or perf_history_size, samples. The default is one.

Every perf_time_interval_secs the (first) samples are reset.

If N is greater than one, the "current" set of samples is reset, and the samples from the other N-1 intervals remain available.

/sys/fs/orangefs/perf_time_interval_secs

Defined on file *sysfs-fs-orangefs*

Length of perf counter intervals in seconds.

/sys/fs/orangefs/slot_timeout_secs

Defined on file *sysfs-fs-orangefs*

"Slot" timeout in seconds. A "slot" is an indexed buffer in the shared memory segment used for communication between the kernel module and userspace. Slots are requested and waited for, the wait times out after slot_timeout_secs.

5.1.9 Symbols under /sys/hypervisor

/sys/hypervisor/compilation/compile_date

Defined on file *sysfs-hypervisor-xen*

If running under Xen: Contains the build time stamp of the Xen hypervisor Might return "<denied>" in case of special security settings in the hypervisor.

/sys/hypervisor/compilation/compiled_by

Defined on file *sysfs-hypervisor-xen*

If running under Xen: Contains information who built the Xen hypervisor Might return "<denied>" in case of special security settings in the hypervisor.

/sys/hypervisor/compilation/compiler

Defined on file *sysfs-hypervisor-xen*

If running under Xen: Compiler which was used to build the Xen hypervisor Might return "<denied>" in case of special security settings in the hypervisor.

/sys/hypervisor/properties/capabilities

Defined on file *sysfs-hypervisor-xen*

If running under Xen: Space separated list of supported guest system types. Each type is in the format: <class>-<major>.<minor>-<arch> With:

<class>:	"xen" -- x86: paravirtualized, arm: standard "hvm" -- x86 only: fully virtualized
<major>:	major guest interface version
<minor>:	minor guest interface version
<arch>:	architecture, e.g.: "x86_32": 32 bit x86 guest without PAE "x86_32p": 32 bit x86 guest with PAE "x86_64": 64 bit x86 guest "armv7l": 32 bit arm guest "aarch64": 64 bit arm guest

/sys/hypervisor/properties/changeset

Defined on file *sysfs-hypervisor-xen*

If running under Xen: Changeset of the hypervisor (git commit) Might return "<denied>" in case of special security settings in the hypervisor.

/sys/hypervisor/properties/features

Defined on file *sysfs-hypervisor-xen*

If running under Xen: Features the Xen hypervisor supports for the guest as defined in include/xen/interface/features.h printed as a hex value.

/sys/hypervisor/properties/pagesize

Defined on file *sysfs-hypervisor-xen*

If running under Xen: Default page size of the hypervisor printed as a hex value. Might return "0" in case of special security settings in the hypervisor.

/sys/hypervisor/properties/virtual_start

Defined on file *sysfs-hypervisor-xen*

If running under Xen: Virtual address of the hypervisor as a hex value.

/sys/hypervisor/start_flags/*

Defined on file *sysfs-hypervisor-xen*

If running under Xen: All bits in Xen's start-flags are represented as boolean files, returning '1' if set, '0' otherwise. This takes the place of the defunct /proc/xen/capabilities, which would

contain ”control_d” on dom0, and be empty otherwise. This flag is now exposed as ”initdomain” in addition to the ”privileged” flag; all other possible flags are accessible as ”unknownXX”.

/sys/hypervisor/type

Defined on file *sysfs-hypervisor-xen*

If running under Xen: Type of hypervisor: ”xen”: Xen hypervisor

/sys/hypervisor/uuid

Defined on file *sysfs-hypervisor-xen*

If running under Xen: UUID of the guest as known to the Xen hypervisor.

/sys/hypervisor/version/extra

Defined on file *sysfs-hypervisor-xen*

If running under Xen: The Xen version is in the format <major>.<minor><extra> This is the <extra> part of it. Might return ”<denied>” in case of special security settings in the hypervisor.

/sys/hypervisor/version/major

Defined on file *sysfs-hypervisor-xen*

If running under Xen: The Xen version is in the format <major>.<minor><extra> This is the <major> part of it.

/sys/hypervisor/version/minor

Defined on file *sysfs-hypervisor-xen*

If running under Xen: The Xen version is in the format <major>.<minor><extra> This is the <minor> part of it.

5.1.10 Symbols under /sys/kernel

/sys/kernel/notes

Defined on file *sysfs-kernel-notes*

The */sys/kernel/notes* file contains the binary representation of the running vmlinux's .notes section.

5.1.11 Symbols under /sys/module

/sys/module/<MODULENAME>

Defined on file *sysfs-module*

The name of the module that is in the kernel. This module name will always show up if the module is loaded as a dynamic module. If it is built directly into the kernel, it will only show up if it has a version or at least one parameter.

Note: The conditions of creation in the built-in case are not by design and may be removed in the future.

/sys/module/<MODULENAME>/parameters

Defined on file *sysfs-module*

This directory contains individual files that are each individual parameters of the module that are able to be changed at runtime. See the individual module documentation as to the contents of these parameters and what they accomplish.

Note: The individual parameter names and values are not considered stable, only the fact that they will be placed in this location within sysfs. See the individual driver documentation for details as to the stability of the different parameters.

/sys/module/<MODULENAME>/refcnt

Defined on file *sysfs-module*

If the module is able to be unloaded from the kernel, this file will contain the current reference count of the module.

Note: If the module is built into the kernel, or if the CONFIG_MODULE_UNLOAD kernel configuration value is not enabled, this file will not be present.

/sys/module/<MODULENAME>/srcversion

Defined on file *sysfs-module*

If the module source has MODULE_VERSION, this file will contain the checksum of the source code.

/sys/module/<MODULENAME>/version

Defined on file *sysfs-module*

If the module source has MODULE_VERSION, this file will contain the version of the source code.

5.1.12 A notification mechanism for thermal related events

A notification mechanism for thermal related events

Defined on file *thermal-notification*

This interface enables notification for thermal related events. The notification is in the form of a netlink event.

5.1.13 Audit Login Session ID

Audit Login Session ID

Defined on file *procfs-audit_loginuid*

The /proc/$pid/sessionid pseudofile is read to get the audit login session ID of process $pid as a decimal unsigned int (%u, u32). It is set automatically, serially assigned with each new login.

Users: audit and login applications

5.1.14 Audit Login UID

Audit Login UID

Defined on file *procfs-audit_loginuid*

The /proc/$pid/loginuid pseudofile is written to set and read to get the audit login UID of process $pid as a decimal unsigned int (%u, u32). If it is unset, permissions are not needed to set it. The accessor must have CAP_AUDIT_CONTROL in the initial user namespace to write it if it has been set. It cannot be written again if AUDIT_FEATURE_LOGINUID_IMMUTABLE is enabled. It cannot be unset if AUDIT_FEATURE_ONLY_UNSET_LOGINUID is enabled.

Users: audit and login applications

5.1.15 The kernel syscall interface

The kernel syscall interface

Defined on file *syscalls*

This interface matches much of the POSIX interface and is based on it and other Unix based interfaces. It will only be added to over time, and not have things removed from it.

Note that this interface is different for every architecture that Linux supports. Please see the architecture-specific documentation for details on the syscall numbers that are to be mapped to each syscall.

5.1.16 vDSO

vDSO

Defined on file *vdso*

On some architectures, when the kernel loads any userspace program it maps an ELF DSO into that program's address space. This DSO is called the vDSO and it often contains useful and highly-optimized alternatives to real syscalls.

These functions are called just like ordinary C function according to your platform's ABI. Call them from a sensible context. (For example, if you set CS on x86 to something strange, the vDSO functions are within their rights to crash.) In addition, if you pass a bad pointer to a vDSO function, you might get SIGSEGV instead of -EFAULT.

To find the DSO, parse the auxiliary vector passed to the program's entry point. The AT_SYSINFO_EHDR entry will point to the vDSO.

The vDSO uses symbol versioning; whenever you request a symbol from the vDSO, specify the version you are expecting.

Programs that dynamically link to glibc will use the vDSO automatically. Otherwise, you can use the reference parser in tools/testing/selftests/vDSO/parse_vdso.c.

Unless otherwise noted, the set of symbols with any given version and the ABI of those symbols is considered stable. It may vary across architectures, though.

Note:
As of this writing, this ABI documentation as been confirmed for x86_64. The maintainers of the other vDSO-using architectures should confirm that it is correct for their architecture.

5.1.17 File stable/firewire-cdev

Has the following ABI:

- */dev/fw[0-9]+*

5.1.18 File stable/o2cb

Has the following ABI:

- */sys/fs/o2cb/*

5.1.19 File stable/procfs-audit_loginuid

Has the following ABI:

- *Audit Login UID*
- *Audit Login Session ID*

5.1.20 File stable/syscalls

Has the following ABI:

- *The kernel syscall interface*

5.1.21 File stable/sysfs-acpi-pmprofile

Has the following ABI:

- */sys/firmware/acpi/pm_profile*

5.1.22 File stable/sysfs-block

Has the following ABI:

- */sys/block/<disk>/alignment_offset*
- */sys/block/<disk>/discard_alignment*
- */sys/block/<disk>/diskseq*
- */sys/block/<disk>/inflight*
- */sys/block/<disk>/integrity/device_is_integrity_capable*
- */sys/block/<disk>/integrity/format*
- */sys/block/<disk>/integrity/protection_interval_bytes*
- */sys/block/<disk>/integrity/read_verify*
- */sys/block/<disk>/integrity/tag_size*
- */sys/block/<disk>/integrity/write_generate*
- */sys/block/<disk>/<partition>/alignment_offset*
- */sys/block/<disk>/<partition>/discard_alignment*
- */sys/block/<disk>/<partition>/stat*
- */sys/block/<disk>/queue/add_random*
- */sys/block/<disk>/queue/chunk_sectors*
- */sys/block/<disk>/queue/crypto/*
- */sys/block/<disk>/queue/crypto/max_dun_bits*
- */sys/block/<disk>/queue/crypto/modes/<mode>*
- */sys/block/<disk>/queue/crypto/num_keyslots*

- */sys/block/<disk>/queue/dax*
- */sys/block/<disk>/queue/discard_granularity*
- */sys/block/<disk>/queue/discard_max_bytes*
- */sys/block/<disk>/queue/discard_max_hw_bytes*
- */sys/block/<disk>/queue/discard_zeroes_data*
- */sys/block/<disk>/queue/dma_alignment*
- */sys/block/<disk>/queue/fua*
- */sys/block/<disk>/queue/hw_sector_size*
- */sys/block/<disk>/queue/independent_access_ranges/*
- */sys/block/<disk>/queue/io_poll*
- */sys/block/<disk>/queue/io_poll_delay*
- */sys/block/<disk>/queue/io_timeout*
- */sys/block/<disk>/queue/iostats*
- */sys/block/<disk>/queue/logical_block_size*
- */sys/block/<disk>/queue/max_active_zones*
- */sys/block/<disk>/queue/max_discard_segments*
- */sys/block/<disk>/queue/max_hw_sectors_kb*
- */sys/block/<disk>/queue/max_integrity_segments*
- */sys/block/<disk>/queue/max_open_zones*
- */sys/block/<disk>/queue/max_sectors_kb*
- */sys/block/<disk>/queue/max_segment_size*
- */sys/block/<disk>/queue/max_segments*
- */sys/block/<disk>/queue/minimum_io_size*
- */sys/block/<disk>/queue/nomerges*
- */sys/block/<disk>/queue/nr_requests*
- */sys/block/<disk>/queue/nr_zones*
- */sys/block/<disk>/queue/optimal_io_size*
- */sys/block/<disk>/queue/physical_block_size*
- */sys/block/<disk>/queue/read_ahead_kb*
- */sys/block/<disk>/queue/rotational*
- */sys/block/<disk>/queue/rq_affinity*
- */sys/block/<disk>/queue/scheduler*
- */sys/block/<disk>/queue/stable_writes*
- */sys/block/<disk>/queue/throttle_sample_time*

- */sys/block/<disk>/queue/virt_boundary_mask*
- */sys/block/<disk>/queue/wbt_lat_usec*
- */sys/block/<disk>/queue/write_cache*
- */sys/block/<disk>/queue/write_same_max_bytes*
- */sys/block/<disk>/queue/write_zeroes_max_bytes*
- */sys/block/<disk>/queue/zone_append_max_bytes*
- */sys/block/<disk>/queue/zone_write_granularity*
- */sys/block/<disk>/queue/zoned*
- */sys/block/<disk>/hidden*
- */sys/block/<disk>/stat*

5.1.23 File stable/sysfs-bus-firewire

Has the following ABI:

- */sys/bus/firewire/devices/fw[0-9]+/*
- */sys/bus/firewire/devices/fw[0-9]+/units*
- */sys/bus/firewire/devices/fw[0-9]+/is_local*
- */sys/bus/firewire/devices/fw[0-9]+[.][0-9]+/*
- */sys/bus/firewire/devices/*/*
- */sys/bus/firewire/drivers/sbp2/fw*/host*/target*/*:*:*:*/ieee1394_id*

5.1.24 File stable/sysfs-bus-fsl-mc

Has the following ABI:

- */sys/bus/fsl-mc/rescan*
- */sys/bus/fsl-mc/autorescan*

5.1.25 File stable/sysfs-bus-mhi

Has the following ABI:

- */sys/bus/mhi/devices/.../serialnumber*
- */sys/bus/mhi/devices/.../oem_pk_hash*
- */sys/bus/mhi/devices/.../soc_reset*

5.1.26 File stable/sysfs-bus-nvmem

Has the following ABI:

- */sys/bus/nvmem/devices/.../nvmem*

5.1.27 File stable/sysfs-bus-usb

Has the following ABI:

- */sys/bus/usb/devices/.../power/persist*
- */sys/bus/usb/devices/.../power/autosuspend*
- */sys/bus/usb/device/.../power/connected_duration*
- */sys/bus/usb/device/.../power/active_duration*
- */sys/bus/usb/devices/<busnum>-<port[.port]>...:<config num>-<interface num>/supports_autosuspend*
- */sys/bus/usb/device/.../avoid_reset_quirk*
- */sys/bus/usb/devices/.../devnum*
- */sys/bus/usb/devices/.../bConfigurationValue*
- */sys/bus/usb/devices/.../busnum*
- */sys/bus/usb/devices/.../descriptors*
- */sys/bus/usb/devices/.../speed*

5.1.28 File stable/sysfs-bus-vmbus

Has the following ABI:

- */sys/bus/vmbus/hibernation*
- */sys/bus/vmbus/devices/<UUID>/id*
- */sys/bus/vmbus/devices/<UUID>/class_id*
- */sys/bus/vmbus/devices/<UUID>/device_id*
- */sys/bus/vmbus/devices/<UUID>/channel_vp_mapping*
- */sys/bus/vmbus/devices/<UUID>/device*
- */sys/bus/vmbus/devices/<UUID>/vendor*
- */sys/bus/vmbus/devices/<UUID>/numa_node*
- */sys/bus/vmbus/devices/<UUID>/channels/<N>*
- */sys/bus/vmbus/devices/<UUID>/channels/<N>/cpu*
- */sys/bus/vmbus/devices/<UUID>/channels/<N>/in_mask*
- */sys/bus/vmbus/devices/<UUID>/channels/<N>/latency*
- */sys/bus/vmbus/devices/<UUID>/channels/<N>/out_mask*

- */sys/bus/vmbus/devices/<UUID>/channels/<N>/pending*
- */sys/bus/vmbus/devices/<UUID>/channels/<N>/read_avail*
- */sys/bus/vmbus/devices/<UUID>/channels/<N>/write_avail*
- */sys/bus/vmbus/devices/<UUID>/channels/<N>/events*
- */sys/bus/vmbus/devices/<UUID>/channels/<N>/interrupts*
- */sys/bus/vmbus/devices/<UUID>/channels/<N>/subchannel_id*
- */sys/bus/vmbus/devices/<UUID>/channels/<N>/monitor_id*
- */sys/bus/vmbus/devices/<UUID>/channels/<N>/ring*
- */sys/bus/vmbus/devices/<UUID>/channels/<N>/intr_in_full*
- */sys/bus/vmbus/devices/<UUID>/channels/<N>/intr_out_empty*
- */sys/bus/vmbus/devices/<UUID>/channels/<N>/out_full_first*
- */sys/bus/vmbus/devices/<UUID>/channels/<N>/out_full_total*

5.1.29 File stable/sysfs-bus-w1

Has the following ABI:

- */sys/bus/w1/devices/.../w1_master_timeout_us*

5.1.30 File stable/sysfs-bus-xen-backend

Has the following ABI:

- */sys/bus/xen-backend/devices/*/devtype*
- */sys/bus/xen-backend/devices/*/nodename*
- */sys/bus/xen-backend/devices/vbd-*/physical_device*
- */sys/bus/xen-backend/devices/vbd-*/mode*
- */sys/bus/xen-backend/devices/vbd-*/statistics/f_req*
- */sys/bus/xen-backend/devices/vbd-*/statistics/oo_req*
- */sys/bus/xen-backend/devices/vbd-*/statistics/rd_req*
- */sys/bus/xen-backend/devices/vbd-*/statistics/rd_sect*
- */sys/bus/xen-backend/devices/vbd-*/statistics/wr_req*
- */sys/bus/xen-backend/devices/vbd-*/statistics/wr_sect*
- */sys/bus/xen-backend/devices/*/state*

5.1.31 File stable/sysfs-class-backlight

Has the following ABI:

- */sys/class/backlight/<backlight>/bl_power*
- */sys/class/backlight/<backlight>/brightness*
- */sys/class/backlight/<backlight>/actual_brightness*
- */sys/class/backlight/<backlight>/max_brightness*
- */sys/class/backlight/<backlight>/type*

5.1.32 File stable/sysfs-class-infiniband

sysfs interface common for all infiniband devices

Has the following ABI:

- */sys/class/infiniband/<device>/node_type*
- */sys/class/infiniband/<device>/node_guid*
- */sys/class/infiniband/<device>/sys_image_guid*
- */sys/class/infiniband/<device>/node_desc*
- */sys/class/infiniband/<device>/fw_ver*
- */sys/class/infiniband/<device>/ports/<port-num>/lid*
- */sys/class/infiniband/<device>/ports/<port-num>/rate*
- */sys/class/infiniband/<device>/ports/<port-num>/lid_mask_count*
- */sys/class/infiniband/<device>/ports/<port-num>/sm_sl*
- */sys/class/infiniband/<device>/ports/<port-num>/sm_lid*
- */sys/class/infiniband/<device>/ports/<port-num>/state*
- */sys/class/infiniband/<device>/ports/<port-num>/phys_state*
- */sys/class/infiniband/<device>/ports/<port-num>/cap_mask*
- */sys/class/infiniband/<device>/ports/<port-num>/link_layer*
- */sys/class/infiniband/<device>/ports/<port-num>/counters/symbol_error*
- */sys/class/infiniband/<device>/ports/<port-num>/counters/port_rcv_errors*
- */sys/class/infiniband/<device>/ports/<port-num>/counters/port_rcv_remote_physical_errors*
- */sys/class/infiniband/<device>/ports/<port-num>/counters/port_rcv_switch_relay_errors*
- */sys/class/infiniband/<device>/ports/<port-num>/counters/link_error_recovery*
- */sys/class/infiniband/<device>/ports/<port-num>/counters/port_xmit_constraint_errors*
- */sys/class/infiniband/<device>/ports/<port-num>/counters/port_rcv_contraint_errors*
- */sys/class/infiniband/<device>/ports/<port-num>/counters/local_link_integrity_errors*
- */sys/class/infiniband/<device>/ports/<port-num>/counters/excessive_buffer_overrun_errors*

- /sys/class/infiniband/<device>/ports/<port-num>/counters/port_xmit_data
- /sys/class/infiniband/<device>/ports/<port-num>/counters/port_rcv_data
- /sys/class/infiniband/<device>/ports/<port-num>/counters/port_xmit_packets
- /sys/class/infiniband/<device>/ports/<port-num>/counters/port_rcv_packets
- /sys/class/infiniband/<device>/ports/<port-num>/counters/unicast_rcv_packets
- /sys/class/infiniband/<device>/ports/<port-num>/counters/unicast_xmit_packets
- /sys/class/infiniband/<device>/ports/<port-num>/counters/multicast_rcv_packets
- /sys/class/infiniband/<device>/ports/<port-num>/counters/multicast_xmit_packets
- /sys/class/infiniband/<device>/ports/<port-num>/counters/link_downed
- /sys/class/infiniband/<device>/ports/<port-num>/counters/port_xmit_discards
- /sys/class/infiniband/<device>/ports/<port-num>/counters/VL15_dropped
- /sys/class/infiniband/<device>/ports/<port-num>/counters/port_xmit_wait
- /sys/class/infiniband/<device-name>/hw_counters/lifespan
- /sys/class/infiniband/<device-name>/ports/<port-num>/hw_counters/lifespan
- /sys/class/infiniband/<hca>/ports/<port-number>/gid_attrs/ndevs/<gid-index>
- /sys/class/infiniband/<hca>/ports/<port-number>/gid_attrs/types/<gid-index>
- /sys/class/infiniband_mad/umad<N>/ibdev
- /sys/class/infiniband_mad/umad<N>/port
- /sys/class/infiniband_mad/issm<N>/ibdev
- /sys/class/infiniband_mad/issm<N>/port
- /sys/class/infiniband_mad/abi_version
- /sys/class/infiniband_verbs/uverbs<N>/ibdev
- /sys/class/infiniband_verbs/uverbs<N>/abi_version
- /sys/class/infiniband_verbs/abi_version
- /sys/class/infiniband/mthcaX/hw_rev
- /sys/class/infiniband/mthcaX/hca_type
- /sys/class/infiniband/mthcaX/board_id
- /sys/class/infiniband/mlx4_X/hw_rev
- /sys/class/infiniband/mlx4_X/hca_type
- /sys/class/infiniband/mlx4_X/board_id
- /sys/class/infiniband/mlx4_X/iov/ports/<port-num>/gids/<n>
- /sys/class/infiniband/mlx4_X/iov/ports/<port-num>/admin_guids/<n>
- /sys/class/infiniband/mlx4_X/iov/ports/<port-num>/pkeys/<n>
- /sys/class/infiniband/mlx4_X/iov/ports/<port-num>/mcgs/

- /sys/class/infiniband/mlx4_X/iov/ports/<pci-slot-num>/ports/<m>/gid_idx/0
- /sys/class/infiniband/mlx4_X/iov/ports/<pci-slot-num>/ports/<m>/pkey_idx/<n>
- /sys/class/infiniband/mlx4_X/iov/<pci-slot-num>/ports/<m>/smi_enabled
- /sys/class/infiniband/mlx4_X/iov/<pci-slot-num>/ports/<m>/enable_smi_admin
- /sys/class/infiniband/cxgb4_X/hw_rev
- /sys/class/infiniband/cxgb4_X/hca_type
- /sys/class/infiniband/cxgb4_X/board_id
- /sys/class/infiniband/qibX/version
- /sys/class/infiniband/qibX/hw_rev
- /sys/class/infiniband/qibX/hca_type
- /sys/class/infiniband/qibX/board_id
- /sys/class/infiniband/qibX/boardversion
- /sys/class/infiniband/qibX/nctxts
- /sys/class/infiniband/qibX/localbus_info
- /sys/class/infiniband/qibX/tempsense
- /sys/class/infiniband/qibX/serial
- /sys/class/infiniband/qibX/nfreectxts
- /sys/class/infiniband/qibX/chip_reset
- /sys/class/infiniband/qibX/ports/<N>/sl2vl/[0-15]
- /sys/class/infiniband/qibX/ports/<N>/CCMgtA/cc_settings_bin
- /sys/class/infiniband/qibX/ports/<N>/CCMgtA/cc_table_bin
- /sys/class/infiniband/qibX/ports/<N>/linkstate/loopback
- /sys/class/infiniband/qibX/ports/<N>/linkstate/led_override
- /sys/class/infiniband/qibX/ports/<N>/linkstate/hrtbt_enable
- /sys/class/infiniband/qibX/ports/<N>/linkstate/status
- /sys/class/infiniband/qibX/ports/<N>/linkstate/status_str
- /sys/class/infiniband/qibX/ports/<N>/diag_counters/rc_resends
- /sys/class/infiniband/qibX/ports/<N>/diag_counters/seq_naks
- /sys/class/infiniband/qibX/ports/<N>/diag_counters/rdma_seq
- /sys/class/infiniband/qibX/ports/<N>/diag_counters/rnr_naks
- /sys/class/infiniband/qibX/ports/<N>/diag_counters/other_naks
- /sys/class/infiniband/qibX/ports/<N>/diag_counters/rc_timeouts
- /sys/class/infiniband/qibX/ports/<N>/diag_counters/look_pkts
- /sys/class/infiniband/qibX/ports/<N>/diag_counters/pkt_drops

- */sys/class/infiniband/qibX/ports/<N>/diag_counters/dma_wait*
- */sys/class/infiniband/qibX/ports/<N>/diag_counters/unaligned*
- */sys/class/infiniband/mlx5_X/hw_rev*
- */sys/class/infiniband/mlx5_X/hca_type*
- */sys/class/infiniband/mlx5_X/reg_pages*
- */sys/class/infiniband/mlx5_X/fw_pages*
- */sys/class/infiniband/usnic_X/board_id*
- */sys/class/infiniband/usnic_X/config*
- */sys/class/infiniband/usnic_X/qp_per_vf*
- */sys/class/infiniband/usnic_X/max_vf*
- */sys/class/infiniband/usnic_X/cq_per_vf*
- */sys/class/infiniband/usnic_X/iface*
- */sys/class/infiniband/usnic_X/qpn/summary*
- */sys/class/infiniband/usnic_X/qpn/context*
- */sys/class/infiniband/ocrdmaX/hw_rev*
- */sys/class/infiniband/ocrdmaX/hca_type*
- */sys/class/infiniband/hfi1_X/hw_rev*
- */sys/class/infiniband/hfi1_X/board_id*
- */sys/class/infiniband/hfi1_X/nctxts*
- */sys/class/infiniband/hfi1_X/serial*
- */sys/class/infiniband/hfi1_X/chip_reset*
- */sys/class/infiniband/hfi1_X/boardversion*
- */sys/class/infiniband/hfi1_X/nfreectxts*
- */sys/class/infiniband/hfi1_X/tempsense*
- */sys/class/infiniband/hfi1_X/ports/<N>/CCMgtA/cc_settings_bin*
- */sys/class/infiniband/hfi1_X/ports/<N>/CCMgtA/cc_table_bin*
- */sys/class/infiniband/hfi1_X/ports/<N>/CCMgtA/cc_prescan*
- */sys/class/infiniband/hfi1_X/ports/<N>/sc2vl/[0-31]*
- */sys/class/infiniband/hfi1_X/ports/<N>/sl2sc/[0-31]*
- */sys/class/infiniband/hfi1_X/ports/<N>/vl2mtu/[0-15]*
- */sys/class/infiniband/hfi1_X/sdma_<N>/cpu_list*
- */sys/class/infiniband/hfi1_X/sdma_<N>/vl*
- */sys/class/infiniband/qedrX/hw_rev*
- */sys/class/infiniband/qedrX/hca_type*

- */sys/class/infiniband/vmw_pvrdmaX/hw_rev*
- */sys/class/infiniband/vmw_pvrdmaX/hca_type*
- */sys/class/infiniband/vmw_pvrdmaX/board_id*
- */sys/class/infiniband/bnxt_reX/hw_rev*
- */sys/class/infiniband/bnxt_reX/hca_type*

5.1.33 File stable/sysfs-class-rfkill

rfkill - radio frequency (RF) connector kill switch support

For details to this subsystem look at /driver-api/rfkill.

For the deprecated `/sys/class/rfkill/*/claim` knobs of this interface look in *removed/sysfs-class-rfkill*.

Has the following ABI:

- */sys/class/rfkill*
- */sys/class/rfkill/rfkill[0-9]+/name*
- */sys/class/rfkill/rfkill[0-9]+/type*
- */sys/class/rfkill/rfkill[0-9]+/persistent*
- */sys/class/rfkill/rfkill[0-9]+/state*
- */sys/class/rfkill/rfkill[0-9]+/hard*
- */sys/class/rfkill/rfkill[0-9]+/soft*

5.1.34 File stable/sysfs-class-tpm

Has the following ABI:

- */sys/class/tpm/tpmX/device/*
- */sys/class/tpm/tpmX/device/active*
- */sys/class/tpm/tpmX/device/cancel*
- */sys/class/tpm/tpmX/device/caps*
- */sys/class/tpm/tpmX/device/durations*
- */sys/class/tpm/tpmX/device/enabled*
- */sys/class/tpm/tpmX/device/owned*
- */sys/class/tpm/tpmX/device/pcrs*
- */sys/class/tpm/tpmX/device/pubek*
- */sys/class/tpm/tpmX/device/temp_deactivated*
- */sys/class/tpm/tpmX/device/timeouts*
- */sys/class/tpm/tpmX/tpm_version_major*

- */sys/class/tpm/tpmX/pcr-<H>/<N>*

5.1.35 File stable/sysfs-class-ubi

Has the following ABI:

- */sys/class/ubi/*
- */sys/class/ubi/version*
- */sys/class/ubiX/*
- */sys/class/ubi/ubiX/avail_eraseblocks*
- */sys/class/ubi/ubiX/bad_peb_count*
- */sys/class/ubi/ubiX/bgt_enabled*
- */sys/class/ubi/ubiX/dev*
- */sys/class/ubi/ubiX/eraseblock_size*
- */sys/class/ubi/ubiX/max_ec*
- */sys/class/ubi/ubiX/max_vol_count*
- */sys/class/ubi/ubiX/min_io_size*
- */sys/class/ubi/ubiX/mtd_num*
- */sys/class/ubi/ubiX/reserved_for_bad*
- */sys/class/ubi/ubiX/ro_mode*
- */sys/class/ubi/ubiX/total_eraseblocks*
- */sys/class/ubi/ubiX/volumes_count*
- */sys/class/ubi/ubiX/ubiX_Y/*
- */sys/class/ubi/ubiX/ubiX_Y/alignment*
- */sys/class/ubi/ubiX/ubiX_Y/corrupted*
- */sys/class/ubi/ubiX/ubiX_Y/data_bytes*
- */sys/class/ubi/ubiX/ubiX_Y/dev*
- */sys/class/ubi/ubiX/ubiX_Y/name*
- */sys/class/ubi/ubiX/ubiX_Y/reserved_ebs*
- */sys/class/ubi/ubiX/ubiX_Y/type*
- */sys/class/ubi/ubiX/ubiX_Y/upd_marker*
- */sys/class/ubi/ubiX/ubiX_Y/usable_eb_size*

5.1.36 File stable/sysfs-class-udc

Has the following ABI:

- */sys/class/udc/<udc>/a_alt_hnp_support*
- */sys/class/udc/<udc>/a_hnp_support*
- */sys/class/udc/<udc>/b_hnp_enable*
- */sys/class/udc/<udc>/current_speed*
- */sys/class/udc/<udc>/is_a_peripheral*
- */sys/class/udc/<udc>/is_otg*
- */sys/class/udc/<udc>/maximum_speed*
- */sys/class/udc/<udc>/soft_connect*
- */sys/class/udc/<udc>/srp*
- */sys/class/udc/<udc>/state*
- */sys/class/udc/<udc>/function*

5.1.37 File stable/sysfs-devices

Note:
This documents additional properties of any device beyond what is documented in *Rules on how to access information in sysfs*

Has the following ABI:

- */sys/devices/*/of_node*
- */sys/devices/*/devspec*
- */sys/devices/*/obppath*
- */sys/devices/*/dev*

5.1.38 File stable/sysfs-devices-node

Has the following ABI:

- */sys/devices/system/node/possible*
- */sys/devices/system/node/online*
- */sys/devices/system/node/has_normal_memory*
- */sys/devices/system/node/has_cpu*
- */sys/devices/system/node/has_high_memory*
- */sys/devices/system/node/nodeX*
- */sys/devices/system/node/nodeX/cpumap*
- */sys/devices/system/node/nodeX/cpulist*

- */sys/devices/system/node/nodeX/meminfo*
- */sys/devices/system/node/nodeX/numastat*
- */sys/devices/system/node/nodeX/distance*
- */sys/devices/system/node/nodeX/vmstat*
- */sys/devices/system/node/nodeX/compact*
- */sys/devices/system/node/nodeX/hugepages/hugepages-<size>/*
- */sys/devices/system/node/nodeX/accessY/*
- */sys/devices/system/node/nodeX/accessY/initiators/*
- */sys/devices/system/node/nodeX/accessY/targets/*
- */sys/devices/system/node/nodeX/accessY/initiators/read_bandwidth*
- */sys/devices/system/node/nodeX/accessY/initiators/read_latency*
- */sys/devices/system/node/nodeX/accessY/initiators/write_bandwidth*
- */sys/devices/system/node/nodeX/accessY/initiators/write_latency*
- */sys/devices/system/node/nodeX/memory_side_cache/indexY/*
- */sys/devices/system/node/nodeX/memory_side_cache/indexY/indexing*
- */sys/devices/system/node/nodeX/memory_side_cache/indexY/line_size*
- */sys/devices/system/node/nodeX/memory_side_cache/indexY/size*
- */sys/devices/system/node/nodeX/memory_side_cache/indexY/write_policy*
- */sys/devices/system/node/nodeX/x86/sgx_total_bytes*
- */sys/devices/system/node/nodeX/memory_failure/total*
- */sys/devices/system/node/nodeX/memory_failure/ignored*
- */sys/devices/system/node/nodeX/memory_failure/failed*
- */sys/devices/system/node/nodeX/memory_failure/delayed*
- */sys/devices/system/node/nodeX/memory_failure/recovered*

5.1.39 File stable/sysfs-devices-system-cpu

Has the following ABI:

- */sys/devices/system/cpu/dscr_default*
- */sys/devices/system/cpu/cpu[0-9]+/dscr*
- */sys/devices/system/cpu/cpuX/topology/physical_package_id*
- */sys/devices/system/cpu/cpuX/topology/die_id*
- */sys/devices/system/cpu/cpuX/topology/core_id*
- */sys/devices/system/cpu/cpuX/topology/cluster_id*
- */sys/devices/system/cpu/cpuX/topology/book_id*

- */sys/devices/system/cpu/cpuX/topology/drawer_id*
- */sys/devices/system/cpu/cpuX/topology/core_cpus*
- */sys/devices/system/cpu/cpuX/topology/core_cpus_list*
- */sys/devices/system/cpu/cpuX/topology/package_cpus*
- */sys/devices/system/cpu/cpuX/topology/package_cpus_list*
- */sys/devices/system/cpu/cpuX/topology/die_cpus*
- */sys/devices/system/cpu/cpuX/topology/ppin*
- */sys/devices/system/cpu/cpuX/topology/die_cpus_list*
- */sys/devices/system/cpu/cpuX/topology/cluster_cpus*
- */sys/devices/system/cpu/cpuX/topology/cluster_cpus_list*
- */sys/devices/system/cpu/cpuX/topology/book_siblings*
- */sys/devices/system/cpu/cpuX/topology/book_siblings_list*
- */sys/devices/system/cpu/cpuX/topology/drawer_siblings*
- */sys/devices/system/cpu/cpuX/topology/drawer_siblings_list*

5.1.40 File stable/sysfs-devices-system-xen_memory

Has the following ABI:

- */sys/devices/system/xen_memory/xen_memory0/max_retry_count*
- */sys/devices/system/xen_memory/xen_memory0/max_schedule_delay*
- */sys/devices/system/xen_memory/xen_memory0/retry_count*
- */sys/devices/system/xen_memory/xen_memory0/schedule_delay*
- */sys/devices/system/xen_memory/xen_memory0/target*
- */sys/devices/system/xen_memory/xen_memory0/target_kb*
- */sys/devices/system/xen_memory/xen_memory0/info/current_kb*
- */sys/devices/system/xen_memory/xen_memory0/info/high_kb*
- */sys/devices/system/xen_memory/xen_memory0/info/low_kb*
- */sys/devices/system/xen_memory/xen_memory0/scrub_pages*

5.1.41 File stable/sysfs-driver-aspeed-vuart

Has the following ABI:

- */sys/bus/platform/drivers/aspeed-vuart/*/lpc_address*
- */sys/bus/platform/drivers/aspeed-vuart/*/sirq*
- */sys/bus/platform/drivers/aspeed-vuart/*/sirq_polarity*

5.1.42 File stable/sysfs-driver-dma-idxd

Has the following ABI:

- */sys/bus/dsa/devices/dsa<m>/version*
- */sys/bus/dsa/devices/dsa<m>/cdev_major*
- */sys/bus/dsa/devices/dsa<m>/errors*
- */sys/bus/dsa/devices/dsa<m>/max_batch_size*
- */sys/bus/dsa/devices/dsa<m>/max_work_queues_size*
- */sys/bus/dsa/devices/dsa<m>/max_engines*
- */sys/bus/dsa/devices/dsa<m>/max_groups*
- */sys/bus/dsa/devices/dsa<m>/max_read_buffers*
- */sys/bus/dsa/devices/dsa<m>/max_transfer_size*
- */sys/bus/dsa/devices/dsa<m>/max_work_queues*
- */sys/bus/dsa/devices/dsa<m>/numa_node*
- */sys/bus/dsa/devices/dsa<m>/op_cap*
- */sys/bus/dsa/devices/dsa<m>/pasid_enabled*
- */sys/bus/dsa/devices/dsa<m>/state*
- */sys/bus/dsa/devices/dsa<m>/group<m>.<n>*
- */sys/bus/dsa/devices/dsa<m>/engine<m>.<n>*
- */sys/bus/dsa/devices/dsa<m>/wq<m>.<n>*
- */sys/bus/dsa/devices/dsa<m>/configurable*
- */sys/bus/dsa/devices/dsa<m>/read_buffer_limit*
- */sys/bus/dsa/devices/dsa<m>/cmd_status*
- */sys/bus/dsa/devices/dsa<m>/iaa_cap*
- */sys/bus/dsa/devices/dsa<m>/event_log_size*
- */sys/bus/dsa/devices/wq<m>.<n>/block_on_fault*
- */sys/bus/dsa/devices/wq<m>.<n>/group_id*
- */sys/bus/dsa/devices/wq<m>.<n>/size*
- */sys/bus/dsa/devices/wq<m>.<n>/type*
- */sys/bus/dsa/devices/wq<m>.<n>/cdev_minor*
- */sys/bus/dsa/devices/wq<m>.<n>/mode*
- */sys/bus/dsa/devices/wq<m>.<n>/priority*
- */sys/bus/dsa/devices/wq<m>.<n>/state*
- */sys/bus/dsa/devices/wq<m>.<n>/threshold*
- */sys/bus/dsa/devices/wq<m>.<n>/max_transfer_size*

- */sys/bus/dsa/devices/wq<m>.<n>/max_batch_size*
- */sys/bus/dsa/devices/wq<m>.<n>/ats_disable*
- */sys/bus/dsa/devices/wq<m>.<n>/prs_disable*
- */sys/bus/dsa/devices/wq<m>.<n>/occupancy*
- */sys/bus/dsa/devices/wq<m>.<n>/enqcmds_retries*
- */sys/bus/dsa/devices/wq<m>.<n>/op_config*
- */sys/bus/dsa/devices/wq<m>.<n>/driver_name*
- */sys/bus/dsa/devices/engine<m>.<n>/group_id*
- */sys/bus/dsa/devices/group<m>.<n>/use_read_buffer_limit*
- */sys/bus/dsa/devices/group<m>.<n>/read_buffers_allowed*
- */sys/bus/dsa/devices/group<m>.<n>/read_buffers_reserved*
- */sys/bus/dsa/devices/group<m>.<n>/desc_progress_limit*
- */sys/bus/dsa/devices/group<m>.<n>/batch_progress_limit*
- */sys/bus/dsa/devices/wq<m>.<n>/dsa<x>\!wq<m>.<n>/file<y>/cr_faults*
- */sys/bus/dsa/devices/wq<m>.<n>/dsa<x>\!wq<m>.<n>/file<y>/cr_fault_failures*
- */sys/bus/dsa/devices/wq<m>.<n>/dsa<x>\!wq<m>.<n>/file<y>/pid*

5.1.43 File stable/sysfs-driver-dma-ioatdma

Has the following ABI:

- */sys/devices/pciXXXX:XX/0000:XX:XX.X/dma/dma<n>chan<n>/quickdata/cap*
- */sys/devices/pciXXXX:XX/0000:XX:XX.X/dma/dma<n>chan<n>/quickdata/ring_active*
- */sys/devices/pciXXXX:XX/0000:XX:XX.X/dma/dma<n>chan<n>/quickdata/ring_size*
- */sys/devices/pciXXXX:XX/0000:XX:XX.X/dma/dma<n>chan<n>/quickdata/version*
- */sys/devices/pciXXXX:XX/0000:XX:XX.X/dma/dma<n>chan<n>/quickdata/intr_coalesce*

5.1.44 File stable/sysfs-driver-firmware-zynqmp

Has the following ABI:

- */sys/devices/platform/firmware\:zynqmp-firmware/ggs**
- */sys/devices/platform/firmware\:zynqmp-firmware/pggs**
- */sys/devices/platform/firmware\:zynqmp-firmware/shutdown_scope*
- */sys/devices/platform/firmware\:zynqmp-firmware/health_status*
- */sys/devices/platform/firmware\:zynqmp-firmware/feature_config_id*
- */sys/devices/platform/firmware\:zynqmp-firmware/feature_config_value*

5.1.45 File stable/sysfs-driver-ib_srp

Has the following ABI:

- */sys/class/infiniband_srp/srp-<hca>-<port_number>/add_target*
- */sys/class/infiniband_srp/srp-<hca>-<port_number>/ibdev*
- */sys/class/infiniband_srp/srp-<hca>-<port_number>/port*
- */sys/class/scsi_host/host<n>/allow_ext_sg*
- */sys/class/scsi_host/host<n>/ch_count*
- */sys/class/scsi_host/host<n>/cmd_sg_entries*
- */sys/class/scsi_host/host<n>/comp_vector*
- */sys/class/scsi_host/host<n>/dgid*
- */sys/class/scsi_host/host<n>/id_ext*
- */sys/class/scsi_host/host<n>/ioc_guid*
- */sys/class/scsi_host/host<n>/local_ib_device*
- */sys/class/scsi_host/host<n>/local_ib_port*
- */sys/class/scsi_host/host<n>/orig_dgid*
- */sys/class/scsi_host/host<n>/pkey*
- */sys/class/scsi_host/host<n>/req_lim*
- */sys/class/scsi_host/host<n>/service_id*
- */sys/class/scsi_host/host<n>/sgid*
- */sys/class/scsi_host/host<n>/zero_req_lim*

5.1.46 File stable/sysfs-driver-mlxreg-io

Has the following ABI:

- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/asic_health*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld1_version*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld2_version*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/fan_dir*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld3_version*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/jtag_enable*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/select_iio*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/psu1_on*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_aux_pwr_or_ref*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_asic_thermal*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_hotswap_or_halt*

- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_hotswap_or_wd*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_fw_reset*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_long_pb*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_main_pwr_fail*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_short_pb*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_sw_reset*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_comex_pwr_fail*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_from_comex*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_system*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_voltmon_upgrade_fail*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld4_version*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_comex_thermal*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_comex_wd*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_from_asic*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_reload_bios*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_sff_wd*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_swb_wd*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/config1*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/config2*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_ac_pwr_fail*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_platform*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_soc*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_sw_pwr_off*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/pcie_asic_reset_dis*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/vpd_wp*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/voltreg_update_status*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/ufm_version*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld1_pn*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld2_pn*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld3_pn*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld4_pn*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld1_version_min*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld2_version_min*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld3_version_min*

- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld4_version_min
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/bios_active_image
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/bios_auth_fail
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/bios_upgrade_fail
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc1_enable
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc2_enable
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc3_enable
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc4_enable
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc5_enable
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc6_enable
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc7_enable
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc8_enable
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc1_pwr
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc2_pwr
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc3_pwr
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc4_pwr
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc5_pwr
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc6_pwr
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc7_pwr
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc8_pwr
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc1_rst_mask
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc2_rst_mask
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc3_rst_mask
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc4_rst_mask
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc5_rst_mask
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc6_rst_mask
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc7_rst_mask
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lc8_rst_mask
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/os_started
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/pm_mgmt_en
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/psu3_on
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/psu4_on
- /sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/shutdown_unlock
- /sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/cpld1_pn

- */sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/cpld1_version*
- */sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/cpld1_version_min*
- */sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/fpga1_pn*
- */sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/fpga1_version*
- */sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/fpga1_version_min*
- */sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/vpd_wp*
- */sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/reset_aux_pwr_or_ref*
- */sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/reset_dc_dc_pwr_fail*
- */sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/reset_fpga_not_done*
- */sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/reset_from_chassis*
- */sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/reset_line_card*
- */sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/reset_pwr_off_from_chassis*
- */sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/cpld_upgrade_en*
- */sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/fpga_upgrade_en*
- */sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/qsfp_pwr_en*
- */sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/pwr_en*
- */sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/agb_spi_burn_en*
- */sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/fpga_spi_burn_en*
- */sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/max_power*
- */sys/devices/platform/mlxplat/i2c_mlxcpld.*/i2c-*/i2c-*/i2c-*/*-0032/mlxreg-io.*/hwmon/hwmon*/config*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/phy_reset*

- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/mac_reset*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/qsfp_pwr_good*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/asic2_health*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/asic_reset*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/asic2_reset*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/comm_chnl_ready*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/config3*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_pwr_converter_fail*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/erot1_ap_reset*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/erot2_ap_reset*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/erot1_recovery*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/erot2_recovery*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/erot1_reset*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/erot2_reset*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/erot1_wp*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/erot2_wp*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/spi_chnl_select*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/asic_pg_fail*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/clk_brd1_boot_fail*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/clk_brd2_boot_fail*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/clk_brd_fail*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/clk_brd_prog_en*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/pwr_converter_prog_en*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_ac_ok_fail*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld5_pn*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld5_version*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/cpld5_version_min*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/jtag_cap*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/lid_open*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_long_pwr_pb*
- */sys/devices/platform/mlxplat/mlxreg-io/hwmon/hwmon*/reset_swb_dc_dc_pwr_fail*

5.1.47 File stable/sysfs-driver-qla2xxx

Has the following ABI:

- */sys/bus/pci/drivers/qla2xxx/.../devices/**

5.1.48 File stable/sysfs-driver-speakup

Has the following ABI:

- */sys/accessibility/speakup/attrib_bleep*
- */sys/accessibility/speakup/bell_pos*
- */sys/accessibility/speakup/bleeps*
- */sys/accessibility/speakup/bleep_time*
- */sys/accessibility/speakup/cursor_time*
- */sys/accessibility/speakup/cur_phonetic*
- */sys/accessibility/speakup/delimiters*
- */sys/accessibility/speakup/ex_num*
- */sys/accessibility/speakup/key_echo*
- */sys/accessibility/speakup/keymap*
- */sys/accessibility/speakup/no_interrupt*
- */sys/accessibility/speakup/punc_all*
- */sys/accessibility/speakup/punc_level*
- */sys/accessibility/speakup/punc_most*
- */sys/accessibility/speakup/punc_some*
- */sys/accessibility/speakup/reading_punc*
- */sys/accessibility/speakup/repeats*
- */sys/accessibility/speakup/say_control*
- */sys/accessibility/speakup/say_word_ctl*
- */sys/accessibility/speakup/silent*
- */sys/accessibility/speakup/spell_delay*
- */sys/accessibility/speakup/synth*
- */sys/accessibility/speakup/synth_direct*
- */sys/accessibility/speakup/version*
- */sys/accessibility/speakup/i18n/announcements*
- */sys/accessibility/speakup/i18n/chartab*
- */sys/accessibility/speakup/i18n/ctl_keys*
- */sys/accessibility/speakup/i18n/function_names*

- */sys/accessibility/speakup/i18n/states*
- */sys/accessibility/speakup/i18n/characters*
- */sys/accessibility/speakup/i18n/colors*
- */sys/accessibility/speakup/i18n/formatted*
- */sys/accessibility/speakup/i18n/key_names*
- */sys/accessibility/speakup/<synth-name>/*
- */sys/accessibility/speakup/<synth-name>/caps_start*
- */sys/accessibility/speakup/<synth-name>/caps_stop*
- */sys/accessibility/speakup/<synth-name>/delay_time*
- */sys/accessibility/speakup/<synth-name>/direct*
- */sys/accessibility/speakup/<synth-name>/freq*
- */sys/accessibility/speakup/<synth-name>/flush_time*
- */sys/accessibility/speakup/<synth-name>/full_time*
- */sys/accessibility/speakup/<synth-name>/jiffy_delta*
- */sys/accessibility/speakup/<synth-name>/pitch*
- */sys/accessibility/speakup/<synth-name>/inflection*
- */sys/accessibility/speakup/<synth-name>/punct*
- */sys/accessibility/speakup/<synth-name>/rate*
- */sys/accessibility/speakup/<synth-name>/tone*
- */sys/accessibility/speakup/<synth-name>/trigger_time*
- */sys/accessibility/speakup/<synth-name>/voice*
- */sys/accessibility/speakup/<synth-name>/vol*

5.1.49 File stable/sysfs-driver-usb-usbtmc

Has the following ABI:

- */sys/bus/usb/drivers/usbtmc/*/interface_capabilities*
- */sys/bus/usb/drivers/usbtmc/*/device_capabilities*
- */sys/bus/usb/drivers/usbtmc/*/usb488_interface_capabilities*
- */sys/bus/usb/drivers/usbtmc/*/usb488_device_capabilities*

5.1.50 File stable/sysfs-driver-w1_ds2438

Has the following ABI:

- */sys/bus/w1/devices/.../page1*
- */sys/bus/w1/devices/.../offset*

5.1.51 File stable/sysfs-driver-w1_ds28e04

Has the following ABI:

- */sys/bus/w1/devices/.../pio*
- */sys/bus/w1/devices/.../eeprom*

5.1.52 File stable/sysfs-driver-w1_ds28ea00

Has the following ABI:

- */sys/bus/w1/devices/.../w1_seq*

5.1.53 File stable/sysfs-firmware-efi-vars

Has the following ABI:

- */sys/firmware/efi/vars*

5.1.54 File stable/sysfs-firmware-opal-dump

Has the following ABI:

- */sys/firmware/opal/dump*

5.1.55 File stable/sysfs-firmware-opal-elog

Has the following ABI:

- */sys/firmware/opal/elog*

5.1.56 File stable/sysfs-fs-orangefs

Has the following ABI:

- */sys/fs/orangefs/perf_counters/**
- */sys/fs/orangefs/perf_counter_reset*
- */sys/fs/orangefs/perf_time_interval_secs*
- */sys/fs/orangefs/perf_history_size*
- */sys/fs/orangefs/op_timeout_secs*

- */sys/fs/orangefs/slot_timeout_secs*
- */sys/fs/orangefs/acache/**
- */sys/fs/orangefs/ncache/**
- */sys/fs/orangefs/capcache/**
- */sys/fs/orangefs/ccache/**

5.1.57 File stable/sysfs-hypervisor-xen

Has the following ABI:

- */sys/hypervisor/compilation/compile_date*
- */sys/hypervisor/compilation/compiled_by*
- */sys/hypervisor/compilation/compiler*
- */sys/hypervisor/properties/capabilities*
- */sys/hypervisor/properties/changeset*
- */sys/hypervisor/properties/features*
- */sys/hypervisor/properties/pagesize*
- */sys/hypervisor/properties/virtual_start*
- */sys/hypervisor/type*
- */sys/hypervisor/uuid*
- */sys/hypervisor/version/extra*
- */sys/hypervisor/version/major*
- */sys/hypervisor/version/minor*
- */sys/hypervisor/start_flags/**

5.1.58 File stable/sysfs-kernel-notes

Has the following ABI:

- */sys/kernel/notes*

5.1.59 File stable/sysfs-module

The /sys/module tree consists of the following structure:

Has the following ABI:

- */sys/module/<MODULENAME>*
- */sys/module/<MODULENAME>/parameters*
- */sys/module/<MODULENAME>/refcnt*
- */sys/module/<MODULENAME>/srcversion*

- */sys/module/<MODULENAME>/version*

5.1.60 File stable/sysfs-platform-wmi-bmof

Has the following ABI:

- */sys/bus/wmi/devices/05901221-D566-11D1-B2F0-00A0C9062910[-X]/bmof*

5.1.61 File stable/sysfs-transport-srp

Has the following ABI:

- */sys/class/srp_remote_ports/port-<h>:<n>/delete*
- */sys/class/srp_remote_ports/port-<h>:<n>/dev_loss_tmo*
- */sys/class/srp_remote_ports/port-<h>:<n>/fast_io_fail_tmo*
- */sys/class/srp_remote_ports/port-<h>:<n>/port_id*
- */sys/class/srp_remote_ports/port-<h>:<n>/reconnect_delay*
- */sys/class/srp_remote_ports/port-<h>:<n>/roles*
- */sys/class/srp_remote_ports/port-<h>:<n>/state*

5.1.62 File stable/thermal-notification

Has the following ABI:

- *A notification mechanism for thermal related events*

5.1.63 File stable/vdso

Has the following ABI:

- *vDSO*

5.2 ABI testing symbols

Documents interfaces that are felt to be stable, as the main development of this interface has been completed.

The interface can be changed to add new features, but the current interface will not break by doing this, unless grave errors or security problems are found in them.

Userspace programs can start to rely on these interfaces, but they must be aware of changes that can occur before these interfaces move to be marked stable.

Programs that use these interfaces are strongly encouraged to add their name to the description of these interfaces, so that the kernel developers can easily notify them if any changes occur.

5.2.1 .../XXXXXXX.ipa/endpoint_id/

.../XXXXXXX.ipa/endpoint_id/

Defined on file *sysfs-devices-platform-soc-ipa*

The .../XXXXXXX.ipa/endpoint_id/ directory contains attributes that define IDs associated with IPA endpoints. The "rx" or "tx" in an endpoint name is from the perspective of the AP. An endpoint ID is a small unsigned integer.

5.2.2 .../XXXXXXX.ipa/endpoint_id/modem_rx

.../XXXXXXX.ipa/endpoint_id/modem_rx

Defined on file *sysfs-devices-platform-soc-ipa*

The .../XXXXXXX.ipa/endpoint_id/modem_rx file contains the ID of the AP endpoint on which packets originating from the embedded modem are received.

5.2.3 .../XXXXXXX.ipa/endpoint_id/modem_tx

.../XXXXXXX.ipa/endpoint_id/modem_tx

Defined on file *sysfs-devices-platform-soc-ipa*

The .../XXXXXXX.ipa/endpoint_id/modem_tx file contains the ID of the AP endpoint on which packets destined for the embedded modem are sent.

5.2.4 .../XXXXXXX.ipa/endpoint_id/monitor_rx

.../XXXXXXX.ipa/endpoint_id/monitor_rx

Defined on file *sysfs-devices-platform-soc-ipa*

The .../XXXXXXX.ipa/endpoint_id/monitor_rx file contains the ID of the AP endpoint on which IPA "monitor" data is received. The monitor endpoint supplies replicas of packets that enter the IPA hardware for processing. Each replicated packet is preceded by a fixed-size "ODL" header (see .../XXXXXXX.ipa/feature/monitor, above). Large packets are truncated, to reduce the bandwidth required to provide the monitor function.

5.2.5 .../XXXXXXX.ipa/feature/

.../XXXXXXX.ipa/feature/

Defined on file *sysfs-devices-platform-soc-ipa*

The .../XXXXXXX.ipa/feature/ directory contains a set of attributes describing features implemented by the IPA hardware.

5.2.6 .../XXXXXXX.ipa/feature/rx_offload

.../XXXXXXX.ipa/feature/rx_offload

Defined on file *sysfs-devices-platform-soc-ipa*

The .../XXXXXXX.ipa/feature/rx_offload file contains a string indicating the type of receive checksum offload that is supported by the hardware. The possible values are "MAPv4" or "MAPv5".

5.2.7 .../XXXXXXX.ipa/feature/tx_offload

.../XXXXXXX.ipa/feature/tx_offload

Defined on file *sysfs-devices-platform-soc-ipa*

The .../XXXXXXX.ipa/feature/tx_offload file contains a string indicating the type of transmit checksum offload that is supported by the hardware. The possible values are "MAPv4" or "MAPv5".

5.2.8 .../XXXXXXX.ipa/modem/

.../XXXXXXX.ipa/modem/

Defined on file *sysfs-devices-platform-soc-ipa*

The .../XXXXXXX.ipa/modem/ directory contains attributes describing properties of the modem embedded in the SoC.

5.2.9 .../XXXXXXX.ipa/modem/rx_endpoint_id

.../XXXXXXX.ipa/modem/rx_endpoint_id

Defined on file *sysfs-devices-platform-soc-ipa*

The .../XXXXXXX.ipa/modem/rx_endpoint_id file duplicates the value found in .../XXXXXXX.ipa/endpoint_id/modem_rx.

5.2.10 .../XXXXXXX.ipa/modem/tx_endpoint_id

.../XXXXXXX.ipa/modem/tx_endpoint_id

Defined on file *sysfs-devices-platform-soc-ipa*

The .../XXXXXXX.ipa/modem/tx_endpoint_id file duplicates the value found in .../XXXXXXX.ipa/endpoint_id/modem_tx.

5.2.11 .../XXXXXXX.ipa/version

.../XXXXXXX.ipa/version

Defined on file *sysfs-devices-platform-soc-ipa*

The .../XXXXXXX.ipa/version file contains the IPA hardware version, as a period-separated set of two or three integers (e.g., "3.5.1" or "4.2").

5.2.12 Symbols under /config/acpi

/config/acpi

Defined on file *configfs-acpi*

This represents the ACPI subsystem entry point directory. It contains sub-groups corresponding to ACPI configurable options.

/config/acpi/table

Defined on file *configfs-acpi*

This group contains the configuration for user defined ACPI tables. The attributes of a user define table are:

aml

- a binary attribute that the user can use to fill in the ACPI aml definitions. Once the aml data is written to this file and the file is closed the table will be loaded and ACPI devices will be enumerated. To check if the operation is successful the user must check the error code for close(). If the operation is successful, subsequent writes to this attribute will fail.

The rest of the attributes are read-only and are valid only after the table has been loaded by filling the aml entry:

signature

- ASCII table signature

length

- length of table in bytes, including the header

revision

- ACPI Specification minor version number

oem_id

- ASCII OEM identification

oem_table_id

- ASCII OEM table identification

oem_revision

- OEM revision number

asl_compiler_id

- ASCII ASL compiler vendor ID

asl_compiler_revision

- ASL compiler version

5.2.13 Symbols under /config/iio

/config/iio

Defined on file *configfs-iio*

This represents Industrial IO configuration entry point directory. It contains sub-groups corresponding to IIO objects.

/config/iio/devices

Defined on file *configfs-iio*

Industrial IO software devices directory.

/config/iio/devices/dummy

Defined on file *configfs-iio*

Dummy IIO devices directory. Creating a directory here will result in creating a dummy IIO device in the IIO subsystem.

/config/iio/triggers

Defined on file *configfs-iio*

Industrial IO software triggers directory.

/config/iio/triggers/hrtimers

Defined on file *configfs-iio*

High resolution timers directory. Creating a directory here will result in creating a hrtimer trigger in the IIO subsystem.

5.2.14 Symbols under /config/pcie-gadget

/config/pcie-gadget

Defined on file *configfs-spear-pcie-gadget*

Interface is used to configure selected dual mode PCIe controller as device and then program its various registers to configure it as a particular device type. This interfaces can be used to show spear's PCIe device capability.

Nodes are only visible when configfs is mounted. To mount configfs in /config directory use:

```
# mount -t configfs none /config/
```

For nth PCIe Device Controller */config/pcie-gadget*.n/:

link	used to enable ltssm and read its status.
int_type	used to configure and read type of supported interrupt
no_of_msi	used to configure number of MSI vector needed and to read no of MSI granted.
inta	write 1 to assert INTA and 0 to de-assert.
send_msi	write MSI vector to be sent.
vendor_id	used to write and read vendor id (hex)
device_id	used to write and read device id (hex)
bar0_size	used to write and read bar0_size
bar0_address	used to write and read bar0 mapped area in hex.
bar0_rw_offset	used to write and read offset of bar0 where bar0_data will be written or read.
bar0_data	used to write and read data at bar0_rw_offset.

5.2.15 Symbols under /config/rdma_cm

/config/rdma_cm

Defined on file *configfs-rdma_cm*

Interface is used to configure RDMA-cable HCAs in respect to RDMA-CM attributes.

Attributes are visible only when configfs is mounted. To mount configfs in /config directory use:
mount -t configfs none /config/

In order to set parameters related to a specific HCA, a directory for this HCA has to be created:
mkdir -p /config/rdma_cm/<hca>

/config/rdma_cm/<hca>/ports/<port-num>/default_roce_mode

Defined on file *configfs-rdma_cm*

RDMA-CM based connections from HCA <hca> at port <port-num> will be initiated with this RoCE type as default. The possible RoCE types are either "IB/RoCE v1" or "RoCE v2". This parameter has RW access.

/config/rdma_cm/<hca>/ports/<port-num>/default_roce_tos

Defined on file *configfs-rdma_cm*

RDMA-CM QPs from HCA <hca> at port <port-num> will be created with this TOS as default. This can be overridden by using the rdma_set_option API. The possible RoCE TOS values are 0-255.

5.2.16 Symbols under /config/stp-policy

/config/stp-policy

Defined on file *configfs-stp-policy*

This group contains policies mandating Master/Channel allocation for software sources wishing to send trace data over an STM device.

/config/stp-policy/<device>.<policy>

Defined on file *configfs-stp-policy*

This group is the root of a policy; its name is a concatenation of an stm device name to which this policy applies and an arbitrary string. If <device> part doesn't match an existing stm device, mkdir will fail with ENODEV; if that device already has a policy assigned to it, mkdir will fail with EBUSY.

/config/stp-policy/<device>.<policy>/<node>

Defined on file *configfs-stp-policy*

Policy node is a string identifier that software clients will use to request a master/channel to be allocated and assigned to them.

/config/stp-policy/<device>.<policy>/<node>/channels

Defined on file *configfs-stp-policy*

Range of channels from which to allocate for users of this node. Write two numbers: the first channel and the last channel number.

/config/stp-policy/<device>.<policy>/<node>/masters

Defined on file *configfs-stp-policy*

Range of masters from which to allocate for users of this node. Write two numbers: the first master and the last master number.

/config/stp-policy/<device>.<policy>/device

Defined on file *configfs-stp-policy*

STM device to which this policy applies, read only. Same as the <device> component of its parent directory.

/config/stp-policy/<device>:p_sys-t.<policy>/<node>/clocksync_interval

Defined on file *configfs-stp-policy-p_sys-t*

Time interval in milliseconds. Send a CLOCKSYNC packet if this many milliseconds have passed since the previous CLOCKSYNC packet from this source. Zero is the default and stands for "never send the CLOCKSYNC". It makes sense to use this option with sources that generate constant and/or periodic data, like stm_heartbeat.

/config/stp-policy/<device>:p_sys-t.<policy>/<node>/do_len

Defined on file *configfs-stp-policy-p_sys-t*

Include payload length in the MIPI SyS-T header, boolean. If enabled, the SyS-T protocol encoder will include payload length in each packet's metadata. This is normally redundant if the underlying transport protocol supports marking message boundaries (which STP does), so this is off by default.

/config/stp-policy/<device>:p_sys-t.<policy>/<node>/ts_interval

Defined on file *configfs-stp-policy-p_sys-t*

Time interval in milliseconds. Include a timestamp in the MIPI SyS-T packet metadata, if this many milliseconds have passed since the previous packet from this source. Zero is the default and stands for "never send the timestamp".

/config/stp-policy/<device>:p_sys-t.<policy>/<node>/uuid

Defined on file *configfs-stp-policy-p_sys-t*

UUID source identifier string, RW. Default value is randomly generated at the mkdir <node> time. Data coming from trace sources that use this <node> will be tagged with this UUID in the MIPI SyS-T packet stream, to allow the decoder to discern between different sources within the same master/channel range, and identify the higher level decoders that may be needed for each source.

5.2.17 Symbols under /config/usb-gadget

/config/usb-gadget

Defined on file *configfs-usb-gadget*

This group contains sub-groups corresponding to created USB gadgets.

/config/usb-gadget/gadget

Defined on file *configfs-usb-gadget*

The attributes of a gadget:

UDC	bind a gadget to UDC/unbind a gadget; write UDC's name found in /sys/class/udc/* to bind a gadget, empty string "" to unbind.
max_speed	maximum speed the driver supports. Valid names are super-speed-plus, super-speed, high-speed, full-speed, and low-speed.
bDeviceClass	USB device class code
bDeviceSubClass	USB device subclass code
bDeviceProtocol	USB device protocol code
bMaxPacketSize0	maximum endpoint 0 packet size
bcdDevice	bcd device release number
bcdUSB	bcd USB specification version number
idProduct	product ID
idVendor	vendor ID

/config/usb-gadget/gadget/configs

Defined on file *configfs-usb-gadget*

This group contains a USB gadget's configurations

/config/usb-gadget/gadget/configs/config

Defined on file *configfs-usb-gadget*

The attributes of a configuration:

bmAttributes	configuration characteristics
MaxPower	maximum power consumption from the bus

/config/usb-gadget/gadget/configs/config/strings

Defined on file *configfs-usb-gadget*

This group contains subdirectories for language-specific strings for this configuration.

/config/usb-gadget/gadget/configs/config/strings/language

Defined on file *configfs-usb-gadget*

The attributes:

configuration	configuration description

/config/usb-gadget/gadget/functions

Defined on file *configfs-usb-gadget*

This group contains functions available to this USB gadget.

/config/usb-gadget/gadget/functions/<func>.<inst>/interface.<n>

Defined on file *configfs-usb-gadget*

This group contains ”Feature Descriptors” specific for one gadget's USB interface or one interface group described by an IAD.

The attributes:

compatible_id	8-byte string for ”Compatible ID”
sub_compatible_id	8-byte string for ”Sub Compatible ID”

/config/usb-gadget/gadget/functions/<func>.<inst>/interface.<n>/<property>

Defined on file *configfs-usb-gadget*

This group contains ”Extended Property Descriptors” specific for one gadget's USB interface or one interface group described by an IAD.

The attributes:

type	value 1..7 for interpreting the data • 1: unicode string • 2: unicode string with environment variable • 3: binary • 4: little-endian 32-bit • 5: big-endian 32-bit • 6: unicode string with a symbolic link • 7: multiple unicode strings
data	blob of data to be interpreted depending on type

/config/usb-gadget/gadget/functions/Loopback.name

Defined on file *configfs-usb-gadget-loopback*

The attributes:

qlen	depth of loopback queue
buflen	buffer length

/config/usb-gadget/gadget/functions/SourceSink.name

Defined on file *configfs-usb-gadget-sourcesink*

The attributes:

pattern	0 (all zeros), 1 (mod63), 2 (none)
isoc_interval	1..16
isoc_maxpacket	0 - 1023 (fs), 0 - 1024 (hs/ss)
isoc_mult	0..2 (hs/ss only)
isoc_maxburst	0..15 (ss only)
buflen	buffer length
bulk_qlen	depth of queue for bulk
iso_qlen	depth of queue for iso

/config/usb-gadget/gadget/functions/acm.name

Defined on file *configfs-usb-gadget-acm*

This item contains just one readonly attribute: port_num. It contains the port number of the /dev/ttyGS<n> device associated with acm function's instance "name".

/config/usb-gadget/gadget/functions/ecm.name

Defined on file *configfs-usb-gadget-ecm*

The attributes:

ifname

- network device interface name associated with this function instance

qmult

- queue length multiplier for high and super speed

host_addr

- MAC address of host's end of this Ethernet over USB link

dev_addr

- MAC address of device's end of this Ethernet over USB link

/config/usb-gadget/gadget/functions/eem.name

Defined on file *configfs-usb-gadget-eem*

The attributes:

ifname	network device interface name associated with this function instance
qmult	queue length multiplier for high and super speed
host_addr	MAC address of host's end of this Ethernet over USB link
dev_addr	MAC address of device's end of this Ethernet over USB link

/config/usb-gadget/gadget/functions/ffs.name

Defined on file *configfs-usb-gadget-ffs*

The purpose of this directory is to create and remove it.

A corresponding USB function instance is created/removed. There are no attributes here.

All parameters are set through FunctionFS.

/config/usb-gadget/gadget/functions/geth.name

Defined on file *configfs-usb-gadget-subset*

The attributes:

ifname	network device interface name associated with this function instance
qmult	queue length multiplier for high and super speed
host_addr	MAC address of host's end of this Ethernet over USB link
dev_addr	MAC address of device's end of this Ethernet over USB link

/config/usb-gadget/gadget/functions/gser.name

Defined on file *configfs-usb-gadget-serial*

This item contains just one readonly attribute: port_num. It contains the port number of the /dev/ttyGS<n> device associated with gser function's instance "name".

/config/usb-gadget/gadget/functions/hid.name

Defined on file *configfs-usb-gadget-hid*

The attributes:

protocol	HID protocol to use
report_desc	blob corresponding to HID report descriptors except the data passed through /dev/hidg<N>
report_length	HID report length
subclass	HID device subclass to use

/config/usb-gadget/gadget/functions/mass_storage.name

Defined on file *configfs-usb-gadget-mass-storage*

The attributes:

stall	Set to permit function to halt bulk endpoints. Disabled on some USB devices known not to work correctly. You should set it to true.
num_buffers	Number of pipeline buffers. Valid numbers are 2..4. Available only if CONFIG_USB_GADGET_DEBUG_FILES is set.

/config/usb-gadget/gadget/functions/mass_storage.name/lun.name

Defined on file *configfs-usb-gadget-mass-storage*

The attributes:

file	The path to the backing file for the LUN. Required if LUN is not marked as removable.
ro	Flag specifying access to the LUN shall be read-only. This is implied if CD-ROM emulation is enabled as well as when it was impossible to open "filename" in R/W mode.
removable	Flag specifying that LUN shall be indicated as being removable.
cdrom	Flag specifying that LUN shall be reported as being a CD-ROM.
nofua	Flag specifying that FUA flag in SCSI WRITE(10,12)
forced_eject	This write-only file is useful only when the function is active. It causes the backing file to be forcibly detached from the LUN, regardless of whether the host has allowed it. Any non-zero number of bytes written will result in ejection.

/config/usb-gadget/gadget/functions/midi.name

Defined on file *configfs-usb-gadget-midi*

The attributes:

index	index value for the USB MIDI adapter
id	ID string for the USB MIDI adapter
buflen	MIDI buffer length
qlen	USB read request queue length
in_ports	number of MIDI input ports
out_ports	number of MIDI output ports

/config/usb-gadget/gadget/functions/midi2.name

Defined on file *configfs-usb-gadget-midi2*

The attributes:

process_ump	Flag to process UMP Stream messages (0 or 1)
static_block	Flag for static blocks (0 or 1)
iface_name	MIDI interface name string

/config/usb-gadget/gadget/functions/midi2.name/ep.number

Defined on file *configfs-usb-gadget-midi2*

This group contains a UMP Endpoint configuration. A new Endpoint starts from 0, and can be up to 3.

The attributes:

protocol_caps	MIDI protocol capabilities (1, 2 or 3 for both)
protocol	Default MIDI protocol (1 or 2)
ep_name	UMP Endpoint name string
product_id	Product ID string
manufacturer	Manufacture ID (24 bit)
family	Device family ID (16 bit)
model	Device model ID (16 bit)
sw_revision	Software Revision (32 bit)

/config/usb-gadget/gadget/functions/midi2.name/ep.number/block.number

Defined on file *configfs-usb-gadget-midi2*

This group contains a UMP Function Block configuration. A new block starts from 0, and can be up to 31.

The attributes:

name	Function Block name string
direction	1: input, 2: output, 3: bidirectional
first_group	The first UMP Group number (0-15)
num_groups	The number of groups in this FB (1-16)
midi1_first_group	The first UMP Group number for MIDI 1.0 (0-15)
midi1_num_groups	The number of groups for MIDI 1.0 (0-16)
ui_hint	0: unknown, 1: receiver, 2: sender, 3: both
midi_ci_verison	Supported MIDI-CI version number (8 bit)
is_midi1	Legacy MIDI 1.0 device (0, 1 or 2)
sysex8_streams	Max number of SysEx8 streams (8 bit)
active	Active FB flag (0 or 1)

/config/usb-gadget/gadget/functions/ncm.name

Defined on file *configfs-usb-gadget-ncm*

The attributes:

ifname - network device interface name associated with
this function instance

qmult - queue length multiplier for high and
super speed

host_addr - MAC address of host's end of this
Ethernet over USB link

dev_addr - MAC address of device's end of this
Ethernet over USB link

/config/usb-gadget/gadget/functions/obex.name

Defined on file *configfs-usb-gadget-obex*

This item contains just one readonly attribute: port_num. It contains the port number of the /dev/ttyGS<n> device associated with obex function's instance "name".

/config/usb-gadget/gadget/functions/phonet.name

Defined on file *configfs-usb-gadget-phonet*

This item contains just one readonly attribute: ifname. It contains the network interface name assigned during network device registration.

/config/usb-gadget/gadget/functions/printer.name

Defined on file *configfs-usb-gadget-printer*

The attributes:

pnp_string	Data to be passed to the host in pnp string
q_len	Number of requests per endpoint

/config/usb-gadget/gadget/functions/rndis.name

Defined on file *configfs-usb-gadget-rndis*

The attributes:

ifname	network device interface name associated with this function instance
qmult	queue length multiplier for high and super speed
host_addr	MAC address of host's end of this Ethernet over USB link
dev_addr	MAC address of device's end of this Ethernet over USB link
class	USB interface class, default is 02 (hex)
subclass	USB interface subclass, default is 06 (hex)
protocol	USB interface protocol, default is 00 (hex)

/config/usb-gadget/gadget/functions/tcm.name

Defined on file *configfs-usb-gadget-tcm*

There are no attributes because all the configuration is performed in the "target" subsystem of configfs.

/config/usb-gadget/gadget/functions/uac1.name

Defined on file *configfs-usb-gadget-uac1*

The attributes:

c_chmask	capture channel mask
c_srate	list of capture sampling rates (comma-separated)
c_ssize	capture sample size (bytes)
c_mute_present	capture mute control enable
c_volume_present	capture volume control enable
c_volume_min	capture volume control min value (in 1/256 dB)
c_volume_max	capture volume control max value (in 1/256 dB)
c_volume_res	capture volume control resolution (in 1/256 dB)
p_chmask	playback channel mask
p_srate	list of playback sampling rates (comma-separated)
p_ssize	playback sample size (bytes)
p_mute_present	playback mute control enable
p_volume_present	playback volume control enable
p_volume_min	playback volume control min value (in 1/256 dB)
p_volume_max	playback volume control max value (in 1/256 dB)
p_volume_res	playback volume control resolution (in 1/256 dB)
req_number	the number of pre-allocated requests for both capture and playback
function_name	name of the interface

/config/usb-gadget/gadget/functions/uac1_legacy.name

Defined on file *configfs-usb-gadget-uac1_legacy*

The attributes:

audio_buf_size - audio buffer size fn_cap - capture pcm device file name fn_cntl - control device file name fn_play - playback pcm device file name req_buf_size - ISO OUT endpoint request buffer size req_count - ISO OUT endpoint request count

/config/usb-gadget/gadget/functions/uac2.name

Defined on file *configfs-usb-gadget-uac2*

The attributes:

c_chmask	capture channel mask
c_srate	list of capture sampling rates (comma-separated)
c_ssize	capture sample size (bytes)
c_hs_bint	capture bInterval for HS/SS (1-4: fixed, 0: auto)
c_sync	capture synchronization type (async/adaptive)
c_mute_present	capture mute control enable
c_volume_present	capture volume control enable
c_volume_min	capture volume control min value (in 1/256 dB)
c_volume_max	capture volume control max value (in 1/256 dB)
c_volume_res	capture volume control resolution (in 1/256 dB)
fb_max	maximum extra bandwidth in async mode
p_chmask	playback channel mask
p_srate	list of playback sampling rates (comma-separated)
p_ssize	playback sample size (bytes)
p_hs_bint	playback bInterval for HS/SS (1-4: fixed, 0: auto)
p_mute_present	playback mute control enable
p_volume_present	playback volume control enable
p_volume_min	playback volume control min value (in 1/256 dB)
p_volume_max	playback volume control max value (in 1/256 dB)
p_volume_res	playback volume control resolution (in 1/256 dB)
req_number	the number of pre-allocated requests for both capture and playback
function_name	name of the interface
c_terminal_type	code of the capture terminal type
p_terminal_type	code of the playback terminal type

/config/usb-gadget/gadget/functions/uvc.name

Defined on file *configfs-usb-gadget-uvc*

UVC function directory

streaming_maxburst	0..15 (ss only)
streaming_maxpacket	1..1023 (fs), 1..3072 (hs/ss)
streaming_interval	1..16
function_name	string [32]

/config/usb-gadget/gadget/functions/uvc.name/control

Defined on file *configfs-usb-gadget-uvc*

Control descriptors

All attributes read only except enable_interrupt_ep:

bInterfaceNumber	USB interface number for this streaming interface
enable_interrupt_ep	flag to enable the interrupt endpoint for the VC interface

/config/usb-gadget/gadget/functions/uvc.name/control/class

Defined on file *configfs-usb-gadget-uvc*

Class descriptors

/config/usb-gadget/gadget/functions/uvc.name/control/class/fs

Defined on file *configfs-usb-gadget-uvc*

Full speed control class descriptors

/config/usb-gadget/gadget/functions/uvc.name/control/class/ss

Defined on file *configfs-usb-gadget-uvc*

Super speed control class descriptors

/config/usb-gadget/gadget/functions/uvc.name/control/extensions

Defined on file *configfs-usb-gadget-uvc*

Extension unit descriptors

/config/usb-gadget/gadget/functions/uvc.name/control/extensions/name

Defined on file *configfs-usb-gadget-uvc*

Extension Unit (XU) Descriptor

bLength, bUnitID and iExtension are read-only. All others are read-write.

bLength	size of the descriptor in bytes
bUnitID	non-zero ID of this unit
guidExtensionCode	Vendor-specific code identifying the XU
bNumControls	number of controls in this XU
bNrInPins	number of input pins for this unit
baSourceID	list of the IDs of the units or terminals to which this XU is connected
bControlSize	size of the bmControls field in bytes
bmControls	list of bitmaps detailing which vendor specific controls are supported
iExtension	index of a string descriptor that describes this extension unit

/config/usb-gadget/gadget/functions/uvc.name/control/header

Defined on file *configfs-usb-gadget-uvc*

Control header descriptors

/config/usb-gadget/gadget/functions/uvc.name/control/header/name

Defined on file *configfs-usb-gadget-uvc*

Specific control header descriptors

dwClockFrequency bcdUVC

/config/usb-gadget/gadget/functions/uvc.name/control/processing

Defined on file *configfs-usb-gadget-uvc*

Processing unit descriptors

/config/usb-gadget/gadget/functions/uvc.name/control/processing/default

Defined on file *configfs-usb-gadget-uvc*

Default processing unit descriptors

All attributes read only except bmControls, which is read/write:

iProcessing	index of string descriptor
bmControls	bitmap specifying which controls are supported for the video stream
wMaxMultiplier	maximum digital magnification x100
bSourceID	id of the terminal to which this unit is connected
bUnitID	a non-zero id of this unit

/config/usb-gadget/gadget/functions/uvc.name/control/terminal

Defined on file *configfs-usb-gadget-uvc*

Terminal descriptors

/config/usb-gadget/gadget/functions/uvc.name/control/terminal/camera

Defined on file *configfs-usb-gadget-uvc*

Camera terminal descriptors

/config/usb-gadget/gadget/functions/uvc.name/control/terminal/camera/default

Defined on file *configfs-usb-gadget-uvc*

Default camera terminal descriptors

All attributes read only except bmControls, which is read/write:

bmControls	bitmap specifying which controls are supported for the video stream
wOcularFocalLength	the value of Locular
wObjectiveFocalLengthMax	the value of Lmin
wObjectiveFocalLengthMin	the value of Lmax
iTerminal	index of string descriptor
bAssocTerminal	id of the output terminal to which this terminal is connected
wTerminalType	terminal type
bTerminalID	a non-zero id of this terminal

/config/usb-gadget/gadget/functions/uvc.name/control/terminal/output

Defined on file *configfs-usb-gadget-uvc*

Output terminal descriptors

/config/usb-gadget/gadget/functions/uvc.name/control/terminal/output/default

Defined on file *configfs-usb-gadget-uvc*

Default output terminal descriptors

All attributes read only except bSourceID:

iTerminal	index of string descriptor
bSourceID	id of the terminal to which this terminal is connected
bAssocTerminal	id of the input terminal to which this output terminal is associated
wTerminalType	terminal type
bTerminalID	a non-zero id of this terminal

/config/usb-gadget/gadget/functions/uvc.name/streaming

Defined on file *configfs-usb-gadget-uvc*

Streaming descriptors

All attributes read only:

bInterfaceNumber	USB interface number for this streaming interface

/config/usb-gadget/gadget/functions/uvc.name/streaming/class

Defined on file *configfs-usb-gadget-uvc*

Streaming class descriptors

/config/usb-gadget/gadget/functions/uvc.name/streaming/class/fs

Defined on file *configfs-usb-gadget-uvc*

Full speed streaming class descriptors

/config/usb-gadget/gadget/functions/uvc.name/streaming/class/hs

Defined on file *configfs-usb-gadget-uvc*

High speed streaming class descriptors

/config/usb-gadget/gadget/functions/uvc.name/streaming/class/ss

Defined on file *configfs-usb-gadget-uvc*

Super speed streaming class descriptors

/config/usb-gadget/gadget/functions/uvc.name/streaming/color_matching

Defined on file *configfs-usb-gadget-uvc*

Color matching descriptors

/config/usb-gadget/gadget/functions/uvc.name/streaming/color_matching/default

Defined on file *configfs-usb-gadget-uvc*

Default color matching descriptors

All attributes read/write:

bMatrixCoefficients	matrix used to compute luma and chroma values from the color primaries
bTransferCharacteristics	optoelectronic transfer characteristic of the source picture, also called the gamma function
bColorPrimaries	color primaries and the reference white

/config/usb-gadget/gadget/functions/uvc.name/streaming/color_matching/name

Defined on file *configfs-usb-gadget-uvc*

Additional color matching descriptors

All attributes read/write:

bMatrixCoefficients	matrix used to compute luma and chroma values from the color primaries
bTransferCharacteristics	optoelectronic transfer characteristic of the source picture, also called the gamma function
bColorPrimaries	color primaries and the reference white

/config/usb-gadget/gadget/functions/uvc.name/streaming/header

Defined on file *configfs-usb-gadget-uvc*

Streaming header descriptors

/config/usb-gadget/gadget/functions/uvc.name/streaming/header/name

Defined on file *configfs-usb-gadget-uvc*

Specific streaming header descriptors

All attributes read only:

bTriggerUsage	how the host software will respond to a hardware trigger interrupt event
bTriggerSupport	flag specifying if hardware triggering is supported
bStillCaptureMethod	method of still image capture supported
bTerminalLink	id of the output terminal to which the video endpoint of this interface is connected
bmInfo	capabilities of this video streaming interface

/config/usb-gadget/gadget/functions/uvc.name/streaming/mjpeg

Defined on file *configfs-usb-gadget-uvc*

MJPEG format descriptors

/config/usb-gadget/gadget/functions/uvc.name/streaming/mjpeg/name

Defined on file *configfs-usb-gadget-uvc*

Specific MJPEG format descriptors

All attributes read only, except bmaControls and bDefaultFrameIndex:

bFormatIndex	unique id for this format descriptor; only defined after parent header is linked into the streaming class; read-only
bmaControls	this format's data for bmaControls in the streaming header
bmInterlaceFlags	specifies interlace information, read-only
bAspectRatioY	the X dimension of the picture aspect ratio, read-only
bAspectRatioX	the Y dimension of the picture aspect ratio, read-only
bmFlags	characteristics of this format, read-only
bDefaultFrameIndex	optimum frame index for this stream

/config/usb-gadget/gadget/functions/uvc.name/streaming/mjpeg/name/name

Defined on file *configfs-usb-gadget-uvc*

Specific MJPEG frame descriptors

bFrameIndex	unique id for this framedescriptor; only defined after parent format is linked into the streaming header; read-only
dwFrameInterval	indicates how frame interval can be programmed; a number of values separated by newline can be specified
dwDefaultFrameInterval	the frame interval the device would like to use as default
dwMaxVideoFrameBufferSize	the maximum number of bytes the compressor will produce for a video frame or still image
dwMaxBitRate	the maximum bit rate at the shortest frame interval in bps
dwMinBitRate	the minimum bit rate at the longest frame interval in bps
wHeight	height of decoded bitmap frame in px
wWidth	width of decoded bitmam frame in px
bmCapabilities	still image support, fixed frame-rate support

/config/usb-gadget/gadget/functions/uvc.name/streaming/uncompressed

Defined on file *configfs-usb-gadget-uvc*

Uncompressed format descriptors

/config/usb-gadget/gadget/functions/uvc.name/streaming/uncompressed/name

Defined on file *configfs-usb-gadget-uvc*

Specific uncompressed format descriptors

bFormatIndex	unique id for this format descriptor; only defined after parent header is linked into the streaming class; read-only
bmaControls	this format's data for bmaControls in the streaming header
bmInterlaceFlags	specifies interlace information, read-only
bAspectRatioY	the X dimension of the picture aspect ratio, read-only
bAspectRatioX	the Y dimension of the picture aspect ratio, read-only
bDefaultFrameIndex	optimum frame index for this stream
bBitsPerPixel	number of bits per pixel used to specify color in the decoded video frame
guidFormat	globally unique id used to identify stream-encoding format

/config/usb-gadget/gadget/functions/uvc.name/streaming/uncompressed/name/name

Defined on file *configfs-usb-gadget-uvc*

Specific uncompressed frame descriptors

bFrameIndex	unique id for this framedescriptor; only defined after parent format is linked into the streaming header; read-only
dwFrameInterval	indicates how frame interval can be programmed; a number of values separated by newline can be specified
dwDefaultFrameInterval	the frame interval the device would like to use as default
dwMaxVideoFrameBufferSize	the maximum number of bytes the compressor will produce for a video frame or still image
dwMaxBitRate	the maximum bit rate at the shortest frame interval in bps
dwMinBitRate	the minimum bit rate at the longest frame interval in bps
wHeight	height of decoded bitmap frame in px
wWidth	width of decoded bitmam frame in px
bmCapabilities	still image support, fixed frame-rate support

/config/usb-gadget/gadget/os_desc

Defined on file *configfs-usb-gadget*

This group contains ”OS String” extension handling attributes.

use	flag turning ”OS Descriptors” support on/off
b_vendor_code	one-byte value used for custom per-device and per-interface requests
qw_sign	an identifier to be reported as ”OS String” proper

/config/usb-gadget/gadget/strings

Defined on file *configfs-usb-gadget*

This group contains subdirectories for language-specific strings for this gadget.

/config/usb-gadget/gadget/strings/language

Defined on file *configfs-usb-gadget*

The attributes:

serialnumber	gadget's serial number (string)
product	gadget's product description
manufacturer	gadget's manufacturer description

/config/usb-gadget/gadget/webusb

Defined on file *configfs-usb-gadget*

This group contains ”WebUSB” extension handling attributes.

use	flag turning ”WebUSB” support on/off
bcdVersion	bcd WebUSB specification version number
bVendorCode	one-byte value used for custom per-device
landingPage	UTF-8 encoded URL of the device's landing page

5.2.18 Symbols under /dev/gpiochip

/dev/gpiochip[0-9]+

Defined on file *gpio-cdev*

The character device files /dev/gpiochip* are the interface between GPIO chips and userspace.

The ioctl(2)-based ABI is defined and documented in [include/uapi]<linux/gpio.h>.

The following file operations are supported:

open(2)
Currently the only useful flags are O_RDWR.

ioctl(2)
Initiate various actions.

See the inline documentation in [include/uapi]<linux/gpio.h> for descriptions of all ioctls.

close(2)
Stops and free up the I/O contexts that was associated with the file descriptor.

Users: TBD

5.2.19 Symbols under /dev/kmsg

/dev/kmsg

Defined on file *dev-kmsg*

The */dev/kmsg* character device node provides userspace access to the kernel's printk buffer.

Injecting messages:

Every write() to the opened device node places a log entry in the kernel's printk buffer.

The logged line can be prefixed with a <N> syslog prefix, which carries the syslog priority and facility. The single decimal prefix number is composed of the 3 lowest bits being the syslog priority and the next 8 bits the syslog facility number.

If no prefix is given, the priority number is the default kernel log priority and the facility number is set to LOG_USER (1). It is not possible to inject messages from userspace with the facility number LOG_KERN (0), to make sure that the origin of the messages can always be reliably determined.

Accessing the buffer:

Every read() from the opened device node receives one record of the kernel's printk buffer.

The first read() directly following an open() always returns first message in the buffer; there is no kernel-internal persistent state; many readers can concurrently open the device and read from it, without affecting other readers.

Every read() will receive the next available record. If no more records are available read() will block, or if O_NONBLOCK is used -EAGAIN returned.

Messages in the record ring buffer get overwritten as whole, there are never partial messages received by read().

In case messages get overwritten in the circular buffer while the device is kept open, the next read() will return -EPIPE, and the seek position be updated to the next available record. Subsequent reads() will return available records again.

Unlike the classic syslog() interface, the 64 bit record sequence numbers allow to calculate the amount of lost messages, in case the buffer gets overwritten. And they allow to reconnect to the buffer and reconstruct the read position if needed, without limiting the interface to a single reader.

The device supports seek with the following parameters:

SEEK_SET, 0
seek to the first entry in the buffer

SEEK_END, 0
seek after the last entry in the buffer

SEEK_DATA, 0
: seek after the last record available at the time the last SYSLOG_ACTION_CLEAR was issued.

Other seek operations or offsets are not supported because of the special behavior this device has. The device allows to read or write only whole variable length messages (records) that are stored in a ring buffer.

Because of the non-standard behavior also the error values are non-standard. -ESPIPE is returned for non-zero offset. -EINVAL is returned for other operations, e.g. SEEK_CUR. This behavior and values are historical and could not be modified without the risk of breaking userspace.

The output format consists of a prefix carrying the syslog prefix including priority and facility, the 64 bit message sequence number and the monotonic timestamp in microseconds, and a flag field. All fields are separated by a ','.

Future extensions might add more comma separated values before the terminating ';'. Unknown fields and values should be gracefully ignored.

The human readable text string starts directly after the ';' and is terminated by a 'n'. Untrusted values derived from hardware or other facilities are printed, therefore all non-printable characters and '' itself in the log message are escaped by "x00" C-style hex encoding.

A line starting with ' ', is a continuation line, adding key/value pairs to the log message, which provide the machine readable context of the message, for reliable processing in userspace.

Example:

```
7,160,424069,-;pci_root PNP0A03:00: host bridge window [io  0x0000-0x0cf7]␣
↪(ignored)
 SUBSYSTEM=acpi
 DEVICE=+acpi:PNP0A03:00
6,339,5140900,-;NET: Registered protocol family 10
30,340,5690716,-;udevd[80]: starting version 181
```

The DEVICE= key uniquely identifies devices the following way:

b12:8	block dev_t
c127:3	char dev_t
n8	netdev ifindex
+sound:card0	subsystem:devname

The flags field carries '-' by default. A 'c' indicates a fragment of a line. Note, that these hints about continuation lines are not necessarily correct, and the stream could be interleaved with unrelated messages, but merging the lines in the output usually produces better human readable results. A similar logic is used internally when messages are printed to the console, /proc/kmsg or the syslog() syscall.

By default, kernel tries to avoid fragments by concatenating when it can and fragments are rare; however, when extended console support is enabled, the in-kernel concatenation is disabled and */dev/kmsg* output will contain more fragments. If the log consumer performs concatenation, the end result should be the same. In the future, the in-kernel concatenation may be removed entirely and */dev/kmsg* users are recommended to implement fragment handling.

Users: dmesg(1), userspace kernel log consumers

5.2.20 Symbols under /dev/rtcX

/dev/rtcX

Defined on file *rtc-cdev*

The ioctl interface to drivers for real-time clocks (RTCs). Following actions are supported:

- RTC_RD_TIME, RTC_SET_TIME: Read or set the RTC time. Time format is a Gregorian calendar date and 24 hour wall clock time.
- RTC_AIE_ON, RTC_AIE_OFF: Enable or disable the alarm interrupt for RTCs that support alarms
- RTC_ALM_READ, RTC_ALM_SET: Read or set the alarm time for RTCs that support alarms. Can be set upto 24 hours in the future. Requires a separate RTC_AIE_ON call to enable the alarm interrupt. (Prefer to use RTC_WKALM_*)
- RTC_WKALM_RD, RTC_WKALM_SET: For RTCs that support a more powerful interface, which can issue alarms beyond 24 hours and enable IRQs in the same request.
- RTC_PIE_ON, RTC_PIE_OFF: Enable or disable the periodic interrupt for RTCs that support periodic interrupts.
- RTC_UIE_ON, RTC_UIE_OFF: Enable or disable the update interrupt for RTCs that support it.
- RTC_IRQP_READ, RTC_IRQP_SET: Read or set the frequency for periodic interrupts for RTCs that support periodic interrupts. Requires a separate RTC_PIE_ON call to enable the periodic interrupts.
- RTC_VL_READ: Read the voltage inputs status of the RTC when supported. The value is a bit field of RTC_VL_*, giving the status of the main and backup voltages.
- RTC_VL_CLEAR: Clear the voltage status of the RTC. Some RTCs need user interaction when the backup power provider is replaced or charged to be able to clear the status.

The ioctl() calls supported by the older /dev/rtc interface are also supported by the newer RTC class framework. However, because the chips and systems are not standardized, some PC/AT functionality might not be provided. And in the same way, some newer features -- including those enabled by ACPI -- are exposed by the RTC class framework, but can't be supported by the older driver.

5.2.21 Symbols under /dev/wmi

/dev/wmi/dell-smbios

Defined on file *dell-smbios-wmi*

Perform SMBIOS calls on supported Dell machines. through the Dell ACPI-WMI interface.

IOCTL's and buffer formats are defined in: <uapi/linux/wmi.h>

1) To perform an SMBIOS call from userspace, you'll need to first determine the minimum size of the calling interface buffer for your machine. Platforms that contain larger buffers

can return larger objects from the system firmware. Commonly this size is either 4k or 32k.

To determine the size of the buffer read() a u64 dword from the WMI character device */dev/wmi/dell-smbios*.

2) After you've determined the minimum size of the calling interface buffer, you can allocate a structure that represents the structure documented above.

3) In the 'length' object store the size of the buffer you determined above and allocated.

4) In this buffer object, prepare as necessary for the SMBIOS call you're interested in. Typically SMBIOS buffers have ”class”, ”select”, and ”input” defined to values that coincide with the data you are interested in. Documenting class/select/input values is outside of the scope of this documentation. Check with the libsmbios project for further documentation on these values.

6) Run the call by using ioctl() as described in the header.

7) The output will be returned in the buffer object.

8) Be sure to free up your allocated object.

5.2.22 Symbols under /kvd

/kvd/

Defined on file *devlink-resource-mlxsw*

The main database in the Spectrum device is a centralized KVD database used for many of the tables used to configure the chip including L2 FDB, L3 LPM, ECMP and more. The KVD is divided into two sections, the first is hash-based table and the second is a linear access table. The division between the linear and hash-based sections is static and require reload before the changes take effect.

5.2.23 Symbols under /kvd/hash_double

/kvd/hash_double

Defined on file *devlink-resource-mlxsw*

The hash based section of the KVD is managed by the switch device. Used in case the key is larger than 64 bit.

5.2.24 Symbols under /kvd/hash_single

/kvd/hash_single

Defined on file *devlink-resource-mlxsw*

The hash based section of the KVD is managed by the switch device. Used in case the key size is smaller or equal to 64bit.

5.2.25 Symbols under /kvd/linear

/kvd/linear

Defined on file *devlink-resource-mlxsw*

The linear section of the KVD is managed by software as a flat memory accessed using an index.

5.2.26 Symbols under /proc

/proc/*/attr/current

Defined on file *procfs-attr-current*

The current security information used by a Linux security module (LSM) that is active on the system. The details of permissions required to read from this interface and hence obtain the security state of the task identified is LSM dependent. A process cannot write to this interface unless it refers to itself. The other details of permissions required to write to this interface and hence change the security state of the task identified are LSM dependent. The format of the data used by this interface is LSM dependent. SELinux, Smack and AppArmor provide this interface.

Users: SELinux user-space Smack user-space AppArmor user-space

/proc/*/attr/exec

Defined on file *procfs-attr-exec*

The security information to be used on the process by a Linux security module (LSM) active on the system after a subsequent exec() call. The details of permissions required to read from this interface and hence obtain the security state of the task identified is LSM dependent. A process cannot write to this interface unless it refers to itself. The other details of permissions required to write to this interface and hence change the security state of the task identified are LSM dependent. The format of the data used by this interface is LSM dependent. SELinux and AppArmor provide this interface.

Users: SELinux user-space AppArmor user-space

/proc/*/attr/prev

Defined on file *procfs-attr-prev*

The security information used on the process by a Linux security module (LSM) active on the system prior to the most recent exec() call. The details of permissions required to read from this interface is LSM dependent. A process cannot write to this interface unless it refers to itself. The other details of permissions required to write to this interface are LSM dependent. The format of the data used by this interface is LSM dependent. SELinux and AppArmor provide this interface.

Users: SELinux user-space AppArmor user-space

5.2.27 Symbols under /proc/diskstats

/proc/diskstats

Defined on file *procfs-diskstats*

The */proc/diskstats* file displays the I/O statistics of block devices. Each line contains the following 14 fields:

1	major number
2	minor number
3	device name
4	reads completed successfully
5	reads merged
6	sectors read
7	time spent reading (ms)
8	writes completed
9	writes merged
10	sectors written
11	time spent writing (ms)
12	I/Os currently in progress
13	time spent doing I/Os (ms)
14	weighted time spent doing I/Os (ms)

Kernel 4.18+ appends four more fields for discard tracking putting the total at 18:

15	discards completed successfully
16	discards merged
17	sectors discarded
18	time spent discarding

Kernel 5.5+ appends two more fields for flush requests:

19	flush requests completed successfully
20	time spent flushing

For more details refer to *I/O statistics fields*

5.2.28 Symbols under /proc/pid

/proc/pid/smaps_rollup

Defined on file *procfs-smaps_rollup*

This file provides pre-summed memory information for a process. The format is almost identical to /proc/pid/smaps, except instead of an entry for each VMA in a process, smaps_rollup has a single entry (tagged "[rollup]") for which each field is the sum of the corresponding fields from all the maps in /proc/pid/smaps. Additionally, the fields Pss_Anon, Pss_File and Pss_Shmem are not present in /proc/pid/smaps. These fields represent the sum of the Pss field of each type (anon, file, shmem). For more details, see /filesystems/proc and the procfs man page.

Typical output looks like this:

```
00100000-ff709000 ---p 00000000 00:00 0          [rollup]
Size:               1192 kB
KernelPageSize:        4 kB
MMUPageSize:           4 kB
Rss:                 884 kB
Pss:                 385 kB
Pss_Dirty:            68 kB
Pss_Anon:            301 kB
Pss_File:             80 kB
Pss_Shmem:             4 kB
Shared_Clean:        696 kB
Shared_Dirty:          0 kB
Private_Clean:       120 kB
Private_Dirty:        68 kB
Referenced:          884 kB
Anonymous:            68 kB
LazyFree:              0 kB
AnonHugePages:         0 kB
ShmemPmdMapped:        0 kB
Shared_Hugetlb:        0 kB
Private_Hugetlb:       0 kB
Swap:                  0 kB
SwapPss:               0 kB
Locked:              385 kB
```

5.2.29 Symbols under /sys

/sys/.../<device>/<UUID>/

Defined on file *sysfs-bus-vfio-mdev*

This directory represents device directory of mediated device. It contains all the attributes related to mediated device.

/sys/.../<device>/<UUID>/mdev_type

Defined on file *sysfs-bus-vfio-mdev*

This is symbolic link pointing to supported type, <type-id> directory of which this mediated device is created.

/sys/.../<device>/<UUID>/remove

Defined on file *sysfs-bus-vfio-mdev*

Writing '1' to this file destroys the mediated device. The vendor driver can fail the remove() callback if that device is active and the vendor driver doesn't support hot unplug. Example:

```
# echo 1 > /sys/bus/mdev/devices/<UUID>/remove
```

/sys/.../<device>/mdev_supported_types/

Defined on file *sysfs-bus-vfio-mdev*

This directory contains list of directories of currently supported mediated device types and their details for <device>. Supported type attributes are defined by the vendor driver who registers with Mediated device framework. Each supported type is a directory whose name is created by adding the device driver string as a prefix to the string provided by the vendor driver.

/sys/.../<device>/mdev_supported_types/<type-id>/

Defined on file *sysfs-bus-vfio-mdev*

This directory gives details of supported type, like name, description, available_instances, device_api etc. 'device_api' and 'available_instances' are mandatory attributes to be provided by vendor driver. 'name', 'description' and other vendor driver specific attributes are optional.

/sys/.../<device>/vfio-dev/vfioX/

Defined on file *sysfs-devices-vfio-dev*

This directory is created when the device is bound to a vfio driver. The layout under this directory matches what exists for a standard 'struct device'. 'X' is a unique index marking this device in vfio.

/sys/.../events/illuminance_threshY_falling_value
/sys/.../events/illuminance_threshY_raising_value

Defined on file *sysfs-bus-iio-light-lm3533-als*

Specifies the value of threshold that the device is comparing against for the events enabled by in_illuminance0_thresh_either_en (0..255), where Y in 0..3.

Note that threshY_falling must be less than or equal to threshY_raising.

These thresholds correspond to the eight zone-boundary registers (boundaryY_{low,high}) and define the five light zones.

/sys/.../events/in_accel_gesture_doubletap_tap2_min_delay

Defined on file *sysfs-bus-iio*

Specifies the minimum quiet time in seconds between the two taps of a double tap.

/sys/.../events/in_accel_gesture_doubletap_tap2_min_delay_available

Defined on file *sysfs-bus-iio*

Lists all available delay values between two taps in the double tap. Units in seconds.

/sys/.../events/in_accel_gesture_singletap_en
/sys/.../events/in_accel_gesture_doubletap_en

Defined on file *sysfs-bus-iio*

Device generates an event on a single or double tap.

/sys/.../events/in_accel_gesture_singletap_reset_timeout
/sys/.../events/in_accel_gesture_doubletap_reset_timeout

Defined on file *sysfs-bus-iio*

Specifies the timeout value in seconds for the tap detector to not to look for another tap event after the event as occurred. Basically the minimum quiet time between the two single-tap's or two double-tap's.

/sys/.../events/in_accel_gesture_singletap_value
/sys/.../events/in_accel_gesture_doubletap_value

Defined on file *sysfs-bus-iio*

Specifies the threshold value that the device is comparing against to generate the tap gesture event. The lower threshold value increases the sensitivity of tap detection. Units and the exact meaning of value are device-specific.

/sys/.../events/in_accel_gesture_tap_maxtomin_time

Defined on file *sysfs-bus-iio*

Specifies the maximum time difference allowed between upper and lower peak of tap to consider it as the valid tap event. Units in seconds.

/sys/.../events/in_accel_gesture_tap_maxtomin_time_available

Defined on file *sysfs-bus-iio*

Lists all available time values between upper peak to lower peak. Units in seconds.

/sys/.../events/in_accel_gesture_tap_reset_timeout_available

Defined on file *sysfs-bus-iio*

Lists all available tap reset timeout values. Units in seconds.

/sys/.../events/in_accel_gesture_tap_value_available

Defined on file *sysfs-bus-iio*

Lists all available threshold values which can be used to modify the sensitivity of the tap detection.

/sys/.../events/in_accel_gesture_tap_wait_dur

Defined on file *sysfs-bus-iio*

Timeout value in seconds for tap gesture confirmation.

/sys/.../events/in_accel_gesture_tap_wait_dur_available

Defined on file *sysfs-bus-iio*

List of available timeout value for tap gesture confirmation.

/sys/.../events/in_accel_gesture_tap_wait_timeout

Defined on file *sysfs-bus-iio*

Enable tap gesture confirmation with timeout.

/sys/.../events/in_accel_raw_mag_value
/sys/.../events/in_accel_x_raw_mag_rising_value
/sys/.../events/in_accel_y_raw_mag_rising_value
/sys/.../events/in_accel_z_raw_mag_rising_value

Defined on file *sysfs-bus-iio*

The value to which the magnitude of the channel is compared. If number or direction is not specified, applies to all channels of this type.

/sys/.../events/in_accel_scale
/sys/.../events/in_accel_peak_scale
/sys/.../events/in_anglvel_scale
/sys/.../events/in_magn_scale
/sys/.../events/in_rot_from_north_magnetic_scale
/sys/.../events/in_rot_from_north_true_scale
/sys/.../events/in_voltage_scale
/sys/.../events/in_voltage_supply_scale
/sys/.../events/in_temp_scale
/sys/.../events/in_illuminance_scale
/sys/.../events/in_proximity_scale

Defined on file *sysfs-bus-iio*

Specifies the conversion factor from the standard units to device specific units used to set the event trigger threshold.

/sys/.../events/in_accel_thresh_rising_high_pass_filter_3db
/sys/.../events/in_anglvel_thresh_rising_high_pass_filter_3db
/sys/.../events/in_magn_thresh_rising_high_pass_filter_3db

Defined on file *sysfs-bus-iio*

If a high pass filter can be applied to the event generation this property gives its 3db frequency in Hz. A value of zero disables the filter.

/sys/.../events/in_accel_thresh_rising_low_pass_filter_3db
/sys/.../events/in_anglvel_thresh_rising_low_pass_filter_3db
/sys/.../events/in_magn_thresh_rising_low_pass_filter_3db

Defined on file *sysfs-bus-iio*

If a low pass filter can be applied to the event generation this property gives its 3db frequency in Hz. A value of zero disables the filter.

/sys/.../events/in_accel_thresh_rising_value
/sys/.../events/in_accel_thresh_falling_value
/sys/.../events/in_accel_x_raw_thresh_rising_value
/sys/.../events/in_accel_x_raw_thresh_falling_value
/sys/.../events/in_accel_y_raw_thresh_rising_value
/sys/.../events/in_accel_y_raw_thresh_falling_value
/sys/.../events/in_accel_z_raw_thresh_rising_value
/sys/.../events/in_accel_z_raw_thresh_falling_value
/sys/.../events/in_anglvel_x_raw_thresh_rising_value
/sys/.../events/in_anglvel_x_raw_thresh_falling_value
/sys/.../events/in_anglvel_y_raw_thresh_rising_value
/sys/.../events/in_anglvel_y_raw_thresh_falling_value
/sys/.../events/in_anglvel_z_raw_thresh_rising_value
/sys/.../events/in_anglvel_z_raw_thresh_falling_value
/sys/.../events/in_magn_x_raw_thresh_rising_value
/sys/.../events/in_magn_x_raw_thresh_falling_value
/sys/.../events/in_magn_y_raw_thresh_rising_value
/sys/.../events/in_magn_y_raw_thresh_falling_value
/sys/.../events/in_magn_z_raw_thresh_rising_value
/sys/.../events/in_magn_z_raw_thresh_falling_value
/sys/.../events/in_rot_from_north_magnetic_raw_thresh_rising_value
/sys/.../events/in_rot_from_north_magnetic_raw_thresh_falling_value
/sys/.../events/in_rot_from_north_true_raw_thresh_rising_value
/sys/.../events/in_rot_from_north_true_raw_thresh_falling_value
/sys/.../events/in_rot_from_north_magnetic_tilt_comp_raw_thresh_rising_value
/sys/.../events/in_rot_from_north_magnetic_tilt_comp_raw_thresh_falling_value
/sys/.../events/in_rot_from_north_true_tilt_comp_raw_thresh_rising_value
/sys/.../events/in_rot_from_north_true_tilt_comp_raw_thresh_falling_value

continues on next page

Table 1 – continued from previous page

/sys/.../events/in_voltageY_supply_raw_thresh_rising_value
/sys/.../events/in_voltageY_supply_raw_thresh_falling_value
/sys/.../events/in_voltageY_raw_thresh_rising_value
/sys/.../events/in_voltageY_raw_thresh_falling_value
/sys/.../events/in_tempY_raw_thresh_rising_value
/sys/.../events/in_tempY_raw_thresh_falling_value
/sys/.../events/in_illuminance0_thresh_falling_value
/sys/.../events/in_illuminance0_thresh_rising_value
/sys/.../events/in_proximity0_thresh_falling_value
/sys/.../events/in_proximity0_thresh_rising_value
/sys/.../events/in_illuminance_thresh_rising_value
/sys/.../events/in_illuminance_thresh_falling_value
/sys/.../events/in_capacitanceY_thresh_rising_value
/sys/.../events/in_capacitanceY_thresh_falling_value
/sys/.../events/in_capacitanceY_thresh_adaptive_rising_value
/sys/.../events/in_capacitanceY_thresh_falling_rising_value

Defined on file *sysfs-bus-iio*

Specifies the value of threshold that the device is comparing against for the events enabled by <type>Y[_name]_thresh[_rising|falling]_en.

If separate attributes exist for the two directions, but direction is not specified for this attribute, then a single threshold value applies to both directions.

The raw or input element of the name indicates whether the value is in raw device units or in processed units (as _raw and _input do on sysfs direct channel read attributes).

/sys/.../events/in_accel_x_raw_roc_rising_value
/sys/.../events/in_accel_x_raw_roc_falling_value
/sys/.../events/in_accel_y_raw_roc_rising_value
/sys/.../events/in_accel_y_raw_roc_falling_value
/sys/.../events/in_accel_z_raw_roc_rising_value
/sys/.../events/in_accel_z_raw_roc_falling_value
/sys/.../events/in_anglvel_x_raw_roc_rising_value
/sys/.../events/in_anglvel_x_raw_roc_falling_value
/sys/.../events/in_anglvel_y_raw_roc_rising_value
/sys/.../events/in_anglvel_y_raw_roc_falling_value
/sys/.../events/in_anglvel_z_raw_roc_rising_value
/sys/.../events/in_anglvel_z_raw_roc_falling_value
/sys/.../events/in_magn_x_raw_roc_rising_value
/sys/.../events/in_magn_x_raw_roc_falling_value
/sys/.../events/in_magn_y_raw_roc_rising_value
/sys/.../events/in_magn_y_raw_roc_falling_value
/sys/.../events/in_magn_z_raw_roc_rising_value
/sys/.../events/in_magn_z_raw_roc_falling_value
/sys/.../events/in_rot_from_north_magnetic_raw_roc_rising_value
/sys/.../events/in_rot_from_north_magnetic_raw_roc_falling_value
/sys/.../events/in_rot_from_north_true_raw_roc_rising_value

continues on next page

Table 2 – continued from previous page

/sys/.../events/in_rot_from_north_true_raw_roc_falling_value
/sys/.../events/in_rot_from_north_magnetic_tilt_comp_raw_roc_rising_value
/sys/.../events/in_rot_from_north_magnetic_tilt_comp_raw_roc_falling_value
/sys/.../events/in_rot_from_north_true_tilt_comp_raw_roc_rising_value
/sys/.../events/in_rot_from_north_true_tilt_comp_raw_roc_falling_value
/sys/.../events/in_voltageY_supply_raw_roc_rising_value
/sys/.../events/in_voltageY_supply_raw_roc_falling_value
/sys/.../events/in_voltageY_raw_roc_rising_value
/sys/.../events/in_voltageY_raw_roc_falling_value
/sys/.../events/in_tempY_raw_roc_rising_value
/sys/.../events/in_tempY_raw_roc_falling_value

Defined on file *sysfs-bus-iio*

Specifies the value of rate of change threshold that the device is comparing against for the events enabled by <type>[Y][_name]_roc[_rising|falling]_en.

If separate attributes exist for the two directions, but direction is not specified for this attribute, then a single threshold value applies to both directions. The raw or input element of the name indicates whether the value is in raw device units or in processed units (as _raw and _input do on sysfs direct channel read attributes).

/sys/.../events/in_accel_x_thresh_rising_hysteresis
/sys/.../events/in_accel_x_thresh_falling_hysteresis
/sys/.../events/in_accel_x_thresh_either_hysteresis
/sys/.../events/in_accel_y_thresh_rising_hysteresis
/sys/.../events/in_accel_y_thresh_falling_hysteresis
/sys/.../events/in_accel_y_thresh_either_hysteresis
/sys/.../events/in_accel_z_thresh_rising_hysteresis
/sys/.../events/in_accel_z_thresh_falling_hysteresis
/sys/.../events/in_accel_z_thresh_either_hysteresis
/sys/.../events/in_anglvel_x_thresh_rising_hysteresis
/sys/.../events/in_anglvel_x_thresh_falling_hysteresis
/sys/.../events/in_anglvel_x_thresh_either_hysteresis
/sys/.../events/in_anglvel_y_thresh_rising_hysteresis
/sys/.../events/in_anglvel_y_thresh_falling_hysteresis
/sys/.../events/in_anglvel_y_thresh_either_hysteresis
/sys/.../events/in_anglvel_z_thresh_rising_hysteresis
/sys/.../events/in_anglvel_z_thresh_falling_hysteresis
/sys/.../events/in_anglvel_z_thresh_either_hysteresis
/sys/.../events/in_magn_x_thresh_rising_hysteresis
/sys/.../events/in_magn_x_thresh_falling_hysteresis
/sys/.../events/in_magn_x_thresh_either_hysteresis
/sys/.../events/in_magn_y_thresh_rising_hysteresis
/sys/.../events/in_magn_y_thresh_falling_hysteresis
/sys/.../events/in_magn_y_thresh_either_hysteresis
/sys/.../events/in_magn_z_thresh_rising_hysteresis
/sys/.../events/in_magn_z_thresh_falling_hysteresis
/sys/.../events/in_magn_z_thresh_either_hysteresis

continues on next page

Table 3 – continued from previous page

/sys/.../events/in_rot_from_north_magnetic_thresh_rising_hysteresis
/sys/.../events/in_rot_from_north_magnetic_thresh_falling_hysteresis
/sys/.../events/in_rot_from_north_magnetic_thresh_either_hysteresis
/sys/.../events/in_rot_from_north_true_thresh_rising_hysteresis
/sys/.../events/in_rot_from_north_true_thresh_falling_hysteresis
/sys/.../events/in_rot_from_north_true_thresh_either_hysteresis
/sys/.../events/in_rot_from_north_magnetic_tilt_comp_thresh_rising_hysteresis
/sys/.../events/in_rot_from_north_magnetic_tilt_comp_thresh_falling_hysteresis
/sys/.../events/in_rot_from_north_magnetic_tilt_comp_thresh_either_hysteresis
/sys/.../events/in_rot_from_north_true_tilt_comp_thresh_rising_hysteresis
/sys/.../events/in_rot_from_north_true_tilt_comp_thresh_falling_hysteresis
/sys/.../events/in_rot_from_north_true_tilt_comp_thresh_either_hysteresis
/sys/.../events/in_voltageY_thresh_rising_hysteresis
/sys/.../events/in_voltageY_thresh_falling_hysteresis
/sys/.../events/in_voltageY_thresh_either_hysteresis
/sys/.../events/in_tempY_thresh_rising_hysteresis
/sys/.../events/in_tempY_thresh_falling_hysteresis
/sys/.../events/in_tempY_thresh_either_hysteresis
/sys/.../events/in_illuminance0_thresh_falling_hysteresis
/sys/.../events/in_illuminance0_thresh_rising_hysteresis
/sys/.../events/in_illuminance0_thresh_either_hysteresis
/sys/.../events/in_proximity0_thresh_falling_hysteresis
/sys/.../events/in_proximity0_thresh_rising_hysteresis
/sys/.../events/in_proximity0_thresh_either_hysteresis

Defined on file *sysfs-bus-iio*

Specifies the hysteresis of threshold that the device is comparing against for the events enabled by <type>Y[_name]_thresh[_(rising|falling)]_hysteresis. If separate attributes exist for the two directions, but direction is not specified for this attribute, then a single hysteresis value applies to both directions.

For falling events the hysteresis is added to the _value attribute for this event to get the upper threshold for when the event goes back to normal, for rising events the hysteresis is subtracted from the _value attribute. E.g. if in_voltage0_raw_thresh_rising_value is set to 1200 and in_voltage0_raw_thresh_rising_hysteresis is set to 50. The event will get activated once in_voltage0_raw goes above 1200 and will become deactivated again once the value falls below 1150.

/sys/.../events/in_accel_x_thresh_rising_period
/sys/.../events/in_accel_x_thresh_falling_period
/sys/.../events/in_accel_x_roc_rising_period
/sys/.../events/in_accel_x_roc_falling_period
/sys/.../events/in_accel_y_thresh_rising_period
/sys/.../events/in_accel_y_thresh_falling_period
/sys/.../events/in_accel_y_roc_rising_period
/sys/.../events/in_accel_y_roc_falling_period
/sys/.../events/in_accel_z_thresh_rising_period
/sys/.../events/in_accel_z_thresh_falling_period

continues on next page

Table 4 – continued from previous page

/sys/.../events/in_accel_z_roc_rising_period
/sys/.../events/in_accel_z_roc_falling_period
/sys/.../events/in_anglvel_x_thresh_rising_period
/sys/.../events/in_anglvel_x_thresh_falling_period
/sys/.../events/in_anglvel_x_roc_rising_period
/sys/.../events/in_anglvel_x_roc_falling_period
/sys/.../events/in_anglvel_y_thresh_rising_period
/sys/.../events/in_anglvel_y_thresh_falling_period
/sys/.../events/in_anglvel_y_roc_rising_period
/sys/.../events/in_anglvel_y_roc_falling_period
/sys/.../events/in_anglvel_z_thresh_rising_period
/sys/.../events/in_anglvel_z_thresh_falling_period
/sys/.../events/in_anglvel_z_roc_rising_period
/sys/.../events/in_anglvel_z_roc_falling_period
/sys/.../events/in_magn_x_thresh_rising_period
/sys/.../events/in_magn_x_thresh_falling_period
/sys/.../events/in_magn_x_roc_rising_period
/sys/.../events/in_magn_x_roc_falling_period
/sys/.../events/in_magn_y_thresh_rising_period
/sys/.../events/in_magn_y_thresh_falling_period
/sys/.../events/in_magn_y_roc_rising_period
/sys/.../events/in_magn_y_roc_falling_period
/sys/.../events/in_magn_z_thresh_rising_period
/sys/.../events/in_magn_z_thresh_falling_period
/sys/.../events/in_magn_z_roc_rising_period
/sys/.../events/in_magn_z_roc_falling_period
/sys/.../events/in_rot_from_north_magnetic_thresh_rising_period
/sys/.../events/in_rot_from_north_magnetic_thresh_falling_period
/sys/.../events/in_rot_from_north_magnetic_roc_rising_period
/sys/.../events/in_rot_from_north_magnetic_roc_falling_period
/sys/.../events/in_rot_from_north_true_thresh_rising_period
/sys/.../events/in_rot_from_north_true_thresh_falling_period
/sys/.../events/in_rot_from_north_true_roc_rising_period
/sys/.../events/in_rot_from_north_true_roc_falling_period
/sys/.../events/in_rot_from_north_magnetic_tilt_comp_thresh_rising_period
/sys/.../events/in_rot_from_north_magnetic_tilt_comp_thresh_falling_period
/sys/.../events/in_rot_from_north_magnetic_tilt_comp_roc_rising_period
/sys/.../events/in_rot_from_north_magnetic_tilt_comp_roc_falling_period
/sys/.../events/in_rot_from_north_true_tilt_comp_thresh_rising_period
/sys/.../events/in_rot_from_north_true_tilt_comp_thresh_falling_period
/sys/.../events/in_rot_from_north_true_tilt_comp_roc_rising_period
/sys/.../events/in_rot_from_north_true_tilt_comp_roc_falling_period
/sys/.../events/in_voltageY_supply_thresh_rising_period
/sys/.../events/in_voltageY_supply_thresh_falling_period
/sys/.../events/in_voltageY_supply_roc_rising_period
/sys/.../events/in_voltageY_supply_roc_falling_period
/sys/.../events/in_voltageY_thresh_rising_period
/sys/.../events/in_voltageY_thresh_falling_period

continues on next page

Table 4 – continued from previous page

/sys/.../events/in_voltageY_roc_rising_period
/sys/.../events/in_voltageY_roc_falling_period
/sys/.../events/in_tempY_thresh_rising_period
/sys/.../events/in_tempY_thresh_falling_period
/sys/.../events/in_tempY_roc_rising_period
/sys/.../events/in_tempY_roc_falling_period
/sys/.../events/in_accel_x&y&z_mag_falling_period
/sys/.../events/in_intensity0_thresh_period
/sys/.../events/in_proximity0_thresh_period
/sys/.../events/in_activity_still_thresh_rising_period
/sys/.../events/in_activity_still_thresh_falling_period
/sys/.../events/in_activity_walking_thresh_rising_period
/sys/.../events/in_activity_walking_thresh_falling_period
/sys/.../events/in_activity_jogging_thresh_rising_period
/sys/.../events/in_activity_jogging_thresh_falling_period
/sys/.../events/in_activity_running_thresh_rising_period
/sys/.../events/in_activity_running_thresh_falling_period
/sys/.../events/in_illuminance_thresh_either_period

Defined on file *sysfs-bus-iio*

Period of time (in seconds) for which the condition must be met before an event is generated. If direction is not specified then this period applies to both directions.

/sys/.../events/in_activity_still_thresh_rising_en
/sys/.../events/in_activity_still_thresh_falling_en
/sys/.../events/in_activity_walking_thresh_rising_en
/sys/.../events/in_activity_walking_thresh_falling_en
/sys/.../events/in_activity_jogging_thresh_rising_en
/sys/.../events/in_activity_jogging_thresh_falling_en
/sys/.../events/in_activity_running_thresh_rising_en
/sys/.../events/in_activity_running_thresh_falling_en

Defined on file *sysfs-bus-iio*

Enables or disables activity events. Depending on direction an event is generated when sensor ENTERS or LEAVES a given state.

/sys/.../events/in_activity_still_thresh_rising_value
/sys/.../events/in_activity_still_thresh_falling_value
/sys/.../events/in_activity_walking_thresh_rising_value
/sys/.../events/in_activity_walking_thresh_falling_value
/sys/.../events/in_activity_jogging_thresh_rising_value
/sys/.../events/in_activity_jogging_thresh_falling_value
/sys/.../events/in_activity_running_thresh_rising_value
/sys/.../events/in_activity_running_thresh_falling_value

Defined on file *sysfs-bus-iio*

Confidence value (in units as percentage) to be used for deciding when an event should be generated. E.g for running: If the confidence value reported by the sensor is greater than in_activity_running_thresh_rising_value then the sensor ENTERS running state. Conversely, if the confidence value reported by the sensor is lower than in_activity_running_thresh_falling_value then the sensor is LEAVING running state.

/sys/.../events/in_capacitanceY_adaptive_thresh_rising_en
/sys/.../events/in_capacitanceY_adaptive_thresh_falling_en

Defined on file *sysfs-bus-iio*

Adaptive thresholds are similar to normal fixed thresholds but the value is expressed as an offset from a value which provides a low frequency approximation of the channel itself. Thus these detect if a rapid change occurs in the specified direction which crosses tracking value + offset. Tracking value calculation is devices specific.

/sys/.../events/in_illuminance0_threshY_hysteresis

Defined on file *sysfs-bus-iio-light-lm3533-als*

Get the hysteresis for thresholds Y, that is, threshY_hysteresis = threshY_raising - threshY_falling

/sys/.../events/in_illuminance0_thresh_either_en

Defined on file *sysfs-bus-iio-light-lm3533-als*

Event generated when channel passes one of the four thresholds in each direction (rising|falling) and a zone change occurs. The corresponding light zone can be read from in_illuminance0_zone.

/sys/.../events/in_proximity_thresh_either_runningcount

Defined on file *sysfs-bus-iio*

Number of conditions that must occur, during a running period, before an event is generated.

/sys/.../events/in_proximity_thresh_either_runningperiod

Defined on file *sysfs-bus-iio*

A running period of time (in seconds) for which in_proximity_thresh_either_runningcount amount of conditions must occur before an event is generated. If direction is not specified then this period applies to both directions.

/sys/.../events/in_steps_change_en

Defined on file *sysfs-bus-iio*

Event generated when channel passes a threshold on the absolute change in value. E.g. for steps: a step change event is generated each time the user takes N steps, where N is set using in_steps_change_value.

/sys/.../events/in_steps_change_value

Defined on file *sysfs-bus-iio*

Specifies the value of change threshold that the device is comparing against for the events enabled by <type>[Y][_name]_roc[_rising|falling|]_en. E.g. for steps: if set to 3, a step change event will be generated every 3 steps.

/sys/.../iio:deviceX/bufferY/in_accel_type
/sys/.../iio:deviceX/bufferY/in_deltaangl_type
/sys/.../iio:deviceX/bufferY/in_deltavelocity_type
/sys/.../iio:deviceX/bufferY/in_anglvel_type
/sys/.../iio:deviceX/bufferY/in_magn_type
/sys/.../iio:deviceX/bufferY/in_incli_type
/sys/.../iio:deviceX/bufferY/in_voltageY_type
/sys/.../iio:deviceX/bufferY/in_voltage_type
/sys/.../iio:deviceX/bufferY/in_voltageY_supply_type
/sys/.../iio:deviceX/bufferY/in_voltageY_i_type
/sys/.../iio:deviceX/bufferY/in_voltageY_q_type
/sys/.../iio:deviceX/bufferY/in_voltage_i_type
/sys/.../iio:deviceX/bufferY/in_voltage_q_type
/sys/.../iio:deviceX/bufferY/in_timestamp_type
/sys/.../iio:deviceX/bufferY/in_pressureY_type
/sys/.../iio:deviceX/bufferY/in_pressure_type
/sys/.../iio:deviceX/bufferY/in_rot_quaternion_type
/sys/.../iio:deviceX/bufferY/in_proximity_type

Defined on file *sysfs-bus-iio*

Description of the scan element data storage within the buffer and hence the form in which it is read from user-space. Form is [be|le]:[s|u]bits/storagebits[>>shift]. be or le specifies big or little endian. s or u specifies if signed (2's complement) or unsigned. bits is the number of bits of data and storagebits is the space (after padding) that it occupies in the buffer. shift if specified, is the shift that needs to be applied prior to masking out unused bits. Some devices put their data in the middle of the transferred elements with additional information on both sides. Note that some devices will have additional information in the unused bits so to get a clean value, the bits value must be used to mask the buffer output value appropriately. The storagebits value also specifies the data alignment. So s48/64>>2 will be a signed 48 bit integer stored in a 64 bit location aligned to a 64 bit boundary. To obtain the clean value, shift right 2 and apply a mask to zero the top 16 bits of the result. For other storage combinations this attribute will be extended appropriately.

/sys/.../iio:deviceX/bufferY/in_accel_x_en
/sys/.../iio:deviceX/bufferY/in_accel_y_en
/sys/.../iio:deviceX/bufferY/in_accel_z_en
/sys/.../iio:deviceX/bufferY/in_deltaangl_x_en
/sys/.../iio:deviceX/bufferY/in_deltaangl_y_en
/sys/.../iio:deviceX/bufferY/in_deltaangl_z_en
/sys/.../iio:deviceX/bufferY/in_deltavelocity_x_en

continues on next page

Table 5 – continued from previous page

/sys/.../iio:deviceX/bufferY/in_deltavelocity_y_en
/sys/.../iio:deviceX/bufferY/in_deltavelocity_z_en
/sys/.../iio:deviceX/bufferY/in_anglvel_x_en
/sys/.../iio:deviceX/bufferY/in_anglvel_y_en
/sys/.../iio:deviceX/bufferY/in_anglvel_z_en
/sys/.../iio:deviceX/bufferY/in_magn_x_en
/sys/.../iio:deviceX/bufferY/in_magn_y_en
/sys/.../iio:deviceX/bufferY/in_magn_z_en
/sys/.../iio:deviceX/bufferY/in_rot_from_north_magnetic_en
/sys/.../iio:deviceX/bufferY/in_rot_from_north_true_en
/sys/.../iio:deviceX/bufferY/in_rot_from_north_magnetic_tilt_comp_en
/sys/.../iio:deviceX/bufferY/in_rot_from_north_true_tilt_comp_en
/sys/.../iio:deviceX/bufferY/in_timestamp_en
/sys/.../iio:deviceX/bufferY/in_voltageY_supply_en
/sys/.../iio:deviceX/bufferY/in_voltageY_en
/sys/.../iio:deviceX/bufferY/in_voltageY-voltageZ_en
/sys/.../iio:deviceX/bufferY/in_voltageY_i_en
/sys/.../iio:deviceX/bufferY/in_voltageY_q_en
/sys/.../iio:deviceX/bufferY/in_voltage_i_en
/sys/.../iio:deviceX/bufferY/in_voltage_q_en
/sys/.../iio:deviceX/bufferY/in_incli_x_en
/sys/.../iio:deviceX/bufferY/in_incli_y_en
/sys/.../iio:deviceX/bufferY/in_pressureY_en
/sys/.../iio:deviceX/bufferY/in_pressure_en
/sys/.../iio:deviceX/bufferY/in_rot_quaternion_en
/sys/.../iio:deviceX/bufferY/in_proximity_en

Defined on file *sysfs-bus-iio*

Scan element control for triggered data capture.

/sys/.../iio:deviceX/bufferY/in_voltageY_index
/sys/.../iio:deviceX/bufferY/in_voltageY_supply_index
/sys/.../iio:deviceX/bufferY/in_voltageY_i_index
/sys/.../iio:deviceX/bufferY/in_voltageY_q_index
/sys/.../iio:deviceX/bufferY/in_voltage_i_index
/sys/.../iio:deviceX/bufferY/in_voltage_q_index
/sys/.../iio:deviceX/bufferY/in_accel_x_index
/sys/.../iio:deviceX/bufferY/in_accel_y_index
/sys/.../iio:deviceX/bufferY/in_accel_z_index
/sys/.../iio:deviceX/bufferY/in_deltaangl_x_index
/sys/.../iio:deviceX/bufferY/in_deltaangl_y_index
/sys/.../iio:deviceX/bufferY/in_deltaangl_z_index
/sys/.../iio:deviceX/bufferY/in_deltavelocity_x_index
/sys/.../iio:deviceX/bufferY/in_deltavelocity_y_index
/sys/.../iio:deviceX/bufferY/in_deltavelocity_z_index
/sys/.../iio:deviceX/bufferY/in_anglvel_x_index
/sys/.../iio:deviceX/bufferY/in_anglvel_y_index

continues on next page

Table 6 – continued from previous page

/sys/.../iio:deviceX/bufferY/in_anglvel_z_index
/sys/.../iio:deviceX/bufferY/in_magn_x_index
/sys/.../iio:deviceX/bufferY/in_magn_y_index
/sys/.../iio:deviceX/bufferY/in_magn_z_index
/sys/.../iio:deviceX/bufferY/in_rot_from_north_magnetic_index
/sys/.../iio:deviceX/bufferY/in_rot_from_north_true_index
/sys/.../iio:deviceX/bufferY/in_rot_from_north_magnetic_tilt_comp_index
/sys/.../iio:deviceX/bufferY/in_rot_from_north_true_tilt_comp_index
/sys/.../iio:deviceX/bufferY/in_incli_x_index
/sys/.../iio:deviceX/bufferY/in_incli_y_index
/sys/.../iio:deviceX/bufferY/in_timestamp_index
/sys/.../iio:deviceX/bufferY/in_pressureY_index
/sys/.../iio:deviceX/bufferY/in_pressure_index
/sys/.../iio:deviceX/bufferY/in_rot_quaternion_index
/sys/.../iio:deviceX/bufferY/in_proximity_index

Defined on file *sysfs-bus-iio*

A single positive integer specifying the position of this scan element in the buffer. Note these are not dependent on what is enabled and may not be contiguous. Thus for user-space to establish the full layout these must be used in conjunction with all _en attributes to establish which channels are present, and the relevant _type attributes to establish the data storage format.

/sys/.../iio:deviceX/events/in_accel_mag_en
/sys/.../iio:deviceX/events/in_accel_mag_rising_en
/sys/.../iio:deviceX/events/in_accel_mag_falling_en
/sys/.../iio:deviceX/events/in_accel_x_mag_en
/sys/.../iio:deviceX/events/in_accel_x_mag_rising_en
/sys/.../iio:deviceX/events/in_accel_x_mag_falling_en
/sys/.../iio:deviceX/events/in_accel_y_mag_en
/sys/.../iio:deviceX/events/in_accel_y_mag_rising_en
/sys/.../iio:deviceX/events/in_accel_y_mag_falling_en
/sys/.../iio:deviceX/events/in_accel_z_mag_en
/sys/.../iio:deviceX/events/in_accel_z_mag_rising_en
/sys/.../iio:deviceX/events/in_accel_z_mag_falling_en
/sys/.../iio:deviceX/events/in_accel_x&y&z_mag_rising_en
/sys/.../iio:deviceX/events/in_accel_x&y&z_mag_falling_en

Defined on file *sysfs-bus-iio*

Similar to in_accel_x_thresh[_rising|_falling]_en, but here the magnitude of the channel is compared to the threshold, not its signed value.

/sys/.../iio:deviceX/events/in_accel_mag_referenced_en
/sys/.../iio:deviceX/events/in_accel_mag_referenced_rising_en
/sys/.../iio:deviceX/events/in_accel_mag_referenced_falling_en
/sys/.../iio:deviceX/events/in_accel_y_mag_referenced_en
/sys/.../iio:deviceX/events/in_accel_y_mag_referenced_rising_en
/sys/.../iio:deviceX/events/in_accel_y_mag_referenced_falling_en

Defined on file *sysfs-bus-iio*

Similar to in_accel_mag[_y][_rising|_falling]_en, but the event value is relative to a reference magnitude. The reference magnitude includes the graviational acceleration.

/sys/.../iio:deviceX/events/in_accel_mag_referenced_value
/sys/.../iio:deviceX/events/in_accel_mag_referenced_rising_value
/sys/.../iio:deviceX/events/in_accel_mag_referenced_falling_value
/sys/.../iio:deviceX/events/in_accel_y_mag_referenced_value
/sys/.../iio:deviceX/events/in_accel_y_mag_referenced_rising_value
/sys/.../iio:deviceX/events/in_accel_y_mag_referenced_falling_value

Defined on file *sysfs-bus-iio*

The value to which the reference magnitude of the channel is compared. If the axis is not specified, it applies to all channels of this type.

/sys/.../iio:deviceX/events/in_accel_x_roc_rising_en
/sys/.../iio:deviceX/events/in_accel_x_roc_falling_en
/sys/.../iio:deviceX/events/in_accel_y_roc_rising_en
/sys/.../iio:deviceX/events/in_accel_y_roc_falling_en
/sys/.../iio:deviceX/events/in_accel_z_roc_rising_en
/sys/.../iio:deviceX/events/in_accel_z_roc_falling_en
/sys/.../iio:deviceX/events/in_anglvel_x_roc_rising_en
/sys/.../iio:deviceX/events/in_anglvel_x_roc_falling_en
/sys/.../iio:deviceX/events/in_anglvel_y_roc_rising_en
/sys/.../iio:deviceX/events/in_anglvel_y_roc_falling_en
/sys/.../iio:deviceX/events/in_anglvel_z_roc_rising_en
/sys/.../iio:deviceX/events/in_anglvel_z_roc_falling_en
/sys/.../iio:deviceX/events/in_magn_x_roc_rising_en
/sys/.../iio:deviceX/events/in_magn_x_roc_falling_en
/sys/.../iio:deviceX/events/in_magn_y_roc_rising_en
/sys/.../iio:deviceX/events/in_magn_y_roc_falling_en
/sys/.../iio:deviceX/events/in_magn_z_roc_rising_en
/sys/.../iio:deviceX/events/in_magn_z_roc_falling_en
/sys/.../iio:deviceX/events/in_rot_from_north_magnetic_roc_rising_en
/sys/.../iio:deviceX/events/in_rot_from_north_magnetic_roc_falling_en
/sys/.../iio:deviceX/events/in_rot_from_north_true_roc_rising_en
/sys/.../iio:deviceX/events/in_rot_from_north_true_roc_falling_en
/sys/.../iio:deviceX/events/in_rot_from_north_magnetic_tilt_comp_roc_rising_en
/sys/.../iio:deviceX/events/in_rot_from_north_magnetic_tilt_comp_roc_falling_en
/sys/.../iio:deviceX/events/in_rot_from_north_true_tilt_comp_roc_rising_en
/sys/.../iio:deviceX/events/in_rot_from_north_true_tilt_comp_roc_falling_en
/sys/.../iio:deviceX/events/in_voltageY_supply_roc_rising_en
/sys/.../iio:deviceX/events/in_voltageY_supply_roc_falling_en
/sys/.../iio:deviceX/events/in_voltageY_roc_rising_en
/sys/.../iio:deviceX/events/in_voltageY_roc_falling_en
/sys/.../iio:deviceX/events/in_tempY_roc_rising_en
/sys/.../iio:deviceX/events/in_tempY_roc_falling_en

Defined on file *sysfs-bus-iio*

Event generated when channel passes a threshold on the rate of change (1st differential) in the specified (_rising|_falling) direction. If the direction is not specified, then either the device will report an event which ever direction a single threshold value is passed in (e.g. <type>[Y][_name]_<raw|input>_roc_value) or <type>[Y][_name]_<raw|input>_roc_rising_value and <type>[Y][_name]_<raw|input>_roc_falling_va may take different values, but the device can only enable both rate of change thresholds or neither.

Note the driver will assume the last p events requested are to be enabled where p is however many it supports (which may vary depending on the exact set requested. So if you want to be sure you have set what you think you have, check the contents of these attributes after everything is configured. Drivers may have to buffer any parameters so that they are consistent when a given event type is enabled a future point (and not those for whatever event was previously enabled).

/sys/.../iio:deviceX/events/in_accel_x_thresh_rising_en
/sys/.../iio:deviceX/events/in_accel_x_thresh_falling_en
/sys/.../iio:deviceX/events/in_accel_y_thresh_rising_en
/sys/.../iio:deviceX/events/in_accel_y_thresh_falling_en
/sys/.../iio:deviceX/events/in_accel_z_thresh_rising_en
/sys/.../iio:deviceX/events/in_accel_z_thresh_falling_en
/sys/.../iio:deviceX/events/in_anglvel_x_thresh_rising_en
/sys/.../iio:deviceX/events/in_anglvel_x_thresh_falling_en
/sys/.../iio:deviceX/events/in_anglvel_y_thresh_rising_en
/sys/.../iio:deviceX/events/in_anglvel_y_thresh_falling_en
/sys/.../iio:deviceX/events/in_anglvel_z_thresh_rising_en
/sys/.../iio:deviceX/events/in_anglvel_z_thresh_falling_en
/sys/.../iio:deviceX/events/in_magn_x_thresh_rising_en
/sys/.../iio:deviceX/events/in_magn_x_thresh_falling_en
/sys/.../iio:deviceX/events/in_magn_y_thresh_rising_en
/sys/.../iio:deviceX/events/in_magn_y_thresh_falling_en
/sys/.../iio:deviceX/events/in_magn_z_thresh_rising_en
/sys/.../iio:deviceX/events/in_magn_z_thresh_falling_en
/sys/.../iio:deviceX/events/in_rot_from_north_magnetic_thresh_rising_en
/sys/.../iio:deviceX/events/in_rot_from_north_magnetic_thresh_falling_en
/sys/.../iio:deviceX/events/in_rot_from_north_true_thresh_rising_en
/sys/.../iio:deviceX/events/in_rot_from_north_true_thresh_falling_en
/sys/.../iio:deviceX/events/in_rot_from_north_magnetic_tilt_comp_thresh_rising_en
/sys/.../iio:deviceX/events/in_rot_from_north_magnetic_tilt_comp_thresh_falling_en
/sys/.../iio:deviceX/events/in_rot_from_north_true_tilt_comp_thresh_rising_en
/sys/.../iio:deviceX/events/in_rot_from_north_true_tilt_comp_thresh_falling_en
/sys/.../iio:deviceX/events/in_voltageY_supply_thresh_rising_en
/sys/.../iio:deviceX/events/in_voltageY_supply_thresh_falling_en
/sys/.../iio:deviceX/events/in_voltageY_thresh_rising_en
/sys/.../iio:deviceX/events/in_voltageY_thresh_falling_en
/sys/.../iio:deviceX/events/in_voltageY_thresh_either_en

continues on next page

Table 8 – continued from previous page

/sys/.../iio:deviceX/events/in_tempY_thresh_rising_en
/sys/.../iio:deviceX/events/in_tempY_thresh_falling_en
/sys/.../iio:deviceX/events/in_capacitanceY_thresh_rising_en
/sys/.../iio:deviceX/events/in_capacitanceY_thresh_falling_en

Defined on file *sysfs-bus-iio*

Event generated when channel passes a threshold in the specified (_rising|_falling) direction. If the direction is not specified, then either the device will report an event which ever direction a single threshold value is passed in (e.g. <type>[Y][_name]_<raw|input>_thresh_value) or <type>[Y][_name]_<raw|input>_thresh_rising_value and <type>[Y][_name]_<raw|input>_thresh_fal may take different values, but the device can only enable both thresholds or neither.

Note the driver will assume the last p events requested are to be enabled where p is how many it supports (which may vary depending on the exact set requested. So if you want to be sure you have set what you think you have, check the contents of these attributes after everything is configured. Drivers may have to buffer any parameters so that they are consistent when a given event type is enabled at a future point (and not those for whatever event was previously enabled).

/sys/.../iio:deviceX/in_accelX_power_mode

Defined on file *sysfs-bus-iio*

Specifies the chip power mode. low_noise: reduce noise level from ADC, low_power: enable low current consumption. For a list of available output power modes read in_accel_power_mode_available.

/sys/.../iio:deviceX/in_activity_still_input
/sys/.../iio:deviceX/in_activity_walking_input
/sys/.../iio:deviceX/in_activity_jogging_input
/sys/.../iio:deviceX/in_activity_running_input

Defined on file *sysfs-bus-iio*

This attribute is used to read the confidence for an activity expressed in units as percentage.

/sys/.../iio:deviceX/in_anglvel_z_quadrature_correction_raw

Defined on file *sysfs-bus-iio*

This attribute is used to read the amount of quadrature error present in the device at a given time.

/sys/.../iio:deviceX/in_capacitableY_calibbias_calibration
/sys/.../iio:deviceX/in_capacitableY_calibscale_calibration

Defined on file *sysfs-bus-iio-cdc-ad7746*

Write 1 to trigger a calibration of the calibbias or calibscale. For calibscale, a full scale capacitance should be connected to the capacitance input and a calibscale_calibration then started. For calibbias see the device datasheet section on "capacitive system offset calibration".

/sys/.../iio:deviceX/in_capacitanceY-capacitanceZ_raw

Defined on file *sysfs-bus-iio*

Raw differential capacitance measurement equivalent to channel Y - channel Z where these channel numbers apply to the physically equivalent inputs when non differential readings are separately available. In differential only parts, then all that is required is a consistent labeling. Units after application of scale and offset are nanofarads.

/sys/.../iio:deviceX/in_capacitanceY-capacitanceZ_zeropoint

Defined on file *sysfs-bus-iio*

For differential channels, this an offset that is applied equally to both inputs. As the reading is of the difference between the two inputs, this should not be applied to the _raw reading by userspace (unlike _offset) and unlike calibbias it does not affect the differential value measured because the effect of _zeropoint cancels out across the two inputs that make up the differential pair. It's purpose is to bring the individual signals, before the differential is measured, within the measurement range of the device. The naming is chosen because if the separate inputs that make the differential pair are drawn on a graph in their _raw units, this is the value that the zero point on the measurement axis represents. It is expressed with the same scaling as _raw.

/sys/.../iio:deviceX/in_distance_input
/sys/.../iio:deviceX/in_distance_raw

Defined on file *sysfs-bus-iio*

This attribute is used to read the measured distance to an object or the distance covered by the user since the last reboot while activated. Units after application of scale are meters.

/sys/.../iio:deviceX/in_energy_en
/sys/.../iio:deviceX/in_distance_en
/sys/.../iio:deviceX/in_velocity_sqrt(x^2+y^2+z^2)_en
/sys/.../iio:deviceX/in_steps_en

Defined on file *sysfs-bus-iio*

Activates a device feature that runs in firmware/hardware. E.g. for steps: the pedometer saves power while not used; when activated, it will count the steps taken by the user in firmware and export them through in_steps_input.

/sys/.../iio:deviceX/in_energy_input
/sys/.../iio:deviceX/in_energy_raw

Defined on file *sysfs-bus-iio*

This attribute is used to read the energy value reported by the device (e.g.: human activity sensors report energy burnt by the user). Units after application of scale are Joules.

/sys/.../iio:deviceX/in_illuminance_input
/sys/.../iio:deviceX/in_illuminance_raw
/sys/.../iio:deviceX/in_illuminanceY_input
/sys/.../iio:deviceX/in_illuminanceY_raw
/sys/.../iio:deviceX/in_illuminanceY_mean_raw
/sys/.../iio:deviceX/in_illuminance_ir_raw
/sys/.../iio:deviceX/in_illuminance_clear_raw

Defined on file *sysfs-bus-iio*

Illuminance measurement, units after application of scale and offset are lux.

/sys/.../iio:deviceX/in_intensityY_raw
/sys/.../iio:deviceX/in_intensityY_ir_raw
/sys/.../iio:deviceX/in_intensityY_both_raw
/sys/.../iio:deviceX/in_intensityY_uv_raw
/sys/.../iio:deviceX/in_intensityY_uva_raw
/sys/.../iio:deviceX/in_intensityY_uvb_raw
/sys/.../iio:deviceX/in_intensityY_duv_raw

Defined on file *sysfs-bus-iio*

Unit-less light intensity. Modifiers both and ir indicate that measurements contain visible and infrared light components or just infrared light, respectively. Modifier uv indicates that measurements contain ultraviolet light components. Modifiers uva, uvb and duv indicate that measurements contain A, B or deep (C) ultraviolet light components respectively.

/sys/.../iio:deviceX/in_intensity_integration_time
/sys/.../iio:deviceX/in_intensity_red_integration_time
/sys/.../iio:deviceX/in_intensity_green_integration_time
/sys/.../iio:deviceX/in_intensity_blue_integration_time
/sys/.../iio:deviceX/in_intensity_clear_integration_time
/sys/.../iio:deviceX/in_illuminance_integration_time

Defined on file *sysfs-bus-iio*

This attribute is used to get/set the integration time in seconds. If shared across all channels of a given type, <type>_integration_time is used.

/sys/.../iio:deviceX/in_proximity_raw
/sys/.../iio:deviceX/in_proximity_input
/sys/.../iio:deviceX/in_proximityY_raw

Defined on file *sysfs-bus-iio*

Proximity measurement indicating that some object is near the sensor, usually by observing reflectivity of infrared or ultrasound emitted.

Often these sensors are unit less and as such conversion to SI units is not possible. Higher proximity measurements indicate closer objects, and vice versa. Units after application of scale and offset are meters.

/sys/.../iio:deviceX/in_steps_debounce_count

Defined on file *sysfs-bus-iio*

Specifies the number of steps that must occur within in_steps_filter_debounce_time for the pedometer to decide the consumer is making steps.

/sys/.../iio:deviceX/in_steps_debounce_time

Defined on file *sysfs-bus-iio*

Specifies number of seconds in which we compute the steps that occur in order to decide if the consumer is making steps.

/sys/.../iio:deviceX/in_steps_input

Defined on file *sysfs-bus-iio*

This attribute is used to read the number of steps taken by the user since the last reboot while activated.

/sys/.../iio:deviceX/in_uvindex_input

Defined on file *sysfs-bus-iio*

UV light intensity index measuring the human skin's response to different wavelength of sunlight weighted according to the standardised CIE Erythemal Action Spectrum. UV index values range from 0 (low) to >=11 (extreme).

/sys/.../iio:deviceX/in_velocity_sqrt(x^2+y^2+z^2)_input
/sys/.../iio:deviceX/in_velocity_sqrt(x^2+y^2+z^2)_raw

Defined on file *sysfs-bus-iio*

This attribute is used to read the current speed value of the user (which is the norm or magnitude of the velocity vector). Units after application of scale are m/s.

/sys/.../iio:deviceX/in_velocity_sqrt(x^2+y^2+z^2)_integration_time

Defined on file *sysfs-bus-iio*

Number of seconds in which to compute speed.

/sys/.../iio:deviceX/out_voltageY_powerdown_mode_available
/sys/.../iio:deviceX/out_voltage_powerdown_mode_available
/sys/.../iio:deviceX/out_altvoltageY_powerdown_mode_available
/sys/.../iio:deviceX/out_altvoltage_powerdown_mode_available

Defined on file *sysfs-bus-iio*

Lists all available output power down modes. If Y is not present the mode is shared across all outputs.

/sys/.../iio:deviceX/scan_elements/in_accel_type_available

Defined on file *sysfs-bus-iio*

If the type parameter can take one of a small set of values, this attribute lists them.

/sys/.../in_accel_filter_high_pass_3db_frequency
/sys/.../in_anglvel_filter_high_pass_3db_frequency
/sys/.../in_magn_filter_high_pass_3db_frequency

Defined on file *sysfs-bus-iio*

If a known or controllable high pass filter is applied to the underlying data channel, then this parameter gives the 3dB frequency of the filter in Hz.

/sys/.../in_accel_filter_low_pass_3db_frequency
/sys/.../in_magn_filter_low_pass_3db_frequency
/sys/.../in_anglvel_filter_low_pass_3db_frequency

Defined on file *sysfs-bus-iio*

If a known or controllable low pass filter is applied to the underlying data channel, then this parameter gives the 3dB frequency of the filter in Hz.

/sys/.../in_capacitanceY_adaptive_thresh_rising_timeout
/sys/.../in_capacitanceY_adaptive_thresh_falling_timeout

Defined on file *sysfs-bus-iio*

When adaptive thresholds are used, the tracking signal may adjust too slowly to step changes in the raw signal. Thus these specify the time in seconds for which the difference between the slow tracking signal and the raw signal is allowed to remain out-of-range before a reset event occurs in which the tracking signal is made equal to the raw signal, allowing slow tracking to resume and the adaptive threshold event detection to function as expected.

/sys/.../mdev_supported_types/<type-id>/available_instances

Defined on file *sysfs-bus-vfio-mdev*

Reading this attribute will show the number of mediated devices of type <type-id> that can be created. This is a readonly file.

Users:

Userspace applications interested in creating mediated device of that type. Userspace application should check the number of available instances could be created before creating mediated device of this type.

/sys/.../mdev_supported_types/<type-id>/create

Defined on file *sysfs-bus-vfio-mdev*

Writing UUID to this file will create mediated device of type <type-id> for parent device <device>. This is a write-only file. For example:

```
# echo "83b8f4f2-509f-382f-3c1e-e6bfe0fa1001" >       \
     /sys/devices/foo/mdev_supported_types/foo-1/create
```

/sys/.../mdev_supported_types/<type-id>/description

Defined on file *sysfs-bus-vfio-mdev*

Reading this attribute will show description of the type of mediated device that will get created of type <type-id>. This is optional attribute. For example: "2 heads, 512M FB, 2560x1600 maximum resolution"

Users:

Userspace applications interested in knowing the details of a particular <type-id> that can help in understanding the features provided by that type of mediated device.

/sys/.../mdev_supported_types/<type-id>/device_api

Defined on file *sysfs-bus-vfio-mdev*

Reading this attribute will show VFIO device API supported by this type. For example, "vfio-pci" for a PCI device, "vfio-platform" for platform device.

/sys/.../mdev_supported_types/<type-id>/devices/

Defined on file *sysfs-bus-vfio-mdev*

This directory contains symbolic links pointing to mdev devices sysfs entries which are created of this <type-id>.

/sys/.../mdev_supported_types/<type-id>/name

Defined on file *sysfs-bus-vfio-mdev*

Reading this attribute will show human readable name of the mediated device that will get created of type <type-id>. This is optional attribute. For example: "Grid M60-0Q"

Users:

Userspace applications interested in knowing the name of a particular <type-id> that can help in understanding the type of mediated device.

/sys/.../uevent

Defined on file *sysfs-uevent*

Enable passing additional variables for synthetic uevents that are generated by writing /sys/.../uevent file.

Recognized extended format is:

```
ACTION [UUID [KEY=VALUE ...]
```

The ACTION is compulsory - it is the name of the uevent action (`add`, `change`, `remove`). There is no change compared to previous functionality here. The rest of the extended format is optional.

You need to pass UUID first before any KEY=VALUE pairs. The UUID must be in `xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx` format where 'x' is a hex digit. The UUID is considered to be a transaction identifier so it's possible to use the same UUID value for one or more synthetic uevents in which case we logically group these uevents together for any userspace listeners. The UUID value appears in uevent as `SYNTH_UUID=xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx` environment variable.

If UUID is not passed in, the generated synthetic uevent gains `SYNTH_UUID=0` environment variable automatically.

The KEY=VALUE pairs can contain alphanumeric characters only.

It's possible to define zero or more pairs - each pair is then delimited by a space character ' '. Each pair appears in synthetic uevent as `SYNTH_ARG_KEY=VALUE`. That means the KEY name gains `SYNTH_ARG_` prefix to avoid possible collisions with existing variables.

Example of valid sequence written to the uevent file:

```
add fe4d7c9d-b8c6-4a70-9ef1-3d8a58d18eed A=1 B=abc
```

This generates synthetic uevent including these variables:

```
ACTION=add
SYNTH_ARG_A=1
SYNTH_ARG_B=abc
SYNTH_UUID=fe4d7c9d-b8c6-4a70-9ef1-3d8a58d18eed
```

Users:

udev, userspace tools generating synthetic uevents

5.2.30 Symbols under /sys/block

/sys/block/*/device/cdl_enable

Defined on file *sysfs-block-device*

(RW) For a device supporting the command duration limits feature, write to the file to turn on or off the feature. By default this feature is turned off. Writing ”1” to this file enables the use of command duration limits for read and write commands in the kernel and turns on the feature on the device. Writing ”0” disables the feature.

/sys/block/*/device/cdl_supported

Defined on file *sysfs-block-device*

(RO) Indicates if the device supports the command duration limits feature found in some ATA and SCSI devices.

/sys/block/*/device/ncq_prio_enable

Defined on file *sysfs-block-device*

(RW) Write to the file to turn on or off the SATA NCQ (native command queueing) priority support. By default this feature is turned off. If the device does not support the SATA NCQ priority feature, writing "1" to this file results in an error (see ncq_prio_supported).

/sys/block/*/device/ncq_prio_supported

Defined on file *sysfs-block-device*

(RO) Indicates if the device supports the SATA NCQ (native command queueing) priority feature.

/sys/block/*/device/sas_ncq_prio_enable

Defined on file *sysfs-block-device*

(RW) This is the equivalent of the ncq_prio_enable attribute file for SATA devices connected to a SAS host-bus-adapter (HBA) implementing support for the SATA NCQ priority feature. This file does not exist if the HBA driver does not implement support for the SATA NCQ priority feature, regardless of the device support for this feature (see sas_ncq_prio_supported).

/sys/block/*/device/sas_ncq_prio_supported

Defined on file *sysfs-block-device*

(RO) This is the equivalent of the ncq_prio_supported attribute file for SATA devices connected to a SAS host-bus-adapter (HBA) implementing support for the SATA NCQ priority feature. This file does not exist if the HBA driver does not implement support for the SATA NCQ priority feature, regardless of the device support for this feature.

/sys/block/*/device/sw_activity

Defined on file *sysfs-block-device*

(RW) Used by drivers which support software controlled activity LEDs.

It has the following valid values:

0	OFF - the LED is not activated on activity
1	BLINK_ON - the LED blinks on every 10ms when activity is detected.
2	BLINK_OFF - the LED is on when idle, and blinks off every 10ms when activity is detected.

Note that the user must turn sw_activity OFF it they wish to control the activity LED via the em_message file.

/sys/block/*/device/unload_heads

Defined on file *sysfs-block-device*

(RW) Hard disk shock protection

Writing an integer value to this file will take the heads of the respective drive off the platter and block all I/O operations for the specified number of milliseconds.

- If the device does not support the unload heads feature, access is denied with -EOPNOTSUPP.
- The maximal value accepted for a timeout is 30000 milliseconds.
- A previously set timeout can be cancelled and disk can resume normal operation immediately by specifying a timeout of 0.
- Some hard drives only comply with an earlier version of the ATA standard, but support the unload feature nonetheless. There is no safe way Linux can detect these devices, so this is not enabled by default. If it is known that your device does support the unload feature, then you can tell the kernel to enable it by writing -1. It can be disabled again by writing -2.
- Values below -2 are rejected with -EINVAL

For more information, see *Hard disk shock protection*

/sys/block/<device>/iosched/target_latency

Defined on file *sysfs-cfq-target-latency*

The */sys/block/<device>/iosched/target_latency* only exists when the user sets cfq to /sys/block/<device>/scheduler. It contains an estimated latency time for the cfq. cfq will use it to calculate the time slice used for every task.

/sys/block/<disk>/bcache/btree_cache_size

Defined on file *sysfs-block-bcache*

Number of btree buckets/nodes that are currently cached in memory; cache dynamically grows and shrinks in response to memory pressure from the rest of the system.

/sys/block/<disk>/bcache/btree_written

Defined on file *sysfs-block-bcache*

For a cache, sum of all btree writes in human readable units.

/sys/block/<disk>/bcache/bucket_size

Defined on file *sysfs-block-bcache*

For a cache, bucket size in human readable units, as set at cache creation time; should match the erase block size of the SSD for optimal performance.

/sys/block/<disk>/bcache/bypassed

Defined on file *sysfs-block-bcache*

Sum of all reads and writes that have bypassed the cache (due to the sequential cutoff). Expressed as bytes in human readable units.

/sys/block/<disk>/bcache/cache

Defined on file *sysfs-block-bcache*

For a backing device that has cache, a symlink to the bcache/ dir of that cache.

/sys/block/<disk>/bcache/cache_hit_ratio

Defined on file *sysfs-block-bcache*

For backing devices: cache hits as a percentage.

/sys/block/<disk>/bcache/cache_hits

Defined on file *sysfs-block-bcache*

For backing devices: integer number of full cache hits, counted per bio. A partial cache hit counts as a miss.

/sys/block/<disk>/bcache/cache_misses

Defined on file *sysfs-block-bcache*

For backing devices: integer number of cache misses.

/sys/block/<disk>/bcache/clear_stats

Defined on file *sysfs-block-bcache*

Writing to this file resets all the statistics for the device.

/sys/block/<disk>/bcache/discard

Defined on file *sysfs-block-bcache*

For a cache, a boolean allowing discard/TRIM to be turned off or back on if the device supports it.

/sys/block/<disk>/bcache/nbuckets

Defined on file *sysfs-block-bcache*

For a cache, the number of usable buckets.

/sys/block/<disk>/bcache/sequential_cutoff

Defined on file *sysfs-block-bcache*

For backing devices: Threshold past which sequential IO will skip the cache. Read and written as bytes in human readable units (i.e. echo 10M > sequntial_cutoff).

/sys/block/<disk>/bcache/synchronous

Defined on file *sysfs-block-bcache*

For a cache, a boolean that allows synchronous mode to be switched on and off. In synchronous mode all writes are ordered such that the cache can reliably recover from unclean shutdown; if disabled bcache will not generally wait for writes to complete but if the cache is not shut down cleanly all data will be discarded from the cache. Should not be turned off with writeback caching enabled.

/sys/block/<disk>/bcache/tree_depth

Defined on file *sysfs-block-bcache*

For a cache, height of the btree excluding leaf nodes (i.e. a one node tree will have a depth of 0).

/sys/block/<disk>/bcache/unregister

Defined on file *sysfs-block-bcache*

A write to this file causes the backing device or cache to be unregistered. If a backing device had dirty data in the cache, writeback mode is automatically disabled and all dirty data is flushed before the device is unregistered. Caches unregister all associated backing devices before unregistering themselves.

/sys/block/<disk>/bcache/writeback

Defined on file *sysfs-block-bcache*

For backing devices: When on, writeback caching is enabled and writes will be buffered in the cache. When off, caching is in writethrough mode; reads and writes will be added to the cache but no write buffering will take place.

/sys/block/<disk>/bcache/writeback_delay

Defined on file *sysfs-block-bcache*

For backing devices: In writeback mode, when dirty data is written to the cache and the cache held no dirty data for that backing device, writeback from cache to backing device starts after this delay, expressed as an integer number of seconds.

/sys/block/<disk>/bcache/writeback_percent

Defined on file *sysfs-block-bcache*

For backing devices: If nonzero, writeback from cache to backing device only takes place when more than this percentage of the cache is used, allowing more write coalescing to take place and reducing total number of writes sent to the backing device. Integer between 0 and 40.

/sys/block/<disk>/bcache/writeback_running

Defined on file *sysfs-block-bcache*

For backing devices: when off, dirty data will not be written from the cache to the backing device. The cache will still be used to buffer writes until it is mostly full, at which point writes transparently revert to writethrough mode. Intended only for benchmarking/testing.

/sys/block/<disk>/bcache/written

Defined on file *sysfs-block-bcache*

For a cache, total amount of data in human readable units written to the cache, excluding all metadata.

/sys/block/dm-<num>/dm/name

Defined on file *sysfs-block-dm*

Device-mapper device name. Read-only string containing mapped device name.

Users: util-linux, device-mapper udev rules

/sys/block/dm-<num>/dm/rq_based_seq_io_merge_deadline

Defined on file *sysfs-block-dm*

Allow control over how long a request that is a reasonable merge candidate can be queued on the request queue. The resolution of this deadline is in microseconds (ranging from 1 to 100000 usecs). Setting this attribute to 0 (the default) will disable request-based DM's merge heuristic and associated extra accounting. This attribute is not applicable to bio-based DM devices so it will only ever report 0 for them.

/sys/block/dm-<num>/dm/suspended

Defined on file *sysfs-block-dm*

Device-mapper device suspend state. Contains the value 1 while the device is suspended. Otherwise it contains 0. Read-only attribute.

Users: util-linux, device-mapper udev rules

/sys/block/dm-<num>/dm/use_blk_mq

Defined on file *sysfs-block-dm*

Request-based Device-mapper blk-mq I/O path mode. Contains the value 1 if the device is using blk-mq. Otherwise it contains 0. Read-only attribute.

/sys/block/dm-<num>/dm/uuid

Defined on file *sysfs-block-dm*

Device-mapper device UUID. Read-only string containing DM-UUID or empty string if DM-UUID is not set.

Users: util-linux, device-mapper udev rules

/sys/block/etherd*/firmware-version

Defined on file *sysfs-block-aoe*

(RO) Version of the firmware in the target.

/sys/block/etherd*/mac

Defined on file *sysfs-block-aoe*

(RO) The ethernet address of the remote Ata over Ethernet (AoE) device.

/sys/block/etherd*/netif

Defined on file *sysfs-block-aoe*

(RO) The names of the network interfaces on the localhost (comma separated) through which we are communicating with the remote AoE device.

/sys/block/etherd*/payload

Defined on file *sysfs-block-aoe*

(RO) The amount of user data transferred (in bytes) inside each AoE command on the network, network headers excluded.

/sys/block/etherd*/state

Defined on file *sysfs-block-aoe*

(RO) Device status. The state attribute is ”up” when the device is ready for I/O and ”down” if detected but unusable. The ”down,closewait” state shows that the device is still open and cannot come up again until it has been closed. The ”up,kickme” state means that the driver wants to send more commands to the target but found out there were already the max number of commands waiting for a response. It will retry again after being kicked by the periodic timer handler routine.

/sys/block/loopX/loop/autoclear

Defined on file *sysfs-block-loop*

(RO) Shows if the device is in autoclear mode or not (”1” or ”0”). Autoclear (if set) indicates that the loopback device will self-distruct after last close.

/sys/block/loopX/loop/backing_file

Defined on file *sysfs-block-loop*

(RO) The path of the backing file that the loop device maps its data blocks to.

/sys/block/loopX/loop/dio

Defined on file *sysfs-block-loop*

(RO) Shows if direct IO is being used to access backing file or not (”1 or ”0”).

/sys/block/loopX/loop/offset

Defined on file *sysfs-block-loop*

(RO) Start offset (in bytes).

/sys/block/loopX/loop/partscan

Defined on file *sysfs-block-loop*

(RO) Shows if automatic partition scanning is enabled for the device or not (”1” or ”0”). This can be requested individually per loop device during its setup by setting LO_FLAGS_PARTSCAN in in the ioctl request. By default, no partition tables are scanned.

/sys/block/loopX/loop/sizelimit

Defined on file *sysfs-block-loop*

(RO) The size (in bytes) that the block device maps, starting from the offset.

/sys/block/rnbd<N>/rnbd/access_mode

Defined on file *sysfs-block-rnbd*

Contains the device access mode: ro, rw or migration.

/sys/block/rnbd<N>/rnbd/mapping_path

Defined on file *sysfs-block-rnbd*

Contains the path that was passed as ”device_path” to the map_device operation.

/sys/block/rnbd<N>/rnbd/nr_poll_queues

Defined on file *sysfs-block-rnbd*

Contains the number of poll-mode queues

/sys/block/rnbd<N>/rnbd/remap_device

Defined on file *sysfs-block-rnbd*

Remap the disconnected device if the session is not destroyed yet.

/sys/block/rnbd<N>/rnbd/resize

Defined on file *sysfs-block-rnbd*

Write the number of sectors to change the size of the disk.

/sys/block/rnbd<N>/rnbd/session

Defined on file *sysfs-block-rnbd*

RNBD uses RTRS session to transport the data between client and server. The entry "session" contains the name of the session, that was used to establish the RTRS session. It's the same name that was passed as server parameter to the map_device entry.

/sys/block/rnbd<N>/rnbd/state

Defined on file *sysfs-block-rnbd*

The file contains the current state of the block device. The state file returns "open" when the device is successfully mapped from the server and accepting I/O requests. When the connection to the server gets disconnected in case of an error (e.g. link failure), the state file returns "closed" and all I/O requests submitted to it will fail with -EIO.

/sys/block/rnbd<N>/rnbd/unmap_device

Defined on file *sysfs-block-rnbd*

To unmap a volume, "normal" or "force" has to be written to: */sys/block/rnbd<N>/rnbd/unmap_device*

When "normal" is used, the operation will fail with EBUSY if any process is using the device. When "force" is used, the device is also unmapped when device is in use. All I/Os that are in progress will fail.

Example:

```
# echo "normal" > /sys/block/rnbd0/rnbd/unmap_device
```

/sys/block/rssd*/status

Defined on file *sysfs-block-rssd*

This is a read-only file. Indicates the status of the device.

/sys/block/zram<id>/backing_dev

Defined on file *sysfs-block-zram*

The backing_dev file is read-write and set up backing device for zram to write incompressible pages. For using, user should enable CONFIG_ZRAM_WRITEBACK.

/sys/block/zram<id>/bd_stat

Defined on file *sysfs-block-zram*

The bd_stat file is read-only and represents backing device's statistics (bd_count, bd_reads, bd_writes) in a format similar to block layer statistics file format.

/sys/block/zram<id>/comp_algorithm

Defined on file *sysfs-block-zram*

The comp_algorithm file is read-write and lets to show available and selected compression algorithms, change compression algorithm selection.

/sys/block/zram<id>/compact

Defined on file *sysfs-block-zram*

The compact file is write-only and trigger compaction for allocator zrm uses. The allocator moves some objects so that it could free fragment space.

/sys/block/zram<id>/debug_stat

Defined on file *sysfs-block-zram*

The debug_stat file is read-only and represents various device's debugging info useful for kernel developers. Its format is not documented intentionally and may change anytime without any notice.

/sys/block/zram<id>/disksize

Defined on file *sysfs-block-zram*

The disksize file is read-write and specifies the disk size which represents the limit on the *uncompressed* worth of data that can be stored in this disk. Unit: bytes

/sys/block/zram<id>/idle

Defined on file *sysfs-block-zram*

idle file is write-only and mark zram slot as idle. If system has mounted debugfs, user can see which slots are idle via /sys/kernel/debug/zram/zram<id>/block_state

/sys/block/zram<id>/initstate

Defined on file *sysfs-block-zram*

The initstate file is read-only and shows the initialization state of the device.

/sys/block/zram<id>/io_stat

Defined on file *sysfs-block-zram*

The io_stat file is read-only and accumulates device's I/O statistics not accounted by block layer. For example, failed_reads, failed_writes, etc. File format is similar to block layer statistics file format.

/sys/block/zram<id>/max_comp_streams

Defined on file *sysfs-block-zram*

The max_comp_streams file is read-write and specifies the number of backend's zcomp_strm compression streams (number of concurrent compress operations).

/sys/block/zram<id>/mem_limit

Defined on file *sysfs-block-zram*

The mem_limit file is write-only and specifies the maximum amount of memory ZRAM can use to store the compressed data. The limit could be changed in run time and "0" means disable the limit. No limit is the initial state. Unit: bytes

/sys/block/zram<id>/mem_used_max

Defined on file *sysfs-block-zram*

The mem_used_max file is write-only and is used to reset the counter of maximum memory zram have consumed to store compressed data. For resetting the value, you should write "0". Otherwise, you could see -EINVAL. Unit: bytes

/sys/block/zram<id>/mm_stat

Defined on file *sysfs-block-zram*

The mm_stat file is read-only and represents device's mm statistics (orig_data_size, compr_data_size, etc.) in a format similar to block layer statistics file format.

/sys/block/zram<id>/recomp_algorithm

Defined on file *sysfs-block-zram*

The recomp_algorithm file is read-write and allows to set or show secondary compression algorithms.

/sys/block/zram<id>/recompress

Defined on file *sysfs-block-zram*

The recompress file is write-only and triggers re-compression with secondary compression algorithms.

/sys/block/zram<id>/reset

Defined on file *sysfs-block-zram*

The reset file is write-only and allows resetting the device. The reset operation frees all the memory associated with this device.

/sys/block/zram<id>/writeback

Defined on file *sysfs-block-zram*

The writeback file is write-only and trigger idle and/or huge page writeback to backing device.

/sys/block/zram<id>/writeback_limit

Defined on file *sysfs-block-zram*

The writeback_limit file is read-write and specifies the maximum amount of writeback ZRAM can do. The limit could be changed in run time.

/sys/block/zram<id>/writeback_limit_enable

Defined on file *sysfs-block-zram*

The writeback_limit_enable file is read-write and specifies eanbe of writeback_limit feature. "1" means eable the feature. No limit "0" is the initial state.

5.2.31 Symbols under /sys/bus

/sys/bus/*/drivers/ufshcd/*/auto_hibern8

Defined on file *sysfs-driver-ufs*

This file contains the auto-hibernate idle timer setting of a UFS host controller. A value of '0' means auto-hibernate is not enabled. Otherwise the value is the number of microseconds of idle time before the UFS host controller will autonomously put the link into hibernate state. That will save power at the expense of increased latency. Note that the hardware supports 10-bit values with a power-of-ten multiplier which allows a maximum value of 102300000. Refer to the UFS Host Controller Interface specification for more details.

/sys/bus/.../drivers/intel-m10-bmc/.../bmc_version

Defined on file *sysfs-driver-intel-m10-bmc*

Read only. Returns the hardware build version of Intel MAX10 BMC chip. Format: "0x%x".

/sys/bus/.../drivers/intel-m10-bmc/.../bmcfw_version

Defined on file *sysfs-driver-intel-m10-bmc*

Read only. Returns the firmware version of Intel MAX10 BMC chip. Format: "0x%x".

/sys/bus/.../drivers/intel-m10-bmc/.../mac_address

Defined on file *sysfs-driver-intel-m10-bmc*

Read only. Returns the first MAC address in a block of sequential MAC addresses assigned to the board that is managed by the Intel MAX10 BMC. It is stored in FLASH storage and is mirrored in the MAX10 BMC register space. Format: "%02x:%02x:%02x:%02x:%02x:%02x".

/sys/bus/.../drivers/intel-m10-bmc/.../mac_count

Defined on file *sysfs-driver-intel-m10-bmc*

Read only. Returns the number of sequential MAC addresses assigned to the board managed by the Intel MAX10 BMC. This value is stored in FLASH and is mirrored in the MAX10 BMC register space. Format: "%u".

/sys/bus/acpi/devices/.../adr

Defined on file *sysfs-bus-acpi*

This attribute contains the output of the device object's _ADR control method, which is present for ACPI device objects representing devices having standard enumeration algorithms, such as PCI.

/sys/bus/acpi/devices/.../description

Defined on file *sysfs-bus-acpi*

This attribute contains the output of the device object's _STR control method, if present.

/sys/bus/acpi/devices/.../eject

Defined on file *sysfs-bus-acpi*

Writing 1 to this attribute will trigger hot removal of this device object. This file exists for every device object that has _EJ0 method.

/sys/bus/acpi/devices/.../hid

Defined on file *sysfs-bus-acpi*

This attribute indicates the hardware ID (_HID) of the device object. For example, PNP0103. This file is present for device objects having the _HID control method.

/sys/bus/acpi/devices/.../hrv

Defined on file *sysfs-bus-acpi*

(RO) Allows users to read the hardware version of non-PCI hardware, if the _HRV control method is present. It is mostly useful for non-PCI devices because lspci can list the hardware version for PCI devices.

/sys/bus/acpi/devices/.../modalias

Defined on file *sysfs-bus-acpi*

This attribute indicates the PNP IDs of the device object. That is acpi:HHHHHHHH:[CCCCCCC:]. Where each HHHHHHHH or CCCCCCCC contains device object's PNPID (_HID or _CID).

/sys/bus/acpi/devices/.../path

Defined on file *sysfs-bus-acpi*

This attribute indicates the full path of ACPI namespace object associated with the device object. For example, _SB_.PCI0.

This file is not present for device objects representing fixed ACPI hardware features (like power and sleep buttons).

/sys/bus/acpi/devices/.../status

Defined on file *sysfs-bus-acpi*

(RO) Returns the ACPI device status: enabled, disabled or functioning or present, if the method _STA is present.

The return value is a decimal integer representing the device's status bitmap:

Bit [0]	Set if the device is present.
Bit [1]	Set if the device is enabled and decoding its resources.
Bit [2]	Set if the device should be shown in the UI.
Bit [3]	Set if the device is functioning properly (cleared if device failed its diagnostics).
Bit [4]	Set if the battery is present.
Bits [31:5]	Reserved (must be cleared)

If bit [0] is clear, then bit 1 must also be clear (a device that is not present cannot be enabled).

Bit 0 can be clear (not present) with bit [3] set (device is functional). This case is used to indicate a valid device for which no device driver should be loaded.

More special cases are covered in the ACPI specification.

/sys/bus/acpi/devices/.../uid

Defined on file *sysfs-bus-acpi*

This attribute contains the output of the device object's _UID control method, if present.

/sys/bus/acpi/intel-rapid-start/wakeup_events

Defined on file *sysfs-driver-intel-rapid-start*

An integer representing a set of wakeup events as follows: 1: Wake to enter hibernation when the wakeup timer expires 2: Wake to enter hibernation when the battery reaches a critical level

These values are ORed together. For example, a value of 3 indicates that the system will wake to enter hibernation when either the wakeup timer expires or the battery reaches a critical level.

/sys/bus/acpi/intel-rapid-start/wakeup_time

Defined on file *sysfs-driver-intel-rapid-start*

An integer representing the length of time the system will remain asleep before waking up to enter hibernation. This value is in minutes.

/sys/bus/amba/devices/.../driver_override

Defined on file *sysfs-bus-amba*

This file allows the driver for a device to be specified which will override standard OF, ACPI, ID table, and name matching. When specified, only a driver with a name matching the value written to driver_override will have an opportunity to bind to the device. The override is specified by writing a string to the driver_override file (echo vfio-amba > driver_override) and may be cleared with an empty string (echo > driver_override). This returns the device to standard matching rules binding. Writing to driver_override does not automatically unbind the device from its current driver or make any attempt to automatically load the specified driver. If no driver with a matching name is currently loaded in the kernel, the device will not bind to any driver. This also allows devices to opt-out of driver binding using a driver_override name such as "none". Only a single driver may be specified in the override, there is no support for parsing delimiters.

/sys/bus/auxiliary/devices/intel_vsec.sdsi.X

Defined on file *sysfs-driver-intel_sdsi*

This directory contains interface files for accessing Intel On Demand (formerly Software Defined Silicon or SDSi) features on a CPU. X represents the socket instance (though not the socket ID). The socket ID is determined by reading the registers file and decoding it per the specification.

Some files communicate with On Demand hardware through a mailbox. Should the operation fail, one of the following error codes may be returned:

Error Code	Cause
EIO	General mailbox failure. Log may indicate cause.
EBUSY	Mailbox is owned by another agent.
EPERM	On Demand capability is not enabled in hardware.
EPROTO	Failure in mailbox protocol detected by driver. See log for details.
EOVERFLOW	For provision commands, the size of the data exceeds what may be written.
ESPIPE	Seeking is not allowed.
ETIMEDOUT	Failure to complete mailbox transaction in time.

/sys/bus/auxiliary/devices/intel_vsec.sdsi.X/guid

Defined on file *sysfs-driver-intel_sdsi*

(RO) The GUID for the registers file. The GUID identifies the layout of the registers file in this directory. Information about the register layouts for a particular GUID is available at http://github.com/intel/intel-sdsi

/sys/bus/auxiliary/devices/intel_vsec.sdsi.X/meter_certificate

Defined on file *sysfs-driver-intel_sdsi*

(RO) Used to read back the current meter certificate for the CPU from Intel On Demand hardware. The meter certificate contains utilization metrics of On Demand enabled features. Mailbox command.

/sys/bus/auxiliary/devices/intel_vsec.sdsi.X/provision_akc

Defined on file *sysfs-driver-intel_sdsi*

(WO) Used to write an Authentication Key Certificate (AKC) to the On Demand NVRAM for the CPU. The AKC is used to authenticate a Capability Activation Payload. Mailbox command.

/sys/bus/auxiliary/devices/intel_vsec.sdsi.X/provision_cap

Defined on file *sysfs-driver-intel_sdsi*

(WO) Used to write a Capability Activation Payload (CAP) to the On Demand NVRAM for the CPU. CAPs are used to activate a given CPU feature. A CAP is validated by On Demand hardware using a previously provisioned AKC file. Upon successful authentication, the CPU configuration is updated. A cold reboot is required to fully activate the feature. Mailbox command.

/sys/bus/auxiliary/devices/intel_vsec.sdsi.X/registers

Defined on file *sysfs-driver-intel_sdsi*

(RO) Contains information needed by applications to provision a CPU and monitor status information. The layout of this file is determined by the GUID in this directory. Information about the layout for a particular GUID is available at http://github.com/intel/intel-sdsi

/sys/bus/auxiliary/devices/intel_vsec.sdsi.X/state_certificate

Defined on file *sysfs-driver-intel_sdsi*

(RO) Used to read back the current state certificate for the CPU from On Demand hardware. The state certificate contains information about the current licenses on the CPU. Mailbox command.

/sys/bus/bcma/devices/.../class

Defined on file *sysfs-bus-bcma*

Each BCMA core is identified by few fields, including class it belongs to. See include/linux/bcma/bcma.h for possible values.

/sys/bus/bcma/devices/.../id

Defined on file *sysfs-bus-bcma*

There are a few types of BCMA cores, they can be identified by id field.

/sys/bus/bcma/devices/.../manuf

Defined on file *sysfs-bus-bcma*

Each BCMA core has its manufacturer id. See include/linux/bcma/bcma.h for possible values.

/sys/bus/bcma/devices/.../rev

Defined on file *sysfs-bus-bcma*

BCMA cores of the same type can still slightly differ depending on their revision. Use it for detailed programming.

/sys/bus/cdx/devices/.../class

Defined on file *sysfs-bus-cdx*

This file contains the class of the CDX device, in hexadecimal. Class is 24 bit identifier specifies the functionality of the device.

/sys/bus/cdx/devices/.../device

Defined on file *sysfs-bus-cdx*

Device ID for this CDX device, in hexadecimal. Device ID is 16 bit identifier to identify a device type within the range of a device manufacturer. Combination of Vendor ID and Device ID identifies a device.

/sys/bus/cdx/devices/.../enable

Defined on file *sysfs-bus-cdx*

CDX bus should be disabled before updating the devices in FPGA. Writing n/0/off will attempt to disable the CDX bus and. writing y/1/on will attempt to enable the CDX bus. Reading this file gives the current state of the bus, 1 for enabled and 0 for disabled.

For example:

```
# echo 1 > /sys/bus/cdx/.../enable
```

/sys/bus/cdx/devices/.../modalias

Defined on file *sysfs-bus-cdx*

This attribute indicates the CDX ID of the device. That is in the format: cdx:vXXXXdXXXXsvXXXXsdXXXXcXXXXXX, where:

- vXXXX contains the vendor ID;
- dXXXX contains the device ID;
- svXXXX contains the subsystem vendor ID;
- sdXXXX contains the subsystem device ID;
- cXXXXXX contains the device class.

/sys/bus/cdx/devices/.../remove

Defined on file *sysfs-bus-cdx*

Writing y/1/on to this file removes the corresponding device from the CDX bus. If the device is to be reconfigured reconfigured in the Hardware, the device can be removed, so that the device driver does not access the device while it is being reconfigured.

For example:

```
# echo 1 > /sys/bus/cdx/devices/.../remove
```

/sys/bus/cdx/devices/.../reset

Defined on file *sysfs-bus-cdx*

Writing y/1/on to this file resets the CDX device or all devices on the bus. On resetting the device, the corresponding driver is notified twice, once before the device is being reset, and again after the reset has been complete.

For example:

```
# echo 1 > /sys/bus/cdx/.../reset
```

/sys/bus/cdx/devices/.../resource<N>

Defined on file *sysfs-bus-cdx*

The resource binary file contains the content of the memory regions. These files can be m'maped from userspace.

/sys/bus/cdx/devices/.../revision

Defined on file *sysfs-bus-cdx*

This file contains the revision field of the CDX device, in hexadecimal. Revision is 8 bit revision identifier of the device.

/sys/bus/cdx/devices/.../subsystem_device

Defined on file *sysfs-bus-cdx*

Subsystem Device ID for this CDX device, in hexadecimal Subsystem Device ID is 16 bit identifier specific to the card manufacturer.

/sys/bus/cdx/devices/.../subsystem_vendor

Defined on file *sysfs-bus-cdx*

Subsystem Vendor ID for this CDX device, in hexadecimal. Subsystem Vendor ID is 16 bit identifier specific to the card manufacturer.

/sys/bus/cdx/devices/.../vendor

Defined on file *sysfs-bus-cdx*

Vendor ID for this CDX device, in hexadecimal. Vendor ID is 16 bit identifier which is specific to the device manufacturer. Combination of Vendor ID and Device ID identifies a device.

/sys/bus/cdx/rescan

Defined on file *sysfs-bus-cdx*

Writing y/1/on to this file will cause rescan of the bus and devices on the CDX bus. Any new devices are scanned and added to the list of Linux devices and any devices removed are also deleted from Linux.

For example:

```
# echo 1 > /sys/bus/cdx/rescan
```

/sys/bus/coreboot

Defined on file *sysfs-bus-coreboot*

The coreboot bus provides a variety of virtual devices used to access data structures created by the Coreboot BIOS.

/sys/bus/coreboot/devices/cbmem-<id>

Defined on file *sysfs-bus-coreboot*

CBMEM is a downwards-growing memory region created by Coreboot, and contains tagged data structures to be shared with payloads in the boot process and the OS. Each CBMEM entry is given a directory in /sys/bus/coreboot/devices based on its id. A list of ids known to Coreboot can be found in the coreboot source tree at `src/commonlib/bsd/include/commonlib/bsd/cbmem_id.h`.

/sys/bus/coreboot/devices/cbmem-<id>/address

Defined on file *sysfs-bus-coreboot*

This is the physical memory address that the CBMEM entry's data begins at, in hexadecimal (e.g., `0x76ffe000`).

/sys/bus/coreboot/devices/cbmem-<id>/mem

Defined on file *sysfs-bus-coreboot*

A file exposing read/write access to the entry's data. Note that this file does not support mmap(), as coreboot does not guarantee that the data will be page-aligned.

The mode of this file is 0600. While there shouldn't be anything security-sensitive contained in CBMEM, read access requires root privileges given this is exposing a small subset of physical memory.

/sys/bus/coreboot/devices/cbmem-<id>/size

Defined on file *sysfs-bus-coreboot*

This is the size of the CBMEM entry's data, in hexadecimal (e.g., `0x1234`).

/sys/bus/coresight/devices/<cti-name>/channels/chan_clear

Defined on file *sysfs-bus-coresight-devices-cti*

(Write) Deactivate a single channel.

/sys/bus/coresight/devices/<cti-name>/channels/chan_free

Defined on file *sysfs-bus-coresight-devices-cti*

(Read) show channels with no attached trigger signals.

/sys/bus/coresight/devices/<cti-name>/channels/chan_gate_disable

Defined on file *sysfs-bus-coresight-devices-cti*

(Write) Disable CTIGATE for single channel.

/sys/bus/coresight/devices/<cti-name>/channels/chan_gate_enable

Defined on file *sysfs-bus-coresight-devices-cti*

(RW) Enable CTIGATE for single channel (Write) or list enabled channels through the gate (R).

/sys/bus/coresight/devices/<cti-name>/channels/chan_inuse

Defined on file *sysfs-bus-coresight-devices-cti*

(Read) show channels with at least one attached trigger signal.

/sys/bus/coresight/devices/<cti-name>/channels/chan_pulse

Defined on file *sysfs-bus-coresight-devices-cti*

(Write) Pulse a single channel - activate for a single clock cycle.

/sys/bus/coresight/devices/<cti-name>/channels/chan_set

Defined on file *sysfs-bus-coresight-devices-cti*

(Write) Activate a single channel.

/sys/bus/coresight/devices/<cti-name>/channels/chan_xtrigs_in

Defined on file *sysfs-bus-coresight-devices-cti*

(Read) Read to see input triggers connected to selected view channel.

/sys/bus/coresight/devices/<cti-name>/channels/chan_xtrigs_out

Defined on file *sysfs-bus-coresight-devices-cti*

(Read) Read to see output triggers connected to selected view channel.

/sys/bus/coresight/devices/<cti-name>/channels/chan_xtrigs_reset

Defined on file *sysfs-bus-coresight-devices-cti*

(Write) Clear all channel / trigger programming.

/sys/bus/coresight/devices/<cti-name>/channels/chan_xtrigs_sel

Defined on file *sysfs-bus-coresight-devices-cti*

(RW) Write channel number to select a channel to view, read to see selected channel number.

/sys/bus/coresight/devices/<cti-name>/channels/trig_filter_enable

Defined on file *sysfs-bus-coresight-devices-cti*

(RW) Enable or disable trigger output signal filtering.

/sys/bus/coresight/devices/<cti-name>/channels/trigin_attach

Defined on file *sysfs-bus-coresight-devices-cti*

(Write) Attach a CTI input trigger to a CTM channel.

/sys/bus/coresight/devices/<cti-name>/channels/trigin_detach

Defined on file *sysfs-bus-coresight-devices-cti*

(Write) Detach a CTI input trigger from a CTM channel.

/sys/bus/coresight/devices/<cti-name>/channels/trigout_attach

Defined on file *sysfs-bus-coresight-devices-cti*

(Write) Attach a CTI output trigger to a CTM channel.

/sys/bus/coresight/devices/<cti-name>/channels/trigout_detach

Defined on file *sysfs-bus-coresight-devices-cti*

(Write) Detach a CTI output trigger from a CTM channel.

/sys/bus/coresight/devices/<cti-name>/channels/trigout_filtered

Defined on file *sysfs-bus-coresight-devices-cti*

(Read) List of output triggers filtered across all connections.

/sys/bus/coresight/devices/<cti-name>/ctmid

Defined on file *sysfs-bus-coresight-devices-cti*

(Read) Display the associated CTM ID

/sys/bus/coresight/devices/<cti-name>/enable

Defined on file *sysfs-bus-coresight-devices-cti*

(RW) Enable/Disable the CTI hardware.

/sys/bus/coresight/devices/<cti-name>/nr_trigger_cons

Defined on file *sysfs-bus-coresight-devices-cti*

(Read) Number of devices connected to triggers on this CTI

/sys/bus/coresight/devices/<cti-name>/powered

Defined on file *sysfs-bus-coresight-devices-cti*

(Read) Indicate if the CTI hardware is powered.

/sys/bus/coresight/devices/<cti-name>/regs/appclear

Defined on file *sysfs-bus-coresight-devices-cti*

(Write) Write APPCLEAR register to deactivate channel.

/sys/bus/coresight/devices/<cti-name>/regs/apppulse

Defined on file *sysfs-bus-coresight-devices-cti*

(Write) Write APPPULSE to pulse a channel active for one clock cycle.

/sys/bus/coresight/devices/<cti-name>/regs/appset

Defined on file *sysfs-bus-coresight-devices-cti*

(RW) Set CTIAPPSET register to activate channel. Read back to determine current value of register.

/sys/bus/coresight/devices/<cti-name>/regs/asicctl

Defined on file *sysfs-bus-coresight-devices-cti*

(RW) Read or write ASICCTL register.

/sys/bus/coresight/devices/<cti-name>/regs/chinstatus

Defined on file *sysfs-bus-coresight-devices-cti*

(Read) Read current status of channel inputs.

/sys/bus/coresight/devices/<cti-name>/regs/choutstatus

Defined on file *sysfs-bus-coresight-devices-cti*

(Read) read current status of channel outputs.

/sys/bus/coresight/devices/<cti-name>/regs/gate

Defined on file *sysfs-bus-coresight-devices-cti*

(RW) Read or write CTIGATE register.

/sys/bus/coresight/devices/<cti-name>/regs/inen

Defined on file *sysfs-bus-coresight-devices-cti*

(RW) Read or write the CTIINEN register selected by inout_sel.

/sys/bus/coresight/devices/<cti-name>/regs/inout_sel

Defined on file *sysfs-bus-coresight-devices-cti*

(RW) Select the index for inen and outen registers.

/sys/bus/coresight/devices/<cti-name>/regs/intack

Defined on file *sysfs-bus-coresight-devices-cti*

(Write) Write the INTACK register.

/sys/bus/coresight/devices/<cti-name>/regs/outen

Defined on file *sysfs-bus-coresight-devices-cti*

(RW) Read or write the CTIOUTEN register selected by inout_sel.

/sys/bus/coresight/devices/<cti-name>/regs/triginstatus

Defined on file *sysfs-bus-coresight-devices-cti*

(Read) read current status of input trigger signals

/sys/bus/coresight/devices/<cti-name>/regs/trigoutstatus

Defined on file *sysfs-bus-coresight-devices-cti*

(Read) read current status of output trigger signals.

/sys/bus/coresight/devices/<cti-name>/triggers<N>/in_signals

Defined on file *sysfs-bus-coresight-devices-cti*

(Read) Input trigger signals from connected device <N>

/sys/bus/coresight/devices/<cti-name>/triggers<N>/in_types

Defined on file *sysfs-bus-coresight-devices-cti*

(Read) Functional types for the input trigger signals from connected device <N>

/sys/bus/coresight/devices/<cti-name>/triggers<N>/name

Defined on file *sysfs-bus-coresight-devices-cti*

(Read) Name of connected device <N>

/sys/bus/coresight/devices/<cti-name>/triggers<N>/out_signals

Defined on file *sysfs-bus-coresight-devices-cti*

(Read) Output trigger signals to connected device <N>

/sys/bus/coresight/devices/<cti-name>/triggers<N>/out_types

Defined on file *sysfs-bus-coresight-devices-cti*

(Read) Functional types for the output trigger signals to connected device <N>

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/addr_acctype

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Used in conjunction with @addr_idx. Specifies characteristics about the address comparator being configure, for example the access type, the kind of instruction to trace, processor contect ID to trigger on, etc. Individual fields in the access type register may vary on the version of the trace entity.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/addr_idx

Defined on file *sysfs-bus-coresight-devices-etm3x*

Select which address comparator or pair (of comparators) to work with.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/addr_range

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Used in conjunction with @addr_idx. Specifies the range of addresses to trigger on. Inclusion or exclusion is specified in the corresponding access type register.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/addr_single

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Used in conjunction with @addr_idx. Specifies the single address to trigger on, highly influenced by the configuration options of the corresponding access type register.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/addr_start

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Used in conjunction with @addr_idx. Specifies the single address to start tracing on, highly influenced by the configuration options of the corresponding access type register.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/addr_stop

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Used in conjunction with @addr_idx. Specifies the single address to stop tracing on, highly influenced by the configuration options of the corresponding access type register.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/cntr_event

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Used in conjunction with cntr_idx, give access to the counter event register.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/cntr_idx

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Specifies the counter to work on.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/cntr_rld_event

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Used in conjunction with cntr_idx, give access to the counter reload event register.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/cntr_rld_val

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Used in conjunction with cntr_idx, give access to the counter reload value register.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/cntr_val

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Used in conjunction with cntr_idx, give access to the counter value register.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/cpu

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RO) Holds the cpu number this tracer is affined to.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/ctxid_idx

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Specifies the index of the context ID register to be selected.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/ctxid_mask

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Mask to apply to all the context ID comparator.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/ctxid_pid

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Used with the ctxid_idx, specify with context ID to trigger on.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/curr_seq_state

Defined on file *sysfs-bus-coresight-devices-etm3x*

(Read) Holds the current state of the sequencer.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/enable_event

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Defines which event triggers a trace.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/enable_source

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Enable/disable tracing on this specific trace entiry. Enabling a source implies the source has been configured properly and a sink has been identidifed for it. The path of coresight components linking the source to the sink is configured and managed automatically by the coresight framework.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/etmsr

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Gives access to the ETM status register, which holds programming information and status on certains events.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/fifofull_level

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Number of byte left in the fifo before considering it full. Depending on the tracer's version, can also hold threshold for data suppression.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/mgmt/etmccer

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RO) Print the content of the ETM Configuration Code Extension register (0x1e8). The value is read directly from the HW.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/mgmt/etmccr

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RO) Print the content of the ETM Configuration Code register (0x004). The value is read directly from the HW.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/mgmt/etmcr

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RO) Print the content of the ETM Main Control register (0x000). The value is read directly from the HW.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/mgmt/etmidr

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RO) Print the content of the ETM ID register (0x1e4). The value is read directly from the HW.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/mgmt/etmscr

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RO) Print the content of the ETM System Configuration register (0x014). The value is read directly from the HW.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/mgmt/etmtecr1

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RO) Print the content of the ETM Enable Control #1 register (0x024). The value is read directly from the HW.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/mgmt/etmtecr2

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RO) Print the content of the ETM Enable Control #2 register (0x01c). The value is read directly from the HW.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/mgmt/etmteevr

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RO) Print the content of the ETM Trace Enable Event register (0x020). The value is read directly from the HW.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/mgmt/etmtraceidr

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RO) Print the content of the ETM Trace ID register (0x200). The value is read directly from the HW.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/mgmt/etmtsscr

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RO) Print the content of the ETM Trace Start/Stop Control register (0x018). The value is read directly from the HW.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/mode

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Interface with the driver's 'mode' field, controlling various aspect of the trace entity such as time stamping, context ID size and cycle accurate tracing. Driver specific and bound to change depending on the driver.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/nr_addr_cmp

Defined on file *sysfs-bus-coresight-devices-etm3x*

(Read) Provides the number of address comparators pairs accessible on a trace unit, as specified by bit 3:0 of register ETMCCR.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/nr_cntr

Defined on file *sysfs-bus-coresight-devices-etm3x*

(Read) Provides the number of counters accessible on a trace unit, as specified by bit 15:13 of register ETMCCR.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/nr_ctxid_cmp

Defined on file *sysfs-bus-coresight-devices-etm3x*

(Read) Provides the number of context ID comparator available on a trace unit, as specified by bit 25:24 of register ETMCCR.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/reset

Defined on file *sysfs-bus-coresight-devices-etm3x*

(Write) Cancels all configuration on a trace unit and set it back to its boot configuration.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/seq_12_event

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Defines the event that causes the sequencer to transition from state 1 to state 2.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/seq_13_event

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Defines the event that causes the sequencer to transition from state 1 to state 3.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/seq_21_event

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Defines the event that causes the sequencer to transition from state 2 to state 1.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/seq_23_event

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Defines the event that causes the sequencer to transition from state 2 to state 3.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/seq_31_event

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Defines the event that causes the sequencer to transition from state 3 to state 1.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/seq_32_event

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Defines the event that causes the sequencer to transition from state 3 to state 2.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/sync_freq

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Holds the trace synchronization frequency value - must be programmed with the various implementation behavior in mind.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/timestamp_event

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Defines an event that requests the insertion of a timestamp into the trace stream.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/traceid

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RO) Holds the trace ID that will appear in the trace stream coming from this trace entity.

/sys/bus/coresight/devices/<memory_map>.[etm|ptm]/trigger_event

Defined on file *sysfs-bus-coresight-devices-etm3x*

(RW) Define the event that controls the trigger.

/sys/bus/coresight/devices/<memory_map>.etb/enable_sink

Defined on file *sysfs-bus-coresight-devices-etb10*

(RW) Add/remove a sink from a trace path. There can be multiple source for a single sink.

ex:

```
echo 1 > /sys/bus/coresight/devices/20010000.etb/enable_sink
```

/sys/bus/coresight/devices/<memory_map>.etb/mgmt/ctl

Defined on file *sysfs-bus-coresight-devices-etb10*

(Read) Shows the value held by the ETB Control register. The value is read directly from HW register CTL, 0x020.

/sys/bus/coresight/devices/<memory_map>.etb/mgmt/ffcr

Defined on file *sysfs-bus-coresight-devices-etb10*

(Read) Shows the value held by the ETB Formatter and Flush Control register. The value is read directly from HW register FFCR, 0x304.

/sys/bus/coresight/devices/<memory_map>.etb/mgmt/ffsr

Defined on file *sysfs-bus-coresight-devices-etb10*

(Read) Shows the value held by the ETB Formatter and Flush Status register. The value is read directly from HW register FFSR, 0x300.

/sys/bus/coresight/devices/<memory_map>.etb/mgmt/rdp

Defined on file *sysfs-bus-coresight-devices-etb10*

(Read) Defines the depth, in words, of the trace RAM in powers of 2. The value is read directly from HW register RDP, 0x004.

/sys/bus/coresight/devices/<memory_map>.etb/mgmt/rrp

Defined on file *sysfs-bus-coresight-devices-etb10*

(Read) Shows the value held by the ETB RAM Read Pointer register that is used to read entries from the Trace RAM over the APB interface. The value is read directly from HW register RRP, 0x014.

/sys/bus/coresight/devices/<memory_map>.etb/mgmt/rwp

Defined on file *sysfs-bus-coresight-devices-etb10*

(Read) Shows the value held by the ETB RAM Write Pointer register that is used to sets the write pointer to write entries from the CoreSight bus into the Trace RAM. The value is read directly from HW register RWP, 0x018.

/sys/bus/coresight/devices/<memory_map>.etb/mgmt/sts

Defined on file *sysfs-bus-coresight-devices-etb10*

(Read) Shows the value held by the ETB status register. The value is read directly from HW register STS, 0x00C.

/sys/bus/coresight/devices/<memory_map>.etb/mgmt/trg

Defined on file *sysfs-bus-coresight-devices-etb10*

(Read) Similar to ”trigger_cntr” above except that this value is read directly from HW register TRG, 0x01C.

/sys/bus/coresight/devices/<memory_map>.etb/trigger_cntr

Defined on file *sysfs-bus-coresight-devices-etb10*

(RW) Disables write access to the Trace RAM by stopping the formatter after a defined number of words have been stored following the trigger event. The number of 32-bit words written into the Trace RAM following the trigger event is equal to the value stored in this register+1 (from ARM ETB-TRM).

/sys/bus/coresight/devices/<memory_map>.funnel/funnel_ctrl

Defined on file *sysfs-bus-coresight-devices-funnel*

(RW) Enables the slave ports and defines the hold time of the slave ports.

/sys/bus/coresight/devices/<memory_map>.funnel/priority

Defined on file *sysfs-bus-coresight-devices-funnel*

(RW) Defines input port priority order.

/sys/bus/coresight/devices/<memory_map>.stm/enable_source

Defined on file *sysfs-bus-coresight-devices-stm*

(RW) Enable/disable tracing on this specific trace macrocell. Enabling the trace macrocell implies it has been configured properly and a sink has been identified for it. The path of coresight components linking the source to the sink is configured and managed automatically by the coresight framework.

/sys/bus/coresight/devices/<memory_map>.stm/hwevent_enable

Defined on file *sysfs-bus-coresight-devices-stm*

(RW) Provides access to the HW event enable register, used in conjunction with HW event bank select register.

/sys/bus/coresight/devices/<memory_map>.stm/hwevent_select

Defined on file *sysfs-bus-coresight-devices-stm*

(RW) Gives access to the HW event block select register (STMHEBSR) in order to configure up to 256 channels. Used in conjunction with "hwevent_enable" register as described above.

/sys/bus/coresight/devices/<memory_map>.stm/port_enable

Defined on file *sysfs-bus-coresight-devices-stm*

(RW) Provides access to the stimulus port enable register (STMSPER). Used in conjunction with "port_select" described below.

/sys/bus/coresight/devices/<memory_map>.stm/port_select

Defined on file *sysfs-bus-coresight-devices-stm*

(RW) Used to determine which bank of stimulus port bit in register STMSPER (see above) apply to.

/sys/bus/coresight/devices/<memory_map>.stm/status

Defined on file *sysfs-bus-coresight-devices-stm*

(Read) List various control and status registers. The specific layout and content is driver specific.

/sys/bus/coresight/devices/<memory_map>.stm/traceid

Defined on file *sysfs-bus-coresight-devices-stm*

(RW) Holds the trace ID that will appear in the trace stream coming from this trace entity.

/sys/bus/coresight/devices/<memory_map>.tmc/buf_mode_preferred

Defined on file *sysfs-bus-coresight-devices-tmc*

(RW) Current Coresight TMC-ETR buffer mode selected. But user could only provide a mode which is supported for a given ETR device. This file is available only for TMC ETR devices.

/sys/bus/coresight/devices/<memory_map>.tmc/buf_modes_available

Defined on file *sysfs-bus-coresight-devices-tmc*

(Read) Shows all supported Coresight TMC-ETR buffer modes available for the users to configure explicitly. This file is avaialble only for TMC ETR devices.

/sys/bus/coresight/devices/<memory_map>.tmc/buffer_size

Defined on file *sysfs-bus-coresight-devices-tmc*

(RW) Size of the trace buffer for TMC-ETR when used in SYSFS mode. Writable only for TMC-ETR configurations. The value should be aligned to the kernel pagesize.

/sys/bus/coresight/devices/<memory_map>.tmc/mgmt/ctl

Defined on file *sysfs-bus-coresight-devices-tmc*

(Read) Shows the value held by the TMC Control register. The value is read directly from HW register CTL, 0x020.

/sys/bus/coresight/devices/<memory_map>.tmc/mgmt/devid

Defined on file *sysfs-bus-coresight-devices-tmc*

(Read) Indicates the capabilities of the Coresight TMC. The value is read directly from the DEVID register, 0xFC8,

/sys/bus/coresight/devices/<memory_map>.tmc/mgmt/ffcr

Defined on file *sysfs-bus-coresight-devices-tmc*

(Read) Shows the value held by the TMC Formatter and Flush Control register. The value is read directly from HW register FFCR, 0x304.

/sys/bus/coresight/devices/<memory_map>.tmc/mgmt/ffsr

Defined on file *sysfs-bus-coresight-devices-tmc*

(Read) Shows the value held by the TMC Formatter and Flush Status register. The value is read directly from HW register FFSR, 0x300.

/sys/bus/coresight/devices/<memory_map>.tmc/mgmt/mode

Defined on file *sysfs-bus-coresight-devices-tmc*

(Read) Shows the value held by the TMC Mode register, which indicate the mode the device has been configured to enact. The The value is read directly from the MODE register, 0x028.

/sys/bus/coresight/devices/<memory_map>.tmc/mgmt/rrp

Defined on file *sysfs-bus-coresight-devices-tmc*

(Read) Shows the value held by the TMC RAM Read Pointer register that is used to read entries from the Trace RAM over the APB interface. The value is read directly from HW register RRP, 0x014.

/sys/bus/coresight/devices/<memory_map>.tmc/mgmt/rsz

Defined on file *sysfs-bus-coresight-devices-tmc*

(Read) Defines the size, in 32-bit words, of the local RAM buffer. The value is read directly from HW register RSZ, 0x004.

/sys/bus/coresight/devices/<memory_map>.tmc/mgmt/rwp

Defined on file *sysfs-bus-coresight-devices-tmc*

(Read) Shows the value held by the TMC RAM Write Pointer register that is used to sets the write pointer to write entries from the CoreSight bus into the Trace RAM. The value is read directly from HW register RWP, 0x018.

/sys/bus/coresight/devices/<memory_map>.tmc/mgmt/sts

Defined on file *sysfs-bus-coresight-devices-tmc*

(Read) Shows the value held by the TMC status register. The value is read directly from HW register STS, 0x00C.

/sys/bus/coresight/devices/<memory_map>.tmc/mgmt/trg

Defined on file *sysfs-bus-coresight-devices-tmc*

(Read) Similar to ”trigger_cntr” above except that this value is read directly from HW register TRG, 0x01C.

/sys/bus/coresight/devices/<memory_map>.tmc/trigger_cntr

Defined on file *sysfs-bus-coresight-devices-tmc*

(RW) Disables write access to the Trace RAM by stopping the formatter after a defined number of words have been stored following the trigger event. Additional interface for this driver are expected to be added as it matures.

/sys/bus/coresight/devices/<tpdm-name>/dsb_edge/ctrl_idx

Defined on file *sysfs-bus-coresight-devices-tpdm*

(RW) Set/Get the index number of the edge detection for the DSB subunit TPDM. Since there are at most 256 edge detections, this value ranges from 0 to 255.

/sys/bus/coresight/devices/<tpdm-name>/dsb_edge/ctrl_mask

Defined on file *sysfs-bus-coresight-devices-tpdm*

Write a data to mask the edge detection corresponding to the index number. Before writing data to this sysfs file, "ctrl_idx" should be written first to configure the index number of the edge detection which needs to be masked.

Accepts only one of the 2 values - 0 or 1.

/sys/bus/coresight/devices/<tpdm-name>/dsb_edge/ctrl_val

Defined on file *sysfs-bus-coresight-devices-tpdm*

Write a data to control the edge detection corresponding to the index number. Before writing data to this sysfs file, "ctrl_idx" should be written first to configure the index number of the edge detection which needs to be controlled.

Accepts only one of the following values. 0 - Rising edge detection 1 - Falling edge detection 2 - Rising and falling edge detection (toggle detection)

/sys/bus/coresight/devices/<tpdm-name>/dsb_edge/edcmr[0:7]

Defined on file *sysfs-bus-coresight-devices-tpdm*

Read a set of the edge control mask of the DSB in TPDM.

/sys/bus/coresight/devices/<tpdm-name>/dsb_edge/edcr[0:15]

Defined on file *sysfs-bus-coresight-devices-tpdm*

Read a set of the edge control value of the DSB in TPDM.

/sys/bus/coresight/devices/<tpdm-name>/dsb_mode

Defined on file *sysfs-bus-coresight-devices-tpdm*

(RW) Set/Get the programming mode of the DSB for tpdm.

Accepts the value needs to be greater than 0. What data bits do is listed below. Bit[0:1] : Test mode control bit for choosing the inputs. Bit[3] : Set to 0 for low performance mode. Set to 1 for high performance mode. Bit[4:8] : Select byte lane for high performance mode.

/sys/bus/coresight/devices/<tpdm-name>/dsb_msr/msr[0:31]

Defined on file *sysfs-bus-coresight-devices-tpdm*

(RW) Set/Get the MSR(mux select register) for the DSB subunit TPDM.

/sys/bus/coresight/devices/<tpdm-name>/dsb_patt/enable_ts

Defined on file *sysfs-bus-coresight-devices-tpdm*

(Write) Set the pattern timestamp of DSB tpdm. Read the pattern timestamp of DSB tpdm.

Accepts only one of the 2 values - 0 or 1. 0 : Disable DSB pattern timestamp. 1 : Enable DSB pattern timestamp.

/sys/bus/coresight/devices/<tpdm-name>/dsb_patt/set_type

Defined on file *sysfs-bus-coresight-devices-tpdm*

(Write) Set the pattern type of DSB tpdm. Read the pattern type of DSB tpdm.

Accepts only one of the 2 values - 0 or 1. 0 : Set the DSB pattern type to value. 1 : Set the DSB pattern type to toggle.

/sys/bus/coresight/devices/<tpdm-name>/dsb_patt/tpmr[0:7]

Defined on file *sysfs-bus-coresight-devices-tpdm*

(RW) Set/Get the mask of the pattern for the DSB subunit TPDM.

/sys/bus/coresight/devices/<tpdm-name>/dsb_patt/tpr[0:7]

Defined on file *sysfs-bus-coresight-devices-tpdm*

(RW) Set/Get the value of the pattern for the DSB subunit TPDM.

/sys/bus/coresight/devices/<tpdm-name>/dsb_trig_patt/xpmr[0:7]

Defined on file *sysfs-bus-coresight-devices-tpdm*

(RW) Set/Get the mask of the trigger pattern for the DSB subunit TPDM.

/sys/bus/coresight/devices/<tpdm-name>/dsb_trig_patt/xpr[0:7]

Defined on file *sysfs-bus-coresight-devices-tpdm*

(RW) Set/Get the value of the trigger pattern for the DSB subunit TPDM.

/sys/bus/coresight/devices/<tpdm-name>/dsb_trig_ts

Defined on file *sysfs-bus-coresight-devices-tpdm*

(RW) Set/Get the trigger timestamp of the DSB for tpdm.

Accepts only one of the 2 values - 0 or 1. 0 : Set the DSB trigger type to false 1 : Set the DSB trigger type to true

/sys/bus/coresight/devices/<tpdm-name>/dsb_trig_type

Defined on file *sysfs-bus-coresight-devices-tpdm*

(RW) Set/Get the trigger type of the DSB for tpdm.

Accepts only one of the 2 values - 0 or 1. 0 : Set the DSB trigger type to false 1 : Set the DSB trigger type to true

/sys/bus/coresight/devices/<tpdm-name>/integration_test

Defined on file *sysfs-bus-coresight-devices-tpdm*

(Write) Run integration test for tpdm. Integration test will generate test data for tpdm. It can help to make sure that the trace path is enabled and the link configurations are fine.

Accepts only one of the 2 values - 1 or 2. 1 : Generate 64 bits data 2 : Generate 32 bits data

/sys/bus/coresight/devices/<tpdm-name>/reset_dataset

Defined on file *sysfs-bus-coresight-devices-tpdm*

(Write) Reset the dataset of the tpdm.

Accepts only one value - 1. 1 : Reset the dataset of the tpdm

/sys/bus/coresight/devices/etm<N>/addr_cmp_view

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Print the current settings for the selected address comparator.

/sys/bus/coresight/devices/etm<N>/addr_exlevel_s_ns

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Set the Exception Level matching bits for secure and non-secure exception levels.

/sys/bus/coresight/devices/etm<N>/addr_idx

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Select which address comparator or pair (of comparators) to work with.

/sys/bus/coresight/devices/etm<N>/addr_instdatatype

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Controls what type of comparison the trace unit performs.

/sys/bus/coresight/devices/etm<N>/addr_range

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Used to setup address range comparator values.

/sys/bus/coresight/devices/etm<N>/addr_single

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Used to setup single address comparator values.

/sys/bus/coresight/devices/etm<N>/bb_ctrl

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Controls which regions in the memory map are enabled to use branch broadcasting.

/sys/bus/coresight/devices/etm<N>/cntr_ctrl

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Controls the operation of the selected counter.

/sys/bus/coresight/devices/etm<N>/cntr_idx

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Select which counter unit to work with.

/sys/bus/coresight/devices/etm<N>/cntr_val

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) This sets or returns the current count value of the specific counter.

/sys/bus/coresight/devices/etm<N>/cntrldvr

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) This sets or returns the reload count value of the specific counter.

/sys/bus/coresight/devices/etm<N>/cpu

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) The CPU this tracing entity is associated with.

/sys/bus/coresight/devices/etm<N>/ctxid_idx

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Select which context ID comparator to work with.

/sys/bus/coresight/devices/etm<N>/ctxid_masks

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Mask for all 8 context ID comparator value registers (if implemented).

/sys/bus/coresight/devices/etm<N>/ctxid_pid

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Get/Set the context ID comparator value to trigger on.

/sys/bus/coresight/devices/etm<N>/cyc_threshold

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Sets the threshold value for cycle counting.

/sys/bus/coresight/devices/etm<N>/enable_source

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Enable/disable tracing on this specific trace entiry. Enabling a source implies the source has been configured properly and a sink has been identidifed for it. The path of coresight components linking the source to the sink is configured and managed automatically by the coresight framework.

/sys/bus/coresight/devices/etm<N>/event

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Controls the tracing of arbitrary events from bank 0 to 3.

/sys/bus/coresight/devices/etm<N>/event_instren

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Controls the behavior of the events in bank 0 to 3.

/sys/bus/coresight/devices/etm<N>/event_ts

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Controls the insertion of global timestamps in the trace streams.

/sys/bus/coresight/devices/etm<N>/event_vinst

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Controls instruction trace filtering.

/sys/bus/coresight/devices/etm<N>/mgmt/trcauthstatus

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Print the content of the Authentication Status Register (0xFB8). The value is taken directly from the HW.

/sys/bus/coresight/devices/etm<N>/mgmt/trcconfig

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Print the content of the trace configuration register (0x010) as currently set by SW.

/sys/bus/coresight/devices/etm<N>/mgmt/trcdevarch

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Print the content of the Device Architecture Register (offset 0xFBC). The value is taken directly read from the HW.

/sys/bus/coresight/devices/etm<N>/mgmt/trcdevid

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Print the content of the Device ID Register (0xFC8). The value is taken directly from the HW.

/sys/bus/coresight/devices/etm<N>/mgmt/trcdevtype

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Print the content of the Device Type Register (0xFCC). The value is taken directly from the HW.

/sys/bus/coresight/devices/etm<N>/mgmt/trclsr

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Print the content of the SW Lock Status Register (0xFB4). The value is taken directly from the HW.

/sys/bus/coresight/devices/etm<N>/mgmt/trcoslsr

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Print the content of the OS Lock Status Register (0x304). The value it taken directly from the HW.

/sys/bus/coresight/devices/etm<N>/mgmt/trcpdcr

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Print the content of the Power Down Control Register (0x310). The value is taken directly from the HW.

/sys/bus/coresight/devices/etm<N>/mgmt/trcpdsr

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Print the content of the Power Down Status Register (0x314). The value is taken directly from the HW.

/sys/bus/coresight/devices/etm<N>/mgmt/trcpidr0

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Print the content of the Peripheral ID0 Register (0xFE0). The value is taken directly from the HW.

/sys/bus/coresight/devices/etm<N>/mgmt/trcpidr1

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Print the content of the Peripheral ID1 Register (0xFE4). The value is taken directly from the HW.

/sys/bus/coresight/devices/etm<N>/mgmt/trcpidr2

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Print the content of the Peripheral ID2 Register (0xFE8). The value is taken directly from the HW.

/sys/bus/coresight/devices/etm<N>/mgmt/trcpidr3

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Print the content of the Peripheral ID3 Register (0xFEC). The value is taken directly from the HW.

/sys/bus/coresight/devices/etm<N>/mgmt/trctraceid

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Print the content of the trace ID register (0x040).

/sys/bus/coresight/devices/etm<N>/mode

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Controls various modes supported by this ETM, for example P0 instruction tracing, branch broadcast, cycle counting and context ID tracing.

/sys/bus/coresight/devices/etm<N>/nr_addr_cmp

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Indicates the number of address comparator pairs that are available for tracing.

/sys/bus/coresight/devices/etm<N>/nr_cntr

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Indicates the number of counters that are available for tracing.

/sys/bus/coresight/devices/etm<N>/nr_ext_inp

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Indicates how many external inputs are implemented.

/sys/bus/coresight/devices/etm<N>/nr_pe_cmp

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Indicates the number of PE comparator inputs that are available for tracing.

/sys/bus/coresight/devices/etm<N>/nr_resource

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Indicates the number of resource selection pairs that are available for tracing.

/sys/bus/coresight/devices/etm<N>/nr_ss_cmp

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Indicates the number of single-shot comparator controls that are available for tracing.

/sys/bus/coresight/devices/etm<N>/nrseqstate

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Indicates the number of sequencer states that are implemented.

/sys/bus/coresight/devices/etm<N>/ns_exlevel_vinst

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) In non-secure state, each bit controls whether instruction tracing is enabled for the corresponding exception level.

/sys/bus/coresight/devices/etm<N>/numcidc

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Indicates the number of Context ID comparators that are available for tracing.

/sys/bus/coresight/devices/etm<N>/numvmidc

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Indicates the number of VMID comparators that are available for tracing.

/sys/bus/coresight/devices/etm<N>/pe

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Controls which PE to trace.

/sys/bus/coresight/devices/etm<N>/res_ctrl

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Controls the selection of the resources in the trace unit.

/sys/bus/coresight/devices/etm<N>/res_idx

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Select which resource selection unit to work with.

/sys/bus/coresight/devices/etm<N>/reset

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Write) Cancels all configuration on a trace unit and set it back to its boot configuration.

/sys/bus/coresight/devices/etm<N>/s_exlevel_vinst

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) In Secure state, each bit controls whether instruction tracing is enabled for the corresponding exception level.

/sys/bus/coresight/devices/etm<N>/seq_event

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Moves the sequencer state to a specific state.

/sys/bus/coresight/devices/etm<N>/seq_idx

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Select which sequensor.

/sys/bus/coresight/devices/etm<N>/seq_reset_event

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Moves the sequencer to state 0 when a programmed event occurs.

/sys/bus/coresight/devices/etm<N>/seq_state

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Use this to set, or read, the sequencer state.

/sys/bus/coresight/devices/etm<N>/sshot_ctrl

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Access the selected single shot control register.

/sys/bus/coresight/devices/etm<N>/sshot_idx

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Select the single shot control register to access.

/sys/bus/coresight/devices/etm<N>/sshot_pe_ctrl

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Access the selected single show PE comparator control register.

/sys/bus/coresight/devices/etm<N>/sshot_status

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Print the current value of the selected single shot status register.

/sys/bus/coresight/devices/etm<N>/syncfreq

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Controls how often trace synchronization requests occur.

/sys/bus/coresight/devices/etm<N>/trcidr/trcidr0

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Returns the tracing capabilities of the trace unit (0x1E0). The value is taken directly from the HW.

/sys/bus/coresight/devices/etm<N>/trcidr/trcidr1

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Returns the tracing capabilities of the trace unit (0x1E4). The value is taken directly from the HW.

/sys/bus/coresight/devices/etm<N>/trcidr/trcidr10

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Returns the number of P1 right-hand keys that the trace unit can use (0x188). The value is taken directly from the HW.

/sys/bus/coresight/devices/etm<N>/trcidr/trcidr11

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Returns the number of special P1 right-hand keys that the trace unit can use (0x18C). The value is taken directly from the HW.

/sys/bus/coresight/devices/etm<N>/trcidr/trcidr12

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Returns the number of conditional P1 right-hand keys that the trace unit can use (0x190). The value is taken directly from the HW.

/sys/bus/coresight/devices/etm<N>/trcidr/trcidr13

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Returns the number of special conditional P1 right-hand keys that the trace unit can use (0x194). The value is taken directly from the HW.

/sys/bus/coresight/devices/etm<N>/trcidr/trcidr2

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Returns the maximum size of the data value, data address, VMID, context ID and instruction address in the trace unit (0x1E8). The value is taken directly from the HW.

/sys/bus/coresight/devices/etm<N>/trcidr/trcidr3

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Returns the value associated with various resources available to the trace unit. See the Trace Macrocell architecture specification for more details (0x1E8). The value is taken directly from the HW.

/sys/bus/coresight/devices/etm<N>/trcidr/trcidr4

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Returns how many resources the trace unit supports (0x1F0). The value is taken directly from the HW.

/sys/bus/coresight/devices/etm<N>/trcidr/trcidr5

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Returns how many resources the trace unit supports (0x1F4). The value is taken directly from the HW.

/sys/bus/coresight/devices/etm<N>/trcidr/trcidr8

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Returns the maximum speculation depth of the instruction trace stream. (0x180). The value is taken directly from the HW.

/sys/bus/coresight/devices/etm<N>/trcidr/trcidr9

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) Returns the number of P0 right-hand keys that the trace unit can use (0x184). The value is taken directly from the HW.

/sys/bus/coresight/devices/etm<N>/ts_source

Defined on file *sysfs-bus-coresight-devices-etm4x*

(Read) When FEAT_TRF is implemented, value of TRFCR_ELx.TS used for trace session. Otherwise -1 indicates an unknown time source. Check trcidr0.tssize to see if a global timestamp is available.

/sys/bus/coresight/devices/etm<N>/vinst_pe_cmp_start_stop

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Access the start stop control register for PE input comparators.

/sys/bus/coresight/devices/etm<N>/vmid_idx

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Select which virtual machine ID comparator to work with.

/sys/bus/coresight/devices/etm<N>/vmid_masks

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Mask for all 8 virtual machine ID comparator value registers (if implemented).

/sys/bus/coresight/devices/etm<N>/vmid_val

Defined on file *sysfs-bus-coresight-devices-etm4x*

(RW) Get/Set the virtual machine ID comparator value to trigger on.

/sys/bus/coresight/devices/trbe<cpu>/align

Defined on file *sysfs-bus-coresight-devices-trbe*

(Read) Shows the TRBE write pointer alignment. This value is fetched from the TRBIDR register.

/sys/bus/coresight/devices/trbe<cpu>/flag

Defined on file *sysfs-bus-coresight-devices-trbe*

(Read) Shows if TRBE updates in the memory are with access and dirty flag updates as well. This value is fetched from the TRBIDR register.

/sys/bus/coresight/devices/ultra_smb<N>/enable_sink

Defined on file *sysfs-bus-coresight-devices-ultra_smb*

(RW) Add/remove a SMB device from a trace path. There can be multiple sources for a single SMB device.

/sys/bus/coresight/devices/ultra_smb<N>/mgmt/buf_size

Defined on file *sysfs-bus-coresight-devices-ultra_smb*

(RO) Shows the buffer size of each UltraSoc SMB device.

/sys/bus/coresight/devices/ultra_smb<N>/mgmt/buf_status

Defined on file *sysfs-bus-coresight-devices-ultra_smb*

(RO) Shows the value of UltraSoc SMB status register. BIT(0) is zero means buffer is empty.

/sys/bus/coresight/devices/ultra_smb<N>/mgmt/read_pos

Defined on file *sysfs-bus-coresight-devices-ultra_smb*

(RO) Shows the value of UltraSoc SMB Read Pointer register.

/sys/bus/coresight/devices/ultra_smb<N>/mgmt/write_pos

Defined on file *sysfs-bus-coresight-devices-ultra_smb*

(RO) Shows the value of UltraSoc SMB Write Pointer register.

/sys/bus/counter/devices/counterX/cascade_counts_enable

Defined on file *sysfs-bus-counter*

Indicates the cascading of Counts on Counter X.

Valid attribute values are boolean.

/sys/bus/counter/devices/counterX/cascade_counts_enable_component_id
/sys/bus/counter/devices/counterX/external_input_phase_clock_select_component_id
/sys/bus/counter/devices/counterX/countY/capture_component_id
/sys/bus/counter/devices/counterX/countY/ceiling_component_id
/sys/bus/counter/devices/counterX/countY/floor_component_id
/sys/bus/counter/devices/counterX/countY/count_mode_component_id
/sys/bus/counter/devices/counterX/countY/direction_component_id
/sys/bus/counter/devices/counterX/countY/enable_component_id
/sys/bus/counter/devices/counterX/countY/error_noise_component_id
/sys/bus/counter/devices/counterX/countY/prescaler_component_id
/sys/bus/counter/devices/counterX/countY/preset_component_id
/sys/bus/counter/devices/counterX/countY/preset_enable_component_id
/sys/bus/counter/devices/counterX/countY/signalZ_action_component_id
/sys/bus/counter/devices/counterX/countY/num_overflows_component_id
/sys/bus/counter/devices/counterX/signalY/cable_fault_component_id
/sys/bus/counter/devices/counterX/signalY/cable_fault_enable_component_id
/sys/bus/counter/devices/counterX/signalY/filter_clock_prescaler_component_id
/sys/bus/counter/devices/counterX/signalY/index_polarity_component_id
/sys/bus/counter/devices/counterX/signalY/polarity_component_id
/sys/bus/counter/devices/counterX/signalY/synchronous_mode_component_id
/sys/bus/counter/devices/counterX/signalY/frequency_component_id

Defined on file *sysfs-bus-counter*

Read-only attribute that indicates the component ID of the respective extension or Synapse.

/sys/bus/counter/devices/counterX/countY/capture

Defined on file *sysfs-bus-counter*

Historical capture of the Count Y count data.

/sys/bus/counter/devices/counterX/countY/ceiling

Defined on file *sysfs-bus-counter*

Count value ceiling for Count Y. This is the upper limit for the respective counter.

/sys/bus/counter/devices/counterX/countY/count

Defined on file *sysfs-bus-counter*

Count data of Count Y represented as a string.

/sys/bus/counter/devices/counterX/countY/count_mode

Defined on file *sysfs-bus-counter*

Count mode for channel Y. The ceiling and floor values for Count Y are used by the count mode where required. The following count modes are available:

normal:
Counting is continuous in either direction.

range limit:
An upper or lower limit is set, mimicking limit switches in the mechanical counterpart. The upper limit is set to the Count Y ceiling value, while the lower limit is set to the Count Y floor value. The counter freezes at count = ceiling when counting up, and at count = floor when counting down. At either of these limits, the counting is resumed only when the count direction is reversed.

non-recycle:
The counter is disabled whenever a counter overflow or underflow takes place. The counter is re-enabled when a new count value is loaded to the counter via a preset operation or direct write.

modulo-n:
A count value boundary is set between the Count Y floor value and the Count Y ceiling value. The counter is reset to the Count Y floor value at count = ceiling when counting up, while the counter is set to the Count Y ceiling value at count = floor when counting down; the counter does not freeze at the boundary points, but counts continuously throughout.

interrupt on terminal count:
The output signal is initially low, and will remain low until the counter reaches zero. The output signal then goes high and remains high until a new preset value is set.

hardware retriggerable one-shot:
The output signal is initially high. The output signal will go low by a trigger input signal, and will remain low until the counter reaches zero. The output will then go high and remain high until the next trigger. A trigger results in loading the counter to the preset value and setting the output signal low, thus starting the one-shot pulse.

rate generator:
The output signal is initially high. When the counter has decremented to 1, the output signal goes low for one clock pulse. The output signal then goes high again, the counter is reloaded to the preset value, and the process repeats in a periodic manner as such.

square wave mode:
The output signal is initially high.

If the initial count is even, the counter is decremented by two on succeeding clock pulses. When the count expires, the output signal changes value and the counter is reloaded to the preset value. The process repeats in periodic manner as such.

If the initial count is odd, the initial count minus one (an even number) is loaded and then is decremented by two on succeeding clock pulses. One clock pulse after the count expires, the output signal goes low and the counter is reloaded to the preset value minus one. Succeeding clock pulses decrement the count by two. When the count expires, the output goes high again and the counter is reloaded to the preset value minus one. The process repeats in a periodic manner as such.

software triggered strobe:
The output signal is initially high. When the count expires, the output will go low for one clock pulse and then go high again. The counting sequence is ”triggered” by setting the preset value.

hardware triggered strobe:
The output signal is initially high. Counting is started by a trigger input signal. When the count expires, the output signal will go low for one clock pulse and then go high again. A trigger results in loading the counter to the preset value.

/sys/bus/counter/devices/counterX/countY/count_mode_available
/sys/bus/counter/devices/counterX/countY/error_noise_available
/sys/bus/counter/devices/counterX/countY/function_available
/sys/bus/counter/devices/counterX/countY/prescaler_available
/sys/bus/counter/devices/counterX/countY/signalZ_action_available

Defined on file *sysfs-bus-counter*

Discrete set of available values for the respective Count Y configuration are listed in this file. Values are delimited by newline characters.

/sys/bus/counter/devices/counterX/countY/direction

Defined on file *sysfs-bus-counter*

Read-only attribute that indicates the count direction of Count Y. Two count directions are available: forward and backward.

Some counter devices are able to determine the direction of their counting. For example, quadrature encoding counters can determine the direction of movement by evaluating the leading phase of the respective A and B quadrature encoding signals. This attribute exposes such count directions.

/sys/bus/counter/devices/counterX/countY/enable

Defined on file *sysfs-bus-counter*

Whether channel Y counter is enabled. Valid attribute values are boolean.

This attribute is intended to serve as a pause/unpause mechanism for Count Y. Suppose a counter device is used to count the total movement of a conveyor belt: this attribute allows an operator to temporarily pause the counter, service the conveyor belt, and then finally unpause the counter to continue where it had left off.

/sys/bus/counter/devices/counterX/countY/error_noise

Defined on file *sysfs-bus-counter*

Read-only attribute that indicates whether excessive noise is present at the channel Y counter inputs.

/sys/bus/counter/devices/counterX/countY/floor

Defined on file *sysfs-bus-counter*

Count value floor for Count Y. This is the lower limit for the respective counter.

/sys/bus/counter/devices/counterX/countY/function

Defined on file *sysfs-bus-counter*

Count function mode of Count Y; count function evaluation is triggered by conditions specified by the Count Y signalZ_action attributes. The following count functions are available:

increase:
Accumulated count is incremented.

decrease:
Accumulated count is decremented.

pulse-direction:
Rising edges on signal A updates the respective count. The input level of signal B determines direction.

quadrature x1 a:
If direction is forward, rising edges on quadrature pair signal A updates the respective count; if the direction is backward, falling edges on quadrature pair signal A updates the respective count. Quadrature encoding determines the direction.

quadrature x1 b:
If direction is forward, rising edges on quadrature pair signal B updates the respective count; if the direction is backward, falling edges on quadrature pair signal B updates the respective count. Quadrature encoding determines the direction.

quadrature x2 a:
Any state transition on quadrature pair signal A updates the respective count. Quadrature encoding determines the direction.

quadrature x2 b:
Any state transition on quadrature pair signal B updates the respective count. Quadrature encoding determines the direction.

quadrature x4:
Any state transition on either quadrature pair signals updates the respective count. Quadrature encoding determines the direction.

/sys/bus/counter/devices/counterX/countY/name

Defined on file *sysfs-bus-counter*

Read-only attribute that indicates the device-specific name of Count Y. If possible, this should match the name of the respective channel as it appears in the device datasheet.

/sys/bus/counter/devices/counterX/countY/num_overflows

Defined on file *sysfs-bus-counter*

This attribute indicates the number of overflows of count Y.

/sys/bus/counter/devices/counterX/countY/prescaler

Defined on file *sysfs-bus-counter*

Configure the prescaler value associated with Count Y. On the FlexTimer, the counter clock source passes through a prescaler (i.e. a counter). This acts like a clock divider.

/sys/bus/counter/devices/counterX/countY/preset

Defined on file *sysfs-bus-counter*

If the counter device supports preset registers -- registers used to load counter channels to a set count upon device-defined preset operation trigger events -- the preset count for channel Y is provided by this attribute.

/sys/bus/counter/devices/counterX/countY/preset_enable

Defined on file *sysfs-bus-counter*

Whether channel Y counter preset operation is enabled. Valid attribute values are boolean.

/sys/bus/counter/devices/counterX/countY/signalZ_action

Defined on file *sysfs-bus-counter*

Action mode of Count Y for Signal Z. This attribute indicates the condition of Signal Z that triggers the count function evaluation for Count Y. The following action modes are available:

none:
Signal does not trigger the count function. In Pulse-Direction count function mode, this Signal is evaluated as Direction.

rising edge:
Low state transitions to high state.

falling edge:
High state transitions to low state.

both edges:
Any state transition.

/sys/bus/counter/devices/counterX/countY/spike_filter_ns

Defined on file *sysfs-bus-counter*

If the counter device supports programmable spike filter this attribute indicates the value in nanoseconds where noise pulses shorter or equal to configured value are ignored. Value 0 means filter is disabled.

/sys/bus/counter/devices/counterX/events_queue_size

Defined on file *sysfs-bus-counter*

Size of the Counter events queue in number of struct counter_event data structures. The number of elements will be rounded-up to a power of 2.

/sys/bus/counter/devices/counterX/external_input_phase_clock_select

Defined on file *sysfs-bus-counter*

Selects the external clock pin for phase counting mode of Counter X.

MTCLKA-MTCLKB:
MTCLKA and MTCLKB pins are selected for the external phase clock.

MTCLKC-MTCLKD:
MTCLKC and MTCLKD pins are selected for the external phase clock.

/sys/bus/counter/devices/counterX/external_input_phase_clock_select_available

Defined on file *sysfs-bus-counter*

Discrete set of available values for the respective device configuration are listed in this file.

/sys/bus/counter/devices/counterX/name

Defined on file *sysfs-bus-counter*

Read-only attribute that indicates the device-specific name of the Counter. This should match the name of the device as it appears in its respective datasheet.

/sys/bus/counter/devices/counterX/num_counts

Defined on file *sysfs-bus-counter*

Read-only attribute that indicates the total number of Counts belonging to the Counter.

/sys/bus/counter/devices/counterX/num_signals

Defined on file *sysfs-bus-counter*

Read-only attribute that indicates the total number of Signals belonging to the Counter.

/sys/bus/counter/devices/counterX/signalY/cable_fault

Defined on file *sysfs-bus-counter*

Read-only attribute that indicates whether a differential encoder cable fault (not connected or loose wires) is detected for the respective channel of Signal Y. Valid attribute values are boolean. Detection must first be enabled via the corresponding cable_fault_enable attribute.

/sys/bus/counter/devices/counterX/signalY/cable_fault_enable

Defined on file *sysfs-bus-counter*

Whether detection of differential encoder cable faults for the respective channel of Signal Y is enabled. Valid attribute values are boolean.

/sys/bus/counter/devices/counterX/signalY/filter_clock_prescaler

Defined on file *sysfs-bus-counter*

Filter clock factor for input Signal Y. This prescaler value affects the inputs of both quadrature pair signals.

/sys/bus/counter/devices/counterX/signalY/frequency

Defined on file *sysfs-bus-counter*

Read-only attribute that indicates the signal Y frequency, in Hz.

/sys/bus/counter/devices/counterX/signalY/index_polarity

Defined on file *sysfs-bus-counter*

Active level of index input Signal Y; irrelevant in non-synchronous load mode.

/sys/bus/counter/devices/counterX/signalY/index_polarity_available
/sys/bus/counter/devices/counterX/signalY/synchronous_mode_available

Defined on file *sysfs-bus-counter*

Discrete set of available values for the respective Signal Y configuration are listed in this file.

/sys/bus/counter/devices/counterX/signalY/name

Defined on file *sysfs-bus-counter*

Read-only attribute that indicates the device-specific name of Signal Y. If possible, this should match the name of the respective signal as it appears in the device datasheet.

/sys/bus/counter/devices/counterX/signalY/polarity

Defined on file *sysfs-bus-counter*

Active level of Signal Y. The following polarity values are available:

positive:
Signal high state considered active level (rising edge).

negative:
Signal low state considered active level (falling edge).

/sys/bus/counter/devices/counterX/signalY/signal

Defined on file *sysfs-bus-counter*

Signal level state of Signal Y. The following signal level states are available:

low:
Low level state.

high:
High level state.

/sys/bus/counter/devices/counterX/signalY/synchronous_mode

Defined on file *sysfs-bus-counter*

Configure the counter associated with Signal Y for non-synchronous or synchronous load mode. Synchronous load mode cannot be selected in non-quadrature (Pulse-Direction) clock mode.

non-synchronous:
A logic low level is the active level at this index input. The index function (as enabled via preset_enable) is performed directly on the active level of the index input.

synchronous:
Intended for interfacing with encoder Index output in quadrature clock mode. The active level is configured via index_polarity. The index function (as enabled via preset_enable) is performed synchronously with the quadrature clock on the active level of the index input.

/sys/bus/css/devices/.../driver_override

Defined on file *sysfs-bus-css*

This file allows the driver for a device to be specified. When specified, only a driver with a name matching the value written to driver_override will have an opportunity to bind to the device. The override is specified by writing a string to the driver_override file (echo vfio-ccw > driver_override) and may be cleared with an empty string (echo > driver_override). This returns the device to standard matching rules binding. Writing to driver_override does not automatically unbind the device from its current driver or make any attempt to automatically load the specified driver. If no driver with a matching name is currently loaded in the kernel, the device will not bind to any driver. This also allows devices to opt-out of driver binding using a driver_override name such as "none". Only a single driver may be specified in the override, there is no support for parsing delimiters.

Note that unlike the mechanism of the same name for pci, this file does not allow to override basic matching rules. I.e., the driver must still match the subchannel type of the device.

/sys/bus/css/devices/.../modalias

Defined on file *sysfs-bus-css*

Contains the module alias as reported with uevents. It is of the format css:t<type> and present for all subchannel types.

/sys/bus/css/devices/.../type

Defined on file *sysfs-bus-css*

Contains the subchannel type, as reported by the hardware. This attribute is present for all subchannel types.

/sys/bus/css/drivers/io_subchannel/.../chpids

Defined on file *sysfs-bus-css*

Contains the ids of the channel paths used by this subchannel, as reported by the channel subsystem during subchannel recognition.

Note: This is an I/O-subchannel specific attribute.

Users: s390-tools, HAL

/sys/bus/css/drivers/io_subchannel/.../pimpampom

Defined on file *sysfs-bus-css*

Contains the PIM/PAM/POM values, as reported by the channel subsystem when last queried by the common I/O layer (this implies that this attribute is not necessarily in sync with the values current in the channel subsystem).

Note: This is an I/O-subchannel specific attribute.

Users: s390-tools, HAL

/sys/bus/cxl/devices/*/devtype

Defined on file *sysfs-bus-cxl*

(RO) CXL device objects export the devtype attribute which mirrors the same value communicated in the DEVTYPE environment variable for uevents for devices on the "cxl" bus.

/sys/bus/cxl/devices/*/modalias

Defined on file *sysfs-bus-cxl*

(RO) CXL device objects export the modalias attribute which mirrors the same value communicated in the MODALIAS environment variable for uevents for devices on the "cxl" bus.

/sys/bus/cxl/devices/decoderX.Y

Defined on file *sysfs-bus-cxl*

(RO) CXL decoder objects are enumerated from either a platform firmware description, or a CXL HDM decoder register set in a PCIe device (see CXL 2.0 section 8.2.5.12 CXL HDM Decoder Capability Structure). The 'X' in decoderX.Y represents the cxl_port container of this decoder, and 'Y' represents the instance id of a given decoder resource.

/sys/bus/cxl/devices/decoderX.Y/cap_{pmem,ram,type2,type3}

Defined on file *sysfs-bus-cxl*

(RO) When a CXL decoder is of devtype "cxl_decoder_root", it represents a fixed memory window identified by platform firmware. A fixed window may only support a subset of memory types. The 'cap_*' attributes indicate whether persistent memory, volatile memory, accelerator memory, and / or expander memory may be mapped behind this decoder's memory window.

/sys/bus/cxl/devices/decoderX.Y/create_{pmem,ram}_region

Defined on file *sysfs-bus-cxl*

(RW) Write a string in the form 'regionZ' to start the process of defining a new persistent, or volatile memory region (interleave-set) within the decode range bounded by root decoder 'decoderX.Y'. The value written must match the current value returned from reading this attribute. An atomic compare exchange operation is done on write to assign the requested id to a region and allocate the region-id for the next creation attempt. EBUSY is returned if the region name written does not match the current cached value.

/sys/bus/cxl/devices/decoderX.Y/delete_region

Defined on file *sysfs-bus-cxl*

(WO) Write a string in the form 'regionZ' to delete that region, provided it is currently idle / not bound to a driver.

/sys/bus/cxl/devices/decoderX.Y/dpa_resource

Defined on file *sysfs-bus-cxl*

(RO) When a CXL decoder is of devtype "cxl_decoder_endpoint", and its 'dpa_size' attribute is non-zero, this attribute indicates the device physical address (DPA) base address of the allocation.

/sys/bus/cxl/devices/decoderX.Y/dpa_size

Defined on file *sysfs-bus-cxl*

(RW) When a CXL decoder is of devtype "cxl_decoder_endpoint" it translates from a host physical address range, to a device local address range. The range, base address plus length in bytes, of DPA allocated to this decoder is conveyed in these 2 attributes. Allocations can be mutated as long as the decoder is in the disabled state. A write to 'dpa_size' releases the previous DPA allocation and then attempts to allocate from the free capacity in the device partition referred to by 'decoderX.Y/mode'. Allocate and free requests can only be performed on the highest instance number disabled decoder with non-zero size. I.e. allocations are enforced to occur in increasing 'decoderX.Y/id' order and frees are enforced to occur in decreasing 'decoderX.Y/id' order.

/sys/bus/cxl/devices/decoderX.Y/interleave_granularity

Defined on file *sysfs-bus-cxl*

(RO) The number of consecutive bytes of host physical address space this decoder claims at address N before the decode rotates to the next target in the interleave at address N + interleave_granularity (assuming N is aligned to interleave_granularity).

/sys/bus/cxl/devices/decoderX.Y/interleave_ways

Defined on file *sysfs-bus-cxl*

(RO) The number of targets across which this decoder's host physical address (HPA) memory range is interleaved. The device maps every Nth block of HPA (of size == 'interleave_granularity') to consecutive DPA addresses. The decoder's position in the interleave is

determined by the device's (endpoint or switch) switch ancestry. For root decoders their interleave is specified by platform firmware and they only specify a downstream target order for host bridges.

/sys/bus/cxl/devices/decoderX.Y/locked

Defined on file *sysfs-bus-cxl*

(RO) CXL HDM decoders have the capability to lock the configuration until the next device reset. For decoders of devtype "cxl_decoder_root" there is no standard facility to unlock them. For decoders of devtype "cxl_decoder_switch" a secondary bus reset, of the PCIe bridge that provides the bus for this decoders uport, unlocks / resets the decoder.

/sys/bus/cxl/devices/decoderX.Y/mode

Defined on file *sysfs-bus-cxl*

(RW) When a CXL decoder is of devtype "cxl_decoder_endpoint" it translates from a host physical address range, to a device local address range. Device-local address ranges are further split into a 'ram' (volatile memory) range and 'pmem' (persistent memory) range. The 'mode' attribute emits one of 'ram', 'pmem', 'mixed', or 'none'. The 'mixed' indication is for error cases when a decoder straddles the volatile/persistent partition boundary, and 'none' indicates the decoder is not actively decoding, or no DPA allocation policy has been set.

'mode' can be written, when the decoder is in the 'disabled' state, with either 'ram' or 'pmem' to set the boundaries for the next allocation.

/sys/bus/cxl/devices/decoderX.Y/qos_class

Defined on file *sysfs-bus-cxl*

(RO) For CXL host platforms that support "QoS Telemmetry" this root-decoder-only attribute conveys a platform specific cookie that identifies a QoS performance class for the CXL Window. This class-id can be compared against a similar "qos_class" published for each memory-type that an endpoint supports. While it is not required that endpoints map their local memory-class to a matching platform class, mismatches are not recommended and there are platform specific side-effects that may result.

/sys/bus/cxl/devices/decoderX.Y/target_list

Defined on file *sysfs-bus-cxl*

(RO) Display a comma separated list of the current decoder target configuration. The list is ordered by the current configured interleave order of the decoder's dport instances. Each entry in the list is a dport id.

/sys/bus/cxl/devices/decoderX.Y/target_type

Defined on file *sysfs-bus-cxl*

(RO) When a CXL decoder is of devtype "cxl_decoder_switch", it can optionally decode either accelerator memory (type-2) or expander memory (type-3). The 'target_type' attribute indicates

the current setting which may dynamically change based on what memory regions are activated in this decode hierarchy.

/sys/bus/cxl/devices/decoderX.Y/{start,size}

Defined on file *sysfs-bus-cxl*

(RO) The 'start' and 'size' attributes together convey the physical address base and number of bytes mapped in the decoder's decode window. For decoders of devtype "cxl_decoder_root" the address range is fixed. For decoders of devtype "cxl_decoder_switch" the address is bounded by the decode range of the cxl_port ancestor of the decoder's cxl_port, and dynamically updates based on the active memory regions in that address space.

/sys/bus/cxl/devices/endpointX/CDAT

Defined on file *sysfs-bus-cxl*

(RO) If this sysfs entry is not present no DOE mailbox was found to support CDAT data. If it is present and the length of the data is 0 reading the CDAT data failed. Otherwise the CDAT data is reported.

/sys/bus/cxl/devices/memX/firmware/

Defined on file *sysfs-bus-cxl*

(RW) Firmware uploader mechanism. The different files under this directory can be used to upload and activate new firmware for CXL devices. The interfaces under this are documented in sysfs-class-firmware.

/sys/bus/cxl/devices/memX/firmware_version

Defined on file *sysfs-bus-cxl*

(RO) "FW Revision" string as reported by the Identify Memory Device Output Payload in the CXL-2.0 specification.

/sys/bus/cxl/devices/memX/numa_node

Defined on file *sysfs-bus-cxl*

(RO) If NUMA is enabled and the platform has affinitized the host PCI device for this memory device, emit the CPU node affinity for this device.

/sys/bus/cxl/devices/memX/pmem/qos_class

Defined on file *sysfs-bus-cxl*

(RO) For CXL host platforms that support "QoS Telemmetry" this attribute conveys a comma delimited list of platform specific cookies that identifies a QoS performance class for the persistent partition of the CXL mem device. These class-ids can be compared against a similar "qos_class"

published for a root decoder. While it is not required that the endpoints map their local memory-class to a matching platform class, mismatches are not recommended and there are platform specific performance related side-effects that may result. First class-id is displayed.

/sys/bus/cxl/devices/memX/pmem/size

Defined on file *sysfs-bus-cxl*

(RO) "Persistent Only Capacity" as bytes. Represents the identically named field in the Identify Memory Device Output Payload in the CXL-2.0 specification.

/sys/bus/cxl/devices/memX/ram/qos_class

Defined on file *sysfs-bus-cxl*

(RO) For CXL host platforms that support "QoS Telemmetry" this attribute conveys a comma delimited list of platform specific cookies that identifies a QoS performance class for the volatile partition of the CXL mem device. These class-ids can be compared against a similar "qos_class" published for a root decoder. While it is not required that the endpoints map their local memory-class to a matching platform class, mismatches are not recommended and there are platform specific performance related side-effects that may result. First class-id is displayed.

/sys/bus/cxl/devices/memX/ram/size

Defined on file *sysfs-bus-cxl*

(RO) "Volatile Only Capacity" as bytes. Represents the identically named field in the Identify Memory Device Output Payload in the CXL-2.0 specification.

/sys/bus/cxl/devices/memX/security/sanitize

Defined on file *sysfs-bus-cxl*

(WO) Write a boolean 'true' string value to this attribute to sanitize the device to securely re-purpose or decommission it. This is done by ensuring that all user data and meta-data, whether it resides in persistent capacity, volatile capacity, or the LSA, is made permanently unavailable by whatever means is appropriate for the media type. This functionality requires the device to be disabled, that is, not actively decoding any HPA ranges. This permits avoiding explicit global CPU cache management, relying instead for it to be done when a region transitions between software programmed and hardware committed states. If this file is not present, then there is no hardware support for the operation.

What /sys/bus/cxl/devices/memX/security/erase (WO) Write a boolean 'true' string value to this attribute to secure erase user data by changing the media encryption keys for all user data areas of the device. This functionality requires the device to be disabled, that is, not actively decoding any HPA ranges. This permits avoiding explicit global CPU cache management, relying instead for it to be done when a region transitions between software programmed and hardware committed states. If this file is not present, then there is no hardware support for the operation.

/sys/bus/cxl/devices/memX/security/state

Defined on file *sysfs-bus-cxl*

(RO) Reading this file will display the CXL security state for that device. Such states can be: 'disabled', 'sanitize', when a sanitization is currently underway; or those available only for persistent memory: 'locked', 'unlocked' or 'frozen'. This sysfs entry is select/poll capable from userspace to notify upon completion of a sanitize operation.

/sys/bus/cxl/devices/memX/serial

Defined on file *sysfs-bus-cxl*

(RO) 64-bit serial number per the PCIe Device Serial Number capability. Mandatory for CXL devices, see CXL 2.0 8.1.12.2 Memory Device PCIe Capabilities and Extended Capabilities.

/sys/bus/cxl/devices/memX/trigger_poison_list

Defined on file *sysfs-bus-cxl*

(WO) When a boolean 'true' is written to this attribute the memdev driver retrieves the poison list from the device. The list consists of addresses that are poisoned, or would result in poison if accessed, and the source of the poison. This attribute is only visible for devices supporting the capability. The retrieved errors are logged as kernel events when cxl_poison event tracing is enabled.

/sys/bus/cxl/devices/portX/decoders_committed

Defined on file *sysfs-bus-cxl*

(RO) A memory device is considered active when any of its decoders are in the ”committed” state (See CXL 3.0 8.2.4.19.7 CXL HDM Decoder n Control Register). Hotplug and destructive operations like ”sanitize” are blocked while device is actively decoding a Host Physical Address range. Note that this number may be elevated without any regionX objects active or even enumerated, as this may be due to decoders established by platform firwmare or a previous kernel (kexec).

/sys/bus/cxl/devices/portX/dportY

Defined on file *sysfs-bus-cxl*

(RO) CXL port objects are enumerated from either a platform firmware device (ACPI0017 and ACPI0016) or PCIe switch upstream port with CXL component registers. The 'dportY' symlink identifies one or more downstream ports that the upstream port may target in its decode of CXL memory resources. The 'Y' integer reflects the hardware port unique-id used in the hardware decoder target list.

/sys/bus/cxl/devices/portX/uport

Defined on file *sysfs-bus-cxl*

(RO) CXL port objects are enumerated from either a platform firmware device (ACPI0017 and ACPI0016) or PCIe switch upstream port with CXL component registers. The 'uport' symlink connects the CXL portX object to the device that published the CXL port capability.

/sys/bus/cxl/devices/regionZ/commit

Defined on file *sysfs-bus-cxl*

(RW) Write a boolean 'true' string value to this attribute to trigger the region to transition from the software programmed state to the actively decoding in hardware state. The commit operation in addition to validating that the region is in proper configured state, validates that the decoders are being committed in spec mandated order (last committed decoder id + 1), and checks that the hardware accepts the commit request. Reading this value indicates whether the region is committed or not.

/sys/bus/cxl/devices/regionZ/interleave_granularity

Defined on file *sysfs-bus-cxl*

(RW) Set the number of consecutive bytes each device in the interleave set will claim. The possible interleave granularity values are determined by the CXL spec and the participating devices.

/sys/bus/cxl/devices/regionZ/interleave_ways

Defined on file *sysfs-bus-cxl*

(RW) Configures the number of devices participating in the region is set by writing this value. Each device will provide 1/interleave_ways of storage for the region.

/sys/bus/cxl/devices/regionZ/mode

Defined on file *sysfs-bus-cxl*

(RO) The mode of a region is established at region creation time and dictates the mode of the endpoint decoder that comprise the region. For more details on the possible modes see /sys/bus/cxl/devices/decoderX.Y/mode

/sys/bus/cxl/devices/regionZ/resource

Defined on file *sysfs-bus-cxl*

(RO) A region is a contiguous partition of a CXL root decoder address space. Region capacity is allocated by writing to the size attribute, the resulting physical address space determined by the driver is reflected here. It is therefore not useful to read this before writing a value to the size attribute.

/sys/bus/cxl/devices/regionZ/size

Defined on file *sysfs-bus-cxl*

(RW) System physical address space to be consumed by the region. When written trigger the driver to allocate space out of the parent root decoder's address space. When read the size of the address space is reported and should match the span of the region's resource attribute. Size shall be set after the interleave configuration parameters. Once set it cannot be changed, only

freed by writing 0. The kernel makes no guarantees that data is maintained over an address space freeing event, and there is no guarantee that a free followed by an allocate results in the same address being allocated.

/sys/bus/cxl/devices/regionZ/target[0..N]

Defined on file *sysfs-bus-cxl*

(RW) Write an endpoint decoder object name to 'targetX' where X is the intended position of the endpoint device in the region interleave and N is the 'interleave_ways' setting for the region. ENXIO is returned if the write results in an impossible to map decode scenario, like the endpoint is unreachable at that position relative to the root decoder interleave. EBUSY is returned if the position in the region is already occupied, or if the region is not in a state to accept interleave configuration changes. EINVAL is returned if the object name is not an endpoint decoder. Once all positions have been successfully written a final validation for decode conflicts is performed before activating the region.

/sys/bus/cxl/devices/regionZ/uuid

Defined on file *sysfs-bus-cxl*

(RW) Write a unique identifier for the region. This field must be set for persistent regions and it must not conflict with the UUID of another region. For volatile ram regions this attribute is a read-only empty string.

/sys/bus/cxl/devices/{port,endpoint}X/parent_dport

Defined on file *sysfs-bus-cxl*

(RO) CXL port objects are instantiated for each upstream port in a CXL/PCIe switch, and for each endpoint to map the corresponding memory device into the CXL port hierarchy. When a descendant CXL port (switch or endpoint) is enumerated it is useful to know which 'dport' object in the parent CXL port routes to this descendant. The 'parent_dport' symlink points to the device representing the downstream port of a CXL switch that routes to {port,endpoint}X.

/sys/bus/cxl/flush

Defined on file *sysfs-bus-cxl*

(WO) If userspace manually unbinds a port the kernel schedules all descendant memdevs for unbind. Writing '1' to this attribute flushes that work.

/sys/bus/dfl/devices/dfl_dev.X/feature_id

Defined on file *sysfs-bus-dfl*

Read-only. It returns feature identifier local to its DFL FIU type.

Format: 0x%x

/sys/bus/dfl/devices/dfl_dev.X/fec_mode

Defined on file *sysfs-bus-dfl-devices-n3000-nios*

Read-only. Returns the FEC mode of the 25G links of the ethernet retimers configured by Nios firmware. "rs" for Reed Solomon FEC, "kr" for Fire Code FEC, "no" for NO FEC. "not supported" if the FEC mode setting is not supported, this happens when the Nios firmware version major < 3, or no link is configured to 25G. Format: string

/sys/bus/dfl/devices/dfl_dev.X/infX_cal_fail

Defined on file *sysfs-bus-dfl-devices-emif*

Read-only. It indicates if the calibration failed on this memory interface. "1" for calibration failure, "0" for OK. Format: %u

/sys/bus/dfl/devices/dfl_dev.X/infX_clear

Defined on file *sysfs-bus-dfl-devices-emif*

Write-only. Writing "1" to this file will zero out all memory data in this memory interface. Writing of other values is invalid. Format: %u

/sys/bus/dfl/devices/dfl_dev.X/infX_init_done

Defined on file *sysfs-bus-dfl-devices-emif*

Read-only. It indicates if the initialization completed on this memory interface. "1" for initialization complete, "0" for not yet. Format: %u

/sys/bus/dfl/devices/dfl_dev.X/nios_fw_version

Defined on file *sysfs-bus-dfl-devices-n3000-nios*

Read-only. Returns the version of the Nios firmware in the FPGA. Its format is "major.minor.patch". Format: %x.%x.%x

/sys/bus/dfl/devices/dfl_dev.X/retimer_A_mode

Defined on file *sysfs-bus-dfl-devices-n3000-nios*

Read-only. Returns the enumeration value of the working mode of the retimer A configured by the Nios firmware. The value is read out from shared registers filled by the Nios firmware. Now the values could be:

- "0": Reset
- "1": 4x10G
- "2": 4x25G
- "3": 2x25G
- "4": 2x25G+2x10G
- "5": 1x25G

If the Nios firmware is updated in future to support more retimer modes, more enumeration value is expected. Format: 0x%x

/sys/bus/dfl/devices/dfl_dev.X/retimer_B_mode

Defined on file *sysfs-bus-dfl-devices-n3000-nios*

Read-only. Returns the enumeration value of the working mode of the retimer B configured by the Nios firmware. The value format is the same as retimer_A_mode.

/sys/bus/dfl/devices/dfl_dev.X/type

Defined on file *sysfs-bus-dfl*

Read-only. It returns type of DFL FIU of the device. Now DFL supports 2 FIU types, 0 for FME, 1 for PORT.

Format: 0x%x

/sys/bus/drivers/corsair/<dev>/current_profile

Defined on file *sysfs-driver-hid-corsair*

Get/set the current selected profile. Values are from 1 to 3.

/sys/bus/drivers/corsair/<dev>/macro_mode

Defined on file *sysfs-driver-hid-corsair*

Get/set the current playback mode. "SW" for software mode where G-keys triggers their regular key codes. "HW" for hardware playback mode where the G-keys play their macro from the on-board memory.

/sys/bus/event_source/devices/<dev>/caps

Defined on file *sysfs-bus-event_source-devices-caps*

Attribute group to describe the capabilities exposed for a particular pmu. Each attribute of this group can expose information specific to a PMU, say pmu_name, so that userspace can understand some of the feature which the platform specific PMU supports.

One of the example available capability in supported platform like Intel is pmu_name, which exposes underlying CPU name known to the PMU driver.

Example output in powerpc: grep . /sys/bus/event_source/devices/cpu/caps/* /sys/bus/event_source/devices/cpu/caps/pmu_name:POWER9

The "branch_counter_nr" in the supported platform exposes the maximum number of counters which can be shown in the u64 counters of PERF_SAMPLE_BRANCH_COUNTERS, while the "branch_counter_width" exposes the width of each counter. Both of them can be used by the perf tool to parse the logged counters in each branch.

/sys/bus/event_source/devices/<dev>/format

Defined on file *sysfs-bus-event_source-devices-format*

Attribute group to describe the magic bits that go into perf_event_attr::config[012] for a particular pmu. Each attribute of this group defines the 'hardware' bitmask we want to export, so that userspace can deal with sane name/value pairs.

Userspace must be prepared for the possibility that attributes define overlapping bit ranges. For example:

```
attr1 = 'config:0-23'
attr2 = 'config:0-7'
attr3 = 'config:12-35'
```

Example: 'config1:1,6-10,44' Defines contents of attribute that occupies bits 1,6-10,44 of perf_event_attr::config1.

/sys/bus/event_source/devices/<pmu>/events/<event>

Defined on file *sysfs-bus-event_source-devices-events*

Per-pmu performance monitoring events specific to the running system

Each file (except for some of those with a '.' in them, '.unit' and '.scale') in the 'events' directory describes a single performance monitoring event supported by the <pmu>. The name of the file is the name of the event.

File contents:

> <term>[=<value>][,<term>[=<value>]]...

Where <term> is one of the terms listed under /sys/bus/event_source/devices/<pmu>/format/ and <value> is a number is base-16 format with a '0x' prefix (lowercase only). If a <term> is specified alone (without an assigned value), it is implied that 0x1 is assigned to that <term>.

Examples (each of these lines would be in a separate file):

> event=0x2abc event=0x423,inv,cmask=0x3 domain=0x1,offset=0x8,starting_index=0xffff domain=0x1,offset=0x8,core=?

Each of the assignments indicates a value to be assigned to a particular set of bits (as defined by the format file corresponding to the <term>) in the perf_event structure passed to the perf_open syscall.

In the case of the last example, a value replacing "?" would need to be provided by the user selecting the particular event. This is referred to as "event parameterization". Event parameters have the format 'param=?'.

/sys/bus/event_source/devices/<pmu>/events/<event>.scale

Defined on file *sysfs-bus-event_source-devices-events*

Perf event scaling factors

A string representing a floating point value expressed in scientific notation to be multiplied by the event count received from the kernel to match the unit specified in the <event>.unit file.

Example:

2.328306436538696289062 5e-10

This is provided to avoid performing floating point arithmetic in the kernel.

/sys/bus/event_source/devices/<pmu>/events/<event>.unit

Defined on file *sysfs-bus-event_source-devices-events*

Perf event units

A string specifying the English plural numerical unit that <event> (once multiplied by <event>.scale) represents.

Example:

Joules

/sys/bus/event_source/devices/dfl_fmeX/cpumask

Defined on file *sysfs-bus-event_source-devices-dfl_fme*

Read-only. This file always returns cpu which the PMU is bound for access to all fme pmu performance monitoring events.

/sys/bus/event_source/devices/dfl_fmeX/events

Defined on file *sysfs-bus-event_source-devices-dfl_fme*

Read-only. Attribute group to describe performance monitoring events specific to fme. Each attribute in this group describes a single performance monitoring event supported by this fme pmu. The name of the file is the name of the event. (See ABI/testing/sysfs-bus-event_source-devices-events).

All supported performance monitoring events are listed below.

Basic events (evtype=0x00):

```
clock = "event=0x00,evtype=0x00,portid=0xff"
```

Cache events (evtype=0x01):

```
cache_read_hit      = "event=0x00,evtype=0x01,portid=0xff"
cache_read_miss     = "event=0x01,evtype=0x01,portid=0xff"
cache_write_hit     = "event=0x02,evtype=0x01,portid=0xff"
cache_write_miss    = "event=0x03,evtype=0x01,portid=0xff"
cache_hold_request  = "event=0x05,evtype=0x01,portid=0xff"
cache_data_write_port_contention =
                      "event=0x06,evtype=0x01,portid=0xff"
cache_tag_write_port_contention =
                      "event=0x07,evtype=0x01,portid=0xff"
cache_tx_req_stall  = "event=0x08,evtype=0x01,portid=0xff"
cache_rx_req_stall  = "event=0x09,evtype=0x01,portid=0xff"
cache_eviction      = "event=0x0a,evtype=0x01,portid=0xff"
```

Fabric events (evtype=0x02):

```
fab_pcie0_read       = "event=0x00,evtype=0x02,portid=0xff"
fab_pcie0_write      = "event=0x01,evtype=0x02,portid=0xff"
fab_pcie1_read       = "event=0x02,evtype=0x02,portid=0xff"
fab_pcie1_write      = "event=0x03,evtype=0x02,portid=0xff"
fab_upi_read         = "event=0x04,evtype=0x02,portid=0xff"
fab_upi_write        = "event=0x05,evtype=0x02,portid=0xff"
fab_mmio_read        = "event=0x06,evtype=0x02,portid=0xff"
fab_mmio_write       = "event=0x07,evtype=0x02,portid=0xff"
fab_port_pcie0_read  = "event=0x00,evtype=0x02,portid=?"
fab_port_pcie0_write = "event=0x01,evtype=0x02,portid=?"
fab_port_pcie1_read  = "event=0x02,evtype=0x02,portid=?"
fab_port_pcie1_write = "event=0x03,evtype=0x02,portid=?"
fab_port_upi_read    = "event=0x04,evtype=0x02,portid=?"
fab_port_upi_write   = "event=0x05,evtype=0x02,portid=?"
fab_port_mmio_read   = "event=0x06,evtype=0x02,portid=?"
fab_port_mmio_write  = "event=0x07,evtype=0x02,portid=?"
```

VTD events (evtype=0x03):

```
vtd_port_read_transaction  = "event=0x00,evtype=0x03,portid=?"
vtd_port_write_transaction = "event=0x01,evtype=0x03,portid=?"
vtd_port_devtlb_read_hit   = "event=0x02,evtype=0x03,portid=?"
vtd_port_devtlb_write_hit  = "event=0x03,evtype=0x03,portid=?"
vtd_port_devtlb_4k_fill    = "event=0x04,evtype=0x03,portid=?"
vtd_port_devtlb_2m_fill    = "event=0x05,evtype=0x03,portid=?"
vtd_port_devtlb_1g_fill    = "event=0x06,evtype=0x03,portid=?"
```

VTD SIP events (evtype=0x04):

```
vtd_sip_iotlb_4k_hit  = "event=0x00,evtype=0x04,portid=0xff"
vtd_sip_iotlb_2m_hit  = "event=0x01,evtype=0x04,portid=0xff"
vtd_sip_iotlb_1g_hit  = "event=0x02,evtype=0x04,portid=0xff"
vtd_sip_slpwc_l3_hit  = "event=0x03,evtype=0x04,portid=0xff"
vtd_sip_slpwc_l4_hit  = "event=0x04,evtype=0x04,portid=0xff"
vtd_sip_rcc_hit       = "event=0x05,evtype=0x04,portid=0xff"
vtd_sip_iotlb_4k_miss = "event=0x06,evtype=0x04,portid=0xff"
vtd_sip_iotlb_2m_miss = "event=0x07,evtype=0x04,portid=0xff"
vtd_sip_iotlb_1g_miss = "event=0x08,evtype=0x04,portid=0xff"
vtd_sip_slpwc_l3_miss = "event=0x09,evtype=0x04,portid=0xff"
vtd_sip_slpwc_l4_miss = "event=0x0a,evtype=0x04,portid=0xff"
vtd_sip_rcc_miss      = "event=0x0b,evtype=0x04,portid=0xff"
```

/sys/bus/event_source/devices/dfl_fmeX/format

Defined on file *sysfs-bus-event_source-devices-dfl_fme*

Read-only. Attribute group to describe the magic bits that go into perf_event_attr.config for a particular pmu. (See ABI/testing/sysfs-bus-event_source-devices-format).

Each attribute under this group defines a bit range of the perf_event_attr.config. All supported attributes are listed below:

```
event  = "config:0-11"  - event ID
evtype = "config:12-15" - event type
portid = "config:16-23" - event source
```

For example:

```
fab_mmio_read = "event=0x06,evtype=0x02,portid=0xff"
```

It shows this fab_mmio_read is a fabric type (0x02) event with 0x06 local event id for overall monitoring (portid=0xff).

/sys/bus/event_source/devices/dmar*/cpumask

Defined on file *sysfs-bus-event_source-devices-iommu*

Read-only. This file always returns the CPU to which the IOMMU pmu is bound for access to all IOMMU pmu performance monitoring events.

/sys/bus/event_source/devices/dmar*/format

Defined on file *sysfs-bus-event_source-devices-iommu*

Read-only. Attribute group to describe the magic bits that go into perf_event_attr.config, perf_event_attr.config1 or perf_event_attr.config2 for the IOMMU pmu. (See also ABI/testing/sysfs-bus-event_source-devices-format).

Each attribute in this group defines a bit range in perf_event_attr.config, perf_event_attr.config1, or perf_event_attr.config2. All supported attributes are listed below (See the VT-d Spec 4.0 for possible attribute values):

```
event               = "config:0-27"   - event ID
event_group         = "config:28-31"  - event group ID

filter_requester_en = "config1:0"     - Enable Requester ID filter
filter_domain_en    = "config1:1"     - Enable Domain ID filter
filter_pasid_en     = "config1:2"     - Enable PASID filter
filter_ats_en       = "config1:3"     - Enable Address Type filter
filter_page_table_en= "config1:4"     - Enable Page Table Level filter
filter_requester_id = "config1:16-31" - Requester ID filter
filter_domain       = "config1:32-47" - Domain ID filter
filter_pasid        = "config2:0-21"  - PASID filter
filter_ats          = "config2:24-28" - Address Type filter
filter_page_table   = "config2:32-36" - Page Table Level filter
```

/sys/bus/event_source/devices/dsa*/cpumask

Defined on file *sysfs-bus-event_source-devices-dsa*

Read-only. This file always returns the cpu to which the IDXD DSA pmu is bound for access to all dsa pmu performance monitoring events.

/sys/bus/event_source/devices/dsa*/format

Defined on file *sysfs-bus-event_source-devices-dsa*

Read-only. Attribute group to describe the magic bits that go into perf_event_attr.config or perf_event_attr.config1 for the IDXD DSA pmu. (See also ABI/testing/sysfs-bus-event_source-devices-format).

Each attribute in this group defines a bit range in perf_event_attr.config or perf_event_attr.config1. All supported attributes are listed below (See the IDXD DSA Spec for possible attribute values):

```
event_category = "config:0-3"    - event category
event          = "config:4-31"   - event ID

filter_wq      = "config1:0-31"  - workqueue filter
filter_tc      = "config1:32-39" - traffic class filter
filter_pgsz    = "config1:40-43" - page size filter
filter_sz      = "config1:44-51" - transfer size filter
filter_eng     = "config1:52-59" - engine filter
```

/sys/bus/event_source/devices/hv_24x7/event_descs/<event-name>

Defined on file *sysfs-bus-event_source-devices-hv_24x7*

Provides the description of a particular event as provided by the firmware. If firmware does not provide a description, no file will be created.

Note that the event-name lacks the domain suffix appended for events in the events/ dir.

/sys/bus/event_source/devices/hv_24x7/event_long_descs/<event-name>

Defined on file *sysfs-bus-event_source-devices-hv_24x7*

Provides the "long" description of a particular event as provided by the firmware. If firmware does not provide a description, no file will be created.

Note that the event-name lacks the domain suffix appended for events in the events/ dir.

/sys/bus/event_source/devices/hv_24x7/format

Defined on file *sysfs-bus-event_source-devices-hv_24x7*

Read-only. Attribute group to describe the magic bits that go into perf_event_attr.config for a particular pmu. (See ABI/testing/sysfs-bus-event_source-devices-format).

Each attribute under this group defines a bit range of the perf_event_attr.config. All supported attributes are listed below:

```
chip = "config:16-31"
core  = "config:16-31"
domain = "config:0-3"
lpar = "config:0-15"
```

```
offset = "config:32-63"
vcpu = "config:16-31"
```

For example:

```
PM_PB_CYC =  "domain=1,offset=0x80,chip=?,lpar=0x0"
```

In this event, '?' after chip specifies that this value will be provided by user while running this event.

/sys/bus/event_source/devices/hv_24x7/interface/catalog

Defined on file *sysfs-bus-event_source-devices-hv_24x7*

Provides access to the binary "24x7 catalog" provided by the hypervisor on POWER7 and 8 systems. This catalog lists events available from the powerpc "hv_24x7" pmu. Its format is documented here: https://raw.githubusercontent.com/jmesmon/catalog-24x7/master/hv-24x7-catalog.h

/sys/bus/event_source/devices/hv_24x7/interface/catalog_length

Defined on file *sysfs-bus-event_source-devices-hv_24x7*

A number equal to the length in bytes of the catalog. This is also extractable from the provided binary "catalog" sysfs entry.

/sys/bus/event_source/devices/hv_24x7/interface/catalog_version

Defined on file *sysfs-bus-event_source-devices-hv_24x7*

Exposes the "version" field of the 24x7 catalog. This is also extractable from the provided binary "catalog" sysfs entry.

/sys/bus/event_source/devices/hv_gpci/format

Defined on file *sysfs-bus-event_source-devices-hv_gpci*

Read-only. Attribute group to describe the magic bits that go into perf_event_attr.config for a particular pmu. (See ABI/testing/sysfs-bus-event_source-devices-format).

Each attribute under this group defines a bit range of the perf_event_attr.config. All supported attributes are listed below:

```
counter_info_version  = "config:16-23"
length  = "config:24-31"
partition_id  = "config:32-63"
request = "config:0-31"
sibling_part_id = "config:32-63"
hw_chip_id = "config:32-63"
offset = "config:32-63"
phys_processor_idx = "config:32-63"
```

```
secondary_index = "config:0-15"
starting_index = "config:32-63"
```

For example:

```
processor_core_utilization_instructions_completed = "request=0x94,
                      phys_processor_idx=?,counter_info_version=0x8,
                      length=8,offset=0x18"
```

In this event, '?' after phys_processor_idx specifies this value this value will be provided by user while running this event.

/sys/bus/event_source/devices/hv_gpci/interface/collect_privileged

Defined on file *sysfs-bus-event_source-devices-hv_gpci*

'0' if the hypervisor is configured to forbid access to event counters being accumulated by other guests and to physical domain event counters.

'1' if that access is allowed.

/sys/bus/event_source/devices/hv_gpci/interface/expanded

Defined on file *sysfs-bus-event_source-devices-hv_gpci*

0 or 1. Indicates whether we have access to "EXPANDED" events (listed in arch/powerpc/perf/hv-gpci.h).

/sys/bus/event_source/devices/hv_gpci/interface/ga

Defined on file *sysfs-bus-event_source-devices-hv_gpci*

0 or 1. Indicates whether we have access to "GA" events (listed in arch/powerpc/perf/hv-gpci.h).

/sys/bus/event_source/devices/hv_gpci/interface/kernel_version

Defined on file *sysfs-bus-event_source-devices-hv_gpci*

A number indicating the latest version of the gpci interface that the kernel is aware of.

/sys/bus/event_source/devices/hv_gpci/interface/lab

Defined on file *sysfs-bus-event_source-devices-hv_gpci*

0 or 1. Indicates whether we have access to "LAB" events (listed in arch/powerpc/perf/hv-gpci.h).

/sys/bus/event_source/devices/hv_gpci/interface/version

Defined on file *sysfs-bus-event_source-devices-hv_gpci*

A number indicating the version of the gpci interface that the hypervisor reports supporting.

/sys/bus/event_source/devices/nmemX/cpumask

Defined on file *sysfs-bus-nvdimm*

(RO) This sysfs file exposes the cpumask which is designated to to retrieve nvdimm pmu event counter data.

/sys/bus/event_source/devices/nmemX/events

Defined on file *sysfs-bus-nvdimm*

(RO) Attribute group to describe performance monitoring events for the nvdimm memory device. Each attribute in this group describes a single performance monitoring event supported by this nvdimm pmu. The name of the file is the name of the event. (See ABI/testing/sysfs-bus-event_source-devices-events). A listing of the events supported by a given nvdimm provider type can be found in Documentation/driver-api/nvdimm/$provider.

/sys/bus/event_source/devices/nmemX/format

Defined on file *sysfs-bus-nvdimm*

(RO) Attribute group to describe the magic bits that go into perf_event_attr.config for a particular pmu. (See ABI/testing/sysfs-bus-event_source-devices-format).

Each attribute under this group defines a bit range of the perf_event_attr.config. Supported attribute is listed below:

```
event  = "config:0-4"  - event ID
```

For example:

```
ctl_res_cnt = "event=0x1"
```

/sys/bus/event_source/devices/uncore_*/alias

Defined on file *sysfs-bus-event_source-devices-uncore*

Read-only. An attribute to describe the alias name of the uncore PMU if an alias exists on some platforms. The 'perf(1)' tool should treat both names the same. They both can be used to access the uncore PMU.

Example:

$ cat /sys/devices/uncore_cha_2/alias uncore_type_0_2

/sys/bus/fcoe/

Defined on file *sysfs-bus-fcoe*

The FCoE bus. Attributes in this directory are control interfaces.

Attributes:

ctlr_create:
'FCoE Controller' instance creation interface. Writing an <ifname> to this file will allocate and populate sysfs with a fcoe_ctlr_device (ctlr_X). The user can then configure any per-port settings and finally write to the fcoe_ctlr_device's 'start' attribute to begin the kernel's discovery and login process.

ctlr_destroy:
'FCoE Controller' instance removal interface. Writing a fcoe_ctlr_device's sysfs name to this file will log the fcoe_ctlr_device out of the fabric or otherwise connected FCoE devices. It will also free all kernel memory allocated for this fcoe_ctlr_device and any structures associated with it, this includes the scsi_host.

/sys/bus/fcoe/devices/ctlr_X

Defined on file *sysfs-bus-fcoe*

'FCoE Controller' instances on the fcoe bus. The FCoE Controller now has a three stage creation process. 1) Write interface name to ctlr_create 2) Configure the FCoE Controller (ctlr_X) 3) Enable the FCoE Controller to begin discovery and login. The FCoE Controller is destroyed by writing its name, i.e. ctlr_X to the ctlr_delete file.

Attributes:

fcf_dev_loss_tmo:
Device loss timeout period (see below). Changing this value will change the dev_loss_tmo for all FCFs discovered by this controller.

mode:
Display or change the FCoE Controller's mode. Possible modes are 'Fabric' and 'VN2VN'. If a FCoE Controller is started in 'Fabric' mode then FIP FCF discovery is initiated and ultimately a fabric login is attempted. If a FCoE Controller is started in 'VN2VN' mode then FIP VN2VN discovery and login is performed. A FCoE Controller only supports one mode at a time.

enabled:
Whether an FCoE controller is enabled or disabled. 0 if disabled, 1 if enabled. Writing either 0 or 1 to this file will enable or disable the FCoE controller.

lesb/link_fail:
Link Error Status Block (LESB) link failure count.

lesb/vlink_fail:
Link Error Status Block (LESB) virtual link failure count.

lesb/miss_fka:
Link Error Status Block (LESB) missed FCoE Initialization Protocol (FIP) Keep-Alives (FKA).

lesb/symb_err:
Link Error Status Block (LESB) symbolic error count.

lesb/err_block:
Link Error Status Block (LESB) block error count.

lesb/fcs_error:
Link Error Status Block (LESB) Fibre Channel Services error count.

Notes: ctlr_X (global increment starting at 0)

/sys/bus/fcoe/devices/fcf_X

Defined on file *sysfs-bus-fcoe*

'FCoE FCF' instances on the fcoe bus. A FCF is a Fibre Channel Forwarder, which is a FCoE switch that can accept FCoE (Ethernet) packets, unpack them, and forward the embedded Fibre Channel frames into a FC fabric. It can also take outbound FC frames and pack them in Ethernet packets to be sent to their destination on the Ethernet segment.

Attributes:

fabric_name:
Identifies the fabric that the FCF services.

switch_name:
Identifies the FCF.

priority:
The switch's priority amongst other FCFs on the same fabric.

selected:
1 indicates that the switch has been selected for use; 0 indicates that the switch will not be used.

fc_map:
The Fibre Channel MAP

vfid:
The Virtual Fabric ID

mac:
The FCF's MAC address

fka_period:
The FIP Keep-Alive period

fabric_state: The internal kernel state

- ”Unknown” - Initialization value
- ”Disconnected” - No link to the FCF/fabric
- ”Connected” - Host is connected to the FCF
- ”Deleted” - FCF is being removed from the system

dev_loss_tmo: The device loss timeout period for this FCF.

Notes: A device loss infrastructure similar to the FC Transport's
is present in fcoe_sysfs. It is nice to have so that a link flapping adapter doesn't continually advance the count used to identify the discovered FCF. FCFs will exist in a ”Disconnected” state until either the timer expires and the FCF becomes ”Deleted” or the FCF is rediscovered and becomes ”Connected.”

Users: The first user of this interface will be the fcoeadm application, which is commonly packaged in the fcoe-utils package.

/sys/bus/fsi/devices/XX.XX.00:06/sbefifoX/timeout

Defined on file *sysfs-bus-fsi-devices-sbefifo*

Indicates whether or not this SBE device has experienced a timeout; i.e. the SBE did not respond within the time allotted by the driver. A value of 1 indicates that a timeout has occurred and no transfers have completed since the timeout. A value of 0 indicates that no timeout has occurred, or if one has, more recent transfers have completed successfully.

/sys/bus/fsl-mc/drivers/.../bind

Defined on file *sysfs-bus-fsl-mc*

Writing a device location to this file will cause the driver to attempt to bind to the device found at this location. The format for the location is Object.Id and is the same as found in /sys/bus/fsl-mc/devices/.

For example:

```
# echo dpni.2 > /sys/bus/fsl-mc/drivers/fsl_dpaa2_eth/bind
```

/sys/bus/fsl-mc/drivers/.../unbind

Defined on file *sysfs-bus-fsl-mc*

Writing a device location to this file will cause the driver to attempt to unbind from the device found at this location. The format for the location is Object.Id and is the same as found in /sys/bus/fsl-mc/devices/.

For example:

```
# echo dpni.2 > /sys/bus/fsl-mc/drivers/fsl_dpaa2_eth/unbind
```

/sys/bus/hid/devices/<bus>:<vid>:<pid>.<n>/speed

Defined on file *sysfs-driver-wacom*

The /sys/bus/hid/devices/<bus>:<vid>:<pid>.<n>/speed file controls reporting speed of Wacom bluetooth tablet. Reading from this file returns 1 if tablet reports in high speed mode or 0 otherwise. Writing to this file one of these values switches reporting speed.

/sys/bus/hid/devices/<bus>:<vid>:<pid>.<n>/wacom_led/button<n>_rawimg

Defined on file *sysfs-driver-wacom*

When writing a 1024 byte raw image in Wacom Intuos 4 interleaving format to the file, the image shows up on Button N of the device. The image is a 64x32 pixel 4-bit gray image. The 1024 byte binary is split up into 16x 64 byte chunks. Each 64 byte chunk encodes the image data for two consecutive lines on the display. The low nibble of each byte contains the first line, and the high nibble contains the second line. When the Wacom Intuos 4 is connected over Bluetooth, the image has to contain 256 bytes (64x32 px 1 bit colour). The format is also scrambled, like in the USB mode, and it can be summarized by converting:

```
76543210 into GECA6420.
HGFEDCBA      HFDB7531
```

/sys/bus/hid/devices/<bus>:<vid>:<pid>.<n>/wacom_led/buttons_luminance

Defined on file *sysfs-driver-wacom*

Writing to this file sets the overall luminance level (0..15) of all eight button OLED displays.

/sys/bus/hid/devices/<bus>:<vid>:<pid>.<n>/wacom_led/led

Defined on file *sysfs-driver-wacom*

Attribute group for control of the status LEDs and the OLEDs. This attribute group is only available for Intuos 4 M, L, and XL (with LEDs and OLEDs), Intuos 4 WL, Intuos 5 (LEDs only), Intuos Pro (LEDs only) and Cintiq 21UX2 and Cintiq 24HD (LEDs only). Therefore its presence implicitly signifies the presence of said LEDs and OLEDs on the tablet device.

/sys/bus/hid/devices/<bus>:<vid>:<pid>.<n>/wacom_led/status0_luminance

Defined on file *sysfs-driver-wacom*

<obsoleted by the LED class API now exported by the driver> Writing to this file sets the status LED luminance (1..127) when the stylus does not touch the tablet surface, and no button is pressed on the stylus. This luminance level is normally lower than the level when a button is pressed.

/sys/bus/hid/devices/<bus>:<vid>:<pid>.<n>/wacom_led/status1_luminance

Defined on file *sysfs-driver-wacom*

<obsoleted by the LED class API now exported by the driver> Writing to this file sets the status LED luminance (1..127) when the stylus touches the tablet surface, or any button is pressed on the stylus.

/sys/bus/hid/devices/<bus>:<vid>:<pid>.<n>/wacom_led/status_led0_select

Defined on file *sysfs-driver-wacom*

<obsoleted by the LED class API now exported by the driver> Writing to this file sets which one of the four (for Intuos 4 and Intuos 5) or of the right four (for Cintiq 21UX2 and Cintiq 24HD) status LEDs is active (0..3). The other three LEDs on the same side are always inactive.

/sys/bus/hid/devices/<bus>:<vid>:<pid>.<n>/wacom_led/status_led1_select

Defined on file *sysfs-driver-wacom*

<obsoleted by the LED class API now exported by the driver> Writing to this file sets which one of the left four (for Cintiq 21UX2 and Cintiq 24HD) status LEDs is active (0..3). The other three LEDs on the left are always inactive.

/sys/bus/hid/devices/<bus>:<vid>:<pid>.<n>/wacom_remote/<serial_number>/remote_mo

Defined on file *sysfs-driver-wacom*

<obsoleted by the LED class API now exported by the driver> Reading from this file reports the mode status of the remote as indicated by the LED lights on the device. If no reports have been received from the paired device, reading from this file will report '-1'. The mode is read-only and cannot be set through the driver.

/sys/bus/hid/devices/<bus>:<vid>:<pid>.<n>/wacom_remote/unpair_remote

Defined on file *sysfs-driver-wacom*

Writing the character sequence '*' followed by a newline to this file will delete all of the current pairings on the device. Other character sequences are reserved. This file is write only.

/sys/bus/hid/drivers/logitech-hidpp-device/<dev>/builtin_power_supply

Defined on file *sysfs-driver-hid-logitech-hidpp*

Presence of this file indicates that HID++ driver is capable of handling battery properties in the kernel. This way, upower can add a udev rule to decide whether or not it should use the internal unifying support or the generic kernel one.

/sys/bus/hid/drivers/logitech-hidpp-device/<dev>/range

Defined on file *sysfs-driver-hid-logitech-hidpp*

(RW) This attribute controls the amount of 'turn' permitted in Logitech G920 wheel. Reading from the file shows the current range of the steering wheel. Writing a value within the min and max boundary sets the range of the wheel.

/sys/bus/hid/drivers/logitech/<dev>/alternate_modes

Defined on file *sysfs-driver-hid-logitech-lg4ff*

Displays a set of alternate modes supported by a wheel. Each mode is listed as follows:

Tag: Mode Name

Currently active mode is marked with an asterisk. List also contains an abstract item ”native” which always denotes the native mode of the wheel. Echoing the mode tag switches the wheel into the corresponding mode. Depending on the exact model of the wheel not all listed modes might always be selectable. If a wheel cannot be switched into the desired mode, -EINVAL is returned accompanied with an explanatory message in the kernel log. This entry is not created for devices that have only one mode.

Currently supported mode switches:

Driving Force Pro:

```
DF-EX --> DFP
```

G25:

```
DF-EX --> DFP --> G25
```

G27:

```
DF-EX <*> DFP <-> G25 <-> G27
DF-EX <*--------> G25 <-> G27
DF-EX <*----------------> G27
```

G29:

```
DF-EX <*> DFP <-> G25 <-> G27 <-> G29
DF-EX <*--------> G25 <-> G27 <-> G29
DF-EX <*----------------> G27 <-> G29
DF-EX <*------------------------> G29
```

DFGT:

```
DF-EX <*> DFP <-> DFGT
DF-EX <*--------> DFGT
```

- hid_logitech module must be loaded with lg4ff_no_autoswitch=1 parameter set in order for the switch to DF-EX mode to work.

/sys/bus/hid/drivers/logitech/<dev>/combine_pedals

Defined on file *sysfs-driver-hid-logitech-lg4ff*

Controls whether a combined value of accelerator and brake is reported on the Y axis of the controller. Useful for older games which can do not work with separate accelerator/brake axis. Off ('0') by default, enabled by setting '1'.

/sys/bus/hid/drivers/logitech/<dev>/range

Defined on file *sysfs-driver-hid-logitech-lg4ff*

Display minimum, maximum and current range of the steering wheel. Writing a value within min and max boundaries sets the range of the wheel.

/sys/bus/hid/drivers/logitech/<dev>/real_id

Defined on file *sysfs-driver-hid-logitech-lg4ff*

Displays the real model of the wheel regardless of any alternate mode the wheel might be switched to. It is a read-only value. This entry is not created for devices that have only one mode.

/sys/bus/hid/drivers/ntrig/<dev>/activate_slack

Defined on file *sysfs-driver-hid-ntrig*

(RW) Number of contact frames ignored before acknowledging the start of activity (activating touch).

/sys/bus/hid/drivers/ntrig/<dev>/activation_width
/sys/bus/hid/drivers/ntrig/<dev>/activation_height

Defined on file *sysfs-driver-hid-ntrig*

Threholds to override activation slack.

activation_width	(RW) Width threshold to immediately start processing touch events.
activation_height	(RW) Height threshold to immediately start processing touch events.

/sys/bus/hid/drivers/ntrig/<dev>/decativate_slack

Defined on file *sysfs-driver-hid-ntrig*

(RW) Number of empty (no contact) frames ignored before acknowledging the end of activity (deactivating touch).

When the last finger is removed from the device, it sends a number of empty frames. By holding off on deactivation for a few frames false erroneous disconnects can be tolerated, where the sensor may mistakenly not detect a finger that is still present.

/sys/bus/hid/drivers/ntrig/<dev>/min_width
/sys/bus/hid/drivers/ntrig/<dev>/min_height

Defined on file *sysfs-driver-hid-ntrig*

Minimum size contact accepted.

min_width	(RW) Minimum touch contact width to decide activation and activity.
min_height	(RW) Minimum touch contact height to decide activation and activity.

/sys/bus/hid/drivers/ntrig/<dev>/sensor_logical_width
/sys/bus/hid/drivers/ntrig/<dev>/sensor_logical_height

Defined on file *sysfs-driver-hid-ntrig*

(RO) The range for positions reported during activity.

/sys/bus/hid/drivers/ntrig/<dev>/sensor_physical_width
/sys/bus/hid/drivers/ntrig/<dev>/sensor_physical_height

Defined on file *sysfs-driver-hid-ntrig*

(RO) These are internal ranges not used for normal events but useful for tuning.

/sys/bus/hid/drivers/prodikeys/.../channel

Defined on file *sysfs-driver-hid-prodikeys*

Allows control (via software) the midi channel to which that the pc-midi keyboard will output.midi data. Range: 0..15 Type: Read/write

/sys/bus/hid/drivers/prodikeys/.../octave

Defined on file *sysfs-driver-hid-prodikeys*

Controls the octave shift modifier in the pc-midi driver. The octave can be shifted via software up/down 2 octaves. 0 means the no ocatve shift. Range: -2..2 (minus 2 to plus 2) Type: Read/Write

/sys/bus/hid/drivers/prodikeys/.../sustain

Defined on file *sysfs-driver-hid-prodikeys*

Allows control (via software) the sustain duration of a note held by the pc-midi driver. 0 means sustain mode is disabled. Range: 0..5000 (milliseconds) Type: Read/write

/sys/bus/hid/drivers/wiimote/<dev>/bboard_calib

Defined on file *sysfs-driver-hid-wiimote*

This attribute is only provided if the device was detected as a balance board. It provides a single line with 3 calibration values for all 4 sensors. The values are separated by colons and are each 2 bytes long (encoded as 4 digit hexadecimal value). First, 0kg values for all 4 sensors are written, followed by the 17kg values for all 4 sensors and last the 34kg values for all 4 sensors.

Calibration data is already applied by the kernel to all input values but may be used by user-space to perform other transformations.

/sys/bus/hid/drivers/wiimote/<dev>/devtype

Defined on file *sysfs-driver-hid-wiimote*

While a device is initialized by the wiimote driver, we perform a device detection and signal a ”change” uevent after it is done. This file shows the detected device type. ”pending” means that the detection is still ongoing, ”unknown” means, that the device couldn't be detected or loaded. ”generic” means, that the device couldn't be detected but supports basic Wii Remote features and can be used. Other strings for each device-type are available and may be added if new device-specific detections are added. Currently supported are:

gen10:	First Wii Remote generation
gen20:	Second Wii Remote Plus generation (builtin MP)
balanceboard:	Wii Balance Board

/sys/bus/hid/drivers/wiimote/<dev>/extension

Defined on file *sysfs-driver-hid-wiimote*

This file contains the currently connected and initialized extensions. It can be one of: none, motionp, nunchuck, classic, motionp+nunchuck, motionp+classic motionp is the official Nintendo Motion+ extension, nunchuck is the official Nintendo Nunchuck extension and classic is the Nintendo Classic Controller extension. The motionp extension can be combined with the other two.

Starting with kernel-version 3.11 Motion Plus hotplugging is supported and if detected, it's no longer reported as static extension. You will get uevent notifications for the motion-plus device then.

/sys/bus/hid/drivers/wiimote/<dev>/led1
/sys/bus/hid/drivers/wiimote/<dev>/led2
/sys/bus/hid/drivers/wiimote/<dev>/led3
/sys/bus/hid/drivers/wiimote/<dev>/led4

Defined on file *sysfs-driver-hid-wiimote*

Make it possible to set/get current led state. Reading from it returns 0 if led is off and 1 if it is on. Writing 0 to it disables the led, writing 1 enables it.

/sys/bus/hid/drivers/wiimote/<dev>/pro_calib

Defined on file *sysfs-driver-hid-wiimote*

This attribute is only provided if the device was detected as a pro-controller. It provides a single line with 4 calibration values for all 4 analog sticks. Format is: "x1:y1 x2:y2". Data is prefixed with a +/-. Each value is a signed 16bit number. Data is encoded as decimal numbers and specifies the offsets of the analog sticks of the pro-controller.

Calibration data is already applied by the kernel to all input values but may be used by user-space to perform other transformations.

Calibration data is detected by the kernel during device setup. You can write "scann" into this file to re-trigger calibration. You can also write data directly in the form "x1:y1 x2:y2" to set the calibration values manually.

/sys/bus/hsi

Defined on file *sysfs-bus-hsi*

High Speed Synchronous Serial Interface (HSI) is a serial interface mainly used for connecting application engines (APE) with cellular modem engines (CMT) in cellular handsets. The bus will be populated with devices (hsi_clients) representing the protocols available in the system. Bus drivers implement those protocols.

/sys/bus/hsi/devices/.../modalias

Defined on file *sysfs-bus-hsi*

Stores the same MODALIAS value emitted by uevent Format: hsi:<hsi_client device name>

/sys/bus/i2c/.../idle_state

Defined on file *sysfs-bus-i2c-devices-pca954x*

Value that exists only for mux devices that can be written to control the behaviour of the multiplexer on idle. Possible values:

-2	disconnect on idle, i.e. deselect the last used channel, which is useful when there is a device with an address that conflicts with another device on another mux on the same parent bus.
-1	leave the mux as-is, which is the most optimal setting in terms of I2C operations and is the default mode.
0..<nchans>	set the mux to a predetermined channel, which is useful if there is one channel that is used almost always, and you want to reduce the latency for normal operations after rare transactions on other channels

/sys/bus/i2c/devices/.../bd9571mwv-regulator.*.auto/backup_mode

Defined on file *sysfs-driver-bd9571mwv-regulator*

Read/write the current state of DDR Backup Mode, which controls if DDR power rails will be kept powered during system suspend. ("on"/"1" = enabled, "off"/"0" = disabled). Two types of power switches (or control signals) can be used:

A. With a momentary power switch (or pulse signal), DDR Backup Mode is enabled by default when available, as the PMIC will be configured only during system suspend.

B. With a toggle power switch (or level signal), the following steps must be followed exactly:

   1. Configure PMIC for backup mode, to change the role of the accessory power switch from a power switch to a wake-up switch,
   2. Switch accessory power switch off, to prepare for system suspend, which is a manual step not controlled by software,
   3. Suspend system,
   4. Switch accessory power switch on, to resume the system.

   DDR Backup Mode must be explicitly enabled by the user, to invoke step 1.

See also Documentation/devicetree/bindings/mfd/rohm,bd9571mwv.yaml.

Users: User space applications for embedded boards equipped with a BD9571MWV PMIC.

/sys/bus/i2c/devices/.../calibration

Defined on file *sysfs-bus-i2c-devices-hm6352*

Sets the calibration on or off (1 = on, 0 = off). See the chip data sheet.

/sys/bus/i2c/devices/.../device

Defined on file *sysfs-bus-i2c-devices-fsa9480*

show what device is attached

NONE	no device
USB	USB device is attached
UART	UART is attached
CHARGER	Charger is attached
JIG	JIG is attached

/sys/bus/i2c/devices/.../heading0_input

Defined on file *sysfs-bus-i2c-devices-hm6352*

Reports the current heading from the compass as a floating point value in degrees.

/sys/bus/i2c/devices/.../output_hvled[n]

Defined on file *sysfs-bus-i2c-devices-lm3533*

Set the controlling backlight device for high-voltage current sink HVLED[n] (n = 1, 2) (0, 1).

/sys/bus/i2c/devices/.../output_lvled[n]

Defined on file *sysfs-bus-i2c-devices-lm3533*

Set the controlling led device for low-voltage current sink LVLED[n] (n = 1..5) (0..3).

/sys/bus/i2c/devices/.../power_state

Defined on file *sysfs-bus-i2c-devices-hm6352*

Sets the power state of the device. 0 sets the device into sleep mode, 1 wakes it up.

/sys/bus/i2c/devices/.../switch

Defined on file *sysfs-bus-i2c-devices-fsa9480*

show or set the state of manual switch

VAUDIO	switch to VAUDIO path
UART	switch to UART path
AUDIO	switch to AUDIO path
DHOST	switch to DHOST path
AUTO	switch automatically by device

/sys/bus/i2c/devices/.../trickle_charge_bypass

Defined on file *sysfs-bus-i2c-devices-bq32k*

Attribute for enable/disable the trickle charge bypass The trickle_charge_bypass attribute allows the userspace to enable/disable the Trickle charge FET bypass.

/sys/bus/i2c/devices/xxx/fw_version

Defined on file *sysfs-driver-input-exc3000*

Reports the firmware version provided by the touchscreen, for example "00_T6" on a EXC80H60

Access: Read

Valid values: Represented as string

/sys/bus/i2c/devices/xxx/model

Defined on file *sysfs-driver-input-exc3000*

Reports the model identification provided by the touchscreen, for example "Orion_1320" on a EXC80H60

Access: Read

Valid values: Represented as string

/sys/bus/i2c/devices/xxx/type

Defined on file *sysfs-driver-input-exc3000*

Reports the type identification provided by the touchscreen, for example "PCAP82H80 Series"

Access: Read

Valid values: Represented as string

/sys/bus/i2c/drivers/ucsi_ccg/.../do_flash

Defined on file *sysfs-driver-ucsi-ccg*

Tell the driver for Cypress CCGx Type-C controller to attempt firmware upgrade by writing [Yy1] to the file.

/sys/bus/i3c/devices/<bus-id>-<device-pid>

Defined on file *sysfs-bus-i3c*

These directories are just symbolic links to */sys/bus/i3c/devices/i3c-<bus-id>/<bus-id>-<device-pid>*.

/sys/bus/i3c/devices/i3c-<bus-id>

Defined on file *sysfs-bus-i3c*

An I3C bus. This directory will contain one sub-directory per I3C device present on the bus.

/sys/bus/i3c/devices/i3c-<bus-id>/<bus-id>-<device-pid>

Defined on file *sysfs-bus-i3c*

An I3C device present on I3C bus identified by <bus-id>. Note that all devices are represented including the master driving the bus.

/sys/bus/i3c/devices/i3c-<bus-id>/<bus-id>-<device-pid>/bcr

Defined on file *sysfs-bus-i3c*

BCR stands for Bus Characteristics Register and express the device capabilities in term of speed, maximum read/write length, etc. See the I3C specification for more details.

/sys/bus/i3c/devices/i3c-<bus-id>/<bus-id>-<device-pid>/dcr

Defined on file *sysfs-bus-i3c*

DCR stands for Device Characteristics Register and express the device capabilities in term of exposed features. See the I3C specification for more details.

/sys/bus/i3c/devices/i3c-<bus-id>/<bus-id>-<device-pid>/dynamic_address

Defined on file *sysfs-bus-i3c*

Dynamic address assigned to device <bus-id>-<device-pid>. This address may change if the bus is re-initialized.

/sys/bus/i3c/devices/i3c-<bus-id>/<bus-id>-<device-pid>/hdrcap

Defined on file *sysfs-bus-i3c*

Expose the HDR (High Data Rate) capabilities of a device. Returns a list of supported HDR mode, each element is separated by space. Modes can be ”hdr-ddr”, ”hdr-tsp” and ”hdr-tsl”.

See the I3C specification for more details about these HDR modes.

/sys/bus/i3c/devices/i3c-<bus-id>/<bus-id>-<device-pid>/pid

Defined on file *sysfs-bus-i3c*

PID stands for Provisioned ID and is used to uniquely identify a device on a bus. This PID contains information about the vendor, the part and an instance ID so that several devices of the same type can be connected on the same bus. See the I3C specification for more details.

/sys/bus/i3c/devices/i3c-<bus-id>/bcr

Defined on file *sysfs-bus-i3c*

BCR stands for Bus Characteristics Register and express the device capabilities in term of speed, maximum read/write length, etc. See the I3C specification for more details. This entry describes the BCR of the master controller driving the bus.

/sys/bus/i3c/devices/i3c-<bus-id>/current_master

Defined on file *sysfs-bus-i3c*

Expose the master that owns the bus (<bus-id>-<master-pid>) at the time this file is read. Note that bus ownership can change overtime, so there's no guarantee that when the read() call returns, the value returned is still valid.

/sys/bus/i3c/devices/i3c-<bus-id>/dcr

Defined on file *sysfs-bus-i3c*

DCR stands for Device Characteristics Register and express the device capabilities in term of exposed features. See the I3C specification for more details. This entry describes the DCR of the master controller driving the bus.

/sys/bus/i3c/devices/i3c-<bus-id>/dynamic_address

Defined on file *sysfs-bus-i3c*

Dynamic address assigned to the master controller. This address may change if the bus is re-initialized.

/sys/bus/i3c/devices/i3c-<bus-id>/hdrcap

Defined on file *sysfs-bus-i3c*

Expose the HDR (High Data Rate) capabilities of a device. Returns a list of supported HDR mode, each element is separated by space. Modes can be ”hdr-ddr”, ”hdr-tsp” and ”hdr-tsl”. See the I3C specification for more details about these HDR modes.

This entry describes the HDRCAP of the master controller driving the bus.

/sys/bus/i3c/devices/i3c-<bus-id>/hotjoin

Defined on file *sysfs-bus-i3c*

I3C’s Hot-Join mechanism allows an I3C Device to inform the Active Controller that a newly-joined Target is present on the I3C Bus and is ready to receive a Dynamic Address, in order to become fully functional on the Bus. Hot-Join is used when the Target is mounted on the same I3C bus and remains depowered until needed or until the Target is physically inserted into the I3C bus

This entry allows to enable or disable Hot-join of the Current Controller driving the bus.

/sys/bus/i3c/devices/i3c-<bus-id>/i2c_scl_frequency

Defined on file *sysfs-bus-i3c*

The frequency (expressed in Hz) of the SCL signal when operating in I2C mode.

/sys/bus/i3c/devices/i3c-<bus-id>/i3c_scl_frequency

Defined on file *sysfs-bus-i3c*

The frequency (expressed in Hz) of the SCL signal when operating in I3C SDR mode.

/sys/bus/i3c/devices/i3c-<bus-id>/mode

Defined on file *sysfs-bus-i3c*

I3C bus mode. Can be "pure", "mixed-fast" or "mixed-slow". See the I3C specification for a detailed description of what each of these modes implies.

/sys/bus/i3c/devices/i3c-<bus-id>/pid

Defined on file *sysfs-bus-i3c*

PID stands for Provisioned ID and is used to uniquely identify a device on a bus. This PID contains information about the vendor, the part and an instance ID so that several devices of the same type can be connected on the same bus. See the I3C specification for more details. This entry describes the PID of the master controller driving the bus.

/sys/bus/iio/devices/device[n]/in_illuminance0_calibrate

Defined on file *sysfs-bus-iio-light-tsl2772*

Causes an internal calibration of the als gain trim value which is later used in calculating illuminance in lux.

/sys/bus/iio/devices/device[n]/in_illuminance_calibrate

Defined on file *sysfs-bus-iio-light-tsl2583*

This property causes an internal calibration of the als gain trim value which is later used in calculating illuminance in lux.

/sys/bus/iio/devices/device[n]/in_illuminance_input_target

Defined on file *sysfs-bus-iio-light-tsl2583*

This property is the known externally illuminance (in lux). It is used in the process of calibrating the device accuracy.

/sys/bus/iio/devices/device[n]/in_illuminance_lux_table

Defined on file *sysfs-bus-iio-light-tsl2583*

This property gets/sets the table of coefficients used in calculating illuminance in lux.

/sys/bus/iio/devices/device[n]/in_proximity0_calibrate

Defined on file *sysfs-bus-iio-light-tsl2772*

Causes a recalculation and adjustment to the proximity_thresh_rising_value.

/sys/bus/iio/devices/iio:device*/buffer/hwfifo_timeout

Defined on file *sysfs-bus-iio*

A read/write property to provide capability to delay reporting of samples till a timeout is reached. This allows host processors to sleep, while the sensor is storing samples in its internal fifo. The maximum timeout in seconds can be specified by setting hwfifo_timeout.The current delay can be read by reading hwfifo_timeout. A value of 0 means that there is no timeout.

/sys/bus/iio/devices/iio:deviceX

Defined on file *sysfs-bus-iio*

Hardware chip or device accessed by one communication port. Corresponds to a grouping of sensor channels. X is the IIO index of the device.

/sys/bus/iio/devices/iio:deviceX/ac_excitation_en

Defined on file *sysfs-bus-iio-adc-ad7192*

This attribute, if available, is used to enable the AC excitation mode found on some converters. In ac excitation mode, the polarity of the excitation voltage is reversed on alternate cycles, to eliminate DC errors.

/sys/bus/iio/devices/iio:deviceX/auto_zeroing_mux_enable

Defined on file *sysfs-bus-iio-adc-mcp3564*

This attribute is used to enable the analog input multiplexer auto-zeroing algorithm (the input multiplexer and the ADC include an offset cancellation algorithm that cancels the offset contribution of the ADC). When the offset cancellation algorithm is enabled, ADC takes two conversions, one with the differential input as VIN+/VIN-, one with VIN+/VIN- inverted. In this case the conversion time is multiplied by two compared to the default case where the algorithm is disabled. This technique allows the cancellation of the ADC offset error and the achievement of ultra-low offset without any digital calibration. The resulting offset is the residue of the difference between the two conversions, which is on the order of magnitude of the noise floor. This offset is effectively canceled at every conversion, so the residual offset error temperature drift is extremely low. Write '1' to enable it, write '0' to disable it.

/sys/bus/iio/devices/iio:deviceX/auto_zeroing_ref_enable

Defined on file *sysfs-bus-iio-adc-mcp3564*

This attribute is used to enable the chopping algorithm for the internal voltage reference buffer. This setting has no effect when external voltage reference is selected. Internal voltage reference buffer injects a certain quantity of 1/f noise into the system that can be modulated with the incoming input signals and can limit the SNR performance at higher Oversampling Ratio values (over 256). To overcome this limitation, the buffer includes an auto-zeroing algorithm that greatly reduces (cancels out) the 1/f noise and cancels the offset value of the reference buffer. As a result, the SNR of the system is not affected by this 1/f noise component of the reference buffer, even at maximum oversampling ratio values. Write '1' to enable it, write '0' to disable it.

/sys/bus/iio/devices/iio:deviceX/battery_low

Defined on file *sysfs-bus-iio-meas-spec*

Reading returns either '1' or '0'. '1' means that the battery level supplied to sensor is below 2.25V. This ABI is available for tsys02d, htu21, ms8607

/sys/bus/iio/devices/iio:deviceX/boost_current_gain

Defined on file *sysfs-bus-iio-adc-mcp3564*

This attribute is used to set the gain of the biasing current circuit of the Delta-Sigma modulator. The different BOOST settings are applied to the entire modulator circuit, including the voltage reference buffers.

/sys/bus/iio/devices/iio:deviceX/boost_current_gain_available

Defined on file *sysfs-bus-iio-adc-mcp3564*

Reading returns a list with the possible gain values for the current biasing circuit of the Delta-Sigma modulator.

/sys/bus/iio/devices/iio:deviceX/bridge_switch_en

Defined on file *sysfs-bus-iio-adc-ad7192*

This attribute, if available, is used to close or open the bridge power down switch found on some converters. In bridge applications, such as strain gauges and load cells, the bridge itself consumes the majority of the current in the system. To minimize the current consumption of the system, the bridge can be disconnected (when it is not being used using the bridge_switch_en attribute.

/sys/bus/iio/devices/iio:deviceX/buffer

Defined on file *sysfs-bus-iio*

Directory of attributes relating to the buffer for the device.

/sys/bus/iio/devices/iio:deviceX/buffer/hwfifo_enabled

Defined on file *sysfs-bus-iio*

A read-only boolean value that indicates if the hardware fifo is currently enabled or disabled. If the device does not have a hardware fifo this entry is not present. The hardware fifo is enabled when the buffer is enabled if the current hardware fifo watermark level is set and other current device settings allows it (e.g. if a trigger is set that samples data differently that the hardware fifo does then hardware fifo will not enabled).

If the hardware fifo is enabled and the level of the hardware fifo reaches the hardware fifo watermark level the device will flush its hardware fifo to the device buffer. Doing a non blocking read on the device when no samples are present in the device buffer will also force a flush.

When the hardware fifo is enabled there is no need to use a trigger to use buffer mode since the watermark settings guarantees that the hardware fifo is flushed to the device buffer.

/sys/bus/iio/devices/iio:deviceX/buffer/hwfifo_watermark

Defined on file *sysfs-bus-iio*

Read-only entry that contains a single integer specifying the current watermark level for the hardware fifo. If the device does not have a hardware fifo this entry is not present. The watermark level for the hardware fifo is set by the driver based on the value set by the user in buffer/watermark but taking into account hardware limitations (e.g. most hardware buffers are limited to 32-64 samples, some hardware buffers watermarks are fixed or have minimum levels). A value of 0 means that the hardware watermark is unset.

/sys/bus/iio/devices/iio:deviceX/buffer/hwfifo_watermark_available

Defined on file *sysfs-bus-iio*

A list of positive integers specifying the available watermark levels for the hardware fifo. This entry is optional and if it is not present it means that all the values between hwfifo_watermark_min and hwfifo_watermark_max are supported.

If the user sets buffer/watermark to a value greater than hwfifo_watermak_min but not equal to any of the values in this list, the driver will chose an appropriate value for the hardware fifo watermark level.

/sys/bus/iio/devices/iio:deviceX/buffer/hwfifo_watermark_max

Defined on file *sysfs-bus-iio*

A single positive integer specifying the maximum watermark level for the hardware fifo of this device. If the device does not have a hardware fifo this entry is not present.

If the user sets buffer/watermark to a value greater than this one, then the hardware watermark will be capped at this value.

/sys/bus/iio/devices/iio:deviceX/buffer/hwfifo_watermark_min

Defined on file *sysfs-bus-iio*

A single positive integer specifying the minimum watermark level for the hardware fifo of this device. If the device does not have a hardware fifo this entry is not present.

If the user sets buffer/watermark to a value less than this one, then the hardware watermark will remain unset.

/sys/bus/iio/devices/iio:deviceX/buffer/length_align_bytes

Defined on file *sysfs-bus-iio-dma-buffer*

DMA buffers tend to have a alignment requirement for the buffers. If this alignment requirement is not met samples might be dropped from the buffer.

This property reports the alignment requirements in bytes. This means that the buffer size in bytes needs to be a integer multiple of the number reported by this file.

The alignment requirements in number of sample sets will depend on the enabled channels and the bytes per channel. This means that the alignment requirement in samples sets might change depending on which and how many channels are enabled. Whereas the alignment requirement reported in bytes by this property will remain static and does not depend on which channels are enabled.

/sys/bus/iio/devices/iio:deviceX/bufferY

Defined on file *sysfs-bus-iio*

Directory containing interfaces for elements that will be captured for a single triggered sample set in the buffer.

Since kernel 5.11 the scan_elements attributes are merged into the bufferY directory, to be configurable per buffer.

/sys/bus/iio/devices/iio:deviceX/bufferY/data_available

Defined on file *sysfs-bus-iio*

A read-only value indicating the bytes of data available in the buffer. In the case of an output buffer, this indicates the amount of empty space available to write data to. In the case of an input buffer, this indicates the amount of data available for reading.

/sys/bus/iio/devices/iio:deviceX/bufferY/enable

Defined on file *sysfs-bus-iio*

Actually start the buffer capture up. Will start trigger if first device and appropriate.

Note that it might be impossible to configure other attributes, (e.g.: events, scale, sampling rate) if they impact the currently active buffer capture session.

/sys/bus/iio/devices/iio:deviceX/bufferY/length

Defined on file *sysfs-bus-iio*

Number of scans contained by the buffer.

/sys/bus/iio/devices/iio:deviceX/bufferY/watermark

Defined on file *sysfs-bus-iio*

A single positive integer specifying the maximum number of scan elements to wait for.

Poll will block until the watermark is reached.

Blocking read will wait until the minimum between the requested read amount or the low water mark is available.

Non-blocking read will retrieve the available samples from the buffer even if there are less samples then watermark level. This allows the application to block on poll with a timeout and read the available samples after the timeout expires and thus have a maximum delay guarantee.

/sys/bus/iio/devices/iio:deviceX/calibrate

Defined on file *sysfs-bus-iio-cros-ec*

Writing '1' will perform a FOC (Fast Online Calibration). The corresponding calibration offsets can be read from *_*calibbias* entries.

/sys/bus/iio/devices/iio:deviceX/calibration_auto_enable

Defined on file *sysfs-bus-iio*

Some sensors have the ability to apply auto calibration at runtime. For example, it may be necessary to compensate for contaminant build-up in a measurement chamber or optical element deterioration that would otherwise lead to sensor drift.

Writing 1 or 0 to this attribute will respectively activate or deactivate this auto calibration function.

Upon reading, the current status is returned.

/sys/bus/iio/devices/iio:deviceX/calibration_data

Defined on file *sysfs-bus-iio-bno055*

Reports the binary calibration data blob for the IMU sensors.

/sys/bus/iio/devices/iio:deviceX/calibration_forced_value

Defined on file *sysfs-bus-iio*

Some sensors have the ability to apply a manual calibration using a known measurement value, perhaps obtained from an external reference device.

Writing a value to this function will force such a calibration change. For the scd30 the value should be from the range [400 1 2000].

Note for the scd30 that a valid value may only be obtained once it is has been written. Until then any read back of this value should be ignored. As for the scd4x an error will be returned immediately if the manual calibration has failed.

/sys/bus/iio/devices/iio:deviceX/calibration_forced_value_available

Defined on file *sysfs-bus-iio*

Available range for the forced calibration value, expressed as:

- a range specified as ”[min step max]”

/sys/bus/iio/devices/iio:deviceX/cleaning_period

Defined on file *sysfs-bus-iio-sps30*

Sensor is capable of triggering self cleaning periodically. Period can be changed by writing a new value here. Upon reading the current one is returned. Units are seconds.

Writing 0 disables periodical self cleaning entirely.

/sys/bus/iio/devices/iio:deviceX/cleaning_period_available

Defined on file *sysfs-bus-iio-sps30*

The range of available values in seconds represented as the minimum value, the step and the maximum value, all enclosed in square brackets.

/sys/bus/iio/devices/iio:deviceX/current_timestamp_clock

Defined on file *sysfs-bus-iio*

String identifying current posix clock used to timestamp buffered samples and events for device X.

/sys/bus/iio/devices/iio:deviceX/error_status

Defined on file *sysfs-bus-iio-chemical-sunrise-co2*

Reading returns the current chip error status.

/sys/bus/iio/devices/iio:deviceX/error_status_available

Defined on file *sysfs-bus-iio-chemical-sunrise-co2*

Reading returns the list of possible chip error status. Available options are: - 'error_fatal': Analog front-end initialization error - 'error_i2c': Read/write to non-existing register - 'error_algorithm': Corrupted parameters - 'error_calibration': Calibration has failed - 'error_self_diagnostic': Internal interface failure - 'error_out_of_range': Measured concentration out of scale - 'error_memory': Error during memory operations - 'error_no_measurement': Cleared at first measurement - 'error_low_voltage': Sensor regulated voltage too low - 'error_measurement_timeout': Unable to complete measurement

/sys/bus/iio/devices/iio:deviceX/events

Defined on file *sysfs-bus-iio*

Configuration of which hardware generated events are passed up to user-space.

/sys/bus/iio/devices/iio:deviceX/events/in_altvoltage0_mag_rising_reset_max

Defined on file *sysfs-bus-iio-resolver-ad2s1210*

Reading returns the current Degradation of Signal Reset Maximum Threshold value in millivolts. Writing sets the value.

/sys/bus/iio/devices/iio:deviceX/events/in_altvoltage0_mag_rising_reset_max_available

Defined on file *sysfs-bus-iio-resolver-ad2s1210*

Reading returns the allowable voltage range for in_altvoltage0_mag_rising_reset_max.

/sys/bus/iio/devices/iio:deviceX/events/in_altvoltage0_mag_rising_reset_min

Defined on file *sysfs-bus-iio-resolver-ad2s1210*

Reading returns the current Degradation of Signal Reset Minimum Threshold value in millivolts. Writing sets the value.

/sys/bus/iio/devices/iio:deviceX/events/in_altvoltage0_mag_rising_reset_min_available

Defined on file *sysfs-bus-iio-resolver-ad2s1210*

Reading returns the allowable voltage range for in_altvoltage0_mag_rising_reset_min.

/sys/bus/iio/devices/iio:deviceX/events/in_altvoltageY_mag_either_label
/sys/bus/iio/devices/iio:deviceX/events/in_altvoltageY_mag_rising_label
/sys/bus/iio/devices/iio:deviceX/events/in_altvoltageY_thresh_falling_label
/sys/bus/iio/devices/iio:deviceX/events/in_altvoltageY_thresh_rising_label
/sys/bus/iio/devices/iio:deviceX/events/in_anglvelY_mag_rising_label
/sys/bus/iio/devices/iio:deviceX/events/in_anglY_thresh_rising_label
/sys/bus/iio/devices/iio:deviceX/events/in_phaseY_mag_rising_label

Defined on file *sysfs-bus-iio*

Optional symbolic label to a device channel event. If a label is defined for this event add that to the event specific attributes. This is useful for userspace to be able to better identify an individual event.

/sys/bus/iio/devices/iio:deviceX/events/in_illuminance_period_available

Defined on file *sysfs-bus-iio*

List of valid periods (in seconds) for which the light intensity must be above the threshold level before interrupt is asserted.

/sys/bus/iio/devices/iio:deviceX/events/in_voltageY_thresh_falling_value

Defined on file *sysfs-bus-iio-adc-hi8435*

Channel Y low voltage threshold. If sensor input voltage goes lower then this value then the threshold falling event is pushed. Depending on in_voltageY_sensing_mode the low voltage threshold is separately set for "GND-Open" and "Supply-Open" modes. Channels 0..31 have common low threshold values, but could have different sensing_modes.

The low voltage threshold range is between 2..21V. Hysteresis between low and high thresholds can not be lower then 2 and can not be odd.

If falling threshold results hysteresis to odd value then rising threshold is automatically subtracted by one.

/sys/bus/iio/devices/iio:deviceX/events/in_voltageY_thresh_rising_value

Defined on file *sysfs-bus-iio-adc-hi8435*

Channel Y high voltage threshold. If sensor input voltage goes higher then this value then the threshold rising event is pushed. Depending on in_voltageY_sensing_mode the high voltage threshold is separately set for "GND-Open" and "Supply-Open" modes.

Channels 0..31 have common high threshold values, but could have different sensing_modes.

The high voltage threshold range is between 3..22V. Hysteresis between low and high thresholds can not be lower then 2 and can not be odd.

If rising threshold results hysteresis to odd value then falling threshold is automatically appended by one.

/sys/bus/iio/devices/iio:deviceX/fault_oc

Defined on file *sysfs-bus-iio-thermocouple*

Open-circuit fault. The detection of open-circuit faults, such as those caused by broken thermocouple wires. Reading returns '1' if fault, '0' otherwise.

/sys/bus/iio/devices/iio:deviceX/fault_ovuv

Defined on file *sysfs-bus-iio-thermocouple*

Overvoltage or Undervoltage Input Fault. The internal circuitry is protected from excessive voltages applied to the thermocouple cables. The device can also detect if such a condition occurs.

Reading returns '1' if input voltage is negative or greater than VDD, otherwise '0'.

/sys/bus/iio/devices/iio:deviceX/filter_mode

Defined on file *sysfs-bus-iio-filter-admv8818*

This attribute configures the filter mode. Reading returns the actual mode.

/sys/bus/iio/devices/iio:deviceX/filter_mode_available

Defined on file *sysfs-bus-iio-filter-admv8818*

Reading this returns the valid values that can be written to the on_altvoltage0_mode attribute:

- auto -> Adjust bandpass filter to track changes in input clock rate.
- manual -> disable/unregister the clock rate notifier / input clock tracking.
- **bypass -> bypass low pass filter, high pass filter and disable/unregister**
  the clock rate notifier

/sys/bus/iio/devices/iio:deviceX/fusion_enable

Defined on file *sysfs-bus-iio-bno055*

Can be 1 or 0. Enables/disables the "sensor fusion" (a.k.a. NDOF) HW function.

/sys/bus/iio/devices/iio:deviceX/heater_enable

Defined on file *sysfs-bus-iio*

'1' (enable) or '0' (disable) specifying the enable of heater function. Same reading values apply.

This ABI is especially applicable for humidity sensors to heatup the device and get rid of any condensation in some humidity environment

/sys/bus/iio/devices/iio:deviceX/id

Defined on file *sysfs-bus-iio-cros-ec*

This attribute is exposed by the CrOS EC sensors driver and represents the sensor ID as exposed by the EC. This ID is used by the Android sensor service hardware abstraction layer (sensor HAL) through the Android container on ChromeOS.

/sys/bus/iio/devices/iio:deviceX/in_X_mean_raw

Defined on file *sysfs-bus-iio*

Averaged raw measurement from channel X. The number of values used for averaging is device specific. The converting rules for normal raw values also applies to the averaged raw values.

/sys/bus/iio/devices/iio:deviceX/in_accel_calibbias_available
/sys/bus/iio/devices/iio:deviceX/in_anglvel_calibbias_available

Defined on file *sysfs-bus-iio*

Available values of calibbias. Maybe expressed as either of:

- a small discrete set of values like "0 2 4 6 8"
- a range specified as "[min step max]"

/sys/bus/iio/devices/iio:deviceX/in_accel_calibration_auto_status

Defined on file *sysfs-bus-iio-bno055*

Reports the autocalibration status for the accelerometer sensor. Can be 0 (calibration non even enabled) or 1 to 5 where the greater the number, the better the calibration status.

/sys/bus/iio/devices/iio:deviceX/in_accel_linear_x_raw
/sys/bus/iio/devices/iio:deviceX/in_accel_linear_y_raw
/sys/bus/iio/devices/iio:deviceX/in_accel_linear_z_raw

Defined on file *sysfs-bus-iio*

As per in_accel_X_raw attributes, but minus the acceleration due to gravity.

/sys/bus/iio/devices/iio:deviceX/in_accel_offset
/sys/bus/iio/devices/iio:deviceX/in_accel_x_offset
/sys/bus/iio/devices/iio:deviceX/in_accel_y_offset
/sys/bus/iio/devices/iio:deviceX/in_accel_z_offset
/sys/bus/iio/devices/iio:deviceX/in_voltageY_offset
/sys/bus/iio/devices/iio:deviceX/in_voltage_offset
/sys/bus/iio/devices/iio:deviceX/in_voltageY_i_offset
/sys/bus/iio/devices/iio:deviceX/in_voltageY_q_offset
/sys/bus/iio/devices/iio:deviceX/in_voltage_q_offset
/sys/bus/iio/devices/iio:deviceX/in_voltage_i_offset
/sys/bus/iio/devices/iio:deviceX/in_currentY_offset
/sys/bus/iio/devices/iio:deviceX/in_current_offset
/sys/bus/iio/devices/iio:deviceX/in_currentY_i_offset
/sys/bus/iio/devices/iio:deviceX/in_currentY_q_offset
/sys/bus/iio/devices/iio:deviceX/in_current_q_offset
/sys/bus/iio/devices/iio:deviceX/in_current_i_offset
/sys/bus/iio/devices/iio:deviceX/in_tempY_offset
/sys/bus/iio/devices/iio:deviceX/in_temp_offset
/sys/bus/iio/devices/iio:deviceX/in_pressureY_offset
/sys/bus/iio/devices/iio:deviceX/in_pressure_offset
/sys/bus/iio/devices/iio:deviceX/in_humidityrelative_offset
/sys/bus/iio/devices/iio:deviceX/in_magn_offset
/sys/bus/iio/devices/iio:deviceX/in_rot_offset
/sys/bus/iio/devices/iio:deviceX/in_angl_offset
/sys/bus/iio/devices/iio:deviceX/in_capacitanceX_offset

Defined on file *sysfs-bus-iio*

If known for a device, offset to be added to <type>[Y]_raw prior to scaling by <type>[Y]_scale in order to obtain value in the <type> units as specified in <type>[Y]_raw documentation. Not present if the offset is always 0 or unknown. If Y or axis <x|y|z> is not present, then the offset applies to all in channels of <type>. May be writable if a variable offset can be applied on the device. Note that this is different to calibbias which is for devices (or drivers) that apply offsets to compensate for variation between different instances of the part, typically adjusted by using some hardware supported calibration procedure. Calibbias is applied internally, offset is applied in userspace to the _raw output.

/sys/bus/iio/devices/iio:deviceX/in_accel_raw_range

Defined on file *sysfs-bus-iio-bno055*

Raw (unscaled) range for acceleration readings. Unit after application of scale is m/s^2. Note that this doesn't affects the scale (which should be used when changing the maximum and minimum readable value affects also the reading scaling factor).

/sys/bus/iio/devices/iio:deviceX/in_accel_raw_range_available

Defined on file *sysfs-bus-iio-bno055*

List of allowed values for in_accel_raw_range attribute

/sys/bus/iio/devices/iio:deviceX/in_accel_scale_available
/sys/.../iio:deviceX/in_anglvel_scale_available
/sys/.../iio:deviceX/in_magn_scale_available
/sys/.../iio:deviceX/in_illuminance_scale_available
/sys/.../iio:deviceX/in_intensity_scale_available
/sys/.../iio:deviceX/in_proximity_scale_available
/sys/.../iio:deviceX/in_voltageX_scale_available
/sys/.../iio:deviceX/in_voltage-voltage_scale_available
/sys/.../iio:deviceX/out_voltageX_scale_available
/sys/.../iio:deviceX/out_altvoltageX_scale_available
/sys/.../iio:deviceX/in_capacitance_scale_available
/sys/.../iio:deviceX/in_pressure_scale_available
/sys/.../iio:deviceX/in_pressureY_scale_available

Defined on file *sysfs-bus-iio*

If a discrete set of scale values is available, they are listed in this attribute. Unlike illumination, multiplying intensity by intensity_scale does not yield value with any standardized unit.

/sys/bus/iio/devices/iio:deviceX/in_accel_x_calibbias
/sys/bus/iio/devices/iio:deviceX/in_accel_y_calibbias
/sys/bus/iio/devices/iio:deviceX/in_accel_z_calibbias
/sys/bus/iio/devices/iio:deviceX/in_anglvel_x_calibbias
/sys/bus/iio/devices/iio:deviceX/in_anglvel_y_calibbias
/sys/bus/iio/devices/iio:deviceX/in_anglvel_z_calibbias
/sys/bus/iio/devices/iio:deviceX/in_illuminance0_calibbias
/sys/bus/iio/devices/iio:deviceX/in_proximity0_calibbias
/sys/bus/iio/devices/iio:deviceX/in_pressureY_calibbias
/sys/bus/iio/devices/iio:deviceX/in_pressure_calibbias

Defined on file *sysfs-bus-iio*

Hardware applied calibration offset (assumed to fix production inaccuracies). icm42600: For this device values are real physical offsets expressed in SI units (m/s^2 for accelerometers and rad/s for gyroscope)/

/sys/bus/iio/devices/iio:deviceX/in_accel_x_peak_raw
/sys/bus/iio/devices/iio:deviceX/in_accel_y_peak_raw
/sys/bus/iio/devices/iio:deviceX/in_accel_z_peak_raw
/sys/bus/iio/devices/iio:deviceX/in_humidityrelative_peak_raw
/sys/bus/iio/devices/iio:deviceX/in_temp_peak_raw

Defined on file *sysfs-bus-iio*

Highest value since some reset condition. These attributes allow access to this and are otherwise the direct equivalent of the <type>Y[_name]_raw attributes.

/sys/bus/iio/devices/iio:deviceX/in_accel_x_raw
/sys/bus/iio/devices/iio:deviceX/in_accel_y_raw
/sys/bus/iio/devices/iio:deviceX/in_accel_z_raw

Defined on file *sysfs-bus-iio*

Acceleration in direction x, y or z (may be arbitrarily assigned but should match other such assignments on device). Has all of the equivalent parameters as per voltageY. Units after application of scale and offset are m/s^2.

/sys/bus/iio/devices/iio:deviceX/in_accel_xyz_squared_peak_raw

Defined on file *sysfs-bus-iio*

A computed peak value based on the sum squared magnitude of the underlying value in the specified directions.

/sys/bus/iio/devices/iio:deviceX/in_activity_calibgender_available
/sys/bus/iio/devices/iio:deviceX/in_energy_calibgender_available
/sys/bus/iio/devices/iio:deviceX/in_distance_calibgender_available
/sys/bus/iio/devices/iio:deviceX/in_velocity_calibgender_available

Defined on file *sysfs-bus-iio*

Lists all available gender values (e.g.: male, female).

/sys/bus/iio/devices/iio:deviceX/in_activity_calibgender
/sys/bus/iio/devices/iio:deviceX/in_energy_calibgender
/sys/bus/iio/devices/iio:deviceX/in_distance_calibgender
/sys/bus/iio/devices/iio:deviceX/in_velocity_calibgender

Defined on file *sysfs-bus-iio*

Gender of the user (e.g.: male, female) used by some pedometers to compute the stride length, distance, speed and activity type.

/sys/bus/iio/devices/iio:deviceX/in_activity_calibheight
/sys/bus/iio/devices/iio:deviceX/in_energy_calibheight
/sys/bus/iio/devices/iio:deviceX/in_distance_calibheight
/sys/bus/iio/devices/iio:deviceX/in_velocity_calibheight

Defined on file *sysfs-bus-iio*

Height of the user (in meters) used by some pedometers to compute the stride length, distance, speed and activity type.

/sys/bus/iio/devices/iio:deviceX/in_allow_async_readout

Defined on file *sysfs-bus-iio-ina2xx-adc*

By default (value '0'), the capture thread checks for the Conversion Ready Flag to being set prior to committing a new value to the sample buffer. This synchronizes the in-chip conversion rate with the in-driver readout rate at the cost of an additional register read.

Writing '1' will remove the polling for the Conversion Ready Flags to save the additional i2c transaction, which will improve the bandwidth available for reading data. However, samples

can be occasionally skipped or repeated, depending on the beat between the capture and conversion rates.

/sys/bus/iio/devices/iio:deviceX/in_altvoltage0-1_i_calibphase

Defined on file *sysfs-bus-iio-frequency-admv1013*

Read/write unscaled value for the Local Oscillatior path quadrature I phase shift.

/sys/bus/iio/devices/iio:deviceX/in_altvoltage0-1_q_calibphase

Defined on file *sysfs-bus-iio-frequency-admv1013*

Read/write unscaled value for the Local Oscillatior path quadrature Q phase shift.

/sys/bus/iio/devices/iio:deviceX/in_altvoltage0_i_calibbias

Defined on file *sysfs-bus-iio-frequency-admv1013*

Read/write value for the Local Oscillatior Feedthrough Offset Calibration I Positive side.

/sys/bus/iio/devices/iio:deviceX/in_altvoltage0_i_calibscale_coarse

Defined on file *sysfs-bus-iio-frequency-admv1014*

Read/write value for the digital attenuator gain (IF_I) with coarse steps.

/sys/bus/iio/devices/iio:deviceX/in_altvoltage0_i_calibscale_fine

Defined on file *sysfs-bus-iio-frequency-admv1014*

Read/write value for the digital attenuator gain (IF_I) with fine steps.

/sys/bus/iio/devices/iio:deviceX/in_altvoltage0_q_calibbias

Defined on file *sysfs-bus-iio-frequency-admv1013*

Read/write value for the Local Oscillatior Feedthrough Offset Calibration Q Positive side.

/sys/bus/iio/devices/iio:deviceX/in_altvoltage0_q_calibscale_coarse

Defined on file *sysfs-bus-iio-frequency-admv1014*

Read/write value for the digital attenuator gain (IF_Q) with coarse steps.

/sys/bus/iio/devices/iio:deviceX/in_altvoltage0_q_calibscale_fine

Defined on file *sysfs-bus-iio-frequency-admv1014*

Read/write value for the digital attenuator gain (IF_Q) with fine steps.

/sys/bus/iio/devices/iio:deviceX/in_altvoltage1_i_calibbias

Defined on file *sysfs-bus-iio-frequency-admv1013*

Read/write raw value for the Local Oscillatior Feedthrough Offset Calibration I Negative side.

/sys/bus/iio/devices/iio:deviceX/in_altvoltage1_q_calibbias

Defined on file *sysfs-bus-iio-frequency-admv1013*

Read/write raw value for the Local Oscillatior Feedthrough Offset Calibration Q Negative side.

/sys/bus/iio/devices/iio:deviceX/in_altvoltageY_compare_interval

Defined on file *sysfs-bus-iio-adc-envelope-detector*

Number of milliseconds to wait for the comparator in each step of the binary search for the input peak level. Needs to relate to the frequency of the input signal.

/sys/bus/iio/devices/iio:deviceX/in_altvoltageY_invert

Defined on file *sysfs-bus-iio-adc-envelope-detector*

The DAC is used to find the peak level of an alternating voltage input signal by a binary search using the output of a comparator wired to an interrupt pin. Like so:

```
                      _
                     | \
input +------>-------|+ \
                     |   \
       .-------.     |    }---.
       |       |     |   /    |
       |    dac|-->--|- /     |
       |       |     |_/      |
       |       |              |
       |       |              |
       |    irq|------<-------'
       |       |
       '-------'
```

The boolean invert attribute (0/1) should be set when the input signal is centered around the maximum value of the dac instead of zero. The envelope detector will search from below in this case and will also invert the result.

The edge/level of the interrupt is also switched to its opposite value.

/sys/bus/iio/devices/iio:deviceX/in_anglY_label

Defined on file *sysfs-bus-iio*

Optional symbolic label for channel Y. For Intel hid hinge sensor, the label values are: hinge, keyboard, screen. It means the three channels each correspond respectively to hinge angle, keyboard angle, and screen angle.

/sys/bus/iio/devices/iio:deviceX/in_angl_raw
/sys/bus/iio/devices/iio:deviceX/in_anglY_raw

Defined on file *sysfs-bus-iio*

Angle of rotation. Units after application of scale and offset are radians.

/sys/bus/iio/devices/iio:deviceX/in_anglvel_raw_range

Defined on file *sysfs-bus-iio-bno055*

Range for angular velocity readings in radians per second. Note that this does not affects the scale (which should be used when changing the maximum and minimum readable value affects also the reading scaling factor).

/sys/bus/iio/devices/iio:deviceX/in_anglvel_raw_range_available

Defined on file *sysfs-bus-iio-bno055*

List of allowed values for in_anglvel_raw_range attribute

/sys/bus/iio/devices/iio:deviceX/in_anglvel_x_raw
/sys/bus/iio/devices/iio:deviceX/in_anglvel_y_raw
/sys/bus/iio/devices/iio:deviceX/in_anglvel_z_raw

Defined on file *sysfs-bus-iio*

Angular velocity about axis x, y or z (may be arbitrarily assigned). Has all the equivalent parameters as per voltageY. Units after application of scale and offset are radians per second.

/sys/bus/iio/devices/iio:deviceX/in_capacitanceY_raw

Defined on file *sysfs-bus-iio*

Raw capacitance measurement from channel Y. Units after application of scale and offset are nanofarads.

/sys/bus/iio/devices/iio:deviceX/in_chromaticity_x_raw
/sys/bus/iio/devices/iio:deviceX/in_chromaticity_y_raw

Defined on file *sysfs-bus-iio*

The x and y light color coordinate on the CIE 1931 chromaticity diagram.

/sys/bus/iio/devices/iio:deviceX/in_colortemp_raw

Defined on file *sysfs-bus-iio*

Represents light color temperature, which measures light color temperature in Kelvin.

/sys/bus/iio/devices/iio:deviceX/in_concentration_VOC_short_raw

Defined on file *sysfs-bus-iio-chemical-vz89x*

Get the raw calibration VOC value from the sensor. This value has little application outside of calibration.

/sys/bus/iio/devices/iio:deviceX/in_concentration_co2_calibration_background

Defined on file *sysfs-bus-iio-chemical-sunrise-co2*

Writing '1' triggers a 'Background' calibration cycle.

/sys/bus/iio/devices/iio:deviceX/in_concentration_co2_calibration_factory

Defined on file *sysfs-bus-iio-chemical-sunrise-co2*

Writing '1' triggers a 'Factory' calibration cycle.

/sys/bus/iio/devices/iio:deviceX/in_concentration_raw
/sys/bus/iio/devices/iio:deviceX/in_concentrationX_raw
/sys/bus/iio/devices/iio:deviceX/in_concentration_co2_raw
/sys/bus/iio/devices/iio:deviceX/in_concentrationX_co2_raw
/sys/bus/iio/devices/iio:deviceX/in_concentration_ethanol_raw
/sys/bus/iio/devices/iio:deviceX/in_concentrationX_ethanol_raw
/sys/bus/iio/devices/iio:deviceX/in_concentration_h2_raw
/sys/bus/iio/devices/iio:deviceX/in_concentrationX_h2_raw
/sys/bus/iio/devices/iio:deviceX/in_concentration_o2_raw
/sys/bus/iio/devices/iio:deviceX/in_concentrationX_o2_raw
/sys/bus/iio/devices/iio:deviceX/in_concentration_voc_raw
/sys/bus/iio/devices/iio:deviceX/in_concentrationX_voc_raw

Defined on file *sysfs-bus-iio*

Raw (unscaled no offset etc.) reading of a substance. Units after application of scale and offset are percents.

/sys/bus/iio/devices/iio:deviceX/in_conversion_mode

Defined on file *sysfs-bus-iio-vf610*

Specifies the hardware conversion mode used. The three available modes are ”normal”, ”high-speed” and ”low-power”, where the last is the default mode.

/sys/bus/iio/devices/iio:deviceX/in_count0_enable_mode

Defined on file *sysfs-bus-iio-timer-stm32*

Configure the device counter enable modes, in all case counting direction is set by in_count0_count_direction attribute and the counter is clocked by the internal clock.

always:
Counter is always ON.

gated:
Counting is enabled when connected trigger signal level is high else counting is disabled.

triggered:
Counting is enabled on rising edge of the connected trigger, and remains enabled for the duration of this selected mode.

/sys/bus/iio/devices/iio:deviceX/in_count0_preset

Defined on file *sysfs-bus-iio-timer-stm32*

Reading returns the current preset value. Writing sets the preset value. When counting up the counter starts from 0 and fires an event when reach preset value. When counting down the counter start from preset value and fire event when reach 0.

/sys/bus/iio/devices/iio:deviceX/in_count0_trigger_mode

Defined on file *sysfs-bus-iio-timer-stm32*

Configure the device counter trigger mode counting direction is set by in_count0_count_direction attribute and the counter is clocked by the connected trigger rising edges.

/sys/bus/iio/devices/iio:deviceX/in_countY_count_direction

Defined on file *sysfs-bus-iio*

This interface is deprecated; please use the Counter subsystem.

Raw counter device counters direction for channel Y.

/sys/bus/iio/devices/iio:deviceX/in_countY_raw

Defined on file *sysfs-bus-iio*

This interface is deprecated; please use the Counter subsystem.

Raw counter device counts from channel Y. For quadrature counters, multiplication by an available [Y]_scale results in the counts of a single quadrature signal phase from channel Y.

/sys/bus/iio/devices/iio:deviceX/in_count_count_direction_available

Defined on file *sysfs-bus-iio*

This interface is deprecated; please use the Counter subsystem.

A list of possible counting directions which are:

- ”up” : counter device is increasing.
- ”down”: counter device is decreasing.

/sys/bus/iio/devices/iio:deviceX/in_count_enable_mode_available

Defined on file *sysfs-bus-iio-timer-stm32*

Reading returns the list possible enable modes.

/sys/bus/iio/devices/iio:deviceX/in_count_trigger_mode_available

Defined on file *sysfs-bus-iio-timer-stm32*

Reading returns the list possible trigger modes.

/sys/bus/iio/devices/iio:deviceX/in_current5_raw

Defined on file *sysfs-bus-iio-adc-mt6360*

Indicated MT6360 IBUS ADC Calculating with scale and offset returns voltage in uA

/sys/bus/iio/devices/iio:deviceX/in_current6_raw

Defined on file *sysfs-bus-iio-adc-mt6360*

Indicated MT6360 IBAT ADC Calculating with scale and offset returns voltage in uA

/sys/bus/iio/devices/iio:deviceX/in_current7_raw

Defined on file *sysfs-bus-iio-adc-mt6360*

Indicated MT6360 CHG_VDDP ADC Calculating with scale and offset returns voltage in uV

/sys/bus/iio/devices/iio:deviceX/in_currentX_raw
/sys/bus/iio/devices/iio:deviceX/in_currentX_i_raw
/sys/bus/iio/devices/iio:deviceX/in_currentX_q_raw

Defined on file *sysfs-bus-iio*

Raw current measurement from channel X. Units are in milliamps after application of scale and offset. If no offset or scale is present, output should be considered as processed with the unit in milliamps.

Channels with 'i' and 'q' modifiers always exist in pairs and both channels refer to the same signal. The 'i' channel contains the in-phase component of the signal while the 'q' channel contains the quadrature component.

/sys/bus/iio/devices/iio:deviceX/in_currentY_raw
/sys/bus/iio/devices/iio:deviceX/in_currentY_supply_raw

Defined on file *sysfs-bus-iio*

Raw (unscaled no bias removal etc.) current measurement from channel Y. In special cases where the channel does not correspond to externally available input one of the named versions may be used. The number must always be specified and unique to allow association with event codes. Units after application of scale and offset are milliamps.

/sys/bus/iio/devices/iio:deviceX/in_current_shunt_resistor

Defined on file *sysfs-bus-iio-adc-max9611*

The value of the shunt resistor used to compute current flowing between RS+ and RS- voltage sense inputs. In Ohms.

These attributes describe a single physical component, exposed as two distinct attributes as it is used to calculate two different values: power load and current flowing between RS+ and RS- inputs.

/sys/bus/iio/devices/iio:deviceX/in_deltaangl_x_raw
/sys/bus/iio/devices/iio:deviceX/in_deltaangl_y_raw
/sys/bus/iio/devices/iio:deviceX/in_deltaangl_z_raw

Defined on file *sysfs-bus-iio*

Angular displacement between two consecutive samples on x, y or z (may be arbitrarily assigned but should match other such assignments on device). In order to compute the total angular displacement during a desired period of time, the application should sum-up the delta angle samples acquired during that time. Units after application of scale and offset are radians.

/sys/bus/iio/devices/iio:deviceX/in_deltavelocity_x_raw
/sys/bus/iio/devices/iio:deviceX/in_deltavelocity_y_raw
/sys/bus/iio/devices/iio:deviceX/in_deltavelocity_z_raw

Defined on file *sysfs-bus-iio*

The linear velocity change between two consecutive samples on x, y or z (may be arbitrarily assigned but should match other such assignments on device). In order to compute the total linear velocity change during a desired period of time, the application should sum-up the delta velocity samples acquired during that time. Units after application of scale and offset are meters per second.

/sys/bus/iio/devices/iio:deviceX/in_electricalconductivity_raw

Defined on file *sysfs-bus-iio*

Raw (unscaled no offset etc.) electric conductivity reading. Units after application of scale and offset are siemens per meter.

/sys/bus/iio/devices/iio:deviceX/in_energy_calibweight

Defined on file *sysfs-bus-iio*

Weight of the user (in kg). It is needed by some pedometers to compute the calories burnt by the user.

/sys/bus/iio/devices/iio:deviceX/in_filter_notch_center_frequency

Defined on file *sysfs-bus-iio*

Center frequency in Hz for a notch filter. Used i.e. for line noise suppression.

/sys/bus/iio/devices/iio:deviceX/in_gravity_x_raw
/sys/bus/iio/devices/iio:deviceX/in_gravity_y_raw
/sys/bus/iio/devices/iio:deviceX/in_gravity_z_raw

Defined on file *sysfs-bus-iio*

Gravity in direction x, y or z (may be arbitrarily assigned but should match other such assignments on device). Units after application of scale and offset are m/s^2.

/sys/bus/iio/devices/iio:deviceX/in_gyro_calibration_auto_status

Defined on file *sysfs-bus-iio-bno055*

Reports the autocalibration status for the gyroscope sensor. Can be 0 (calibration non even enabled) or 1 to 5 where the greater the number, the better the calibration status.

/sys/bus/iio/devices/iio:deviceX/in_gyro_matrix
/sys/bus/iio/devices/iio:deviceX/in_accel_matrix
/sys/bus/iio/devices/iio:deviceX/in_magn_matrix

Defined on file *sysfs-bus-iio-mpu6050*

This is mounting matrix for motion sensors. Mounting matrix is a 3x3 unitary matrix. A typical mounting matrix would look like [0, 1, 0; 1, 0, 0; 0, 0, -1]. Using this information, it would be easy to tell the relative positions among sensors as well as their positions relative to the board that holds these sensors. Identity matrix [1, 0, 0; 0, 1, 0; 0, 0, 1] means sensor chip and device are perfectly aligned with each other. All axes are exactly the same.

/sys/bus/iio/devices/iio:deviceX/in_humidityrelative_input

Defined on file *sysfs-bus-iio*

Scaled humidity measurement in milli percent.

/sys/bus/iio/devices/iio:deviceX/in_humidityrelative_raw

Defined on file *sysfs-bus-iio*

Raw humidity measurement of air. Units after application of scale and offset are milli percent.

/sys/bus/iio/devices/iio:deviceX/in_humidityrelative_trough_raw
/sys/bus/iio/devices/iio:deviceX/in_temp_trough_raw

Defined on file *sysfs-bus-iio*

Lowest value since some reset condition. These attributes allow access to this and are otherwise the direct equivalent of the <type>Y[_name]_raw attributes.

/sys/bus/iio/devices/iio:deviceX/in_illuminance0_zone

Defined on file *sysfs-bus-iio-light-lm3533-als*

Get the current light zone (0..4) as defined by the in_illuminance0_threshY_{falling,rising} thresholds.

/sys/bus/iio/devices/iio:deviceX/in_illuminance_hysteresis_relative
/sys/bus/iio/devices/iio:deviceX/in_intensity_hysteresis_relative

Defined on file *sysfs-bus-iio*

Specify the percent for light sensor relative to the channel absolute value that a data field should change before an event is generated. Units are a percentage of the prior reading.

/sys/bus/iio/devices/iio:deviceX/in_incli_x_raw
/sys/bus/iio/devices/iio:deviceX/in_incli_y_raw
/sys/bus/iio/devices/iio:deviceX/in_incli_z_raw

Defined on file *sysfs-bus-iio*

Inclination raw reading about axis x, y or z (may be arbitrarily assigned). Data converted by application of offset and scale to degrees.

/sys/bus/iio/devices/iio:deviceX/in_indexY_raw

Defined on file *sysfs-bus-iio*

This interface is deprecated; please use the Counter subsystem.

Raw counter device index value from channel Y. This attribute provides an absolute positional reference (e.g. a pulse once per revolution) which may be used to home positional systems as required.

/sys/bus/iio/devices/iio:deviceX/in_intensityY_offset

Defined on file *sysfs-bus-iio-health-afe440x*

Get and set the offset cancellation DAC setting for these stages. The values are expressed in 5-bit sign-magnitude.

/sys/bus/iio/devices/iio:deviceX/in_intensityY_raw

Defined on file *sysfs-bus-iio-health-afe440x*

Get measured values from the ADC for these stages. Y is the specific stage number corresponding to datasheet stage names as follows:

1	LED2
2	ALED2/LED3
3	LED1
4	ALED1/LED4

Note that channels 5 and 6 represent LED2-ALED2 and LED1-ALED1 respectively which simply helper channels containing the calculated difference in the value of stage 1 - 2 and 3 - 4. The

values are expressed in 24-bit twos complement. The LED current for the stage is controlled via out_currentY_raw.

/sys/bus/iio/devices/iio:deviceX/in_intensityY_resistance
/sys/bus/iio/devices/iio:deviceX/in_intensityY_capacitance

Defined on file *sysfs-bus-iio-health-afe440x*

Get and set the resistance and the capacitance settings for the Transimpedance Amplifier during the associated stage.

/sys/bus/iio/devices/iio:deviceX/in_intensity_hardwaregain_available

Defined on file *sysfs-bus-iio*

Lists all available hardware applied gain factors. Shared across all channels.

/sys/bus/iio/devices/iio:deviceX/in_intensity_ir_large_raw

Defined on file *sysfs-bus-iio-light-si1133*

Unit-less infrared intensity. The intensity is measured from 4 dark photodiodes. "large" indicate the surface area capturing infrared.

/sys/bus/iio/devices/iio:deviceX/in_intensity_ir_small_raw

Defined on file *sysfs-bus-iio-light-si1133*

Unit-less infrared intensity. The intensity is measured from 1 dark photodiode. "small" indicate the surface area capturing infrared.

/sys/bus/iio/devices/iio:deviceX/in_intensity_large_raw

Defined on file *sysfs-bus-iio-light-si1133*

Unit-less light intensity with more diodes.

/sys/bus/iio/devices/iio:deviceX/in_intensity_x_raw
/sys/bus/iio/devices/iio:deviceX/in_intensity_y_raw
/sys/bus/iio/devices/iio:deviceX/in_intensity_z_raw

Defined on file *sysfs-bus-iio*

Unscaled light intensity according to CIE 1931/DIN 5033 color space. Units after application of scale are nano nanowatts per square meter.

/sys/bus/iio/devices/iio:deviceX/in_magn_calibration_auto_status

Defined on file *sysfs-bus-iio-bno055*

Reports the autocalibration status for the magnetometer sensor. Can be 0 (calibration non even enabled) or 1 to 5 where the greater the number, the better the calibration status.

/sys/bus/iio/devices/iio:deviceX/in_magn_calibration_fast_enable

Defined on file *sysfs-bus-iio-bno055*

Can be 1 or 0. Enables/disables the "Fast Magnetometer Calibration" HW function.

/sys/bus/iio/devices/iio:deviceX/in_magn_x_oversampling_ratio
/sys/bus/iio/devices/iio:deviceX/in_magn_y_oversampling_ratio
/sys/bus/iio/devices/iio:deviceX/in_magn_z_oversampling_ratio

Defined on file *sysfs-bus-iio*

Hardware applied number of measurements for acquiring one data point. The HW will do <type>[_name]_oversampling_ratio measurements and return the average value as output data. Each value resulted from <type>[_name]_oversampling_ratio measurements is considered as one sample for <type>[_name]_sampling_frequency.

/sys/bus/iio/devices/iio:deviceX/in_magn_x_raw
/sys/bus/iio/devices/iio:deviceX/in_magn_y_raw
/sys/bus/iio/devices/iio:deviceX/in_magn_z_raw

Defined on file *sysfs-bus-iio*

Magnetic field along axis x, y or z (may be arbitrarily assigned). Data converted by application of offset then scale to Gauss.

/sys/bus/iio/devices/iio:deviceX/in_massconcentration_pm1_input
/sys/bus/iio/devices/iio:deviceX/in_massconcentrationY_pm1_input
/sys/bus/iio/devices/iio:deviceX/in_massconcentration_pm2p5_input
/sys/bus/iio/devices/iio:deviceX/in_massconcentrationY_pm2p5_input
/sys/bus/iio/devices/iio:deviceX/in_massconcentration_pm4_input
/sys/bus/iio/devices/iio:deviceX/in_massconcentrationY_pm4_input
/sys/bus/iio/devices/iio:deviceX/in_massconcentration_pm10_input
/sys/bus/iio/devices/iio:deviceX/in_massconcentrationY_pm10_input

Defined on file *sysfs-bus-iio*

Mass concentration reading of particulate matter in ug / m3. pmX consists of particles with aerodynamic diameter less or equal to X micrometers.

/sys/bus/iio/devices/iio:deviceX/in_ph_raw

Defined on file *sysfs-bus-iio*

Raw (unscaled no offset etc.) pH reading of a substance as a negative base-10 logarithm of hydrodium ions in a litre of water.

/sys/bus/iio/devices/iio:deviceX/in_phaseY_raw

Defined on file *sysfs-bus-iio*

Raw (unscaled) phase difference reading from channel Y. Units after application of scale and offset are radians.

/sys/bus/iio/devices/iio:deviceX/in_positionrelative_x_raw
/sys/bus/iio/devices/iio:deviceX/in_positionrelative_y_raw

Defined on file *sysfs-bus-iio*

Relative position in direction x or y on a pad (may be arbitrarily assigned but should match other such assignments on device). Units after application of scale and offset are milli percents from the pad's size in both directions. Should be calibrated by the consumer.

/sys/bus/iio/devices/iio:deviceX/in_powerY_raw

Defined on file *sysfs-bus-iio*

Raw (unscaled no bias removal etc.) power measurement from channel Y. The number must always be specified and unique to allow association with event codes. Units after application of scale and offset are milliwatts.

/sys/bus/iio/devices/iio:deviceX/in_power_shunt_resistor

Defined on file *sysfs-bus-iio-adc-max9611*

The value of the shunt resistor used to compute power drain on common input voltage pin (RS+). In Ohms.

/sys/bus/iio/devices/iio:deviceX/in_pressureY_input
/sys/bus/iio/devices/iio:deviceX/in_pressure_input

Defined on file *sysfs-bus-iio*

Scaled pressure measurement from channel Y, in kilopascal.

/sys/bus/iio/devices/iio:deviceX/in_pressureY_raw
/sys/bus/iio/devices/iio:deviceX/in_pressure_raw

Defined on file *sysfs-bus-iio*

Raw pressure measurement from channel Y. Units after application of scale and offset are kilopascal.

/sys/bus/iio/devices/iio:deviceX/in_proximity0_agc_gain
/sys/bus/iio/devices/iio:deviceX/in_proximity0_agc_gain_bias

Defined on file *sysfs-bus-iio-isl29501*

This sensor has an automatic gain control (agc) loop which sets the analog signal levels at an optimum level by controlling programmable gain amplifiers. The criteria for optimal gain is determined by the sensor.

Return the actual gain value as an integer in [0; 65536] range when read from.

The agc gain read when measuring crosstalk shall be written into in_proximity0_agc_gain_bias.

/sys/bus/iio/devices/iio:deviceX/in_proximity0_calib_phase_temp_a
/sys/bus/iio/devices/iio:deviceX/in_proximity0_calib_phase_temp_b
/sys/bus/iio/devices/iio:deviceX/in_proximity0_calib_phase_light_a
/sys/bus/iio/devices/iio:deviceX/in_proximity0_calib_phase_light_b

Defined on file *sysfs-bus-iio-isl29501*

The sensor is able to perform correction of distance measurements due to changing temperature and ambient light conditions. It can be programmed to correct for a second order error polynomial.

Phase data has to be collected when temperature and ambient light are modulated independently.

Then a least squares curve fit to a second order polynomial has to be generated from the data. The resultant curves have the form ax^2 + bx + c.

From those two curves, a and b coefficients shall be stored in in_proximity0_calib_phase_temp_a and in_proximity0_calib_phase_temp_b for temperature and in in_proximity0_calib_phase_light_a and in_proximity0_calib_phase_light_b for ambient light.

Those values must be integer in [0; 8355840] range.

Finally, the c constant is set by the sensor internally.

The value stored in sensor is displayed when read from.

/sys/bus/iio/devices/iio:deviceX/in_proximity3_comb_raw

Defined on file *sysfs-bus-iio-sx9310*

Proximity measurement indicating that some object is near the combined sensor. The combined sensor presents proximity measurements constructed by hardware by combining measurements taken from a given set of physical sensors.

/sys/bus/iio/devices/iio:deviceX/in_proximity<id>_setup

Defined on file *sysfs-bus-iio-sx9324*

SX9324 has 3 inputs, CS0, CS1 and CS2. Hardware layout defines if the input is

- not connected (HZ),
- grounded (GD),
- connected to an antenna where it can act as a base (DS - data shield), or measured input (MI).

The sensor rotates measurement across 4 phases (PH0, PH1, PH2, PH3), where the inputs are configured and then measured.

By default, during the first phase, [PH0], CS0 is measured, while CS1 and CS2 are used as shields. *cat in_proximity0_setup* returns "MI,DS,DS". [PH1], CS1 is measured, CS0 and CS2 are shield: *cat in_proximity1_setup* returns "DS,MI,DS". [PH2], CS2 is measured, CS0 and CS1

are shield: *cat in_proximity1_setup* returns "DS,DS,MI". [PH3], CS1 and CS2 are measured (combo mode): *cat in_proximity1_setup* returns "DS,MI,MI".

Note, these are the chip default. Hardware layout will most likely dictate different output. The entry is read-only.

/sys/bus/iio/devices/iio:deviceX/in_proximity_input

Defined on file *sysfs-bus-iio-proximity-as3935*

Get the current distance in meters of storm (1km steps) 1000-40000 = distance in meters

What /sys/bus/iio/devices/iio:deviceX/noise_level_tripped When 1 the noise level is over the trip level and not reporting valid data

/sys/bus/iio/devices/iio:deviceX/in_proximity_nearlevel

Defined on file *sysfs-bus-iio-proximity*

Near level for proximity sensors. This is a single integer value that tells user space when an object should be considered close to the device. If the value read from the sensor is above or equal to the value in this file an object should typically be considered near.

/sys/bus/iio/devices/iio:deviceX/in_resistance_calibbias

Defined on file *sysfs-bus-iio-chemical-sgp40*

Set the bias value for the resistance which is used for calculation of in_concentration_input as follows:

x = (in_resistance_raw - in_resistance_calibbias) * 0.65

in_concentration_input = 500 / (1 + e^x)

Default value: 30000

/sys/bus/iio/devices/iio:deviceX/in_resistance_raw
/sys/bus/iio/devices/iio:deviceX/in_resistanceX_raw
/sys/bus/iio/devices/iio:deviceX/out_resistance_raw
/sys/bus/iio/devices/iio:deviceX/out_resistanceX_raw

Defined on file *sysfs-bus-iio*

Raw (unscaled no offset etc.) resistance reading. Units after application of scale and offset are ohms.

/sys/bus/iio/devices/iio:deviceX/in_rot_from_north_magnetic_tilt_comp_raw
/sys/bus/iio/devices/iio:deviceX/in_rot_from_north_true_tilt_comp_raw
/sys/bus/iio/devices/iio:deviceX/in_rot_from_north_magnetic_raw
/sys/bus/iio/devices/iio:deviceX/in_rot_from_north_true_raw

Defined on file *sysfs-bus-iio*

Raw value of rotation from true/magnetic north measured with or without compensation from tilt sensors.

/sys/bus/iio/devices/iio:deviceX/in_rot_quaternion_raw

Defined on file *sysfs-bus-iio*

Raw value of quaternion components using a format x y z w. Here x, y, and z component represents the axis about which a rotation will occur and w component represents the amount of rotation.

/sys/bus/iio/devices/iio:deviceX/in_rot_yaw_raw
/sys/bus/iio/devices/iio:deviceX/in_rot_pitch_raw
/sys/bus/iio/devices/iio:deviceX/in_rot_roll_raw

Defined on file *sysfs-bus-iio*

Raw (unscaled) euler angles readings. Units after application of scale are deg.

/sys/bus/iio/devices/iio:deviceX/in_shunt_resistor

Defined on file *sysfs-bus-iio-ina2xx-adc*

The value of the shunt resistor may be known only at runtime fom an eeprom content read by a client application. This attribute allows to set its value in ohms.

/sys/bus/iio/devices/iio:deviceX/in_temp8_raw

Defined on file *sysfs-bus-iio-adc-mt6360*

Indicated MT6360 IC junction temperature Calculating with scale and offset returns temperature in degree

/sys/bus/iio/devices/iio:deviceX/in_tempX_input

Defined on file *sysfs-bus-iio*

Scaled temperature measurement in milli degrees Celsius.

/sys/bus/iio/devices/iio:deviceX/in_temp_calibemissivity
/sys/bus/iio/devices/iio:deviceX/in_tempX_calibemissivity
/sys/bus/iio/devices/iio:deviceX/in_temp_object_calibemissivity
/sys/bus/iio/devices/iio:deviceX/in_tempX_object_calibemissivity

Defined on file *sysfs-bus-iio*

The emissivity ratio of the surface in the field of view of the contactless temperature sensor. Emissivity varies from 0 to 1, with 1 being the emissivity of a black body.

/sys/bus/iio/devices/iio:deviceX/in_temp_object_calibambient
/sys/bus/iio/devices/iio:deviceX/in_tempX_object_calibambient

Defined on file *sysfs-bus-iio*

Calibrated ambient temperature for object temperature calculation in milli degrees Celsius.

/sys/bus/iio/devices/iio:deviceX/in_temp_raw
/sys/bus/iio/devices/iio:deviceX/in_tempX_raw
/sys/bus/iio/devices/iio:deviceX/in_temp_x_raw
/sys/bus/iio/devices/iio:deviceX/in_temp_y_raw
/sys/bus/iio/devices/iio:deviceX/in_temp_ambient_raw
/sys/bus/iio/devices/iio:deviceX/in_temp_object_raw

Defined on file *sysfs-bus-iio*

Raw (unscaled no bias removal etc.) temperature measurement. If an axis is specified it generally means that the temperature sensor is associated with one part of a compound device (e.g. a gyroscope axis). The ambient and object modifiers distinguish between ambient (reference) and distant temperature for contact- less measurements. Units after application of scale and offset are milli degrees Celsius.

/sys/bus/iio/devices/iio:deviceX/in_temp_thermocouple_type

Defined on file *sysfs-bus-iio*

One of the following thermocouple types: B, E, J, K, N, R, S, T.

/sys/bus/iio/devices/iio:deviceX/in_voltage-voltage_filter_mode_available

Defined on file *sysfs-bus-iio-adc-ad4130*

Reading returns a list with the possible filter modes.

- ”sinc4” - Sinc 4. Excellent noise performance. Long 1st conversion time. No natural 50/60Hz rejection.
- ”sinc4+sinc1” - Sinc4 + averaging by 8. Low 1st conversion time.
- ”sinc3” - Sinc3. Moderate 1st conversion time. Good noise performance.
- ”sinc3+rej60” - Sinc3 + 60Hz rejection. At a sampling frequency of 50Hz, achieves simultaneous 50Hz and 60Hz rejection.
- ”sinc3+sinc1” - Sinc3 + averaging by 8. Low 1st conversion time. Best used with a sampling frequency of at least 216.19Hz.
- ”sinc3+pf1” - Sinc3 + Post Filter 1. 53dB rejection @ 50Hz, 58dB rejection @ 60Hz.
- ”sinc3+pf2” - Sinc3 + Post Filter 2. 70dB rejection @ 50Hz, 70dB rejection @ 60Hz.
- ”sinc3+pf3” - Sinc3 + Post Filter 3. 99dB rejection @ 50Hz, 103dB rejection @ 60Hz.
- ”sinc3+pf4” - Sinc3 + Post Filter 4. 103dB rejection @ 50Hz, 109dB rejection @ 60Hz.

/sys/bus/iio/devices/iio:deviceX/in_voltage0_raw

Defined on file *sysfs-bus-iio-adc-mt6360*

Indicated MT6360 USBID ADC which connected to connector ID pin. Calculating with scale and offset returns voltage in uV

/sys/bus/iio/devices/iio:deviceX/in_voltage10_raw

Defined on file *sysfs-bus-iio-adc-mt6360*

Indicated MT6360 TS ADC Calculating with scale and offset returns voltage in uV

/sys/bus/iio/devices/iio:deviceX/in_voltage1_raw

Defined on file *sysfs-bus-iio-adc-mt6360*

Indicated MT6360 VBUS ADC with lower accuracy(+-75mA) higher measure range(1~22mV) Calculating with scale and offset returns voltage in uV

/sys/bus/iio/devices/iio:deviceX/in_voltage2-voltage2_shorted_raw

Defined on file *sysfs-bus-iio-adc-ad7192*

Measure voltage from AIN2 pin connected to AIN(+) and AIN(-) shorted.

/sys/bus/iio/devices/iio:deviceX/in_voltage2_raw

Defined on file *sysfs-bus-iio-adc-mt6360*

Indicated MT6360 VBUS ADC with higher accuracy(+-30mA) lower measure range(1~9.76V) Calculating with scale and offset returns voltage in uV

/sys/bus/iio/devices/iio:deviceX/in_voltage3_raw

Defined on file *sysfs-bus-iio-adc-mt6360*

Indicated MT6360 VSYS ADC Calculating with scale and offset returns voltage in uV

/sys/bus/iio/devices/iio:deviceX/in_voltage4_raw

Defined on file *sysfs-bus-iio-adc-mt6360*

Indicated MT6360 VBAT ADC Calculating with scale and offset returns voltage in uV

/sys/bus/iio/devices/iio:deviceX/in_voltage9_raw

Defined on file *sysfs-bus-iio-adc-mt6360*

Indicated MT6360 VREF_TS ADC Calculating with scale and offset returns voltage in uV

/sys/bus/iio/devices/iio:deviceX/in_voltageX_sampling_frequency
/sys/bus/iio/devices/iio:deviceX/in_powerY_sampling_frequency
/sys/bus/iio/devices/iio:deviceX/in_currentZ_sampling_frequency

Defined on file *sysfs-bus-iio*

Some devices have separate controls of sampling frequency for individual channels. If multiple channels are enabled in a scan, then the sampling_frequency of the scan may be computed from the per channel sampling frequencies.

/sys/bus/iio/devices/iio:deviceX/in_voltageY-voltageZ_balance_switch_en

Defined on file *sysfs-bus-iio-adc-ad7280a*

Used to enable an output for balancing cells for time controlled via in_voltage_Y-voltageZ_balance_switch_timer.

/sys/bus/iio/devices/iio:deviceX/in_voltageY-voltageZ_balance_switch_timer

Defined on file *sysfs-bus-iio-adc-ad7280a*

Time in seconds for which balance switch will be turned on. Multiple of 71.5 seconds.

/sys/bus/iio/devices/iio:deviceX/in_voltageY-voltageZ_filter_mode

Defined on file *sysfs-bus-iio-adc-ad4130*

Set the filter mode of the differential channel. When the filter mode changes, the in_voltageY-voltageZ_sampling_frequency and in_voltageY-voltageZ_sampling_frequency_available attributes might also change to accommodate the new filter mode. If the current sampling frequency is out of range for the new filter mode, the sampling frequency will be changed to the closest valid one.

/sys/bus/iio/devices/iio:deviceX/in_voltageY-voltageZ_raw

Defined on file *sysfs-bus-iio*

Raw (unscaled) differential voltage measurement equivalent to channel Y - channel Z where these channel numbers apply to the physically equivalent inputs when non differential readings are separately available. In differential only parts, then all that is required is a consistent labeling. Units after application of scale and offset are millivolts.

/sys/bus/iio/devices/iio:deviceX/in_voltageY_calibscale
/sys/bus/iio/devices/iio:deviceX/in_voltageY_supply_calibscale
/sys/bus/iio/devices/iio:deviceX/in_voltageY_i_calibscale
/sys/bus/iio/devices/iio:deviceX/in_voltageY_q_calibscale
/sys/bus/iio/devices/iio:deviceX/in_voltage_i_calibscale
/sys/bus/iio/devices/iio:deviceX/in_voltage_q_calibscale
/sys/bus/iio/devices/iio:deviceX/in_altvoltage_calibscale
/sys/bus/iio/devices/iio:deviceX/in_voltage_calibscale
/sys/bus/iio/devices/iio:deviceX/in_accel_x_calibscale
/sys/bus/iio/devices/iio:deviceX/in_accel_y_calibscale
/sys/bus/iio/devices/iio:deviceX/in_accel_z_calibscale
/sys/bus/iio/devices/iio:deviceX/in_anglvel_x_calibscale
/sys/bus/iio/devices/iio:deviceX/in_anglvel_y_calibscale
/sys/bus/iio/devices/iio:deviceX/in_anglvel_z_calibscale
/sys/bus/iio/devices/iio:deviceX/in_illuminance0_calibscale
/sys/bus/iio/devices/iio:deviceX/in_proximity0_calibscale
/sys/bus/iio/devices/iio:deviceX/in_pressureY_calibscale
/sys/bus/iio/devices/iio:deviceX/in_pressure_calibscale
/sys/bus/iio/devices/iio:deviceX/in_illuminance_calibscale

Defined on file *sysfs-bus-iio*

Hardware applied calibration scale factor (assumed to fix production inaccuracies). If shared across all channels, <type>_calibscale is used.

/sys/bus/iio/devices/iio:deviceX/in_voltageY_dither_enable

Defined on file *sysfs-bus-iio-dac-ad5766*

Dither enable. Write 1 to enable dither or 0 to disable it.

/sys/bus/iio/devices/iio:deviceX/in_voltageY_dither_invert

Defined on file *sysfs-bus-iio-dac-ad5766*

Inverts the dither applied to the selected DAC channel. Dither is not inverted by default. Write "1" to invert dither.

/sys/bus/iio/devices/iio:deviceX/in_voltageY_dither_scale

Defined on file *sysfs-bus-iio-dac-ad5766*

Scales the dither before it is applied to the selected channel.

/sys/bus/iio/devices/iio:deviceX/in_voltageY_dither_scale_available

Defined on file *sysfs-bus-iio-dac-ad5766*

Returns possible scalings available for the current channel.

/sys/bus/iio/devices/iio:deviceX/in_voltageY_dither_source

Defined on file *sysfs-bus-iio-dac-ad5766*

Selects dither source applied to the selected channel. Write "0" to select N0 source, write "1" to select N1 source.

/sys/bus/iio/devices/iio:deviceX/in_voltageY_label
/sys/bus/iio/devices/iio:deviceX/out_voltageY_label

Defined on file *sysfs-bus-iio*

Optional symbolic label to a device channel. If a label is defined for this channel add that to the channel specific attributes. This is useful for userspace to be able to better identify an individual channel.

/sys/bus/iio/devices/iio:deviceX/in_voltageY_raw
/sys/bus/iio/devices/iio:deviceX/in_voltageY_supply_raw
/sys/bus/iio/devices/iio:deviceX/in_voltageY_i_raw
/sys/bus/iio/devices/iio:deviceX/in_voltageY_q_raw

Defined on file *sysfs-bus-iio*

Raw (unscaled no bias removal etc.) voltage measurement from channel Y. In special cases where the channel does not correspond to externally available input one of the named versions may be used. The number must always be specified and unique to allow association with event codes. Units after application of scale and offset are millivolts.

Channels with 'i' and 'q' modifiers always exist in pairs and both channels refer to the same signal. The 'i' channel contains the in-phase component of the signal while the 'q' channel contains the quadrature component.

/sys/bus/iio/devices/iio:deviceX/in_voltageY_scale
/sys/bus/iio/devices/iio:deviceX/in_voltageY_i_scale
/sys/bus/iio/devices/iio:deviceX/in_voltageY_q_scale
/sys/bus/iio/devices/iio:deviceX/in_voltageY_supply_scale
/sys/bus/iio/devices/iio:deviceX/in_voltage_scale
/sys/bus/iio/devices/iio:deviceX/in_voltage_i_scale
/sys/bus/iio/devices/iio:deviceX/in_voltage_q_scale
/sys/bus/iio/devices/iio:deviceX/in_voltage-voltage_scale
/sys/bus/iio/devices/iio:deviceX/out_voltageY_scale
/sys/bus/iio/devices/iio:deviceX/out_altvoltageY_scale
/sys/bus/iio/devices/iio:deviceX/in_currentY_scale
/sys/bus/iio/devices/iio:deviceX/in_currentY_supply_scale
/sys/bus/iio/devices/iio:deviceX/in_current_scale
/sys/bus/iio/devices/iio:deviceX/in_currentY_i_scale
/sys/bus/iio/devices/iio:deviceX/in_currentY_q_scale
/sys/bus/iio/devices/iio:deviceX/in_current_i_scale
/sys/bus/iio/devices/iio:deviceX/in_current_q_scale
/sys/bus/iio/devices/iio:deviceX/in_accel_scale
/sys/bus/iio/devices/iio:deviceX/in_accel_peak_scale
/sys/bus/iio/devices/iio:deviceX/in_anglvel_scale

continues on next page

Table 9 – continued from previous page

/sys/bus/iio/devices/iio:deviceX/in_energy_scale
/sys/bus/iio/devices/iio:deviceX/in_distance_scale
/sys/bus/iio/devices/iio:deviceX/in_magn_scale
/sys/bus/iio/devices/iio:deviceX/in_magn_x_scale
/sys/bus/iio/devices/iio:deviceX/in_magn_y_scale
/sys/bus/iio/devices/iio:deviceX/in_magn_z_scale
/sys/bus/iio/devices/iio:deviceX/in_rot_from_north_magnetic_scale
/sys/bus/iio/devices/iio:deviceX/in_rot_from_north_true_scale
/sys/bus/iio/devices/iio:deviceX/in_rot_from_north_magnetic_tilt_comp_scale
/sys/bus/iio/devices/iio:deviceX/in_rot_from_north_true_tilt_comp_scale
/sys/bus/iio/devices/iio:deviceX/in_pressureY_scale
/sys/bus/iio/devices/iio:deviceX/in_pressure_scale
/sys/bus/iio/devices/iio:deviceX/in_humidityrelative_scale
/sys/bus/iio/devices/iio:deviceX/in_velocity_sqrt(x^2+y^2+z^2)_scale
/sys/bus/iio/devices/iio:deviceX/in_illuminance_scale
/sys/bus/iio/devices/iio:deviceX/in_countY_scale
/sys/bus/iio/devices/iio:deviceX/in_deltaangl_scale
/sys/bus/iio/devices/iio:deviceX/in_deltavelocity_scale
/sys/bus/iio/devices/iio:deviceX/in_angl_scale
/sys/bus/iio/devices/iio:deviceX/in_intensity_x_scale
/sys/bus/iio/devices/iio:deviceX/in_intensity_y_scale
/sys/bus/iio/devices/iio:deviceX/in_intensity_z_scale
/sys/bus/iio/devices/iio:deviceX/in_concentration_co2_scale

Defined on file *sysfs-bus-iio*

If known for a device, scale to be applied to <type>Y[_name]_raw post addition of <type>[Y][_name]_offset in order to obtain the measured value in <type> units as specified in <type>[Y][_name]_raw documentation. If shared across all in channels then Y and <x|y|z> are not present and the value is called <type>[Y][_name]_scale. The peak modifier means this value is applied to <type>Y[_name]_peak_raw values.

/sys/bus/iio/devices/iio:deviceX/in_voltageY_sensing_mode

Defined on file *sysfs-bus-iio-adc-hi8435*

Program sensor type for threshold detector inputs. Could be either "GND-Open" or "Supply-Open" mode. Y is a threshold detector input channel. Channels 0..7, 8..15, 16..23 and 24..31 has common sensor types.

/sys/bus/iio/devices/iio:deviceX/in_voltage_filterY_notch_center

Defined on file *sysfs-bus-iio-adc-max11410*

Center frequency of the notch filter in Hz.

/sys/bus/iio/devices/iio:deviceX/in_voltage_filterY_notch_en

Defined on file *sysfs-bus-iio-adc-max11410*

Enable or disable a notch filter.

/sys/bus/iio/devices/iio:deviceX/in_voltage_spi_clk_freq

Defined on file *sysfs-bus-iio-dfsdm-adc-stm32*

For audio purpose only.

Used by audio driver to set/get the spi input frequency.

This is mandatory if DFSDM is slave on SPI bus, to provide information on the SPI clock frequency during runtime Notice that the SPI frequency should be a multiple of sample frequency to ensure the precision.

if DFSDM input is SPI master:

Reading SPI clkout frequency, error on writing

If DFSDM input is SPI Slave:

Reading returns value previously set. Writing value before starting conversions.

/sys/bus/iio/devices/iio:deviceX/in_voltagex_sys_calibration

Defined on file *sysfs-bus-iio-adc-ad7192*

Initiates the system calibration procedure. This is done on a single channel at a time. Write '1' to start the calibration.

/sys/bus/iio/devices/iio:deviceX/in_voltagex_sys_calibration_mode

Defined on file *sysfs-bus-iio-adc-ad7192*

Sets up the calibration mode used in the system calibration procedure. Reading returns the current calibration mode. Writing sets the system calibration mode.

/sys/bus/iio/devices/iio:deviceX/in_voltagex_sys_calibration_mode_available

Defined on file *sysfs-bus-iio-adc-ad7192*

Reading returns a list with the possible calibration modes. There are two available options: ”zero_scale” - calibrate to zero scale ”full_scale” - calibrate to full scale

/sys/bus/iio/devices/iio:deviceX/label

Defined on file *sysfs-bus-iio*

Optional symbolic label for a device. This is useful for userspace to be able to better identify an individual device.

The contents of the label are free-form, but there are some standardized uses:

For proximity sensors which give the proximity (of a person) to a certain wlan or wwan antenna the following standardized labels are used:

- ”proximity-wifi”

- ”proximity-lte”
- ”proximity-wifi-lte”
- ”proximity-wifi-left”
- ”proximity-wifi-right”

These are used to indicate to userspace that these proximity sensors may be used to tune transmit power to ensure that Specific Absorption Rate (SAR) limits are honored. The ”-left” and ”-right” labels are for devices with multiple antennas.

In some laptops/tablets the standardized proximity sensor labels instead indicate proximity to a specific part of the device:

- ”proximity-palmrest” indicates proximity to the keyboard's palmrest
- ”proximity-palmrest-left” indicates proximity to the left part of the palmrest
- ”proximity-palmrest-right” indicates proximity to the right part of the palmrest
- ”proximity-lap” indicates the device is being used on someone's lap

Note ”proximity-lap” is special in that its value may be calculated by firmware from other sensor readings, rather then being a raw sensor reading.

For accelerometers used in 2-in-1s with 360° (yoga-style) hinges, which have an accelerometer in both their base and their display, the following standardized labels are used:

- ”accel-base”
- ”accel-display”

For devices where an accelerometer is housed in the swivel camera subassembly (for AR application), the following standardized label is used:

- ”accel-camera”

/sys/bus/iio/devices/iio:deviceX/location

Defined on file *sysfs-bus-iio-cros-ec*

This attribute returns a string with the physical location where the motion sensor is placed. For example, in a laptop a motion sensor can be located on the base or on the lid. Current valid values are 'base' and 'lid'.

/sys/bus/iio/devices/iio:deviceX/meas_conf
/sys/bus/iio/devices/iio:deviceX/meas_conf_available

Defined on file *sysfs-bus-iio-magnetometer-hmc5843*

Current configuration and available configurations for the bias current.

normal	Normal measurement configurations (default)
positivebias	Positive bias configuration
negativebias	Negative bias configuration
disabled	Only available on HMC5983. Disables magnetic sensor and enables temperature sensor.

Note:
The effect of this configuration may vary according to the device. For exact documentation check the device's datasheet.

/sys/bus/iio/devices/iio:deviceX/mount_matrix
/sys/bus/iio/devices/iio:deviceX/in_mount_matrix
/sys/bus/iio/devices/iio:deviceX/out_mount_matrix
/sys/bus/iio/devices/iio:deviceX/in_anglvel_mount_matrix
/sys/bus/iio/devices/iio:deviceX/in_accel_mount_matrix

Defined on file *sysfs-bus-iio*

Mounting matrix for IIO sensors. This is a rotation matrix which informs userspace about sensor chip's placement relative to the main hardware it is mounted on.

Main hardware placement is defined according to the local reference frame related to the physical quantity the sensor measures.

Given that the rotation matrix is defined in a board specific way (platform data and / or device-tree), the main hardware reference frame definition is left to the implementor's choice (see below for a magnetometer example).

Applications should apply this rotation matrix to samples so that when main hardware reference frame is aligned onto local reference frame, then sensor chip reference frame is also perfectly aligned with it.

Matrix is a 3x3 unitary matrix and typically looks like [0, 1, 0; 1, 0, 0; 0, 0, -1]. Identity matrix [1, 0, 0; 0, 1, 0; 0, 0, 1] means sensor chip and main hardware are perfectly aligned with each other.

For example, a mounting matrix for a magnetometer sensor informs userspace about sensor chip's ORIENTATION relative to the main hardware.

More specifically, main hardware orientation is defined with respect to the LOCAL EARTH GEOMAGNETIC REFERENCE FRAME where :

- Y is in the ground plane and positive towards magnetic North ;
- X is in the ground plane, perpendicular to the North axis and positive towards the East ;
- Z is perpendicular to the ground plane and positive upwards.

An implementor might consider that for a hand-held device, a 'natural' orientation would be 'front facing camera at the top'. The main hardware reference frame could then be described as :

- Y is in the plane of the screen and is positive towards the top of the screen ;
- X is in the plane of the screen, perpendicular to Y axis, and positive towards the right hand side of the screen ;
- Z is perpendicular to the screen plane and positive out of the screen.

Another example for a quadrotor UAV might be :

- Y is in the plane of the propellers and positive towards the front-view camera;
- X is in the plane of the propellers, perpendicular to Y axis, and positive towards the starboard side of the UAV ;

- Z is perpendicular to propellers plane and positive upwards.

/sys/bus/iio/devices/iio:deviceX/name

Defined on file *sysfs-bus-iio*

Description of the physical chip / device for device X. Typically a part number.

/sys/bus/iio/devices/iio:deviceX/out_altvoltageY_frequency

Defined on file *sysfs-bus-iio*

Output frequency for channel Y in Hz. The number must always be specified and unique if the output corresponds to a single channel. Some drivers have additional constraints: ADF4371 has an integrated VCO with fundamendal output frequency ranging from 4000000000 Hz 8000000000 Hz.

out_altvoltage0_frequency:
A divide by 1, 2, 4, 8, 16, 32 or circuit generates frequencies from 62500000 Hz to 8000000000 Hz.

out_altvoltage1_frequency:
This channel duplicates the channel 0 frequency

out_altvoltage2_frequency:
A frequency doubler generates frequencies from 8000000000 Hz to 16000000000 Hz.

out_altvoltage3_frequency:
A frequency quadrupler generates frequencies from 16000000000 Hz to 32000000000 Hz.

Note: writes to one of the channels will affect the frequency of all the other channels, since it involves changing the VCO fundamental output frequency.

/sys/bus/iio/devices/iio:deviceX/out_altvoltageY_frequency_increment

Defined on file *sysfs-bus-iio-impedance-analyzer-ad5933*

Frequency increment in Hz (step size) between consecutive frequency points along the sweep.

/sys/bus/iio/devices/iio:deviceX/out_altvoltageY_frequency_points

Defined on file *sysfs-bus-iio-impedance-analyzer-ad5933*

Number of frequency points (steps) in the frequency sweep. This value, in conjunction with the out_altvoltageY_frequency_start and the out_altvoltageY_frequency_increment, determines the frequency sweep range for the sweep operation.

/sys/bus/iio/devices/iio:deviceX/out_altvoltageY_frequency_resolution

Defined on file *sysfs-bus-iio-frequency-adf4350*

Stores channel Y frequency resolution/channel spacing in Hz. The value given directly influences the MODULUS used by the fractional-N PLL. It is assumed that the algorithm that is

used to compute the various dividers, is able to generate proper values for multiples of channel spacing.

/sys/bus/iio/devices/iio:deviceX/out_altvoltageY_frequency_start

Defined on file *sysfs-bus-iio-impedance-analyzer-ad5933*

Frequency sweep start frequency in Hz.

/sys/bus/iio/devices/iio:deviceX/out_altvoltageY_name

Defined on file *sysfs-bus-iio-frequency-adf4371*

Reading returns the datasheet name for channel Y:

```
out_altvoltage0_name: RF8x
out_altvoltage1_name: RFAUX8x
out_altvoltage2_name: RF16x
out_altvoltage3_name: RF32x
```

/sys/bus/iio/devices/iio:deviceX/out_altvoltageY_phase

Defined on file *sysfs-bus-iio*

Phase in radians of one frequency/clock output Y (out_altvoltageY) relative to another frequency/clock output (out_altvoltageZ) of the device X. The number must always be specified and unique if the output corresponds to a single channel.

/sys/bus/iio/devices/iio:deviceX/out_altvoltageY_refin_frequency

Defined on file *sysfs-bus-iio-frequency-adf4350*

Sets channel Y REFin frequency in Hz. In some clock chained applications, the reference frequency used by the PLL may change during runtime. This attribute allows the user to adjust the reference frequency accordingly. The value written has no effect until out_altvoltageY_frequency is updated. Consider to use out_altvoltageY_powerdown to power down the PLL and its RFOut buffers during REFin changes.

/sys/bus/iio/devices/iio:deviceX/out_altvoltageY_settling_cycles

Defined on file *sysfs-bus-iio-impedance-analyzer-ad5933*

Number of output excitation cycles (settling time cycles) that are allowed to pass through the unknown impedance, after each frequency increment, and before the ADC is triggered to perform a conversion sequence of the response signal.

/sys/bus/iio/devices/iio:deviceX/out_conversion_mode

Defined on file *sysfs-bus-iio-vf610*

Specifies the hardware conversion mode used within DAC. The two available modes are ”high-power” and ”low-power”, where ”low-power” mode is the default mode.

/sys/bus/iio/devices/iio:deviceX/out_currentY_currentZ_raw

Defined on file *sysfs-bus-iio-light-lm3533-als*

Set the output current for channel out_currentY when in zone Z (0..255), where Y in 0..2 and Z in 0..4.

These values correspond to the ALS-mapper target registers for ALS-mapper Y + 1.

Note that out_currentY_raw provides the current for the current zone.

/sys/bus/iio/devices/iio:deviceX/out_currentY_raw

Defined on file *sysfs-bus-iio*

Set/get output current for channel Y. Units after application of scale and offset are milliamps. For some devices current channels are used to specify current supplied to elements used in taking a measurement of a different type. E.g. LED currents.

/sys/bus/iio/devices/iio:deviceX/out_current_heater_raw
/sys/bus/iio/devices/iio:deviceX/out_current_heater_raw_available

Defined on file *sysfs-bus-iio-humidity*

Controls the heater device within the humidity sensor to get rid of excess condensation.

In some devices, this is just a switch in which case 0 = OFF, and 1 = ON.

/sys/bus/iio/devices/iio:deviceX/out_humidityrelative_raw

Defined on file *sysfs-bus-iio-chemical-sgp40*

Set the relative humidity. This value is sent to the sensor for humidity compensation. Default value: 50000 (50 % relative humidity)

/sys/bus/iio/devices/iio:deviceX/out_resistance_raw_available

Defined on file *sysfs-bus-iio-potentiometer-mcp4531*

The range of available values represented as the minimum value, the step and the maximum value, all enclosed in square brackets. Example: [0 1 256]

/sys/bus/iio/devices/iio:deviceX/out_temp_raw

Defined on file *sysfs-bus-iio-chemical-sgp40*

Set the temperature. This value is sent to the sensor for temperature compensation. Default value: 25000 (25 °C)

/sys/bus/iio/devices/iio:deviceX/out_voltageY&Z_raw
/sys/bus/iio/devices/iio:deviceX/out_altvoltageY&Z_raw

Defined on file *sysfs-bus-iio*

Raw (unscaled, no bias etc.) output voltage for an aggregate of channel Y, channel Z, etc. This interface is available in cases where a single output sets the value for multiple channels simultaneously.

/sys/bus/iio/devices/iio:deviceX/out_voltageY_dither_en

Defined on file *sysfs-bus-iio-dac-ltc2688*

Dither enable. Write 1 to enable dither or 0 to disable it. This is useful for changing the dither parameters. They way it should be done is:

- disable dither operation;
- change dither parameters (eg: frequency, phase...);
- enabled dither operation

/sys/bus/iio/devices/iio:deviceX/out_voltageY_dither_frequency

Defined on file *sysfs-bus-iio-dac-ltc2688*

Sets the dither signal frequency. Units are in Hz.

/sys/bus/iio/devices/iio:deviceX/out_voltageY_dither_frequency_available

Defined on file *sysfs-bus-iio-dac-ltc2688*

Returns the available values for the dither frequency.

/sys/bus/iio/devices/iio:deviceX/out_voltageY_dither_offset

Defined on file *sysfs-bus-iio-dac-ltc2688*

Offset applied to out_voltageY_dither_raw. Read only attribute always set to 0.

/sys/bus/iio/devices/iio:deviceX/out_voltageY_dither_phase

Defined on file *sysfs-bus-iio-dac-ltc2688*

Sets the dither signal phase. Units are in Radians.

/sys/bus/iio/devices/iio:deviceX/out_voltageY_dither_phase_available

Defined on file *sysfs-bus-iio-dac-ltc2688*

Returns the available values for the dither phase.

/sys/bus/iio/devices/iio:deviceX/out_voltageY_dither_raw

Defined on file *sysfs-bus-iio-dac-ltc2688*

This raw, unscaled value refers to the dither signal amplitude. The same scale as in out_voltageY_raw applies. However, the offset might be different as it's always 0 for this attribute.

/sys/bus/iio/devices/iio:deviceX/out_voltageY_dither_raw_available

Defined on file *sysfs-bus-iio-dac-ltc2688*

Available range for dither raw amplitude values.

/sys/bus/iio/devices/iio:deviceX/out_voltageY_hardwaregain
/sys/bus/iio/devices/iio:deviceX/in_intensity_hardwaregain
/sys/bus/iio/devices/iio:deviceX/in_intensity_red_hardwaregain
/sys/bus/iio/devices/iio:deviceX/in_intensity_green_hardwaregain
/sys/bus/iio/devices/iio:deviceX/in_intensity_blue_hardwaregain
/sys/bus/iio/devices/iio:deviceX/in_intensity_clear_hardwaregain

Defined on file *sysfs-bus-iio*

Hardware applied gain factor. If shared across all channels, <type>_hardwaregain is used.

/sys/bus/iio/devices/iio:deviceX/out_voltageY_powerdown_mode
/sys/bus/iio/devices/iio:deviceX/out_voltage_powerdown_mode
/sys/bus/iio/devices/iio:deviceX/out_altvoltageY_powerdown_mode
/sys/bus/iio/devices/iio:deviceX/out_altvoltage_powerdown_mode

Defined on file *sysfs-bus-iio*

Specifies the output powerdown mode. DAC output stage is disconnected from the amplifier and 1kohm_to_gnd: connected to ground via an 1kOhm resistor, 2.5kohm_to_gnd: connected to ground via a 2.5kOhm resistor, 6kohm_to_gnd: connected to ground via a 6kOhm resistor, 20kohm_to_gnd: connected to ground via a 20kOhm resistor, 90kohm_to_gnd: connected to ground via a 90kOhm resistor, 100kohm_to_gnd: connected to ground via an 100kOhm resistor, 125kohm_to_gnd: connected to ground via an 125kOhm resistor, 500kohm_to_gnd: connected to ground via a 500kOhm resistor, 640kohm_to_gnd: connected to ground via a 640kOhm resistor, three_state: left floating. For a list of available output power down options read outX_powerdown_mode_available. If Y is not present the mode is shared across all outputs.

/sys/bus/iio/devices/iio:deviceX/out_voltageY_powerdown
/sys/bus/iio/devices/iio:deviceX/out_voltage_powerdown
/sys/bus/iio/devices/iio:deviceX/out_altvoltageY_powerdown
/sys/bus/iio/devices/iio:deviceX/out_altvoltage_powerdown

Defined on file *sysfs-bus-iio*

Writing 1 causes output Y to enter the power down mode specified by the corresponding outY_powerdown_mode. DAC output stage is disconnected from the amplifier. Clearing returns to normal operation. Y may be suppressed if all outputs are controlled together.

/sys/bus/iio/devices/iio:deviceX/out_voltageY_raw0
/sys/bus/iio/devices/iio:deviceX/out_voltageY_raw1

Defined on file *sysfs-bus-iio-dac-ltc2688*

It has the same meaning as out_voltageY_raw. This attribute is specific to toggle enabled channels and refers to the DAC output code in INPUT_A (_raw0) and INPUT_B (_raw1). The same scale and offset as in out_voltageY_raw applies.

/sys/bus/iio/devices/iio:deviceX/out_voltageY_raw_available

Defined on file *sysfs-bus-iio-dac-dpot-dac*

The range of available values represented as the minimum value, the step and the maximum value, all enclosed in square brackets. Example: [0 1 256]

/sys/bus/iio/devices/iio:deviceX/out_voltageY_raw
/sys/bus/iio/devices/iio:deviceX/out_altvoltageY_raw

Defined on file *sysfs-bus-iio*

Raw (unscaled, no bias etc.) output voltage for channel Y. The number must always be specified and unique if the output corresponds to a single channel. While DAC like devices typically use out_voltage, a continuous frequency generating device, such as a DDS or PLL should use out_altvoltage.

/sys/bus/iio/devices/iio:deviceX/out_voltageY_symbol

Defined on file *sysfs-bus-iio-dac-ltc2688*

Performs a SW toggle. This attribute is specific to toggle enabled channels and allows to toggle between out_voltageY_raw0 and out_voltageY_raw1 through software. Writing 0 will select out_voltageY_raw0 while 1 selects out_voltageY_raw1.

/sys/bus/iio/devices/iio:deviceX/out_voltageY_toggle_en

Defined on file *sysfs-bus-iio-dac-ltc2688*

Toggle enable. Write 1 to enable toggle or 0 to disable it. This is useful when one wants to change the DAC output codes. The way it should be done is:

- disable toggle operation;
- change out_voltageY_raw0 and out_voltageY_raw1;
- enable toggle operation.

/sys/bus/iio/devices/iio:deviceX/oversampling_ratio

Defined on file *sysfs-bus-iio*

Hardware dependent ADC oversampling. Controls the sampling ratio of the digital filter if available.

/sys/bus/iio/devices/iio:deviceX/oversampling_ratio_available

Defined on file *sysfs-bus-iio*

Hardware dependent values supported by the oversampling filter.

/sys/bus/iio/devices/iio:deviceX/pll2_feedback_clk_present
/sys/bus/iio/devices/iio:deviceX/pll2_reference_clk_present
/sys/bus/iio/devices/iio:deviceX/pll1_reference_clk_a_present
/sys/bus/iio/devices/iio:deviceX/pll1_reference_clk_b_present
/sys/bus/iio/devices/iio:deviceX/pll1_reference_clk_test_present
/sys/bus/iio/devices/iio:deviceX/vcxo_clk_present

Defined on file *sysfs-bus-iio-frequency-ad9523*

Reading returns either '1' or '0'.

'1' means that the clock in question is present.

'0' means that the clock is missing.

/sys/bus/iio/devices/iio:deviceX/pllY_locked

Defined on file *sysfs-bus-iio-frequency-ad9523*

Reading returns either '1' or '0'. '1' means that the pllY is locked.

/sys/bus/iio/devices/iio:deviceX/proximity_on_chip_ambient_infrared_suppression

Defined on file *sysfs-bus-iio-light-isl29018*

From ISL29018 Data Sheet (FN6619.4, Oct 8, 2012) regarding the infrared suppression:

Scheme 0, makes full n (4, 8, 12, 16) bits (unsigned) proximity detection. The range of Scheme 0 proximity count is from 0 to 2^n. Logic 1 of this bit, Scheme 1, makes n-1 (3, 7, 11, 15) bits (2's complementary) proximity_less_ambient detection. The range of Scheme 1 proximity count is from -2^(n-1) to 2^(n-1). The sign bit is extended for resolutions less than 16. While Scheme 0 has wider dynamic range, Scheme 1 proximity detection is less affected by the ambient IR noise variation.

0	Sensing IR from LED and ambient
1	Sensing IR from LED with ambient IR rejection

/sys/bus/iio/devices/iio:deviceX/sampling_frequency_available
/sys/bus/iio/devices/iio:deviceX/in_intensity_sampling_frequency_available
/sys/bus/iio/devices/iio:deviceX/in_proximity_sampling_frequency_available
/sys/.../iio:deviceX/buffer/sampling_frequency_available
/sys/bus/iio/devices/triggerX/sampling_frequency_available

Defined on file *sysfs-bus-iio*

When the internal sampling clock can only take a specific set of frequencies, we can specify the available values with:

- a small discrete set of values like "0 2 4 6 8"

- a range with minimum, step and maximum frequencies like "[min step max]"

/sys/bus/iio/devices/iio:deviceX/sampling_frequency
/sys/bus/iio/devices/iio:deviceX/in_intensity_sampling_frequency
/sys/bus/iio/devices/iio:deviceX/buffer/sampling_frequency
/sys/bus/iio/devices/triggerX/sampling_frequency

Defined on file *sysfs-bus-iio*

Some devices have internal clocks. This parameter sets the resulting sampling frequency. In many devices this parameter has an effect on input filters etc. rather than simply controlling when the input is sampled. As this affects data ready triggers, hardware buffers and the sysfs direct access interfaces, it may be found in any of the relevant directories. If it affects all of the above then it is to be found in the base device directory.

The stm32-timer-trigger has the additional characteristic that a sampling_frequency of 0 is defined to stop sampling.

/sys/bus/iio/devices/iio:deviceX/sensor_max_range

Defined on file *sysfs-bus-iio-distance-srf08*

Show or set the maximum range between the sensor and the first object echoed in meters. Default value is 6.020. This setting limits the time the driver is waiting for a echo.

Showing the range of available values is represented as the minimum value, the step and the maximum value, all enclosed in square brackets.

Example:

```
[0.043 0.043 11.008]
```

/sys/bus/iio/devices/iio:deviceX/sensor_sensitivity

Defined on file *sysfs-bus-iio-proximity*

Proximity sensors sometimes have a controllable amplifier on the signal from which time of flight measurements are taken. The appropriate values to take is dependent on both the sensor and its operating environment: * as3935 (0-31 range) 18 = indoors (default) 14 = outdoors

/sys/bus/iio/devices/iio:deviceX/serialnumber

Defined on file *sysfs-bus-iio*

An example format is 16-bytes, 2-digits-per-byte, HEX-string representing the sensor unique ID number.

/sys/bus/iio/devices/iio:deviceX/start_cleaning

Defined on file *sysfs-bus-iio-sps30*

Writing 1 starts sensor self cleaning. Internal fan accelerates to its maximum speed and keeps spinning for about 10 seconds in order to blow out accumulated dust.

/sys/bus/iio/devices/iio:deviceX/store_eeprom

Defined on file *sysfs-bus-iio*

Writing '1' stores the current device configuration into on-chip EEPROM. After power-up or chip reset the device will automatically load the saved configuration.

/sys/bus/iio/devices/iio:deviceX/sync_dividers

Defined on file *sysfs-bus-iio-frequency-ad9523*

Writing '1' triggers the clock distribution synchronization functionality. All dividers are reset and the channels start with their predefined phase offsets (out_altvoltageY_phase). Writing this file has the effect as driving the external /SYNC pin low.

/sys/bus/iio/devices/iio:deviceX/sys_calibration_auto_status

Defined on file *sysfs-bus-iio-bno055*

Reports the status for the IMU overall autocalibration. Can be 0 (calibration non even enabled) or 1 to 5 where the greater the number, the better the calibration status.

/sys/bus/iio/devices/iio:deviceX/trigger/current_trigger

Defined on file *sysfs-bus-iio*

The name of the trigger source being used, as per string given in /sys/class/iio/triggerY/name.

/sys/bus/iio/devices/iio_sysfs_trigger/add_trigger

Defined on file *sysfs-bus-iio-trigger-sysfs*

This attribute is provided by the iio-trig-sysfs stand-alone driver and it is used to activate the creation of a new trigger. In order to achieve this, one should write a positive integer into the associated file, which will serve as the id of the trigger. If the trigger with the specified id is already present in the system, an invalid argument message will be returned.

/sys/bus/iio/devices/iio_sysfs_trigger/remove_trigger

Defined on file *sysfs-bus-iio-trigger-sysfs*

This attribute is used to unregister and delete a previously created trigger from the list of available triggers. In order to achieve this, one should write a positive integer into the associated file, representing the id of the trigger that needs to be removed. If the trigger can't be found, an invalid argument message will be returned to the user.

/sys/bus/iio/devices/triggerX

Defined on file *sysfs-bus-iio*

An event driven driver of data capture to an in kernel buffer. May be provided by a device driver that also has an IIO device based on hardware generated events (e.g. data ready) or provided by a separate driver for other hardware (e.g. periodic timer, GPIO or high resolution timer).

Contains trigger type specific elements. These do not generalize well and hence are not documented in this file. X is the IIO index of the trigger.

/sys/bus/iio/devices/triggerX/master_mode

Defined on file *sysfs-bus-iio-timer-stm32*

Reading returns the current master modes. Writing set the master mode

/sys/bus/iio/devices/triggerX/master_mode_available

Defined on file *sysfs-bus-iio-timer-stm32*

Reading returns the list possible master modes which are:

- **"reset"**
  The UG bit from the TIMx_EGR register is used as trigger output (TRGO).
- **"enable"**
  The Counter Enable signal CNT_EN is used as trigger output.
- **"update"**
  The update event is selected as trigger output. For instance a master timer can then be used as a prescaler for a slave timer.
- **"compare_pulse"**
  The trigger output send a positive pulse when the CC1IF flag is to be set.
- **"OC1REF"**
  OC1REF signal is used as trigger output.
- **"OC2REF"**
  OC2REF signal is used as trigger output.
- **"OC3REF"**
  OC3REF signal is used as trigger output.
- **"OC4REF"**
  OC4REF signal is used as trigger output.

Additional modes (on TRGO2 only):

- **"OC5REF"**
  OC5REF signal is used as trigger output.
- **"OC6REF"**
  OC6REF signal is used as trigger output.
- **"compare_pulse_OC4REF":**
  OC4REF rising or falling edges generate pulses.
- **"compare_pulse_OC6REF":**
  OC6REF rising or falling edges generate pulses.

- **"compare_pulse_OC4REF_r_or_OC6REF_r":**
  OC4REF or OC6REF rising edges generate pulses.
- **"compare_pulse_OC4REF_r_or_OC6REF_f":**
  OC4REF rising or OC6REF falling edges generate pulses.
- **"compare_pulse_OC5REF_r_or_OC6REF_r":**
  OC5REF or OC6REF rising edges generate pulses.
- **"compare_pulse_OC5REF_r_or_OC6REF_f":**
  OC5REF rising or OC6REF falling edges generate pulses.

```
+-----------+   +-------------+            +---------+
| Prescaler +-> | Counter     |        +-> | Master  | TRGO(2)
+-----------+   +--+--------+-+        |-> | Control +-->
                   |        |          ||  +---------+
                +--v--------+-+ OCxREF ||  +---------+
                | Chx compare +----------> | Output  | ChX
                +-----------+-+         |  | Control +-->
                      .     |           |  +---------+
                      .     |           |    .
                +-----------v-+ OC6REF  |    .
                | Ch6 compare +---------+>
                +-------------+
```

Example with: "compare_pulse_OC4REF_r_or_OC6REF_r":

```
                X
              X   X
            X .   . X
          X   .   .   X
        X     .   .     X
count X .     .   .     . X
        .     .   .     .
        .     .   .     .
        +---------------+
OC4REF  |     .   .     |
      +-+     .   .     +-+
        .     +---+     .
OC6REF  .     |   |     .
      +-------+   +-------+
        +-+   +-+
TRGO2   | |   | |
      +-+ +---+ +---------+
```

/sys/bus/iio/devices/triggerX/name

Defined on file *sysfs-bus-iio-trigger-sysfs*

The name attribute holds a description string for the current trigger. In order to associate the trigger with an IIO device one should write this name string to /sys/bus/iio/devices/iio:deviceY/trigger/current_trigger.

/sys/bus/iio/devices/triggerX/name = "adxl372-devX-peak"

Defined on file *sysfs-bus-iio-accel-adxl372*

The adxl372 accelerometer kernel module provides an additional trigger, which sets the device in a mode in which it will record only the peak acceleration sensed over the set period of time in the events sysfs.

/sys/bus/iio/devices/triggerX/name = "bmc150_accel-any-motion-devX"

Defined on file *sysfs-bus-iio-accel-bmc150*

The BMC150 accelerometer kernel module provides an additional trigger, which sets driver in a mode, where data is pushed to the buffer only when there is any motion.

/sys/bus/iio/devices/triggerX/name = "bmg160-any-motion-devX"

Defined on file *sysfs-bus-iio-gyro-bmg160*

The BMG160 gyro kernel module provides an additional trigger, which sets driver in a mode, where data is pushed to the buffer only when there is any motion.

/sys/bus/iio/devices/triggerX/trigger_now

Defined on file *sysfs-bus-iio-trigger-sysfs*

This file is provided by the iio-trig-sysfs stand-alone trigger driver. Writing this file with any value triggers an event driven driver, associated with this trigger, to capture data into an in kernel buffer. This approach can be valuable during automated testing or in situations, where other trigger methods are not applicable. For example no RTC or spare GPIOs. X is the IIO index of the trigger.

/sys/bus/iio/devices/triggerX/trigger_polarity

Defined on file *sysfs-bus-iio-adc-stm32*

The STM32 ADC can be configured to use external trigger sources (e.g. timers, pwm or exti gpio). Then, it can be tuned to start conversions on external trigger by either:

- "rising-edge"
- "falling-edge"
- "both-edges".

Reading returns current trigger polarity.

Writing value before enabling conversions sets trigger polarity.

/sys/bus/iio/devices/triggerX/trigger_polarity_available

Defined on file *sysfs-bus-iio-adc-stm32*

List all available trigger_polarity settings.

/sys/bus/intel_th/devices/<intel_th_id>-<device><id>/active

Defined on file *sysfs-bus-intel_th-output-devices*

(RW) Writes of 1 or 0 enable or disable trace output to this output device. Reads return current status. Requires that the correstponding output port driver be loaded.

/sys/bus/intel_th/devices/<intel_th_id>-gth/masters/*

Defined on file *sysfs-bus-intel_th-devices-gth*

(RW) Configure output ports for STP masters. Writing -1 disables a master; any

/sys/bus/intel_th/devices/<intel_th_id>-gth/outputs/[0-7]_drop

Defined on file *sysfs-bus-intel_th-devices-gth*

(RW) Data retention policy setting: keep (0) or drop (1) incoming data while output port is in reset.

/sys/bus/intel_th/devices/<intel_th_id>-gth/outputs/[0-7]_flush

Defined on file *sysfs-bus-intel_th-devices-gth*

(RW) Force flush data from byte packing buffer for the output port.

/sys/bus/intel_th/devices/<intel_th_id>-gth/outputs/[0-7]_null

Defined on file *sysfs-bus-intel_th-devices-gth*

(RW) STP NULL packet generation: enabled (1) or disabled (0).

/sys/bus/intel_th/devices/<intel_th_id>-gth/outputs/[0-7]_port

Defined on file *sysfs-bus-intel_th-devices-gth*

(RO) Output port type:

0	not present,
1	MSU (Memory Storage Unit)
2	CTP (Common Trace Port)
4	PTI (MIPI PTI).

/sys/bus/intel_th/devices/<intel_th_id>-gth/outputs/[0-7]_reset

Defined on file *sysfs-bus-intel_th-devices-gth*

(RO) Output port is in reset (1).

/sys/bus/intel_th/devices/<intel_th_id>-gth/outputs/[0-7]_smcfreq

Defined on file *sysfs-bus-intel_th-devices-gth*

(RW) STP sync packet frequency for the port. Specifies the number of clocks between maintenance packets.

/sys/bus/intel_th/devices/<intel_th_id>-msc<msc-id>/mode

Defined on file *sysfs-bus-intel_th-devices-msc*

(RW) Configure MSC operating mode:

- "single", for contiguous buffer mode (high-order alloc);
- "multi", for multiblock mode;
- "ExI", for DCI handler mode;
- "debug", for debug mode;
- any of the currently loaded buffer sinks.

If operating mode changes, existing buffer is deallocated, provided there are no active users and tracing is not enabled, otherwise the write will fail.

/sys/bus/intel_th/devices/<intel_th_id>-msc<msc-id>/nr_pages

Defined on file *sysfs-bus-intel_th-devices-msc*

(RW) Configure MSC buffer size for "single" or "multi" modes.

In single mode, this is a single number of pages, has to be power of 2. In multiblock mode, this is a comma-separated list of numbers of pages for each window to be allocated. Number of windows is not limited.

Writing to this file deallocates existing buffer (provided there are no active users and tracing is not enabled) and then allocates a new one.

/sys/bus/intel_th/devices/<intel_th_id>-msc<msc-id>/port

Defined on file *sysfs-bus-intel_th-output-devices*

(RO) Port number, corresponding to this output device on the switch (GTH) or "unassigned" if the corresponding output port driver is not loaded.

/sys/bus/intel_th/devices/<intel_th_id>-msc<msc-id>/stop_on_full

Defined on file *sysfs-bus-intel_th-devices-msc*

(RW) Configure whether trace stops when the last available window becomes full (1/y/Y) or wraps around and continues until the next window becomes available again (0/n/N).

/sys/bus/intel_th/devices/<intel_th_id>-msc<msc-id>/win_switch

Defined on file *sysfs-bus-intel_th-devices-msc*

(RW) Trigger window switch for the MSC's buffer, in multi-window mode. In "multi" mode, accepts writes of "1", thereby triggering a window switch for the buffer. Returns an error in any other operating mode or attempts to write something other than "1".

/sys/bus/intel_th/devices/<intel_th_id>-msc<msc-id>/wrap

Defined on file *sysfs-bus-intel_th-devices-msc*

(RW) Configure MSC buffer wrapping. 1 == wrapping enabled.

/sys/bus/intel_th/devices/<intel_th_id>-pti/clock_divider

Defined on file *sysfs-bus-intel_th-devices-pti*

(RW) Configure PTI port clock divider:

- 0: Intel TH clock rate,
- 1: 1/2 Intel TH clock rate,
- 2: 1/4 Intel TH clock rate,
- 3: 1/8 Intel TH clock rate.

/sys/bus/intel_th/devices/<intel_th_id>-pti/freerunning_clock

Defined on file *sysfs-bus-intel_th-devices-pti*

(RW) 0: PTI trace clock acts as a strobe which only toggles when there is trace data to send. 1: PTI trace clock is a free-running clock.

/sys/bus/intel_th/devices/<intel_th_id>-pti/mode

Defined on file *sysfs-bus-intel_th-devices-pti*

(RW) Configure PTI output width. Currently supported values are 4, 8, 12, 16.

/sys/bus/mcb/devices/mcb:X

Defined on file *sysfs-bus-mcb*

Hardware chip or device hosting the MEN chameleon bus

/sys/bus/mcb/devices/mcb:X/minor

Defined on file *sysfs-bus-mcb*

The FPGA's minor number

/sys/bus/mcb/devices/mcb:X/model

Defined on file *sysfs-bus-mcb*

The FPGA's model number

/sys/bus/mcb/devices/mcb:X/name

Defined on file *sysfs-bus-mcb*

The FPGA's name

/sys/bus/mcb/devices/mcb:X/revision

Defined on file *sysfs-bus-mcb*

The FPGA's revision number

/sys/bus/mdio_bus/devices/.../statistics/errors_<addr>
/sys/class/mdio_bus/.../statistics/errors_<addr>

Defined on file *sysfs-bus-mdio*

Total number of transfer errors for this MDIO bus address.

/sys/bus/mdio_bus/devices/.../statistics/errors
/sys/class/mdio_bus/.../statistics/errors

Defined on file *sysfs-bus-mdio*

Total number of transfer errors for this MDIO bus.

/sys/bus/mdio_bus/devices/.../statistics/reads_<addr>
/sys/class/mdio_bus/.../statistics/reads_<addr>

Defined on file *sysfs-bus-mdio*

Total number of read transactions for this MDIO bus address.

/sys/bus/mdio_bus/devices/.../statistics/reads
/sys/class/mdio_bus/.../statistics/reads

Defined on file *sysfs-bus-mdio*

Total number of read transactions for this MDIO bus.

/sys/bus/mdio_bus/devices/.../statistics/transfers_<addr>
/sys/class/mdio_bus/.../statistics/transfers_<addr>

Defined on file *sysfs-bus-mdio*

Total number of transfers for this MDIO bus address.

/sys/bus/mdio_bus/devices/.../statistics/transfers
/sys/class/mdio_bus/.../transfers

Defined on file *sysfs-bus-mdio*

Total number of transfers for this MDIO bus.

/sys/bus/mdio_bus/devices/.../statistics/writes_<addr>
/sys/class/mdio_bus/.../statistics/writes_<addr>

Defined on file *sysfs-bus-mdio*

Total number of write transactions for this MDIO bus address.

/sys/bus/mdio_bus/devices/.../statistics/writes
/sys/class/mdio_bus/.../statistics/writes

Defined on file *sysfs-bus-mdio*

Total number of write transactions for this MDIO bus.

/sys/bus/mdio_bus/devices/.../statistics/
/sys/class/mdio_bus/.../statistics/

Defined on file *sysfs-bus-mdio*

This folder contains statistics about global and per MDIO bus address statistics.

/sys/bus/media/devices/.../model

Defined on file *sysfs-bus-media*

Contains the device model name in UTF-8. The device version is is not be appended to the model name.

/sys/bus/mei/devices/.../fixed

Defined on file *sysfs-bus-mei*

Stores mei client fixed address, if any Format: %d

/sys/bus/mei/devices/.../max_conn

Defined on file *sysfs-bus-mei*

Stores mei client maximum number of connections Format: %d

/sys/bus/mei/devices/.../max_len

Defined on file *sysfs-bus-mei*

Stores mei client maximum message length Format: %d

/sys/bus/mei/devices/.../modalias

Defined on file *sysfs-bus-mei*

Stores the same MODALIAS value emitted by uevent Format: mei:<mei device name>:<device uuid>:<protocol version>

/sys/bus/mei/devices/.../name

Defined on file *sysfs-bus-mei*

Stores mei client device name Format: string

/sys/bus/mei/devices/.../uuid

Defined on file *sysfs-bus-mei*

Stores mei client device uuid Format: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx

/sys/bus/mei/devices/.../version

Defined on file *sysfs-bus-mei*

Stores mei client protocol version Format: %d

/sys/bus/mei/devices/.../vtag

Defined on file *sysfs-bus-mei*

Stores mei client vtag support status Format: %d

/sys/bus/mmc/devices/.../rev

Defined on file *sysfs-bus-mmc*

Extended CSD revision number

/sys/bus/most/devices/<dev>/<channel>/

Defined on file *sysfs-bus-most*

For every channel of the device a directory is created, whose name is dictated by the HDM. This enables an application to collect information about the channel's capabilities and configure it.

/sys/bus/most/devices/<dev>/<channel>/available_datatypes

Defined on file *sysfs-bus-most*

Indicates the data types the channel can transport.

/sys/bus/most/devices/<dev>/<channel>/available_directions

Defined on file *sysfs-bus-most*

Indicates the directions the channel is capable of.

/sys/bus/most/devices/<dev>/<channel>/channel_starving

Defined on file *sysfs-bus-most*

Indicates whether channel ran out of buffers.

/sys/bus/most/devices/<dev>/<channel>/number_of_packet_buffers

Defined on file *sysfs-bus-most*

Indicates the number of packet buffers the channel can handle.

/sys/bus/most/devices/<dev>/<channel>/number_of_stream_buffers

Defined on file *sysfs-bus-most*

Indicates the number of streaming buffers the channel can handle.

/sys/bus/most/devices/<dev>/<channel>/set_buffer_size

Defined on file *sysfs-bus-most*

This is to read back the configured buffer size of the channel.

/sys/bus/most/devices/<dev>/<channel>/set_datatype

Defined on file *sysfs-bus-most*

This is to read back the configured data type of the channel. The following strings will be accepted:

```
'control',
'async',
'sync',
'isoc_avp'
```

/sys/bus/most/devices/<dev>/<channel>/set_direction

Defined on file *sysfs-bus-most*

This is to read back the configured direction of the channel. The following strings will be accepted:

```
'tx',
'rx'
```

/sys/bus/most/devices/<dev>/<channel>/set_number_of_buffers

Defined on file *sysfs-bus-most*

This is to read back the configured number of buffers of the channel.

/sys/bus/most/devices/<dev>/<channel>/set_packets_per_xact

Defined on file *sysfs-bus-most*

This is to read back the configured number of packets per transaction of the channel. This is only applicable when connected via USB.

/sys/bus/most/devices/<dev>/<channel>/set_subbuffer_size

Defined on file *sysfs-bus-most*

This is to read back the configured subbuffer size of the channel.

/sys/bus/most/devices/<dev>/<channel>/size_of_packet_buffer

Defined on file *sysfs-bus-most*

Indicates the size of a packet buffer the channel can handle.

/sys/bus/most/devices/<dev>/<channel>/size_of_stream_buffer

Defined on file *sysfs-bus-most*

Indicates the size of a streaming buffer the channel can handle.

/sys/bus/most/devices/<dev>/dci

Defined on file *sysfs-bus-most*

If the network interface controller is attached via USB, a dci directory is created that allows applications to read and write the controller's DCI registers.

/sys/bus/most/devices/<dev>/dci/arb_address

Defined on file *sysfs-bus-most*

This is used to set an arbitrary DCI register address an application wants to read from or write to.

/sys/bus/most/devices/<dev>/dci/arb_value

Defined on file *sysfs-bus-most*

This is used to read and write the DCI register whose address is stored in arb_address.

/sys/bus/most/devices/<dev>/dci/mep_eui48_hi

Defined on file *sysfs-bus-most*

This is used to check and configure the MAC address.

/sys/bus/most/devices/<dev>/dci/mep_eui48_lo

Defined on file *sysfs-bus-most*

This is used to check and configure the MAC address.

/sys/bus/most/devices/<dev>/dci/mep_eui48_mi

Defined on file *sysfs-bus-most*

This is used to check and configure the MAC address.

/sys/bus/most/devices/<dev>/dci/mep_filter

Defined on file *sysfs-bus-most*

This is used to check and configure the MEP filter address.

/sys/bus/most/devices/<dev>/dci/mep_hash0

Defined on file *sysfs-bus-most*

This is used to check and configure the MEP hash table.

/sys/bus/most/devices/<dev>/dci/mep_hash1

Defined on file *sysfs-bus-most*

This is used to check and configure the MEP hash table.

/sys/bus/most/devices/<dev>/dci/mep_hash2

Defined on file *sysfs-bus-most*

This is used to check and configure the MEP hash table.

/sys/bus/most/devices/<dev>/dci/mep_hash3

Defined on file *sysfs-bus-most*

This is used to check and configure the MEP hash table.

/sys/bus/most/devices/<dev>/dci/ni_state

Defined on file *sysfs-bus-most*

Indicates the current network interface state.

/sys/bus/most/devices/<dev>/dci/node_address

Defined on file *sysfs-bus-most*

Indicates the current node address.

/sys/bus/most/devices/<dev>/dci/node_position

Defined on file *sysfs-bus-most*

Indicates the current node position.

/sys/bus/most/devices/<dev>/dci/packet_bandwidth

Defined on file *sysfs-bus-most*

Indicates the configured packet bandwidth.

/sys/bus/most/devices/<dev>/dci/sync_ep

Defined on file *sysfs-bus-most*

Triggers the controller's synchronization process for a certain endpoint.

/sys/bus/most/devices/<dev>/description

Defined on file *sysfs-bus-most*

Provides information about the physical location of the device. Hardware attached via USB, for instance, might return <1-1.1:1.0>

/sys/bus/most/devices/<dev>/interface

Defined on file *sysfs-bus-most*

Indicates the type of peripheral interface the device uses.

/sys/bus/most/drivers/most_core/components

Defined on file *sysfs-bus-most*

This is used to retrieve a list of registered components.

/sys/bus/most/drivers/most_core/links

Defined on file *sysfs-bus-most*

This is used to retrieve a list of established links.

/sys/bus/moxtet/devices/moxtet-<name>.<addr>/module_description

Defined on file *sysfs-bus-moxtet-devices*

(Read) Moxtet module description. Format: string

/sys/bus/moxtet/devices/moxtet-<name>.<addr>/module_id

Defined on file *sysfs-bus-moxtet-devices*

(Read) Moxtet module ID. Format: %x

/sys/bus/moxtet/devices/moxtet-<name>.<addr>/module_name

Defined on file *sysfs-bus-moxtet-devices*

(Read) Moxtet module name. Format: string

/sys/bus/nd/devices/ndbusX/nfit/dsm_mask

Defined on file *sysfs-bus-nfit*

(RO) The bitmask indicates the supported bus specific control functions. See the section named 'NVDIMM Root Device _DSMs' in the ACPI specification.

/sys/bus/nd/devices/ndbusX/nfit/firmware_activate_noidle

Defined on file *sysfs-bus-nfit*

(RW) The Intel platform implementation of firmware activate support exposes an option let the platform force idle devices in the system over the activation event, or trust that the OS will do it. The safe default is to let the platform force idle devices since the kernel is already in a suspend state, and on the chance that a driver does not properly quiesce bus-mastering after a suspend callback the platform will handle it. However, the activation might abort if, for example, platform firmware determines that the activation time exceeds the max PCI-E completion timeout. Since the platform does not know whether the OS is running the activation from a suspend context it aborts, but if the system owner trusts driver suspend callback to be sufficient then 'firmware_activation_noidle' can be enabled to bypass the activation abort.

/sys/bus/nd/devices/ndbusX/nfit/hw_error_scrub

Defined on file *sysfs-bus-nfit*

(RW) Provides a way to toggle the behavior between just adding the address (cache line) where the MCE happened to the poison list and doing a full scrub. The former (selective insertion of the address) is done unconditionally.

This attribute can have the following values written to it:

'0': Switch to the default mode where an exception will only insert the address of the memory error into the poison and badblocks lists. '1': Enable a full scrub to happen if an exception for a memory error is received.

/sys/bus/nd/devices/ndbusX/nfit/revision

Defined on file *sysfs-bus-nfit*

(RO) ACPI NFIT table revision number.

/sys/bus/nd/devices/ndbusX/nfit/scrub

Defined on file *sysfs-bus-nfit*

(RW) This shows the number of full Address Range Scrubs (ARS) that have been completed since driver load time. Userspace can wait on this using select/poll etc. A '+' at the end indicates an ARS is in progress

Writing a value of 1 triggers an ARS scan.

/sys/bus/nd/devices/nmemX/cxl/id

Defined on file *sysfs-bus-nvdimm*

(RO) Show the id (serial) of the device. This is CXL specific.

/sys/bus/nd/devices/nmemX/cxl/provider

Defined on file *sysfs-bus-nvdimm*

(RO) Shows the CXL bridge device that ties to a CXL memory device to this NVDIMM device. I.e. the parent of the device returned is a /sys/bus/cxl/devices/memX instance.

/sys/bus/nd/devices/nmemX/nfit/device

Defined on file *sysfs-bus-nfit*

(RO) Device id for the NVDIMM, assigned by the module vendor.

/sys/bus/nd/devices/nmemX/nfit/dsm_mask

Defined on file *sysfs-bus-nfit*

(RO) The bitmask indicates the supported device specific control functions relative to the NVDIMM command family supported by the device

/sys/bus/nd/devices/nmemX/nfit/family

Defined on file *sysfs-bus-nfit*

(RO) Displays the NVDIMM family command sets. Values 0, 1, 2 and 3 correspond to NVDIMM_FAMILY_INTEL, NVDIMM_FAMILY_HPE1, NVDIMM_FAMILY_HPE2 and NVDIMM_FAMILY_MSFT respectively.

See the specifications for these command families here: http://pmem.io/documents/NVDIMM_DSM_Interface-V1.6.pdf https://github.com/HewlettPackard/hpe-nvm/blob/master/Documentation/ https://msdn.microsoft.com/library/windows/hardware/mt604741”

/sys/bus/nd/devices/nmemX/nfit/flags

Defined on file *sysfs-bus-nfit*

(RO) The flags in the NFIT memory device sub-structure indicate the state of the data on the nvdimm relative to its energy source or last ”flush to persistence”.

The attribute is a translation of the 'NVDIMM State Flags' field in section 5.2.25.3 'NVDIMM Region Mapping' Structure of the ACPI specification 6.2.

The health states are ”save_fail”, ”restore_fail”, ”flush_fail”, ”not_armed”, ”smart_event”, ”map_fail” and ”smart_notify”.

/sys/bus/nd/devices/nmemX/nfit/format
/sys/bus/nd/devices/nmemX/nfit/format1
/sys/bus/nd/devices/nmemX/nfit/formats

Defined on file *sysfs-bus-nfit*

(RO) The interface codes indicate support for persistent memory mapped directly into system physical address space and / or a block aperture access mechanism to the NVDIMM media. The 'formats' attribute displays the number of supported interfaces.

This layout is compatible with existing libndctl binaries that only expect one code per-dimm as they will ignore nmemX/nfit/formats and nmemX/nfit/formatN.

/sys/bus/nd/devices/nmemX/nfit/handle

Defined on file *sysfs-bus-nfit*

(RO) The address (given by the _ADR object) of the device on its parent bus of the NVDIMM device containing the NVDIMM region.

/sys/bus/nd/devices/nmemX/nfit/id

Defined on file *sysfs-bus-nfit*

(RO) ACPI specification 6.2 section 5.2.25.9, defines an identifier for an NVDIMM, which reflects the id attribute.

/sys/bus/nd/devices/nmemX/nfit/phys_id

Defined on file *sysfs-bus-nfit*

(RO) Handle (i.e., instance number) for the SMBIOS (system management BIOS) Memory Device structure describing the NVDIMM containing the NVDIMM region.

/sys/bus/nd/devices/nmemX/nfit/rev_id

Defined on file *sysfs-bus-nfit*

(RO) Revision of the NVDIMM, assigned by the module vendor.

/sys/bus/nd/devices/nmemX/nfit/serial

Defined on file *sysfs-bus-nfit*

(RO) Serial number of the NVDIMM (non-volatile dual in-line memory module), assigned by the module vendor.

/sys/bus/nd/devices/nmemX/nfit/subsystem_device

Defined on file *sysfs-bus-nfit*

(RO) Sub-system device id for the NVDIMM non-volatile memory subsystem controller, assigned by the non-volatile memory subsystem controller vendor.

/sys/bus/nd/devices/nmemX/nfit/subsystem_rev_id

Defined on file *sysfs-bus-nfit*

(RO) Sub-system revision id of the NVDIMM non-volatile memory subsystem controller, assigned by the non-volatile memory subsystem controller vendor.

/sys/bus/nd/devices/nmemX/nfit/subsystem_vendor

Defined on file *sysfs-bus-nfit*

(RO) Sub-system vendor id of the NVDIMM non-volatile memory subsystem controller.

/sys/bus/nd/devices/nmemX/nfit/vendor

Defined on file *sysfs-bus-nfit*

(RO) Vendor id of the NVDIMM.

/sys/bus/nd/devices/nmemX/papr/flags

Defined on file *sysfs-bus-papr-pmem*

(RO) Report flags indicating various states of a papr-pmem NVDIMM device. Each flag maps to a one or more bits set in the dimm-health-bitmap retrieved in response to H_SCM_HEALTH hcall. The details of the bit flags returned in response to this hcall is available at '/arch/powerpc/papr_hcalls' . Below are the flags reported in this sysfs file:

- **"not_armed"**
  Indicates that NVDIMM contents will not survive a power cycle.
- **"flush_fail"**
  Indicates that NVDIMM contents couldn't be flushed during last shut-down event.
- **"restore_fail"**
  Indicates that NVDIMM contents couldn't be restored during NVDIMM initialization.
- **"encrypted"**
  NVDIMM contents are encrypted.
- **"smart_notify"**
  There is health event for the NVDIMM.
- **"scrubbed"**
  Indicating that contents of the NVDIMM have been scrubbed.
- **"locked"**
  Indicating that NVDIMM contents can't be modified until next power cycle.

/sys/bus/nd/devices/nmemX/papr/health_bitmap_inject

Defined on file *sysfs-bus-papr-pmem*

(RO) Reports the health bitmap inject bitmap that is applied to bitmap received from PowerVM via the H_SCM_HEALTH. This is used to forcibly set specific bits returned from Hcall. These is then used to simulate various health or shutdown states for an nvdimm and are set by user-space tools like ndctl by issuing a PAPR DSM.

/sys/bus/nd/devices/nmemX/papr/perf_stats

Defined on file *sysfs-bus-papr-pmem*

(RO) Report various performance stats related to papr-scm NVDIMM device. This attribute is only available for NVDIMM devices that support reporting NVDIMM performance stats. Each stat is reported on a new line with each line composed of a stat-identifier followed by it value. Below are currently known dimm performance stats which are reported:

- ”CtlResCt” : Controller Reset Count
- ”CtlResTm” : Controller Reset Elapsed Time
- ”PonSecs ” : Power-on Seconds
- ”MemLife ” : Life Remaining
- ”CritRscU” : Critical Resource Utilization
- ”HostLCnt” : Host Load Count
- ”HostSCnt” : Host Store Count
- ”HostSDur” : Host Store Duration
- ”HostLDur” : Host Load Duration
- ”MedRCnt ” : Media Read Count
- ”MedWCnt ” : Media Write Count
- ”MedRDur ” : Media Read Duration
- ”MedWDur ” : Media Write Duration
- ”CchRHCnt” : Cache Read Hit Count
- ”CchWHCnt” : Cache Write Hit Count
- ”FastWCnt” : Fast Write Count

/sys/bus/nd/devices/regionX/nfit/range_index

Defined on file *sysfs-bus-nfit*

(RO) A unique number provided by the BIOS to identify an address range. Used by NVDIMM Region Mapping Structure to uniquely refer to this structure. Value of 0 is reserved and not used as an index.

/sys/bus/nvmem/devices/.../cells/<cell-name>

Defined on file *sysfs-nvmem-cells*

The "cells" folder contains one file per cell exposed by the NVMEM device. The name of the file is: "<name>@<byte>,<bit>", with <name> being the cell name and <where> its location in the NVMEM device, in hexadecimal bytes and bits (without the '0x' prefix, to mimic device tree node names). The length of the file is the size of the cell (when known). The content of the file is the binary content of the cell (may sometimes be ASCII, likely without trailing character). Note: This file is only present if CONFIG_NVMEM_SYSFS is enabled.

Example:

```
hexdump -C /sys/bus/nvmem/devices/1-00563/cells/product-name@d,0
00000000  54 4e 34 38 4d 2d 50 2d  44 4e         |TN48M-P-DN|
0000000a
```

/sys/bus/pci/devices/.../acpi_index

Defined on file *sysfs-bus-pci*

Reading this attribute will provide the firmware given instance (ACPI _DSM instance number) of the PCI device. The attribute will be created only if the firmware has given an instance number to the PCI device. ACPI _DSM instance number will be given priority if the system firmware provides SMBIOS type 41 device type instance also.

Users:

Userspace applications interested in knowing the firmware assigned instance number of the PCI device that can help in understanding the firmware intended order of the PCI device.

/sys/bus/pci/devices/.../d3cold_allowed

Defined on file *sysfs-bus-pci*

d3cold_allowed is bit to control whether the corresponding PCI device can be put into D3Cold state. If it is cleared, the device will never be put into D3Cold state. If it is set, the device may be put into D3Cold state if other requirements are satisfied too. Reading this attribute will show the current value of d3cold_allowed bit. Writing this attribute will set the value of d3cold_allowed bit.

/sys/bus/pci/devices/.../dep_link

Defined on file *sysfs-bus-pci*

This symbolic link appears when hardware supports the SR-IOV capability and the Physical Function driver has enabled it, and this device has vendor specific dependencies with others. The symbolic link points to the PCI device sysfs entry of Physical Function this device depends on.

/sys/bus/pci/devices/.../driver_override

Defined on file *sysfs-bus-pci*

This file allows the driver for a device to be specified which will override standard static and dynamic ID matching. When specified, only a driver with a name matching the value written to driver_override will have an opportunity to bind to the device. The override is specified by writing a string to the driver_override file (echo pci-stub > driver_override) and may be cleared with an empty string (echo > driver_override). This returns the device to standard matching rules binding. Writing to driver_override does not automatically unbind the device from its current driver or make any attempt to automatically load the specified driver. If no driver with a matching name is currently loaded in the kernel, the device will not bind to any driver. This also allows devices to opt-out of driver binding using a driver_override name such as "none". Only a single driver may be specified in the override, there is no support for parsing delimiters.

/sys/bus/pci/devices/.../index

Defined on file *sysfs-bus-pci*

Reading this attribute will provide the firmware given instance number of the PCI device. Depending on the platform this can be for example the SMBIOS type 41 device type instance or the user-defined ID (UID) on s390. The attribute will be created only if the firmware has given an instance number to the PCI device and that number is guaranteed to uniquely identify the device in the system.

Users:

Userspace applications interested in knowing the firmware assigned device type instance of the PCI device that can help in understanding the firmware intended order of the PCI device.

/sys/bus/pci/devices/.../irq

Defined on file *sysfs-bus-pci*

If a driver has enabled MSI (not MSI-X), "irq" contains the IRQ of the first MSI vector. Otherwise "irq" contains the IRQ of the legacy INTx interrupt.

"irq" being set to 0 indicates that the device isn't capable of generating legacy INTx interrupts.

/sys/bus/pci/devices/.../label

Defined on file *sysfs-bus-pci*

Reading this attribute will provide the firmware given name (SMBIOS type 41 string or ACPI _DSM string) of the PCI device. The attribute will be created only if the firmware has given a name to the PCI device. ACPI _DSM string name will be given priority if the system firmware provides SMBIOS type 41 string also.

Users:

Userspace applications interested in knowing the firmware assigned name of the PCI device.

/sys/bus/pci/devices/.../link/clkpm

Defined on file *sysfs-bus-pci*

If ASPM is supported for an endpoint, these files can be used to disable or enable the individual power management states. Write y/1/on to enable, n/0/off to disable.

/sys/bus/pci/devices/.../modalias

Defined on file *sysfs-bus-pci*

This attribute indicates the PCI ID of the device object.

That is in the format: pci:vXXXXXXXXdXXXXXXXXsvXXXXXXXXsdXXXXXXXXbcXXscXXiXX, where:

- vXXXXXXXX contains the vendor ID;
- dXXXXXXXX contains the device ID;
- svXXXXXXXX contains the sub-vendor ID;
- sdXXXXXXXX contains the subsystem device ID;
- bcXX contains the device class;
- scXX contains the device subclass;
- iXX contains the device class programming interface.

/sys/bus/pci/devices/.../msi_bus

Defined on file *sysfs-bus-pci*

Writing a zero value to this attribute disallows MSI and MSI-X for any future drivers of the device. If the device is a bridge, MSI and MSI-X will be disallowed for future drivers of all child devices under the bridge. Drivers must be reloaded for the new setting to take effect.

/sys/bus/pci/devices/.../msi_irqs/

Defined on file *sysfs-bus-pci*

The /sys/devices/.../msi_irqs directory contains a variable set of files, with each file being named after a corresponding msi irq vector allocated to that device.

/sys/bus/pci/devices/.../msi_irqs/<N>

Defined on file *sysfs-bus-pci*

This attribute indicates the mode that the irq vector named by the file is in (msi vs. msix)

/sys/bus/pci/devices/.../numa_node

Defined on file *sysfs-bus-pci*

This file contains the NUMA node to which the PCI device is attached, or -1 if the node is unknown. The initial value comes from an ACPI _PXM method or a similar firmware source. If that is missing or incorrect, this file can be written to override the node. In that case, please report a firmware bug to the system vendor. Writing to this file taints the kernel with TAINT_FIRMWARE_WORKAROUND, which reduces the supportability of your system.

/sys/bus/pci/devices/.../p2pmem/allocate

Defined on file *sysfs-bus-pci*

This file allows mapping p2pmem into userspace. For each mmap() call on this file, the kernel will allocate a chunk of Peer-to-Peer memory for use in Peer-to-Peer transactions. This memory can be used in O_DIRECT calls to NVMe backed files for Peer-to-Peer copies.

/sys/bus/pci/devices/.../p2pmem/available

Defined on file *sysfs-bus-pci*

If the device has any Peer-to-Peer memory registered, this file contains the amount of memory that has not been allocated (in decimal).

/sys/bus/pci/devices/.../p2pmem/published

Defined on file *sysfs-bus-pci*

If the device has any Peer-to-Peer memory registered, this file contains a '1' if the memory has been published for use outside the driver that owns the device.

/sys/bus/pci/devices/.../p2pmem/size

Defined on file *sysfs-bus-pci*

If the device has any Peer-to-Peer memory registered, this file contains the total amount of memory that the device provides (in decimal).

/sys/bus/pci/devices/.../pci_bus/.../rescan

Defined on file *sysfs-bus-pci*

Writing a non-zero value to this attribute will force a rescan of the bus and all child buses, and re-discover devices removed earlier from this part of the device tree.

/sys/bus/pci/devices/.../physfn

Defined on file *sysfs-bus-pci*

This symbolic link appears when a device is a Virtual Function. The symbolic link points to the PCI device sysfs entry of the Physical Function this device associates with.

/sys/bus/pci/devices/.../power_state

Defined on file *sysfs-bus-pci*

This file contains the current PCI power state of the device. The value comes from the PCI kernel device state and can be one of: ”unknown”, ”error”, ”D0”, D1”, ”D2”, ”D3hot”, ”D3cold”. The file is read only.

/sys/bus/pci/devices/.../remove

Defined on file *sysfs-bus-pci*

Writing a non-zero value to this attribute will hot-remove the PCI device and any of its children.

/sys/bus/pci/devices/.../rescan

Defined on file *sysfs-bus-pci*

Writing a non-zero value to this attribute will force a rescan of the device's parent bus and all child buses, and re-discover devices removed earlier from this part of the device tree.

/sys/bus/pci/devices/.../reset

Defined on file *sysfs-bus-pci*

Some devices allow an individual function to be reset without affecting other functions in the same device. For devices that have this support, a file named reset will be present in sysfs. Writing 1 to this file will perform reset.

/sys/bus/pci/devices/.../reset_method

Defined on file *sysfs-bus-pci*

Some devices allow an individual function to be reset without affecting other functions in the same slot.

For devices that have this support, a file named reset_method is present in sysfs. Reading this file gives names of the supported and enabled reset methods and their ordering. Writing a space-separated list of names of reset methods sets the reset methods and ordering to be used when resetting the device. Writing an empty string disables the ability to reset the device. Writing "default" enables all supported reset methods in the default ordering.

/sys/bus/pci/devices/.../resourceN_resize

Defined on file *sysfs-bus-pci*

These files provide an interface to PCIe Resizable BAR support. A file is created for each BAR resource (N) supported by the PCIe Resizable BAR extended capability of the device. Reading each file exposes the bitmap of available resource sizes:

cat resource1_resize 00000000000001c0

The bitmap represents supported resource sizes for the BAR, where bit0 = 1MB, bit1 = 2MB, bit2 = 4MB, etc. In the above example the device supports 64MB, 128MB, and 256MB BAR sizes.

When writing the file, the user provides the bit position of the desired resource size, for example:

echo 7 > resource1_resize

This indicates to set the size value corresponding to bit 7, 128MB. The resulting size is 2 ^ (bit# + 20). This definition matches the PCIe specification of this capability.

In order to make use of resource resizing, all PCI drivers must be unbound from the device and peer devices under the same parent bridge may need to be soft removed. In the case of VGA devices, writing a resize value will remove low level console drivers from the device. Raw users of pci-sysfs resourceN attributes must be terminated prior to resizing. Success of the resizing operation is not guaranteed.

/sys/bus/pci/devices/.../revision

Defined on file *sysfs-bus-pci*

This file contains the revision field of the PCI device. The value comes from device config space. The file is read only.

/sys/bus/pci/devices/.../sriov_drivers_autoprobe

Defined on file *sysfs-bus-pci*

This file is associated with the PF of a device that supports SR-IOV. It determines whether newly-enabled VFs are immediately bound to a driver. It initially contains 1, which means the kernel automatically binds VFs to a compatible driver immediately after they are enabled. If an application writes 0 to the file before enabling VFs, the kernel will not bind VFs to a driver.

A typical use case is to write 0 to this file, then enable VFs, then assign the newly-created VFs to virtual machines. Note that changing this file does not affect already- enabled VFs. In this scenario, the user must first disable the VFs, write 0 to sriov_drivers_autoprobe, then re-enable the VFs.

This is similar to /sys/bus/pci/drivers_autoprobe, but affects only the VFs associated with a specific PF.

/sys/bus/pci/devices/.../sriov_numvfs

Defined on file *sysfs-bus-pci*

This file appears when a physical PCIe device supports SR-IOV. Userspace applications can read and write to this file to determine and control the enablement or disablement of Virtual Functions (VFs) on the physical function (PF). A read of this file will return the number of VFs that are enabled on this PF. A number written to this file will enable the specified number of VFs. A userspace application would typically read the file and check that the value is zero, and then write the number of VFs that should be enabled on the PF; the value written should be less than or equal to the value in the sriov_totalvfs file. A userspace application wanting to disable the VFs would write a zero to this file. The core ensures that valid values are written to this file, and returns errors when values are not valid. For example, writing a 2 to this file when sriov_numvfs is not 0 and not 2 already will return an error. Writing a 10 when the value of sriov_totalvfs is 8 will return an error.

/sys/bus/pci/devices/.../sriov_totalvfs

Defined on file *sysfs-bus-pci*

This file appears when a physical PCIe device supports SR-IOV. Userspace applications can read this file to determine the maximum number of Virtual Functions (VFs) a PCIe physical function (PF) can support. Typically, this is the value reported in the PF's SR-IOV extended capability

structure's TotalVFs element. Drivers have the ability at probe time to reduce the value read from this file via the pci_sriov_set_totalvfs() function.

/sys/bus/pci/devices/.../sriov_vf_msix_count

Defined on file *sysfs-bus-pci*

This file is associated with a SR-IOV virtual function (VF). It allows configuration of the number of MSI-X vectors for the VF. This allows devices that have a global pool of MSI-X vectors to optimally divide them between VFs based on VF usage.

The values accepted are:

- **> 0 - this number will be reported as the Table Size in the** VF's MSI-X capability
- < 0 - not valid
- = 0 - will reset to the device default value

The file is writable if the PF is bound to a driver that implements ->sriov_set_msix_vec_count().

/sys/bus/pci/devices/.../sriov_vf_total_msix

Defined on file *sysfs-bus-pci*

This file is associated with a SR-IOV physical function (PF). It contains the total number of MSI-X vectors available for assignment to all virtual functions (VFs) associated with PF. The value will be zero if the device doesn't support this functionality. For supported devices, the value will be constant and won't be changed after MSI-X vectors assignment.

/sys/bus/pci/devices/.../virtfn<N>

Defined on file *sysfs-bus-pci*

This symbolic link appears when hardware supports the SR-IOV capability and the Physical Function driver has enabled it. The symbolic link points to the PCI device sysfs entry of the Virtual Function whose index is N (0...MaxVFs-1).

/sys/bus/pci/devices/.../vpd

Defined on file *sysfs-bus-pci*

A file named vpd in a device directory will be a binary file containing the Vital Product Data for the device. It should follow the VPD format defined in PCI Specification 2.1 or 2.2, but users should consider that some devices may have incorrectly formatted data. If the underlying VPD has a writable section then the corresponding section of this file will be writable.

/sys/bus/pci/devices/<BDF>/anti_rollback_status

Defined on file *sysfs-driver-ccp*

The */sys/bus/pci/devices/<BDF>/anti_rollback_status* file reports whether the PSP is enforcing rollback protection. Possible values: 0: Not enforcing 1: Enforcing

/sys/bus/pci/devices/<BDF>/bootloader_version

Defined on file *sysfs-driver-ccp*

The */sys/bus/pci/devices/<BDF>/bootloader_version* file reports the firmware version of the AMD AGESA bootloader.

/sys/bus/pci/devices/<BDF>/debug_lock_on

Defined on file *sysfs-driver-ccp*

The */sys/bus/pci/devices/<BDF>/debug_lock_on* reports whether the AMD CPU or APU has been unlocked for debugging. Possible values: 0: Not locked 1: Locked

/sys/bus/pci/devices/<BDF>/fused_part

Defined on file *sysfs-driver-ccp*

The */sys/bus/pci/devices/<BDF>/fused_part* file reports whether the CPU or APU has been fused to prevent tampering. 0: Not fused 1: Fused

/sys/bus/pci/devices/<BDF>/hsp_tpm_available

Defined on file *sysfs-driver-ccp*

The */sys/bus/pci/devices/<BDF>/hsp_tpm_available* file reports whether the HSP TPM has been activated. Possible values: 0: Not activated or present 1: Activated

/sys/bus/pci/devices/<BDF>/qat/cfg_services

Defined on file *sysfs-driver-qat*

(RW) Reports the current configuration of the QAT device. Write to the file to change the configured services.

The values are:

- sym;asym: the device is configured for running crypto services
- asym;sym: identical to sym;asym
- dc: the device is configured for running compression services
- dcc: identical to dc but enables the dc chaining feature, hash then compression. If this is not required chose dc
- sym: the device is configured for running symmetric crypto services
- asym: the device is configured for running asymmetric crypto services
- asym;dc: the device is configured for running asymmetric crypto services and compression services
- dc;asym: identical to asym;dc

- sym;dc: the device is configured for running symmetric crypto services and compression services
- dc;sym: identical to sym;dc

It is possible to set the configuration only if the device is in the *down* state (see */sys/bus/pci/devices/<BDF>/qat/state*)

The following example shows how to change the configuration of a device configured for running crypto services in order to run data compression:

```
# cat /sys/bus/pci/devices/<BDF>/qat/state
up
# cat /sys/bus/pci/devices/<BDF>/qat/cfg_services
sym;asym
# echo down > /sys/bus/pci/devices/<BDF>/qat/state
# echo dc > /sys/bus/pci/devices/<BDF>/qat/cfg_services
# echo up > /sys/bus/pci/devices/<BDF>/qat/state
# cat /sys/bus/pci/devices/<BDF>/qat/cfg_services
dc
```

This attribute is only available for qat_4xxx devices.

/sys/bus/pci/devices/<BDF>/qat/num_rps

Defined on file *sysfs-driver-qat*

(RO) Returns the number of ring pairs that a single device has.

Example usage:

```
# cat /sys/bus/pci/devices/<BDF>/qat/num_rps
64
```

This attribute is only available for qat_4xxx devices.

/sys/bus/pci/devices/<BDF>/qat/pm_idle_enabled

Defined on file *sysfs-driver-qat*

(RW) This configuration option provides a way to force the device into remaining in the MAX power state. If idle support is enabled the device will transition to the *MIN* power state when idle, otherwise will stay in the MAX power state. Write to the file to enable or disable idle support.

The values are:

- 0: idle support is disabled
- 1: idle support is enabled

Default value is 1.

It is possible to set the pm_idle_enabled value only if the device is in the *down* state (see */sys/bus/pci/devices/<BDF>/qat/state*)

The following example shows how to change the pm_idle_enabled of a device:

```
# cat /sys/bus/pci/devices/<BDF>/qat/state
up
# cat /sys/bus/pci/devices/<BDF>/qat/pm_idle_enabled
1
# echo down > /sys/bus/pci/devices/<BDF>/qat/state
# echo 0 > /sys/bus/pci/devices/<BDF>/qat/pm_idle_enabled
# echo up > /sys/bus/pci/devices/<BDF>/qat/state
# cat /sys/bus/pci/devices/<BDF>/qat/pm_idle_enabled
0
```

This attribute is only available for qat_4xxx devices.

/sys/bus/pci/devices/<BDF>/qat/rp2srv

Defined on file *sysfs-driver-qat*

(RW) This attribute provides a way for a user to query a specific ring pair for the type of service that it is currently configured for.

When written to, the value is cached and used to perform the read operation. Allowed values are in the range 0 to N-1, where N is the max number of ring pairs supported by a device. This can be queried using the attribute qat/num_rps.

A read returns the service associated to the ring pair queried.

The values are:

- dc: the ring pair is configured for running compression services
- sym: the ring pair is configured for running symmetric crypto services
- asym: the ring pair is configured for running asymmetric crypto services

Example usage:

```
# echo 1 > /sys/bus/pci/devices/<BDF>/qat/rp2srv
# cat /sys/bus/pci/devices/<BDF>/qat/rp2srv
sym
```

This attribute is only available for qat_4xxx devices.

/sys/bus/pci/devices/<BDF>/qat/state

Defined on file *sysfs-driver-qat*

(RW) Reports the current state of the QAT device. Write to the file to start or stop the device.

The values are:

- up: the device is up and running
- down: the device is down

It is possible to transition the device from up to down only if the device is up and vice versa.

This attribute is only available for qat_4xxx devices.

/sys/bus/pci/devices/<BDF>/qat_ras/errors_correctable

Defined on file *sysfs-driver-qat_ras*

(RO) Reports the number of correctable errors detected by the device.

This attribute is only available for qat_4xxx devices.

/sys/bus/pci/devices/<BDF>/qat_ras/errors_fatal

Defined on file *sysfs-driver-qat_ras*

(RO) Reports the number of fatal errors detected by the device.

This attribute is only available for qat_4xxx devices.

/sys/bus/pci/devices/<BDF>/qat_ras/errors_nonfatal

Defined on file *sysfs-driver-qat_ras*

(RO) Reports the number of non fatal errors detected by the device.

This attribute is only available for qat_4xxx devices.

/sys/bus/pci/devices/<BDF>/qat_ras/reset_error_counters

Defined on file *sysfs-driver-qat_ras*

(WO) Write to resets all error counters of a device.

The following example reports how to reset the counters:

```
# echo 1 > /sys/bus/pci/devices/<BDF>/qat_ras/reset_error_counters
# cat /sys/bus/pci/devices/<BDF>/qat_ras/errors_correctable
0
# cat /sys/bus/pci/devices/<BDF>/qat_ras/errors_nonfatal
0
# cat /sys/bus/pci/devices/<BDF>/qat_ras/errors_fatal
0
```

This attribute is only available for qat_4xxx devices.

/sys/bus/pci/devices/<BDF>/qat_rl/cap_rem

Defined on file *sysfs-driver-qat_rl*

(RW) This file will return the remaining capability for a particular service/sla. This is the remaining value that a new SLA can be set to or a current SLA can be increased with.

Example usage:

```
# echo "asym" > /sys/bus/pci/devices/<BDF>/qat_rl/cap_rem
# cat /sys/bus/pci/devices/<BDF>/qat_rl/cap_rem
250
```

```
# echo 250 > /sys/bus/pci/devices/<BDF>/qat_rl/cir
# echo "add" > /sys/bus/pci/devices/<BDF>/qat_rl/sla_op
# cat /sys/bus/pci/devices/<BDF>/qat_rl/cap_rem
0
```

This attribute is only available for qat_4xxx devices.

/sys/bus/pci/devices/<BDF>/qat_rl/cir

Defined on file *sysfs-driver-qat_rl*

(RW) Committed information rate (CIR). Rate guaranteed to be achieved by a particular SLA. The value is expressed in permille scale, i.e. 1000 refers to the maximum device throughput for a selected service.

After sending a "get" to sla_op, this will be populated with the CIR for that queried SLA. Write to this file before sending an "add/update" sla_op, to set the SLA to the specified value.

Applicability in sla_op:

- WRITE: add and update operations
- READ: get operation

Example usage:

```
## Write
# echo 500 > /sys/bus/pci/devices/<BDF>/qat_rl/cir
# echo "add" /sys/bus/pci/devices/<BDF>/qat_rl/sla_op

## Read
# echo 4 > /sys/bus/pci/devices/<BDF>/qat_rl/id
# echo "get" > /sys/bus/pci/devices/<BDF>/qat_rl/sla_op
# cat /sys/bus/pci/devices/<BDF>/qat_rl/cir
500
```

This attribute is only available for qat_4xxx devices.

/sys/bus/pci/devices/<BDF>/qat_rl/id

Defined on file *sysfs-driver-qat_rl*

(RW) If written to, the value is used to retrieve a particular SLA and operate on it. This is valid only for the following operations: update, rm, and get. A read of this attribute is only guaranteed to have correct data after creation of an SLA.

Applicability in sla_op:

- WRITE: rm and update operations
- READ: add and get operations

Example usage:

```
## Read
## Set attributes e.g. cir, pir, srv, etc
# echo "add" > /sys/bus/pci/devices/<BDF>/qat_rl/sla_op
# cat /sys/bus/pci/devices/<BDF>/qat_rl/id
4

## Write
# echo 7 > /sys/bus/pci/devices/<BDF>/qat_rl/id
# echo "get" > /sys/bus/pci/devices/<BDF>/qat_rl/sla_op
# cat /sys/bus/pci/devices/<BDF>/qat_rl/rp
0x5  ## ring pair ID 0 and ring pair ID 2
```

This attribute is only available for qat_4xxx devices.

/sys/bus/pci/devices/<BDF>/qat_rl/pir

Defined on file *sysfs-driver-qat_rl*

(RW) Peak information rate (PIR). The maximum rate that can be achieved by that particular SLA. An SLA can reach a value between CIR and PIR when the device is not fully utilized by requests from other users (assigned to different SLAs).

After sending a "get" to sla_op, this will be populated with the PIR for that queried SLA. Write to this file before sending an "add/update" sla_op, to set the SLA to the specified value.

Applicability in sla_op:

- WRITE: add and update operations
- READ: get operation

Example usage:

```
## Write
# echo 750 > /sys/bus/pci/devices/<BDF>/qat_rl/pir
# echo "add" > /sys/bus/pci/devices/<BDF>/qat_rl/sla_op

## Read
# echo 4 > /sys/bus/pci/devices/<BDF>/qat_rl/id
# echo "get" > /sys/bus/pci/devices/<BDF>/qat_rl/sla_op
# cat /sys/bus/pci/devices/<BDF>/qat_rl/pir
750
```

This attribute is only available for qat_4xxx devices.

/sys/bus/pci/devices/<BDF>/qat_rl/rp

Defined on file *sysfs-driver-qat_rl*

(RW) When read, reports the current assigned ring pairs for the queried SLA. When wrote to, configures the ring pairs associated to a new SLA.

The value is a 64-bit bit mask and is written/displayed in hex. Each bit of this mask represents a single ring pair i.e., bit 1 == ring pair id 0; bit 3 == ring pair id 2.

Selected ring pairs must to be assigned to a single service, i.e. the one provided with the srv attribute. The service assigned to a certain ring pair can be checked by querying the attribute qat/rp2srv.

The maximum number of ring pairs is 4 per SLA.

Applicability in sla_op:

- WRITE: add operation
- READ: get operation

Example usage:

```
## Read
# echo 4 > /sys/bus/pci/devices/<BDF>/qat_rl/id
# cat /sys/bus/pci/devices/<BDF>/qat_rl/rp
0x5

## Write
# echo 0x5 > /sys/bus/pci/devices/<BDF>/qat_rl/rp
```

This attribute is only available for qat_4xxx devices.

/sys/bus/pci/devices/<BDF>/qat_rl/sla_op

Defined on file *sysfs-driver-qat_rl*

(WO) This attribute is used to perform an operation on an SLA. The supported operations are: add, update, rm, rm_all, and get.

Input values must be filled through the associated attribute in this group before a write to this file. If the operation completes successfully, the associated attributes will be updated. The associated attributes are: cir, pir, srv, rp, and id.

Supported operations:

- **add: Creates a new SLA with the provided inputs from user.**
  - Inputs: cir, pir, srv, and rp
  - Output: id
- **get: Returns the configuration of the specified SLA in id attribute**
  - Inputs: id
  - Outputs: cir, pir, srv, and rp
- **update: Updates the SLA with new values set in the following attributes**
  - Inputs: id, cir, and pir
- **rm: Removes the specified SLA in the id attribute.**
  - Inputs: id
- **rm_all: Removes all the configured SLAs.**
  - Inputs: None

This attribute is only available for qat_4xxx devices.

/sys/bus/pci/devices/<BDF>/qat_rl/srv

Defined on file *sysfs-driver-qat_rl*

(RW) Service (SRV). Represents the service (sym, asym, dc) associated to an SLA. Can be written to or queried to set/show the SRV type for an SLA. The SRV attribute is used to specify the SRV type before adding an SLA. After an SLA is configured, reports the service associated to that SLA.

Applicability in sla_op:

- WRITE: add and update operations
- READ: get operation

Example usage:

```
## Write
# echo "dc" > /sys/bus/pci/devices/<BDF>/qat_rl/srv
# echo "add" > /sys/bus/pci/devices/<BDF>/qat_rl/sla_op
# cat /sys/bus/pci/devices/<BDF>/qat_rl/id
4

## Read
# echo 4 > /sys/bus/pci/devices/<BDF>/qat_rl/id
# echo "get" > /sys/bus/pci/devices/<BDF>/qat_rl/sla_op
# cat /sys/bus/pci/devices/<BDF>/qat_rl/srv
dc
```

This attribute is only available for qat_4xxx devices.

/sys/bus/pci/devices/<BDF>/rom_armor_enforced

Defined on file *sysfs-driver-ccp*

The */sys/bus/pci/devices/<BDF>/rom_armor_enforced* file reports whether RomArmor SPI protection is enforced. Possible values: 0: Not enforced 1: Enforced

/sys/bus/pci/devices/<BDF>/rpmc_production_enabled

Defined on file *sysfs-driver-ccp*

The */sys/bus/pci/devices/<BDF>/rpmc_production_enabled* file reports whether Replay Protected Monotonic Counter support has been enabled. Possible values: 0: Not enabled 1: Enabled

/sys/bus/pci/devices/<BDF>/rpmc_spirom_available

Defined on file *sysfs-driver-ccp*

The */sys/bus/pci/devices/<BDF>/rpmc_spirom_available* file reports whether an Replay Protected Monotonic Counter supported SPI is installed on the system. Possible values: 0: Not present 1: Present

/sys/bus/pci/devices/<BDF>/tee_version

Defined on file *sysfs-driver-ccp*

The */sys/bus/pci/devices/<BDF>/tee_version* file reports the firmware version of the AMD Trusted Execution Environment (TEE).

/sys/bus/pci/devices/<BDF>/tsme_status

Defined on file *sysfs-driver-ccp*

The */sys/bus/pci/devices/<BDF>/tsme_status* file reports the status of transparent secure memory encryption on AMD systems. Possible values: 0: Not active 1: Active

/sys/bus/pci/devices/<dev>/aer_dev_correctable

Defined on file *sysfs-bus-pci-devices-aer_stats*

List of correctable errors seen and reported by this PCI device using ERR_COR. Note that since multiple errors may be reported using a single ERR_COR message, thus TOTAL_ERR_COR at the end of the file may not match the actual total of all the errors in the file. Sample output:

```
localhost /sys/devices/pci0000:00/0000:00:1c.0 # cat aer_dev_correctable
Receiver Error 2
Bad TLP 0
Bad DLLP 0
RELAY_NUM Rollover 0
Replay Timer Timeout 0
Advisory Non-Fatal 0
Corrected Internal Error 0
Header Log Overflow 0
TOTAL_ERR_COR 2
```

/sys/bus/pci/devices/<dev>/aer_dev_fatal

Defined on file *sysfs-bus-pci-devices-aer_stats*

List of uncorrectable fatal errors seen and reported by this PCI device using ERR_FATAL. Note that since multiple errors may be reported using a single ERR_FATAL message, thus TOTAL_ERR_FATAL at the end of the file may not match the actual total of all the errors in the file. Sample output:

```
localhost /sys/devices/pci0000:00/0000:00:1c.0 # cat aer_dev_fatal
Undefined 0
Data Link Protocol 0
Surprise Down Error 0
Poisoned TLP 0
```

```
Flow Control Protocol 0
Completion Timeout 0
Completer Abort 0
Unexpected Completion 0
Receiver Overflow 0
Malformed TLP 0
ECRC 0
Unsupported Request 0
ACS Violation 0
Uncorrectable Internal Error 0
MC Blocked TLP 0
AtomicOp Egress Blocked 0
TLP Prefix Blocked Error 0
TOTAL_ERR_FATAL 0
```

/sys/bus/pci/devices/<dev>/aer_dev_nonfatal

Defined on file *sysfs-bus-pci-devices-aer_stats*

List of uncorrectable nonfatal errors seen and reported by this PCI device using ERR_NONFATAL. Note that since multiple errors may be reported using a single ERR_FATAL message, thus TOTAL_ERR_NONFATAL at the end of the file may not match the actual total of all the errors in the file. Sample output:

```
localhost /sys/devices/pci0000:00/0000:00:1c.0 # cat aer_dev_nonfatal
Undefined 0
Data Link Protocol 0
Surprise Down Error 0
Poisoned TLP 0
Flow Control Protocol 0
Completion Timeout 0
Completer Abort 0
Unexpected Completion 0
Receiver Overflow 0
Malformed TLP 0
ECRC 0
Unsupported Request 0
ACS Violation 0
Uncorrectable Internal Error 0
MC Blocked TLP 0
AtomicOp Egress Blocked 0
TLP Prefix Blocked Error 0
TOTAL_ERR_NONFATAL 0
```

PCIe Rootport AER statistics

These attributes show up under only the rootports (or root complex event collectors) that are AER capable. These indicate the number of error messages as ”reported to” the rootport. Please note that the rootports also transmit (internally) the ERR_* messages for errors seen by the internal rootport PCI device, so these counters include them and are thus cumulative of all the error messages on the PCI hierarchy originating at that root port.

/sys/bus/pci/devices/<dev>/aer_stats/aer_rootport_total_err_cor

Defined on file *sysfs-bus-pci-devices-aer_stats*

Total number of ERR_COR messages reported to rootport.

/sys/bus/pci/devices/<dev>/aer_stats/aer_rootport_total_err_fatal

Defined on file *sysfs-bus-pci-devices-aer_stats*

Total number of ERR_FATAL messages reported to rootport.

/sys/bus/pci/devices/<dev>/aer_stats/aer_rootport_total_err_nonfatal

Defined on file *sysfs-bus-pci-devices-aer_stats*

Total number of ERR_NONFATAL messages reported to rootport.

/sys/bus/pci/devices/<dev>/ccissX/cXdY/block:cciss!cXdY

Defined on file *sysfs-bus-pci-devices-cciss*

A symbolic link to /sys/block/cciss!cXdY

/sys/bus/pci/devices/<dev>/ccissX/cXdY/lunid

Defined on file *sysfs-bus-pci-devices-cciss*

Displays the 8-byte LUN ID used to address logical drive Y of controller X.

/sys/bus/pci/devices/<dev>/ccissX/cXdY/model

Defined on file *sysfs-bus-pci-devices-cciss*

Displays the SCSI INQUIRY page 0 model for logical drive Y of controller X.

/sys/bus/pci/devices/<dev>/ccissX/cXdY/raid_level

Defined on file *sysfs-bus-pci-devices-cciss*

Displays the RAID level of logical drive Y of controller X.

/sys/bus/pci/devices/<dev>/ccissX/cXdY/rev

Defined on file *sysfs-bus-pci-devices-cciss*

Displays the SCSI INQUIRY page 0 revision for logical drive Y of controller X.

/sys/bus/pci/devices/<dev>/ccissX/cXdY/unique_id

Defined on file *sysfs-bus-pci-devices-cciss*

Displays the SCSI INQUIRY page 83 serial number for logical drive Y of controller X.

/sys/bus/pci/devices/<dev>/ccissX/cXdY/usage_count

Defined on file *sysfs-bus-pci-devices-cciss*

Displays the usage count (number of opens) of logical drive Y of controller X.

/sys/bus/pci/devices/<dev>/ccissX/cXdY/vendor

Defined on file *sysfs-bus-pci-devices-cciss*

Displays the SCSI INQUIRY page 0 vendor for logical drive Y of controller X.

/sys/bus/pci/devices/<dev>/ccissX/rescan

Defined on file *sysfs-bus-pci-devices-cciss*

Kicks of a rescan of the controller to discover logical drive topology changes.

/sys/bus/pci/devices/<dev>/ccissX/resettable

Defined on file *sysfs-bus-pci-devices-cciss*

Value of 1 indicates the controller can honor the reset_devices kernel parameter. Value of 0 indicates reset_devices cannot be honored. This is to allow, for example, kexec tools to be able to warn the user if they designate an unresettable device as a dump device, as kdump requires resetting the device in order to work reliably.

/sys/bus/pci/devices/<dev>/ccissX/transport_mode

Defined on file *sysfs-bus-pci-devices-cciss*

Value of ”simple” indicates that the controller has been placed in ”simple mode”. Value of ”performant” indicates that the controller has been placed in ”performant mode”.

/sys/bus/pci/drivers/.../bind
/sys/devices/pciX/.../bind

Defined on file *sysfs-bus-pci*

Writing a device location to this file will cause the driver to attempt to bind to the device found at this location. This is useful for overriding default bindings. The format for the location is: DDDD:BB:DD.F. That is Domain:Bus:Device.Function and is the same as found in /sys/bus/pci/devices/. For example:

```
# echo 0000:00:19.0 > /sys/bus/pci/drivers/foo/bind
```

(Note: kernels before 2.6.28 may require echo -n).

/sys/bus/pci/drivers/.../new_id
/sys/devices/pciX/.../new_id

Defined on file *sysfs-bus-pci*

Writing a device ID to this file will attempt to dynamically add a new device ID to a PCI device driver. This may allow the driver to support more hardware than was included in the driver's static device ID support table at compile time. The format for the device ID is: VVVV DDDD SVVV SDDD CCCC MMMM PPPP. That is Vendor ID, Device ID, Subsystem Vendor ID, Subsystem Device ID, Class, Class Mask, and Private Driver Data. The Vendor ID and Device ID fields are required, the rest are optional. Upon successfully adding an ID, the driver will probe for the device and attempt to bind to it. For example:

```
# echo "8086 10f5" > /sys/bus/pci/drivers/foo/new_id
```

/sys/bus/pci/drivers/.../remove_id
/sys/devices/pciX/.../remove_id

Defined on file *sysfs-bus-pci*

Writing a device ID to this file will remove an ID that was dynamically added via the new_id sysfs entry. The format for the device ID is: VVVV DDDD SVVV SDDD CCCC MMMM. That is Vendor ID, Device ID, Subsystem Vendor ID, Subsystem Device ID, Class, and Class Mask. The Vendor ID and Device ID fields are required, the rest are optional. After successfully removing an ID, the driver will no longer support the device. This is useful to ensure auto probing won't match the driver to the device. For example:

```
# echo "8086 10f5" > /sys/bus/pci/drivers/foo/remove_id
```

/sys/bus/pci/drivers/.../unbind
/sys/devices/pciX/.../unbind

Defined on file *sysfs-bus-pci*

Writing a device location to this file will cause the driver to attempt to unbind from the device found at this location. This may be useful when overriding default bindings. The format for the location is: DDDD:BB:DD.F. That is Domain:Bus:Device.Function and is the same as found in /sys/bus/pci/devices/. For example:

```
# echo 0000:00:19.0 > /sys/bus/pci/drivers/foo/unbind
```

(Note: kernels before 2.6.28 may require echo -n).

/sys/bus/pci/drivers/altera-cvp/chkcfg

Defined on file *sysfs-driver-altera-cvp*

Contains either 1 or 0 and controls if configuration error checking in altera-cvp driver is turned on or off.

/sys/bus/pci/drivers/ehci_hcd/.../companion

Defined on file *sysfs-bus-pci-drivers-ehci_hcd*

PCI-based EHCI USB controllers (i.e., high-speed USB-2.0 controllers) are often implemented along with a set of ”companion” full/low-speed USB-1.1 controllers. When a high-speed device is plugged in, the connection is routed to the EHCI controller; when a full- or low-speed device is plugged in, the connection is routed to the companion controller.

Sometimes you want to force a high-speed device to connect at full speed, which can be accomplished by forcing the connection to be routed to the companion controller. That's what this file does. Writing a port number to the file causes connections on that port to be routed to the companion controller, and writing the negative of a port number returns the port to normal operation.

For example: To force the high-speed device attached to port 4 on bus 2 to run at full speed:

```
echo 4 >/sys/bus/usb/devices/usb2/../companion
```

To return the port to high-speed operation:

```
echo -4 >/sys/bus/usb/devices/usb2/../companion
```

Reading the file gives the list of ports currently forced to the companion controller.

Note: Some EHCI controllers do not have companions; they may contain an internal ”transaction translator” or they may be attached directly to a ”rate-matching hub”. This mechanism will not work with such controllers. Also, it cannot be used to force a port on a high-speed hub to connect at full speed.

Note: When this file was first added, it appeared in a different sysfs directory. The location given above is correct for 2.6.35 (and probably several earlier kernel versions as well).

/sys/bus/pci/drivers/i915/.../hwmon/hwmon<i>/curr1_crit

Defined on file *sysfs-driver-intel-i915-hwmon*

RW. Card reactive critical (I1) power limit in milliamperes.

Card reactive critical (I1) power limit in milliamperes is exposed for server products. The power controller will throttle the operating frequency if the power averaged over a window exceeds this limit.

Only supported for particular Intel i915 graphics platforms.

/sys/bus/pci/drivers/i915/.../hwmon/hwmon<i>/energy1_input

Defined on file *sysfs-driver-intel-i915-hwmon*

RO. Energy input of device or gt in microjoules.

For i915 device level hwmon devices (name ”i915”) this reflects energy input for the entire device. For gt level hwmon devices (name ”i915_gtN”) this reflects energy input for the gt.

Only supported for particular Intel i915 graphics platforms.

/sys/bus/pci/drivers/i915/.../hwmon/hwmon<i>/in0_input

Defined on file *sysfs-driver-intel-i915-hwmon*

RO. Current Voltage in millivolt.

Only supported for particular Intel i915 graphics platforms.

/sys/bus/pci/drivers/i915/.../hwmon/hwmon<i>/power1_crit

Defined on file *sysfs-driver-intel-i915-hwmon*

RW. Card reactive critical (I1) power limit in microwatts.

Card reactive critical (I1) power limit in microwatts is exposed for client products. The power controller will throttle the operating frequency if the power averaged over a window exceeds this limit.

Only supported for particular Intel i915 graphics platforms.

/sys/bus/pci/drivers/i915/.../hwmon/hwmon<i>/power1_max

Defined on file *sysfs-driver-intel-i915-hwmon*

RW. Card reactive sustained (PL1/Tau) power limit in microwatts.

The power controller will throttle the operating frequency if the power averaged over a window (typically seconds) exceeds this limit. A read value of 0 means that the PL1 power limit is disabled, writing 0 disables the limit. Writing values > 0 will enable the power limit.

Only supported for particular Intel i915 graphics platforms.

/sys/bus/pci/drivers/i915/.../hwmon/hwmon<i>/power1_max_interval

Defined on file *sysfs-driver-intel-i915-hwmon*

RW. Sustained power limit interval (Tau in PL1/Tau) in milliseconds over which sustained power is averaged.

Only supported for particular Intel i915 graphics platforms.

/sys/bus/pci/drivers/i915/.../hwmon/hwmon<i>/power1_rated_max

Defined on file *sysfs-driver-intel-i915-hwmon*

RO. Card default power limit (default TDP setting).

Only supported for particular Intel i915 graphics platforms.

/sys/bus/pci/drivers/janz-cmodio/.../modulbus_number

Defined on file *sysfs-bus-pci-drivers-janz-cmodio*

Value representing the HEX switch S2 of the janz carrier board CMOD-IO or CAN-PCI2

Read-only: value of the configuration switch (0..15)

/sys/bus/pci/drivers/pciback/allow_interrupt_control

Defined on file *sysfs-driver-pciback*

List of devices which can have interrupt control flag (INTx, MSI, MSI-X) set by a connected guest. It is meant to be set only when the guest is a stubdomain hosting device model (qemu) and the actual device is assigned to a HVM. It is not safe (similar to permissive attribute) to set for a devices assigned to a PV guest. The device is automatically removed from this list when the connected pcifront terminates.

/sys/bus/pci/drivers/pciback/quirks

Defined on file *sysfs-driver-pciback*

If the permissive attribute is set, then writing a string in the format of DDDD:BB:DD.F-REG:SIZE:MASK will allow the guest to write and read from the PCI device. That is Domain:Bus:Device.Function-Register:Size:Mask (Domain is optional). For example:

```
#echo 00:19.0-E0:2:FF > /sys/bus/pci/drivers/pciback/quirks
```

will allow the guest to read and write to the configuration register 0x0E.

/sys/bus/pci/drivers/xe/.../hwmon/hwmon<i>/curr1_crit

Defined on file *sysfs-driver-intel-xe-hwmon*

RW. Card reactive critical (I1) power limit in milliamperes.

Card reactive critical (I1) power limit in milliamperes is exposed for server products. The power controller will throttle the operating frequency if the power averaged over a window exceeds this limit.

/sys/bus/pci/drivers/xe/.../hwmon/hwmon<i>/energy1_input

Defined on file *sysfs-driver-intel-xe-hwmon*

RO. Energy input of device in microjoules.

Only supported for particular Intel xe graphics platforms.

/sys/bus/pci/drivers/xe/.../hwmon/hwmon<i>/in0_input

Defined on file *sysfs-driver-intel-xe-hwmon*

RO. Current Voltage in millivolt.

Only supported for particular Intel xe graphics platforms.

/sys/bus/pci/drivers/xe/.../hwmon/hwmon<i>/power1_crit

Defined on file *sysfs-driver-intel-xe-hwmon*

RW. Card reactive critical (I1) power limit in microwatts.

Card reactive critical (I1) power limit in microwatts is exposed for client products. The power controller will throttle the operating frequency if the power averaged over a window exceeds this limit.

Only supported for particular Intel xe graphics platforms.

/sys/bus/pci/drivers/xe/.../hwmon/hwmon<i>/power1_max

Defined on file *sysfs-driver-intel-xe-hwmon*

RW. Card reactive sustained (PL1) power limit in microwatts.

The power controller will throttle the operating frequency if the power averaged over a window (typically seconds) exceeds this limit. A read value of 0 means that the PL1 power limit is disabled, writing 0 disables the limit. Writing values > 0 and <= TDP will enable the power limit.

Only supported for particular Intel xe graphics platforms.

/sys/bus/pci/drivers/xe/.../hwmon/hwmon<i>/power1_max_interval

Defined on file *sysfs-driver-intel-xe-hwmon*

RW. Sustained power limit interval (Tau in PL1/Tau) in milliseconds over which sustained power is averaged.

Only supported for particular Intel xe graphics platforms.

/sys/bus/pci/drivers/xe/.../hwmon/hwmon<i>/power1_rated_max

Defined on file *sysfs-driver-intel-xe-hwmon*

RO. Card default power limit (default TDP setting).

Only supported for particular Intel xe graphics platforms.

/sys/bus/pci/drivers/xhci_hcd/.../dbc

Defined on file *sysfs-bus-pci-drivers-xhci_hcd*

xHCI compatible USB host controllers (i.e. super-speed USB3 controllers) are often implemented with the Debug Capability (DbC). It can present a debug device which is fully compliant with the USB framework and provides the equivalent of a very high performance full-duplex serial link for debug purpose.

The DbC debug device shares a root port with xHCI host. When the DbC is enabled, the root port will be assigned to the Debug Capability. Otherwise, it will be assigned to xHCI.

Writing "enable" to this attribute will enable the DbC functionality and the shared root port will be assigned to the DbC device. Writing "disable" to this attribute will disable the DbC functionality and the shared root port will roll back to the xHCI.

Reading this attribute gives the state of the DbC. It can be one of the following states: disabled, enabled, initialized, connected, configured and stalled.

/sys/bus/pci/drivers/xhci_hcd/.../dbc_bInterfaceProtocol

Defined on file *sysfs-bus-pci-drivers-xhci_hcd*

This attribute lets us change the bInterfaceProtocol field presented in the USB Interface descriptor by the xhci debug device. Value can only be changed while debug capability (DbC) is in disabled state to prevent USB descriptor change while connected to a USB host. The default value is 1 (GNU Remote Debug command). Other permissible value is 0 which is for vendor defined debug target.

/sys/bus/pci/drivers/xhci_hcd/.../dbc_bcdDevice

Defined on file *sysfs-bus-pci-drivers-xhci_hcd*

This dbc_bcdDevice attribute lets us change the bcdDevice field presented in the USB device descriptor by this xhci debug device. Value can only be changed while debug capability (DbC) is in disabled state to prevent USB device descriptor change while connected to a USB host. The default value is 0x0010. (device rev 0.10) It can be any 16-bit integer.

/sys/bus/pci/drivers/xhci_hcd/.../dbc_idProduct

Defined on file *sysfs-bus-pci-drivers-xhci_hcd*

This dbc_idProduct attribute lets us change the idProduct field presented in the USB device descriptor by this xhci debug device. Value can only be changed while debug capability (DbC) is in disabled state to prevent USB device descriptor change while connected to a USB host. The default value is 0x0010. It can be any 16-bit integer.

/sys/bus/pci/drivers/xhci_hcd/.../dbc_idVendor

Defined on file *sysfs-bus-pci-drivers-xhci_hcd*

This dbc_idVendor attribute lets us change the idVendor field presented in the USB device descriptor by this xhci debug device. Value can only be changed while debug capability (DbC) is in disabled state to prevent USB device descriptor change while connected to a USB host. The default value is 0x1d6b (Linux Foundation). It can be any 16-bit integer.

/sys/bus/pci/rescan

Defined on file *sysfs-bus-pci*

Writing a non-zero value to this attribute will force a rescan of all PCI buses in the system, and re-discover previously removed devices.

/sys/bus/pci/slots/.../module

Defined on file *sysfs-bus-pci*

This symbolic link points to the PCI hotplug controller driver module that manages the hotplug slot.

/sys/bus/peci/devices/<controller_id>-<device_addr>/remove

Defined on file *sysfs-bus-peci*

Writing a non-zero value to this attribute will remove the PECI device and any of its children.

/sys/bus/peci/rescan

Defined on file *sysfs-bus-peci*

Writing a non-zero value to this attribute will initiate scan for PECI devices on all PECI controllers in the system.

/sys/bus/platform/devices/*/frequency

Defined on file *sysfs-platform-brcmstb-memc*

DDR PHY frequency in Hz.

/sys/bus/platform/devices/*/srpd

Defined on file *sysfs-platform-brcmstb-memc*

Self Refresh Power Down (SRPD) inactivity timeout counted in internal DDR controller clock cycles. Possible values range from 0 (disable inactivity timeout) to 65535 (0xffff).

/sys/bus/platform/devices/*twl4030-usb/vbus

Defined on file *sysfs-platform-twl4030-usb*

Read-only status reporting if VBUS (approx 5V) is being supplied by the USB bus.

Possible values: ”on”, ”off”.

Changes are notified via select/poll.

/sys/bus/platform/devices/.../driver_override

Defined on file *sysfs-bus-platform*

This file allows the driver for a device to be specified which will override standard OF, ACPI, ID table, and name matching. When specified, only a driver with a name matching the value written to driver_override will have an opportunity to bind to the device. The override is specified by writing a string to the driver_override file (echo vfio-platform > driver_override) and may be cleared with an empty string (echo > driver_override). This returns the device to standard matching rules binding. Writing to driver_override does not automatically unbind the device from its current driver or make any attempt to automatically load the specified driver. If no driver with a matching name is currently loaded in the kernel, the device will not bind to any driver. This also allows devices to opt-out of driver binding using a driver_override name such as ”none”. Only a single driver may be specified in the override, there is no support for parsing delimiters.

/sys/bus/platform/devices/.../modalias

Defined on file *sysfs-bus-platform*

Same as MODALIAS in the uevent at device creation.

A platform device that it is exposed via devicetree uses:

- of:N`of node name`T`type`

Other platform devices use, instead:

- platform:*driver name*

/sys/bus/platform/devices/.../msi_irqs/

Defined on file *sysfs-bus-platform*

The /sys/devices/.../msi_irqs directory contains a variable set of files, with each file being named after a corresponding msi irq vector allocated to that device.

/sys/bus/platform/devices/.../msi_irqs/<N>

Defined on file *sysfs-bus-platform*

This attribute will show "msi" if <N> is a valid msi irq

/sys/bus/platform/devices/.../numa_node

Defined on file *sysfs-bus-platform*

This file contains the NUMA node to which the platform device is attached. It won't be visible if the node is unknown. The value comes from an ACPI _PXM method or a similar firmware source. Initial users for this file would be devices like arm smmu which are populated by arm64 acpi_iort.

/sys/bus/platform/devices/../cfam_reset

Defined on file *sysfs-bus-fsi*

Provides a means of resetting the cfam that is attached to the FSI device.

/sys/bus/platform/devices/../fsi-master/fsi0/break

Defined on file *sysfs-bus-fsi*

Sends an FSI BREAK command on a master's communication link to any connected slaves. A BREAK resets connected device's logic and preps it to receive further commands from the master.

/sys/bus/platform/devices/../fsi-master/fsi0/rescan

Defined on file *sysfs-bus-fsi*

Initiates a FSI master scan for all connected slave devices on its links.

/sys/bus/platform/devices/../fsi-master/fsi0/slave@00:00/raw

Defined on file *sysfs-bus-fsi*

Provides a means of reading/writing a 32 bit value from/to a specified FSI bus address.

/sys/bus/platform/devices/../fsi-master/fsi0/slave@00:00/term

Defined on file *sysfs-bus-fsi*

Sends an FSI terminate command from the master to its connected slave. A terminate resets the slave's state machines that control access to the internally connected engines. In addition the slave freezes its internal error register for debugging purposes. This command is also needed to abort any ongoing operation in case of an expired 'Master Time Out' timer.

/sys/bus/platform/devices/<dev>/always_powered_in_suspend

Defined on file *sysfs-bus-platform-onboard-usb-hub*

(RW) Controls whether the USB hub remains always powered during system suspend or not.

/sys/bus/platform/devices/ACPI000E:00/ac_alarm

Defined on file *sysfs-devices-platform-ACPI-TAD*

(RW) The AC alarm timer value.

Reads return the current AC alarm timer value in seconds or ”disabled”, if the AC alarm is not set to wake up the system.

Write a new AC alarm timer value in seconds or ”disabled” to it to set the AC alarm timer or to disable it, respectively.

If the AC alarm timer is set through this attribute and it expires, it will immediately wake up the system from the S3 sleep state (and from S4/S5 too if supported) until its status is explicitly cleared via the ac_status attribute.

/sys/bus/platform/devices/ACPI000E:00/ac_policy

Defined on file *sysfs-devices-platform-ACPI-TAD*

(RW) The AC alarm expired timer wake policy (see ACPI 6.2, Section 9.18 for details).

Reads return the current expired timer wake delay for the AC alarm timer or ”never”, if the policy is to discard AC timer wakeups if the system is on DC power.

Write a new expired timer wake delay for the AC alarm timer in seconds or ”never” to it to set the expired timer wake delay for the AC alarm timer or to set its expired wake policy to discard wakeups if the system is on DC power, respectively.

/sys/bus/platform/devices/ACPI000E:00/ac_status

Defined on file *sysfs-devices-platform-ACPI-TAD*

(RW) The AC alarm status.

Reads return a hexadecimal bitmask representing the AC alarm timer status with the following meaning of bits (see ACPI 6.2, Section 9.18.5):

Bit(0):	The timer has expired if set.
Bit(1):	The timer has woken up the system from a sleep state (S3 or S4/S5 if supported) if set.

The other bits are reserved.

Reads also cause the AC alarm timer status to be reset.

Another way to reset the status of the AC alarm timer is to write (the number) 0 to this file.

If the status return value indicates that the timer has expired, it will immediately wake up the system from the S3 sleep state (and from S4/S5 too if supported) until its status is explicitly cleared through this attribute.

/sys/bus/platform/devices/ACPI000E:00/caps

Defined on file *sysfs-devices-platform-ACPI-TAD*

(RO) Hexadecimal bitmask of the TAD attributes are reported by the platform firmware (see ACPI 6.2, section 9.18.2):

BIT(0):	AC wakeup implemented if set
BIT(1):	DC wakeup implemented if set
BIT(2):	Get/set real time features implemented if set
BIT(3):	Real time accuracy in milliseconds if set
BIT(4):	Correct status reported for wakeups from S4/S5 if set
BIT(5):	The AC timer wakes up from S4 if set
BIT(6):	The AC timer wakes up from S5 if set
BIT(7):	The DC timer wakes up from S4 if set
BIT(8):	The DC timer wakes up from S5 if set

The other bits are reserved.

/sys/bus/platform/devices/ACPI000E:00/dc_alarm

Defined on file *sysfs-devices-platform-ACPI-TAD*

(RW,optional) The DC alarm timer value.

This attribute is only present if the TAD supports a separate DC timer.

It is analogous to the ac_alarm attribute.

/sys/bus/platform/devices/ACPI000E:00/dc_policy

Defined on file *sysfs-devices-platform-ACPI-TAD*

(RW,optional) The DC alarm expired timer wake policy.

This attribute is only present if the TAD supports a separate DC timer.

It is analogous to the ac_policy attribute.

/sys/bus/platform/devices/ACPI000E:00/dc_status

Defined on file *sysfs-devices-platform-ACPI-TAD*

(RW,optional) The DC alarm status.

This attribute is only present if the TAD supports a separate DC timer.

It is analogous to the ac_status attribute.

/sys/bus/platform/devices/GGL0001:*/BINF.2

Defined on file *sysfs-driver-chromeos-acpi*

Returns active EC firmware of current boot (boolean).

0	Read only (recovery) firmware.
1	Rewritable firmware.

/sys/bus/platform/devices/GGL0001:*/BINF.3

Defined on file *sysfs-driver-chromeos-acpi*

Returns main firmware type for current boot (integer).

0	Recovery.
1	Normal.
2	Developer.
3	Netboot (factory installation only).

/sys/bus/platform/devices/GGL0001:*/CHSW

Defined on file *sysfs-driver-chromeos-acpi*

Returns switch position for Chrome OS specific hardware switches when the firmware is booted (integer).

0	No changes.
2	Recovery button was pressed.
4	Recovery button was pressed (EC firmware).
32	Developer switch was enabled.
512	Firmware write protection was disabled.

/sys/bus/platform/devices/GGL0001:*/FMAP

Defined on file *sysfs-driver-chromeos-acpi*

Returns physical memory address of the start of the main processor firmware flashmap.

/sys/bus/platform/devices/GGL0001:*/FRID

Defined on file *sysfs-driver-chromeos-acpi*

Returns firmware version for the read-only portion of the main processor firmware.

/sys/bus/platform/devices/GGL0001:*/FWID

Defined on file *sysfs-driver-chromeos-acpi*

Returns firmware version for the rewritable portion of the main processor firmware.

/sys/bus/platform/devices/GGL0001:*/GPIO.X/GPIO.0

Defined on file *sysfs-driver-chromeos-acpi*

Returns type of the GPIO signal for the Chrome OS specific GPIO assignments (integer).

1	Recovery button.
2	Developer mode switch.
3	Firmware write protection switch.
256 to 511	Debug header GPIO 0 to GPIO 255.

/sys/bus/platform/devices/GGL0001:*/GPIO.X/GPIO.1

Defined on file *sysfs-driver-chromeos-acpi*

Returns signal attributes of the GPIO signal (integer bitfield).

0	Signal is active low.
1	Signal is active high.

/sys/bus/platform/devices/GGL0001:*/GPIO.X/GPIO.2

Defined on file *sysfs-driver-chromeos-acpi*

Returns the GPIO number on the specified GPIO controller.

/sys/bus/platform/devices/GGL0001:*/GPIO.X/GPIO.3

Defined on file *sysfs-driver-chromeos-acpi*

Returns name of the GPIO controller.

/sys/bus/platform/devices/GGL0001:*/HWID

Defined on file *sysfs-driver-chromeos-acpi*

Returns hardware ID for the Chromebook.

/sys/bus/platform/devices/GGL0001:*/MECK

Defined on file *sysfs-driver-chromeos-acpi*

Returns the SHA-1 or SHA-256 hash that is read out of the Management Engine extended registers during boot. The hash is exported via ACPI so the OS can verify that the Management Engine firmware has not changed. If Management Engine is not present, or if the firmware was unable to read the extended registers, this buffer size can be zero.

/sys/bus/platform/devices/GGL0001:*/VBNV.0

Defined on file *sysfs-driver-chromeos-acpi*

Returns offset in CMOS bank 0 of the verified boot non-volatile storage block, counting from the first writable CMOS byte (that is, 'offset = 0' is the byte following the 14 bytes of clock data).

/sys/bus/platform/devices/GGL0001:*/VBNV.1

Defined on file *sysfs-driver-chromeos-acpi*

Return the size in bytes of the verified boot non-volatile storage block.

/sys/bus/platform/devices/GGL0001:*/VDAT

Defined on file *sysfs-driver-chromeos-acpi*

Returns the verified boot data block shared between the firmware verification step and the kernel verification step (hex dump).

/sys/bus/platform/devices/GOOG000C\:00/boot_on_ac

Defined on file *sysfs-platform-wilco-ec*

Boot on AC is a policy which makes the device boot from S5 when AC power is connected. This is useful for users who want to run their device headless or with a dock.

Input should be parseable by kstrtou8() to 0 or 1.

/sys/bus/platform/devices/GOOG000C\:00/build_date

Defined on file *sysfs-platform-wilco-ec*

Display Wilco Embedded Controller firmware build date. Output will a MM/DD/YY string.

/sys/bus/platform/devices/GOOG000C\:00/build_revision

Defined on file *sysfs-platform-wilco-ec*

Display Wilco Embedded Controller build revision. Output will a version string be similar to the example below: d2592cae0

/sys/bus/platform/devices/GOOG000C\:00/model_number

Defined on file *sysfs-platform-wilco-ec*

Display Wilco Embedded Controller model number. Output will a version string be similar to the example below: 08B6

/sys/bus/platform/devices/GOOG000C\:00/usb_charge

Defined on file *sysfs-platform-wilco-ec*

Control the USB PowerShare Policy. USB PowerShare is a policy which affects charging via the special USB PowerShare port (marked with a small lightning bolt or battery icon) when in low power states:

- In S0, the port will always provide power.
- In S0ix, if usb_charge is enabled, then power will be supplied to the port when on AC or if battery is > 50%. Else no power is supplied.
- In S5, if usb_charge is enabled, then power will be supplied to the port when on AC. Else no power is supplied.

Input should be either "0" or "1".

/sys/bus/platform/devices/GOOG000C\:00/version

Defined on file *sysfs-platform-wilco-ec*

Display Wilco Embedded Controller firmware version. The format of the string is x.y.z. Where x is major, y is minor and z is the build number. For example: 95.00.06

/sys/bus/platform/devices/INT3407:00/dptf_power/adapter_rating_mw

Defined on file *sysfs-platform-dptf*

(RO) Adapter rating in milliwatts (the maximum Adapter power). Must be 0 if no AC Adaptor is plugged in.

/sys/bus/platform/devices/INT3407:00/dptf_power/battery_steady_power

Defined on file *sysfs-platform-dptf*

(RO) The maximum sustained power for battery in milliwatts.

/sys/bus/platform/devices/INT3407:00/dptf_power/charger_type

Defined on file *sysfs-platform-dptf*

(RO) The charger type - Traditional, Hybrid or NVDC.

/sys/bus/platform/devices/INT3407:00/dptf_power/max_platform_power_mw

Defined on file *sysfs-platform-dptf*

(RO) Maximum platform power that can be supported by the battery in milliwatts.

/sys/bus/platform/devices/INT3407:00/dptf_power/platform_power_source

Defined on file *sysfs-platform-dptf*

(RO) Display the platform power source

bits[3:0]	Current power source - 0x00 = DC - 0x01 = AC - 0x02 = USB - 0x03 = Wireless Charger
bits[7:4]	Power source sequence number

/sys/bus/platform/devices/INT3407:00/dptf_power/prochot_confirm

Defined on file *sysfs-platform-dptf*

(WO) Confirm embedded controller about a prochot notification.

/sys/bus/platform/devices/INT3407:00/dptf_power/rest_of_platform_power_mw

Defined on file *sysfs-platform-dptf*

(RO) Shows the rest (outside of SoC) of worst-case platform power.

/sys/bus/platform/devices/INT3532:00/dptf_battery/current_discharge_capbility_ma

Defined on file *sysfs-platform-dptf*

(RO) The battery discharge current capability obtained from battery fuel gauge in milli Amps.

/sys/bus/platform/devices/INT3532:00/dptf_battery/high_freq_impedance_mohm

Defined on file *sysfs-platform-dptf*

(RO) The high frequency impedance value that can be obtained from battery fuel gauge in milli Ohms.

/sys/bus/platform/devices/INT3532:00/dptf_battery/max_platform_power_mw

Defined on file *sysfs-platform-dptf*

(RO) The maximum platform power that can be supported by the battery in milli watts.

/sys/bus/platform/devices/INT3532:00/dptf_battery/max_steady_state_power_mw

Defined on file *sysfs-platform-dptf*

(RO) The maximum sustained power for battery in milli watts.

/sys/bus/platform/devices/INT3532:00/dptf_battery/no_load_voltage_mv

Defined on file *sysfs-platform-dptf*

(RO) The no-load voltage that can be obtained from battery fuel gauge in milli volts.

/sys/bus/platform/devices/INTC1045:00/pch_fivr_switch_frequency/fivr_switching_fault_stat

Defined on file *sysfs-platform-dptf*

(RO) Read the FIVR switching frequency control fault status.

/sys/bus/platform/devices/INTC1045:00/pch_fivr_switch_frequency/fivr_switching_freq_mhz

Defined on file *sysfs-platform-dptf*

(RO) Get the FIVR switching control frequency in MHz.

/sys/bus/platform/devices/INTC1045:00/pch_fivr_switch_frequency/freq_mhz_high_clock

Defined on file *sysfs-platform-dptf*

(RW) The PCH FIVR (Fully Integrated Voltage Regulator) switching frequency in MHz, when FIVR clock is 38.4MHz.

/sys/bus/platform/devices/INTC1045:00/pch_fivr_switch_frequency/freq_mhz_low_clock

Defined on file *sysfs-platform-dptf*

(RW) The PCH FIVR (Fully Integrated Voltage Regulator) switching frequency in MHz, when FIVR clock is 19.2MHz or 24MHz.

/sys/bus/platform/devices/INTC1045:00/pch_fivr_switch_frequency/ssc_clock_info

Defined on file *sysfs-platform-dptf*

(RO) Presents SSC (spread spectrum clock) information for EMI (Electro magnetic interference) control. This is a bit mask.

Bits	Description
[7:0]	Sets clock spectrum spread percentage: 0x00=0.2% , 0x3F=10% 1 LSB = 0.1% increase in spread (for settings 0x01 thru 0x1C) 1 LSB = 0.2% increase in spread (for settings 0x1E thru 0x3F)
[8]	When set to 1, enables spread spectrum clock
[9]	0: Triangle mode. FFC frequency walks around the Fcenter in a linear fashion 1: Random walk mode. FFC frequency changes randomly within the SSC (Spread spectrum clock) range
[10]	0: No white noise. 1: Add white noise to spread waveform
[11]	When 1, future writes are ignored.

/sys/bus/platform/devices/INTC1092:00/intc_data

Defined on file *sysfs-driver-intc_sar*

This sysfs entry is used to retrieve Dynamic SAR information emitted/maintained by a BIOS that supports Dynamic SAR.

The retrieved information is in the order given below: - device_mode - bandtable_index - antennatable_index - sartable_index

The above information is sent as integer values separated by a single space. This information can then be pushed to a WWAN modem that uses this to control the transmit signal level using the Band/Antenna/SAR table index information. These parameters are derived/decided by aggregating device-mode like laptop/tablet/clamshell etc. and the proximity-sensor data available to the embedded controller on given host. The regulatory mode configured on Dynamic SAR driver also influences these values.

The userspace applications can poll for changes to this file using POLLPRI event on file-descriptor (fd) obtained by opening this sysfs entry. Application can then read this information from the sysfs node and consume the given information.

/sys/bus/platform/devices/INTC1092:00/intc_reg

Defined on file *sysfs-driver-intc_sar*

Specific Absorption Rate (SAR) regulatory mode is typically derived based on information like mcc (Mobile Country Code) and mnc (Mobile Network Code) that is available for the currently attached LTE network. A userspace application is required to set the current SAR regulatory mode on the Dynamic SAR driver using this sysfs node. Such an application can also read back using this sysfs node, the currently configured regulatory mode value from the Dynamic SAR driver.

Acceptable regulatory modes are:

0	FCC
1	CE
2	ISED

- The regulatory mode value has one of the above values.
- The default regulatory mode used in the driver is 0.

/sys/bus/platform/devices/MLNXBF04:00/bootfifo

Defined on file *sysfs-platform-mellanox-bootctl*

The file used to access the BlueField boot fifo.

/sys/bus/platform/devices/MLNXBF04:00/lifecycle_state

Defined on file *sysfs-platform-mellanox-bootctl*

The Life-cycle state of the SoC, which could be one of the following values.

Production	Production state and can be updated to secure
GA Secured	Secure chip and not able to change state
GA Non-Secured	Non-Secure chip and not able to change state
RMA	Return Merchandise Authorization

/sys/bus/platform/devices/MLNXBF04:00/mfg_lock

Defined on file *sysfs-platform-mellanox-bootctl*

The ”mfg_lock” sysfs attribute is write-only. A successful write to this attribute will latch the board-level attributes into EEPROM, making them read-only.

/sys/bus/platform/devices/MLNXBF04:00/modl

Defined on file *sysfs-platform-mellanox-bootctl*

The ”modl” sysfs attribute holds the board's model number. This value is provided on a board-level label.

/sys/bus/platform/devices/MLNXBF04:00/oob_mac

Defined on file *sysfs-platform-mellanox-bootctl*

The ”oob_mac” sysfs attribute holds the MAC address for the out-of-band 1Gbps Ethernet port. This MAC address is provided on a board-level label.

/sys/bus/platform/devices/MLNXBF04:00/opn

Defined on file *sysfs-platform-mellanox-bootctl*

The ”opn” sysfs attribute holds the board's part number. This value is provided on a board-level label.

/sys/bus/platform/devices/MLNXBF04:00/post_reset_wdog

Defined on file *sysfs-platform-mellanox-bootctl*

The watchdog setting in seconds for the next booting. It's used to reboot the chip and recover it to the old state if the new boot partition fails.

/sys/bus/platform/devices/MLNXBF04:00/reset_action

Defined on file *sysfs-platform-mellanox-bootctl*

The source of the boot stream for the next reset. It could be one of the following values:

external	boot from external source (USB or PCIe)
emmc	boot from the onchip eMMC
emmc_legacy	boot from the onchip eMMC in legacy (slow) mode

/sys/bus/platform/devices/MLNXBF04:00/rev

Defined on file *sysfs-platform-mellanox-bootctl*

The ”rev” sysfs attribute holds the board's revision. This value is provided on a board-level label.

/sys/bus/platform/devices/MLNXBF04:00/rsh_log

Defined on file *sysfs-platform-mellanox-bootctl*

The file used to write BlueField boot log with the format ”[INFO|WARN|ERR|ASSERT]<msg>”. Log level 'INFO' is used by default if not specified.

/sys/bus/platform/devices/MLNXBF04:00/second_reset_action

Defined on file *sysfs-platform-mellanox-bootctl*

Update the source of the boot stream after next reset. It could be one of the following values and will be applied after next reset.

external	boot from external source (USB or PCIe)
emmc	boot from the onchip eMMC
emmc_legacy	boot from the onchip eMMC in legacy (slow) mode
swap_emmc	swap the primary / secondary boot partition
none	cancel the action

/sys/bus/platform/devices/MLNXBF04:00/secure_boot_fuse_state

Defined on file *sysfs-platform-mellanox-bootctl*

The state of eFuse versions with the following values.

InUse	burnt, valid and currently in use
Used	burnt and valid
Free	not burnt and free to use
Skipped	not burnt but not free (skipped)
Wasted	burnt and invalid
Invalid	not burnt but marked as valid (error state).

/sys/bus/platform/devices/MLNXBF04:00/sku

Defined on file *sysfs-platform-mellanox-bootctl*

The ”sku” sysfs attribute holds the board's SKU number. This value is provided on a board-level label.

/sys/bus/platform/devices/MLNXBF04:00/sn

Defined on file *sysfs-platform-mellanox-bootctl*

The ”sn” sysfs attribute holds the board's serial number. This value is provided on a board-level label.

/sys/bus/platform/devices/MLNXBF04:00/uuid

Defined on file *sysfs-platform-mellanox-bootctl*

The ”uuid” sysfs attribute holds the board's UUID. This value is provided by the manufacturing team.

/sys/bus/platform/devices/VPC2004:*/camera_power

Defined on file *sysfs-platform-ideapad-laptop*

Control the power of camera module. 1 means on, 0 means off.

/sys/bus/platform/devices/VPC2004:*/conservation_mode

Defined on file *sysfs-platform-ideapad-laptop*

Controls whether the conservation mode is enabled or not. This feature limits the maximum battery charge percentage to around 50-60% in order to prolong the lifetime of the battery.

/sys/bus/platform/devices/VPC2004:*/fan_mode

Defined on file *sysfs-platform-ideapad-laptop*

Change fan mode There are four available modes:

- 0 -> Super Silent Mode
- 1 -> Standard Mode
- 2 -> Dust Cleaning
- 4 -> Efficient Thermal Dissipation Mode

/sys/bus/platform/devices/VPC2004:*/fn_lock

Defined on file *sysfs-platform-ideapad-laptop*

Control fn-lock mode.

- 1 -> Switched On
- 0 -> Switched Off

For example:

```
# echo "0" >  \
/sys/bus/pci/devices/0000:00:1f.0/PNP0C09:00/VPC2004:00/fn_lock
```

/sys/bus/platform/devices/VPC2004:*/touchpad

Defined on file *sysfs-platform-ideapad-laptop*

Control touchpad mode.

- 1 -> Switched On
- 0 -> Switched Off

/sys/bus/platform/devices/VPC2004:*/usb_charging

Defined on file *sysfs-platform-ideapad-laptop*

Controls whether the "always on USB charging" feature is enabled or not. This feature enables charging USB devices even if the computer is not turned on.

/sys/bus/platform/devices/[..]/fsi-master-gpio/external_mode

Defined on file *sysfs-driver-fsi-master-gpio*

Controls access arbitration for GPIO-based FSI master. A value of 0 (the default) sets normal mode, where the driver performs FSI bus transactions, 1 sets external mode, where the FSI bus is driven externally (for example, by a debug device).

/sys/bus/platform/devices/ci_hdrc.0/inputs/a_bus_drop

Defined on file *sysfs-platform-chipidea-usb-otg*

Can be set and read The a_bus_drop(A-device bus drop) input is 1 when the application running on the A-device wants to power down the bus, and is 0 otherwise, When a_bus_drop is 1, then the a_bus_req shall be 0.

Valid values are "1" and "0".

Reading: returns 1 if the bus is off(vbus is turned off) by
A-device, otherwise 0.

/sys/bus/platform/devices/ci_hdrc.0/inputs/a_bus_req

Defined on file *sysfs-platform-chipidea-usb-otg*

Can be set and read. Set a_bus_req(A-device bus request) input to be 1 if the application running on the A-device wants to use the bus, and to be 0 when the application no longer wants to use the bus(or wants to work as peripheral). a_bus_req can also be set to 1 by kernel in response to remote wakeup signaling from the B-device, the A-device should decide to resume the bus.

Valid values are "1" and "0".

Reading: returns 1 if the application running on the A-device is using the bus as host role, otherwise 0.

/sys/bus/platform/devices/ci_hdrc.0/inputs/a_clr_err

Defined on file *sysfs-platform-chipidea-usb-otg*

Only can be set. The a_clr_err(A-device Vbus error clear) input is used to clear vbus error, then A-device will power down the bus.

Valid value is "1"

/sys/bus/platform/devices/ci_hdrc.0/inputs/b_bus_req

Defined on file *sysfs-platform-chipidea-usb-otg*

Can be set and read. The b_bus_req(B-device bus request) input is 1 during the time that the application running on the B-device wants to use the bus as host, and is 0 when the application no longer wants to work as host and decides to switch back to be peripheral.

Valid values are "1" and "0".

Reading: returns if the application running on the B device is using the bus as host role, otherwise 0.

/sys/bus/platform/devices/ci_hdrc.0/role

Defined on file *sysfs-platform-chipidea-usb2*

When read, it returns string "gadget" or "host", indicating the current controller role.

It will do role switch when "gadget" or "host" is written to it. Only controller at dual-role configuration supports writing.

/sys/bus/platform/devices/dfl-fme.0/bitstream_id

Defined on file *sysfs-platform-dfl-fme*

Read-only. It returns Bitstream (static FPGA region) identifier number, which includes the detailed version and other information of this static FPGA region.

/sys/bus/platform/devices/dfl-fme.0/bitstream_metadata

Defined on file *sysfs-platform-dfl-fme*

Read-only. It returns Bitstream (static FPGA region) meta data, which includes the synthesis date, seed and other information of this static FPGA region.

/sys/bus/platform/devices/dfl-fme.0/cache_size

Defined on file *sysfs-platform-dfl-fme*

Read-only. It returns cache size of this FPGA device.

/sys/bus/platform/devices/dfl-fme.0/errors/catfatal_errors

Defined on file *sysfs-platform-dfl-fme*

Read-only. It returns catastrophic and fatal errors detected.

/sys/bus/platform/devices/dfl-fme.0/errors/first_error

Defined on file *sysfs-platform-dfl-fme*

Read-only. Read this file to get the first error detected by hardware.

/sys/bus/platform/devices/dfl-fme.0/errors/fme_errors

Defined on file *sysfs-platform-dfl-fme*

Read-Write. Read this file to get errors detected on FME. Write this file to clear errors logged in fme_errors. Write fails with -EINVAL if input parsing fails or input error code doesn't match.

/sys/bus/platform/devices/dfl-fme.0/errors/inject_errors

Defined on file *sysfs-platform-dfl-fme*

Read-Write. Read this file to check errors injected. Write this file to inject errors for testing purpose. Write fails with -EINVAL if input parsing fails or input inject error code isn't supported.

/sys/bus/platform/devices/dfl-fme.0/errors/next_error

Defined on file *sysfs-platform-dfl-fme*

Read-only. Read this file to get the second error detected by hardware.

/sys/bus/platform/devices/dfl-fme.0/errors/nonfatal_errors

Defined on file *sysfs-platform-dfl-fme*

Read-only. It returns non-fatal errors detected.

/sys/bus/platform/devices/dfl-fme.0/errors/pcie0_errors

Defined on file *sysfs-platform-dfl-fme*

Read-Write. Read this file for errors detected on pcie0 link. Write this file to clear errors logged in pcie0_errors. Write fails with -EINVAL if input parsing fails or input error code doesn't match.

/sys/bus/platform/devices/dfl-fme.0/errors/pcie1_errors

Defined on file *sysfs-platform-dfl-fme*

Read-Write. Read this file for errors detected on pcie1 link. Write this file to clear errors logged in pcie1_errors. Write fails with -EINVAL if input parsing fails or input error code doesn't match.

/sys/bus/platform/devices/dfl-fme.0/fabric_version

Defined on file *sysfs-platform-dfl-fme*

Read-only. It returns fabric version of this FPGA device. Userspace applications need this information to select best data channels per different fabric design.

/sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/name

Defined on file *sysfs-platform-dfl-fme*

Read-Only. Read this file to get the name of hwmon device, it supports values:

'dfl_fme_thermal'	thermal hwmon device name
'dfl_fme_power'	power hwmon device name

/sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/power1_crit

Defined on file *sysfs-platform-dfl-fme*

Read-Write. Read this file to get current hardware power threshold2 in uW. If power consumption rises at or above this threshold, hardware starts 90% throttling. Write this file to set current hardware power threshold2 in uW. As hardware only accepts values in Watts, so input value will be round down per Watts (< 1 watts part will be discarded) and clamped within the range from 0 to 127 Watts. Write fails with -EINVAL if input parsing fails.

/sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/power1_crit_alarm

Defined on file *sysfs-platform-dfl-fme*

Read-only. It returns 1 if power consumption is currently at or above hardware threshold2 (see 'power1_crit'), otherwise 0.

/sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/power1_fpga_limit

Defined on file *sysfs-platform-dfl-fme*

Read-Only. It returns power limit for FPGA in uW.

/sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/power1_input

Defined on file *sysfs-platform-dfl-fme*

Read-Only. It returns current FPGA power consumption in uW.

/sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/power1_ltr

Defined on file *sysfs-platform-dfl-fme*

Read-only. Read this file to get current Latency Tolerance Reporting (ltr) value. It returns 1 if all Accelerated Function Units (AFUs) can tolerate latency >= 40us for memory access or 0 if any AFU is latency sensitive (< 40us).

/sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/power1_max

Defined on file *sysfs-platform-dfl-fme*

Read-Write. Read this file to get current hardware power threshold1 in uW. If power consumption rises at or above this threshold, hardware starts 50% throttling. Write this file to set current hardware power threshold1 in uW. As hardware only accepts values in Watts, so input value will be round down per Watts (< 1 watts part will be discarded) and clamped within the range from 0 to 127 Watts. Write fails with -EINVAL if input parsing fails.

/sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/power1_max_alarm

Defined on file *sysfs-platform-dfl-fme*

Read-only. It returns 1 if power consumption is currently at or above hardware threshold1 (see 'power1_max'), otherwise 0.

/sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/power1_xeon_limit

Defined on file *sysfs-platform-dfl-fme*

Read-Only. It returns power limit for XEON in uW.

/sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/temp1_crit

Defined on file *sysfs-platform-dfl-fme*

Read-Only. It returns hardware threshold2 temperature in millidegrees Celsius. If temperature rises at or above this threshold, hardware starts 100% throttling.

/sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/temp1_crit_alarm

Defined on file *sysfs-platform-dfl-fme*

Read-only. It returns 1 if temperature is currently at or above hardware threshold2 (see 'temp1_crit'), otherwise 0.

/sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/temp1_emergency

Defined on file *sysfs-platform-dfl-fme*

Read-Only. It returns hardware trip threshold temperature in millidegrees Celsius. If temperature rises at or above this threshold, a fatal event will be triggered to board management controller (BMC) to shutdown FPGA.

/sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/temp1_input

Defined on file *sysfs-platform-dfl-fme*

Read-Only. It returns FPGA device temperature in millidegrees Celsius.

/sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/temp1_max

Defined on file *sysfs-platform-dfl-fme*

Read-Only. It returns hardware threshold1 temperature in millidegrees Celsius. If temperature rises at or above this threshold, hardware starts 50% or 90% throttling (see 'temp1_max_policy').

/sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/temp1_max_alarm

Defined on file *sysfs-platform-dfl-fme*

Read-only. It returns 1 if temperature is currently at or above hardware threshold1 (see 'temp1_max'), otherwise 0.

/sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/temp1_max_policy

Defined on file *sysfs-platform-dfl-fme*

Read-Only. Read this file to get the policy of hardware threshold1 (see 'temp1_max'). It only supports two values (policies):

0	AP2 state (90% throttling)
1	AP1 state (50% throttling)

/sys/bus/platform/devices/dfl-fme.0/ports_num

Defined on file *sysfs-platform-dfl-fme*

Read-only. One DFL FPGA device may have more than 1 port/Accelerator Function Unit (AFU). It returns the number of ports on the FPGA device when read it.

/sys/bus/platform/devices/dfl-fme.0/socket_id

Defined on file *sysfs-platform-dfl-fme*

Read-only. It returns socket_id to indicate which socket this FPGA belongs to, only valid for integrated solution. User only needs this information, in case standard numa node can't provide correct information.

/sys/bus/platform/devices/dfl-port.0/afu_id

Defined on file *sysfs-platform-dfl-port*

Read-only. User can program different PR bitstreams to FPGA Accelerator Function Unit (AFU) for different functions. It returns uuid which could be used to identify which PR bitstream is programmed in this AFU.

/sys/bus/platform/devices/dfl-port.0/ap1_event

Defined on file *sysfs-platform-dfl-port*

Read-write. Read this file for AP1 (AFU Power State 1) event. It's used to indicate transient AP1 state. Write 1 to this file to clear AP1 event.

/sys/bus/platform/devices/dfl-port.0/ap2_event

Defined on file *sysfs-platform-dfl-port*

Read-write. Read this file for AP2 (AFU Power State 2) event. It's used to indicate transient AP2 state. Write 1 to this file to clear AP2 event.

/sys/bus/platform/devices/dfl-port.0/errors/errors

Defined on file *sysfs-platform-dfl-port*

Read-Write. Read this file to get errors detected on port and Accelerated Function Unit (AFU). Write error code to this file to clear errors. Write fails with -EINVAL if input parsing fails or input error code doesn't match. Write fails with -EBUSY or -ETIMEDOUT if error can't be cleared as hardware in low power state (-EBUSY) or not respoding (-ETIMEDOUT).

/sys/bus/platform/devices/dfl-port.0/errors/first_error

Defined on file *sysfs-platform-dfl-port*

Read-only. Read this file to get the first error detected by hardware.

/sys/bus/platform/devices/dfl-port.0/errors/first_malformed_req

Defined on file *sysfs-platform-dfl-port*

Read-only. Read this file to get the first malformed request captured by hardware.

/sys/bus/platform/devices/dfl-port.0/id

Defined on file *sysfs-platform-dfl-port*

Read-only. It returns id of this port. One DFL FPGA device may have more than one port. Userspace could use this id to distinguish different ports under same FPGA device.

/sys/bus/platform/devices/dfl-port.0/ltr

Defined on file *sysfs-platform-dfl-port*

Read-write. Read or set AFU latency tolerance reporting value. Set ltr to 1 if the AFU can tolerate latency >= 40us or set it to 0 if it is latency sensitive.

/sys/bus/platform/devices/dfl-port.0/power_state

Defined on file *sysfs-platform-dfl-port*

Read-only. It reports the APx (AFU Power) state, different APx means different throttling level. When reading this file, it returns ”0” - Normal / ”1” - AP1 / ”2” - AP2 / ”6” - AP6.

/sys/bus/platform/devices/dfl-port.0/userclk_freqcmd

Defined on file *sysfs-platform-dfl-port*

Write-only. User writes command to this interface to set userclock to AFU.

/sys/bus/platform/devices/dfl-port.0/userclk_freqcntrcmd

Defined on file *sysfs-platform-dfl-port*

Write-only. User writes command to this interface to set userclock counter.

/sys/bus/platform/devices/dfl-port.0/userclk_freqcntrsts

Defined on file *sysfs-platform-dfl-port*

Read-only. Read this file to get the status of issued command to userclck_freqcntrcmd.

/sys/bus/platform/devices/dfl-port.0/userclk_freqsts

Defined on file *sysfs-platform-dfl-port*

Read-only. Read this file to get the status of issued command to userclck_freqcmd.

/sys/bus/platform/devices/occ-hwmon.X/ffdc

Defined on file *sysfs-bus-platform-devices-occ-hwmon*

Contains the First Failure Data Capture from the SBEFIFO hardware, if there is any from a previous transfer. Otherwise, the file is empty. The data is cleared when it's been completely read by a user. As the name suggests, only the data from the first error is saved, until it's cleared upon read. The OCC hwmon driver, running on a Baseboard Management Controller (BMC), communicates with POWER9 and up processors over the Self-Boot Engine (SBE) FIFO. In many error conditions, the SBEFIFO will return error data indicating the type of error and system state, etc.

/sys/bus/platform/devices/smpro-errmon.*/[error|warn]_[smpro|pmpro]

Defined on file *sysfs-bus-platform-devices-ampere-smpro*

(RO) Contains the internal firmware error/warning printed as hex format.

The detail of each sysfs entries is as below:

Error	Sysfs entry	Description
SMpro error	/sys/bus/platform/devices/smpro-errmon.*/error_smpro	system has SMpro error
SMpro warning	/sys/bus/platform/devices/smpro-errmon.*/warn_smpro	system has SMpro warning
PMpro error	/sys/bus/platform/devices/smpro-errmon.*/error_pmpro	system has PMpro error
PMpro warning	/sys/bus/platform/devices/smpro-errmon.*/warn_pmpro	system has PMpro warning

For details, see section *5.10 RAS Internal Error Register Definitions, Altra Family Soc BMC Interface Specification*.

/sys/bus/platform/devices/smpro-errmon.*/error_[core|mem|pcie|other]_[ce|ue]

Defined on file *sysfs-bus-platform-devices-ampere-smpro*

(RO) Contains the 48-byte Ampere (Vendor-Specific) Error Record printed in hex format according to the table below:

Offset	Field	Size (byte)	Description
00	Error Type	1	See *the table below* for details
01	Subtype	1	See *the table below* for details
02	Instance	2	See *the table below* for details
04	Error status	4	See ARM RAS specification for details
08	Error Address	8	See ARM RAS specification for details
16	Error Misc 0	8	See ARM RAS specification for details
24	Error Misc 1	8	See ARM RAS specification for details
32	Error Misc 2	8	See ARM RAS specification for details
40	Error Misc 3	8	See ARM RAS specification for details

The table below defines the value of error types, their subtype, subcomponent and instance:

Error Group	Error Type	Sub type	Sub component	Instance
CPM (core)	0	0	Snoop-Logic	CPM #
CPM (core)	0	2	Armv8 Core 1	CPM #
MCU (mem)	1	1	ERR1	MCU # | SLOT << 11
MCU (mem)	1	2	ERR2	MCU # | SLOT << 11
MCU (mem)	1	3	ERR3	MCU #
MCU (mem)	1	4	ERR4	MCU #
MCU (mem)	1	5	ERR5	MCU #
MCU (mem)	1	6	ERR6	MCU #
MCU (mem)	1	7	Link Error	MCU #
Mesh (other)	2	0	Cross Point	X | (Y << 5) | NS <<11
Mesh (other)	2	1	Home Node(IO)	X | (Y << 5) | NS <<11
Mesh (other)	2	2	Home Node(Mem)	X | (Y << 5) | NS <<11 | device<<
Mesh (other)	2	4	CCIX Node	X | (Y << 5) | NS <<11
2P Link (other)	3	0	N/A	Altra 2P Link #
GIC (other)	5	0	ERR0	0
GIC (other)	5	1	ERR1	0
GIC (other)	5	2	ERR2	0
GIC (other)	5	3	ERR3	0
GIC (other)	5	4	ERR4	0
GIC (other)	5	5	ERR5	0
GIC (other)	5	6	ERR6	0
GIC (other)	5	7	ERR7	0
GIC (other)	5	8	ERR8	0
GIC (other)	5	9	ERR9	0

continues on next pa

Table 10 – continued from previous page

GIC (other)	5	10	ERR10	0
GIC (other)	5	11	ERR11	0
GIC (other)	5	12	ERR12	0
GIC (other)	5	13-21	ERR13	RC # + 1
SMMU (other)	6	TCU	100	RC #
SMMU (other)	6	TBU0	0	RC #
SMMU (other)	6	TBU1	1	RC #
SMMU (other)	6	TBU2	2	RC #
SMMU (other)	6	TBU3	3	RC #
SMMU (other)	6	TBU4	4	RC #
SMMU (other)	6	TBU5	5	RC #
SMMU (other)	6	TBU6	6	RC #
SMMU (other)	6	TBU7	7	RC #
SMMU (other)	6	TBU8	8	RC #
SMMU (other)	6	TBU9	9	RC #
PCIe AER (pcie)	7	Root	0	RC #
PCIe AER (pcie)	7	Device	1	RC #
PCIe RC (pcie)	8	RCA HB	0	RC #
PCIe RC (pcie)	8	RCB HB	1	RC #
PCIe RC (pcie)	8	RASDP	8	RC #
OCM (other)	9	ERR0	0	0
OCM (other)	9	ERR1	1	0
OCM (other)	9	ERR2	2	0
SMpro (other)	10	ERR0	0	0
SMpro (other)	10	ERR1	1	0
SMpro (other)	10	MPA_ERR	2	0
PMpro (other)	11	ERR0	0	0
PMpro (other)	11	ERR1	1	0
PMpro (other)	11	MPA_ERR	2	0

Example:

```
# cat error_other_ue
880807001e0040104010401015000000010040104010401 00c0000000000000000000000000000000000
```

The detail of each sysfs entries is as below:

Error	Sysfs entry	Description (when triggered)
Core's CE	/sys/bus/platform/devices/smpro-errmon.*/error_core_ce	Core has CE error
Core's UE	/sys/bus/platform/devices/smpro-errmon.*/error_core_ue	Core has UE error
Memory's CE	/sys/bus/platform/devices/smpro-errmon.*/error_mem_ce	Memory has CE error
Memory's UE	/sys/bus/platform/devices/smpro-errmon.*/error_mem_ue	Memory has UE error
PCIe's CE	/sys/bus/platform/devices/smpro-errmon.*/error_pcie_ce	any PCIe controller has CE error
PCIe's UE	/sys/bus/platform/devices/smpro-errmon.*/error_pcie_ue	any PCIe controller has UE error
Other's CE	/sys/bus/platform/devices/smpro-errmon.*/error_other_ce	any other CE error
Other's UE	/sys/bus/platform/devices/smpro-errmon.*/error_other_ue	any other UE error

UE: Uncorrect-able Error CE: Correct-able Error

For details, see section *3.3 Ampere (Vendor-Specific) Error Record Formats, Altra Family RAS Supplement*.

/sys/bus/platform/devices/smpro-errmon.*/event_[vrd_warn_fault|vrd_hot|dimm_hot|dimm_

Defined on file *sysfs-bus-platform-devices-ampere-smpro*

(RO) Contains the detail information in case of VRD/DIMM warning/hot events in hex format as below:

```
AAAA
```

where:

- `AAAA`: The event detail information data

The detail of each sysfs entries is as below:

Event	Sysfs entry	Description
VRD HOT	/sys/bus/platform/devices/smpro-errmon.*/event_vrd_hot	VRD Hot
VR Warn/Fault	/sys/bus/platform/devices/smpro-errmon.*/event_vrd_warn_fault	VR Warning or Fault
DIMM HOT	/sys/bus/platform/devices/smpro-errmon.*/event_dimm_hot	DIMM Hot
DIMM 2X REFRESH RATE	/sys/bus/platform/devices/smpro-errmon.*/event_dimm_2x_refresh	DIMM 2x refresh rate event in high temp

For more details, see section *5.7 GPI Status Registers and 5.9 Memory Error Register Definitions, Altra Family Soc BMC Interface Specification*.

/sys/bus/platform/devices/smpro-errmon.*/event_dimm[0-15]_syndrome

Defined on file *sysfs-bus-platform-devices-ampere-smpro*

(RO) The sysfs returns the 2-byte DIMM failure syndrome data for slot 0-15 if it failed to initialize.

For more details, see section *5.11 Boot Stage Register Definitions, Altra Family Soc BMC Interface Specification*.

/sys/bus/platform/devices/smpro-errmon.*/overflow_[core|mem|pcie|other]_[ce|ue]

Defined on file *sysfs-bus-platform-devices-ampere-smpro*

(RO) Return the overflow status of each type HW error reported:

- 0 : No overflow
- 1 : There is an overflow and the oldest HW errors are dropped

The detail of each sysfs entries is as below:

Overflow	Sysfs entry	Description
Core's CE	/sys/bus/platform/devices/smpro-errmon.*/overflow_core_ce	Core CE error overflow
Core's UE	/sys/bus/platform/devices/smpro-errmon.*/overflow_core_ue	Core UE error overflow
Memory's CE	/sys/bus/platform/devices/smpro-errmon.*/overflow_mem_ce	Memory CE error overflow
Memory's UE	/sys/bus/platform/devices/smpro-errmon.*/overflow_mem_ue	Memory UE error overflow
PCIe's CE	/sys/bus/platform/devices/smpro-errmon.*/overflow_pcie_ce	any PCIe controller CE error overflow
PCIe's UE	/sys/bus/platform/devices/smpro-errmon.*/overflow_pcie_ue	any PCIe controller UE error overflow
Other's CE	/sys/bus/platform/devices/smpro-errmon.*/overflow_other_ce	any other CE error overflow
Other's UE	/sys/bus/platform/devices/smpro-errmon.*/overflow_other_ue	other UE error overflow

where:

- UE: Uncorrect-able Error
- CE: Correct-able Error

/sys/bus/platform/devices/smpro-misc*/soc_power_limit

Defined on file *sysfs-bus-platform-devices-ampere-smpro*

(RW) Contains the desired SoC power limit in Watt. Writes to this sysfs set the desired SoC power limit (W). Reads from this register return the current SoC power limit (W). The value ranges:

- Minimum: 120 W
- Maximum: Socket TDP power

/sys/bus/platform/devices/smpro-misc.*/boot_progress

Defined on file *sysfs-bus-platform-devices-ampere-smpro*

(RO) Contains the boot stages information in hex as format below:

```
AABBCCCCCCCC
```

where:

- `AA` : The boot stages
  - 00: SMpro firmware booting
  - 01: PMpro firmware booting
  - 02: ATF BL1 firmware booting
  - 03: DDR initialization
  - 04: DDR training report status
  - 05: ATF BL2 firmware booting
  - 06: ATF BL31 firmware booting
  - 07: ATF BL32 firmware booting
  - 08: UEFI firmware booting
  - 09: OS booting
- `BB` : Boot status
  - 00: Not started
  - 01: Started
  - 02: Completed without error
  - 03: Failed.
- `CCCCCCCC`: Boot status information defined for each boot stages

For details, see section *5.11 Boot Stage Register Definitions* and section *6. Processor Boot Progress Codes, Altra Family Soc BMC Interface Specification*.

/sys/bus/platform/devices/tahvo-usb/otg_mode

Defined on file *sysfs-platform-tahvo-usb*

Set or read the current OTG mode. Valid values are ”host” and ”peripheral”.

Reading: returns the current mode.

/sys/bus/platform/devices/tahvo-usb/vbus

Defined on file *sysfs-platform-tahvo-usb*

Read the current VBUS state.

Reading: returns "on" or "off".

/sys/bus/platform/drivers/amd_pmc/*/smu_fw_version

Defined on file *sysfs-amd-pmc*

Reading this file reports the version of the firmware loaded to System Management Unit (SMU) contained in AMD CPUs and APUs.

/sys/bus/platform/drivers/amd_pmc/*/smu_program

Defined on file *sysfs-amd-pmc*

Reading this file reports the program corresponding to the SMU firmware version. The program field is used to disambiguate two APU/CPU models that can share the same firmware binary.

/sys/bus/platform/drivers/aspeed-uart-routing/*/io*

Defined on file *sysfs-driver-aspeed-uart-routing*

Selects the RX source of IOx serial port. The current selection will be marked by brackets "[]".

Users: OpenBMC. Proposed changes should be mailed to openbmc@lists.ozlabs.org

/sys/bus/platform/drivers/aspeed-uart-routing/*/uart*

Defined on file *sysfs-driver-aspeed-uart-routing*

Selects the RX source of the UARTx device.

When read, each file shows the list of available options with currently selected option marked by brackets "[]". The list of available options depends on the selected file.

e.g. cat /sys/bus/platform/drivers/aspeed-uart-routing/*.uart_routing/uart1 [io1] io2 io3 io4 uart2 uart3 uart4 io6

In this case, UART1 gets its input from IO1 (physical serial port 1).

Users: OpenBMC. Proposed changes should be mailed to openbmc@lists.ozlabs.org

/sys/bus/platform/drivers/intel-m10bmc-sec-update/.../security/bmc_canceled_csks

Defined on file *sysfs-driver-intel-m10-bmc-sec-update*

Read only. Returns a list of indices for canceled code signing keys for the BMC. The standard bitmap list format is used (e.g. "1,2-6,9").

/sys/bus/platform/drivers/intel-m10bmc-sec-update/.../security/bmc_root_entry_hash

Defined on file *sysfs-driver-intel-m10-bmc-sec-update*

Read only. Returns the root entry hash for the BMC image if one is programmed, else it returns the string: "hash not programmed". This file is only visible if the underlying device supports it. Format: string.

/sys/bus/platform/drivers/intel-m10bmc-sec-update/.../security/flash_count

Defined on file *sysfs-driver-intel-m10-bmc-sec-update*

Read only. Returns number of times the secure update staging area has been flashed. Format: "%u".

/sys/bus/platform/drivers/intel-m10bmc-sec-update/.../security/pr_canceled_csks

Defined on file *sysfs-driver-intel-m10-bmc-sec-update*

Read only. Returns a list of indices for canceled code signing keys for the partial reconfiguration region. The standard bitmap list format is used (e.g. "1,2-6,9").

/sys/bus/platform/drivers/intel-m10bmc-sec-update/.../security/pr_root_entry_hash

Defined on file *sysfs-driver-intel-m10-bmc-sec-update*

Read only. Returns the root entry hash for the partial reconfiguration region if one is programmed, else it returns the string: "hash not programmed". This file is only visible if the underlying device supports it. Format: string.

/sys/bus/platform/drivers/intel-m10bmc-sec-update/.../security/sr_canceled_csks

Defined on file *sysfs-driver-intel-m10-bmc-sec-update*

Read only. Returns a list of indices for canceled code signing keys for the static region. The standard bitmap list format is used (e.g. "1,2-6,9").

/sys/bus/platform/drivers/intel-m10bmc-sec-update/.../security/sr_root_entry_hash

Defined on file *sysfs-driver-intel-m10-bmc-sec-update*

Read only. Returns the root entry hash for the static region if one is programmed, else it returns the string: "hash not programmed". This file is only visible if the underlying device supports it. Format: string.

/sys/bus/platform/drivers/qcom_eud/.../enable

Defined on file *sysfs-driver-eud*

The Enable/Disable sysfs interface for Embedded USB Debugger(EUD). This enables and disables the EUD based on a 1 or a 0 value. By enabling EUD, the user is able to activate the mini-usb hub of EUD for debug and trace capabilities.

/sys/bus/platform/drivers/ufshcd/*/attributes/active_icc_level
/sys/bus/platform/devices/*.ufs/attributes/active_icc_level

Defined on file *sysfs-driver-ufs*

This file provides the active icc level UFS device attribute. The full information about the attribute could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/attributes/bkops_status
/sys/bus/platform/devices/*.ufs/attributes/bkops_status

Defined on file *sysfs-driver-ufs*

This file provides the background operations status UFS device attribute. The full information about the attribute could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/attributes/boot_lun_enabled
/sys/bus/platform/devices/*.ufs/attributes/boot_lun_enabled

Defined on file *sysfs-driver-ufs*

This file provides the boot lun enabled UFS device attribute. The full information about the attribute could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/attributes/configuration_descriptor_lock
/sys/bus/platform/devices/*.ufs/attributes/configuration_descriptor_lock

Defined on file *sysfs-driver-ufs*

This file shows whether the configuration descriptor is locked. The full information about the attribute could be found at UFS specifications 2.1. The file is read only.

/sys/bus/platform/drivers/ufshcd/*/attributes/current_power_mode
/sys/bus/platform/devices/*.ufs/attributes/current_power_mode

Defined on file *sysfs-driver-ufs*

This file provides the current power mode UFS device attribute. The full information about the attribute could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/attributes/exception_event_control
/sys/bus/platform/devices/*.ufs/attributes/exception_event_control

Defined on file *sysfs-driver-ufs*

This file provides the exception event control UFS device attribute. The full information about the attribute could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/attributes/exception_event_status
/sys/bus/platform/devices/*.ufs/attributes/exception_event_status

Defined on file *sysfs-driver-ufs*

This file provides the exception event status UFS device attribute. The full information about the attribute could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/attributes/ffu_status
/sys/bus/platform/devices/*.ufs/attributes/ffu_status

Defined on file *sysfs-driver-ufs*

This file provides the ffu status UFS device attribute. The full information about the attribute could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/attributes/max_data_in_size
/sys/bus/platform/devices/*.ufs/attributes/max_data_in_size

Defined on file *sysfs-driver-ufs*

This file shows the maximum data size in a DATA IN UPIU. The full information about the attribute could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/attributes/max_data_out_size
/sys/bus/platform/devices/*.ufs/attributes/max_data_out_size

Defined on file *sysfs-driver-ufs*

This file shows the maximum number of bytes that can be requested with a READY TO TRANSFER UPIU. The full information about the attribute could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/attributes/max_number_of_rtt
/sys/bus/platform/devices/*.ufs/attributes/max_number_of_rtt

Defined on file *sysfs-driver-ufs*

This file provides the maximum current number of outstanding RTTs in device that is allowed. The full information about the attribute could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/attributes/ooo_data_enabled
/sys/bus/platform/devices/*.ufs/attributes/ooo_data_enabled

Defined on file *sysfs-driver-ufs*

This file provides the out of order data transfer enabled UFS device attribute. The full information about the attribute could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/attributes/psa_data_size
/sys/bus/platform/devices/*.ufs/attributes/psa_data_size

Defined on file *sysfs-driver-ufs*

This file shows the amount of data that the host plans to load to all logical units in pre-soldering state. The full information about the attribute could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/attributes/psa_state
/sys/bus/platform/devices/*.ufs/attributes/psa_state

Defined on file *sysfs-driver-ufs*

This file show the PSA feature status. The full information about the attribute could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/attributes/purge_status
/sys/bus/platform/devices/*.ufs/attributes/purge_status

Defined on file *sysfs-driver-ufs*

This file provides the purge operation status UFS device attribute. The full information about the attribute could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/attributes/reference_clock_frequency
/sys/bus/platform/devices/*.ufs/attributes/reference_clock_frequency

Defined on file *sysfs-driver-ufs*

This file provides the reference clock frequency UFS device attribute. The full information about the attribute could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/attributes/wb_avail_buf
/sys/bus/platform/devices/*.ufs/attributes/wb_avail_buf

Defined on file *sysfs-driver-ufs*

This entry shows the amount of unused WriteBooster buffer available.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/attributes/wb_cur_buf
/sys/bus/platform/devices/*.ufs/attributes/wb_cur_buf

Defined on file *sysfs-driver-ufs*

This entry shows the amount of unused current buffer.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/attributes/wb_flush_status
/sys/bus/platform/devices/*.ufs/attributes/wb_flush_status

Defined on file *sysfs-driver-ufs*

This entry shows the flush operation status.

00h	idle
01h	Flush operation in progress
02h	Flush operation stopped prematurely.
03h	Flush operation completed successfully
04h	Flush operation general failure

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/attributes/wb_life_time_est
/sys/bus/platform/devices/*.ufs/attributes/wb_life_time_est

Defined on file *sysfs-driver-ufs*

This entry shows an indication of the WriteBooster Buffer lifetime based on the amount of performed program/erase cycles

01h	0% - 10% WriteBooster Buffer life time used
...	
0Ah	90% - 100% WriteBooster Buffer life time used

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/capabilities/clock_scaling
/sys/bus/platform/devices/*.ufs/capabilities/clock_scaling

Defined on file *sysfs-driver-ufs*

Indicates status of clock scaling.

0	Clock scaling is not supported.
1	Clock scaling is supported.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/capabilities/write_booster
/sys/bus/platform/devices/*.ufs/capabilities/write_booster

Defined on file *sysfs-driver-ufs*

Indicates status of Write Booster.

0	Write Booster can not be enabled.
1	Write Booster can be enabled.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/capabilities/
/sys/bus/platform/devices/*.ufs/capabilities/

Defined on file *sysfs-driver-ufs*

The group represents the effective capabilities of the host-device pair. i.e. the capabilities which are enabled in the driver for the specific host controller, supported by the host controller and are supported and/or have compatible configuration on the device side.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/bkops_termination_latency
/sys/bus/platform/devices/*.ufs/device_descriptor/bkops_termination_latency

Defined on file *sysfs-driver-ufs*

This file shows the background operations termination latency. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/boot_enable
/sys/bus/platform/devices/*.ufs/device_descriptor/boot_enable

Defined on file *sysfs-driver-ufs*

This file shows value that indicates whether the device is enabled for boot. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/descriptor_access_enable
/sys/bus/platform/devices/*.ufs/device_descriptor/descriptor_access_enable

Defined on file *sysfs-driver-ufs*

This file shows value that indicates whether the device descriptor could be read after partial initialization phase of the boot sequence. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/device_class
/sys/bus/platform/devices/*.ufs/device_descriptor/device_class

Defined on file *sysfs-driver-ufs*

This file shows the device class. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/device_sub_class
/sys/bus/platform/devices/*.ufs/device_descriptor/device_sub_class

Defined on file *sysfs-driver-ufs*

This file shows the UFS storage subclass. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/device_type
/sys/bus/platform/devices/*.ufs/device_descriptor/device_type

Defined on file *sysfs-driver-ufs*

This file shows the device type. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/device_version
/sys/bus/platform/devices/*.ufs/device_descriptor/device_version

Defined on file *sysfs-driver-ufs*

This file shows the device version. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/ffu_timeout
/sys/bus/platform/devices/*.ufs/device_descriptor/ffu_timeout

Defined on file *sysfs-driver-ufs*

This file shows the FFU timeout. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/high_priority_lun
/sys/bus/platform/devices/*.ufs/device_descriptor/high_priority_lun

Defined on file *sysfs-driver-ufs*

This file shows the high priority lun. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/initial_active_icc_level
/sys/bus/platform/devices/*.ufs/device_descriptor/initial_active_icc_level

Defined on file *sysfs-driver-ufs*

This file shows the initial active ICC level. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/initial_power_mode
/sys/bus/platform/devices/*.ufs/device_descriptor/initial_power_mode

Defined on file *sysfs-driver-ufs*

This file shows value that defines the power mode after device initialization or hardware reset. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/manufacturer_id
/sys/bus/platform/devices/*.ufs/device_descriptor/manufacturer_id

Defined on file *sysfs-driver-ufs*

This file shows the manufacturer ID. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/manufacturing_date
/sys/bus/platform/devices/*.ufs/device_descriptor/manufacturing_date

Defined on file *sysfs-driver-ufs*

This file shows the manufacturing date in BCD format. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/number_of_luns
/sys/bus/platform/devices/*.ufs/device_descriptor/number_of_luns

Defined on file *sysfs-driver-ufs*

This file shows number of logical units. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/number_of_secure_wpa
/sys/bus/platform/devices/*.ufs/device_descriptor/number_of_secure_wpa

Defined on file *sysfs-driver-ufs*

This file shows number of secure write protect areas supported by the device. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/number_of_wluns
/sys/bus/platform/devices/*.ufs/device_descriptor/number_of_wluns

Defined on file *sysfs-driver-ufs*

This file shows number of well known logical units. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/protocol
/sys/bus/platform/devices/*.ufs/device_descriptor/protocol

Defined on file *sysfs-driver-ufs*

This file shows the protocol supported by an UFS device. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/psa_max_data_size
/sys/bus/platform/devices/*.ufs/device_descriptor/psa_max_data_size

Defined on file *sysfs-driver-ufs*

This file shows the maximum amount of data that may be written during the pre-soldering phase of the PSA flow. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/psa_state_timeout
/sys/bus/platform/devices/*.ufs/device_descriptor/psa_state_timeout

Defined on file *sysfs-driver-ufs*

This file shows the command maximum timeout for a change in PSA state. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/queue_depth
/sys/bus/platform/devices/*.ufs/device_descriptor/queue_depth

Defined on file *sysfs-driver-ufs*

This file shows the device queue depth. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/rtc_update
/sys/bus/platform/devices/*.ufs/device_descriptor/rtc_update

Defined on file *sysfs-driver-ufs*

This file shows the frequency and method of the realtime clock update. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/rtt_capability
/sys/bus/platform/devices/*.ufs/device_descriptor/rtt_capability

Defined on file *sysfs-driver-ufs*

This file shows the maximum number of outstanding RTTs supported by the device. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/secure_removal_type
/sys/bus/platform/devices/*.ufs/device_descriptor/secure_removal_type

Defined on file *sysfs-driver-ufs*

This file shows the secure removal type. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/specification_version
/sys/bus/platform/devices/*.ufs/device_descriptor/specification_version

Defined on file *sysfs-driver-ufs*

This file shows the specification version. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/support_security_lun
/sys/bus/platform/devices/*.ufs/device_descriptor/support_security_lun

Defined on file *sysfs-driver-ufs*

This file shows whether the security lun is supported. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/ufs_features
/sys/bus/platform/devices/*.ufs/device_descriptor/ufs_features

Defined on file *sysfs-driver-ufs*

This file shows which features are supported by the device. This is one of the UFS device descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/wb_presv_us_en
/sys/bus/platform/devices/*.ufs/device_descriptor/wb_presv_us_en

Defined on file *sysfs-driver-ufs*

This entry shows if preserve user-space was configured

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/wb_shared_alloc_units
/sys/bus/platform/devices/*.ufs/device_descriptor/wb_shared_alloc_units

Defined on file *sysfs-driver-ufs*

This entry shows the shared allocated units of WB buffer

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/device_descriptor/wb_type
/sys/bus/platform/devices/*.ufs/device_descriptor/wb_type

Defined on file *sysfs-driver-ufs*

This entry shows the configured WB type. 0x1 for shared buffer mode. 0x0 for dedicated buffer mode.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/enable_wb_buf_flush
/sys/bus/platform/devices/*.ufs/enable_wb_buf_flush

Defined on file *sysfs-driver-ufs*

This entry shows the status of WriteBooster buffer flushing and it can be used to enable or disable the flushing. If flushing is enabled, the device executes the flush operation when the command queue is empty.

/sys/bus/platform/drivers/ufshcd/*/flags/bkops_enable
/sys/bus/platform/devices/*.ufs/flags/bkops_enable

Defined on file *sysfs-driver-ufs*

This file shows whether the device background operations are enabled. The full information about the flag could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/flags/busy_rtc
/sys/bus/platform/devices/*.ufs/flags/busy_rtc

Defined on file *sysfs-driver-ufs*

This file shows whether the device is executing internal operation related to real time clock. The full information about the flag could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/flags/device_init
/sys/bus/platform/devices/*.ufs/flags/device_init

Defined on file *sysfs-driver-ufs*

This file shows the device init status. The full information about the flag could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/flags/disable_fw_update
/sys/bus/platform/devices/*.ufs/flags/disable_fw_update

Defined on file *sysfs-driver-ufs*

This file shows whether the device FW update is permanently disabled. The full information about the flag could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/flags/life_span_mode_enable
/sys/bus/platform/devices/*.ufs/flags/life_span_mode_enable

Defined on file *sysfs-driver-ufs*

This file shows whether the device life span mode is enabled. The full information about the flag could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/flags/permanent_wpe
/sys/bus/platform/devices/*.ufs/flags/permanent_wpe

Defined on file *sysfs-driver-ufs*

This file shows whether permanent write protection is enabled. The full information about the flag could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/flags/phy_resource_removal
/sys/bus/platform/devices/*.ufs/flags/phy_resource_removal

Defined on file *sysfs-driver-ufs*

This file shows whether physical resource removal is enable. The full information about the flag could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/flags/power_on_wpe
/sys/bus/platform/devices/*.ufs/flags/power_on_wpe

Defined on file *sysfs-driver-ufs*

This file shows whether write protection is enabled on all logical units configured as power on write protected. The full information about the flag could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/flags/wb_enable
/sys/bus/platform/devices/*.ufs/flags/wb_enable

Defined on file *sysfs-driver-ufs*

This entry shows the status of WriteBooster.

0	WriteBooster is not enabled.
1	WriteBooster is enabled

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/flags/wb_flush_during_h8
/sys/bus/platform/devices/*.ufs/flags/wb_flush_during_h8

Defined on file *sysfs-driver-ufs*

Flush WriteBooster Buffer during hibernate state.

0	Device is not allowed to flush the WriteBooster Buffer during link hibernate state.
1	Device is allowed to flush the WriteBooster Buffer during link hibernate state.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/flags/wb_flush_en
/sys/bus/platform/devices/*.ufs/flags/wb_flush_en

Defined on file *sysfs-driver-ufs*

This entry shows if flush is enabled.

0	Flush operation is not performed.
1	Flush operation is performed.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/*_memory_capacity_adjustment_fac
/sys/bus/platform/devices/*.ufs/geometry_descriptor/*_memory_capacity_adjustment_factor

Defined on file *sysfs-driver-ufs*

This file shows the memory capacity adjustment factor for different memory types (system code, non persistent, enhanced type 1-4). This is one of the UFS geometry descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/*_memory_max_alloc_units
/sys/bus/platform/devices/*.ufs/geometry_descriptor/*_memory_max_alloc_units

Defined on file *sysfs-driver-ufs*

This file shows the maximum number of allocation units for different memory types (system code, non persistent, enhanced type 1-4). This is one of the UFS geometry descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/allocation_unit_size
/sys/bus/platform/devices/*.ufs/geometry_descriptor/allocation_unit_size

Defined on file *sysfs-driver-ufs*

This file shows the allocation unit size. This is one of the UFS geometry descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/data_ordering
/sys/bus/platform/devices/*.ufs/geometry_descriptor/data_ordering

Defined on file *sysfs-driver-ufs*

This file shows support for out-of-order data transfer. This is one of the UFS geometry descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/dyn_capacity_resource_policy
/sys/bus/platform/devices/*.ufs/geometry_descriptor/dyn_capacity_resource_policy

Defined on file *sysfs-driver-ufs*

This file shows the dynamic capacity resource policy. This is one of the UFS geometry descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/max_in_buffer_size
/sys/bus/platform/devices/*.ufs/geometry_descriptor/max_in_buffer_size

Defined on file *sysfs-driver-ufs*

This file shows the maximum data-in buffer size. This is one of the UFS geometry descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/max_number_of_contexts
/sys/bus/platform/devices/*.ufs/geometry_descriptor/max_number_of_contexts

Defined on file *sysfs-driver-ufs*

This file shows maximum available number of contexts which are supported by the device. This is one of the UFS geometry descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/max_number_of_luns
/sys/bus/platform/devices/*.ufs/geometry_descriptor/max_number_of_luns

Defined on file *sysfs-driver-ufs*

This file shows the maximum number of logical units supported by the UFS device. This is one of the UFS geometry descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/max_out_buffer_size
/sys/bus/platform/devices/*.ufs/geometry_descriptor/max_out_buffer_size

Defined on file *sysfs-driver-ufs*

This file shows the maximum data-out buffer size. This is one of the UFS geometry descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/memory_types
/sys/bus/platform/devices/*.ufs/geometry_descriptor/memory_types

Defined on file *sysfs-driver-ufs*

This file shows supported memory types. This is one of the UFS geometry descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/min_addressable_block_size
/sys/bus/platform/devices/*.ufs/geometry_descriptor/min_addressable_block_size

Defined on file *sysfs-driver-ufs*

This file shows the minimum addressable block size. This is one of the UFS geometry descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/optimal_read_block_size
/sys/bus/platform/devices/*.ufs/geometry_descriptor/optimal_read_block_size

Defined on file *sysfs-driver-ufs*

This file shows the optimal read block size. This is one of the UFS geometry descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/optimal_write_block_size
/sys/bus/platform/devices/*.ufs/geometry_descriptor/optimal_write_block_size

Defined on file *sysfs-driver-ufs*

This file shows the optimal write block size. This is one of the UFS geometry descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/raw_device_capacity
/sys/bus/platform/devices/*.ufs/geometry_descriptor/raw_device_capacity

Defined on file *sysfs-driver-ufs*

This file shows the total memory quantity available to the user to configure the device logical units. This is one of the UFS geometry descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/rpmb_rw_size
/sys/bus/platform/devices/*.ufs/geometry_descriptor/rpmb_rw_size

Defined on file *sysfs-driver-ufs*

This file shows the maximum number of RPMB frames allowed in Security Protocol In/Out. This is one of the UFS geometry descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/secure_removal_types
/sys/bus/platform/devices/*.ufs/geometry_descriptor/secure_removal_types

Defined on file *sysfs-driver-ufs*

This file shows supported secure removal types. This is one of the UFS geometry descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/segment_size
/sys/bus/platform/devices/*.ufs/geometry_descriptor/segment_size

Defined on file *sysfs-driver-ufs*

This file shows the segment size. This is one of the UFS geometry descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/sys_data_tag_resource_size
/sys/bus/platform/devices/*.ufs/geometry_descriptor/sys_data_tag_resource_size

Defined on file *sysfs-driver-ufs*

This file shows maximum storage area size allocated by the device to handle system data by the tagging mechanism. This is one of the UFS geometry descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/sys_data_tag_unit_size
/sys/bus/platform/devices/*.ufs/geometry_descriptor/sys_data_tag_unit_size

Defined on file *sysfs-driver-ufs*

This file shows system data tag unit size. This is one of the UFS geometry descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/wb_buff_cap_adj
/sys/bus/platform/devices/*.ufs/geometry_descriptor/wb_buff_cap_adj

Defined on file *sysfs-driver-ufs*

This entry shows the total user-space decrease in shared buffer mode. The value of this parameter is 3 for TLC NAND when SLC mode is used as WriteBooster Buffer. 2 for MLC NAND.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/wb_max_alloc_units
/sys/bus/platform/devices/*.ufs/geometry_descriptor/wb_max_alloc_units

Defined on file *sysfs-driver-ufs*

This entry shows the Maximum total WriteBooster Buffer size which is supported by the entire device.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/wb_max_wb_luns
/sys/bus/platform/devices/*.ufs/geometry_descriptor/wb_max_wb_luns

Defined on file *sysfs-driver-ufs*

This entry shows the maximum number of luns that can support WriteBooster.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/wb_sup_red_type
/sys/bus/platform/devices/*.ufs/geometry_descriptor/wb_sup_red_type

Defined on file *sysfs-driver-ufs*

The supportability of user space reduction mode and preserve user space mode. 00h: WriteBooster Buffer can be configured only in user space reduction type. 01h: WriteBooster Buffer can be configured only in preserve user space type. 02h: Device can be configured in either user space reduction type or preserve user space type.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/wb_sup_wb_type
/sys/bus/platform/devices/*.ufs/geometry_descriptor/wb_sup_wb_type

Defined on file *sysfs-driver-ufs*

The supportability of WriteBooster Buffer type.

00h	LU based WriteBooster Buffer configuration
01h	Single shared WriteBooster Buffer configuration
02h	Supporting both LU based WriteBooster. Buffer and Single shared WriteBooster Buffer configuration

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/health_descriptor/eol_info
/sys/bus/platform/devices/*.ufs/health_descriptor/eol_info

Defined on file *sysfs-driver-ufs*

This file shows preend of life information. This is one of the UFS health descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/health_descriptor/life_time_estimation_a
/sys/bus/platform/devices/*.ufs/health_descriptor/life_time_estimation_a

Defined on file *sysfs-driver-ufs*

This file shows indication of the device life time (method a). This is one of the UFS health descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/health_descriptor/life_time_estimation_b
/sys/bus/platform/devices/*.ufs/health_descriptor/life_time_estimation_b

Defined on file *sysfs-driver-ufs*

This file shows indication of the device life time (method b). This is one of the UFS health descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/interconnect_descriptor/mphy_version
/sys/bus/platform/devices/*.ufs/interconnect_descriptor/mphy_version

Defined on file *sysfs-driver-ufs*

This file shows the MIPI M-PHY version number in BCD format. This is one of the UFS interconnect descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/interconnect_descriptor/unipro_version
/sys/bus/platform/devices/*.ufs/interconnect_descriptor/unipro_version

Defined on file *sysfs-driver-ufs*

This file shows the MIPI UniPro version number in BCD format. This is one of the UFS interconnect descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/monitor/monitor_chunk_size
/sys/bus/platform/devices/*.ufs/monitor/monitor_chunk_size

Defined on file *sysfs-driver-ufs*

This file tells the monitor to focus on requests transferring data of specific chunk size (in Bytes). 0 means any chunk size. It can only be changed when monitor is disabled.

/sys/bus/platform/drivers/ufshcd/*/monitor/monitor_enable
/sys/bus/platform/devices/*.ufs/monitor/monitor_enable

Defined on file *sysfs-driver-ufs*

This file shows the status of performance monitor enablement and it can be used to start/stop the monitor. When the monitor is stopped, the performance data collected is also cleared.

/sys/bus/platform/drivers/ufshcd/*/monitor/read_nr_requests
/sys/bus/platform/devices/*.ufs/monitor/read_nr_requests

Defined on file *sysfs-driver-ufs*

This file shows how many read requests have been sent after monitor gets started.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/monitor/read_req_latency_avg
/sys/bus/platform/devices/*.ufs/monitor/read_req_latency_avg

Defined on file *sysfs-driver-ufs*

This file shows the average latency (in micro seconds) of read requests after monitor gets started.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/monitor/read_req_latency_max
/sys/bus/platform/devices/*.ufs/monitor/read_req_latency_max

Defined on file *sysfs-driver-ufs*

This file shows the maximum latency (in micro seconds) of read requests after monitor gets started.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/monitor/read_req_latency_min
/sys/bus/platform/devices/*.ufs/monitor/read_req_latency_min

Defined on file *sysfs-driver-ufs*

This file shows the minimum latency (in micro seconds) of read requests after monitor gets started.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/monitor/read_req_latency_sum
/sys/bus/platform/devices/*.ufs/monitor/read_req_latency_sum

Defined on file *sysfs-driver-ufs*

This file shows the total latency (in micro seconds) of read requests sent after monitor gets started.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/monitor/read_total_busy
/sys/bus/platform/devices/*.ufs/monitor/read_total_busy

Defined on file *sysfs-driver-ufs*

This file shows how long (in micro seconds) has been spent sending data from device to host after monitor gets started.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/monitor/read_total_sectors
/sys/bus/platform/devices/*.ufs/monitor/read_total_sectors

Defined on file *sysfs-driver-ufs*

This file shows how many sectors (in 512 Bytes) have been sent from device to host after monitor gets started.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/monitor/write_nr_requests
/sys/bus/platform/devices/*.ufs/monitor/write_nr_requests

Defined on file *sysfs-driver-ufs*

This file shows how many write requests have been sent after monitor gets started.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/monitor/write_req_latency_avg
/sys/bus/platform/devices/*.ufs/monitor/write_req_latency_avg

Defined on file *sysfs-driver-ufs*

This file shows the average latency (in micro seconds) of write requests after monitor gets started.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/monitor/write_req_latency_max
/sys/bus/platform/devices/*.ufs/monitor/write_req_latency_max

Defined on file *sysfs-driver-ufs*

This file shows the maximum latency (in micro seconds) of write requests after monitor gets started.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/monitor/write_req_latency_min
/sys/bus/platform/devices/*.ufs/monitor/write_req_latency_min

Defined on file *sysfs-driver-ufs*

This file shows the minimum latency (in micro seconds) of write requests after monitor gets started.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/monitor/write_req_latency_sum
/sys/bus/platform/devices/*.ufs/monitor/write_req_latency_sum

Defined on file *sysfs-driver-ufs*

This file shows the total latency (in micro seconds) of write requests after monitor gets started.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/monitor/write_total_busy
/sys/bus/platform/devices/*.ufs/monitor/write_total_busy

Defined on file *sysfs-driver-ufs*

This file shows how long (in micro seconds) has been spent sending data from host to device after monitor gets started.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/monitor/write_total_sectors
/sys/bus/platform/devices/*.ufs/monitor/write_total_sectors

Defined on file *sysfs-driver-ufs*

This file shows how many sectors (in 512 Bytes) have been sent from host to device after monitor gets started.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/power_descriptor/active_icc_levels_vcc*
/sys/bus/platform/devices/*.ufs/power_descriptor/active_icc_levels_vcc*

Defined on file *sysfs-driver-ufs*

This file shows maximum VCC, VCCQ and VCCQ2 value for active ICC levels from 0 to 15. This is one of the UFS power descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/power_info/dev_pm
/sys/bus/platform/devices/*.ufs/power_info/dev_pm

Defined on file *sysfs-driver-ufs*

This file shows the UFS device power mode.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/power_info/gear
/sys/bus/platform/devices/*.ufs/power_info/gear

Defined on file *sysfs-driver-ufs*

This file shows the gear of UFS link.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/power_info/lane
/sys/bus/platform/devices/*.ufs/power_info/lane

Defined on file *sysfs-driver-ufs*

This file shows how many lanes are enabled on the UFS link, i.e., an output 2 means UFS link is operating with 2 lanes.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/power_info/link_state
/sys/bus/platform/devices/*.ufs/power_info/link_state

Defined on file *sysfs-driver-ufs*

This file shows the state of UFS link.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/power_info/mode
/sys/bus/platform/devices/*.ufs/power_info/mode

Defined on file *sysfs-driver-ufs*

This file shows the PA power mode of UFS.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/power_info/rate
/sys/bus/platform/devices/*.ufs/power_info/rate

Defined on file *sysfs-driver-ufs*

This file shows the speed rate of UFS link.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/rpm_lvl
/sys/bus/platform/devices/*.ufs/rpm_lvl

Defined on file *sysfs-driver-ufs*

This entry could be used to set or show the UFS device runtime power management level. The current driver implementation supports 7 levels with next target states:

0	UFS device will stay active, UIC link will stay active
1	UFS device will stay active, UIC link will hibernate
2	UFS device will be moved to sleep, UIC link will stay active
3	UFS device will be moved to sleep, UIC link will hibernate
4	UFS device will be powered off, UIC link will hibernate
5	UFS device will be powered off, UIC link will be powered off
6	UFS device will be moved to deep sleep, UIC link will be powered off. Note, deep sleep might not be supported in which case this value will not be accepted

/sys/bus/platform/drivers/ufshcd/*/rpm_target_dev_state
/sys/bus/platform/devices/*.ufs/rpm_target_dev_state

Defined on file *sysfs-driver-ufs*

This entry shows the target power mode of an UFS device for the chosen runtime power management level.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/rpm_target_link_state
/sys/bus/platform/devices/*.ufs/rpm_target_link_state

Defined on file *sysfs-driver-ufs*

This entry shows the target state of an UFS UIC link for the chosen runtime power management level.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/rtc_update_ms
/sys/bus/platform/devices/*.ufs/rtc_update_ms

Defined on file *sysfs-driver-ufs*

rtc_update_ms indicates how often the host should synchronize or update the UFS RTC. If set to 0, this will disable UFS RTC periodic update.

/sys/bus/platform/drivers/ufshcd/*/spm_lvl
/sys/bus/platform/devices/*.ufs/spm_lvl

Defined on file *sysfs-driver-ufs*

This entry could be used to set or show the UFS device system power management level. The current driver implementation supports 7 levels with next target states:

0	UFS device will stay active, UIC link will stay active
1	UFS device will stay active, UIC link will hibernate
2	UFS device will be moved to sleep, UIC link will stay active
3	UFS device will be moved to sleep, UIC link will hibernate
4	UFS device will be powered off, UIC link will hibernate
5	UFS device will be powered off, UIC link will be powered off
6	UFS device will be moved to deep sleep, UIC link will be powered off. Note, deep sleep might not be supported in which case this value will not be accepted

/sys/bus/platform/drivers/ufshcd/*/spm_target_dev_state
/sys/bus/platform/devices/*.ufs/spm_target_dev_state

Defined on file *sysfs-driver-ufs*

This entry shows the target power mode of an UFS device for the chosen system power management level.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/spm_target_link_state
/sys/bus/platform/devices/*.ufs/spm_target_link_state

Defined on file *sysfs-driver-ufs*

This entry shows the target state of an UFS UIC link for the chosen system power management level.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/string_descriptors/manufacturer_name
/sys/bus/platform/devices/*.ufs/string_descriptors/manufacturer_name

Defined on file *sysfs-driver-ufs*

This file contains a device manufacturer name string. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/string_descriptors/oem_id
/sys/bus/platform/devices/*.ufs/string_descriptors/oem_id

Defined on file *sysfs-driver-ufs*

This file contains a OEM ID string. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/string_descriptors/product_name
/sys/bus/platform/devices/*.ufs/string_descriptors/product_name

Defined on file *sysfs-driver-ufs*

This file contains a product name string. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/string_descriptors/product_revision
/sys/bus/platform/devices/*.ufs/string_descriptors/product_revision

Defined on file *sysfs-driver-ufs*

This file contains a product revision string. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/string_descriptors/serial_number
/sys/bus/platform/devices/*.ufs/string_descriptors/serial_number

Defined on file *sysfs-driver-ufs*

This file contains a device serial number string. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/bus/platform/drivers/ufshcd/*/wb_flush_threshold
/sys/bus/platform/devices/*.ufs/wb_flush_threshold

Defined on file *sysfs-driver-ufs*

wb_flush_threshold represents the threshold for flushing WriteBooster buffer, whose value expressed in unit of 10% granularity, such as '1' representing 10%, '2' representing 20%, and so on. If avail_wb_buff < wb_flush_threshold, it indicates that WriteBooster buffer needs to be flushed, otherwise it is not necessary.

/sys/bus/platform/drivers/ufshcd/*/wb_on
/sys/bus/platform/devices/*.ufs/wb_on

Defined on file *sysfs-driver-ufs*

This node is used to set or display whether UFS WriteBooster is enabled. Echo 0 to this file to disable UFS WriteBooster or 1 to enable it. The WriteBooster is enabled after power-on/reset, however, it will be disabled/enable while CLK scaling down/up (if the platform supports UFSHCD_CAP_CLK_SCALING). For a platform that doesn't support UFSHCD_CAP_CLK_SCALING, we can disable/enable WriteBooster through this sysfs node.

/sys/bus/platform/drivers/zynqmp_fpga_manager/firmware:zynqmp-firmware:pcap/status

Defined on file *sysfs-driver-zynqmp-fpga*

(RO) Read fpga status. Read returns a hexadecimal value that tells the current status of the FPGA device. Each bit position in the status value is described Below(see ug570 chapter 9). https://docs.xilinx.com/v/u/en-US/ug570-ultrascale-configuration

BIT(0)	0: No CRC error 1: CRC error
BIT(1)	0: Decryptor security not set 1: Decryptor security set
BIT(2)	0: MMCMs/PLLs are not locked 1: MMCMs/PLLs are locked
BIT(3)	0: DCI not matched 1: DCI matched
BIT(4)	0: Start-up sequence has not finished 1: Start-up sequence has finished
BIT(5)	0: All I/Os are placed in High-Z state 1: All I/Os behave as configured
BIT(6)	0: Flip-flops and block RAM are write disabled 1: Flip-flops and block RAM are write enabled
BIT(7)	0: GHIGH_B_STATUS asserted 1: GHIGH_B_STATUS deasserted
BIT(8) to BIT(10)	Status of the mode pins
BIT(11)	0: Initialization has not finished 1: Initialization finished
BIT(12)	Value on INIT_B_PIN pin
BIT(13)	0: Signal not released 1: Signal released
BIT(14)	Value on DONE_PIN pin.
BIT(15)	0: No IDCODE_ERROR 1: IDCODE_ERROR
BIT(16)	0: No SECURITY_ERROR 1: SECURITY_ERROR
BIT(17)	System Monitor over-temperature if set
BIT(18) to BIT(20)	Start-up state machine (0 to 7) Phase 0 = 000 Phase 1 = 001 Phase 2 = 011 Phase 3 = 010 Phase 4 = 110 Phase 5 = 111 Phase 6 = 101 Phase 7 = 100
BIT(25) to BIT(26)	Indicates the detected bus width 00 = x1 01 = x8 10 = x16 11 = x32

The other bits are reserved.

/sys/bus/rapidio/devices/<nn>:<d>:<iiii>

Defined on file *sysfs-bus-rapidio*

For each RapidIO device, the RapidIO subsystem creates files in an individual subdirectory with the following name format of device_name "nn:d:iiii", where:

nn	two-digit hexadecimal ID of RapidIO network where the device resides
d	device type: 'e' - for endpoint or 's' - for switch
iiii	four-digit device destID for endpoints, or switchID for switches

For example, below is a list of device directories that represents a typical RapidIO network with one switch, one host, and two agent endpoints, as it is seen by the enumerating host (with destID = 1):

```
/sys/bus/rapidio/devices/00:e:0000
/sys/bus/rapidio/devices/00:e:0002
/sys/bus/rapidio/devices/00:s:0001
```

NOTE:

An enumerating or discovering endpoint does not create a sysfs entry for itself, this is why an endpoint with destID=1 is not shown in the list.

Attributes Common for All RapidIO Devices

/sys/bus/rapidio/devices/<nn>:<d>:<iiii>/asm_did

Defined on file *sysfs-bus-rapidio*

(RO) returns identifier for the assembly containing the device

/sys/bus/rapidio/devices/<nn>:<d>:<iiii>/asm_rev

Defined on file *sysfs-bus-rapidio*

(RO) returns revision level of the assembly containing the device

/sys/bus/rapidio/devices/<nn>:<d>:<iiii>/asm_vid

Defined on file *sysfs-bus-rapidio*

(RO) returns vendor identifier of the assembly containing the device

/sys/bus/rapidio/devices/<nn>:<d>:<iiii>/config

Defined on file *sysfs-bus-rapidio*

(RW) Binary attribute to read from and write to the device configuration registers using the RapidIO maintenance transactions. This attribute is similar in behaviour to the ”config” attribute of PCI devices and provides an access to the RapidIO device registers using standard file read and write operations.

RapidIO Switch Device Attributes

RapidIO switches have additional attributes in sysfs. RapidIO subsystem supports common and device-specific sysfs attributes for switches. Because switches are integrated into the RapidIO subsystem, it offers a method to create device-specific sysfs attributes by specifying a callback function that may be set by the switch initialization routine during enumeration or discovery process.

/sys/bus/rapidio/devices/<nn>:<d>:<iiii>/destid

Defined on file *sysfs-bus-rapidio*

(RO) returns device destination ID assigned by the enumeration routine

/sys/bus/rapidio/devices/<nn>:<d>:<iiii>/device_rev

Defined on file *sysfs-bus-rapidio*

(RO) returns the device revision level

/sys/bus/rapidio/devices/<nn>:<d>:<iiii>/did

Defined on file *sysfs-bus-rapidio*

(RO) returns the device identifier

/sys/bus/rapidio/devices/<nn>:<d>:<iiii>/lprev

Defined on file *sysfs-bus-rapidio*

(RO) returns name of previous device (switch) on the path to the device that that owns this attribute

/sys/bus/rapidio/devices/<nn>:<d>:<iiii>/modalias

Defined on file *sysfs-bus-rapidio*

(RO) returns the device modalias

/sys/bus/rapidio/devices/<nn>:<d>:<iiii>/vid

Defined on file *sysfs-bus-rapidio*

(RO) returns the device vendor identifier

/sys/bus/rapidio/devices/<nn>:<s>:<iiii>/destid

Defined on file *sysfs-bus-rapidio*

(RO) device destination ID of the associated device that defines a route to the switch

/sys/bus/rapidio/devices/<nn>:<s>:<iiii>/errlog

Defined on file *sysfs-bus-rapidio*

(RO) reads contents of device error log until it is empty.

RapidIO Bus Attributes

/sys/bus/rapidio/devices/<nn>:<s>:<iiii>/hopcount

Defined on file *sysfs-bus-rapidio*

(RO) number of hops on the path to the switch

/sys/bus/rapidio/devices/<nn>:<s>:<iiii>/lnext

Defined on file *sysfs-bus-rapidio*

(RO) returns names of devices linked to the switch except one of a device linked to the ingress port (reported as ”lprev”). This is an array names with number of lines equal to number of ports in switch. If a switch port has no attached device, returns ”null” instead of a device name.

Device-specific Switch Attributes

IDT_GEN2-

/sys/bus/rapidio/devices/<nn>:<s>:<iiii>/routes

Defined on file *sysfs-bus-rapidio*

(RO) reports switch routing information in "destID port" format. This attribute reports only valid routing table entries, one line for each entry.

/sys/bus/rapidio/scan

Defined on file *sysfs-bus-rapidio*

(WO) Allows to trigger enumeration discovery process from user space. To initiate an enumeration or discovery process on specific mport device, a user needs to write mport_ID (not RapidIO destination ID) into this file. The mport_ID is a sequential number (0 ... RIO_MAX_MPORTS) assigned to the mport device. For example, for a machine with a single RapidIO controller, mport_ID for that controller always will be 0. To initiate RapidIO enumeration/discovery on all available mports a user must write '-1' (or RIO_MPORT_ANY) into this attribute file.

/sys/bus/rbd/add

Defined on file *sysfs-bus-rbd*

(WO) Add rbd block device.

Usage: <mon ip addr> <options> <pool name> <rbd image name> [<snap name>]

Example:

```
$ echo "192.168.0.1 name=admin rbd foo" > /sys/bus/rbd/add
```

The snapshot name can be "-" or omitted to map the image read/write. A <dev-id> will be assigned for any registered block device. If snapshot is used, it will be mapped read-only.

/sys/bus/rbd/add_single_major

Defined on file *sysfs-bus-rbd*

(WO) Available only if rbd module is inserted with single_major parameter set to true.

Usage is the same as for */sys/bus/rbd/add*. If present, this should be used instead of the latter: any attempts to use */sys/bus/rbd/add* if */sys/bus/rbd/add_single_major* is available will fail for backwards compatibility reasons.

/sys/bus/rbd/devices/<dev-id>/image_id
/sys/bus/rbd/devices/<dev-id>/features

Defined on file *sysfs-bus-rbd*

image_id	(RO) The unique id for the rbd image. (For rbd image format 1 this is empty.)
features	(RO) A hexadecimal encoding of the feature bits for this image.

/sys/bus/rbd/devices/<dev-id>/minor

Defined on file *sysfs-bus-rbd*

(RO) The block device minor number.

/sys/bus/rbd/devices/<dev-id>/parent

Defined on file *sysfs-bus-rbd*

(RO) Information identifying the chain of parent images in a layered rbd image. Entries are separated by empty lines.

/sys/bus/rbd/devices/<dev-id>/pool_id

Defined on file *sysfs-bus-rbd*

(RO) The unique identifier for the rbd image's pool. This is a permanent attribute of the pool. A pool's id will never change.

/sys/bus/rbd/devices/<dev-id>/size
/sys/bus/rbd/devices/<dev-id>/major
/sys/bus/rbd/devices/<dev-id>/client_id
/sys/bus/rbd/devices/<dev-id>/pool
/sys/bus/rbd/devices/<dev-id>/name
/sys/bus/rbd/devices/<dev-id>/refresh
/sys/bus/rbd/devices/<dev-id>/current_snap

Defined on file *sysfs-bus-rbd*

size	(RO) The size (in bytes) of the mapped block device.
major	(RO) The block device major number.
client_id	(RO) The ceph unique client id that was assigned for this specific session.
pool	(RO) The name of the storage pool where this rbd image resides. An rbd image name is unique within its pool.
name	(RO) The name of the rbd image.
refresh	(WO) Writing to this file will reread the image header data and set all relevant data structures accordingly.
current_snap	(RO) The current snapshot for which the device is mapped.

/sys/bus/rbd/devices/<dev-id>/snap_id
/sys/bus/rbd/devices/<dev-id>/config_info
/sys/bus/rbd/devices/<dev-id>/cluster_fsid
/sys/bus/rbd/devices/<dev-id>/client_addr

Defined on file *sysfs-bus-rbd*

snap_id	(RO) The current snapshot's id.
config_info	(RO) The string written into /sys/bus/rbd/add{,_single_major}.
cluster_fsid	(RO) The ceph cluster UUID.
client_addr	(RO) The ceph unique client entity_addr_t (address + nonce). The format is <address>:<port>/<nonce>: '1.2.3.4:1234/5678' or '[1:2:3:4:5:6:7:8]:1234/5678'.

/sys/bus/rbd/remove

Defined on file *sysfs-bus-rbd*

(WO) Remove rbd block device.

Usage: <dev-id> [force]

Example:

```
$ echo 2 > /sys/bus/rbd/remove
```

Optional "force" argument which when passed will wait for running requests and then unmap the image. Requests sent to the driver after initiating the removal will be failed. (August 2016, since 4.9.)

/sys/bus/rbd/remove_single_major

Defined on file *sysfs-bus-rbd*

(WO) Available only if rbd module is inserted with single_major parameter set to true.

Usage is the same as for */sys/bus/rbd/remove*. If present, this should be used instead of the latter: any attempts to use */sys/bus/rbd/remove* if */sys/bus/rbd/remove_single_major* is available will fail for backwards compatibility reasons.

/sys/bus/rbd/supported_features

Defined on file *sysfs-bus-rbd*

(RO) Displays the features supported by the rbd module so that userspace can generate meaningful error messages and spell out unsupported features that need to be disabled.

/sys/bus/rpmsg/devices/.../announce

Defined on file *sysfs-bus-rpmsg*

Every rpmsg device is a communication channel with a remote processor. Channels are identified by a textual name (see /sys/bus/rpmsg/devices/.../name above) and have a local ("source") rpmsg address, and remote ("destination") rpmsg address.

A channel is first created when an entity, whether local or remote, starts listening on it for messages (and is thus called an rpmsg server).

When that happens, a ”name service” announcement is sent to the other processor, in order to let it know about the creation of the channel (this way remote clients know they can start sending messages).

This sysfs entry tells us whether the channel is a local server channel that is announced (values are either true or false).

/sys/bus/rpmsg/devices/.../driver_override

Defined on file *sysfs-bus-rpmsg*

Every rpmsg device is a communication channel with a remote processor. Channels are identified by a textual name (see /sys/bus/rpmsg/devices/.../name above) and have a local (”source”) rpmsg address, and remote (”destination”) rpmsg address.

The listening entity (or client) which communicates with a remote processor is referred as rpmsg driver. The rpmsg device and rpmsg driver are matched based on rpmsg device name and rpmsg driver ID table.

This sysfs entry allows the rpmsg driver for a rpmsg device to be specified which will override standard OF, ID table and name matching.

/sys/bus/rpmsg/devices/.../dst

Defined on file *sysfs-bus-rpmsg*

Every rpmsg device is a communication channel with a remote processor. Channels have a local (”source”) rpmsg address, and remote (”destination”) rpmsg address. When an entity starts listening on one end of a channel, it assigns it with a unique rpmsg address (a 32 bits integer). This way when inbound messages arrive to this address, the rpmsg core dispatches them to the listening entity.

This sysfs entry contains the dst (remote) rpmsg address of this channel. If it contains 0xffffffff, then an address wasn't assigned (can happen if the kernel driver that is attached to this channel is exposing a service to the remote processor. This make it a local rpmsg server, and it is listening for inbound messages that may be sent from any remote rpmsg client; it is not bound to a single remote entity).

/sys/bus/rpmsg/devices/.../name

Defined on file *sysfs-bus-rpmsg*

Every rpmsg device is a communication channel with a remote processor. Channels are identified with a (textual) name, which is maximum 32 bytes long (defined as RPMSG_NAME_SIZE in rpmsg.h).

This sysfs entry contains the name of this channel.

/sys/bus/rpmsg/devices/.../src

Defined on file *sysfs-bus-rpmsg*

Every rpmsg device is a communication channel with a remote processor. Channels have a local (”source”) rpmsg address, and remote (”destination”) rpmsg address. When an entity

starts listening on one end of a channel, it assigns it with a unique rpmsg address (a 32 bits integer). This way when inbound messages arrive to this address, the rpmsg core dispatches them to the listening entity (a kernel driver).

This sysfs entry contains the src (local) rpmsg address of this channel. If it contains 0xffffffff, then an address wasn't assigned (can happen if no driver exists for this channel).

/sys/bus/scsi/drivers/st/debug_flag

Defined on file *sysfs-driver-st*

This file allows you to turn debug output from the st driver off if you write a '0' to the file or on if you write a '1'. Note that debug output requires that the module be compiled with the #define DEBUG set to a non-zero value (this is the default). If DEBUG is set to 0 then this file will not appear in sysfs as its presence is conditional upon debug output support being compiled into the module.

/sys/bus/siox/devices/siox-X-Y/connected

Defined on file *sysfs-bus-siox*

Read-only value. ”0” means the Yth device on siox bus X isn't ”connected” i.e. communication with it is not ensured. ”1” signals a working connection.

/sys/bus/siox/devices/siox-X-Y/inbytes

Defined on file *sysfs-bus-siox*

Read-only value reporting the inbytes value provided to siox-X/device_add

/sys/bus/siox/devices/siox-X-Y/outbytes

Defined on file *sysfs-bus-siox*

Read-only value reporting the outbytes value provided to siox-X/device_add.

/sys/bus/siox/devices/siox-X-Y/status_errors

Defined on file *sysfs-bus-siox*

Counts the number of time intervals when the read status byte doesn't yield the expected value.

/sys/bus/siox/devices/siox-X-Y/type

Defined on file *sysfs-bus-siox*

Read-only value reporting the type value provided to siox-X/device_add.

/sys/bus/siox/devices/siox-X-Y/watchdog

Defined on file *sysfs-bus-siox*

Read-only value reporting if the watchdog of the siox device is active. ”0” means the watchdog is not active and the device is expected to be operational. ”1” means the watchdog keeps the device in reset.

/sys/bus/siox/devices/siox-X-Y/watchdog_errors

Defined on file *sysfs-bus-siox*

Read-only value reporting the number to time intervals when the watchdog was active.

/sys/bus/siox/devices/siox-X/active

Defined on file *sysfs-bus-siox*

On reading represents the current state of the bus. If it contains a ”0” the bus is stopped and connected devices are expected to not do anything because their watchdog triggered. When the file contains a ”1” the bus is operated and periodically does a push-pull cycle to write and read data from the connected devices.

When writing a ”0” or ”1” the bus moves to the described state.

/sys/bus/siox/devices/siox-X/device_add

Defined on file *sysfs-bus-siox*

Write-only file. Write

<type> <inbytes> <outbytes> <statustype>

to add a new device dynamically. <type> is the name that is used to match to a driver (similar to the platform bus). <inbytes> and <outbytes> define the length of the input and output shift register in bytes respectively.

<statustype> defines the 4 bit device type that is check to identify connection problems.

The new device is added to the end of the existing chain.

/sys/bus/siox/devices/siox-X/device_remove

Defined on file *sysfs-bus-siox*

Write-only file. A single write removes the last device in the siox chain.

/sys/bus/siox/devices/siox-X/poll_interval_ns

Defined on file *sysfs-bus-siox*

Defines the interval between two poll cycles in nano seconds. Note this is rounded to jiffies on writing. On reading the current value is returned.

/sys/bus/soc

Defined on file *sysfs-devices-soc*

The /sys/bus/soc/ directory contains the usual sub-folders expected under most buses. /sys/bus/soc/devices is of particular interest, as it contains a symlink for each SoC device found on the system. Each symlink points back into the aforementioned */sys/devices/socX* devices.

/sys/bus/soundwire/devices/sdw-master-<N>/revision

Defined on file *sysfs-bus-soundwire-master*

SoundWire Master-N DisCo properties. These properties are defined by MIPI DisCo Specification for SoundWire. They define various properties of the Master and are used by the bus to configure the Master. clk_stop_modes is a bitmask for simplifications and combines the clock-stop-mode0 and clock-stop-mode1 properties.

/sys/bus/soundwire/devices/sdw:.../dev-properties/mipi_revision

Defined on file *sysfs-bus-soundwire-slave*

SoundWire Slave DisCo properties. These properties are defined by MIPI DisCo Specification for SoundWire. They define various properties of the SoundWire Slave and are used by the bus to configure the Slave

/sys/bus/soundwire/devices/sdw:.../dp0/max_word

Defined on file *sysfs-bus-soundwire-slave*

SoundWire Slave Data Port-0 DisCo properties. These properties are defined by MIPI DisCo Specification for the SoundWire. They define various properties of the Data port 0 are used by the bus to configure the Data Port 0.

/sys/bus/soundwire/devices/sdw:.../dp<N>_src/max_word

Defined on file *sysfs-bus-soundwire-slave*

SoundWire Slave Data Source/Sink Port-N DisCo properties. These properties are defined by MIPI DisCo Specification for SoundWire. They define various properties of the Source/Sink Data port N and are used by the bus to configure the Data Port N.

/sys/bus/soundwire/devices/sdw:.../status

Defined on file *sysfs-bus-soundwire-slave*

SoundWire Slave status

These properties report the Slave status, e.g. if it is UNATTACHED or not, and in the latter case show the device_number. This status information is useful to detect devices exposed by platform firmware but not physically present on the bus, and conversely devices not exposed in platform firmware but enumerated.

/sys/bus/spi/<dev>/reset

Defined on file *sysfs-driver-ge-achc*

This file represents the microcontroller's reset line. 1 means the reset line is asserted, 0 means it's not asserted. The file is read and writable.

/sys/bus/spi/<dev>/update_firmware

Defined on file *sysfs-driver-ge-achc*

Write 1 to this file to update the ACHC microcontroller firmware via the EzPort interface. For this the kernel will load "achc.bin" via the firmware API (so usually from /lib/firmware). The write will block until the FW has either been flashed successfully or an error occurred.

/sys/bus/spi/devices/.../spi-nor/jedec_id

Defined on file *sysfs-bus-spi-devices-spi-nor*

(RO) The JEDEC ID of the SPI NOR flash as reported by the flash device.

The attribute is not present if the flash doesn't support the "Read JEDEC ID" command (9Fh). This is the case for non-JEDEC compliant flashes.

/sys/bus/spi/devices/.../spi-nor/manufacturer

Defined on file *sysfs-bus-spi-devices-spi-nor*

(RO) Manufacturer of the SPI NOR flash.

The attribute is not present if the flash device isn't known to the kernel and is only probed by its SFDP tables.

/sys/bus/spi/devices/.../spi-nor/partname

Defined on file *sysfs-bus-spi-devices-spi-nor*

(RO) Part name of the SPI NOR flash.

The attribute is optional. User space should not rely on it to be present or even correct. Instead, user space should read the jedec_id attribute.

/sys/bus/spi/devices/.../spi-nor/sfdp

Defined on file *sysfs-bus-spi-devices-spi-nor*

(RO) This attribute is only present if the SPI NOR flash device supports the "Read SFDP" command (5Ah).

If present, it contains the complete SFDP (serial flash discoverable parameters) binary data of the flash.

/sys/bus/surface_aggregator/devices/01:0e:01:00:01/state

Defined on file *sysfs-bus-surface_aggregator-tabletsw*

This attribute returns a string with the current type-cover or device posture, as indicated by the embedded controller. Currently returned posture states are:

- ”disconnected”: The type-cover has been disconnected.
- ”closed”: The type-cover has been folded closed and lies on top of the display.
- ”laptop”: The type-cover is open and in laptop-mode, i.e., ready for normal use.
- ”folded-canvas”: The type-cover has been folded back part-ways, but does not lie flush with the back side of the device. In general, this means that the kick-stand is used and extended atop of the cover.
- ”folded-back”: The type cover has been fully folded back and lies flush with the back side of the device.
- ”<unknown>”: The current state is unknown to the driver, for example due to newer as-of-yet unsupported hardware.

New states may be introduced with new hardware. Users therefore must not rely on this list of states being exhaustive and gracefully handle unknown states.

/sys/bus/surface_aggregator/devices/01:26:01:00:01/state

Defined on file *sysfs-bus-surface_aggregator-tabletsw*

This attribute returns a string with the current device posture, as indicated by the embedded controller. Currently returned posture states are:

- ”closed”: The lid of the device is closed.
- ”laptop”: The lid of the device is opened and the device operates as a normal laptop.
- ”slate”: The screen covers the keyboard or has been flipped back and the device operates mainly based on touch input.
- ”tablet”: The device operates as tablet and exclusively relies on touch input (or external peripherals).
- ”<unknown>”: The current state is unknown to the driver, for example due to newer as-of-yet unsupported hardware.

New states may be introduced with new hardware. Users therefore must not rely on this list of states being exhaustive and gracefully handle unknown states.

/sys/bus/tee/devices/optee-ta-<uuid>/

Defined on file *sysfs-bus-optee-devices*

OP-TEE bus provides reference to registered drivers under this directory. The <uuid> matches Trusted Application (TA) driver and corresponding TA in secure OS. Drivers are free to create needed API under optee-ta-<uuid> directory.

/sys/bus/tee/devices/optee-ta-<uuid>/need_supplicant

Defined on file *sysfs-bus-optee-devices*

Allows to distinguish whether an OP-TEE based TA/device requires user-space tee-supplicant to function properly or not. This attribute will be present for devices which depend on tee-supplicant to be running.

/sys/bus/thunderbolt/devices/.../authorized

Defined on file *sysfs-bus-thunderbolt*

This attribute is used to authorize Thunderbolt devices after they have been connected. If the device is not authorized, no PCIe devices are available to the system.

Contents of this attribute will be 0 when the device is not yet authorized.

Possible values are supported:

0	The device will be de-authorized (only supported if deauthorization attribute under domain contains 1)
1	The device will be authorized and connected

When key attribute contains 32 byte hex string the possible values are:

0	The device will be de-authorized (only supported if deauthorization attribute under domain contains 1)
1	The 32 byte hex string is added to the device NVM and the device is authorized.
2	Send a challenge based on the 32 byte hex string. If the challenge response from device is valid, the device is authorized. In case of failure errno will be ENOKEY if the device did not contain a key at all, and EKEYREJECTED if the challenge response did not match.

/sys/bus/thunderbolt/devices/.../boot

Defined on file *sysfs-bus-thunderbolt*

This attribute contains 1 if Thunderbolt device was already authorized on boot and 0 otherwise.

/sys/bus/thunderbolt/devices/.../device

Defined on file *sysfs-bus-thunderbolt*

This attribute contains id of this device extracted from the device DROM.

/sys/bus/thunderbolt/devices/.../device_name

Defined on file *sysfs-bus-thunderbolt*

This attribute contains name of this device extracted from the device DROM.

/sys/bus/thunderbolt/devices/.../domainX/boot_acl

Defined on file *sysfs-bus-thunderbolt*

Holds a comma separated list of device unique_ids that are allowed to be connected automatically during system startup (e.g boot devices). The list always contains maximum supported number of unique_ids where unused entries are empty. This allows the userspace software to determine how many entries the controller supports. If there are multiple controllers, each controller has its own ACL list and size may be different between the controllers.

System BIOS may have an option "Preboot ACL" or similar that needs to be selected before this list is taken into consideration.

Software always updates a full list in each write.

If a device is authorized automatically during boot its boot attribute is set to 1.

/sys/bus/thunderbolt/devices/.../domainX/deauthorization

Defined on file *sysfs-bus-thunderbolt*

This attribute tells whether the system supports de-authorization of devices. Value of 1 means user can de-authorize PCIe tunnel by writing 0 to authorized attribute under each device.

/sys/bus/thunderbolt/devices/.../domainX/iommu_dma_protection

Defined on file *sysfs-bus-thunderbolt*

This attribute tells whether the system uses IOMMU for DMA protection. Value of 1 means IOMMU is used 0 means it is not (DMA protection is solely based on Thunderbolt security levels).

/sys/bus/thunderbolt/devices/.../domainX/security

Defined on file *sysfs-bus-thunderbolt*

This attribute holds current Thunderbolt security level set by the system BIOS. Possible values are:

none	All devices are automatically authorized
user	Devices are only authorized based on writing appropriate value to the authorized attribute
secure	Require devices that support secure connect at minimum. User needs to authorize each device.
dponly	Automatically tunnel Display port (and USB). No PCIe tunnels are created.
usbonly	Automatically tunnel USB controller of the connected Thunderbolt dock (and Display Port). All PCIe links downstream of the dock are removed.
nopcie	USB4 system where PCIe tunneling is disabled from the BIOS.

/sys/bus/thunderbolt/devices/.../generation

Defined on file *sysfs-bus-thunderbolt*

This attribute contains the generation of the Thunderbolt controller associated with the device. It will contain 4 for USB4.

/sys/bus/thunderbolt/devices/.../key

Defined on file *sysfs-bus-thunderbolt*

When a devices supports Thunderbolt secure connect it will have this attribute. Writing 32 byte hex string changes authorization to use the secure connection method instead. Writing an empty string clears the key and regular connection method can be used again.

/sys/bus/thunderbolt/devices/.../maxhopid

Defined on file *sysfs-bus-thunderbolt*

Only set for XDomains. The maximum HopID the other host supports as its input HopID.

/sys/bus/thunderbolt/devices/.../nvm_authenticate

Defined on file *sysfs-bus-thunderbolt*

When new NVM image is written to the non-active NVM area (through non_activeX NVMem device), the authentication procedure is started by writing to this file. If everything goes well, the device is restarted with the new NVM firmware. If the image verification fails an error code is returned instead.

This file will accept writing values "1", "2" or "3".

- Writing "1" will flush the image to the storage area and authenticate the image in one action.
- Writing "2" will run some basic validation on the image and flush it to the storage area.
- Writing "3" will authenticate the image that is currently written in the storage area. This is only supported with USB4 devices and retimers.

When read holds status of the last authentication operation if an error occurred during the process. This is directly the status value from the DMA configuration based mailbox before the device is power cycled. Writing 0 here clears the status.

/sys/bus/thunderbolt/devices/.../nvm_authenticate_on_disconnect

Defined on file *sysfs-bus-thunderbolt*

For supported devices, automatically authenticate the new Thunderbolt image when the device is disconnected from the host system.

This file will accept writing values "1" or "2"

- Writing "1" will flush the image to the storage area and prepare the device for authentication on disconnect.
- Writing "2" will run some basic validation on the image and flush it to the storage area.

/sys/bus/thunderbolt/devices/.../nvm_version

Defined on file *sysfs-bus-thunderbolt*

If the device has upgradeable firmware the version number is available here. Format: %x.%x, major.minor. If the device is in safe mode reading the file returns -ENODATA instead as the NVM version is not available.

/sys/bus/thunderbolt/devices/.../rx_lanes

Defined on file *sysfs-bus-thunderbolt*

This attribute reports number of RX lanes the device is using simultaneously through its upstream port.

/sys/bus/thunderbolt/devices/.../rx_speed

Defined on file *sysfs-bus-thunderbolt*

This attribute reports the device RX speed per lane. All RX lanes run at the same speed.

/sys/bus/thunderbolt/devices/.../tx_lanes

Defined on file *sysfs-bus-thunderbolt*

This attribute reports number of TX lanes the device is using simultaneously through its upstream port.

/sys/bus/thunderbolt/devices/.../tx_speed

Defined on file *sysfs-bus-thunderbolt*

This attribute reports the TX speed per lane. All TX lanes run at the same speed.

/sys/bus/thunderbolt/devices/.../unique_id

Defined on file *sysfs-bus-thunderbolt*

This attribute contains unique_id string of this device. This is either read from hardware registers (UUID on newer hardware) or based on UID from the device DROM. Can be used to uniquely identify particular device.

/sys/bus/thunderbolt/devices/.../vendor

Defined on file *sysfs-bus-thunderbolt*

This attribute contains vendor id of this device extracted from the device DROM.

/sys/bus/thunderbolt/devices/.../vendor_name

Defined on file *sysfs-bus-thunderbolt*

This attribute contains vendor name of this device extracted from the device DROM.

/sys/bus/thunderbolt/devices/<device>:<port>.<index>/device

Defined on file *sysfs-bus-thunderbolt*

Retimer device identifier read from the hardware.

/sys/bus/thunderbolt/devices/<device>:<port>.<index>/nvm_authenticate

Defined on file *sysfs-bus-thunderbolt*

When new NVM image is written to the non-active NVM area (through non_activeX NVMem device), the authentication procedure is started by writing 1 to this file. If everything goes well, the device is restarted with the new NVM firmware. If the image verification fails an error code is returned instead.

When read holds status of the last authentication operation if an error occurred during the process. Format: %x.

/sys/bus/thunderbolt/devices/<device>:<port>.<index>/nvm_version

Defined on file *sysfs-bus-thunderbolt*

Holds retimer NVM version number. Format: %x.%x, major.minor.

/sys/bus/thunderbolt/devices/<device>:<port>.<index>/vendor

Defined on file *sysfs-bus-thunderbolt*

Retimer vendor identifier read from the hardware.

/sys/bus/thunderbolt/devices/<xdomain>.<service>/key

Defined on file *sysfs-bus-thunderbolt*

This contains name of the property directory the XDomain service exposes. This entry describes the protocol in question. Following directories are already reserved by the Apple XDomain specification:

network	IP/ethernet over Thunderbolt
targetdm	Target disk mode protocol over Thunderbolt
extdisp	External display mode protocol over Thunderbolt

/sys/bus/thunderbolt/devices/<xdomain>.<service>/modalias

Defined on file *sysfs-bus-thunderbolt*

Stores the same MODALIAS value emitted by uevent for the XDomain service. Format: tbtsvc:kSpNvNrN

/sys/bus/thunderbolt/devices/<xdomain>.<service>/prtcid

Defined on file *sysfs-bus-thunderbolt*

This contains XDomain protocol identifier the XDomain service supports.

/sys/bus/thunderbolt/devices/<xdomain>.<service>/prtcrevs

Defined on file *sysfs-bus-thunderbolt*

This contains XDomain software version the XDomain service supports.

/sys/bus/thunderbolt/devices/<xdomain>.<service>/prtcstns

Defined on file *sysfs-bus-thunderbolt*

This contains XDomain service specific settings as bitmask. Format: %x

/sys/bus/thunderbolt/devices/<xdomain>.<service>/prtcvers

Defined on file *sysfs-bus-thunderbolt*

This contains XDomain protocol version the XDomain service supports.

/sys/bus/thunderbolt/devices/usb4_portX/connector

Defined on file *sysfs-bus-thunderbolt*

Symlink to the USB Type-C connector. This link is only created when USB Type-C Connector Class is enabled, and only if the system firmware is capable of describing the connection between a port and its connector.

/sys/bus/thunderbolt/devices/usb4_portX/link

Defined on file *sysfs-bus-thunderbolt*

Returns the current link mode. Possible values are ”usb4”, ”tbt” and ”none”.

/sys/bus/thunderbolt/devices/usb4_portX/offline

Defined on file *sysfs-bus-thunderbolt*

Writing 1 to this attribute puts the USB4 port into offline mode. Only allowed when there is nothing connected to the port (link attribute returns ”none”). Once the port is in offline mode it does not receive any hotplug events. This is used to update NVM firmware of on-board retimers. Writing 0 puts the port back to online mode.

This attribute is only visible if the platform supports powering on retimers when there is no cable connected.

/sys/bus/thunderbolt/devices/usb4_portX/rescan

Defined on file *sysfs-bus-thunderbolt*

When the USB4 port is in offline mode writing 1 to this attribute forces rescan of the sideband for on-board retimers. Each retimer appear under the USB4 port as if the USB4 link was up. These retimers act in the same way as if the cable was connected so upgrading their NVM firmware can be done the usual way.

/sys/bus/typec/devices/.../active

Defined on file *sysfs-bus-typec*

Shows if the mode is active or not. The attribute can be used for entering/exiting the mode. Entering/exiting modes is supported as synchronous operation so write(2) to the attribute does not return until the enter/exit mode operation has finished. The attribute is notified when the mode is entered/exited so poll(2) on the attribute wakes up. Entering/exiting a mode will also generate uevent KOBJ_CHANGE.

Valid values are boolean.

/sys/bus/typec/devices/.../description

Defined on file *sysfs-bus-typec*

Shows description of the mode. The description is optional for the drivers, just like with the Billboard Devices.

/sys/bus/typec/devices/.../displayport/configuration

Defined on file *sysfs-driver-typec-displayport*

Shows the current DisplayPort configuration for the connector. Valid values are USB, source and sink. Source means DisplayPort source, and sink means DisplayPort sink.

All supported configurations are listed as space separated list with the active one wrapped in square brackets.

Source example:

USB [source] sink

The configuration can be changed by writing to the file

Note. USB configuration does not equal to Exit Mode. It is separate configuration defined in VESA DisplayPort Alt Mode on USB Type-C Standard. Functionally it equals to the situation where the mode has been exited (to exit the mode, see *testing/sysfs-bus-typec*, and use file /sys/bus/typec/devices/.../active).

/sys/bus/typec/devices/.../displayport/hpd

Defined on file *sysfs-driver-typec-displayport*

VESA DisplayPort Alt Mode on USB Type-C Standard defines how HotPlugDetect(HPD) shall be supported on the USB-C connector when operating in DisplayPort Alt Mode. This is a read only node which reflects the current state of HPD.

Valid values:

- **1: when HPD's logical state is high (HPD_High) as defined** by VESA DisplayPort Alt Mode on USB Type-C Standard.
- **0 when HPD's logical state is low (HPD_Low) as defined by** VESA DisplayPort Alt Mode on USB Type-C Standard.

/sys/bus/typec/devices/.../displayport/pin_assignment

Defined on file *sysfs-driver-typec-displayport*

VESA DisplayPort Alt Mode on USB Type-C Standard defines six different pin assignments for USB Type-C connector that are labeled A, B, C, D, E, and F. The supported pin assignments are listed as space separated list with the active one wrapped in square brackets.

Example:

C [D]

Pin assignment can be changed by writing to the file. It is possible to set pin assignment before configuration has been set, but the assignment will not be active before the connector is actually configured.

Note. As of VESA DisplayPort Alt Mode on USB Type-C Standard version 1.0b, pin assignments A, B, and F are deprecated. Only pin assignment D can now carry simultaneously one channel of USB SuperSpeed protocol. From user perspective pin assignments C and E are equal, where all channels on the connector are used for carrying DisplayPort protocol (allowing higher resolutions).

/sys/bus/typec/devices/.../mode

Defined on file *sysfs-bus-typec*

The index number of the mode returned by Discover Modes USB Power Delivery command. Depending on the alternate mode, the mode index may be significant.

With some alternate modes (SVIDs), the mode index is assigned for specific functionality in the specification for that alternate mode.

With other alternate modes, the mode index values are not assigned, and can not be therefore used for identification. When the mode index is not assigned, identifying the alternate mode must be done with either mode VDO or the description.

/sys/bus/typec/devices/.../svid

Defined on file *sysfs-bus-typec*

The Standard or Vendor ID (SVID) assigned by USB-IF for this alternate mode.

/sys/bus/typec/devices/.../vdo

Defined on file *sysfs-bus-typec*

Shows the VDO in hexadecimal returned by Discover Modes command for this mode.

/sys/bus/usb-serial/drivers/.../new_id

Defined on file *sysfs-bus-usb*

For serial USB drivers, this attribute appears under the extra bus folder ”usb-serial” in sysfs; apart from that difference, all descriptions from the entry ”/sys/bus/usb/drivers/.../new_id” apply.

/sys/bus/usb/.../decimals

Defined on file *sysfs-bus-usb-devices-usbsevseg*

Controls the decimal places on the device. To set the nth decimal place, give this field the value of `10 ** n`. Assume this field has the value k and has 1 or more decimal places set, to set the mth place (where m is not already set), change this fields value to `k + 10 ** m`.

/sys/bus/usb/.../mode_msb
/sys/bus/usb/.../mode_lsb

Defined on file *sysfs-bus-usb-devices-usbsevseg*

Controls the devices display mode. For a 6 character display the values are

MSB 0x06; LSB 0x3F, and

for an 8 character display the values are

MSB 0x08; LSB 0xFF.

/sys/bus/usb/.../powered

Defined on file *sysfs-bus-usb-devices-usbsevseg*

Controls whether the device's display will powered. A value of 0 is off and a non-zero value is on.

/sys/bus/usb/.../text

Defined on file *sysfs-bus-usb-devices-usbsevseg*

The text (or data) for the device to display

/sys/bus/usb/.../textmode

Defined on file *sysfs-bus-usb-devices-usbsevseg*

Controls the way the device interprets its text buffer. raw: each character controls its segment manually hex: each character is between 0-15 ascii: each character is between '0'-'9' and 'A'-'F'.

/sys/bus/usb/device/.../authorized

Defined on file *sysfs-bus-usb*

Authorized devices are available for use by device drivers, non-authorized one are not. By default, wired USB devices are authorized.

/sys/bus/usb/devices/.../<hub_interface>/port<X>

Defined on file *sysfs-bus-usb*

The /sys/bus/usb/devices/.../<hub_interface>/port<X> is usb port device's sysfs directory.

/sys/bus/usb/devices/.../<hub_interface>/port<X>/connect_type

Defined on file *sysfs-bus-usb*

Some platforms provide usb port connect types through ACPI. This attribute is to expose these information to user space. The file will read ”hotplug”, ”hardwired” and ”not used” if the information is available, and ”unknown” otherwise.

/sys/bus/usb/devices/.../<hub_interface>/port<X>/connector

Defined on file *sysfs-bus-usb*

Link to the USB Type-C connector when available. This link is only created when USB Type-C Connector Class is enabled, and only if the system firmware is capable of describing the connection between a port and its connector.

/sys/bus/usb/devices/.../<hub_interface>/port<X>/disable

Defined on file *sysfs-bus-usb*

This file controls the state of a USB port, including Vbus power output (but only on hubs that support power switching -- most hubs don't support it). If a port is disabled, the port is unusable: Devices attached to the port will not be detected, initialized, or enumerated.

/sys/bus/usb/devices/.../<hub_interface>/port<X>/early_stop

Defined on file *sysfs-bus-usb*

Some USB hosts have some watchdog mechanisms so that the device may enter ramdump if it takes a long time during port initialization. This attribute allows each port just has two attempts so that the port initialization will be failed quickly. In addition, if a port which is marked with early_stop has failed to initialize, it will ignore all future connections until this attribute is clear.

/sys/bus/usb/devices/.../<hub_interface>/port<X>/location

Defined on file *sysfs-bus-usb*

Some platforms provide usb port physical location through firmware. This is used by the kernel to pair up logical ports mapping to the same physical connector. The attribute exposes the raw location value as a hex integer.

/sys/bus/usb/devices/.../<hub_interface>/port<X>/over_current_count

Defined on file *sysfs-bus-usb*

Most hubs are able to detect over-current situations on their ports and report them to the kernel. This attribute is to expose the number of over-current situation occurred on a specific port to user space. This file will contain an unsigned 32 bit value which wraps to 0 after its maximum is reached. This file supports poll() for monitoring changes to this value in user space.

Any time this value changes the corresponding hub device will send a udev event with the following attributes:

```
OVER_CURRENT_PORT=/sys/bus/usb/devices/.../<hub_interface>/port<X>
OVER_CURRENT_COUNT=[current value of this sysfs attribute]
```

/sys/bus/usb/devices/.../<hub_interface>/port<X>/quirks

Defined on file *sysfs-bus-usb*

In some cases, we care about time-to-active for devices connected on a specific port (e.g. non-standard USB port like pogo pins), where the device to be connected is known in advance, and behaves well according to the specification. This attribute is a bit-field that controls the behavior of a specific port:

- Bit 0 of this field selects the "old" enumeration scheme, as it is considerably faster (it only causes one USB reset instead of 2).

  The old enumeration scheme can also be selected globally using /sys/module/usbcore/parameters/old_scheme_first, but it is often not desirable as the new scheme was introduced to increase compatibility with more devices.

- Bit 1 reduces TRSTRCY to the 10 ms that are required by the USB 2.0 specification, instead of the 50 ms that are normally used to help make enumeration work better on some high speed devices.

/sys/bus/usb/devices/.../<hub_interface>/port<X>/state

Defined on file *sysfs-bus-usb*

Indicates current state of the USB device attached to the port. Valid states are: 'not-attached', 'attached', 'powered', 'reconnecting', 'unauthenticated', 'default', 'addressed', 'configured', and 'suspended'. This file supports poll() to monitor the state change from user space.

/sys/bus/usb/devices/.../<hub_interface>/port<X>/usb3_lpm_permit

Defined on file *sysfs-bus-usb*

Some USB3.0 devices are not friendly to USB3 LPM. usb3_lpm_permit attribute allows enabling/disabling usb3 lpm of a port. It takes effect both before and after a usb device is enumerated. Supported values are "0" if both u1 and u2 are NOT permitted, "u1" if only u1 is permitted, "u2" if only u2 is permitted, "u1_u2" if both u1 and u2 are permitted.

/sys/bus/usb/devices/.../enable_compliance

Defined on file *sysfs-bus-usb-lvstest*

Write to this node to set the port to compliance mode to test with Link Layer Validation device. It is needed for TD.7.34.

/sys/bus/usb/devices/.../get_dev_desc

Defined on file *sysfs-bus-usb-lvstest*

Write to this node to issue "Get Device Descriptor" for Link Layer Validation device. It is needed for TD.7.06.

/sys/bus/usb/devices/.../hot_reset

Defined on file *sysfs-bus-usb-lvstest*

Write to this node to issue "Reset" for Link Layer Validation device. It is needed for TD.7.29, TD.7.31, TD.7.34 and TD.7.35.

/sys/bus/usb/devices/.../ltm_capable

Defined on file *sysfs-bus-usb*

USB 3.0 devices may optionally support Latency Tolerance Messaging (LTM). They indicate their support by setting a bit in the bmAttributes field of their SuperSpeed BOS descriptors. If that bit is set for the device, ltm_capable will read "yes". If the device doesn't support LTM, the file will read "no". The file will be present for all speeds of USB devices, and will always read "no" for USB 1.1 and USB 2.0 devices.

/sys/bus/usb/devices/.../power/usb2_hardware_lpm

Defined on file *sysfs-bus-usb*

If CONFIG_PM is set and a USB 2.0 lpm-capable device is plugged in to a xHCI host which support link PM, it will perform a LPM test; if the test is passed and host supports USB2 hardware LPM (xHCI 1.0 feature), USB2 hardware LPM will be enabled for the device and the USB device directory will contain a file named power/usb2_hardware_lpm. The file holds a string value (enable or disable) indicating whether or not USB2 hardware LPM is enabled for the device. Developer can write y/Y/1 or n/N/0 to the file to enable/disable the feature.

/sys/bus/usb/devices/.../power/usb2_lpm_besl

Defined on file *sysfs-bus-usb*

USB 2.0 devices that support hardware link power management (LPM) L1 sleep state now use a best effort service latency value (BESL) to indicate the best effort to resumption of service to the device after the initiation of the resume event. If the device does not have a preferred besl value then the host can select one instead. This usb2_lpm_besl attribute allows to tune the host selected besl value in order to tune power saving and service latency.

Supported values are 0 - 15. More information on how besl values map to microseconds can be found in USB 2.0 ECN Errata for Link Power Management, section 4.10)

/sys/bus/usb/devices/.../power/usb2_lpm_l1_timeout

Defined on file *sysfs-bus-usb*

USB 2.0 devices may support hardware link power management (LPM) L1 sleep state. The usb2_lpm_l1_timeout attribute allows tuning the timeout for L1 inactivity timer (LPM timer), e.g. needed inactivity time before host requests the device to go to L1 sleep. Useful for power management tuning. Supported values are 0 - 65535 microseconds.

/sys/bus/usb/devices/.../power/usb3_hardware_lpm_u1

Defined on file *sysfs-bus-usb*

If CONFIG_PM is set and a USB 3.0 lpm-capable device is plugged in to a xHCI host which supports link PM, it will check if U1 and U2 exit latencies have been set in the BOS descriptor; if the check is passed and the host supports USB3 hardware LPM, USB3 hardware LPM will be enabled for the device and the USB device directory will contain two files named power/usb3_hardware_lpm_u1 and power/usb3_hardware_lpm_u2. These files hold a string value (enable or disable) indicating whether or not USB3 hardware LPM U1 or U2 is enabled for the device.

/sys/bus/usb/devices/.../rx_lanes

Defined on file *sysfs-bus-usb*

Number of rx lanes the device is using. USB 3.2 adds Dual-lane support, 2 rx and 2 tx lanes over Type-C. Inter-Chip SSIC devices support asymmetric lanes up to 4 lanes per direction. Devices before USB 3.2 are single lane (rx_lanes = 1)

/sys/bus/usb/devices/.../tx_lanes

Defined on file *sysfs-bus-usb*

Number of tx lanes the device is using. USB 3.2 adds Dual-lane support, 2 rx and 2 tx -lanes over Type-C. Inter-Chip SSIC devices support asymmetric lanes up to 4 lanes per direction. Devices before USB 3.2 are single lane (tx_lanes = 1)

/sys/bus/usb/devices/.../typec

Defined on file *sysfs-bus-usb*

Symlink to the USB Type-C partner device. USB Type-C partner represents the component that communicates over the Configuration Channel (CC signal on USB Type-C connectors and cables) with the local port.

/sys/bus/usb/devices/.../u1_timeout

Defined on file *sysfs-bus-usb-lvstest*

Set "U1 timeout" for the downstream port where Link Layer Validation device is connected. Timeout value must be between 0 and 127. It is needed for TD.7.18, TD.7.19, TD.7.20 and TD.7.21.

/sys/bus/usb/devices/.../u2_timeout

Defined on file *sysfs-bus-usb-lvstest*

Set "U2 timeout" for the downstream port where Link Layer Validation device is connected. Timeout value must be between 0 and 127. It is needed for TD.7.18, TD.7.19, TD.7.20 and TD.7.21.

/sys/bus/usb/devices/.../u3_entry

Defined on file *sysfs-bus-usb-lvstest*

Write to this node to issue "U3 entry" for Link Layer Validation device. It is needed for TD.7.35 and TD.7.36.

/sys/bus/usb/devices/.../u3_exit

Defined on file *sysfs-bus-usb-lvstest*

Write to this node to issue "U3 exit" for Link Layer Validation device. It is needed for TD.7.36.

/sys/bus/usb/devices/.../warm_reset

Defined on file *sysfs-bus-usb-lvstest*

Write to this node to issue "Warm Reset" for Link Layer Validation device. It may be needed to properly reset an xHCI 1.1 host port if compliance mode needed to be explicitly enabled.

/sys/bus/usb/devices/<INTERFACE>/authorized

Defined on file *sysfs-bus-usb*

This allows to authorize (1) or deauthorize (0) individual interfaces instead a whole device in contrast to the device authorization. If a deauthorized interface will be authorized so the driver probing must be triggered manually by writing INTERFACE to /sys/bus/usb/drivers_probe This allows to avoid side-effects with drivers that need multiple interfaces.

A deauthorized interface cannot be probed or claimed.

/sys/bus/usb/devices/<INTERFACE>/wireless_status

Defined on file *sysfs-bus-usb*

Some USB devices use a USB receiver dongle to communicate wirelessly with their device using proprietary protocols. This attribute allows user-space to know whether the device is connected to its receiver dongle, and, for example, consider the device to be absent when choosing whether to show the device's battery, show a headset in a list of outputs, or show an on-screen keyboard if the only wireless keyboard is turned off. This attribute is not to be used to replace protocol specific statuses available in WWAN, WLAN/Wi-Fi, Bluetooth, etc. If the device does not use a receiver dongle with a wireless device, then this attribute will not exist.

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/country
/sys/class/bluetooth/hci<addr>/<hid-bus>:<vendor-id>:<product-id>.<num>/country
/sys/class/hidraw/hidraw<num>/device/country

Defined on file *sysfs-driver-hid*

When read, this file returns the hex integer value in ASCII of the device's HID country code (e.g. 21 for US). This file cannot be written.

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/dragging

Defined on file *sysfs-driver-hid-lenovo*

If this setting is enabled, it is possible to do dragging by pressing the trackpoint. This requires press_to_select to be enabled.

Values are 0 or 1.

Applies to Thinkpad USB Keyboard with TrackPoint.

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/fb_update_rate

Defined on file *sysfs-driver-hid-picolcd*

Make it possible to adjust defio refresh rate.

Reading: returns list of available refresh rates (expressed in Hz), the active refresh rate being enclosed in brackets ('[' and ']')

Writing: accepts new refresh rate expressed in integer Hz within permitted rates.

Note: As device can barely do 2 complete refreshes a second it only makes sense to adjust this value if only one or two tiles get changed and it's not appropriate to expect the application to flush its tiny changes explicitly at higher than default rate.

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/fn_lock

Defined on file *sysfs-driver-hid-lenovo*

This setting controls whether Fn Lock is enabled on the keyboard (i.e. if F1 is Mute or F1)

Values are 0 or 1

Applies to ThinkPad Compact (USB|Bluetooth) Keyboard with TrackPoint.

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kone/roccatkone<minor>/actual_dpi

Defined on file *sysfs-driver-hid-roccat-kone*

It is possible to switch the dpi setting of the mouse with the press of a button.

When read, this file returns the raw number of the actual dpi setting reported by the mouse. This number has to be further processed to receive the real dpi value:

VALUE	DPI
1	800
2	1200
3	1600
4	2000
5	2400
6	3200

This file is readonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kone/roccatkone<minor>/actual_profile

Defined on file *sysfs-driver-hid-roccat-kone*

When read, this file returns the number of the actual profile.

This file is readonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kone/roccatkone<minor>/firmware_version

Defined on file *sysfs-driver-hid-roccat-kone*

When read, this file returns the raw integer version number of the firmware reported by the mouse. Using the integer value eases further usage in other programs. To receive the real version number the decimal point has to be shifted 2 positions to the left. E.g. a returned value of 138 means 1.38

This file is readonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kone/roccatkone<minor>/profile[1-5]

Defined on file *sysfs-driver-hid-roccat-kone*

The mouse can store 5 profiles which can be switched by the press of a button. A profile holds information like button mappings, sensitivity, the colors of the 5 leds and light effects.

When read, these files return the respective profile. The returned data is 975 bytes in size.

When written, this file lets one write the respective profile data back to the mouse. The data has to be 975 bytes long.

The mouse will reject invalid data, whereas the profile number stored in the profile doesn't need to fit the number of the store.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kone/roccatkone<minor>/settings

Defined on file *sysfs-driver-hid-roccat-kone*

When read, this file returns the settings stored in the mouse. The size of the data is 36 bytes and holds information like the startup_profile, tcu state and calibration_data.

When written, this file lets write settings back to the mouse. The data has to be 36 bytes long. The mouse will reject invalid data.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kone/roccatkone<minor>/startup_profile

Defined on file *sysfs-driver-hid-roccat-kone*

The integer value of this attribute ranges from 1 to 5.

When read, this attribute returns the number of the profile that's active when the mouse is powered on.

When written, this file sets the number of the startup profile and the mouse activates this profile immediately.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kone/roccatkone<minor>/tcu

Defined on file *sysfs-driver-hid-roccat-kone*

The mouse has a ”Tracking Control Unit” which lets the user calibrate the laser power to fit the mousepad surface. When read, this file returns the current state of the TCU, where 0 means off and 1 means on.

Writing 0 in this file will switch the TCU off.

Writing 1 in this file will start the calibration which takes around 6 seconds to complete and activates the TCU.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kone/roccatkone<minor>/weight

Defined on file *sysfs-driver-hid-roccat-kone*

The mouse can be equipped with one of four supplied weights ranging from 5 to 20 grams which are recognized by the mouse and its value can be read out. When read, this file returns the raw value returned by the mouse which eases further processing in other software.

The values map to the weights as follows:

VALUE	WEIGHT
0	none
1	5g
2	10g
3	15g
4	20g

This file is readonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/operation_mode

Defined on file *sysfs-driver-hid-picolcd*

Make it possible to switch the PicoLCD device between LCD (firmware) and bootloader (flasher) operation modes.

Reading: returns list of available modes, the active mode being enclosed in brackets ('[' and ']')

Writing: causes operation mode switch. Permitted values are the non-active mode names listed when read.

Note: when switching mode the current PicoLCD HID device gets disconnected and reconnects after above delay (see attribute operation_mode_delay for its value).

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/operation_mode_delay

Defined on file *sysfs-driver-hid-picolcd*

Delay PicoLCD waits before restarting in new mode when operation_mode has changed.

Reading/Writing: It is expressed in ms and permitted range is 0..30000ms.

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/press_speed

Defined on file *sysfs-driver-hid-lenovo*

This setting controls how fast the trackpoint needs to be pressed to generate a mouse click if press_to_select is enabled.

Values are decimal integers from 1 (slowest) to 255 (fastest).

Applies to Thinkpad USB Keyboard with TrackPoint.

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/press_to_select

Defined on file *sysfs-driver-hid-lenovo*

This controls if mouse clicks should be generated if the trackpoint is quickly pressed. How fast this press has to be is being controlled by press_speed.

Values are 0 or 1.

Applies to Thinkpad USB Keyboard with TrackPoint.

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/quirks

Defined on file *sysfs-driver-hid-multitouch*

The integer value of this attribute corresponds to the quirks actually in place to handle the device's protocol. When read, this attribute returns the current settings (see MT_QUIRKS_* in hid-multitouch.c). When written this attribute change on the fly the quirks, then the protocol to handle the device.

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/release_to_select

Defined on file *sysfs-driver-hid-lenovo*

For details regarding this setting please refer to http://www.pc.ibm.com/ww/healthycomputing/trkpntb.html Values are 0 or 1. Applies to Thinkpad USB Keyboard with TrackPoint.

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/report_descriptor
/sys/class/bluetooth/hci<addr>/<hid-bus>:<vendor-id>:<product-id>.<num>/report_descriptor
/sys/class/hidraw/hidraw<num>/device/report_descriptor

Defined on file *sysfs-driver-hid*

When read, this file returns the device's raw binary HID report descriptor. This file cannot be written.

Users: HIDAPI library (http://www.signal11.us/oss/hidapi)

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/select_right

Defined on file *sysfs-driver-hid-lenovo*

This setting controls if the mouse click events generated by pressing the trackpoint (if press_to_select is enabled) generate a left or right mouse button click.

Values are 0 or 1.

Applies to Thinkpad USB Keyboard with TrackPoint.

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/sensitivity

Defined on file *sysfs-driver-hid-lenovo*

This file contains the trackpoint sensitivity. Values are decimal integers from 1 (lowest sensitivity) to 255 (highest sensitivity). Applies to Thinkpad USB Keyboard with TrackPoint.

/sys/bus/usb/devices/usbX/avoid_reset_quirk

Defined on file *sysfs-bus-usb*

Most devices have this set to zero.

If the value is 1, enable a USB quirk that prevents this device to use reset.

(read/write)

/sys/bus/usb/devices/usbX/bAlternateSetting

Defined on file *sysfs-bus-usb*

The current interface alternate setting number, in decimal.

See USB specs for its meaning.

/sys/bus/usb/devices/usbX/bConfigurationValue

Defined on file *sysfs-bus-usb*

While a USB device typically have just one configuration setting, some devices support multiple configurations.

This value shows the current configuration, in decimal.

Changing its value will change the device's configuration to another setting.

The number of configurations supported by a device is at:

/sys/bus/usb/devices/usbX/bNumConfigurations

See USB specs for its meaning.

/sys/bus/usb/devices/usbX/bDeviceClass

Defined on file *sysfs-bus-usb*

Class code of the device, in hexadecimal.

See USB specs for its meaning.

/sys/bus/usb/devices/usbX/bDeviceProtocol

Defined on file *sysfs-bus-usb*

Protocol code of the device, in hexadecimal.

See USB specs for its meaning.

/sys/bus/usb/devices/usbX/bDeviceSubClass

Defined on file *sysfs-bus-usb*

Subclass code of the device, in hexadecimal.

See USB specs for its meaning.

/sys/bus/usb/devices/usbX/bInterfaceClass

Defined on file *sysfs-bus-usb*

Class code of the interface, in hexadecimal.

See USB specs for its meaning.

/sys/bus/usb/devices/usbX/bInterfaceNumber

Defined on file *sysfs-bus-usb*

Interface number, in hexadecimal.

See USB specs for its meaning.

/sys/bus/usb/devices/usbX/bInterfaceProtocol

Defined on file *sysfs-bus-usb*

Protocol code of the interface, in hexadecimal.

See USB specs for its meaning.

/sys/bus/usb/devices/usbX/bInterfaceSubClass

Defined on file *sysfs-bus-usb*

Subclass code of the interface, in hexadecimal.

See USB specs for its meaning.

/sys/bus/usb/devices/usbX/bMaxPacketSize0

Defined on file *sysfs-bus-usb*

Maximum endpoint 0 packet size, in decimal.

See USB specs for its meaning.

/sys/bus/usb/devices/usbX/bMaxPower

Defined on file *sysfs-bus-usb*

Maximum power consumption of the active configuration of the device, in miliamperes.

/sys/bus/usb/devices/usbX/bNumConfigurations

Defined on file *sysfs-bus-usb*

Number of the possible configurations of the device, in decimal. The current configuration is controlled via:

/sys/bus/usb/devices/usbX/bConfigurationValue

See USB specs for its meaning.

/sys/bus/usb/devices/usbX/bNumEndpoints

Defined on file *sysfs-bus-usb*

Number of endpoints used on this interface, in hexadecimal.

See USB specs for its meaning.

/sys/bus/usb/devices/usbX/bNumInterfaces

Defined on file *sysfs-bus-usb*

Number of interfaces on this device, in decimal.

/sys/bus/usb/devices/usbX/bcdDevice

Defined on file *sysfs-bus-usb*

The device's release number, in hexadecimal.

See USB specs for its meaning.

/sys/bus/usb/devices/usbX/bmAttributes

Defined on file *sysfs-bus-usb*

Attributes of the current configuration, in hexadecimal.

See USB specs for its meaning.

/sys/bus/usb/devices/usbX/busnum

Defined on file *sysfs-bus-usb*

Number of the bus.

/sys/bus/usb/devices/usbX/configuration

Defined on file *sysfs-bus-usb*

Contents of the string descriptor associated with the current configuration. It may include the firmware version of a device and/or its serial number.

/sys/bus/usb/devices/usbX/descriptors

Defined on file *sysfs-bus-usb*

Contains the interface descriptors, in binary.

/sys/bus/usb/devices/usbX/devnum

Defined on file *sysfs-bus-usb*

USB interface device number, in decimal.

/sys/bus/usb/devices/usbX/devpath

Defined on file *sysfs-bus-usb*

String containing the USB interface device path.

/sys/bus/usb/devices/usbX/devspec

Defined on file *sysfs-bus-usb*

Displays the Device Tree Open Firmware node of the interface.

/sys/bus/usb/devices/usbX/ep_<N>/bEndpointAddress

Defined on file *sysfs-bus-usb*

The address of the endpoint described by this descriptor, in hexadecimal. The endpoint direction on this bitmapped field is also shown at:

/sys/bus/usb/devices/usbX/ep_<N>/direction

See USB specs for its meaning.

/sys/bus/usb/devices/usbX/ep_<N>/bInterval

Defined on file *sysfs-bus-usb*

The interval of the endpoint as described on its descriptor, in hexadecimal. The actual interval depends on the version of the USB. Also shown in time units at */sys/bus/usb/devices/usbX/ep_<N>/interval*.

/sys/bus/usb/devices/usbX/ep_<N>/bLength

Defined on file *sysfs-bus-usb*

Number of bytes of the endpoint descriptor, in hexadecimal.

/sys/bus/usb/devices/usbX/ep_<N>/bmAttributes

Defined on file *sysfs-bus-usb*

Attributes which apply to the endpoint as described on its descriptor, in hexadecimal. The endpoint type on this bitmapped field is also shown at:

/sys/bus/usb/devices/usbX/ep_<N>/type

See USB specs for its meaning.

/sys/bus/usb/devices/usbX/ep_<N>/direction

Defined on file *sysfs-bus-usb*

Direction of the endpoint. Can be:

- both (on control endpoints)
- in
- out

/sys/bus/usb/devices/usbX/ep_<N>/interval

Defined on file *sysfs-bus-usb*

Interval for polling endpoint for data transfers, in milisseconds or microseconds.

/sys/bus/usb/devices/usbX/ep_<N>/type

Defined on file *sysfs-bus-usb*

Descriptor type. Can be:

- Control
- Isoc
- Bulk
- Interrupt
- unknown

/sys/bus/usb/devices/usbX/ep_<N>/wMaxPacketSize

Defined on file *sysfs-bus-usb*

Maximum packet size this endpoint is capable of sending or receiving, in hexadecimal.

/sys/bus/usb/devices/usbX/idProduct

Defined on file *sysfs-bus-usb*

Product ID, in hexadecimal.

/sys/bus/usb/devices/usbX/idVendor

Defined on file *sysfs-bus-usb*

Vendor ID, in hexadecimal.

/sys/bus/usb/devices/usbX/interface_authorized_default

Defined on file *sysfs-bus-usb*

This is used as value that determines if interfaces would be authorized by default. The value can be 1 or 0. It's by default 1.

/sys/bus/usb/devices/usbX/manufacturer

Defined on file *sysfs-bus-usb*

Vendor specific string containing the name of the manufacturer of the device.

/sys/bus/usb/devices/usbX/maxchild

Defined on file *sysfs-bus-usb*

Number of ports of an USB hub

/sys/bus/usb/devices/usbX/persist

Defined on file *sysfs-bus-usb*

Keeps the device even if it gets disconnected.

/sys/bus/usb/devices/usbX/power/active_duration

Defined on file *sysfs-bus-usb*

The total time the device has not been suspended.

/sys/bus/usb/devices/usbX/power/autosuspend

Defined on file *sysfs-bus-usb*

Time in milliseconds for the device to autosuspend. If the value is negative, then autosuspend is prevented.

(read/write)

/sys/bus/usb/devices/usbX/power/connected_duration

Defined on file *sysfs-bus-usb*

The total time (in msec) that the device has been connected.

/sys/bus/usb/devices/usbX/power/level

Defined on file *sysfs-bus-usb*

DESCRIPTION MISSING for /sys/bus/usb/devices/usbX/power/level

/sys/bus/usb/devices/usbX/product

Defined on file *sysfs-bus-usb*

Vendor specific string containing the name of the device's product.

/sys/bus/usb/devices/usbX/speed

Defined on file *sysfs-bus-usb*

Shows the device's max speed, according to the USB version, in Mbps. Can be:

Unknown	speed unknown
1.5	Low speed
15	Full speed
480	High Speed
5000	Super Speed
10000	Super Speed+
20000	Super Speed+ Gen 2x2

/sys/bus/usb/devices/usbX/supports_autosuspend

Defined on file *sysfs-bus-usb*

Returns 1 if the device doesn't support autosuspend. Otherwise, returns 0.

/sys/bus/usb/devices/usbX/urbnum

Defined on file *sysfs-bus-usb*

Number of URBs submitted for the whole device.

/sys/bus/usb/devices/usbX/version

Defined on file *sysfs-bus-usb*

String containing the USB device version, as encoded at the BCD descriptor.

/sys/bus/usb/drivers/.../new_id

Defined on file *sysfs-bus-usb*

Writing a device ID to this file will attempt to dynamically add a new device ID to a USB device driver. This may allow the driver to support more hardware than was included in the driver's static device ID support table at compile time. The format for the device ID is: idVendor idProduct bInterfaceClass RefIdVendor RefIdProduct The vendor ID and device ID fields are required, the rest is optional. The *Ref** tuple can be used to tell the driver to use the same driver_data for the new device as it is used for the reference device. Upon successfully adding an ID, the driver will probe for the device and attempt to bind to it. For example:

```
# echo "8086 10f5" > /sys/bus/usb/drivers/foo/new_id
```

Here add a new device (0458:7045) using driver_data from an already supported device (0458:704c):

```
# echo "0458 7045 0 0458 704c" > /sys/bus/usb/drivers/foo/new_id
```

Reading from this file will list all dynamically added device IDs in the same format, with one entry per line. For example:

```
# cat /sys/bus/usb/drivers/foo/new_id
8086 10f5
dead beef 06
f00d cafe
```

The list will be truncated at PAGE_SIZE bytes due to sysfs restrictions.

/sys/bus/usb/drivers/.../remove_id

Defined on file *sysfs-bus-usb*

Writing a device ID to this file will remove an ID that was dynamically added via the new_id sysfs entry. The format for the device ID is: idVendor idProduct. After successfully removing an ID, the driver will no longer support the device. This is useful to ensure auto probing won't match the driver to the device. For example: # echo "046d c315" > /sys/bus/usb/drivers/foo/remove_id

Reading from this file will list the dynamically added device IDs, exactly like reading from the entry "/sys/bus/usb/drivers/.../new_id"

/sys/bus/vdpa/devices/.../driver_override

Defined on file *sysfs-bus-vdpa*

This file allows the driver for a device to be specified. When specified, only a driver with a name matching the value written to driver_override will have an opportunity to bind to the device. The override is specified by writing a string to the driver_override file (echo vhost-vdpa > driver_override) and may be cleared with an empty string (echo > driver_override). This returns the device to standard matching rules binding. Writing to driver_override does not automatically unbind the device from its current driver or make any attempt to automatically load the specified driver. If no driver with a matching name is currently loaded in the kernel, the device will not bind to any driver. This also allows devices to opt-out of driver binding using a driver_override name such as "none". Only a single driver may be specified in the override, there is no support for parsing delimiters.

/sys/bus/vdpa/driver_probe

Defined on file *sysfs-bus-vdpa*

Writing a device name to this file will cause the kernel binds devices to a compatible driver.

This can be useful when */sys/bus/vdpa/drivers_autoprobe* is disabled.

/sys/bus/vdpa/drivers/.../bind

Defined on file *sysfs-bus-vdpa*

Writing a device name to this file will cause the driver to attempt to bind to the device. This is useful for overriding default bindings.

/sys/bus/vdpa/drivers/.../unbind

Defined on file *sysfs-bus-vdpa*

Writing a device name to this file will cause the driver to attempt to unbind from the device. This may be useful when overriding default bindings.

/sys/bus/vdpa/drivers_autoprobe

Defined on file *sysfs-bus-vdpa*

This file determines whether new devices are immediately bound to a driver after the creation. It initially contains 1, which means the kernel automatically binds devices to a compatible driver immediately after they are created.

Writing "0" to this file disable this feature, any other string enable it.

/sys/bus/vmbus/devices/.../driver_override

Defined on file *sysfs-bus-vmbus*

This file allows the driver for a device to be specified which will override standard static and dynamic ID matching. When specified, only a driver with a name matching the value written to driver_override will have an opportunity to bind to the device. The override is specified by writing a string to the driver_override file (echo uio_hv_generic > driver_override) and may be cleared with an empty string (echo > driver_override). This returns the device to standard matching rules binding. Writing to driver_override does not automatically unbind the device from its current driver or make any attempt to automatically load the specified driver. If no driver with a matching name is currently loaded in the kernel, the device will not bind to any driver. This also allows devices to opt-out of driver binding using a driver_override name such as "none". Only a single driver may be specified in the override, there is no support for parsing delimiters.

/sys/bus/w1/devices/.../alarms

Defined on file *sysfs-driver-w1_therm*

(RW) read or write TH and TL (Temperature High an Low) alarms. Values shall be space separated and in the device range (typical -55 degC to 125 degC), if not values will be trimmed to device min/max capabilities. Values are integer as they are stored in a 8bit register in the device. Lowest value is automatically put to TL. Once set, alarms could be search at master level, refer to /w1/w1-generic for detailed information

Users: any user space application which wants to communicate with w1_term device

/sys/bus/w1/devices/.../conv_time

Defined on file *sysfs-driver-w1_therm*

(RW) Get, set, or measure a temperature conversion time. The setting remains active until a resolution change. Then it is reset to default (datasheet) conversion time for a new resolution.

***read*:**
Actual conversion time in milliseconds.

***write*:**

- **'0':**
  Set the default conversion time from the datasheet.
- **'1':**
  Measure and set the conversion time. Make a single temperature conversion, measure an actual value. Increase it by 20% for temperature range. A new conversion time can be obtained by reading this same attribute.
- **other positive value:**
  Set the conversion time in milliseconds.

Users: An application using the w1_term device

/sys/bus/w1/devices/.../eeprom_cmd

Defined on file *sysfs-driver-w1_therm*

(WO) writing that file will either trigger a save of the device data to its embedded EEPROM, either restore data embedded in device EEPROM. Be aware that devices support limited EEPROM writing cycles (typical 50k)

- 'save': save device RAM to EEPROM
- 'restore': restore EEPROM data in device RAM

Users: any user space application which wants to communicate with w1_term device

/sys/bus/w1/devices/.../ext_power

Defined on file *sysfs-driver-w1_therm*

(RO) return the power status by asking the device

- '0': device parasite powered
- '1': device externally powered
- '-xx': xx is kernel error when reading power status

Users: any user space application which wants to communicate with w1_term device

/sys/bus/w1/devices/.../features

Defined on file *sysfs-driver-w1_therm*

(RW) Control optional driver settings. Bit masks to read/write (bitwise OR):

1	Enable check for conversion success. If byte 6 of scratchpad memory is 0xC after conversion, and temperature reads 85.00 (powerup value) or 127.94 (insufficient power) - return a conversion error.
2	Enable poll for conversion completion. Generate read cycles after the conversion start and wait for 1's. In parasite power mode this feature is not available.

***read*:**
Currently selected features.

***write*:**
Select features.

Users: An application using the w1_term device

/sys/bus/w1/devices/.../resolution

Defined on file *sysfs-driver-w1_therm*

(RW) get or set the device resolution (on supported devices, if not, this entry is not present). Note that the resolution will be changed only in device RAM, so it will be cleared when power is lost. Trigger a 'save' to EEPROM command to keep values after power-on. Read or write are :

- '9..14': device resolution in bit or resolution to set in bit
- '-xx': xx is kernel error when reading the resolution
- Anything else: do nothing

Some DS18B20 clones are fixed in 12-bit resolution, so the actual resolution is read back from the chip and verified. Error is reported if the results differ.

Users: any user space application which wants to communicate with w1_term device

/sys/bus/w1/devices/.../temperature

Defined on file *sysfs-driver-w1_therm*

(RO) return the temperature in 1/1000 degC.

- If a bulk read has been triggered, it will directly return the temperature computed when the bulk read occurred, if available. If not yet available, nothing is returned (a debug kernel message is sent), you should retry later on.
- If no bulk read has been triggered, it will trigger a conversion and send the result. Note that the conversion duration depend on the resolution (if device support this feature). It takes 94ms in 9bits resolution, 750ms for 12bits.

Users: any user space application which wants to communicate with w1_term device

/sys/bus/w1/devices/.../w1_slave

Defined on file *sysfs-driver-w1_therm*

(RW) return the temperature in 1/1000 degC. *read*: return 2 lines with the hexa output data sent on the bus, return the CRC check and temperature in 1/1000 degC *write*:

- '0' : save the 2 or 3 bytes to the device EEPROM (i.e. TH, TL and config register)
- '9..14' : set the device resolution in RAM (if supported)
- Anything else: do nothing

refer to /w1/slaves/w1_therm for detailed information.

Users: any user space application which wants to communicate with w1_term device

/sys/bus/w1/devices/19-<id>/speed

Defined on file *sysfs-driver-w1_ds28e17*

When written, this file sets the I2C speed on the connected DS28E17 chip. When read, it reads the current setting from the DS28E17 chip.

Valid values: 100, 400, 900 [kBaud].

Default 100, can be set by w1_ds28e17.speed= module parameter.

Users: w1_ds28e17 driver

/sys/bus/w1/devices/19-<id>/stretch

Defined on file *sysfs-driver-w1_ds28e17*

When written, this file sets the multiplier used to calculate the busy timeout for I2C operations on the connected DS28E17 chip. When read, returns the current setting. Valid values: 1 to 9.

Default 1, can be set by w1_ds28e17.stretch= module parameter.

Users: w1_ds28e17 driver

/sys/bus/w1/devices/w1_bus_masterXX/therm_bulk_read

Defined on file *sysfs-driver-w1_therm*

(RW) trigger a bulk read conversion. read the status

***read*:**

- **'-1':**
  conversion in progress on at least 1 sensor
- **'1' :**
  conversion complete but at least one sensor value has not been read yet
- **'0' :**
  no bulk operation. Reading temperature will trigger a conversion on each device

***write*:**
'trigger': trigger a bulk read on all supporting devices on the bus

Note that if a bulk read is sent but one sensor is not read immediately, the next access to temperature on this device will return the temperature measured at the time of issue of the bulk read command (not the current temperature).

Users: any user space application which wants to communicate with w1_term device

/sys/bus/wmi/devices/44FADEB1-B204-40F2-8581-394BBDC1B651/firmware_update_reque

Defined on file *sysfs-platform-intel-wmi-sbl-fw-update*

Allow user space entities to trigger update of Slim Bootloader (SBL). This attribute normally has a value of 0 and userspace can signal SBL to update firmware, on next reboot, by writing a value of 1. There are two available states:

- 0 -> Skip firmware update while rebooting
- 1 -> Attempt firmware update on next reboot

/sys/bus/wmi/devices/6932965F-1671-4CEB-B988-D3AB0A901919/dell_privacy_current_sta

Defined on file *sysfs-platform-dell-privacy-wmi*

Allow user space to check current dell privacy device state. Describes the Device State class exposed by BIOS which can be consumed by various applications interested in knowing the Privacy feature capabilities

Attributes:

Microphone:
Identifies the local microphone can be muted by hardware, no applications is available to capture system mic sound

Camera Shutter:
Identifies camera shutter controlled by hardware, which is a micromechanical shutter assembly that is built onto the camera module to block capturing images from outside the laptop

Values:

muted:
Identifies the privacy device is turned off and cannot send stream to OS applications

unmuted:
Identifies the privacy device is turned on, audio or camera driver can get stream from mic and camera module to OS applications

For example to check all supported current privacy device states:

```
# cat /sys/bus/wmi/drivers/dell-privacy/6932965F-1671-4CEB-B988-D3AB0A901919/
↪dell_privacy_current_state
[Microphone] [unmuted]
[Camera Shutter] [unmuted]
```

/sys/bus/wmi/devices/6932965F-1671-4CEB-B988-D3AB0A901919/dell_privacy_supported_t

Defined on file *sysfs-platform-dell-privacy-wmi*

Display which dell hardware level privacy devices are supported "Dell Privacy" is a set of HW, FW, and SW features to enhance Dell's commitment to platform privacy for MIC, Camera, and ePrivacy screens. The supported hardware privacy devices are:

Attributes:

Microphone Mute:
Identifies the local microphone can be muted by hardware, no applications is available to capture system mic sound

Camera Shutter:
Identifies camera shutter controlled by hardware, which is a micromechanical shutter assembly that is built onto the camera module to block capturing images from outside the laptop

Values:

supported:
The privacy device is supported by this system

unsupported:
The privacy device is not supported on this system

For example to check which privacy devices are supported:

```
# cat /sys/bus/wmi/drivers/dell-privacy/6932965F-1671-4CEB-B988-D3AB0A901919/
↪dell_privacy_supported_type
[Microphone Mute] [supported]
```

```
[Camera Shutter] [supported]
[ePrivacy Screen] [unsupported]
```

5.2.32 Symbols under /sys/class

/sys/class/

Defined on file *sysfs-class*

The /sys/class directory will consist of a group of subdirectories describing individual classes of devices in the kernel. The individual directories will consist of either subdirectories, or symlinks to other directories.

All programs that use this directory tree must be able to handle both subdirectories or symlinks in order to work properly.

Users:

udev <linux-hotplug-devel@lists.sourceforge.net>

/sys/class/accel/accel<n>/device/armcp_kernel_ver

Defined on file *sysfs-driver-habanalabs*

Version of the Linux kernel running on the device's CPU. Will be DEPRECATED in Linux kernel version 5.10, and be replaced with cpucp_kernel_ver

/sys/class/accel/accel<n>/device/armcp_ver

Defined on file *sysfs-driver-habanalabs*

Version of the application running on the device's CPU Will be DEPRECATED in Linux kernel version 5.10, and be replaced with cpucp_ver

/sys/class/accel/accel<n>/device/clk_cur_freq_mhz

Defined on file *sysfs-driver-habanalabs*

Displays the current frequency, in MHz, of the device clock. This property is valid only for the Gaudi ASIC family

/sys/class/accel/accel<n>/device/clk_max_freq_mhz

Defined on file *sysfs-driver-habanalabs*

Allows the user to set the maximum clock frequency, in MHz. The device clock might be set to lower value than the maximum. The user should read the clk_cur_freq_mhz to see the actual frequency value of the device clock. This property is valid only for the Gaudi ASIC family

/sys/class/accel/accel<n>/device/cpld_ver

Defined on file *sysfs-driver-habanalabs*

Version of the Device's CPLD F/W

/sys/class/accel/accel<n>/device/cpucp_kernel_ver

Defined on file *sysfs-driver-habanalabs*

Version of the Linux kernel running on the device's CPU

/sys/class/accel/accel<n>/device/cpucp_ver

Defined on file *sysfs-driver-habanalabs*

Version of the application running on the device's CPU

/sys/class/accel/accel<n>/device/device_type

Defined on file *sysfs-driver-habanalabs*

Displays the code name of the device according to its type. The supported values are: "GOYA"

/sys/class/accel/accel<n>/device/eeprom

Defined on file *sysfs-driver-habanalabs*

A binary file attribute that contains the contents of the on-board EEPROM

/sys/class/accel/accel<n>/device/fuse_ver

Defined on file *sysfs-driver-habanalabs*

Displays the device's version from the eFuse

/sys/class/accel/accel<n>/device/fw_os_ver

Defined on file *sysfs-driver-habanalabs*

Version of the firmware OS running on the device's CPU

/sys/class/accel/accel<n>/device/hard_reset

Defined on file *sysfs-driver-habanalabs*

Interface to trigger a hard-reset operation for the device. Hard-reset will reset ALL internal components of the device except for the PCI interface and the internal PLLs

/sys/class/accel/accel<n>/device/hard_reset_cnt

Defined on file *sysfs-driver-habanalabs*

Displays how many times the device have undergone a hard-reset operation since the driver was loaded

/sys/class/accel/accel<n>/device/high_pll

Defined on file *sysfs-driver-habanalabs*

Allows the user to set the maximum clock frequency for MME, TPC and IC when the power management profile is set to "automatic". This property is valid only for the Goya ASIC family

/sys/class/accel/accel<n>/device/ic_clk

Defined on file *sysfs-driver-habanalabs*

Allows the user to set the maximum clock frequency, in Hz, of the Interconnect fabric. Writes to this parameter affect the device only when the power management profile is set to "manual" mode. The device IC clock might be set to lower value than the maximum. The user should read the ic_clk_curr to see the actual frequency value of the IC. This property is valid only for the Goya ASIC family

/sys/class/accel/accel<n>/device/ic_clk_curr

Defined on file *sysfs-driver-habanalabs*

Displays the current clock frequency, in Hz, of the Interconnect fabric. This property is valid only for the Goya ASIC family

/sys/class/accel/accel<n>/device/infineon_ver

Defined on file *sysfs-driver-habanalabs*

Version of the Device's power supply F/W code. Relevant only to GOYA and GAUDI

/sys/class/accel/accel<n>/device/max_power

Defined on file *sysfs-driver-habanalabs*

Allows the user to set the maximum power consumption of the device in milliwatts.

/sys/class/accel/accel<n>/device/mme_clk

Defined on file *sysfs-driver-habanalabs*

Allows the user to set the maximum clock frequency, in Hz, of the MME compute engine. Writes to this parameter affect the device only when the power management profile is set to "manual" mode. The device MME clock might be set to lower value than the maximum. The user should read the mme_clk_curr to see the actual frequency value of the MME. This property is valid only for the Goya ASIC family

/sys/class/accel/accel<n>/device/mme_clk_curr

Defined on file *sysfs-driver-habanalabs*

Displays the current clock frequency, in Hz, of the MME compute engine. This property is valid only for the Goya ASIC family

/sys/class/accel/accel<n>/device/module_id

Defined on file *sysfs-driver-habanalabs*

Displays the device's module id

/sys/class/accel/accel<n>/device/parent_device

Defined on file *sysfs-driver-habanalabs*

Displays the name of the parent device of the accel device

/sys/class/accel/accel<n>/device/pci_addr

Defined on file *sysfs-driver-habanalabs*

Displays the PCI address of the device. This is needed so the user would be able to open a device based on its PCI address

/sys/class/accel/accel<n>/device/pm_mng_profile

Defined on file *sysfs-driver-habanalabs*

Power management profile. Values are "auto", "manual". In "auto" mode, the driver will set the maximum clock frequency to a high value when a user-space process opens the device's file (unless it was already opened by another process). The driver will set the max clock frequency to a low value when there are no user processes that are opened on the device's file. In "manual" mode, the user sets the maximum clock frequency by writing to ic_clk, mme_clk and tpc_clk. This property is valid only for the Goya ASIC family

/sys/class/accel/accel<n>/device/preboot_btl_ver

Defined on file *sysfs-driver-habanalabs*

Version of the device's preboot F/W code

/sys/class/accel/accel<n>/device/security_enabled

Defined on file *sysfs-driver-habanalabs*

Displays the device's security status

/sys/class/accel/accel<n>/device/soft_reset

Defined on file *sysfs-driver-habanalabs*

Interface to trigger a soft-reset operation for the device. Soft-reset will reset only the compute and DMA engines of the device

/sys/class/accel/accel<n>/device/soft_reset_cnt

Defined on file *sysfs-driver-habanalabs*

Displays how many times the device have undergone a soft-reset operation since the driver was loaded

/sys/class/accel/accel<n>/device/status

Defined on file *sysfs-driver-habanalabs*

Status of the card:

- ”operational” - Device is available for work.
- ”in reset” - Device is going through reset, will be available shortly.
- ”disabled” - Device is not usable.
- ”needs reset” - Device is not usable until a hard reset is initiated.
- ”in device creation” - Device is not available yet, as it is still initializing.
- ”in reset after device release” - Device is going through a compute-reset which is executed after a device release (relevant for Gaudi2 only).

/sys/class/accel/accel<n>/device/thermal_ver

Defined on file *sysfs-driver-habanalabs*

Version of the Device's thermal daemon

/sys/class/accel/accel<n>/device/tpc_clk

Defined on file *sysfs-driver-habanalabs*

Allows the user to set the maximum clock frequency, in Hz, of the TPC compute engines. Writes to this parameter affect the device only when the power management profile is set to ”manual” mode. The device TPC clock might be set to lower value than the maximum. The user should read the tpc_clk_curr to see the actual frequency value of the TPC. This property is valid only for Goya ASIC family

/sys/class/accel/accel<n>/device/tpc_clk_curr

Defined on file *sysfs-driver-habanalabs*

Displays the current clock frequency, in Hz, of the TPC compute engines. This property is valid only for the Goya ASIC family

/sys/class/accel/accel<n>/device/uboot_ver

Defined on file *sysfs-driver-habanalabs*

Version of the u-boot running on the device's CPU

/sys/class/accel/accel<n>/device/vrm_ver

Defined on file *sysfs-driver-habanalabs*

Version of the Device's Voltage Regulator Monitor F/W code. N/A to GOYA and GAUDI

/sys/class/ata_*

Defined on file *sysfs-ata*

Provide a place in sysfs for storing the ATA topology of the system. This allows retrieving various information about ATA objects.

Files under /sys/class/ata_port

For each port, a directory ataX is created where X is the ata_port_id of the port. The device parent is the ata host device.

/sys/class/ata_device/devX[.Y].Z/spdn_cnt
/sys/class/ata_device/devX[.Y].Z/gscr
/sys/class/ata_device/devX[.Y].Z/ering
/sys/class/ata_device/devX[.Y].Z/id
/sys/class/ata_device/devX[.Y].Z/pio_mode
/sys/class/ata_device/devX[.Y].Z/xfer_mode
/sys/class/ata_device/devX[.Y].Z/dma_mode
/sys/class/ata_device/devX[.Y].Z/class

Defined on file *sysfs-ata*

spdn_cnt: (RO) Number of times libata decided to lower the
speed of link due to errors.

gscr: (RO) Cached result of the dump of PM GSCR
register. Valid registers are:

0: SATA_PMP_GSCR_PROD_ID, 1: SATA_PMP_GSCR_REV, 2: SATA_PMP_GSCR_PORT_INFO, 32: SATA_PMP_GSCR_ERROR, 33: SATA_PMP_GSCR_ERROR_EN, 64: SATA_PMP_GSCR_FEAT, 96: SATA_PMP_GSCR_FEAT_EN, 130: SATA_PMP_GSCR_SII_GPIO

Only valid if the device is a PM.

ering: (RO) Formatted output of the error ring of the
device.

id: (RO) Cached result of IDENTIFY command, as
described in ATA8 7.16 and 7.17. Only valid if the device is not a PM.

pio_mode: (RO) PIO transfer mode used by the device.
Mostly used by PATA devices.

xfer_mode: (RO) Current transfer mode. Mostly used by
PATA devices.

dma_mode: (RO) DMA transfer mode used by the device.
Mostly used by PATA devices.

class: (RO) Device class. Can be "ata" for disk,
"atapi" for packet device, "pmp" for PM, or "none" if no device was found behind the link.

/sys/class/ata_device/devX[.Y].Z/trim

Defined on file *sysfs-ata*

(RO) Shows the DSM TRIM mode currently used by the device. Valid values are:

unsupported: Drive does not support DSM TRIM

unqueued: Drive supports unqueued DSM TRIM only

queued: Drive supports queued DSM TRIM

forced_unqueued: Drive's queued DSM support is known to
be buggy and only unqueued TRIM commands are sent

/sys/class/ata_link/linkX[.Y]/hw_sata_spd_limit
/sys/class/ata_link/linkX[.Y]/sata_spd_limit
/sys/class/ata_link/linkX[.Y]/sata_spd

Defined on file *sysfs-ata*

hw_sata_spd_limit: (RO) Maximum speed supported by the
connected SATA device.

sata_spd_limit: (RO) Maximum speed imposed by libata.

sata_spd: (RO) Current speed of the link
eg. 1.5, 3 Gbps etc.

Files under /sys/class/ata_device

Behind each link, up to two ata devices are created. The name of the directory is devX[.Y].Z where: - X is ata_port_id of the port where the device is connected, - Y the port of the PM if any, and - Z the device id: for PATA, there is usually 2 devices [0,1], only 1 for SATA.

/sys/class/ata_port/ataX/nr_pmp_links
/sys/class/ata_port/ataX/idle_irq

Defined on file *sysfs-ata*

nr_pmp_links: (RO) If a SATA Port Multiplier (PM) is
connected, the number of links behind it.

idle_irq: (RO) Number of IRQ received by the port while
idle [some ata HBA only].

/sys/class/ata_port/ataX/port_no

Defined on file *sysfs-ata*

(RO) Host local port number. While registering host controller, port numbers are tracked based upon number of ports available on the controller. This attribute is needed by udev for composing persistent links in /dev/disk/by-path.

Files under /sys/class/ata_link

Behind each port, there is a ata_link. If there is a SATA PM in the topology, 15 ata_link objects are created.

If a link is behind a port, the directory name is linkX, where X is ata_port_id of the port. If a link is behind a PM, its name is linkX.Y where X is ata_port_id of the parent port and Y the PM port.

/sys/class/backlight/<backlight>/<ambient light zone>_dim

Defined on file *sysfs-class-backlight*

Control the dim brightness for <ambient light zone> on this <backlight>. Values are between 0 and 127, typically set to 0. Full off when the backlight is disabled. This file will also show the dim brightness level stored for this <ambient light zone>.

The <ambient light zone> is device-driver specific:

For ADP5520 and ADP5501, <ambient light zone> can be:

Ambient light zone	sysfs entry
daylight	/sys/class/backlight/<backlight>/daylight_dim
office	/sys/class/backlight/<backlight>/office_dim
dark	/sys/class/backlight/<backlight>/dark_dim

For ADP8860, <ambient light zone> can be:

Ambient light zone	sysfs entry
l1_daylight	/sys/class/backlight/<backlight>/l1_daylight_dim
l2_office	/sys/class/backlight/<backlight>/l2_office_dim
l3_dark	/sys/class/backlight/<backlight>/l3_dark_dim

For ADP8870, <ambient light zone> can be:

Ambient light zone	sysfs entry
l1_daylight	/sys/class/backlight/<backlight>/l1_daylight_dim
l2_bright	/sys/class/backlight/<backlight>/l2_bright_dim
l3_office	/sys/class/backlight/<backlight>/l3_office_dim
l4_indoor	/sys/class/backlight/<backlight>/l4_indoor_dim
l5_dark	/sys/class/backlight/<backlight>/l5_dark_dim

See also: */sys/class/backlight/<backlight>/ambient_light_zone*.

/sys/class/backlight/<backlight>/<ambient light zone>_max

Defined on file *sysfs-class-backlight*

Control the maximum brightness for <ambient light zone> on this <backlight>. Values are between 0 and 127. This file will also show the brightness level stored for this <ambient light zone>.

The <ambient light zone> is device-driver specific:

For ADP5520 and ADP5501, <ambient light zone> can be:

Ambient light zone	sysfs entry
daylight	/sys/class/backlight/<backlight>/daylight_max
office	/sys/class/backlight/<backlight>/office_max
dark	/sys/class/backlight/<backlight>/dark_max

For ADP8860, <ambient light zone> can be:

Ambient light zone	sysfs entry
l1_daylight	/sys/class/backlight/<backlight>/l1_daylight_max
l2_office	/sys/class/backlight/<backlight>/l2_office_max
l3_dark	/sys/class/backlight/<backlight>/l3_dark_max

For ADP8870, <ambient light zone> can be:

Ambient light zone	sysfs entry
l1_daylight	/sys/class/backlight/<backlight>/l1_daylight_max
l2_bright	/sys/class/backlight/<backlight>/l2_bright_max
l3_office	/sys/class/backlight/<backlight>/l3_office_max
l4_indoor	/sys/class/backlight/<backlight>/l4_indoor_max
l5_dark	/sys/class/backlight/<backlight>/l5_dark_max

See also: */sys/class/backlight/<backlight>/ambient_light_zone*.

/sys/class/backlight/<backlight>/als_channel

Defined on file *sysfs-class-backlight-driver-lm3533*

Get the ALS output channel used as input in ALS-current-control mode (0, 1), where:

0	out_current0 (backlight 0)
1	out_current1 (backlight 1)

/sys/class/backlight/<backlight>/als_en

Defined on file *sysfs-class-backlight-driver-lm3533*

Enable ALS-current-control mode (0, 1).

/sys/class/backlight/<backlight>/ambient_light_level

Defined on file *sysfs-class-backlight*

(RO) Get conversion value of the light sensor.

The value is automatically updated every 80 ms when the light sensor is enabled.

The value range is device-driver specific:

For ADP8870:

It returns integer between 0 (dark) and 8000 (max ambient brightness).

For ADP8860:

It returns a 13-bits integer.

/sys/class/backlight/<backlight>/ambient_light_zone

Defined on file *sysfs-class-backlight*

(RW) Read or write the specific brightness level at which the backlight operates.

The value meaning is device-driver specific:

For ADP8860:

0	Off: Backlight set to 0 mA
1	Level 1: daylight
2	Level 2: bright
3	Level 3: dark

For ADP8870:

0	Off: Backlight set to 0 mA
1	Level 1: daylight
2	Level 2: bright
3	Level 3: office
4	Level 4: indoor
5	Level 5: dark

Writing 0 returns to normal/automatic ambient light level operation.

It can be enabled by writing the value stored in /sys/class/backlight/<backlight>/max_brightness to /sys/class/backlight/<backlight>/brightness.

/sys/class/backlight/<backlight>/bled_mode

Defined on file *sysfs-class-backlight-lm3639*

(WO) Write to the backlight mapping mode. The backlight current can be mapped for either exponential (value "0") or linear mapping modes (default).

/sys/class/backlight/<backlight>/id

Defined on file *sysfs-class-backlight-driver-lm3533*

Get the id of this backlight (0, 1).

/sys/class/backlight/<backlight>/linear

Defined on file *sysfs-class-backlight-driver-lm3533*

Set the brightness-mapping mode (0, 1), where:

0	exponential mode
1	linear mode

/sys/class/backlight/<backlight>/pwm

Defined on file *sysfs-class-backlight-driver-lm3533*

Set the PWM-input control mask (5 bits), where:

bit 5	PWM-input enabled in Zone 4
bit 4	PWM-input enabled in Zone 3
bit 3	PWM-input enabled in Zone 2
bit 2	PWM-input enabled in Zone 1
bit 1	PWM-input enabled in Zone 0
bit 0	PWM-input enabled

/sys/class/backlight/<backlight>/scale

Defined on file *sysfs-class-backlight*

Description of the scale of the brightness curve.

The human eye senses brightness approximately logarithmically, hence linear changes in brightness are perceived as being non-linear. To achieve a linear perception of brightness changes controls like sliders need to apply a logarithmic mapping for backlights with a linear brightness curve.

Possible values of the attribute are:

unknown
: The scale of the brightness curve is unknown.

linear
: The brightness changes linearly with each step. Brightness controls should apply a logarithmic mapping for a linear perception.

non-linear
The brightness changes non-linearly with each step. Brightness controls should use a linear mapping for a linear perception.

/sys/class/bdi/<bdi>/

Defined on file *sysfs-class-bdi*

Provide a place in sysfs for the backing_dev_info object. This allows setting and retrieving various BDI specific variables.

The <bdi> identifier can be either of the following:

MAJOR:MINOR

Device number for block devices, or value of st_dev on non-block filesystems which provide their own BDI, such as NFS and FUSE.

MAJOR:MINOR-fuseblk

Value of st_dev on fuseblk filesystems.

default

The default backing dev, used for non-block device backed filesystems which do not provide their own BDI.

/sys/class/bdi/<bdi>/max_bytes

Defined on file *sysfs-class-bdi*

Allows limiting a particular device to use not more than the given 'max_bytes' of the write-back cache. This is useful in situations where we want to avoid one device taking all or most of the write-back cache. For example in case of an NFS mount that is prone to get stuck, a FUSE mount which cannot be trusted to play fair, or a nbd device.

(read-write)

/sys/class/bdi/<bdi>/max_ratio

Defined on file *sysfs-class-bdi*

Allows limiting a particular device to use not more than the given percentage of the write-back cache. This is useful in situations where we want to avoid one device taking all or most of the write-back cache. For example in case of an NFS mount that is prone to get stuck, or a FUSE mount which cannot be trusted to play fair.

(read-write)

/sys/class/bdi/<bdi>/max_ratio_fine

Defined on file *sysfs-class-bdi*

Allows limiting a particular device to use not more than the given value of the write-back cache. The value is given as part of 1 million. This is useful in situations where we want to avoid one

device taking all or most of the write-back cache. For example in case of an NFS mount that is prone to get stuck, or a FUSE mount which cannot be trusted to play fair.

(read-write)

/sys/class/bdi/<bdi>/min_bytes

Defined on file *sysfs-class-bdi*

Under normal circumstances each device is given a part of the total write-back cache that relates to its current average writeout speed in relation to the other devices.

The 'min_bytes' parameter allows assigning a minimum percentage of the write-back cache to a particular device expressed in bytes. For example, this is useful for providing a minimum QoS.

(read-write)

/sys/class/bdi/<bdi>/min_ratio

Defined on file *sysfs-class-bdi*

Under normal circumstances each device is given a part of the total write-back cache that relates to its current average writeout speed in relation to the other devices.

The 'min_ratio' parameter allows assigning a minimum percentage of the write-back cache to a particular device. For example, this is useful for providing a minimum QoS.

(read-write)

/sys/class/bdi/<bdi>/min_ratio_fine

Defined on file *sysfs-class-bdi*

Under normal circumstances each device is given a part of the total write-back cache that relates to its current average writeout speed in relation to the other devices.

The 'min_ratio_fine' parameter allows assigning a minimum reserve of the write-back cache to a particular device. The value is expressed as part of 1 million. For example, this is useful for providing a minimum QoS.

(read-write)

/sys/class/bdi/<bdi>/read_ahead_kb

Defined on file *sysfs-class-bdi*

Size of the read-ahead window in kilobytes

(read-write)

/sys/class/bdi/<bdi>/stable_pages_required

Defined on file *sysfs-class-bdi*

If set, the backing device requires that all pages comprising a write request must not be changed until writeout is complete.

(read-only)

/sys/class/bdi/<bdi>/strict_limit

Defined on file *sysfs-class-bdi*

Forces per-BDI checks for the share of given device in the write-back cache even before the global background dirty limit is reached. This is useful in situations where the global limit is much higher than affordable for given relatively slow (or untrusted) device. Turning strictlimit on has no visible effect if max_ratio is equal to 100%.

(read-write)

/sys/class/bsr/bsr*/bsr_length

Defined on file *sysfs-class-bsr*

(RO) The length of memory region that can be mapped in bytes.

/sys/class/bsr/bsr*/bsr_size

Defined on file *sysfs-class-bsr*

(RO) Size of the barrier-synchronization register (BSR) register in bytes.

/sys/class/bsr/bsr*/bsr_stride

Defined on file *sysfs-class-bsr*

(RO) The stride or the interval at which the allocated BSR bytes repeat within the mapping.

/sys/class/c2port/

Defined on file *sysfs-c2port*

The */sys/class/c2port/* directory will contain files and directories that will provide a unified interface to the C2 port interface.

/sys/class/c2port/c2portX

Defined on file *sysfs-c2port*

The /sys/class/c2port/c2portX/ directory is related to X-th C2 port into the system. Each directory will contain files to manage and control its C2 port.

/sys/class/c2port/c2portX/access

Defined on file *sysfs-c2port*

The */sys/class/c2port/c2portX/access* file enable the access to the C2 port from the system. No commands can be sent till this entry is set to 0.

/sys/class/c2port/c2portX/dev_id

Defined on file *sysfs-c2port*

The */sys/class/c2port/c2portX/dev_id* file show the device ID of the connected micro.

/sys/class/c2port/c2portX/flash_access

Defined on file *sysfs-c2port*

The */sys/class/c2port/c2portX/flash_access* file enable the access to the on-board flash of the connected micro. No commands can be sent till this entry is set to 0.

/sys/class/c2port/c2portX/flash_block_size

Defined on file *sysfs-c2port*

The */sys/class/c2port/c2portX/flash_block_size* file show the on-board flash block size of the connected micro.

/sys/class/c2port/c2portX/flash_blocks_num

Defined on file *sysfs-c2port*

The */sys/class/c2port/c2portX/flash_blocks_num* file show the on-board flash blocks number of the connected micro.

/sys/class/c2port/c2portX/flash_data

Defined on file *sysfs-c2port*

The */sys/class/c2port/c2portX/flash_data* file export the content of the on-board flash of the connected micro.

/sys/class/c2port/c2portX/flash_erase

Defined on file *sysfs-c2port*

The */sys/class/c2port/c2portX/flash_erase* file execute the ”erase” command on the on-board flash of the connected micro.

/sys/class/c2port/c2portX/reset

Defined on file *sysfs-c2port*

The */sys/class/c2port/c2portX/reset* file execute a ”reset” command on the connected micro.

/sys/class/c2port/c2portX/rev_id

Defined on file *sysfs-c2port*

The */sys/class/c2port/c2portX/rev_id* file show the revision ID of the connected micro.

/sys/class/chromeos/<ec-device-name>/flashinfo

Defined on file *sysfs-class-chromeos*

Show the EC flash information.

/sys/class/chromeos/<ec-device-name>/kb_wake_angle

Defined on file *sysfs-class-chromeos*

Control the keyboard wake lid angle. Values are between 0 and 360. This file will also show the keyboard wake lid angle by querying the hardware.

/sys/class/chromeos/<ec-device-name>/lightbar/brightness

Defined on file *sysfs-class-chromeos-driver-cros-ec-lightbar*

Writing to this file adjusts the overall brightness of the lightbar, separate from any color intensity. The valid range is 0 (off) to 255 (maximum brightness).

/sys/class/chromeos/<ec-device-name>/lightbar/interval_msec

Defined on file *sysfs-class-chromeos-driver-cros-ec-lightbar*

The lightbar is controlled by an embedded controller (EC), which also manages the keyboard, battery charging, fans, and other system hardware. To prevent unprivileged users from interfering with the other EC functions, the rate at which the lightbar control files can be read or written is limited.

Reading this file will return the number of milliseconds that must elapse between accessing any of the lightbar functions through this interface. Going faster will simply block until the necessary interval has lapsed. The interval applies uniformly to all accesses of any kind by any user.

/sys/class/chromeos/<ec-device-name>/lightbar/led_rgb

Defined on file *sysfs-class-chromeos-driver-cros-ec-lightbar*

This allows you to control each LED segment. If the lightbar is already running one of the automatic sequences, you probably won't see anything change because your color setting will be almost immediately replaced. To get useful results, you should stop the lightbar sequence first.

The values written to this file are sets of four integers, indicating LED, RED, GREEN, BLUE. The LED number is 0 to 3 to select a single segment, or 4 to set all four segments to the same value at once. The RED, GREEN, and BLUE numbers should be in the range 0 (off) to 255 (maximum). You can update more than one segment at a time by writing more than one set of four integers.

/sys/class/chromeos/<ec-device-name>/lightbar/program

Defined on file *sysfs-class-chromeos-driver-cros-ec-lightbar*

This allows you to upload and run custom lightbar sequences.

/sys/class/chromeos/<ec-device-name>/lightbar/sequence

Defined on file *sysfs-class-chromeos-driver-cros-ec-lightbar*

The Pixel lightbar has a number of built-in sequences that it displays under various conditions, such as at power on, shut down, or while running. Reading from this file displays the current sequence that the lightbar is displaying. Writing to this file allows you to change the sequence.

/sys/class/chromeos/<ec-device-name>/lightbar/userspace_control

Defined on file *sysfs-class-chromeos-driver-cros-ec-lightbar*

This allows you to take the control of the lightbar. This prevents the kernel from going through its normal sequences.

/sys/class/chromeos/<ec-device-name>/lightbar/version

Defined on file *sysfs-class-chromeos-driver-cros-ec-lightbar*

Show the information about the lightbar version.

/sys/class/chromeos/<ec-device-name>/reboot

Defined on file *sysfs-class-chromeos*

Tell the EC to reboot in various ways. Options are:

- ”cancel”: Cancel a pending reboot.
- ”ro”: Jump to RO without rebooting.
- ”rw”: Jump to RW without rebooting.
- ”cold”: Cold reboot.
- ”disable-jump”: Disable jump until next reboot.
- ”hibernate”: Hibernate the EC.
- ”at-shutdown”: Reboot after an AP shutdown.

/sys/class/chromeos/<ec-device-name>/vbc/vboot_context

Defined on file *sysfs-class-chromeos-driver-cros-ec-vbc*

Read/write the verified boot context data included on a small nvram space on some EC implementations.

/sys/class/chromeos/<ec-device-name>/version

Defined on file *sysfs-class-chromeos*

Show the information about the EC software and hardware.

/sys/class/cxl/<afu>/afu_err_buf

Defined on file *sysfs-class-cxl*

read only AFU Error Buffer contents. The contents of this file are application specific and depends on the AFU being used. Applications interacting with the AFU can use this attribute to know about the current error condition and take appropriate action like logging the event etc.

/sys/class/cxl/<afu>/api_version

Defined on file *sysfs-class-cxl*

read only Decimal value of the current version of the kernel/user API.

Users: https://github.com/ibm-capi/libcxl

/sys/class/cxl/<afu>/api_version_compatible

Defined on file *sysfs-class-cxl*

read only Decimal value of the lowest version of the userspace API this kernel supports.

Users: https://github.com/ibm-capi/libcxl AFU configuration records (eg. /sys/class/cxl/afu0.0/cr0): An AFU may optionally export one or more PCIe like configuration records, known as AFU configuration records, which will show up here (if present).

/sys/class/cxl/<afu>/cr<config num>/class

Defined on file *sysfs-class-cxl*

read only Hexadecimal value of the class code found in this AFU configuration record.

Users: https://github.com/ibm-capi/libcxl

/sys/class/cxl/<afu>/cr<config num>/config

Defined on file *sysfs-class-cxl*

read only This binary file provides raw access to the AFU configuration record. The format is expected to match the either the standard or extended configuration space defined by the PCIe specification.

Users: https://github.com/ibm-capi/libcxl Master contexts (eg. /sys/class/cxl/afu0.0m)

/sys/class/cxl/<afu>/cr<config num>/device

Defined on file *sysfs-class-cxl*

read only Hexadecimal value of the device ID found in this AFU configuration record.

Users: https://github.com/ibm-capi/libcxl

/sys/class/cxl/<afu>/cr<config num>/vendor

Defined on file *sysfs-class-cxl*

read only Hexadecimal value of the vendor ID found in this AFU configuration record.

Users: https://github.com/ibm-capi/libcxl

/sys/class/cxl/<afu>/irqs_max

Defined on file *sysfs-class-cxl*

read/write Decimal value of maximum number of interrupts that can be requested by userspace. The default on probe is the maximum that hardware can support (eg. 2037). Write values will limit userspace applications to that many userspace interrupts. Must be >= irqs_min.

Users: https://github.com/ibm-capi/libcxl

/sys/class/cxl/<afu>/irqs_min

Defined on file *sysfs-class-cxl*

read only Decimal value of the minimum number of interrupts that userspace must request on a CXL_START_WORK ioctl. Userspace may omit the num_interrupts field in the START_WORK IOCTL to get this minimum automatically.

Users: https://github.com/ibm-capi/libcxl

/sys/class/cxl/<afu>/mmio_size

Defined on file *sysfs-class-cxl*

read only Decimal value of the size of the MMIO space that may be mmapped by userspace.

Users: https://github.com/ibm-capi/libcxl

/sys/class/cxl/<afu>/mode

Defined on file *sysfs-class-cxl*

read/write The current mode the AFU is using. Will be one of the modes given in modes_supported. Writing will change the mode provided that no user contexts are attached.

Users: https://github.com/ibm-capi/libcxl

/sys/class/cxl/<afu>/modes_supported

Defined on file *sysfs-class-cxl*

read only List of the modes this AFU supports. One per line. Valid entries are: "dedicated_process" and "afu_directed"

Users: https://github.com/ibm-capi/libcxl

/sys/class/cxl/<afu>/prefault_mode

Defined on file *sysfs-class-cxl*

read/write Set the mode for prefaulting in segments into the segment table when performing the START_WORK ioctl. Only applicable when running under hashed page table mmu. Possible values:

none	No prefaulting (default)
work_element_descriptor	Treat the work element descriptor as an effective address and prefault what it points to.
all	all segments process calling START_WORK maps.

Users: https://github.com/ibm-capi/libcxl

/sys/class/cxl/<afu>/reset

Defined on file *sysfs-class-cxl*

write only Writing 1 here will reset the AFU provided there are not contexts active on the AFU.

Users: https://github.com/ibm-capi/libcxl

/sys/class/cxl/<afu>m/mmio_size

Defined on file *sysfs-class-cxl*

read only Decimal value of the size of the MMIO space that may be mmapped by userspace. This includes all slave contexts space also.

Users: https://github.com/ibm-capi/libcxl

/sys/class/cxl/<afu>m/pp_mmio_len

Defined on file *sysfs-class-cxl*

read only Decimal value of the Per Process MMIO space length.

Users: https://github.com/ibm-capi/libcxl

/sys/class/cxl/<afu>m/pp_mmio_off

Defined on file *sysfs-class-cxl*

read only (not in a guest) Decimal value of the Per Process MMIO space offset.

Users: https://github.com/ibm-capi/libcxl Card info (eg. /sys/class/cxl/card0)

/sys/class/cxl/<card>/base_image

Defined on file *sysfs-class-cxl*

read only (not in a guest) Identifies the revision level of the base image for devices that support loadable PSLs. For FPGAs this field identifies the image contained in the on-adapter flash which is loaded during the initial program load.

Users: https://github.com/ibm-capi/libcxl

/sys/class/cxl/<card>/caia_version

Defined on file *sysfs-class-cxl*

read only Identifies the CAIA Version the card implements.

Users: https://github.com/ibm-capi/libcxl

/sys/class/cxl/<card>/image_loaded

Defined on file *sysfs-class-cxl*

read only (not in a guest) Will return "user" or "factory" depending on the image loaded onto the card.

Users: https://github.com/ibm-capi/libcxl

/sys/class/cxl/<card>/load_image_on_perst

Defined on file *sysfs-class-cxl*

read/write (not in a guest) Valid entries are "none", "user", and "factory". "none" means PERST will not cause image to be loaded to the card. A power cycle is required to load the image. "none" could be useful for debugging because the trace arrays are preserved.

"user" and "factory" means PERST will cause either the user or user or factory image to be loaded. Default is to reload on PERST whichever image the card has loaded.

Users: https://github.com/ibm-capi/libcxl

/sys/class/cxl/<card>/perst_reloads_same_image

Defined on file *sysfs-class-cxl*

read/write (not in a guest) Trust that when an image is reloaded via PERST, it will not have changed.

0	don't trust, the image may be different (default)
1	trust that the image will not change.

Users: https://github.com/ibm-capi/libcxl

/sys/class/cxl/<card>/psl_revision

Defined on file *sysfs-class-cxl*

read only Identifies the revision level of the PSL.

Users: https://github.com/ibm-capi/libcxl

/sys/class/cxl/<card>/psl_timebase_synced

Defined on file *sysfs-class-cxl*

read only Returns 1 if the psl timebase register is synchronized with the core timebase register, 0 otherwise.

Users: https://github.com/ibm-capi/libcxl

/sys/class/cxl/<card>/reset

Defined on file *sysfs-class-cxl*

write only Writing 1 will issue a PERST to card provided there are no contexts active on any one of the card AFUs. This may cause the card to reload the FPGA depending on load_image_on_perst. Writing -1 will do a force PERST irrespective of any active contexts on the card AFUs.

Users: https://github.com/ibm-capi/libcxl

/sys/class/cxl/<card>/tunneled_ops_supported

Defined on file *sysfs-class-cxl*

read only Returns 1 if tunneled operations are supported in capi mode, 0 otherwise.

Users: https://github.com/ibm-capi/libcxl

/sys/class/devfreq-event/event<x>/

Defined on file *sysfs-class-devfreq-event*

Provide a place in sysfs for the devfreq-event objects. This allows accessing various devfreq-event specific variables. The name of devfreq-event object denoted as 'event<x>' which includes the unique number of 'x' for each devfreq-event object.

/sys/class/devfreq-event/event<x>/enable_count

Defined on file *sysfs-class-devfreq-event*

The */sys/class/devfreq-event/event<x>/enable_count* attribute contains the reference count to enable the devfreq-event object. If the device is enabled, the value of attribute is greater than zero.

/sys/class/devfreq-event/event<x>/name

Defined on file *sysfs-class-devfreq-event*

The */sys/class/devfreq-event/event<x>/name* attribute contains the name of the devfreq-event object. This attribute is read-only.

/sys/class/devfreq/.../

Defined on file *sysfs-class-devfreq*

Provide a place in sysfs for the devfreq objects. This allows accessing various devfreq specific variables. The name of devfreq object denoted as ... is same as the name of device using devfreq.

/sys/class/devfreq/.../available_frequencies

Defined on file *sysfs-class-devfreq*

The /sys/class/devfreq/.../available_frequencies shows the available frequencies of the corresponding devfreq object. This is a snapshot of available frequencies and not limited by the min/max frequency restrictions.

/sys/class/devfreq/.../available_governors

Defined on file *sysfs-class-devfreq*

The /sys/class/devfreq/.../available_governors shows currently available governors in the system.

/sys/class/devfreq/.../cur_freq

Defined on file *sysfs-class-devfreq*

The /sys/class/devfreq/.../cur_freq shows the current frequency of the corresponding devfreq object. Same as target_freq when get_cur_freq() is not implemented by devfreq driver.

/sys/class/devfreq/.../governor

Defined on file *sysfs-class-devfreq*

The /sys/class/devfreq/.../governor show or set the name of the governor used by the corresponding devfreq object.

/sys/class/devfreq/.../max_freq

Defined on file *sysfs-class-devfreq*

The /sys/class/devfreq/.../max_freq shows and stores the maximum frequency requested by users. It is 0 if the user does not care. max_freq overrides the frequency requested by governors and min_freq. The max_freq overrides min_freq because max_freq may be used to throttle devices to avoid overheating.

/sys/class/devfreq/.../min_freq

Defined on file *sysfs-class-devfreq*

The /sys/class/devfreq/.../min_freq shows and stores the minimum frequency requested by users. It is 0 if the user does not care. min_freq overrides the frequency requested by governors.

/sys/class/devfreq/.../name

Defined on file *sysfs-class-devfreq*

The /sys/class/devfreq/.../name shows the name of device of the corresponding devfreq object.

/sys/class/devfreq/.../polling_interval

Defined on file *sysfs-class-devfreq*

The /sys/class/devfreq/.../polling_interval shows and sets the requested polling interval of the corresponding devfreq object. The values are represented in ms. If the value is less than 1 jiffy, it is considered to be 0, which means no polling. This value is meaningless if the governor is not polling.

A list of governors that support the node: - simple_ondmenad - tegra_actmon

/sys/class/devfreq/.../target_freq

Defined on file *sysfs-class-devfreq*

The /sys/class/devfreq/.../target_freq shows the next governor predicted target frequency of the corresponding devfreq object.

/sys/class/devfreq/.../timer

Defined on file *sysfs-class-devfreq*

This ABI shows and stores the kind of work timer by users. This work timer is used by devfreq workqueue in order to monitor the device status such as utilization. The user can change the work timer on runtime according to their demand as following:

```
echo deferrable > /sys/class/devfreq/.../timer
echo delayed > /sys/class/devfreq/.../timer
```

A list of governors that support the node: - simple_ondemand

/sys/class/devfreq/.../trans_stat

Defined on file *sysfs-class-devfreq*

This ABI shows or clears the statistics of devfreq behavior on a specific device. It shows the time spent in each state and the number of transitions between states. In order to activate this ABI, the devfreq target device driver should provide the list of available frequencies with its profile. If need to reset the statistics of devfreq behavior on a specific device, enter 0(zero) to 'trans_stat' as following:

```
echo 0 > /sys/class/devfreq/.../trans_stat
```

If the transition table is bigger than PAGE_SIZE, reading this will return an -EFBIG error.

/sys/class/devfreq/.../userspace/set_freq

Defined on file *sysfs-class-devfreq*

The /sys/class/devfreq/.../userspace/set_freq shows and sets the requested frequency for the devfreq object if userspace governor is in effect.

A list of governors that support the node: - userspace

/sys/class/devlink/.../

Defined on file *sysfs-class-devlink*

Provide a place in sysfs for the device link objects in the kernel at any given time. The name of a device link directory, denoted as ... above, is of the form <supplier>--<consumer> where <supplier> is the supplier bus:device name and <consumer> is the consumer bus:device name.

/sys/class/devlink/.../auto_remove_on

Defined on file *sysfs-class-devlink*

This file indicates if the device link will ever be automatically removed by the driver core when the consumer and supplier devices themselves are still present.

This will be one of the following strings:

- 'consumer unbind'
- 'supplier unbind'
- 'never'

'consumer unbind' means the device link will be removed when the consumer's driver is unbound from the consumer device.

'supplier unbind' means the device link will be removed when the supplier's driver is unbound from the supplier device.

'never' means the device link will not be automatically removed when as long as the supplier and consumer devices themselves are still present.

/sys/class/devlink/.../consumer

Defined on file *sysfs-class-devlink*

This file is a symlink to the consumer device's sysfs directory.

/sys/class/devlink/.../runtime_pm

Defined on file *sysfs-class-devlink*

This file indicates if the device link has any impact on the runtime power management behavior of the consumer and supplier devices. For example: Making sure the supplier doesn't enter runtime suspend while the consumer is active.

This will be one of the following strings:

'0'	Does not affect runtime power management
'1'	Affects runtime power management

/sys/class/devlink/.../status

Defined on file *sysfs-class-devlink*

This file indicates the status of the device link. The status of a device link is affected by whether the supplier and consumer devices have been bound to their corresponding drivers. The status of a device link also affects the binding and unbinding of the supplier and consumer devices with their drivers and also affects whether the software state of the supplier device is synced with the hardware state of the supplier device after boot up. See also: sysfs-devices-state_synced.

This will be one of the following strings:

- 'not tracked'
- 'dormant'
- 'available'
- 'consumer probing'
- 'active'
- 'supplier unbinding'
- 'unknown'

'not tracked' means this device link does not track the status and has no impact on the binding, unbinding and syncing the hardware and software device state.

'dormant' means the supplier and the consumer devices have not bound to their driver.

'available' means the supplier has bound to its driver and is available to supply resources to the consumer device.

'consumer probing' means the consumer device is currently trying to bind to its driver.

'active' means the supplier and consumer devices have both bound successfully to their drivers.

'supplier unbinding' means the supplier devices is currently in the process of unbinding from its driver.

'unknown' means the state of the device link is not any of the above. If this is ever the value, there's a bug in the kernel.

/sys/class/devlink/.../supplier

Defined on file *sysfs-class-devlink*

This file is a symlink to the supplier device's sysfs directory.

/sys/class/devlink/.../sync_state_only

Defined on file *sysfs-class-devlink*

This file indicates if the device link is limited to only affecting the syncing of the hardware and software state of the supplier device.

This will be one of the following strings:

'0'	
'1'	Affects runtime power management

'0' means the device link can affect other device behaviors like binding/unbinding, suspend/resume, runtime power management, etc.

'1' means the device link will only affect the syncing of hardware and software state of the supplier device after boot up and doesn't not affect other behaviors of the devices.

/sys/class/extcon/.../

Defined on file *sysfs-class-extcon*

Provide a place in sysfs for the extcon objects. This allows accessing extcon specific variables. The name of extcon object denoted as ... is the name given with extcon_dev_register.

One extcon device denotes a single external connector port. An external connector may have multiple cables attached simultaneously. Many of docks, cradles, and accessory cables have such capability. For example, the 30-pin port of Nuri board (/arch/arm/mach-exynos) may have both HDMI and Charger attached, or analog audio, video, and USB cables attached simultaneously.

If there are cables mutually exclusive with each other, such binary relations may be expressed with extcon_dev's mutually_exclusive array.

/sys/class/extcon/.../cable.X/name

Defined on file *sysfs-class-extcon*

The /sys/class/extcon/.../cable.X/name shows the name of cable "X" (integer between 0 and 31) of an extcon device.

/sys/class/extcon/.../cable.X/state

Defined on file *sysfs-class-extcon*

The /sys/class/extcon/.../cable.X/state shows and stores the state of cable "X" (integer between 0 and 31) of an extcon device. The state value is either 0 (detached) or 1 (attached).

/sys/class/extcon/.../mutually_exclusive/..

Defined on file *sysfs-class-extcon*

Shows the relations of mutually exclusiveness. For example, if the mutually_exclusive array of extcon device is {0x3, 0x5, 0xC, 0x0}, then the output is:

```
# ls mutually_exclusive/
0x3
0x5
0xc
#
```

Note that mutually_exclusive is a sub-directory of the extcon device and the file names under the mutually_exclusive directory show the mutually-exclusive sets, not the contents of the files.

/sys/class/extcon/.../name

Defined on file *sysfs-class-extcon*

The /sys/class/extcon/.../name shows the name of the extcon object. If the extcon object has an optional callback "show_name" defined, the callback will provide the name with this sysfs node.

/sys/class/extcon/.../state

Defined on file *sysfs-class-extcon*

The /sys/class/extcon/.../state shows and stores the cable attach/detach information of the corresponding extcon object. If the extcon object has an optional callback "show_state" defined, the showing function is overridden with the optional callback.

If the default callback for showing function is used, the format is like this:

```
# cat state
USB_OTG=1
HDMI=0
TA=1
EAR_JACK=0
#
```

In this example, the extcon device has USB_OTG and TA cables attached and HDMI and EAR_JACK cables detached.

In order to update the state of an extcon device, enter a hex state number starting with 0x:

```
# echo 0xHEX > state
```

This updates the whole state of the extcon device. Inputs of all the methods are required to meet the mutually_exclusive conditions if they exist.

It is recommended to use this "global" state interface if you need to set the value atomically. The later state interface associated with each cable cannot update multiple cable states of an extcon device simultaneously.

/sys/class/fc/fc_udev_device/appid_store

Defined on file *sysfs-class-fc*

This interface allows an admin to set an FC application identifier in the blkcg associated with a cgroup id. The identifier is typically a UUID that is associated with an application or logical entity such as a virtual machine or container group. The application or logical entity utilizes a block device via the cgroup id. FC adapter drivers may query the identifier and tag FC traffic based on the identifier. FC host and FC fabric entities can utilize the application id and FC traffic tag to identify traffic sources.

The interface expects a string "<cgroupid>:<appid>" where: <cgroupid> is inode of the cgroup in hexadecimal <appid> is user provided string upto 128 characters in length.

If an appid_store is done for a cgroup id that already has an appid set, the new value will override the previous value.

If an admin wants to remove an FC application identifier from a cgroup, an appid_store should be done with the following string: "<cgroupid>:"

/sys/class/fc_host/hostX/statistics/fpin_cn_yyy

Defined on file *sysfs-class-fc_host*

These files contain the number of congestion notification events recorded by the F_Port, reported using fabric performance impact notification (FPIN) event.

/sys/class/fc_host/hostX/statistics/fpin_dn_yyy

Defined on file *sysfs-class-fc_host*

These files contain the number of delivery related errors recorded by the F_Port/Nx_Port, reported using fabric performance impact notification (FPIN) event.

/sys/class/fc_host/hostX/statistics/fpin_li_yyy

Defined on file *sysfs-class-fc_host*

These files contain the number of link integrity error events recorded by the F_Port/Nx_Port, reported using fabric performance impact notification (FPIN) event.

/sys/class/fc_remote_ports/rport-X:Y-Z/statistics/fpin_cn_yyy

Defined on file *sysfs-class-fc_remote_ports*

These files contain the number of congestion notification events recorded by the F_Port/Nx_Port, reported using fabric performance impact notification (FPIN) event.

/sys/class/fc_remote_ports/rport-X:Y-Z/statistics/fpin_dn_yyy

Defined on file *sysfs-class-fc_remote_ports*

These files contain the number of delivery related errors recorded by the F_Port/Nx_Port, reported using fabric performance impact notification (FPIN) event.

/sys/class/fc_remote_ports/rport-X:Y-Z/statistics/fpin_li_yyy

Defined on file *sysfs-class-fc_remote_ports*

These files contain the number of link integrity error events recorded by the F_Port/Nx_Port, reported using fabric performance impact notification (FPIN) event.

/sys/class/firmware-attributes/*/attributes/*/

Defined on file *sysfs-class-firmware-attributes*

A sysfs interface for systems management software to enable configuration capability on supported systems. This directory exposes interfaces for interacting with configuration options.

Unless otherwise specified in an attribute description all attributes are optional and will accept UTF-8 input.

type:
A file that can be read to obtain the type of attribute. This attribute is mandatory.

The following are known types:

- enumeration: a set of pre-defined valid values
- integer: a range of numerical values
- string

HP specific types

- ordered-list - a set of ordered list valid values

All attribute types support the following values:

current_value:
A file that can be read to obtain the current value of the <attr>.

This file can also be written to in order to update the value of a <attr>

This attribute is mandatory.

default_value:
A file that can be read to obtain the default value of the <attr>

display_name:
A file that can be read to obtain a user friendly description of the at <attr>

display_name_language_code:
A file that can be read to obtain the IETF language tag corresponding to the "display_name" of the <attr>

"enumeration"-type specific properties:

possible_values:
A file that can be read to obtain the possible values of the <attr>. Values are separated using semi-colon (;).

"integer"-type specific properties:

min_value:
A file that can be read to obtain the lower bound value of the <attr>

max_value:
A file that can be read to obtain the upper bound value of the <attr>

scalar_increment:
A file that can be read to obtain the scalar value used for increments of current_value this attribute accepts.

”string”-type specific properties:

max_length:
A file that can be read to obtain the maximum length value of the <attr>

min_length:
A file that can be read to obtain the minimum length value of the <attr>

Dell specific class extensions

On Dell systems the following additional attributes are available:

dell_modifier:
A file that can be read to obtain attribute-level dependency rule. It says an attribute X will become read-only or suppressed, if/if-not attribute Y is configured.

modifier rules can be in following format:

```
[ReadOnlyIf:<attribute>=<value>]
[ReadOnlyIfNot:<attribute>=<value>]
[SuppressIf:<attribute>=<value>]
[SuppressIfNot:<attribute>=<value>]
```

For example:

```
AutoOnFri/dell_modifier has value,
        [SuppressIfNot:AutoOn=SelectDays]
```

This means AutoOnFri will be suppressed in BIOS setup if AutoOn attribute is not ”SelectDays” and its value will not be effective through sysfs until this rule is met.

Enumeration attributes also support the following:

dell_value_modifier:
A file that can be read to obtain value-level dependency. This file is similar to dell_modifier but here, an attribute's current value will be forcefully changed based dependent attributes value.

dell_value_modifier rules can be in following format:

```
<value>[ForceIf:<attribute>=<value>]
<value>[ForceIfNot:<attribute>=<value>]
```

For example:

```
LegacyOrom/dell_value_modifier has value:
        Disabled[ForceIf:SecureBoot=Enabled]
```

This means LegacyOrom's current value will be forced to ”Disabled” in BIOS setup if SecureBoot is Enabled and its value will not be effective through sysfs until this rule is met.

HP specific class extensions

On HP systems the following additional attributes are available:

”ordered-list”-type specific properties:

elements:
A file that can be read to obtain the possible list of values of the <attr>. Values are separated using semi-colon (;) and listed according to their priority. An element listed first has the highest priority. Writing the list in a different order to current_value alters the priority order for the particular attribute.

/sys/class/firmware-attributes/*/attributes/Sure_Start/audit_log_entries

Defined on file *sysfs-class-firmware-attributes*

'audit_log_entries' is a read-only file that returns the events in the log.

Audit log entry format

Byte 0-15: Requested Audit Log entry (Each Audit log is 16 bytes) Byte 16-127: Unused

/sys/class/firmware-attributes/*/attributes/Sure_Start/audit_log_entry_count

Defined on file *sysfs-class-firmware-attributes*

'audit_log_entry_count' is a read-only file that returns the number of existing audit log events available to be read. Values are separated using comma. (,)

[No of entries],[log entry size],[Max number of entries supported]

log entry size identifies audit log size for the current BIOS version. The current size is 16 bytes but it can be up to 128 bytes long in future BIOS versions.

/sys/class/firmware-attributes/*/attributes/debug_cmd

Defined on file *sysfs-class-firmware-attributes*

This write only attribute can be used to send debug commands to the BIOS. This should only be used when recommended by the BIOS vendor. Vendors may use it to enable extra debug attributes or BIOS features for testing purposes.

Note that any changes to this attribute requires a reboot for changes to take effect.

HP specific class extensions - Secure Platform Manager (SPM)

/sys/class/firmware-attributes/*/attributes/pending_reboot

Defined on file *sysfs-class-firmware-attributes*

A read-only attribute reads 1 if a reboot is necessary to apply pending BIOS attribute changes. Also, an uevent_KOBJ_CHANGE is generated when it changes to 1.

0	All BIOS attributes setting are current
1	A reboot is necessary to get pending BIOS attribute changes applied

Note, userspace applications need to follow below steps for efficient BIOS management,

1. Check if admin password is set. If yes, follow session method for password management as briefed under authentication section above.
2. Before setting any attribute, check if it has any modifiers or value_modifiers. If yes, incorporate them and then modify attribute.

Drivers may emit a CHANGE uevent when this value changes and userspace may check it again.

/sys/class/firmware-attributes/*/attributes/reset_bios

Defined on file *sysfs-class-firmware-attributes*

This attribute can be used to reset the BIOS Configuration. Specifically, it tells which type of reset BIOS configuration is being requested on the host.

Reading from it returns a list of supported options encoded as:

- 'builtinsafe' (Built in safe configuration profile)
- 'lastknowngood' (Last known good saved configuration profile)
- 'factory' (Default factory settings configuration profile)
- 'custom' (Custom saved configuration profile)

The currently selected option is printed in square brackets as shown below:

```
# echo "factory" > /sys/class/firmware-attributes/*/device/attributes/reset_
↪bios
# cat /sys/class/firmware-attributes/*/device/attributes/reset_bios
builtinsafe lastknowngood [factory] custom
```

Note that any changes to this attribute requires a reboot for changes to take effect.

/sys/class/firmware-attributes/*/attributes/save_settings

Defined on file *sysfs-class-firmware-attributes*

On Lenovo platforms there is a limitation in the number of times an attribute can be saved. This is an architectural limitation and it limits the number of attributes that can be modified to 48. A solution for this is instead of the attribute being saved after every modification, to allow a user to bulk set the attributes, and then trigger a final save. This allows unlimited attributes.

Read the attribute to check what save mode is enabled (single or bulk). E.g: # cat /sys/class/firmware-attributes/thinklmi/attributes/save_settings single

Write the attribute with 'bulk' to enable bulk save mode. Write the attribute with 'single' to enable saving, after every attribute set. The default setting is single mode. E.g: # echo bulk > /sys/class/firmware-attributes/thinklmi/attributes/save_settings

When in bulk mode write 'save' to trigger a save of all currently modified attributes. Note, once a save has been triggered, in bulk mode, attributes can no longer be set and will return a permissions error. This is to prevent users hitting the 48+ save limitation (which requires entering the BIOS to clear the error condition) E.g: # echo save > /sys/class/firmware-attributes/thinklmi/attributes/save_settings

/sys/class/firmware-attributes/*/authentication/

Defined on file *sysfs-class-firmware-attributes*

Devices support various authentication mechanisms which can be exposed as a separate configuration object.

For example a "BIOS Admin" password and "System" Password can be set, reset or cleared using these attributes.

- An "Admin" password is used for preventing modification to the BIOS settings.
- A "System" password is required to boot a machine.

Change in any of these two authentication methods will also generate an uevent KOBJ_CHANGE.

is_enabled:
A file that can be read to obtain a 0/1 flag to see if <attr> authentication is enabled. This attribute is mandatory.

role:
The type of authentication used. This attribute is mandatory.

Known types:

bios-admin:
Representing BIOS administrator password

power-on:
Representing a password required to use the system

system-mgmt:
Representing System Management password. See Lenovo extensions section for details

HDD:
Representing HDD password See Lenovo extensions section for details

NVMe:
Representing NVMe password See Lenovo extensions section for details

mechanism:
The means of authentication. This attribute is mandatory. Only supported type currently is "password".

max_password_length:
A file that can be read to obtain the maximum length of the Password

min_password_length:
A file that can be read to obtain the minimum length of the Password

current_password:
A write only value used for privileged access such as setting attributes when a system or admin password is set or resetting to a new password

This attribute is mandatory when mechanism == "password".

new_password:
A write only value that when used in tandem with current_password will reset a system or admin password.

Note, password management is session specific. If Admin password is set, same password must be written into current_password file (required for password-validation) and must be cleared once the session is over. For example:

```
echo "password" > current_password
echo "disabled" > TouchScreen/current_value
echo "" > current_password
```

Drivers may emit a CHANGE uevent when a password is set or unset userspace may check it again.

On Dell, Lenovo and HP systems, if Admin password is set, then all BIOS attributes require password validation. On Lenovo systems if you change the Admin password the new password is not active until the next boot.

Lenovo specific class extensions

On Lenovo systems the following additional settings are available:

role: system-mgmt This gives the same authority as the bios-admin password to control
security related features. The authorities allocated can be set via the BIOS menu SMP Access Control Policy

role: HDD & NVMe This password is used to unlock access to the drive at boot. Note see
'level' and 'index' extensions below.

lenovo_encoding:
The encoding method that is used. This can be either ”ascii” or ”scancode”. Default is set to ”ascii”

lenovo_kbdlang:
The keyboard language method that is used. This is generally a two char code (e.g. ”us”, ”fr”, ”gr”) and may vary per platform. Default is set to ”us”

level:
Available for HDD and NVMe authentication to set 'user' or 'master' privilege level. If only the user password is configured then this should be used to unlock the drive at boot. If both master and user passwords are set then either can be used. If a master password is set a user password is required. This attribute defaults to 'user' level

index:
Used with HDD and NVME authentication to set the drive index that is being referenced (e.g hdd1, hdd2 etc) This attribute defaults to device 1.

certificate, signature, save_signature:
These attributes are used for certificate based authentication. This is used in conjunction with a signing server as an alternative to password based authentication. The user writes to the attribute(s) with a BASE64 encoded string obtained from the signing server. The attributes can be displayed to check the stored value.

Some usage examples:

Installing a certificate to enable feature:

```
echo "supervisor password" > authentication/Admin/current_password
echo "signed certificate" > authentication/Admin/certificate
```

Updating the installed certificate:

```
echo "signature" > authentication/Admin/signature
echo "signed certificate" > authentication/Admin/certificate
```

Removing the installed certificate:

```
echo "signature" > authentication/Admin/signature
echo "" > authentication/Admin/certificate
```

Changing a BIOS setting:

```
echo "signature" > authentication/Admin/signature
echo "save signature" > authentication/Admin/save_signature
echo Enable > attribute/PasswordBeep/current_value
```

You cannot enable certificate authentication if a supervisor password has not been set. Clearing the certificate results in no bios-admin authentication method being configured allowing anyone to make changes. After any of these operations the system must reboot for the changes to take effect.

certificate_thumbprint:
Read only attribute used to display the MD5, SHA1 and SHA256 thumbprints for the certificate installed in the BIOS.

certificate_to_password:
Write only attribute used to switch from certificate based authentication back to password based. Usage:

```
echo "signature" > authentication/Admin/signature
echo "password" > authentication/Admin/certificate_to_password
```

HP specific class extensions

On HP systems the following additional settings are available:

role: enhanced-bios-auth:
This role is specific to Secure Platform Management (SPM) attribute. It requires configuring an endorsement (kek) and signing certificate (sk).

/sys/class/firmware-attributes/*/authentication/SPM/kek

Defined on file *sysfs-class-firmware-attributes*

'kek' Key-Encryption-Key is a write-only file that can be used to configure the RSA public key that will be used by the BIOS to verify signatures when setting the signing key. When written, the bytes should correspond to the KEK certificate (x509 .DER format containing an OU). The size of the certificate must be less than or equal to 4095 bytes.

/sys/class/firmware-attributes/*/authentication/SPM/sk

Defined on file *sysfs-class-firmware-attributes*

'sk' Signature Key is a write-only file that can be used to configure the RSA public key that will be used by the BIOS to verify signatures when configuring BIOS settings and security features.

When written, the bytes should correspond to the modulus of the public key. The exponent is assumed to be 0x10001.

/sys/class/firmware-attributes/*/authentication/SPM/status

Defined on file *sysfs-class-firmware-attributes*

'status' is a read-only file that returns ASCII text in JSON format reporting the status information.

> "State": "not provisioned | provisioned | provisioning in progress", "Version": "Major.Minor", "Nonce": <16-bit unsigned number display in base 10>, "FeaturesInUse": <16-bit unsigned number display in base 10>, "EndorsementKeyMod": "<256 bytes in base64>", "SigningKeyMod": "<256 bytes in base64>"

/sys/class/firmware/.../cancel

Defined on file *sysfs-class-firmware*

Write-only. For firmware uploads, write a "1" to this file to request that the transfer of firmware data to the lower-level device be canceled. This request will be rejected (EBUSY) if the update cannot be canceled (e.g. a FLASH write is in progress) or (ENODEV) if there is no firmware update in progress.

/sys/class/firmware/.../data

Defined on file *sysfs-class-firmware*

The data sysfs file is used for firmware-fallback and for firmware uploads. Cat a firmware image to this sysfs file after you echo 1 to the loading sysfs file. When the firmware image write is complete, echo 0 to the loading sysfs file. This sequence will signal the completion of the firmware write and signal the lower-level driver that the firmware data is available.

/sys/class/firmware/.../error

Defined on file *sysfs-class-firmware*

Read-only. Returns a string describing a failed firmware upload. This string will be in the form of <STATUS>:<ERROR>, where <STATUS> will be one of the status strings described for the status sysfs file and <ERROR> will be one of the following: "hw-error", "timeout", "user-abort", "device-busy", "invalid-file-size", "read-write-error", "flash-wearout". The error sysfs file is only meaningful when the current firmware upload status is "idle". If this file is read while a firmware transfer is in progress, then the read will fail with EBUSY.

/sys/class/firmware/.../loading

Defined on file *sysfs-class-firmware*

The loading sysfs file is used for both firmware-fallback and for firmware uploads. Echo 1 onto the loading file to indicate you are writing a firmware file to the data sysfs node. Echo -1 onto this file to abort the data write or echo 0 onto this file to indicate that the write is complete.

For firmware uploads, the zero value also triggers the transfer of the firmware data to the lower-level device driver.

/sys/class/firmware/.../remaining_size

Defined on file *sysfs-class-firmware*

Read-only. For firmware upload, this file contains the size of the firmware data that remains to be transferred to the lower-level device driver. The size value is initialized to the full size of the firmware image that was previously written to the data sysfs file. This value is periodically updated during the "transferring" phase of the firmware upload. Format: "%u".

/sys/class/firmware/.../status

Defined on file *sysfs-class-firmware*

Read-only. Returns a string describing the current status of a firmware upload. The string will be one of the following: idle, "receiving", "preparing", "transferring", "programming".

/sys/class/firmware/.../timeout

Defined on file *sysfs-class-firmware*

This file supports the timeout mechanism for firmware fallback. This file has no affect on firmware uploads. For more information on timeouts please see the documentation for firmware fallback.

/sys/class/fpga_bridge/<bridge>/name

Defined on file *sysfs-class-fpga-bridge*

Name of low level FPGA bridge driver.

/sys/class/fpga_bridge/<bridge>/state

Defined on file *sysfs-class-fpga-bridge*

Show bridge state as "enabled" or "disabled"

/sys/class/fpga_manager/<fpga>/name

Defined on file *sysfs-class-fpga-manager*

Name of low level fpga manager driver.

/sys/class/fpga_manager/<fpga>/state

Defined on file *sysfs-class-fpga-manager*

Read fpga manager state as a string. The intent is to provide enough detail that if something goes wrong during FPGA programming (something that the driver can't fix) then userspace can

know, i.e. if the firmware request fails, that could be due to not being able to find the firmware file.

This is a superset of FPGA states and fpga manager driver states. The fpga manager driver is walking through these steps to get the FPGA into a known operating state. It's a sequence, though some steps may get skipped. Valid FPGA states will vary by manufacturer; this is a superset.

- unknown = can't determine state
- power off = FPGA power is off
- power up = FPGA reports power is up
- reset = FPGA held in reset state
- firmware request = firmware class request in progress
- firmware request error = firmware request failed
- write init = preparing FPGA for programming
- write init error = Error while preparing FPGA for programming
- write = FPGA ready to receive image data
- write error = Error while programming
- write complete = Doing post programming steps
- write complete error = Error while doing post programming
- operating = FPGA is programmed and operating

/sys/class/fpga_manager/<fpga>/status

Defined on file *sysfs-class-fpga-manager*

Read fpga manager status as a string. If FPGA programming operation fails, it could be caused by crc error or incompatible bitstream image. The intent of this interface is to provide more detailed information for FPGA programming errors to userspace. This is a list of strings for the supported status.

- **reconfig operation error - invalid operations detected by**
  reconfiguration hardware. e.g. start reconfiguration with errors not cleared
- **reconfig CRC error - CRC error detected by**
  reconfiguration hardware.
- **reconfig incompatible image - reconfiguration image is**
  incompatible with hardware
- **reconfig IP protocol error - protocol errors detected by**
  reconfiguration hardware
- **reconfig fifo overflow error - FIFO overflow detected by**
  reconfiguration hardware

/sys/class/fpga_region/<region>/compat_id

Defined on file *sysfs-class-fpga-region*

FPGA region id for compatibility check, e.g. compatibility of the FPGA reconfiguration hardware and image. This value is defined or calculated by the layer that is creating the FPGA region. This interface returns the compat_id value or just error code -ENOENT in case compat_id is not used.

/sys/class/genwqe/genwqe<n>_card/appid

Defined on file *sysfs-driver-genwqe*

Identifies the currently active card application e.g. 'GZIP' for compression/decompression.

/sys/class/genwqe/genwqe<n>_card/base_clock

Defined on file *sysfs-driver-genwqe*

Base clock frequency of the card.

/sys/class/genwqe/genwqe<n>_card/curr_bitstream

Defined on file *sysfs-driver-genwqe*

Currently active bitstream. 1 is default, 0 is backup.

/sys/class/genwqe/genwqe<n>_card/device/sriov_numvfs

Defined on file *sysfs-driver-genwqe*

Enable VFs (1..15):

```
sudo sh -c 'echo 15 > \
  /sys/bus/pci/devices/0000\:1b\:00.0/sriov_numvfs'
```

Disable VFs:

```
Write a 0 into the same sysfs entry.
```

/sys/class/genwqe/genwqe<n>_card/freerunning_timer

Defined on file *sysfs-driver-genwqe*

Interface to read the cards free running timer. Used for performance and utilization measurements.

/sys/class/genwqe/genwqe<n>_card/next_bitstream

Defined on file *sysfs-driver-genwqe*

Interface to set the next bitstream to be used.

/sys/class/genwqe/genwqe<n>_card/queue_working_time

Defined on file *sysfs-driver-genwqe*

Interface to read queue working time. Used for performance and utilization measurements.

/sys/class/genwqe/genwqe<n>_card/reload_bitstream

Defined on file *sysfs-driver-genwqe*

Interface to trigger a PCIe card reset to reload the bitstream.

```
sudo sh -c 'echo 1 > \
  /sys/class/genwqe/genwqe0_card/reload_bitstream'
```

If successfully, the card will come back with the bitstream set on 'next_bitstream'.

/sys/class/genwqe/genwqe<n>_card/state

Defined on file *sysfs-driver-genwqe*

State of the card: ”unused”, ”used”, ”error”.

/sys/class/genwqe/genwqe<n>_card/tempsens

Defined on file *sysfs-driver-genwqe*

Interface to read the cards temperature sense register.

/sys/class/genwqe/genwqe<n>_card/type

Defined on file *sysfs-driver-genwqe*

Type of the card e.g. 'GenWQE5-A7'.

/sys/class/genwqe/genwqe<n>_card/version

Defined on file *sysfs-driver-genwqe*

Unique bitstream identification e.g. '0000000330336283.00000000475a4950'.

/sys/class/gnss/gnss<N>/type

Defined on file *sysfs-class-gnss*

The GNSS receiver type. The currently identified types reflect the protocol(s) supported by the receiver:

”NMEA”	NMEA 0183
”SiRF”	SiRF Binary
”UBX”	UBX

Note that also non-"NMEA" type receivers typically support a subset of NMEA 0183 with vendor extensions (e.g. to allow switching to a vendor protocol).

/sys/class/hwmon/hwmonX/cpuY_vid

Defined on file *sysfs-class-hwmon*

CPU core reference voltage.

Unit: millivolt

RO

Not always correct.

/sys/class/hwmon/hwmonX/currY_average

Defined on file *sysfs-class-hwmon*

Average current use

Unit: milliampere

RO

/sys/class/hwmon/hwmonX/currY_crit

Defined on file *sysfs-class-hwmon*

Current critical high value.

Unit: milliampere

RW

/sys/class/hwmon/hwmonX/currY_enable

Defined on file *sysfs-class-hwmon*

Enable or disable the sensors.

When disabled the sensor read will return -ENODATA.

- 1: Enable
- 0: Disable

RW

/sys/class/hwmon/hwmonX/currY_highest

Defined on file *sysfs-class-hwmon*

Historical maximum current Unit: milliampere RO

/sys/class/hwmon/hwmonX/currY_input

Defined on file *sysfs-class-hwmon*

Current input value

Unit: milliampere

RO

/sys/class/hwmon/hwmonX/currY_lcrit

Defined on file *sysfs-class-hwmon*

Current critical low value

Unit: milliampere

RW

/sys/class/hwmon/hwmonX/currY_lowest

Defined on file *sysfs-class-hwmon*

Historical minimum current

Unit: milliampere

RO

/sys/class/hwmon/hwmonX/currY_max

Defined on file *sysfs-class-hwmon*

Current max value

Unit: milliampere

RW

/sys/class/hwmon/hwmonX/currY_min

Defined on file *sysfs-class-hwmon*

Current min value.

Unit: milliampere

RW

/sys/class/hwmon/hwmonX/currY_rated_max

Defined on file *sysfs-class-hwmon*

Maximum rated current.

Unit: milliampere

RO

/sys/class/hwmon/hwmonX/currY_rated_min

Defined on file *sysfs-class-hwmon*

Minimum rated current.

Unit: milliampere

RO

/sys/class/hwmon/hwmonX/currY_reset_history

Defined on file *sysfs-class-hwmon*

Reset currX_lowest and currX_highest

WO

/sys/class/hwmon/hwmonX/curr_reset_history

Defined on file *sysfs-class-hwmon*

Reset currX_lowest and currX_highest for all sensors

WO

/sys/class/hwmon/hwmonX/device/pec

Defined on file *sysfs-class-hwmon*

PEC support on I2C devices

- 0, off, n: disable
- 1, on, y: enable

RW

/sys/class/hwmon/hwmonX/energyY_enable

Defined on file *sysfs-class-hwmon*

Enable or disable the sensors.

When disabled the sensor read will return -ENODATA.

- 1: Enable
- 0: Disable

RW

/sys/class/hwmon/hwmonX/energyY_input

Defined on file *sysfs-class-hwmon*

Cumulative energy use

Unit: microJoule

RO

/sys/class/hwmon/hwmonX/fanY_div

Defined on file *sysfs-class-hwmon*

Fan divisor.

Integer value in powers of two (1, 2, 4, 8, 16, 32, 64, 128).

RW

Some chips only support values 1, 2, 4 and 8. Note that this is actually an internal clock divisor, which affects the measurable speed range, not the read value.

/sys/class/hwmon/hwmonX/fanY_enable

Defined on file *sysfs-class-hwmon*

Enable or disable the sensors.

When disabled the sensor read will return -ENODATA.

- 1: Enable
- 0: Disable

RW

/sys/class/hwmon/hwmonX/fanY_fault

Defined on file *sysfs-class-hwmon*

Reports if a fan has reported failure.

- 1: Failed
- 0: Ok

RO

/sys/class/hwmon/hwmonX/fanY_input

Defined on file *sysfs-class-hwmon*

Fan input value.

Unit: revolution/min (RPM)

RO

/sys/class/hwmon/hwmonX/fanY_label

Defined on file *sysfs-class-hwmon*

Suggested fan channel label.

Text string

Should only be created if the driver has hints about what this fan channel is being used for, and user-space doesn't. In all other cases, the label is provided by user-space.

RO

/sys/class/hwmon/hwmonX/fanY_max

Defined on file *sysfs-class-hwmon*

Fan maximum value

Unit: revolution/min (RPM)

Only rarely supported by the hardware. RW

/sys/class/hwmon/hwmonX/fanY_min

Defined on file *sysfs-class-hwmon*

Fan minimum value

Unit: revolution/min (RPM)

RW

/sys/class/hwmon/hwmonX/fanY_pulses

Defined on file *sysfs-class-hwmon*

Number of tachometer pulses per fan revolution.

Integer value, typically between 1 and 4.

RW

This value is a characteristic of the fan connected to the device's input, so it has to be set in accordance with the fan model.

Should only be created if the chip has a register to configure the number of pulses. In the absence of such a register (and thus attribute) the value assumed by all devices is 2 pulses per fan revolution.

/sys/class/hwmon/hwmonX/fanY_target

Defined on file *sysfs-class-hwmon*

Desired fan speed

Unit: revolution/min (RPM)

RW

Only makes sense if the chip supports closed-loop fan speed control based on the measured fan speed.

/sys/class/hwmon/hwmonX/humidityY_alarm

Defined on file *sysfs-class-hwmon*

Humidity limit detection

- 0: OK
- 1: Humidity limit has been reached

RO

/sys/class/hwmon/hwmonX/humidityY_enable

Defined on file *sysfs-class-hwmon*

Enable or disable the sensors

When disabled the sensor read will return -ENODATA.

- 1: Enable
- 0: Disable

RW

/sys/class/hwmon/hwmonX/humidityY_fault

Defined on file *sysfs-class-hwmon*

Reports a humidity sensor failure.

- 1: Failed
- 0: Ok

RO

/sys/class/hwmon/hwmonX/humidityY_input

Defined on file *sysfs-class-hwmon*

Humidity

Unit: milli-percent (per cent mille, pcm)

RO

/sys/class/hwmon/hwmonX/humidityY_label

Defined on file *sysfs-class-hwmon*

Suggested humidity channel label.

Text string

Should only be created if the driver has hints about what this humidity channel is being used for, and user-space doesn't. In all other cases, the label is provided by user-space.

RO

/sys/class/hwmon/hwmonX/humidityY_max

Defined on file *sysfs-class-hwmon*

Humidity max value.

Unit: milli-percent (per cent mille, pcm)

RW

/sys/class/hwmon/hwmonX/humidityY_max_hyst

Defined on file *sysfs-class-hwmon*

Humidity hysteresis value for max limit.

Unit: milli-percent (per cent mille, pcm)

Must be reported as an absolute humidity, NOT a delta from the max value.

RW

/sys/class/hwmon/hwmonX/humidityY_min

Defined on file *sysfs-class-hwmon*

Humidity min value.

Unit: milli-percent (per cent mille, pcm)

RW

/sys/class/hwmon/hwmonX/humidityY_min_hyst

Defined on file *sysfs-class-hwmon*

Humidity hysteresis value for min limit.

Unit: milli-percent (per cent mille, pcm)

Must be reported as an absolute humidity, NOT a delta from the min value.

RW

/sys/class/hwmon/hwmonX/humidityY_rated_max

Defined on file *sysfs-class-hwmon*

Maximum rated humidity.

Unit: milli-percent (per cent mille, pcm)

RO

/sys/class/hwmon/hwmonX/humidityY_rated_min

Defined on file *sysfs-class-hwmon*

Minimum rated humidity.

Unit: milli-percent (per cent mille, pcm)

RO

/sys/class/hwmon/hwmonX/inY_average

Defined on file *sysfs-class-hwmon*

Average voltage

Unit: millivolt

RO

/sys/class/hwmon/hwmonX/inY_crit

Defined on file *sysfs-class-hwmon*

Voltage critical max value.

Unit: millivolt

RW

If voltage reaches or exceeds this limit, the system may take drastic action such as power down or reset. At the very least, it should report a fault.

/sys/class/hwmon/hwmonX/inY_enable

Defined on file *sysfs-class-hwmon*

Enable or disable the sensors.

When disabled the sensor read will return -ENODATA.

- 1: Enable
- 0: Disable

RW

/sys/class/hwmon/hwmonX/inY_highest

Defined on file *sysfs-class-hwmon*

Historical maximum voltage

Unit: millivolt

RO

/sys/class/hwmon/hwmonX/inY_input

Defined on file *sysfs-class-hwmon*

Voltage input value.

Unit: millivolt

RO

Voltage measured on the chip pin.

Actual voltage depends on the scaling resistors on the motherboard, as recommended in the chip datasheet.

This varies by chip and by motherboard. Because of this variation, values are generally NOT scaled by the chip driver, and must be done by the application. However, some drivers (notably lm87 and via686a) do scale, because of internal resistors built into a chip. These drivers will output the actual voltage. Rule of thumb: drivers should report the voltage values at the "pins" of the chip.

/sys/class/hwmon/hwmonX/inY_label

Defined on file *sysfs-class-hwmon*

Suggested voltage channel label.

Text string

Should only be created if the driver has hints about what this voltage channel is being used for, and user-space doesn't. In all other cases, the label is provided by user-space.

RO

/sys/class/hwmon/hwmonX/inY_lcrit

Defined on file *sysfs-class-hwmon*

Voltage critical min value.

Unit: millivolt

RW

If voltage drops to or below this limit, the system may take drastic action such as power down or reset. At the very least, it should report a fault.

/sys/class/hwmon/hwmonX/inY_lowest

Defined on file *sysfs-class-hwmon*

Historical minimum voltage

Unit: millivolt

RO

/sys/class/hwmon/hwmonX/inY_max

Defined on file *sysfs-class-hwmon*

Voltage max value.

Unit: millivolt

RW

/sys/class/hwmon/hwmonX/inY_min

Defined on file *sysfs-class-hwmon*

Voltage min value.

Unit: millivolt

RW

/sys/class/hwmon/hwmonX/inY_rated_max

Defined on file *sysfs-class-hwmon*

Maximum rated voltage.

Unit: millivolt

RO

/sys/class/hwmon/hwmonX/inY_rated_min

Defined on file *sysfs-class-hwmon*

Minimum rated voltage.

Unit: millivolt

RO

/sys/class/hwmon/hwmonX/inY_reset_history

Defined on file *sysfs-class-hwmon*

Reset inX_lowest and inX_highest

WO

/sys/class/hwmon/hwmonX/in_reset_history

Defined on file *sysfs-class-hwmon*

Reset inX_lowest and inX_highest for all sensors

WO

/sys/class/hwmon/hwmonX/intrusionY_alarm

Defined on file *sysfs-class-hwmon*

Chassis intrusion detection

- 0: OK
- 1: intrusion detected

RW

Contrary to regular alarm flags which clear themselves automatically when read, this one sticks until cleared by the user. This is done by writing 0 to the file. Writing other values is unsupported.

/sys/class/hwmon/hwmonX/intrusionY_beep

Defined on file *sysfs-class-hwmon*

Chassis intrusion beep

- 0: disable
- 1: enable

RW

/sys/class/hwmon/hwmonX/label

Defined on file *sysfs-class-hwmon*

A descriptive label that allows to uniquely identify a device within the system. The contents of the label are free-form.

RO

/sys/class/hwmon/hwmonX/name

Defined on file *sysfs-class-hwmon*

The chip name. This should be a short, lowercase string, not containing whitespace, dashes, or the wildcard character '*'. This attribute represents the chip name. It is the only mandatory attribute. I2C devices get this attribute created automatically.

RO

/sys/class/hwmon/hwmonX/powerY_accuracy

Defined on file *sysfs-class-hwmon*

Accuracy of the power meter.

Unit: Percent

RO

/sys/class/hwmon/hwmonX/powerY_average

Defined on file *sysfs-class-hwmon*

Average power use

Unit: microWatt

RO

/sys/class/hwmon/hwmonX/powerY_average_highest

Defined on file *sysfs-class-hwmon*

Historical average maximum power use

Unit: microWatt

RO

/sys/class/hwmon/hwmonX/powerY_average_interval

Defined on file *sysfs-class-hwmon*

Power use averaging interval. A poll notification is sent to this file if the hardware changes the averaging interval.

Unit: milliseconds

RW

/sys/class/hwmon/hwmonX/powerY_average_interval_max

Defined on file *sysfs-class-hwmon*

Maximum power use averaging interval

Unit: milliseconds

RO

/sys/class/hwmon/hwmonX/powerY_average_interval_min

Defined on file *sysfs-class-hwmon*

Minimum power use averaging interval

Unit: milliseconds

RO

/sys/class/hwmon/hwmonX/powerY_average_lowest

Defined on file *sysfs-class-hwmon*

Historical average minimum power use

Unit: microWatt

RO

/sys/class/hwmon/hwmonX/powerY_average_max

Defined on file *sysfs-class-hwmon*

A poll notification is sent to *powerY_average* when power use rises above this value.

Unit: microWatt

RW

/sys/class/hwmon/hwmonX/powerY_average_min

Defined on file *sysfs-class-hwmon*

A poll notification is sent to *powerY_average* when power use sinks below this value.

Unit: microWatt

RW

/sys/class/hwmon/hwmonX/powerY_cap

Defined on file *sysfs-class-hwmon*

If power use rises above this limit, the system should take action to reduce power use. A poll notification is sent to this file if the cap is changed by the hardware. The *_*cap* files only appear if the cap is known to be enforced by hardware.

Unit: microWatt

RW

/sys/class/hwmon/hwmonX/powerY_cap_hyst

Defined on file *sysfs-class-hwmon*

Margin of hysteresis built around capping and notification.

Unit: microWatt

RW

/sys/class/hwmon/hwmonX/powerY_cap_max

Defined on file *sysfs-class-hwmon*

Maximum cap that can be set.

Unit: microWatt

RO

/sys/class/hwmon/hwmonX/powerY_cap_min

Defined on file *sysfs-class-hwmon*

Minimum cap that can be set.

Unit: microWatt

RO

/sys/class/hwmon/hwmonX/powerY_crit

Defined on file *sysfs-class-hwmon*

Critical maximum power.

If power rises to or above this limit, the system is expected take drastic action to reduce power consumption, such as a system shutdown or a forced powerdown of some devices.

Unit: microWatt

RW

/sys/class/hwmon/hwmonX/powerY_enable

Defined on file *sysfs-class-hwmon*

Enable or disable the sensors.

When disabled the sensor read will return -ENODATA.

- 1: Enable
- 0: Disable

RW

/sys/class/hwmon/hwmonX/powerY_input

Defined on file *sysfs-class-hwmon*

Instantaneous power use

Unit: microWatt

RO

/sys/class/hwmon/hwmonX/powerY_input_highest

Defined on file *sysfs-class-hwmon*

Historical maximum power use

Unit: microWatt

RO

/sys/class/hwmon/hwmonX/powerY_input_lowest

Defined on file *sysfs-class-hwmon*

Historical minimum power use

Unit: microWatt

RO

/sys/class/hwmon/hwmonX/powerY_max

Defined on file *sysfs-class-hwmon*

Maximum power.

Unit: microWatt

RW

/sys/class/hwmon/hwmonX/powerY_rated_max

Defined on file *sysfs-class-hwmon*

Maximum rated power.

Unit: microWatt

RO

/sys/class/hwmon/hwmonX/powerY_rated_min

Defined on file *sysfs-class-hwmon*

Minimum rated power.

Unit: microWatt

RO

/sys/class/hwmon/hwmonX/powerY_reset_history

Defined on file *sysfs-class-hwmon*

Reset input_highest, input_lowest, average_highest and average_lowest.

WO

/sys/class/hwmon/hwmonX/pwmY

Defined on file *sysfs-class-hwmon*

Pulse width modulation fan control.

Integer value in the range 0 to 255

RW

255 is max or 100%.

/sys/class/hwmon/hwmonX/pwmY_auto_channels_temp

Defined on file *sysfs-class-hwmon*

Select which temperature channels affect this PWM output in auto mode.

Bitfield, 1 is temp1, 2 is temp2, 4 is temp3 etc... Which values are possible depend on the chip used.

RW

/sys/class/hwmon/hwmonX/pwmY_auto_pointZ_pwm
/sys/class/hwmon/hwmonX/pwmY_auto_pointZ_temp
/sys/class/hwmon/hwmonX/pwmY_auto_pointZ_temp_hyst

Defined on file *sysfs-class-hwmon*

Define the PWM vs temperature curve.

Number of trip points is chip-dependent. Use this for chips which associate trip points to PWM output channels.

RW

/sys/class/hwmon/hwmonX/pwmY_enable

Defined on file *sysfs-class-hwmon*

Fan speed control method:

- 0: no fan speed control (i.e. fan at full speed)
- 1: manual fan speed control enabled (using *pwmY*)
- 2+: automatic fan speed control enabled

Check individual chip documentation files for automatic mode details.

RW

/sys/class/hwmon/hwmonX/pwmY_freq

Defined on file *sysfs-class-hwmon*

Base PWM frequency in Hz.

Only possibly available when pwmN_mode is PWM, but not always present even then.

RW

/sys/class/hwmon/hwmonX/pwmY_mode

Defined on file *sysfs-class-hwmon*

- 0: DC mode (direct current)
- 1: PWM mode (pulse-width modulation)

RW

/sys/class/hwmon/hwmonX/tempY_auto_pointZ_pwm
/sys/class/hwmon/hwmonX/tempY_auto_pointZ_temp
/sys/class/hwmon/hwmonX/tempY_auto_pointZ_temp_hyst

Defined on file *sysfs-class-hwmon*

Define the PWM vs temperature curve.

Number of trip points is chip-dependent. Use this for chips which associate trip points to temperature channels.

RW

/sys/class/hwmon/hwmonX/tempY_crit

Defined on file *sysfs-class-hwmon*

Temperature critical max value, typically greater than corresponding temp_max values.

Unit: millidegree Celsius

RW

/sys/class/hwmon/hwmonX/tempY_crit_alarm

Defined on file *sysfs-class-hwmon*

Critical high temperature alarm flag.

- 0: OK
- 1: temperature has reached tempY_crit

RO

/sys/class/hwmon/hwmonX/tempY_crit_hyst

Defined on file *sysfs-class-hwmon*

Temperature hysteresis value for critical limit.

Unit: millidegree Celsius

Must be reported as an absolute temperature, NOT a delta from the critical value.

RW

/sys/class/hwmon/hwmonX/tempY_emergency

Defined on file *sysfs-class-hwmon*

Temperature emergency max value, for chips supporting more than two upper temperature limits. Must be equal or greater than corresponding temp_crit values.

Unit: millidegree Celsius

RW

/sys/class/hwmon/hwmonX/tempY_emergency_alarm

Defined on file *sysfs-class-hwmon*

Emergency high temperature alarm flag.

- 0: OK

- 1: temperature has reached tempY_emergency

RO

/sys/class/hwmon/hwmonX/tempY_emergency_hyst

Defined on file *sysfs-class-hwmon*

Temperature hysteresis value for emergency limit.

Unit: millidegree Celsius

Must be reported as an absolute temperature, NOT a delta from the emergency value.

RW

/sys/class/hwmon/hwmonX/tempY_enable

Defined on file *sysfs-class-hwmon*

Enable or disable the sensors.

When disabled the sensor read will return -ENODATA.

- 1: Enable
- 0: Disable

RW

/sys/class/hwmon/hwmonX/tempY_highest

Defined on file *sysfs-class-hwmon*

Historical maximum temperature

Unit: millidegree Celsius

RO

/sys/class/hwmon/hwmonX/tempY_input

Defined on file *sysfs-class-hwmon*

Temperature input value.

Unit: millidegree Celsius

RO

/sys/class/hwmon/hwmonX/tempY_label

Defined on file *sysfs-class-hwmon*

Suggested temperature channel label.

Text string

Should only be created if the driver has hints about what this temperature channel is being used for, and user-space doesn't. In all other cases, the label is provided by user-space.

RO

/sys/class/hwmon/hwmonX/tempY_lcrit

Defined on file *sysfs-class-hwmon*

Temperature critical min value, typically lower than corresponding temp_min values.

Unit: millidegree Celsius

RW

/sys/class/hwmon/hwmonX/tempY_lcrit_hyst

Defined on file *sysfs-class-hwmon*

Temperature hysteresis value for critical min limit.

Unit: millidegree Celsius

Must be reported as an absolute temperature, NOT a delta from the critical min value.

RW

/sys/class/hwmon/hwmonX/tempY_lowest

Defined on file *sysfs-class-hwmon*

Historical minimum temperature

Unit: millidegree Celsius

RO

/sys/class/hwmon/hwmonX/tempY_max

Defined on file *sysfs-class-hwmon*

Temperature max value.

Unit: millidegree Celsius (or millivolt, see below)

RW

/sys/class/hwmon/hwmonX/tempY_max_alarm

Defined on file *sysfs-class-hwmon*

Maximum temperature alarm flag.

- 0: OK
- 1: temperature has reached tempY_max

RO

/sys/class/hwmon/hwmonX/tempY_max_hyst

Defined on file *sysfs-class-hwmon*

Temperature hysteresis value for max limit.

Unit: millidegree Celsius

Must be reported as an absolute temperature, NOT a delta from the max value.

RW

/sys/class/hwmon/hwmonX/tempY_min

Defined on file *sysfs-class-hwmon*

Temperature min value.

Unit: millidegree Celsius

RW

/sys/class/hwmon/hwmonX/tempY_min_alarm

Defined on file *sysfs-class-hwmon*

Minimum temperature alarm flag.

- 0: OK
- 1: temperature has reached tempY_min

RO

/sys/class/hwmon/hwmonX/tempY_min_hyst

Defined on file *sysfs-class-hwmon*

Temperature hysteresis value for min limit. Unit: millidegree Celsius

Must be reported as an absolute temperature, NOT a delta from the min value.

RW

/sys/class/hwmon/hwmonX/tempY_offset

Defined on file *sysfs-class-hwmon*

Temperature offset which is added to the temperature reading by the chip.

Unit: millidegree Celsius

Read/Write value.

/sys/class/hwmon/hwmonX/tempY_rated_max

Defined on file *sysfs-class-hwmon*

Maximum rated temperature.

Unit: millidegree Celsius

RO

/sys/class/hwmon/hwmonX/tempY_rated_min

Defined on file *sysfs-class-hwmon*

Minimum rated temperature.

Unit: millidegree Celsius

RO

/sys/class/hwmon/hwmonX/tempY_reset_history

Defined on file *sysfs-class-hwmon*

Reset temp_lowest and temp_highest

WO

/sys/class/hwmon/hwmonX/tempY_type

Defined on file *sysfs-class-hwmon*

Sensor type selection.

Integers 1 to 6

RW

- 1: CPU embedded diode
- 2: 3904 transistor
- 3: thermal diode
- 4: thermistor
- 5: AMD AMDSI
- 6: Intel PECI

Not all types are supported by all chips

/sys/class/hwmon/hwmonX/temp_reset_history

Defined on file *sysfs-class-hwmon*

Reset temp_lowest and temp_highest for all sensors

WO

/sys/class/hwmon/hwmonX/update_interval

Defined on file *sysfs-class-hwmon*

The interval at which the chip will update readings. Unit: millisecond

RW

Some devices have a variable update rate or interval. This attribute can be used to change it to the desired value.

/sys/class/hwmon/hwmonX/vrm

Defined on file *sysfs-class-hwmon*

Voltage Regulator Module version number.

RW (but changing it should no more be necessary)

Originally the VRM standard version multiplied by 10, but now an arbitrary number, as not all standards have a version number.

Affects the way the driver calculates the CPU core reference voltage from the vid pins.

/sys/class/input/input(x)/device/function_row_physmap

Defined on file *sysfs-driver-input-cros-ec-keyb*

A space separated list of scancodes for the top row keys, ordered by the physical positions of the keys, from left to right.

/sys/class/input/input(x)/device/shutdown

Defined on file *sysfs-driver-input-axp-pek*

Shutdown time in us. Board is powered off if the button is pressed for more than <shutdown_time>

/sys/class/input/input(x)/device/startup

Defined on file *sysfs-driver-input-axp-pek*

Startup time in us. Board is powered on if the button is pressed for more than <startup_time>

/sys/class/intel_pmt/

Defined on file *sysfs-class-intel_pmt*

The intel_pmt/ class directory contains information for devices that expose hardware telemetry using Intel Platform Monitoring Technology (PMT)

/sys/class/intel_pmt/crashlog<x>

Defined on file *sysfs-class-intel_pmt*

The crashlog<x> directory contains files for configuring an instance of a PMT crashlog device that can perform crash data recording. Each crashlog<x> device has an associated crashlog

file. This file can be opened and mapped or read to access the resulting crashlog buffer. The register layout for the buffer can be determined from an XML file of specified GUID for the parent device.

/sys/class/intel_pmt/crashlog<x>/crashlog

Defined on file *sysfs-class-intel_pmt*

(RO) The crashlog buffer for this crashlog device. This file may be mapped or read to obtain the data.

/sys/class/intel_pmt/crashlog<x>/enable

Defined on file *sysfs-class-intel_pmt*

(RW) Boolean value controlling if the crashlog functionality is enabled for the crashlog device.

/sys/class/intel_pmt/crashlog<x>/guid

Defined on file *sysfs-class-intel_pmt*

(RO) The GUID for this crashlog device. The GUID identifies the version of the XML file for the parent device that should be used to determine the register layout.

/sys/class/intel_pmt/crashlog<x>/offset

Defined on file *sysfs-class-intel_pmt*

(RO) The offset of the buffer in bytes that corresponds to the mapping for the crashlog device.

/sys/class/intel_pmt/crashlog<x>/size

Defined on file *sysfs-class-intel_pmt*

(RO) The length of the result buffer in bytes that corresponds to the size for the crashlog buffer.

/sys/class/intel_pmt/crashlog<x>/trigger

Defined on file *sysfs-class-intel_pmt*

(RW) Boolean value controlling the triggering of the crashlog device node. When read it provides data on if the crashlog has been triggered. When written to it can be used to either clear the current trigger by writing false, or to trigger a new event if the trigger is not currently set.

/sys/class/intel_pmt/telem<x>

Defined on file *sysfs-class-intel_pmt*

The telem<x> directory contains files describing an instance of a PMT telemetry device that exposes hardware telemetry. Each telem<x> directory has an associated telem file. This file may be opened and mapped or read to access the telemetry space of the device. The register

layout of the telemetry space is determined from an XML file that matches the PCI device id and GUID for the device.

/sys/class/intel_pmt/telem<x>/guid

Defined on file *sysfs-class-intel_pmt*

(RO) The GUID for this telemetry device. The GUID identifies the version of the XML file for the parent device that is to be used to get the register layout.

/sys/class/intel_pmt/telem<x>/offset

Defined on file *sysfs-class-intel_pmt*

(RO) The offset of telemetry region in bytes that corresponds to the mapping for the telem file.

/sys/class/intel_pmt/telem<x>/size

Defined on file *sysfs-class-intel_pmt*

(RO) The size of telemetry region in bytes that corresponds to the mapping size for the telem file.

/sys/class/intel_pmt/telem<x>/telem

Defined on file *sysfs-class-intel_pmt*

(RO) The telemetry data for this telemetry device. This file may be mapped or read to obtain the data.

/sys/class/iommu/<iommu>/amd-iommu/cap

Defined on file *sysfs-class-iommu-amd-iommu*

IOMMU capability header as documented in the AMD IOMMU specification. Format: %x

/sys/class/iommu/<iommu>/amd-iommu/features

Defined on file *sysfs-class-iommu-amd-iommu*

Extended features of the IOMMU. Format: %llx

/sys/class/iommu/<iommu>/devices/

Defined on file *sysfs-class-iommu*

IOMMU drivers are able to link devices managed by a given IOMMU here to allow association of IOMMU to device.

/sys/class/iommu/<iommu>/intel-iommu/address

Defined on file *sysfs-class-iommu-intel-iommu*

Physical address of the VT-d DRHD for this IOMMU. Format: %llx. This allows association of a sysfs intel-iommu with a DMAR DRHD table entry.

/sys/class/iommu/<iommu>/intel-iommu/cap

Defined on file *sysfs-class-iommu-intel-iommu*

The cached hardware capability register value of this DRHD unit. Format: %llx.

/sys/class/iommu/<iommu>/intel-iommu/ecap

Defined on file *sysfs-class-iommu-intel-iommu*

The cached hardware extended capability register value of this DRHD unit. Format: %llx.

/sys/class/iommu/<iommu>/intel-iommu/version

Defined on file *sysfs-class-iommu-intel-iommu*

The architecture version as reported from the VT-d VER_REG. Format: %d:%d, major:minor

/sys/class/lcd/<lcd>/contrast

Defined on file *sysfs-class-lcd*

Current contrast of this LCD device. Value is between 0 and */sys/class/lcd/<lcd>/max_contrast*.

/sys/class/lcd/<lcd>/lcd_power

Defined on file *sysfs-class-lcd*

Control LCD power, values are FB_BLANK_* from fb.h

- FB_BLANK_UNBLANK (0) : power on.
- FB_BLANK_POWERDOWN (4) : power off

/sys/class/lcd/<lcd>/max_contrast

Defined on file *sysfs-class-lcd*

Maximum contrast for this LCD device.

/sys/class/leds/<led>/als_channel

Defined on file *sysfs-class-led-driver-lm3533*

Set the ALS output channel to use as input in ALS-current-control mode (1, 2), where:

1	out_current1
2	out_current2

/sys/class/leds/<led>/als_en

Defined on file *sysfs-class-led-driver-lm3533*

Enable ALS-current-control mode (0, 1).

/sys/class/leds/<led>/brightness

Defined on file *sysfs-class-led*

Set the brightness of the LED.

Most LEDs don't have hardware brightness support, so will just be turned on for non-zero brightness settings.

Note: For multicolor LEDs, writing to this file will update all LEDs within the group to a calculated percentage of what each color LED intensity is set to.

The percentage is calculated for each grouped LED via the equation below:

```
led_brightness = brightness * multi_intensity/max_brightness
```

For additional details please refer to /leds/leds-class-multicolor.

The value is between 0 and */sys/class/leds/<led>/max_brightness*.

Writing 0 to this file clears active trigger.

Writing non-zero to this file while trigger is active changes the top brightness trigger is going to use.

/sys/class/leds/<led>/brightness_hw_changed

Defined on file *sysfs-class-led*

Last hardware set brightness level for this LED. Some LEDs may be changed autonomously by hardware/firmware. Only LEDs where this happens and the driver can detect this, will have this file.

This file supports poll() to detect when the hardware changes the brightness.

Reading this file will return the last brightness level set by the hardware, this may be different from the current brightness. Reading this file when no hw brightness change event has happened will return an ENODATA error.

/sys/class/leds/<led>/cts

Defined on file *sysfs-class-led-trigger-tty*

CTS = Clear To Send DCE is ready to accept data from the DTE. If the line state is detected, the LED is switched on. If set to 0 (default), the LED will not evaluate CTS. If set to 1, the LED will evaluate CTS.

/sys/class/leds/<led>/dcd

Defined on file *sysfs-class-led-trigger-tty*

DCD = Data Carrier Detect DTE is receiving a carrier from the DCE. If the line state is detected, the LED is switched on. If set to 0 (default), the LED will not evaluate CAR (DCD). If set to 1, the LED will evaluate CAR (DCD).

/sys/class/leds/<led>/delay_off

Defined on file *sysfs-class-led-trigger-oneshot*

Specifies for how many milliseconds the LED has to stay at LED_OFF brightness after it has been armed. Defaults to 100 ms.

/sys/class/leds/<led>/delay_on

Defined on file *sysfs-class-led-trigger-oneshot*

Specifies for how many milliseconds the LED has to stay at LED_FULL brightness after it has been armed. Defaults to 100 ms.

/sys/class/leds/<led>/device/brightness

Defined on file *sysfs-class-led-driver-turris-omnia*

(RW) On the front panel of the Turris Omnia router there is also a button which can be used to control the intensity of all the LEDs at once, so that if they are too bright, user can dim them.

The microcontroller cycles between 8 levels of this global brightness (from 100% to 0%), but this setting can have any integer value between 0 and 100. It is therefore convenient to be able to change this setting from software.

Format: %i

/sys/class/leds/<led>/device/gamma_correction

Defined on file *sysfs-class-led-driver-turris-omnia*

(RW) Newer versions of the microcontroller firmware of the Turris Omnia router support gamma correction for the RGB LEDs. This feature can be enabled/disabled by writing to this file.

If the feature is not supported because the MCU firmware is too old, the file always reads as 0, and writing to the file results in the EOPNOTSUPP error.

Format: %i

/sys/class/leds/<led>/device_name

Defined on file *sysfs-class-led-trigger-netdev*

Specifies the network device name to monitor.

/sys/class/leds/<led>/dim

Defined on file *sysfs-class-led-driver-aw200xx*

64-level DIM current. If you write a negative value or "auto", the dim will be calculated according to the brightness.

/sys/class/leds/<led>/dsr

Defined on file *sysfs-class-led-trigger-tty*

DSR = Data Set Ready DCE is ready to receive and send data. If the line state is detected, the LED is switched on. If set to 0 (default), the LED will not evaluate DSR. If set to 1, the LED will evaluate DSR.

/sys/class/leds/<led>/falltime
/sys/class/leds/<led>/risetime

Defined on file *sysfs-class-led-driver-lm3533*

Set the pattern generator fall and rise times (0..7), where:

0	2048 us
1	262 ms
2	524 ms
3	1.049 s
4	2.097 s
5	4.194 s
6	8.389 s
7	16.78 s

/sys/class/leds/<led>/flash_brightness

Defined on file *sysfs-class-led-flash*

read/write Set the brightness of this LED in the flash strobe mode, in microamperes. The file is created only for the flash LED devices that support setting flash brightness.

The value is between 0 and */sys/class/leds/<led>/max_flash_brightness*.

/sys/class/leds/<led>/flash_fault

Defined on file *sysfs-class-led-flash*

read only Space separated list of flash faults that may have occurred. Flash faults are re-read after strobing the flash. Possible flash faults:

- **led-over-voltage**
  flash controller voltage to the flash LED has exceeded the limit specific to the flash controller

- **flash-timeout-exceeded**
  the flash strobe was still on when the timeout set by the user has expired; not all flash controllers may set this in all such conditions
- **controller-over-temperature**
  the flash controller has overheated
- **controller-short-circuit**
  the short circuit protection of the flash controller has been triggered
- **led-power-supply-over-current**
  current in the LED power supply has exceeded the limit specific to the flash controller
- **indicator-led-fault**
  the flash controller has detected a short or open circuit condition on the indicator LED
- **led-under-voltage**
  flash controller voltage to the flash LED has been below the minimum limit specific to the flash
- **controller-under-voltage**
  the input voltage of the flash controller is below the limit under which strobing the flash at full current will not be possible; the condition persists until this flag is no longer set
- **led-over-temperature**
  the temperature of the LED has exceeded its allowed upper limit

/sys/class/leds/<led>/flash_strobe

Defined on file *sysfs-class-led-flash*

read/write Flash strobe state. When written with 1 it triggers flash strobe and when written with 0 it turns the flash off.

On read 1 means that flash is currently strobing and 0 means that flash is off.

/sys/class/leds/<led>/flash_timeout

Defined on file *sysfs-class-led-flash*

read/write Hardware timeout for flash, in microseconds. The flash strobe is stopped after this period of time has passed from the start of the strobe. The file is created only for the flash LED devices that support setting flash timeout.

/sys/class/leds/<led>/full_duplex

Defined on file *sysfs-class-led-trigger-netdev*

Signal the link full duplex state of the named network device.

If set to 0 (default), the LED's normal state is off.

If set to 1, the LED's normal state reflects the link full duplex state of the named network device. Setting this value also immediately changes the LED state.

/sys/class/leds/<led>/gt683r/mode

Defined on file *sysfs-class-leds-gt683r*

Set the mode of LEDs. You should notice that changing the mode of one LED will update the mode of its two sibling devices as well. Possible values are:

0	normal
1	audio
2	breathing

Normal: LEDs are fully on when enabled Audio: LEDs brightness depends on sound level Breathing: LEDs brightness varies at human breathing rate

/sys/class/leds/<led>/half_duplex

Defined on file *sysfs-class-led-trigger-netdev*

Signal the link half duplex state of the named network device.

If set to 0 (default), the LED's normal state is off.

If set to 1, the LED's normal state reflects the link half duplex state of the named network device. Setting this value also immediately changes the LED state.

/sys/class/leds/<led>/hw_pattern

Defined on file *sysfs-class-led-trigger-pattern*

Specify a hardware pattern for the LED, for LED hardware that supports autonomously controlling brightness over time, according to some preprogrammed hardware patterns. It deactivates any active software pattern.

Since different LED hardware can have different semantics of hardware patterns, each driver is expected to provide its own description for the hardware patterns in their documentation file at Documentation/leds/.

/sys/class/leds/<led>/id

Defined on file *sysfs-class-led-driver-lm3533*

Get the id of this led (0..3).

/sys/class/leds/<led>/interval

Defined on file *sysfs-class-led-trigger-netdev*

Specifies the duration of the LED blink in milliseconds. Defaults to 50 ms.

When offloaded is true, the interval value MUST be set to the default value and cannot be changed. Trying to set any value in this specific mode will return an EINVAL error.

/sys/class/leds/<led>/invert

Defined on file *sysfs-class-led-trigger-oneshot*

Reverse the blink logic. If set to 0 (default) blink on for delay_on ms, then blink off for delay_off ms, leaving the LED normally off. If set to 1, blink off for delay_off ms, then blink on for delay_on ms, leaving the LED normally on. Setting this value also immediately changes the LED state.

/sys/class/leds/<led>/inverted

Defined on file *sysfs-class-led*

Invert the LED on/off state. This parameter is specific to gpio and backlight triggers. In case of the backlight trigger, it is useful when driving a LED which is intended to indicate a device in a standby like state.

/sys/class/leds/<led>/linear

Defined on file *sysfs-class-led-driver-lm3533*

Set the brightness-mapping mode (0, 1), where:

0	exponential mode
1	linear mode

/sys/class/leds/<led>/link

Defined on file *sysfs-class-led-trigger-netdev*

Signal the link state of the named network device.

If set to 0 (default), the LED's normal state is off.

If set to 1, the LED's normal state reflects the link state of the named network device. Setting this value also immediately changes the LED state.

/sys/class/leds/<led>/link_10

Defined on file *sysfs-class-led-trigger-netdev*

Signal the link speed state of 10Mbps of the named network device.

If set to 0 (default), the LED's normal state is off.

If set to 1, the LED's normal state reflects the link state speed of 10MBps of the named network device. Setting this value also immediately changes the LED state.

/sys/class/leds/<led>/link_100

Defined on file *sysfs-class-led-trigger-netdev*

Signal the link speed state of 100Mbps of the named network device.

If set to 0 (default), the LED's normal state is off.

If set to 1, the LED's normal state reflects the link state speed of 100Mbps of the named network device. Setting this value also immediately changes the LED state.

/sys/class/leds/<led>/link_1000

Defined on file *sysfs-class-led-trigger-netdev*

Signal the link speed state of 1000Mbps of the named network device.

If set to 0 (default), the LED's normal state is off.

If set to 1, the LED's normal state reflects the link state speed of 1000Mbps of the named network device. Setting this value also immediately changes the LED state.

/sys/class/leds/<led>/link_10000

Defined on file *sysfs-class-led-trigger-netdev*

Signal the link speed state of 10000Mbps of the named network device.

If set to 0 (default), the LED's normal state is off.

If set to 1, the LED's normal state reflects the link state speed of 10000Mbps of the named network device. Setting this value also immediately changes the LED state.

/sys/class/leds/<led>/link_2500

Defined on file *sysfs-class-led-trigger-netdev*

Signal the link speed state of 2500Mbps of the named network device.

If set to 0 (default), the LED's normal state is off.

If set to 1, the LED's normal state reflects the link state speed of 2500Mbps of the named network device. Setting this value also immediately changes the LED state.

/sys/class/leds/<led>/link_5000

Defined on file *sysfs-class-led-trigger-netdev*

Signal the link speed state of 5000Mbps of the named network device.

If set to 0 (default), the LED's normal state is off.

If set to 1, the LED's normal state reflects the link state speed of 5000Mbps of the named network device. Setting this value also immediately changes the LED state.

/sys/class/leds/<led>/max_brightness

Defined on file *sysfs-class-led*

Maximum brightness level for this LED, default is 255 (LED_FULL).

If the LED does not support different brightness levels, this should be 1.

/sys/class/leds/<led>/max_flash_brightness

Defined on file *sysfs-class-led-flash*

read only Maximum brightness level for this LED in the flash strobe mode, in microamperes.

/sys/class/leds/<led>/max_flash_timeout

Defined on file *sysfs-class-led-flash*

read only Maximum flash timeout for this LED, in microseconds.

/sys/class/leds/<led>/multi_index

Defined on file *sysfs-class-led-multicolor*

read The multi_index array, when read, will output the LED colors as an array of strings as they are indexed in the multi_intensity file.

For additional details please refer to /leds/leds-class-multicolor.

/sys/class/leds/<led>/multi_intensity

Defined on file *sysfs-class-led-multicolor*

read/write This file contains array of integers. Order of components is described by the multi_index array. The maximum intensity should not exceed */sys/class/leds/<led>/max_brightness*.

For additional details please refer to /leds/leds-class-multicolor.

/sys/class/leds/<led>/offloaded

Defined on file *sysfs-class-led-trigger-netdev*

Communicate whether the LED trigger modes are offloaded to hardware or whether software fallback is used.

If 0, the LED is using software fallback to blink.

If 1, the LED blinking in requested mode is offloaded to hardware.

/sys/class/leds/<led>/pattern

Defined on file *sysfs-class-led-trigger-pattern*

Specify a software pattern for the LED, that supports altering the brightness for the specified duration with one software timer. It can do gradual dimming and step change of brightness.

The pattern is given by a series of tuples, of brightness and duration (ms).

The exact format is described in: Documentation/devicetree/bindings/leds/leds-trigger-pattern.txt

/sys/class/leds/<led>/ports/<port>

Defined on file *sysfs-class-led-trigger-usbport*

Every dir entry represents a single USB port that can be selected for the USB port trigger. Selecting ports makes trigger observing them for any connected devices and lighting on LED if there are any.

Echoing ”1” value selects USB port. Echoing ”0” unselects it. Current state can be also read.

/sys/class/leds/<led>/pwm

Defined on file *sysfs-class-led-driver-lm3533*

Set the PWM-input control mask (5 bits), where:

bit 5	PWM-input enabled in Zone 4
bit 4	PWM-input enabled in Zone 3
bit 3	PWM-input enabled in Zone 2
bit 2	PWM-input enabled in Zone 1
bit 1	PWM-input enabled in Zone 0
bit 0	PWM-input enabled

/sys/class/leds/<led>/repeat

Defined on file *sysfs-class-led-trigger-pattern*

Specify a pattern repeat number. -1 means repeat indefinitely, other negative numbers and number 0 are invalid.

This file will always return the originally written repeat number.

It should be noticed that some leds, like EL15203000 may only support indefinitely patterns, so they always store -1.

/sys/class/leds/<led>/rng

Defined on file *sysfs-class-led-trigger-tty*

RNG = Ring Indicator DCE has detected an incoming ring signal on the telephone line. If the line state is detected, the LED is switched on. If set to 0 (default), the LED will not evaluate RNG. If set to 1, the LED will evaluate RNG.

/sys/class/leds/<led>/rx

Defined on file *sysfs-class-led-trigger-netdev*

Defined on file *sysfs-class-led-trigger-tty*

Signal reception of data on the named network device.

If set to 0 (default), the LED will not blink on reception.

If set to 1, the LED will blink for the milliseconds specified in interval to signal reception.

When offloaded is true, the blink interval is controlled by hardware and won't reflect the value set in interval.

Signal reception (rx) of data on the named tty device. If set to 0, the LED will not blink on reception. If set to 1 (default), the LED will blink on reception.

/sys/class/leds/<led>/shot

Defined on file *sysfs-class-led-trigger-oneshot*

Write any non-empty string to signal an events, this starts a blink sequence if not already running.

/sys/class/leds/<led>/trigger

Defined on file *sysfs-class-led*

Set the trigger for this LED. A trigger is a kernel based source of LED events.

You can change triggers in a similar manner to the way an IO scheduler is chosen. Trigger specific parameters can appear in /sys/class/leds/<led> once a given trigger is selected. For their documentation see *sysfs-class-led-trigger-**.

/sys/class/leds/<led>/ttyname

Defined on file *sysfs-class-led-trigger-tty*

Specifies the tty device name of the triggering tty

/sys/class/leds/<led>/tx

Defined on file *sysfs-class-led-trigger-netdev*

Defined on file *sysfs-class-led-trigger-tty*

Signal transmission of data on the named network device.

If set to 0 (default), the LED will not blink on transmission.

If set to 1, the LED will blink for the milliseconds specified in interval to signal transmission.

When offloaded is true, the blink interval is controlled by hardware and won't reflect the value set in interval.

Signal transmission (tx) of data on the named tty device. If set to 0, the LED will not blink on transmission. If set to 1 (default), the LED will blink on transmission.

/sys/class/leds/SRWS1::<serial>::RPM1
/sys/class/leds/SRWS1::<serial>::RPM2
/sys/class/leds/SRWS1::<serial>::RPM3
/sys/class/leds/SRWS1::<serial>::RPM4
/sys/class/leds/SRWS1::<serial>::RPM5
/sys/class/leds/SRWS1::<serial>::RPM6
/sys/class/leds/SRWS1::<serial>::RPM7
/sys/class/leds/SRWS1::<serial>::RPM8
/sys/class/leds/SRWS1::<serial>::RPM9
/sys/class/leds/SRWS1::<serial>::RPM10
/sys/class/leds/SRWS1::<serial>::RPM11
/sys/class/leds/SRWS1::<serial>::RPM12
/sys/class/leds/SRWS1::<serial>::RPM13
/sys/class/leds/SRWS1::<serial>::RPM14
/sys/class/leds/SRWS1::<serial>::RPM15
/sys/class/leds/SRWS1::<serial>::RPMALL

Defined on file *sysfs-driver-hid-srws1*

Provides a control for turning on/off the LEDs which form an RPM meter on the front of the controller

/sys/class/leds/dell::kbd_backlight/als_enabled

Defined on file *sysfs-platform-dell-laptop*

This file allows to control the automatic keyboard illumination mode on some systems that have an ambient light sensor. Write 1 to this file to enable the auto mode, 0 to disable it.

/sys/class/leds/dell::kbd_backlight/als_setting

Defined on file *sysfs-platform-dell-laptop*

This file allows to specify the on/off threshold value, as reported by the ambient light sensor.

/sys/class/leds/dell::kbd_backlight/start_triggers

Defined on file *sysfs-platform-dell-laptop*

This file allows to control the input triggers that turn on the keyboard backlight illumination that is disabled because of inactivity. Read the file to see the triggers available. The ones enabled are preceded by '+', those disabled by '-'.

To enable a trigger, write its name preceded by '+' to this file. To disable a trigger, write its name preceded by '-' instead.

For example, to enable the keyboard as trigger run:

```
echo +keyboard > /sys/class/leds/dell::kbd_backlight/start_triggers
```

To disable it:

```
echo -keyboard > /sys/class/leds/dell::kbd_backlight/start_triggers
```

Note that not all the available triggers can be configured.

/sys/class/leds/dell::kbd_backlight/stop_timeout

Defined on file *sysfs-platform-dell-laptop*

This file allows to specify the interval after which the keyboard illumination is disabled because of inactivity. The timeouts are expressed in seconds, minutes, hours and days, for which the symbols are 's', 'm', 'h' and 'd' respectively.

To configure the timeout, write to this file a value along with any the above units. If no unit is specified, the value is assumed to be expressed in seconds.

For example, to set the timeout to 10 minutes run:

```
echo 10m > /sys/class/leds/dell::kbd_backlight/stop_timeout
```

Note that when this file is read, the returned value might be expressed in a different unit than the one used when the timeout was set.

Also note that only some timeouts are supported and that some systems might fall back to a specific timeout in case an invalid timeout is written to this file.

/sys/class/mdio_bus/<bus>/<device>/attached_dev

Defined on file *sysfs-class-net-phydev*

Symbolic link to the network device this PHY device is attached to.

/sys/class/mdio_bus/<bus>/<device>/phy_dev_flags

Defined on file *sysfs-class-net-phydev*

32-bit hexadecimal number representing a bit mask of the configuration bits passed from the consumer of the PHY (Ethernet MAC, switch, etc.) to the PHY driver. The flags are only used internally by the kernel and their placement are not meant to be stable across kernel versions. This is intended for facilitating the debugging of PHY drivers.

/sys/class/mdio_bus/<bus>/<device>/phy_has_fixups

Defined on file *sysfs-class-net-phydev*

This attribute contains the boolean value whether a given PHY device has had any ”fixup” workaround running on it, encoded as a boolean. This information is provided to help troubleshooting PHY configurations.

/sys/class/mdio_bus/<bus>/<device>/phy_id

Defined on file *sysfs-class-net-phydev*

This attribute contains the 32-bit PHY Identifier as reported by the device during bus enumeration, encoded in hexadecimal. This ID is used to match the device with the appropriate driver.

/sys/class/mdio_bus/<bus>/<device>/phy_interface

Defined on file *sysfs-class-net-phydev*

This attribute contains the PHY interface as configured by the Ethernet driver during bus enumeration, encoded in string. This interface mode is used to configure the Ethernet MAC with the appropriate mode for its data lines to the PHY hardware.

Possible values are:

<empty> (not available), mii, gmii, sgmii, tbi, rev-mii, rmii, rgmii, rgmii-id, rgmii-rxid, rgmii-txid, rtbi, smii xgmii, moca, qsgmii, trgmii, 1000base-x, 2500base-x, rxaui, xaui, 10gbase-kr, unknown

/sys/class/mdio_bus/<bus>/<device>/phy_standalone

Defined on file *sysfs-class-net-phydev*

Boolean value indicating whether the PHY device is used in standalone mode, without a net_device associated, by PHYLINK. Attribute created only when this is the case.

/sys/class/mei/

Defined on file *sysfs-class-mei*

The mei/ class sub-directory belongs to mei device class

/sys/class/mei/mei<N>/

Defined on file *sysfs-class-mei*

The /sys/class/mei/meiN directory is created for each probed mei device

/sys/class/mei/mei<N>/dev_state

Defined on file *sysfs-class-mei*

Display the ME device state.

The device state can have following values: INITIALIZING INIT_CLIENTS ENABLED RESETTING DISABLED POWER_DOWN POWER_UP

/sys/class/mei/mei<N>/fw_status

Defined on file *sysfs-class-mei*

Display fw status registers content

The ME FW writes its status information into fw status registers for BIOS and OS to monitor fw health.

The register contains running state, power management state, error codes, and others. The way the registers are decoded depends on PCH or SoC generation. Also number of registers varies between 1 and 6 depending on generation.

/sys/class/mei/mei<N>/fw_ver

Defined on file *sysfs-class-mei*

Display the ME firmware version.

The version of the platform ME firmware is in format: <platform>:<major>.<minor>.<milestone>.<build_no>. There can be up to three such blocks for different FW components.

/sys/class/mei/mei<N>/hbm_ver

Defined on file *sysfs-class-mei*

Display the negotiated HBM protocol version.

The HBM protocol version negotiated between the driver and the device.

/sys/class/mei/mei<N>/hbm_ver_drv

Defined on file *sysfs-class-mei*

Display the driver HBM protocol version.

The HBM protocol version supported by the driver.

/sys/class/mei/mei<N>/kind

Defined on file *sysfs-class-mei*

Display kind of the device

Generic devices are marked as ”mei” while special purpose have their own names. Available options: - mei: generic mei device. - itouch: itouch (ipts) mei device.

/sys/class/mei/mei<N>/trc

Defined on file *sysfs-class-mei*

Display trc status register content

The ME FW writes Glitch Detection HW (TRC) status information into trc status register for BIOS and OS to monitor fw health.

/sys/class/mei/mei<N>/tx_queue_limit

Defined on file *sysfs-class-mei*

Configure tx queue limit

Set maximal number of pending writes per opened session.

/sys/class/mic/

Defined on file *sysfs-class-mic*

The mic class directory belongs to Intel MIC devices and provides information per MIC device. An Intel MIC device is a PCIe form factor add-in Coprocessor card based on the Intel Many Integrated Core (MIC) architecture that runs a Linux OS.

/sys/class/mic/mic<X>

Defined on file *sysfs-class-mic*

The directories /sys/class/mic/mic0, /sys/class/mic/mic1 etc., represent MIC devices (0,1,..etc). Each directory has information specific to that MIC device.

/sys/class/mic/mic<X>/bootmode

Defined on file *sysfs-class-mic*

When read, this sysfs entry provides the current bootmode for the card. This sysfs entry can be written with the following valid strings: a) linux - Boot a Linux image. b) flash - Boot an image for flash updates.

/sys/class/mic/mic<X>/cmdline

Defined on file *sysfs-class-mic*

An Intel MIC device runs a Linux OS during its operation. Before booting this card OS, it is possible to pass kernel command line options to configure various features in it, similar to self-bootable machines. When read, this entry provides information about the current kernel command line options set to boot the card OS. This entry can be written to change the existing kernel command line options. Typically, the user would want to read the current command line options, append new ones or modify existing ones and then write the whole kernel command line back to this entry.

/sys/class/mic/mic<X>/family

Defined on file *sysfs-class-mic*

Provides information about the Coprocessor family for an Intel MIC device. For example - ”x100”

/sys/class/mic/mic<X>/firmware

Defined on file *sysfs-class-mic*

When read, this sysfs entry provides the path name under /lib/firmware/ where the firmware image to be booted on the card can be found. The entry can be written to change the firmware image location under /lib/firmware/.

/sys/class/mic/mic<X>/heartbeat_enable

Defined on file *sysfs-class-mic*

The MIC drivers detect and inform user space about card crashes via a heartbeat mechanism (see the description of shutdown_status above). User space can turn off this notification by setting heartbeat_enable to 0 and enable it by setting this entry to 1. If this notification is disabled it is the responsibility of user space to detect card crashes via alternative means such as a network ping. This setting is enabled by default.

/sys/class/mic/mic<X>/log_buf_addr

Defined on file *sysfs-class-mic*

An Intel MIC device runs a Linux OS during its operation. For debugging purpose and early kernel boot messages, the user can access the card OS log buffer via debugfs. When read, this entry provides the kernel virtual address of the buffer where the card OS log buffer can be read. This entry is written by the host configuration daemon to set the log buffer address. The correct log buffer address to be written can be found in the System.map file of the card OS.

/sys/class/mic/mic<X>/log_buf_len

Defined on file *sysfs-class-mic*

An Intel MIC device runs a Linux OS during its operation. For debugging purpose and early kernel boot messages, the user can access the card OS log buffer via debugfs. When read, this entry provides the kernel virtual address where the card OS log buffer length can be read. This entry is written by host configuration daemon to set the log buffer length address. The correct log buffer length address to be written can be found in the System.map file of the card OS.

/sys/class/mic/mic<X>/ramdisk

Defined on file *sysfs-class-mic*

When read, this sysfs entry provides the path name under /lib/firmware/ where the ramdisk image to be used during card OS boot can be found. The entry can be written to change the ramdisk image location under /lib/firmware/.

/sys/class/mic/mic<X>/shutdown_status

Defined on file *sysfs-class-mic*

An Intel MIC device runs a Linux OS during its operation. This OS can shutdown because of various reasons. When read, this entry provides the status on why the card OS was shutdown. Possible values are:

”nop”	shutdown status is not applicable, when the card OS is ”online”
”crashed”	Shutdown because of a HW or SW crash.
”halted”	Shutdown because of a halt command.
”poweroff”	Shutdown because of a poweroff command.
”restart”	Shutdown because of a restart command.

/sys/class/mic/mic<X>/state

Defined on file *sysfs-class-mic*

When read, this entry provides the current state of an Intel MIC device in the context of the card OS. Possible values that will be read are:

”ready”	The MIC device is ready to boot the card OS. On reading this entry after an OSPM resume, a ”boot” has to be written to this entry if the card was previously shutdown during OSPM suspend.
”booting”	The MIC device has initiated booting a card OS.
”online”	The MIC device has completed boot and is online
”shutting_down”	The card OS is shutting down.
”resetting”	A reset has been initiated for the MIC device
”reset_failed”	The MIC device has failed to reset.

When written, this sysfs entry triggers different state change operations depending upon the current state of the card OS. Acceptable values are:

”boot”	Boot the card OS image specified by the combination of firmware, ramdisk, cmdline and bootmode sysfs entries.
”reset”	Initiates device reset.
”shutdown”	Initiates card OS shutdown.

/sys/class/mic/mic<X>/stepping

Defined on file *sysfs-class-mic*

Provides information about the silicon stepping for an Intel MIC device. For example - ”A0” or ”B0”

/sys/class/misc/drivers/dw-xdata-pcie.<device>/write

Defined on file *sysfs-driver-xdata*

Allows the user to enable the PCIe traffic generator which will create write TLPs frames - from the Root Complex to the Endpoint direction or to disable the PCIe traffic generator in all directions.

Write y/1/on to enable, n/0/off to disable

Usage e.g.
echo 1 > /sys/class/misc/dw-xdata-pcie.<device>/write

or
echo 0 > /sys/class/misc/dw-xdata-pcie.<device>/write

The user can read the current PCIe link throughput generated through this generator in MB/s.

Usage e.g.
cat /sys/class/misc/dw-xdata-pcie.<device>/write 204

The file is read and write.

/sys/class/misc/dw-xdata-pcie.<device>/read

Defined on file *sysfs-driver-xdata*

Allows the user to enable the PCIe traffic generator which will create read TLPs frames - from the Endpoint to the Root Complex direction or to disable the PCIe traffic generator in all directions.

Write y/1/on to enable, n/0/off to disable

Usage e.g.
echo 1 > /sys/class/misc/dw-xdata-pcie.<device>/read

or
echo 0 > /sys/class/misc/dw-xdata-pcie.<device>/read

The user can read the current PCIe link throughput generated through this generator in MB/s.

Usage e.g.
cat /sys/class/misc/dw-xdata-pcie.<device>/read 199

The file is read and write.

/sys/class/mtd/

Defined on file *sysfs-class-mtd*

The mtd/ class subdirectory belongs to the MTD subsystem (MTD core).

/sys/class/mtd/mtdX/

Defined on file *sysfs-class-mtd*

The /sys/class/mtd/mtd{0,1,2,3,...} directories correspond to each /dev/mtdX character device. These may represent physical/simulated flash devices, partitions on a flash device, or concatenated flash devices.

/sys/class/mtd/mtdX/bad_blocks

Defined on file *sysfs-class-mtd*

The number of blocks marked as bad, if any, in this partition.

/sys/class/mtd/mtdX/bbt_blocks

Defined on file *sysfs-class-mtd*

The number of blocks that are marked as reserved, if any, in this partition. These are typically used to store the in-flash bad block table (BBT).

/sys/class/mtd/mtdX/bitflip_threshold

Defined on file *sysfs-class-mtd*

This allows the user to examine and adjust the criteria by which mtd returns -EUCLEAN from mtd_read() and mtd_read_oob(). If the maximum number of bit errors that were corrected on any single region comprising an ecc step (as reported by the driver) equals or exceeds this

value, -EUCLEAN is returned. Otherwise, absent an error, 0 is returned. Higher layers (e.g., UBI) use this return code as an indication that an erase block may be degrading and should be scrutinized as a candidate for being marked as bad.

The initial value may be specified by the flash device driver. If not, then the default value is ecc_strength.

The introduction of this feature brings a subtle change to the meaning of the -EUCLEAN return code. Previously, it was interpreted to mean simply "one or more bit errors were corrected". Its new interpretation can be phrased as "a dangerously high number of bit errors were corrected on one or more regions comprising an ecc step". The precise definition of "dangerously high" can be adjusted by the user with bitflip_threshold. Users are discouraged from doing this, however, unless they know what they are doing and have intimate knowledge of the properties of their device. Broadly speaking, bitflip_threshold should be low enough to detect genuine erase block degradation, but high enough to avoid the consequences of a persistent return value of -EUCLEAN on devices where sticky bitflips occur. Note that if bitflip_threshold exceeds ecc_strength, -EUCLEAN is never returned by the read operations. Conversely, if bitflip_threshold is zero, -EUCLEAN is always returned, absent a hard error.

This is generally applicable only to NAND flash devices with ECC capability. It is ignored on devices lacking ECC capability; i.e., devices for which ecc_strength is zero.

/sys/class/mtd/mtdX/corrected_bits

Defined on file *sysfs-class-mtd*

The number of bits that have been corrected by means of the device's ECC.

It will always be a non-negative integer. In the case of devices lacking any ECC capability, it is 0.

/sys/class/mtd/mtdX/dev

Defined on file *sysfs-class-mtd*

Major and minor numbers of the character device corresponding to this MTD device (in <major>:<minor> format). This is the read-write device so <minor> will be even.

/sys/class/mtd/mtdX/ecc_failures

Defined on file *sysfs-class-mtd*

The number of failures reported by this device's ECC. Typically, these failures are associated with failed read operations.

It will always be a non-negative integer. In the case of devices lacking any ECC capability, it is 0.

/sys/class/mtd/mtdX/ecc_step_size

Defined on file *sysfs-class-mtd*

The size of a single region covered by ECC, known as the ECC step. Devices may have several equally sized ECC steps within each writesize region.

It will always be a non-negative integer. In the case of devices lacking any ECC capability, it is 0.

/sys/class/mtd/mtdX/ecc_strength

Defined on file *sysfs-class-mtd*

Maximum number of bit errors that the device is capable of correcting within each region covering an ECC step (see ecc_step_size). This will always be a non-negative integer.

In the case of devices lacking any ECC capability, it is 0.

/sys/class/mtd/mtdX/erasesize

Defined on file *sysfs-class-mtd*

"Major" erase size for the device. If numeraseregions is zero, this is the eraseblock size for the entire device. Otherwise, the MEMGETREGIONCOUNT/MEMGETREGIONINFO ioctls can be used to determine the actual eraseblock layout.

/sys/class/mtd/mtdX/flags

Defined on file *sysfs-class-mtd*

A hexadecimal value representing the device flags, ORed together:

0x0400: MTD_WRITEABLE - device is writable 0x0800: MTD_BIT_WRITEABLE - single bits can be flipped 0x1000: MTD_NO_ERASE - no erase necessary 0x2000: MTD_POWERUP_LOCK - always locked after reset

/sys/class/mtd/mtdX/name

Defined on file *sysfs-class-mtd*

A human-readable ASCII name for the device or partition. This will match the name in /proc/mtd .

/sys/class/mtd/mtdX/numeraseregions

Defined on file *sysfs-class-mtd*

For devices that have variable eraseblock sizes, this provides the total number of erase regions. Otherwise, it will read back as zero.

/sys/class/mtd/mtdX/offset

Defined on file *sysfs-class-mtd*

For a partition, the offset of that partition from the start of the parent (another partition or a flash device) in bytes. This attribute is absent on flash devices, so it can be used to distinguish them from partitions.

/sys/class/mtd/mtdX/oobavail

Defined on file *sysfs-class-mtd*

Number of bytes available for a client to place data into the out of band area.

/sys/class/mtd/mtdX/oobsize

Defined on file *sysfs-class-mtd*

Number of OOB bytes per page.

/sys/class/mtd/mtdX/size

Defined on file *sysfs-class-mtd*

Total size of the device/partition, in bytes.

/sys/class/mtd/mtdX/type

Defined on file *sysfs-class-mtd*

One of the following ASCII strings, representing the device type:

absent, ram, rom, nor, nand, mlc-nand, dataflash, ubi, unknown

/sys/class/mtd/mtdX/writesize

Defined on file *sysfs-class-mtd*

Minimal writable flash unit size. This will always be a positive integer.

In the case of NOR flash it is 1 (even though individual bits can be cleared).

In the case of NAND flash it is one NAND page (or a half page, or a quarter page).

In the case of ECC NOR, it is the ECC block size.

/sys/class/mtd/mtdXro/

Defined on file *sysfs-class-mtd*

These directories provide the corresponding read-only device nodes for */sys/class/mtd/mtdX/* .

/sys/class/mtd/mtdXro/dev

Defined on file *sysfs-class-mtd*

Major and minor numbers of the character device corresponding to the read-only variant of the MTD device (in <major>:<minor> format). In this case <minor> will be odd.

/sys/class/mux/

Defined on file *sysfs-class-mux*

The mux/ class sub-directory belongs to the Generic MUX Framework and provides a sysfs interface for using MUX controllers.

/sys/class/mux/muxchip<N>/

Defined on file *sysfs-class-mux*

A /sys/class/mux/muxchipN directory is created for each probed MUX chip where N is a simple enumeration.

/sys/class/net/<bridge iface>/bridge/group_fwd_mask

Defined on file *sysfs-class-net*

Bitmask to allow forwarding of link local frames with address 01-80-C2-00-00-0X on a bridge device. Only values that set bits not matching BR_GROUPFWD_RESTRICTED in net/bridge/br_private.h allowed. Default value 0 does not forward any link local frames.

Restricted bits:

0	01-80-C2-00-00-00 Bridge Group Address used for STP
1	01-80-C2-00-00-01 (MAC Control) 802.3 used for MAC PAUSE
2	01-80-C2-00-00-02 (Link Aggregation) 802.3ad

Any values not setting these bits can be used. Take special care when forwarding control frames e.g. 802.1X-PAE or LLDP.

/sys/class/net/<iface>/addr_assign_type

Defined on file *sysfs-class-net*

Indicates the address assignment type. Possible values are:

0	permanent address
1	randomly generated
2	stolen from another device
3	set using dev_set_mac_address

/sys/class/net/<iface>/addr_len

Defined on file *sysfs-class-net*

Indicates the hardware address size in bytes. Values vary based on the lower-level protocol used by the interface (Ethernet, FDDI, ATM, IEEE 802.15.4...). See include/uapi/linux/if_*.h for actual values.

/sys/class/net/<iface>/address

Defined on file *sysfs-class-net*

Hardware address currently assigned to this interface. Format is a string, e.g: 00:11:22:33:44:55 for an Ethernet MAC address.

/sys/class/net/<iface>/broadcast

Defined on file *sysfs-class-net*

Hardware broadcast address for this interface. Format is a string, e.g: ff:ff:ff:ff:ff:ff for an Ethernet broadcast MAC address.

/sys/class/net/<iface>/carrier

Defined on file *sysfs-class-net*

Indicates the current physical link state of the interface. Possible values are:

0	physical link is down
1	physical link is up

Note: some special devices, e.g: bonding and team drivers will allow this attribute to be written to force a link state for operating correctly and designating another fallback interface.

/sys/class/net/<iface>/carrier_changes

Defined on file *sysfs-class-net*

32-bit unsigned integer counting the number of times the link has seen a change from UP to DOWN and vice versa

/sys/class/net/<iface>/carrier_down_count

Defined on file *sysfs-class-net*

32-bit unsigned integer counting the number of times the link has been down

/sys/class/net/<iface>/carrier_up_count

Defined on file *sysfs-class-net*

32-bit unsigned integer counting the number of times the link has been up

/sys/class/net/<iface>/cdc_ncm/bmNtbFormatsSupported

Defined on file *sysfs-class-net-cdc_ncm*

- Bit 0: 16-bit NTB supported (set to 1)
- Bit 1: 32-bit NTB supported
- Bits 2 – 15: reserved (reset to zero; must be ignored by host)

/sys/class/net/<iface>/cdc_ncm/dwNtbInMaxSize

Defined on file *sysfs-class-net-cdc_ncm*

IN NTB Maximum Size in bytes

/sys/class/net/<iface>/cdc_ncm/dwNtbOutMaxSize

Defined on file *sysfs-class-net-cdc_ncm*

OUT NTB Maximum Size

/sys/class/net/<iface>/cdc_ncm/min_tx_pkt

Defined on file *sysfs-class-net-cdc_ncm*

The driver will pad NCM Transfer Blocks (NTBs) longer than this to tx_max, allowing the device to receive tx_max sized frames with no terminating short packet. NTBs shorter than this limit are transmitted as-is, without any padding, and are terminated with a short USB packet.

Padding to tx_max allows the driver to transmit NTBs back-to-back without any interleaving short USB packets. This reduces the number of short packet interrupts in the device, and represents a tradeoff between USB bus bandwidth and device DMA optimization.

Set to 0 to pad all frames. Set greater than tx_max to disable all padding.

/sys/class/net/<iface>/cdc_ncm/ndp_to_end

Defined on file *sysfs-class-net-cdc_ncm*

Boolean attribute showing the status of the ”NDP to end” quirk. Defaults to 'N', except for devices already known to need it enabled.

The ”NDP to end” quirk makes the driver place the NDP (the packet index table) after the payload. The NCM specification does not mandate this, but some devices are known to be more restrictive. Write 'Y' to this attribute for temporary testing of a suspect device failing to work with the default driver settings.

A device entry should be added to the driver if this quirk is found to be required.

/sys/class/net/<iface>/cdc_ncm/rx_max

Defined on file *sysfs-class-net-cdc_ncm*

The maximum NTB size for RX. Cannot exceed the maximum value supported by the device. Must allow at least one max sized datagram plus headers.

The actual limits are device dependent. See dwNtbInMaxSize.

Note: Some devices will silently ignore changes to this value, resulting in oversized NTBs and corresponding framing errors.

/sys/class/net/<iface>/cdc_ncm/tx_max

Defined on file *sysfs-class-net-cdc_ncm*

The maximum NTB size for TX. Cannot exceed the maximum value supported by the device. Must allow at least one max sized datagram plus headers.

The actual limits are device dependent. See dwNtbOutMaxSize.

/sys/class/net/<iface>/cdc_ncm/tx_timer_usecs

Defined on file *sysfs-class-net-cdc_ncm*

Datagram aggregation timeout in µs. The driver will wait up to 3 times this timeout for more datagrams to aggregate before transmitting an NTB frame.

Valid range: 5 to 4000000

Set to 0 to disable aggregation.

The following read-only attributes all represent fields of the structure defined in section 6.2.1 ”GetNtbParameters” of ”Universal Serial Bus Communications Class Subclass Specifications for Network Control Model Devices” (CDC NCM), Revision 1.0 (Errata 1), November 24, 2010 from USB Implementers Forum, Inc. The descriptions are quoted from table 6-3 of CDC NCM: ”NTB Parameter Structure”.

/sys/class/net/<iface>/cdc_ncm/wNdpInAlignment

Defined on file *sysfs-class-net-cdc_ncm*

NDP alignment modulus for NTBs on the IN pipe. Shall be a power of 2, and shall be at least 4.

/sys/class/net/<iface>/cdc_ncm/wNdpInDivisor

Defined on file *sysfs-class-net-cdc_ncm*

Divisor used for IN NTB Datagram payload alignment

/sys/class/net/<iface>/cdc_ncm/wNdpInPayloadRemainder

Defined on file *sysfs-class-net-cdc_ncm*

Remainder used to align input datagram payload within the NTB: (Payload Offset) mod (wNdpInDivisor) = wNdpInPayloadRemainder

/sys/class/net/<iface>/cdc_ncm/wNdpOutAlignment

Defined on file *sysfs-class-net-cdc_ncm*

NDP alignment modulus for use in NTBs on the OUT pipe. Shall be a power of 2, and shall be at least 4.

/sys/class/net/<iface>/cdc_ncm/wNdpOutDivisor

Defined on file *sysfs-class-net-cdc_ncm*

OUT NTB Datagram alignment modulus

/sys/class/net/<iface>/cdc_ncm/wNdpOutPayloadRemainder

Defined on file *sysfs-class-net-cdc_ncm*

Remainder used to align output datagram payload offsets within the NTB: Padding, shall be transmitted as zero by function, and ignored by host. (Payload Offset) mod (wNdpOutDivisor) = wNdpOutPayloadRemainder

/sys/class/net/<iface>/cdc_ncm/wNtbOutMaxDatagrams

Defined on file *sysfs-class-net-cdc_ncm*

Maximum number of datagrams that the host may pack into a single OUT NTB. Zero means that the device imposes no limit.

/sys/class/net/<iface>/dev_id

Defined on file *sysfs-class-net*

Indicates the device unique identifier. Format is an hexadecimal value. This is used to disambiguate interfaces which might be stacked (e.g: VLAN interfaces) but still have the same MAC address as their parent device.

/sys/class/net/<iface>/dev_port

Defined on file *sysfs-class-net*

Indicates the port number of this network device, formatted as a decimal value. Some NICs have multiple independent ports on the same PCI bus, device and function. This attribute allows userspace to distinguish the respective interfaces.

Note: some device drivers started to use 'dev_id' for this purpose since long before 3.15 and have not adopted the new attribute ever since. To query the port number, some tools look exclusively at 'dev_port', while others only consult 'dev_id'. If a network device has multiple client adapter ports as described in the previous paragraph and does not set this attribute to its port number, it's a kernel bug.

/sys/class/net/<iface>/dormant

Defined on file *sysfs-class-net*

Indicates whether the interface is in dormant state. Possible values are: 0: interface is not dormant 1: interface is dormant

This attribute can be used by supplicant software to signal that the device is not usable unless some supplicant-based authentication is performed (e.g: 802.1x). 'link_mode' attribute will also reflect the dormant state.

/sys/class/net/<iface>/dsa/tagging

Defined on file *sysfs-class-net-dsa*

On read, this file returns a string indicating the type of tagging protocol used by the DSA network devices that are attached to this master interface. On write, this file changes the tagging protocol of the attached DSA switches, if this operation is supported by the driver. Changing the tagging protocol must be done with the DSA interfaces and the master interface all administratively down. See the "name" field of each registered struct dsa_device_ops for a list of valid values.

/sys/class/net/<iface>/duplex

Defined on file *sysfs-class-net*

Indicates the interface latest or current duplex value. Possible values are:

half	half duplex
full	full duplex

Note: This attribute is only valid for interfaces that implement the ethtool get_link_ksettings method (mostly Ethernet).

/sys/class/net/<iface>/flags

Defined on file *sysfs-class-net*

Indicates the interface flags as a bitmask in hexadecimal. See include/uapi/linux/if.h for a list of all possible values and the flags semantics.

/sys/class/net/<iface>/fwinfo

Defined on file *sysfs-class-net-janz-ican3*

Firmware stamp of ican3 module Read-only: 32 byte string identification of the ICAN3 module (known values: "JANZ-ICAN3 ICANOS 1.xx", "JANZ-ICAN3 CAL/CANopen 1.xx")

/sys/class/net/<iface>/grcan/enable0

Defined on file *sysfs-class-net-grcan*

Hardware configuration of physical interface 0. This file reads and writes the "Enable 0" bit of the configuration register. Possible values: 0 or 1. See the GRCAN chapter of the GRLIB IP core library documentation for details. The default value is 0 or set by the module parameter grcan.enable0 and can be read at /sys/module/grcan/parameters/enable0.

/sys/class/net/<iface>/grcan/enable1

Defined on file *sysfs-class-net-grcan*

Hardware configuration of physical interface 1. This file reads and writes the "Enable 1" bit of the configuration register. Possible values: 0 or 1. See the GRCAN chapter of the GRLIB IP core library documentation for details. The default value is 0 or set by the module parameter grcan.enable1 and can be read at /sys/module/grcan/parameters/enable1.

/sys/class/net/<iface>/grcan/select

Defined on file *sysfs-class-net-grcan*

Configuration of which physical interface to be used. Possible values: 0 or 1. See the GRCAN chapter of the GRLIB IP core library documentation for details. The default value is 0 or is set by the module parameter grcan.select and can be read at /sys/module/grcan/parameters/select.

/sys/class/net/<iface>/ifalias

Defined on file *sysfs-class-net*

Indicates/stores an interface alias name as a string. This can be used for system management purposes.

/sys/class/net/<iface>/ifindex

Defined on file *sysfs-class-net*

Indicates the system-wide interface unique index identifier as a decimal number. This attribute is used for mapping an interface identifier to an interface name. It is used throughout the networking stack for specifying the interface specific requests/events.

/sys/class/net/<iface>/iflink

Defined on file *sysfs-class-net*

Indicates the system-wide interface unique index identifier a the interface is linked to. Format is decimal. This attribute is used to resolve interfaces chaining, linking and stacking. Physical interfaces have the same 'ifindex' and 'iflink' values.

/sys/class/net/<iface>/link_mode

Defined on file *sysfs-class-net*

Indicates the interface link mode, as a decimal number. This attribute should be used in conjunction with 'dormant' attribute to determine the interface usability. Possible values:

0	default link mode
1	dormant link mode

/sys/class/net/<iface>/mtu

Defined on file *sysfs-class-net*

Indicates the interface currently configured MTU value, in bytes, and in decimal format. Specific values depends on the lower-level interface protocol used. Ethernet devices will show a 'mtu' attribute value of 1500 unless changed.

/sys/class/net/<iface>/name_assign_type

Defined on file *sysfs-class-net*

Indicates the name assignment type. Possible values are:

1	enumerated by the kernel, possibly in an unpredictable way
2	predictably named by the kernel
3	named by userspace
4	renamed

/sys/class/net/<iface>/netdev_group

Defined on file *sysfs-class-net*

Indicates the interface network device group, as a decimal integer. Default value is 0 which corresponds to the initial network devices group. The group can be changed to affect routing decisions (see: net/ipv4/fib_rules and net/ipv6/fib6_rules.c).

/sys/class/net/<iface>/operstate

Defined on file *sysfs-class-net*

Indicates the interface RFC2863 operational state as a string.

Possible values are:

"unknown", "notpresent", "down", "lowerlayerdown", "testing", "dormant", "up".

/sys/class/net/<iface>/peak_usb/can_channel_id

Defined on file *sysfs-class-net-peak_usb*

PEAK PCAN-USB devices support user-configurable CAN channel identifiers. Contrary to a USB serial number, these identifiers are writable and can be set per CAN interface. This means that if a USB device exports multiple CAN interfaces, each of them can be assigned a unique channel ID. This attribute provides read-only access to the currently configured value of the channel identifier. Depending on the device type, the identifier has a length of 8 or 32 bit. The value read from this attribute is always an 8 digit 32 bit hexadecimal value in big endian format. If the device only supports an 8 bit identifier, the upper 24 bit of the value are set to zero.

/sys/class/net/<iface>/phydev

Defined on file *sysfs-class-net*

Symbolic link to the PHY device this network device is attached to.

/sys/class/net/<iface>/phys_port_id

Defined on file *sysfs-class-net*

Indicates the interface unique physical port identifier within the NIC, as a string.

/sys/class/net/<iface>/phys_port_name

Defined on file *sysfs-class-net*

Indicates the interface physical port name within the NIC, as a string.

/sys/class/net/<iface>/phys_switch_id

Defined on file *sysfs-class-net*

Indicates the unique physical switch identifier of a switch this port belongs to, as a string.

/sys/class/net/<iface>/qmi/add_mux

Defined on file *sysfs-class-net-qmi*

Unsigned integer.

Write a number ranging from 1 to 254 to add a qmap mux based network device, supported by recent Qualcomm based modems.

The network device will be called qmimux.

Userspace is in charge of managing the qmux network device activation and data stream setup on the modem side by using the proper QMI protocol requests.

/sys/class/net/<iface>/qmi/del_mux

Defined on file *sysfs-class-net-qmi*

Unsigned integer.

Write a number ranging from 1 to 254 to delete a previously created qmap mux based network device.

/sys/class/net/<iface>/qmi/pass_through

Defined on file *sysfs-class-net-qmi*

Boolean. Default: 'N'

Set this to 'Y' to enable 'pass-through' mode, allowing packets in MAP format to be passed on to the stack.

Normally the rmnet driver (CONFIG_RMNET) is then used to process and demultiplex these packets.

'Pass-through' mode can be enabled when the device is in 'raw-ip' mode only.

/sys/class/net/<iface>/qmi/raw_ip

Defined on file *sysfs-class-net-qmi*

Boolean. Default: 'N'

Set this to 'Y' to change the network device link framing from '802.3' to 'raw-ip'.

The netdev will change to reflect the link framing mode. The netdev is an ordinary ethernet device in '802.3' mode, and the driver expects to exchange frames with an ethernet header over

the USB link. The netdev is a headerless p-t-p device in 'raw-ip' mode, and the driver expects to echange IPv4 or IPv6 packets without any L2 header over the USB link.

Userspace is in full control of firmware configuration through the delegation of the QMI protocol. Userspace is responsible for coordination of driver and firmware link framing mode, changing this setting to 'Y' if the firmware is configured for 'raw-ip' mode.

/sys/class/net/<iface>/queues/rx-<queue>/rps_cpus

Defined on file *sysfs-class-net-queues*

Mask of the CPU(s) currently enabled to participate into the Receive Packet Steering packet processing flow for this network device queue. Possible values depend on the number of available CPU(s) in the system.

/sys/class/net/<iface>/queues/rx-<queue>/rps_flow_cnt

Defined on file *sysfs-class-net-queues*

Number of Receive Packet Steering flows being currently processed by this particular network device receive queue.

/sys/class/net/<iface>/queues/tx-<queue>/byte_queue_limits/hold_time

Defined on file *sysfs-class-net-queues*

Indicates the hold time in milliseconds to measure the slack of this particular network device transmit queue. Default value is 1000.

/sys/class/net/<iface>/queues/tx-<queue>/byte_queue_limits/inflight

Defined on file *sysfs-class-net-queues*

Indicates the number of bytes (objects) in flight on this network device transmit queue.

/sys/class/net/<iface>/queues/tx-<queue>/byte_queue_limits/limit

Defined on file *sysfs-class-net-queues*

Indicates the current limit of bytes allowed to be queued on this network device transmit queue. This value is clamped to be within the bounds defined by limit_max and limit_min.

/sys/class/net/<iface>/queues/tx-<queue>/byte_queue_limits/limit_max

Defined on file *sysfs-class-net-queues*

Indicates the absolute maximum limit of bytes allowed to be queued on this network device transmit queue. See include/linux/dynamic_queue_limits.h for the default value.

/sys/class/net/<iface>/queues/tx-<queue>/byte_queue_limits/limit_min

Defined on file *sysfs-class-net-queues*

Indicates the absolute minimum limit of bytes allowed to be queued on this network device transmit queue. Default value is 0.

/sys/class/net/<iface>/queues/tx-<queue>/tx_maxrate

Defined on file *sysfs-class-net-queues*

A Mbps max-rate set for the queue, a value of zero means disabled, default is disabled.

/sys/class/net/<iface>/queues/tx-<queue>/tx_timeout

Defined on file *sysfs-class-net-queues*

Indicates the number of transmit timeout events seen by this network interface transmit queue.

/sys/class/net/<iface>/queues/tx-<queue>/xps_cpus

Defined on file *sysfs-class-net-queues*

Mask of the CPU(s) currently enabled to participate into the Transmit Packet Steering packet processing flow for this network device transmit queue. Possible values depend on the number of available CPU(s) in the system.

/sys/class/net/<iface>/queues/tx-<queue>/xps_rxqs

Defined on file *sysfs-class-net-queues*

Mask of the receive queue(s) currently enabled to participate into the Transmit Packet Steering packet processing flow for this network device transmit queue. Possible values depend on the number of available receive queue(s) in the network device. Default is disabled.

/sys/class/net/<iface>/speed

Defined on file *sysfs-class-net*

Indicates the interface latest or current speed value. Value is an integer representing the link speed in Mbits/sec.

Note: this attribute is only valid for interfaces that implement the ethtool get_link_ksettings method (mostly Ethernet).

/sys/class/net/<iface>/statistics/collisions

Defined on file *sysfs-class-net-statistics*

Indicates the number of collisions seen by this network device. This value might not be relevant with all MAC layers.

/sys/class/net/<iface>/statistics/multicast

Defined on file *sysfs-class-net-statistics*

Indicates the number of multicast packets received by this network device.

/sys/class/net/<iface>/statistics/rx_bytes

Defined on file *sysfs-class-net-statistics*

Indicates the number of bytes received by this network device. See the network driver for the exact meaning of when this value is incremented.

/sys/class/net/<iface>/statistics/rx_compressed

Defined on file *sysfs-class-net-statistics*

Indicates the number of compressed packets received by this network device. This value might only be relevant for interfaces that support packet compression (e.g: PPP).

/sys/class/net/<iface>/statistics/rx_crc_errors

Defined on file *sysfs-class-net-statistics*

Indicates the number of packets received with a CRC (FCS) error by this network device. Note that the specific meaning might depend on the MAC layer used by the interface.

/sys/class/net/<iface>/statistics/rx_dropped

Defined on file *sysfs-class-net-statistics*

Indicates the number of packets received by the network device but dropped, that are not forwarded to the upper layers for packet processing. See the network driver for the exact meaning of this value.

/sys/class/net/<iface>/statistics/rx_errors

Defined on file *sysfs-class-net-statistics*

Indicates the number of receive errors on this network device. See the network driver for the exact meaning of this value.

/sys/class/net/<iface>/statistics/rx_fifo_errors

Defined on file *sysfs-class-net-statistics*

Indicates the number of receive FIFO errors seen by this network device. See the network driver for the exact meaning of this value.

/sys/class/net/<iface>/statistics/rx_frame_errors

Defined on file *sysfs-class-net-statistics*

Indicates the number of received frames with error, such as alignment errors. Note that the specific meaning depends on on the MAC layer protocol used. See the network driver for the exact meaning of this value.

/sys/class/net/<iface>/statistics/rx_length_errors

Defined on file *sysfs-class-net-statistics*

Indicates the number of received error packet with a length error, oversized or undersized. See the network driver for the exact meaning of this value.

/sys/class/net/<iface>/statistics/rx_missed_errors

Defined on file *sysfs-class-net-statistics*

Indicates the number of received packets that have been missed due to lack of capacity in the receive side. See the network driver for the exact meaning of this value.

/sys/class/net/<iface>/statistics/rx_nohandler

Defined on file *sysfs-class-net-statistics*

Indicates the number of received packets that were dropped on an inactive device by the network core.

/sys/class/net/<iface>/statistics/rx_over_errors

Defined on file *sysfs-class-net-statistics*

Indicates the number of received packets that are oversized compared to what the network device is configured to accept (e.g: larger than MTU). See the network driver for the exact meaning of this value.

/sys/class/net/<iface>/statistics/rx_packets

Defined on file *sysfs-class-net-statistics*

Indicates the total number of good packets received by this network device.

/sys/class/net/<iface>/statistics/tx_aborted_errors

Defined on file *sysfs-class-net-statistics*

Indicates the number of packets that have been aborted during transmission by a network device (e.g: because of a medium collision). See the network driver for the exact meaning of this value.

/sys/class/net/<iface>/statistics/tx_bytes

Defined on file *sysfs-class-net-statistics*

Indicates the number of bytes transmitted by a network device. See the network driver for the exact meaning of this value, in particular whether this accounts for all successfully transmitted packets or all packets that have been queued for transmission.

/sys/class/net/<iface>/statistics/tx_carrier_errors

Defined on file *sysfs-class-net-statistics*

Indicates the number of packets that could not be transmitted because of carrier errors (e.g: physical link down). See the network driver for the exact meaning of this value.

/sys/class/net/<iface>/statistics/tx_compressed

Defined on file *sysfs-class-net-statistics*

Indicates the number of transmitted compressed packets. Note this might only be relevant for devices that support compression (e.g: PPP).

/sys/class/net/<iface>/statistics/tx_dropped

Defined on file *sysfs-class-net-statistics*

Indicates the number of packets dropped during transmission. See the driver for the exact reasons as to why the packets were dropped.

/sys/class/net/<iface>/statistics/tx_errors

Defined on file *sysfs-class-net-statistics*

Indicates the number of packets in error during transmission by a network device. See the driver for the exact reasons as to why the packets were dropped.

/sys/class/net/<iface>/statistics/tx_fifo_errors

Defined on file *sysfs-class-net-statistics*

Indicates the number of packets having caused a transmit FIFO error. See the driver for the exact reasons as to why the packets were dropped.

/sys/class/net/<iface>/statistics/tx_heartbeat_errors

Defined on file *sysfs-class-net-statistics*

Indicates the number of packets transmitted that have been reported as heartbeat errors. See the driver for the exact reasons as to why the packets were dropped.

/sys/class/net/<iface>/statistics/tx_packets

Defined on file *sysfs-class-net-statistics*

Indicates the number of packets transmitted by a network device. See the driver for whether this reports the number of all attempted or successful transmissions.

/sys/class/net/<iface>/statistics/tx_window_errors

Defined on file *sysfs-class-net-statistics*

Indicates the number of packets not successfully transmitted due to a window collision. The specific meaning depends on the MAC layer used. On Ethernet this is usually used to report late collisions errors.

/sys/class/net/<iface>/termination

Defined on file *sysfs-class-net-janz-ican3*

Value representing the can bus termination

Default: 1 (termination active) Reading: get actual termination state Writing: set actual termination state (0=no termination, 1=termination active)

/sys/class/net/<iface>/testing

Defined on file *sysfs-class-net*

Indicates whether the interface is under test. Possible values are:

0	interface is not being tested
1	interface is being tested

When an interface is under test, it cannot be expected to pass packets as normal.

/sys/class/net/<iface>/threaded

Defined on file *sysfs-class-net*

Boolean value to control the threaded mode per device. User could set this value to enable/disable threaded mode for all napi belonging to this device, without the need to do device up/down.

Possible values: == ========== 0 threaded mode disabled for this dev 1 threaded mode enabled for this dev == ==========

/sys/class/net/<iface>/tx_queue_len

Defined on file *sysfs-class-net*

Indicates the interface transmit queue len in number of packets, as an integer value. Value depend on the type of interface, Ethernet network adapters have a default value of 1000 unless configured otherwise

/sys/class/net/<iface>/type

Defined on file *sysfs-class-net*

Indicates the interface protocol type as a decimal value. See include/uapi/linux/if_arp.h for all possible values.

/sys/class/net/<qmimux iface>/qmap/mux_id

Defined on file *sysfs-class-net-qmi*

Unsigned integer

Indicates the mux id associated to the qmimux network interface during its creation.

/sys/class/ocxl/<afu name>/afu_version

Defined on file *sysfs-class-ocxl*

read only Version of the AFU, in the format <major>:<minor> Reflects what is read in the configuration space of the AFU

/sys/class/ocxl/<afu name>/contexts

Defined on file *sysfs-class-ocxl*

read only Number of contexts for the AFU, in the format <n>/<max> where:

n	number of currently active contexts, for debug
max	maximum number of contexts supported by the AFU

/sys/class/ocxl/<afu name>/global_mmio_area

Defined on file *sysfs-class-ocxl*

read/write Give access the global mmio area for the AFU

/sys/class/ocxl/<afu name>/global_mmio_size

Defined on file *sysfs-class-ocxl*

read only Size of the global mmio area, as defined in the configuration space of the AFU

/sys/class/ocxl/<afu name>/pp_mmio_size

Defined on file *sysfs-class-ocxl*

read only Size of the per-process mmio area, as defined in the configuration space of the AFU

/sys/class/ocxl/<afu name>/reload_on_reset

Defined on file *sysfs-class-ocxl*

read/write Control whether the FPGA is reloaded on a link reset. Enabled through a vendor-specific logic block on the FPGA.

0	Do not reload FPGA image from flash
1	Reload FPGA image from flash
unavailable	The device does not support this capability

/sys/class/pktcdvd/add
/sys/class/pktcdvd/remove
/sys/class/pktcdvd/device_map

Defined on file *sysfs-class-pktcdvd*

add	(WO) Write a block device id (major:minor) to create a new pktcdvd device and map it to the block device.
remove	(WO) Write the pktcdvd device id (major:minor) to remove the pktcdvd device.
device_map	(RO) Shows the device mapping in format: pktcdvd[0-7] <pktdevid> <blkdevid>

/sys/class/pktcdvd/pktcdvd[0-7]/dev
/sys/class/pktcdvd/pktcdvd[0-7]/uevent

Defined on file *sysfs-class-pktcdvd*

dev: (RO) Device id

uevent: (WO) To send a uevent

/sys/class/pktcdvd/pktcdvd[0-7]/stat/packets_started
/sys/class/pktcdvd/pktcdvd[0-7]/stat/packets_finished
/sys/class/pktcdvd/pktcdvd[0-7]/stat/kb_written
/sys/class/pktcdvd/pktcdvd[0-7]/stat/kb_read
/sys/class/pktcdvd/pktcdvd[0-7]/stat/kb_read_gather
/sys/class/pktcdvd/pktcdvd[0-7]/stat/reset

Defined on file *sysfs-class-pktcdvd*

packets_started: (RO) Number of started packets.

packets_finished: (RO) Number of finished packets.

kb_written: (RO) kBytes written.

kb_read: (RO) kBytes read.

kb_read_gather: (RO) kBytes read to fill write packets.

reset: (WO) Write any value to it to reset
pktcdvd device statistic values, like bytes read/written.

/sys/class/pktcdvd/pktcdvd[0-7]/write_queue/size
/sys/class/pktcdvd/pktcdvd[0-7]/write_queue/congestion_off
/sys/class/pktcdvd/pktcdvd[0-7]/write_queue/congestion_on

Defined on file *sysfs-class-pktcdvd*

size	(RO) Contains the size of the bio write queue.
congestion_off	(RW) If bio write queue size is below this mark, accept new bio requests from the block layer.
congestion_on	(RW) If bio write queue size is higher as this mark, do no longer accept bio write requests from the block layer and wait till the pktcdvd device has processed enough bio's so that bio write queue size is below congestion off mark. A value of <= 0 disables congestion control.

Example:

To use the pktcdvd sysfs interface directly, you can do:

```
# create a new pktcdvd device mapped to /dev/hdc
echo "22:0" >/sys/class/pktcdvd/add
cat /sys/class/pktcdvd/device_map
# assuming device pktcdvd0 was created, look at stat's
cat /sys/class/pktcdvd/pktcdvd0/stat/kb_written
# print the device id of the mapped block device
fgrep pktcdvd0 /sys/class/pktcdvd/device_map
# remove device, using pktcdvd0 device id   253:0
echo "253:0" >/sys/class/pktcdvd/remove
```

/sys/class/power/ds2760-battery.*/charge_full

Defined on file *sysfs-class-power*

This file is writeable and can be used to set the assumed battery 'full level'. As batteries age, this value has to be amended over time.

/sys/class/power/ds2760-battery.*/charge_now

Defined on file *sysfs-class-power*

This file is writeable and can be used to set the current coloumb counter value inside the battery monitor chip. This is needed for unavoidable corrections of aging batteries. A userspace daemon can monitor the battery charging logic and once the counter drops out of considerable bounds, take appropriate action.

/sys/class/power_supply/<battery_name>/eppid

Defined on file *sysfs-platform-dell-wmi-ddv*

Reports the Dell ePPID (electronic Piece Part Identification) of the ACPI battery.

See /wmi/devices/dell-wmi-ddv for details.

/sys/class/power_supply/<supply_name>/alarm

Defined on file *sysfs-class-power-surface*

Battery trip point. When the remaining battery capacity crosses this value in either direction, the system will be notified and if necessary woken.

Set to zero to clear/disable.

Access: Read, Write

Valid values: In micro-Wh or micro-Ah, depending on the power unit of the battery

/sys/class/power_supply/<supply_name>/capacity

Defined on file *sysfs-class-power*

Fine grain representation of battery capacity.

Access: Read

Valid values: 0 - 100 (percent)

/sys/class/power_supply/<supply_name>/capacity_alert_max

Defined on file *sysfs-class-power*

Maximum battery capacity trip-wire value where the supply will notify user-space of the event. This is normally used for the battery discharging scenario where user-space needs to know the battery has dropped to an upper level so it can take appropriate action (e.g. warning user that battery level is low).

Access: Read, Write

Valid values: 0 - 100 (percent)

/sys/class/power_supply/<supply_name>/capacity_alert_min

Defined on file *sysfs-class-power*

Minimum battery capacity trip-wire value where the supply will notify user-space of the event. This is normally used for the battery discharging scenario where user-space needs to know the battery has dropped to a lower level so it can take appropriate action (e.g. warning user that battery level is critically low).

Access: Read, Write

Valid values: 0 - 100 (percent)

/sys/class/power_supply/<supply_name>/capacity_error_margin

Defined on file *sysfs-class-power*

Battery capacity measurement becomes unreliable without recalibration. This values provides the maximum error margin expected to exist by the fuel gauge in percent. Values close to 0% will be returned after (re-)calibration has happened. Over time the error margin will increase. 100% means, that the capacity related values are basically completely useless.

Access: Read

Valid values: 0 - 100 (percent)

/sys/class/power_supply/<supply_name>/capacity_level

Defined on file *sysfs-class-power*

Coarse representation of battery capacity.

Access: Read

Valid values:
"Unknown", "Critical", "Low", "Normal", "High", "Full"

/sys/class/power_supply/<supply_name>/charge_behaviour

Defined on file *sysfs-class-power*

Represents the charging behaviour.

Access: Read, Write

Valid values:

auto:	Charge normally, respect thresholds
inhibit-charge:	Do not charge while AC is attached
force-discharge:	Force discharge while AC is attached

/sys/class/power_supply/<supply_name>/charge_control_end_threshold

Defined on file *sysfs-class-power*

Represents a battery percentage level, above which charging will stop. Not all hardware is capable of setting this to an arbitrary percentage. Drivers will round written values to the nearest supported value. Reading back the value will show the actual threshold set by the driver.

Access: Read, Write

Valid values: 0 - 100 (percent)

/sys/class/power_supply/<supply_name>/charge_control_limit

Defined on file *sysfs-class-power*

Maximum allowable charging current. Used for charge rate throttling for thermal cooling or improving battery health.

Access: Read, Write

Valid values: Represented in microamps

/sys/class/power_supply/<supply_name>/charge_control_limit_max

Defined on file *sysfs-class-power*

Maximum legal value for the charge_control_limit property.

Access: Read

Valid values: Represented in microamps

/sys/class/power_supply/<supply_name>/charge_control_start_threshold

Defined on file *sysfs-class-power*

Represents a battery percentage level, below which charging will begin.

Access: Read, Write Valid values: 0 - 100 (percent)

/sys/class/power_supply/<supply_name>/charge_term_current

Defined on file *sysfs-class-power*

Reports the charging current value which is used to determine when the battery is considered full and charging should end.

Access: Read

Valid values: Represented in microamps

/sys/class/power_supply/<supply_name>/charge_type

Defined on file *sysfs-class-power*

Represents the type of charging currently being applied to the battery. "Trickle", "Fast", and "Standard" all mean different charging speeds. "Adaptive" means that the charger uses some algorithm to adjust the charge rate dynamically, without any user configuration required. "Custom" means that the charger uses the charge_control_* properties as configuration for some different algorithm. "Long Life" means the charger reduces its charging rate in order to prolong the battery health. "Bypass" means the charger bypasses the charging path around the integrated converter allowing for a "smart" wall adaptor to perform the power conversion externally.

Access: Read, Write

Valid values:

"Unknown", "N/A", "Trickle", "Fast", "Standard", "Adaptive", "Custom", "Long Life", "Bypass"

/sys/class/power_supply/<supply_name>/current_avg

Defined on file *sysfs-class-power*

Battery:

Reports an average IBAT current reading for the battery, over a fixed period. Normally devices will provide a fixed interval in which they average readings to smooth out the reported value.

USB:

Reports an average IBUS current reading over a fixed period. Normally devices will provide a fixed interval in which they average readings to smooth out the reported value.

Access: Read

Valid values: Represented in microamps. Negative values are used for discharging batteries, positive values for charging batteries and for USB IBUS current.

/sys/class/power_supply/<supply_name>/current_max

Defined on file *sysfs-class-power*

Battery:

Reports the maximum IBAT current allowed into the battery.

USB:

Reports the maximum IBUS current the supply can support.

Access: Read Valid values: Represented in microamps

/sys/class/power_supply/<supply_name>/current_now

Defined on file *sysfs-class-power*

Battery:

Reports an instant, single IBAT current reading for the battery. This value is not averaged/smoothed.

Access: Read

USB:

Reports the IBUS current supplied now. This value is generally read-only reporting, unless the 'online' state of the supply is set to be programmable, in which case this value can be set within the reported min/max range.

Access: Read, Write

Valid values: Represented in microamps. Negative values are used for discharging batteries, positive values for charging batteries and for USB IBUS current.

/sys/class/power_supply/<supply_name>/cycle_count

Defined on file *sysfs-class-power*

Reports the number of full charge + discharge cycles the battery has undergone.

Access: Read

Valid values:
Integer > 0: representing full cycles Integer = 0: cycle_count info is not available

USB Properties

/sys/class/power_supply/<supply_name>/health

Defined on file *sysfs-class-power*

Reports the health of the battery or battery side of charger functionality.

Access: Read

Valid values:
"Unknown", "Good", "Overheat", "Dead", "Over voltage", "Unspecified failure", "Cold", "Watchdog timer expire", "Safety timer expire", "Over current", "Calibration required", "Warm", "Cool", "Hot", "No battery"

/sys/class/power_supply/<supply_name>/input_current_limit

Defined on file *sysfs-class-power*

Details the incoming IBUS current limit currently set in the supply. Normally this is configured based on the type of connection made (e.g. A configured SDP should output a maximum of 500mA so the input current limit is set to the same value). Use preferably input_power_limit, and for problems that can be solved using power limit use input_current_limit.

Access: Read, Write

Valid values: Represented in microamps

/sys/class/power_supply/<supply_name>/input_power_limit

Defined on file *sysfs-class-power*

This entry configures the incoming power limit currently set in the supply. Normally this is configured based on system-level knowledge or user input. Use preferably this feature to limit the incoming power and use current/voltage limit only for problems that can be solved using power limit.

Access: Read, Write

Valid values: Represented in microwatts

/sys/class/power_supply/<supply_name>/input_voltage_limit

Defined on file *sysfs-class-power*

This entry configures the incoming VBUS voltage limit currently set in the supply. Normally this is configured based on system-level knowledge or user input (e.g. This is part of the Pixel C's thermal management strategy to effectively limit the input power to 5V when the screen is on to meet Google's skin temperature targets). Note that this feature should not be used for safety critical things. Use preferably input_power_limit, and for problems that can be solved using power limit use input_voltage_limit.

Access: Read, Write

Valid values: Represented in microvolts

/sys/class/power_supply/<supply_name>/manufacture_day

Defined on file *sysfs-class-power*

Reports the day of month when the device has been manufactured.

Access: Read Valid values: 1-31

/sys/class/power_supply/<supply_name>/manufacture_month

Defined on file *sysfs-class-power*

Reports the month when the device has been manufactured.

Access: Read

Valid values: 1-12

/sys/class/power_supply/<supply_name>/manufacture_year

Defined on file *sysfs-class-power*

Reports the year (following Gregorian calendar) when the device has been manufactured.

Access: Read

Valid values: Reported as integer

/sys/class/power_supply/<supply_name>/manufacturer

Defined on file *sysfs-class-power*

Reports the name of the device manufacturer.

Access: Read Valid values: Represented as string

/sys/class/power_supply/<supply_name>/model_name

Defined on file *sysfs-class-power*

Reports the name of the device model.

Access: Read Valid values: Represented as string

/sys/class/power_supply/<supply_name>/online

Defined on file *sysfs-class-power*

Indicates if VBUS is present for the supply. When the supply is online, and the supply allows it, then it's possible to switch between online states (e.g. Fixed -> Programmable for a PD_PPS USB supply so voltage and current can be controlled).

Access: Read, Write

Valid values:

0:	Offline
1:	Online Fixed - Fixed Voltage Supply
2:	Online Programmable - Programmable Voltage Supply

/sys/class/power_supply/<supply_name>/precharge_current

Defined on file *sysfs-class-power*

Reports the charging current applied during pre-charging phase for a battery charge cycle.

Access: Read

Valid values: Represented in microamps

/sys/class/power_supply/<supply_name>/present

Defined on file *sysfs-class-power*

Reports whether a battery is present or not in the system. If the property does not exist, the battery is considered to be present.

Access: Read

Valid values:

0:	Absent
1:	Present

/sys/class/power_supply/<supply_name>/serial_number

Defined on file *sysfs-class-power*

Reports the serial number of the device.

Access: Read Valid values: Represented as string

/sys/class/power_supply/<supply_name>/status

Defined on file *sysfs-class-power*

Represents the charging status of the battery. Normally this is read-only reporting although for some supplies this can be used to enable/disable charging to the battery.

Access: Read, Write

Valid values:
"Unknown", "Charging", "Discharging", "Not charging", "Full"

/sys/class/power_supply/<supply_name>/technology

Defined on file *sysfs-class-power*

Describes the battery technology supported by the supply.

Access: Read

Valid values:
"Unknown", "NiMH", "Li-ion", "Li-poly", "LiFe", "NiCd", "LiMn"

/sys/class/power_supply/<supply_name>/temp

Defined on file *sysfs-class-power*

Battery:

Reports the current TBAT battery temperature reading.

USB:

Reports the current supply temperature reading. This would normally be the internal temperature of the device itself (e.g TJUNC temperature of an IC)

Access: Read

Valid values: Represented in 1/10 Degrees Celsius

/sys/class/power_supply/<supply_name>/temp_alert_max

Defined on file *sysfs-class-power*

Battery:

Maximum TBAT temperature trip-wire value where the supply will notify user-space of the event.

USB:

Maximum supply temperature trip-wire value where the supply will notify user-space of the event.

This is normally used for the charging scenario where user-space needs to know if the temperature has crossed an upper threshold so it can take appropriate action (e.g. warning user that the temperature is critically high, and charging has stopped).

Access: Read

Valid values: Represented in 1/10 Degrees Celsius

/sys/class/power_supply/<supply_name>/temp_alert_min

Defined on file *sysfs-class-power*

Battery:

Minimum TBAT temperature trip-wire value where the supply will notify user-space of the event.

USB:

Minimum supply temperature trip-wire value where the supply will notify user-space of the event.

This is normally used for the charging scenario where user-space needs to know if the temperature has crossed a lower threshold so it can take appropriate action (e.g. warning user that temperature level is high, and charging current has been reduced accordingly to remedy the situation).

Access: Read

Valid values: Represented in 1/10 Degrees Celsius

/sys/class/power_supply/<supply_name>/temp_max

Defined on file *sysfs-class-power*

Battery:

Reports the maximum allowed TBAT battery temperature for charging.

USB:

Reports the maximum allowed supply temperature for operation.

Access: Read

Valid values: Represented in 1/10 Degrees Celsius

/sys/class/power_supply/<supply_name>/temp_min

Defined on file *sysfs-class-power*

Battery:

Reports the minimum allowed TBAT battery temperature for charging.

USB:

Reports the minimum allowed supply temperature for operation.

Access: Read

Valid values: Represented in 1/10 Degrees Celsius

/sys/class/power_supply/<supply_name>/type

Defined on file *sysfs-class-power*

Describes the main type of the supply.

Access: Read Valid values: ”Battery”, ”UPS”, ”Mains”, ”USB”, ”Wireless”

Battery and USB properties

/sys/class/power_supply/<supply_name>/usb_type

Defined on file *sysfs-class-power*

Reports what type of USB connection is currently active for the supply, for example it can show if USB-PD capable source is attached.

Access: Read-Only

Valid values:
”Unknown”, ”SDP”, ”DCP”, ”CDP”, ”ACA”, ”C”, ”PD”, ”PD_DRP”, ”PD_PPS”, ”BrickID”

Device Specific Properties

/sys/class/power_supply/<supply_name>/voltage_avg

Defined on file *sysfs-class-power*

Reports an average VBAT voltage reading for the battery, over a fixed period. Normally devices will provide a fixed interval in which they average readings to smooth out the reported value.

Access: Read

Valid values: Represented in microvolts

/sys/class/power_supply/<supply_name>/voltage_max

Defined on file *sysfs-class-power*

Battery:

Reports the maximum safe VBAT voltage permitted for the battery, during charging.

USB:

Reports the maximum VBUS voltage the supply can support.

Access: Read

Valid values: Represented in microvolts

/sys/class/power_supply/<supply_name>/voltage_min

Defined on file *sysfs-class-power*

Battery:

Reports the minimum safe VBAT voltage permitted for the battery, during discharging.

USB:

Reports the minimum VBUS voltage the supply can support.

Access: Read

Valid values: Represented in microvolts

/sys/class/power_supply/<supply_name>/voltage_now

Defined on file *sysfs-class-power*

Battery:

Reports an instant, single VBAT voltage reading for the battery. This value is not averaged/smoothed.

Access: Read

USB:

Reports the VBUS voltage supplied now. This value is generally read-only reporting, unless the 'online' state of the supply is set to be programmable, in which case this value can be set within the reported min/max range.

Access: Read, Write

Valid values: Represented in microvolts

Battery Properties

/sys/class/power_supply/bq24257-charger/high_impedance_enable

Defined on file *sysfs-class-power*

This entry allows enabling the high-impedance mode of bq24257-type charger devices. If enabled, it places the charger IC into low power standby mode with the switch mode controller disabled. When disabled, the charger operates normally. See device datasheet for details.

Valid values:

- 1: enabled
- 0: disabled

/sys/class/power_supply/bq24257-charger/in_dpm_voltage

Defined on file *sysfs-class-power*

This entry configures the input dynamic power path management voltage of bq24257-type charger devices. Once the supply drops to the configured voltage, the input current limit is reduced down to prevent the further drop of the supply. When the IC enters this mode, the charge current is lower than the set value. See device datasheet for details.

Valid values:

- 4200000, 4280000, 4360000, 4440000, 4520000, 4600000, 4680000, 4760000 (all uV)

/sys/class/power_supply/bq24257-charger/ovp_voltage

Defined on file *sysfs-class-power*

This entry configures the overvoltage protection feature of bq24257- type charger devices. This feature protects the device and other components against damage from overvoltage on the input supply. See device datasheet for details.

Valid values:

- 6000000, 6500000, 7000000, 8000000, 9000000, 9500000, 10000000, 10500000 (all uV)

/sys/class/power_supply/bq24257-charger/sysoff_enable

Defined on file *sysfs-class-power*

This entry allows enabling the sysoff mode of bq24257-type charger devices. If enabled and the input is removed, the internal battery FET is turned off in order to reduce the leakage from the BAT pin to less than 1uA. Note that on some devices/systems this disconnects the battery from the system. See device datasheet for details.

Valid values:

- 1: enabled
- 0: disabled

/sys/class/power_supply/ltc4162-l/arm_ship_mode

Defined on file *sysfs-class-power-ltc4162l*

The charger will normally drain the battery while inactive, typically drawing about 54 microamps. Write a "1" to this property to arm a special "ship" mode that extends shelf life by reducing the leakage to about 2.8 microamps. The chip will remain in this mode (and no longer respond to I2C commands) until some external power-supply is attached raising the input voltage above 1V. It will then automatically revert to "0". Writing a "0" to the property cancels the "ship" mode request. The ship mode, when armed, activates once the input voltage drops below 1V.

Access: Read, Write

Valid values: 0 (disable) or 1 (enable)

/sys/class/power_supply/ltc4162-l/charge_status

Defined on file *sysfs-class-power-ltc4162l*

Detailed charge status information as reported by the chip.

Access: Read

Valid values:

ilim_reg_active thermal_reg_active vin_uvcl_active iin_limit_active constant_current constant_voltage charger_off

/sys/class/power_supply/ltc4162-l/force_telemetry

Defined on file *sysfs-class-power-ltc4162l*

To save battery current, the measurement system is disabled if the battery is the only source of power. This affects all voltage, current and temperature measurements. Write a "1" to this to keep performing telemetry once every few seconds, even when running on battery (as reported by the online property, which is "1" when external power is available and "0" when the system runs on battery).

Access: Read, Write

Valid values: 0 (disabled) or 1 (enabled)

/sys/class/power_supply/ltc4162-l/ibat

Defined on file *sysfs-class-power-ltc4162l*

Battery input current as measured by the charger. Negative value means that the battery is discharging.

Access: Read

Valid values: Signed value in microamps

/sys/class/power_supply/ltc4162-l/vbat

Defined on file *sysfs-class-power-ltc4162l*

Battery voltage as measured by the charger.

Access: Read

Valid values: In microvolts

/sys/class/power_supply/ltc4162-l/vbat_avg

Defined on file *sysfs-class-power-ltc4162l*

Battery voltage, averaged over time, as measured by the charger.

Access: Read

Valid values: In microvolts

/sys/class/power_supply/max14577-charger/device/fast_charge_timer

Defined on file *sysfs-class-power*

This entry shows and sets the maximum time the max14577 charger operates in fast-charge mode. When the timer expires the device will terminate fast-charge mode (charging current will drop to 0 A) and will trigger interrupt.

Valid values:

- 5, 6 or 7 (hours),
- 0: disabled.

/sys/class/power_supply/max77693-charger/device/fast_charge_timer

Defined on file *sysfs-class-power*

This entry shows and sets the maximum time the max77693 charger operates in fast-charge mode. When the timer expires the device will terminate fast-charge mode (charging current will drop to 0 A) and will trigger interrupt.

Valid values:

- 4 - 16 (hours), step by 2 (rounded down)
- 0: disabled.

/sys/class/power_supply/max77693-charger/device/top_off_threshold_current

Defined on file *sysfs-class-power*

This entry shows and sets the charging current threshold for entering top-off charging mode. When charging current in fast charge mode drops below this value, the charger will trigger interrupt and start top-off charging mode.

Valid values:

- 100000 - 200000 (microamps), step by 25000 (rounded down)
- 200000 - 350000 (microamps), step by 50000 (rounded down)

- 0: disabled.

/sys/class/power_supply/max77693-charger/device/top_off_timer

Defined on file *sysfs-class-power*

This entry shows and sets the maximum time the max77693 charger operates in top-off charge mode. When the timer expires the device will terminate top-off charge mode (charging current will drop to 0 A) and will trigger interrupt.

Valid values:

- 0 - 70 (minutes), step by 10 (rounded down)

/sys/class/power_supply/mp2629_battery/batt_impedance_compen

Defined on file *sysfs-class-power-mp2629*

Represents a battery impedance compensation to accelerate charging.

Access: Read, Write

Valid values: Represented in milli-ohms. Valid range is [0, 140].

/sys/class/power_supply/rt9467-*/sysoff_enable

Defined on file *sysfs-class-power-rt9467*

This entry allows enabling the sysoff mode of rt9467 charger devices. If enabled and the input is removed, the internal battery FET is turned off to reduce the leakage from the BAT pin. See device datasheet for details. It's commonly used when the product enter shipping stage. After entering shipping mode, only 'VBUS' or 'Power key" pressed can make it leave this mode. 'Disable' also can help to leave it, but it's more like to abort the action before the device really enter shipping mode.

Access: Read, Write Valid values: - 1: enabled - 0: disabled

/sys/class/power_supply/rt9471-*/port_detect_enable

Defined on file *sysfs-class-power-rt9471*

This entry allows enabling the USB BC12 port detect function of rt9471 charger devices. If enabled and VBUS is inserted, device will start to do the BC12 port detect and report the usb port type when port detect is done. See datasheet for details. Normally controlled when TypeC/USBPD port integrated.

Access: Read, Write Valid values: - 1: enabled - 0: disabled

/sys/class/power_supply/rt9471-*/sysoff_enable

Defined on file *sysfs-class-power-rt9471*

This entry allows enabling the sysoff mode of rt9471 charger devices. If enabled and the input is removed, the internal battery FET is turned off to reduce the leakage from the BAT pin. See device datasheet for details. It's commonly used when the product enter shipping stage.

After entering shipping mode, only 'VBUS' or 'Power key" pressed can make it leave this mode. 'Disable' also can help to leave it, but it's more like to abort the action before the device really enter shipping mode.

Access: Read, Write Valid values: - 1: enabled - 0: disabled

/sys/class/power_supply/twl4030_ac/mode

Defined on file *sysfs-class-power-twl4030*

Changing mode for 'ac' port. Writing to this can disable charging.

Possible values are:

"auto"	draw power as appropriate for detected power source and battery status.
"off"	do not draw any power.

/sys/class/power_supply/twl4030_usb/mode

Defined on file *sysfs-class-power-twl4030*

Changing mode for USB port. Writing to this can disable charging.

Possible values are:

"auto"	draw power as appropriate for detected power source and battery status.
"off"	do not draw any power.
"continuous"	activate mode described as "linear" in TWL data sheets. This uses whatever current is available and doesn't switch off when voltage drops. This is useful for unstable power sources such as bicycle dynamo, but care should be taken that battery is not over-charged.

/sys/class/power_supply/wilco-charger/charge_control_end_threshold

Defined on file *sysfs-class-power-wilco*

Used when charge_type="Custom", as described above. Measured in percentages. The valid range is [55, 100].

/sys/class/power_supply/wilco-charger/charge_control_start_threshold

Defined on file *sysfs-class-power-wilco*

Used when charge_type="Custom", as described above. Measured in percentages. The valid range is [50, 95].

/sys/class/power_supply/wilco-charger/charge_type

Defined on file *sysfs-class-power-wilco*

What charging algorithm to use:

Standard:
Fully charges battery at a standard rate.

Adaptive:
Battery settings adaptively optimized based on typical battery usage pattern.

Fast:
Battery charges over a shorter period.

Trickle:
Extends battery lifespan, intended for users who primarily use their Chromebook while connected to AC.

Custom:
A low and high threshold percentage is specified. Charging begins when level drops below charge_control_start_threshold, and ceases when level is above charge_control_end_threshold.

Long Life:
Customized charge rate for last longer battery life. On Wilco device this mode is pre-configured in the factory through EC's private PID. Switching to a different mode will be denied by Wilco EC when Long Life mode is enabled.

/sys/class/powercap/

Defined on file *sysfs-class-powercap*

The powercap/ class sub directory belongs to the power cap subsystem. Refer to /power/powercap/powercap for details.

/sys/class/powercap/.../<power zone>/constraint_X_max_time_window_us

Defined on file *sysfs-class-powercap*

Maximum allowed time window in micro seconds for this constraint. Here "X" can have values from 0 to max integer.

/sys/class/powercap/.../<power zone>/constraint_X_min_time_window_us

Defined on file *sysfs-class-powercap*

Minimum allowed time window in micro seconds for this constraint. Here "X" can have values from 0 to max integer.

/sys/class/powercap/.../<power zone>/constraint_X_name

Defined on file *sysfs-class-powercap*

Each power zone can define one or more constraints. Each constraint can have an optional name. Here "X" can have values from 0 to max integer.

/sys/class/powercap/.../<power zone>/constraint_X_power_limit_uw

Defined on file *sysfs-class-powercap*

Power limit in micro-watts should be applicable for the time window specified by "constraint_X_time_window_us". Here "X" can have values from 0 to max integer.

/sys/class/powercap/.../<power zone>/constraint_X_time_window_us

Defined on file *sysfs-class-powercap*

Time window in micro seconds. This is used along with constraint_X_power_limit_uw to define a power constraint. Here "X" can have values from 0 to max integer.

/sys/class/powercap/.../<power zone>/enabled

Defined on file *sysfs-class-powercap*

This allows to enable/disable power capping at power zone level. This applies to current power zone and its children.

/sys/class/powercap/.../<power zone>/energy_uj

Defined on file *sysfs-class-powercap*

Current energy counter in micro-joules. Write "0" to reset. If the counter can not be reset, then this attribute is read-only.

/sys/class/powercap/.../<power zone>/max_energy_range_uj

Defined on file *sysfs-class-powercap*

Range of the above energy counter in micro-joules.

/sys/class/powercap/.../<power zone>/max_power_range_uw

Defined on file *sysfs-class-powercap*

Range of the above power value in micro-watts.

/sys/class/powercap/.../<power zone>/name

Defined on file *sysfs-class-powercap*

Specifies the name of this power zone.

/sys/class/powercap/.../<power zone>/power_uw

Defined on file *sysfs-class-powercap*

Current power in micro-watts.

/sys/class/powercap/<control type>

Defined on file *sysfs-class-powercap*

A <control type> is a unique name under /sys/class/powercap. Here <control type> determines how the power is going to be controlled. A <control type> can contain multiple power zones.

/sys/class/powercap/<control type>/.../constraint_X_max_power_uw

Defined on file *sysfs-class-powercap*

Maximum allowed power in micro watts for this constraint. Here "X" can have values from 0 to max integer.

/sys/class/powercap/<control type>/.../constraint_X_min_power_uw

Defined on file *sysfs-class-powercap*

Minimum allowed power in micro watts for this constraint. Here "X" can have values from 0 to max integer.

/sys/class/powercap/<control type>/<power zone>

Defined on file *sysfs-class-powercap*

A power zone is a single or a collection of devices, which can be independently monitored and controlled. A power zone sysfs entry is qualified with the name of the <control type>. E.g. intel-rapl:0:1:1.

/sys/class/powercap/<control type>/<power zone>/<child power zone>

Defined on file *sysfs-class-powercap*

Power zones may be organized in a hierarchy in which child power zones provide monitoring and control for a subset of devices under the parent. For example, if there is a parent power zone for a whole CPU package, each CPU core in it can be a child power zone.

/sys/class/powercap/<control type>/enabled

Defined on file *sysfs-class-powercap*

This allows to enable/disable power capping for a "control type". This status affects every power zone using this "control_type.

/sys/class/pps/

Defined on file *sysfs-pps*

The */sys/class/pps/* directory will contain files and directories that will provide a unified interface to the PPS sources.

/sys/class/pps/ppsX/

Defined on file *sysfs-pps*

The */sys/class/pps/ppsX/* directory is related to X-th PPS source into the system. Each directory will contain files to manage and control its PPS source.

/sys/class/pps/ppsX/assert

Defined on file *sysfs-pps*

The */sys/class/pps/ppsX/assert* file reports the assert events and the assert sequence number of the X-th source in the form:

<secs>.<nsec>#<sequence>

If the source has no assert events the content of this file is empty.

/sys/class/pps/ppsX/clear

Defined on file *sysfs-pps*

The */sys/class/pps/ppsX/clear* file reports the clear events and the clear sequence number of the X-th source in the form:

<secs>.<nsec>#<sequence>

If the source has no clear events the content of this file is empty.

/sys/class/pps/ppsX/echo

Defined on file *sysfs-pps*

The */sys/class/pps/ppsX/echo* file reports if the X-th does or does not support an "echo" function.

/sys/class/pps/ppsX/mode

Defined on file *sysfs-pps*

The */sys/class/pps/ppsX/mode* file reports the functioning mode of the X-th source in hexadecimal encoding.

Please, refer to linux/include/linux/pps.h for further info.

/sys/class/pps/ppsX/name

Defined on file *sysfs-pps*

The */sys/class/pps/ppsX/name* file reports the name of the X-th source.

/sys/class/pps/ppsX/path

Defined on file *sysfs-pps*

The */sys/class/pps/ppsX/path* file reports the path name of the device connected with the X-th source.

If the source is not connected with any device the content of this file is empty.

/sys/class/ptp/

Defined on file *sysfs-ptp*

This directory contains files and directories providing a standardized interface to the ancillary features of PTP hardware clocks.

/sys/class/ptp/ptp<N>/

Defined on file *sysfs-ptp*

This directory contains the attributes of the Nth PTP hardware clock registered into the PTP class driver subsystem.

/sys/class/ptp/ptp<N>/clock_name

Defined on file *sysfs-ptp*

This file contains the name of the PTP hardware clock as a human readable string. The purpose of this attribute is to provide the user with a "friendly name" and to help distinguish PHY based devices from MAC based ones. The string does not necessarily have to be any kind of unique id.

/sys/class/ptp/ptp<N>/extts_enable

Defined on file *sysfs-ptp*

This write-only file enables or disables external timestamps. To enable external timestamps, write the channel index followed by a "1" into the file. To disable external timestamps, write the channel index followed by a "0" into the file.

/sys/class/ptp/ptp<N>/fifo

Defined on file *sysfs-ptp*

This file provides timestamps on external events, in the form of three integers: channel index, seconds, and nanoseconds.

/sys/class/ptp/ptp<N>/max_adjustment

Defined on file *sysfs-ptp*

This file contains the PTP hardware clock's maximum frequency adjustment value (a positive integer) in parts per billion.

/sys/class/ptp/ptp<N>/max_vclocks

Defined on file *sysfs-ptp*

This file contains the maximum number of ptp vclocks. Write integer to re-configure it.

/sys/class/ptp/ptp<N>/n_alarms

Defined on file *sysfs-ptp*

This file contains the number of periodic or one shot alarms offer by the PTP hardware clock.

/sys/class/ptp/ptp<N>/n_external_timestamps

Defined on file *sysfs-ptp*

This file contains the number of external timestamp channels offered by the PTP hardware clock.

/sys/class/ptp/ptp<N>/n_periodic_outputs

Defined on file *sysfs-ptp*

This file contains the number of programmable periodic output channels offered by the PTP hardware clock.

/sys/class/ptp/ptp<N>/n_pins

Defined on file *sysfs-ptp*

This file contains the number of programmable pins offered by the PTP hardware clock.

/sys/class/ptp/ptp<N>/n_vclocks

Defined on file *sysfs-ptp*

This file contains the number of virtual PTP clocks in use. By default, the value is 0 meaning that only the physical clock is in use. Setting the value creates the corresponding number of virtual clocks and causes the physical clock to become free running. Setting the value back to 0 deletes the virtual clocks and switches the physical clock back to normal, adjustable operation.

/sys/class/ptp/ptp<N>/period

Defined on file *sysfs-ptp*

This write-only file enables or disables periodic outputs. To enable a periodic output, write five integers into the file: channel index, start time seconds, start time nanoseconds, period seconds, and period nanoseconds. To disable a periodic output, set all the seconds and nanoseconds values to zero.

/sys/class/ptp/ptp<N>/pins

Defined on file *sysfs-ptp*

This directory contains one file for each programmable pin offered by the PTP hardware clock. The file name is the hardware dependent pin name. Reading from this file produces two numbers, the assigned function (see the *PTP_PF_* enumeration values in linux/ptp_clock.h) and the channel number. The function and channel assignment may be changed by two writing numbers into the file.

/sys/class/ptp/ptp<N>/pps_available

Defined on file *sysfs-ptp*

This file indicates whether the PTP hardware clock supports a Pulse Per Second to the host CPU. Reading "1" means that the PPS is supported, while "0" means not supported.

/sys/class/ptp/ptp<N>/pps_enable

Defined on file *sysfs-ptp*

This write-only file enables or disables delivery of PPS events to the Linux PPS subsystem. To enable PPS events, write a "1" into the file. To disable events, write a "0" into the file.

/sys/class/pwm/

Defined on file *sysfs-class-pwm*

The pwm/ class sub-directory belongs to the Generic PWM Framework and provides a sysfs interface for using PWM channels.

/sys/class/pwm/pwmchip<N>/

Defined on file *sysfs-class-pwm*

A /sys/class/pwm/pwmchipN directory is created for each probed PWM controller/chip where N is the base of the PWM chip.

/sys/class/pwm/pwmchip<N>/export

Defined on file *sysfs-class-pwm*

Exports a PWM channel from the PWM chip for sysfs control. Value is between 0 and /sys/class/pwm/pwmchipN/npwm - 1.

/sys/class/pwm/pwmchip<N>/npwm

Defined on file *sysfs-class-pwm*

The number of PWM channels supported by the PWM chip.

/sys/class/pwm/pwmchip<N>/pwmX

Defined on file *sysfs-class-pwm*

A /sys/class/pwm/pwmchipN/pwmX directory is created for each exported PWM channel where X is the exported PWM channel number.

/sys/class/pwm/pwmchip<N>/pwmX/capture

Defined on file *sysfs-class-pwm*

Capture information about a PWM signal. The output format is a pair unsigned integers (period and duty cycle), separated by a single space.

/sys/class/pwm/pwmchip<N>/pwmX/duty_cycle

Defined on file *sysfs-class-pwm*

Sets the PWM signal duty cycle in nanoseconds.

/sys/class/pwm/pwmchip<N>/pwmX/enable

Defined on file *sysfs-class-pwm*

Enable/disable the PWM signal. 0 is disabled 1 is enabled

/sys/class/pwm/pwmchip<N>/pwmX/period

Defined on file *sysfs-class-pwm*

Sets the PWM signal period in nanoseconds.

/sys/class/pwm/pwmchip<N>/pwmX/polarity

Defined on file *sysfs-class-pwm*

Sets the output polarity of the PWM signal to ”normal” or ”inversed”.

/sys/class/pwm/pwmchip<N>/unexport

Defined on file *sysfs-class-pwm*

Unexports a PWM channel.

/sys/class/rapidio_port

Defined on file *sysfs-class-rapidio*

On-chip RapidIO controllers and PCIe-to-RapidIO bridges (referenced as ”Master Port” or ”mport”) are presented in sysfs as the special class of devices: ”rapidio_port”. The */sys/class/rapidio_port* subdirectory contains individual subdirectories named as ”rapidioN” where N = mport ID registered with RapidIO subsystem.

NOTE: An mport ID is not a RapidIO destination ID assigned to a given local mport device.

/sys/class/rapidio_port/rapidio<N>/port_destid

Defined on file *sysfs-class-rapidio*

(RO) reports RapidIO destination ID assigned to the given RapidIO mport device. If value 0xFFFFFFFF is returned this means that no valid destination ID have been assigned to the mport (yet). Normally, before enumeration/discovery have been executed only fabric enumerating mports have a valid destination ID assigned to them using "hdid=..." rapidio module parameter.

After enumeration or discovery was performed for a given mport device, the corresponding subdirectory will also contain subdirectories for each child RapidIO device connected to the mport.

The example below shows mport device subdirectory with several child RapidIO devices attached to it:

```
[rio@rapidio ~]$ ls /sys/class/rapidio_port/rapidio0/ -l
total 0
drwxr-xr-x 3 root root    0 Feb 11 15:10 00:e:0001
drwxr-xr-x 3 root root    0 Feb 11 15:10 00:e:0004
drwxr-xr-x 3 root root    0 Feb 11 15:10 00:e:0007
drwxr-xr-x 3 root root    0 Feb 11 15:10 00:s:0002
drwxr-xr-x 3 root root    0 Feb 11 15:10 00:s:0003
drwxr-xr-x 3 root root    0 Feb 11 15:10 00:s:0005
lrwxrwxrwx 1 root root    0 Feb 11 15:11 device -> ../../../0000:01:00.0
-r--r--r-- 1 root root 4096 Feb 11 15:11 port_destid
drwxr-xr-x 2 root root    0 Feb 11 15:11 power
lrwxrwxrwx 1 root root    0 Feb 11 15:04 subsystem -> ../../../../../../class/
↪rapidio_port
-r--r--r-- 1 root root 4096 Feb 11 15:11 sys_size
-rw-r--r-- 1 root root 4096 Feb 11 15:04 uevent
```

/sys/class/rapidio_port/rapidio<N>/sys_size

Defined on file *sysfs-class-rapidio*

(RO) reports RapidIO common transport system size:

0 = small (8-bit destination ID, max. 256 devices),

1 = large (16-bit destination ID, max. 65536 devices).

/sys/class/rc/

Defined on file *sysfs-class-rc*

The rc/ class sub-directory belongs to the Remote Controller core and provides a sysfs interface for configuring infrared remote controller receivers.

/sys/class/rc/rc<N>/

Defined on file *sysfs-class-rc*

A /sys/class/rc/rcN directory is created for each remote control receiver device where N is the number of the receiver.

/sys/class/rc/rc<N>/filter

Defined on file *sysfs-class-rc*

Sets the scancode filter expected value.

Use in combination with /sys/class/rc/rcN/filter_mask to set the expected value of the bits set in the filter mask. If the hardware supports it then scancodes which do not match the filter will be ignored. Otherwise the write will fail with an error.

This value may be reset to 0 if the current protocol is altered.

/sys/class/rc/rc<N>/filter_mask

Defined on file *sysfs-class-rc*

Sets the scancode filter mask of bits to compare. Use in combination with /sys/class/rc/rcN/filter to set the bits of the scancode which should be compared against the expected value. A value of 0 disables the filter to allow all valid scancodes to be processed.

If the hardware supports it then scancodes which do not match the filter will be ignored. Otherwise the write will fail with an error.

This value may be reset to 0 if the current protocol is altered.

/sys/class/rc/rc<N>/protocols

Defined on file *sysfs-class-rc*

Reading this file returns a list of available protocols, something like:

```
"rc5 [rc6] nec jvc [sony]"
```

Enabled protocols are shown in [] brackets.

Writing ”+proto” will add a protocol to the list of enabled protocols.

Writing ”-proto” will remove a protocol from the list of enabled protocols.

Writing ”proto” will enable only ”proto”.

Writing ”none” will disable all protocols.

Write fails with EINVAL if an invalid protocol combination or unknown protocol name is used.

/sys/class/rc/rc<N>/wakeup_data

Defined on file *sysfs-class-rc-nuvoton*

Reading this file returns the stored CIR wakeup sequence. It starts with a pulse, followed by a space, pulse etc. All values are in microseconds. The same format can be used to store a wakeup sequence in the Nuvoton chip by writing to this file.

Note: Some systems reset the stored wakeup sequence to a factory default on each boot. On such systems store the wakeup sequence in a file and set it on boot using e.g. a udev rule.

/sys/class/rc/rc<N>/wakeup_filter

Defined on file *sysfs-class-rc*

Sets the scancode wakeup filter expected value.

Use in combination with /sys/class/rc/rcN/wakeup_filter_mask to set the expected value of the bits set in the wakeup filter mask to trigger a system wake event.

If the hardware supports it and wakeup_filter_mask is not 0 then scancodes which match the filter will wake the system from e.g. suspend to RAM or power off.

Otherwise the write will fail with an error.

This value may be reset to 0 if the wakeup protocol is altered.

/sys/class/rc/rc<N>/wakeup_filter_mask

Defined on file *sysfs-class-rc*

Sets the scancode wakeup filter mask of bits to compare.

Use in combination with /sys/class/rc/rcN/wakeup_filter to set the bits of the scancode which should be compared against the expected value to trigger a system wake event.

If the hardware supports it and wakeup_filter_mask is not 0 then scancodes which match the filter will wake the system from e.g. suspend to RAM or power off.

Otherwise the write will fail with an error.

This value may be reset to 0 if the wakeup protocol is altered.

/sys/class/rc/rc<N>/wakeup_protocols

Defined on file *sysfs-class-rc*

Reading this file returns a list of available protocols to use for the wakeup filter, something like:

```
"rc-5 nec nec-x rc-6-0 rc-6-6a-24 [rc-6-6a-32] rc-6-mce"
```

Note that protocol variants are listed, so ”nec”, ”sony”, ”rc-5”, ”rc-6” have their different bit length encodings listed if available.

The enabled wakeup protocol is shown in [] brackets.

Only one protocol can be selected at a time.

Writing ”proto” will use ”proto” for wakeup events.

Writing ”none” will disable wakeup.

Write fails with EINVAL if an invalid protocol combination or unknown protocol name is used, or if wakeup is not supported by the hardware.

/sys/class/regulator/.../bypass

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called bypass. This indicates if the device is in bypass mode.

This will be one of the following strings:

- 'enabled'
- 'disabled'
- 'unknown'

'enabled' means the regulator is in bypass mode.

'disabled' means that the regulator is regulating.

'unknown' means software cannot determine the state, or the reported state is invalid.

/sys/class/regulator/.../fail

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called fail. This indicates if the device reports an output failure (1) or not (0).

/sys/class/regulator/.../max_microamps

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called max_microamps. This holds the maximum safe working regulator output current limit setting for this domain measured in microamps, for regulators which support current constraints.

NOTE: this will return the string 'constraint not defined' if the power domain has no max microamps constraint defined by platform code.

/sys/class/regulator/.../max_microvolts

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called max_microvolts. This holds the maximum safe working regulator output voltage setting for this domain measured in microvolts, for regulators which support voltage constraints.

NOTE: this will return the string 'constraint not defined' if the power domain has no max microvolts constraint defined by platform code.

/sys/class/regulator/.../microamps

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called microamps. This holds the regulator output current limit setting measured in microamps (i.e. E-6 Amps), for regulators which can report the control input for a current limit.

NOTE: This value should not be used to determine the regulator output current level as this value is the same regardless of whether the regulator is enabled or disabled.

/sys/class/regulator/.../microvolts

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called microvolts. This holds the regulator output voltage setting measured in microvolts (i.e. E-6 Volts), for regulators which can report the control input for voltage.

NOTE: This value should not be used to determine the regulator output voltage level as this value is the same regardless of whether the regulator is enabled or disabled.

/sys/class/regulator/.../min_microamps

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called min_microamps. This holds the minimum safe working regulator output current limit setting for this domain measured in microamps, for regulators which support current constraints.

NOTE: this will return the string 'constraint not defined' if the power domain has no min microamps constraint defined by platform code.

/sys/class/regulator/.../min_microvolts

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called min_microvolts. This holds the minimum safe working regulator output voltage setting for this domain measured in microvolts, for regulators which support voltage constraints.

NOTE: this will return the string 'constraint not defined' if the power domain has no min microvolts constraint defined by platform code.

/sys/class/regulator/.../name

Defined on file *sysfs-class-regulator*

Each regulator directory will contain a field called name. This holds a string identifying the regulator for display purposes.

NOTE: this will be empty if no suitable name is provided by platform or regulator drivers.

/sys/class/regulator/.../num_users

Defined on file *sysfs-class-regulator*

Each regulator directory will contain a field called num_users. This holds the number of consumer devices that have called regulator_enable() on this regulator.

/sys/class/regulator/.../opmode

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called opmode. This holds the current regulator operating mode, for regulators which can report that control input value.

The opmode value can be one of the following strings:

- 'fast'
- 'normal'
- 'idle'
- 'standby'
- 'unknown'

The modes are described in include/linux/regulator/consumer.h

NOTE: This value should not be used to determine the regulator output operating mode as this value is the same regardless of whether the regulator is enabled or disabled. A "status" attribute may be available to determine the actual mode.

/sys/class/regulator/.../over_current

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called over_current. This indicates if the device reports an over-current fault (1) or not (0).

/sys/class/regulator/.../over_current_warn

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called over_current_warn. This indicates if the device reports an over-current warning (1) or not (0).

/sys/class/regulator/.../over_temp

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called over_temp. This indicates if the device reports an over-temperature fault (1) or not (0).

/sys/class/regulator/.../over_temp_warn

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called over_temp_warn. This indicates if the device reports an over-temperature warning (1) or not (0).

/sys/class/regulator/.../over_voltage_warn

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called over_voltage_warn. This indicates if the device reports an over-voltage warning (1) or not (0).

/sys/class/regulator/.../parent

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a link called parent. This points to the parent or supply regulator if one exists.

/sys/class/regulator/.../regulation_out

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called regulation_out. This indicates if the device reports an out-of-regulation fault (1) or not (0).

/sys/class/regulator/.../requested_microamps

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called requested_microamps. This holds the total requested load current in microamps for this regulator from all its consumer devices.

/sys/class/regulator/.../state

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called state. This reports the regulator enable control, for regulators which can report that input value.

This will be one of the following strings:

'enabled' 'disabled' 'unknown'

'enabled' means the regulator output is ON and is supplying power to the system (assuming no error prevents it).

'disabled' means the regulator output is OFF and is not supplying power to the system (unless some non-Linux control has enabled it).

'unknown' means software cannot determine the state, or the reported state is invalid.

NOTE: this field can be used in conjunction with microvolts or microamps to determine configured regulator output levels.

/sys/class/regulator/.../status

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called "status". This reports the current regulator status, for regulators which can report that output value.

This will be one of the following strings:

- off
- on

- error
- fast
- normal
- idle
- standby

”off” means the regulator is not supplying power to the system.

”on” means the regulator is supplying power to the system, and the regulator can't report a detailed operation mode.

”error” indicates an out-of-regulation status such as being disabled due to thermal shutdown, or voltage being unstable because of problems with the input power supply.

”fast”, ”normal”, ”idle”, and ”standby” are all detailed regulator operation modes (described elsewhere). They imply ”on”, but provide more detail.

Note that regulator status is a function of many inputs, not limited to control inputs from Linux. For example, the actual load presented may trigger ”error” status; or a regulator may be enabled by another user, even though Linux did not enable it.

/sys/class/regulator/.../suspend_disk_microvolts

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called suspend_disk_microvolts. This holds the regulator output voltage setting for this domain measured in microvolts when the system is suspended to disk, for voltage regulators implementing suspend voltage configuration constraints.

/sys/class/regulator/.../suspend_disk_mode

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called suspend_disk_mode. This holds the regulator operating mode setting for this domain when the system is suspended to disk, for regulators implementing suspend mode configuration constraints.

/sys/class/regulator/.../suspend_disk_state

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called suspend_disk_state. This holds the regulator operating state when suspended to disk, for regulators implementing suspend configuration constraints.

This will be one of the same strings reported by the ”state” attribute.

/sys/class/regulator/.../suspend_mem_microvolts

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called suspend_mem_microvolts. This holds the regulator output voltage setting for this domain measured in microvolts when the system is

suspended to memory, for voltage regulators implementing suspend voltage configuration constraints.

/sys/class/regulator/.../suspend_mem_mode

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called suspend_mem_mode. This holds the regulator operating mode setting for this domain when the system is suspended to memory, for regulators implementing suspend mode configuration constraints.

/sys/class/regulator/.../suspend_mem_state

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called suspend_mem_state. This holds the regulator operating state when suspended to memory, for regulators implementing suspend configuration constraints.

This will be one of the same strings reported by the "state" attribute.

/sys/class/regulator/.../suspend_standby_microvolts

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called suspend_standby_microvolts. This holds the regulator output voltage setting for this domain measured in microvolts when the system is suspended to standby, for voltage regulators implementing suspend voltage configuration constraints.

/sys/class/regulator/.../suspend_standby_mode

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called suspend_standby_mode. This holds the regulator operating mode setting for this domain when the system is suspended to standby, for regulators implementing suspend mode configuration constraints.

/sys/class/regulator/.../suspend_standby_state

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called suspend_standby_state. This holds the regulator operating state when suspended to standby, for regulators implementing suspend configuration constraints.

This will be one of the same strings reported by the "state" attribute.

/sys/class/regulator/.../type

Defined on file *sysfs-class-regulator*

Each regulator directory will contain a field called type. This holds the regulator type.

This will be one of the following strings:

- 'voltage'
- 'current'
- 'unknown'

'voltage' means the regulator output voltage can be controlled by software.

'current' means the regulator output current limit can be controlled by software.

'unknown' means software cannot control either voltage or current limit.

/sys/class/regulator/.../under_voltage

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called under_voltage. This indicates if the device reports an under-voltage fault (1) or not (0).

/sys/class/regulator/.../under_voltage_warn

Defined on file *sysfs-class-regulator*

Some regulator directories will contain a field called under_voltage_warn. This indicates if the device reports an under-voltage warning (1) or not (0).

/sys/class/remoteproc/.../coredump

Defined on file *sysfs-class-remoteproc*

Remote processor coredump configuration

Reports the coredump configuration of the remote processor, which will be one of:

”disabled” ”enabled” ”inline”

”disabled” means no dump will be collected.

”enabled” means when the remote processor's coredump is collected it will be copied to a separate buffer and that buffer is exposed to userspace.

”inline” means when the remote processor's coredump is collected userspace will directly read from the remote processor's device memory. Extra buffer will not be used to copy the dump. Also recovery process will not proceed until all data is read by userspace.

/sys/class/remoteproc/.../firmware

Defined on file *sysfs-class-remoteproc*

Remote processor firmware

Reports the name of the firmware currently loaded to the remote processor.

To change the running firmware, ensure the remote processor is stopped (using /sys/class/remoteproc/.../state) and write a new filename.

/sys/class/remoteproc/.../name

Defined on file *sysfs-class-remoteproc*

Remote processor name

Reports the name of the remote processor. This can be used by userspace in exactly identifying a remote processor and ease up the usage in modifying the 'firmware' or 'state' files.

/sys/class/remoteproc/.../recovery

Defined on file *sysfs-class-remoteproc*

Remote processor recovery mechanism

Reports the recovery mechanism of the remote processor, which will be one of:

”enabled” ”disabled”

”enabled” means, the remote processor will be automatically recovered whenever it crashes. Moreover, if the remote processor crashes while recovery is disabled, it will be automatically recovered too as soon as recovery is enabled.

”disabled” means, a remote processor will remain in a crashed state if it crashes. This is useful for debugging purposes; without it, debugging a crash is substantially harder.

/sys/class/remoteproc/.../state

Defined on file *sysfs-class-remoteproc*

Remote processor state

Reports the state of the remote processor, which will be one of:

- ”offline”
- ”suspended”
- ”running”
- ”crashed”
- ”invalid”

”offline” means the remote processor is powered off.

”suspended” means that the remote processor is suspended and must be woken to receive messages.

”running” is the normal state of an available remote processor

”crashed” indicates that a problem/crash has been detected on the remote processor.

”invalid” is returned if the remote processor is in an unknown state.

Writing this file controls the state of the remote processor. The following states can be written:

- ”start”
- ”stop”

Writing "start" will attempt to start the processor running the firmware indicated by, or written to, /sys/class/remoteproc/.../firmware. The remote processor should transition to "running" state.

Writing "stop" will attempt to halt the remote processor and return it to the "offline" state.

/sys/class/rnbd-client

Defined on file *sysfs-class-rnbd-client*

Provide information about RNBD-client. All sysfs files that are not read-only provide the usage information on read:

Example:

```
# cat /sys/class/rnbd-client/ctl/map_device

> Usage: echo "sessname=<name of the rtrs session> path=<[srcaddr,]dstaddr>
> [path=<[srcaddr,]dstaddr>] device_path=<full path on remote side>
> [access_mode=<ro|rw|migration>] > map_device
>
> addr ::= [ ip:<ipv4> | ip:<ipv6> | gid:<gid> ]
```

/sys/class/rnbd-client/ctl/devices/

Defined on file *sysfs-class-rnbd-client*

For each device mapped on the client a new symbolic link is created as /sys/class/rnbd-client/ctl/devices/<device_id>@<session_name>, which points to the block device created by rnbd (/sys/block/rnbd<N>/). The <device_id> of each device is created as follows:

- If the 'device_path' provided during mapping contains slashes ("/"), they are replaced by exclamation mark ("!") and used as as the <device_id>. Otherwise, the <device_id> will be the same as the "device_path" provided.

/sys/class/rnbd-client/ctl/map_device

Defined on file *sysfs-class-rnbd-client*

Expected format is the following:

```
sessname=<name of the rtrs session>
path=<[srcaddr,]dstaddr> [path=<[srcaddr,]dstaddr> ...]
device_path=<full path on remote side>
[access_mode=<ro|rw|migration>]
```

Where:

sessname:
accepts a string not bigger than 256 chars, which identifies a given session on the client and on the server. I.e. "clt_hostname-srv_hostname" could be a natural choice.

path:
describes a connection between the client and the server by specifying destination and,

when required, the source address. The addresses are to be provided in the following format:

```
ip:<IPv6>
ip:<IPv4>
gid:<GID>
```

for example:

```
path=ip:10.0.0.66
```

The single addr is treated as the destination. The connection will be established to this server from any client IP address.

```
path=ip:10.0.0.66,ip:10.0.1.66
```

First addr is the source address and the second is the destination.

If multiple ”path=” options are specified multiple connection will be established and data will be sent according to the selected multipath policy (see RTRS mp_policy sysfs entry description).

device_path:
Path to the block device on the server side. Path is specified relative to the directory on server side configured in the 'dev_search_path' module parameter of the rnbd_server. The rnbd_server prepends the <device_path> received from client with <dev_search_path> and tries to open the <dev_search_path>/<device_path> block device. On success, a /dev/rnbd<N> device file, a /sys/block/rnbd<N>/ directory and an entry in /sys/class/rnbd-client/ctl/devices will be created.

If 'dev_search_path' contains '%SESSNAME%', then each session can have different devices namespace, e.g. server was configured with the following parameter ”dev_search_path=/run/rnbd-devs/%SESSNAME%”, client has this string ”sessname=blya device_path=sda”, then server will try to open: /run/rnbd-devs/blya/sda.

access_mode:
the access_mode parameter specifies if the device is to be mapped as ”ro” read-only or ”rw” read-write. The server allows a device to be exported in rw mode only once. The ”migration” access mode has to be specified if a second mapping in read-write mode is desired.

By default ”rw” is used.

nr_poll_queues
specifies the number of poll-mode queues. If the IO has HIPRI flag, the block-layer will send the IO via the poll-mode queue. For fast network and device the polling is faster than interrupt-base IO handling because it saves time for context switching, switching to another process, handling the interrupt and switching back to the issuing process.

Set -1 if you want to set it as the number of CPUs By default rnbd client creates only irq-mode queues.

NOTICE: MUST make a unique session for a device using the poll-mode queues.

Exit Codes:

If the device is already mapped it will fail with EEXIST. If the input has an invalid format it will return EINVAL. If the device path cannot be found on the server, it will fail with ENOENT.

Finding device file after mapping

After mapping, the device file can be found by: o The symlink /sys/class/rnbd-client/ctl/devices/<device_id>@<session_name> points to /sys/block/<dev-name>. The last part of the symlink destination is the same as the device name. By extracting the last part of the path the path to the device /dev/<dev-name> can be build.

- /dev/block/$(cat /sys/class/rnbd-client/ctl/devices/<device_id>@<session_name>/dev)

How to find the <device_id> of the device is described on the next section.

/sys/class/rnbd-server

Defined on file *sysfs-class-rnbd-server*

provide information about RNBD-server.

/sys/class/rnbd-server/ctl/

Defined on file *sysfs-class-rnbd-server*

When a client maps a device, a directory entry with the name of the block device is created under /sys/class/rnbd-server/ctl/devices/.

/sys/class/rnbd-server/ctl/devices/<device_name>/block_dev

Defined on file *sysfs-class-rnbd-server*

Is a symlink to the sysfs entry of the exported device.

Example: block_dev -> ../../../../class/block/ram0

/sys/class/rnbd-server/ctl/devices/<device_name>/sessions/

Defined on file *sysfs-class-rnbd-server*

For each client a particular device is exported to, following directory will be created:

/sys/class/rnbd-server/ctl/devices/<device_name>/sessions/<session-name>/

When the device is unmapped by that client, the directory will be removed.

/sys/class/rnbd-server/ctl/devices/<device_name>/sessions/<session-name>/access_mode

Defined on file *sysfs-class-rnbd-server*

Contains the device access mode: ro, rw or migration.

/sys/class/rnbd-server/ctl/devices/<device_name>/sessions/<session-name>/force_close

Defined on file *sysfs-class-rnbd-server*

Write "1" to the file to close the device on server side. Please note that the client side device will not be closed, read or write to the device will get -ENOTCONN.

/sys/class/rnbd-server/ctl/devices/<device_name>/sessions/<session-name>/mapping_path

Defined on file *sysfs-class-rnbd-server*

Contains the relative device path provided by the user during mapping.

/sys/class/rnbd-server/ctl/devices/<device_name>/sessions/<session-name>/read_only

Defined on file *sysfs-class-rnbd-server*

Contains '1' if device is mapped read-only, otherwise '0'.

/sys/class/rtc/

Defined on file *sysfs-class-rtc*

The rtc/ class subdirectory belongs to the RTC subsystem.

/sys/class/rtc/rtc0/device/rtc_calibration

Defined on file *sysfs-class-rtc-rtc0-device-rtc_calibration*

Attribute for calibrating ST-Ericsson AB8500 Real Time Clock The rtc_calibration attribute allows the userspace to calibrate the AB8500.s 32KHz Real Time Clock. Every 60 seconds the AB8500 will correct the RTC's value by adding to it the value of this attribute.

The range of the attribute is -127 to +127 in units of 30.5 micro-seconds (half-parts-per-million of the 32KHz clock)

Users: The /vendor/st-ericsson/base_utilities/core/rtc_calibration daemon uses this interface.

/sys/class/rtc/rtcX/

Defined on file *sysfs-class-rtc*

The /sys/class/rtc/rtc{0,1,2,3,...} directories correspond to each RTC device.

/sys/class/rtc/rtcX/date

Defined on file *sysfs-class-rtc*

(RO) RTC-provided date in YYYY-MM-DD format

/sys/class/rtc/rtcX/hctosys

Defined on file *sysfs-class-rtc*

(RO) 1 if the RTC provided the system time at boot via the CONFIG_RTC_HCTOSYS kernel option, 0 otherwise

/sys/class/rtc/rtcX/max_user_freq

Defined on file *sysfs-class-rtc*

(RW) The maximum interrupt rate an unprivileged user may request from this RTC.

/sys/class/rtc/rtcX/name

Defined on file *sysfs-class-rtc*

(RO) The name of the RTC corresponding to this sysfs directory

/sys/class/rtc/rtcX/offset

Defined on file *sysfs-class-rtc*

(RW) The amount which the rtc clock has been adjusted in firmware. Visible only if the driver supports clock offset adjustment. The unit is parts per billion, i.e. The number of clock ticks which are added to or removed from the rtc's base clock per billion ticks. A positive value makes a day pass more slowly, longer, and a negative value makes a day pass more quickly.

/sys/class/rtc/rtcX/range

Defined on file *sysfs-class-rtc*

Valid time range for the RTC, as seconds from epoch, formatted as [min, max]

/sys/class/rtc/rtcX/since_epoch

Defined on file *sysfs-class-rtc*

(RO) RTC-provided time as the number of seconds since the epoch

/sys/class/rtc/rtcX/time

Defined on file *sysfs-class-rtc*

(RO) RTC-provided time in 24-hour notation (hh:mm:ss)

/sys/class/rtc/rtcX/wakealarm

Defined on file *sysfs-class-rtc*

(RW) The time at which the clock will generate a system wakeup event. This is a one shot wakeup event, so must be reset after wake if a daily wakeup is required. Format is seconds since the epoch by default, or if there's a leading +, seconds in the future, or if there is a leading +=, seconds ahead of the current alarm.

/sys/class/rtrs-client

Defined on file *sysfs-class-rtrs-client*

When a user of RTRS API creates a new session, a directory entry with the name of that session is created under /sys/class/rtrs-client/<session-name>/

/sys/class/rtrs-client/<session-name>/add_path

Defined on file *sysfs-class-rtrs-client*

RW, adds a new path (connection) to an existing session. Expected format is the following:

```
<[source addr,]destination addr>
*addr ::= [ ip:<ipv4|ipv6> | gid:<gid> ]
```

/sys/class/rtrs-client/<session-name>/max_reconnect_attempts

Defined on file *sysfs-class-rtrs-client*

Maximum number reconnect attempts the client should make before giving up after connection breaks unexpectedly.

/sys/class/rtrs-client/<session-name>/mp_policy

Defined on file *sysfs-class-rtrs-client*

Multipath policy specifies which path should be selected on each IO:

round-robin (0):
select path in per CPU round-robin manner.

min-inflight (1):
select path with minimum inflights.

min-latency (2):
select path with minimum latency.

/sys/class/rtrs-client/<session-name>/paths/

Defined on file *sysfs-class-rtrs-client*

Each path belonging to a given session is listed here by its source and destination address. When a new path is added to a session by writing to the "add_path" entry, a directory <src@dst> is created.

/sys/class/rtrs-client/<session-name>/paths/<src@dst>/cur_latency

Defined on file *sysfs-class-rtrs-client*

RO, Contains the latency time calculated by the heart-beat messages. Whenever the client sends heart-beat message, it checks the time gap between sending the heart-beat message and receiving the ACK. This value can be changed regularly.

/sys/class/rtrs-client/<session-name>/paths/<src@dst>/disconnect

Defined on file *sysfs-class-rtrs-client*

Write "1" to the file in order to disconnect the path. Operation blocks until RTRS path is disconnected.

/sys/class/rtrs-client/<session-name>/paths/<src@dst>/dst_addr

Defined on file *sysfs-class-rtrs-client*

RO, Contains the destination address of the path

/sys/class/rtrs-client/<session-name>/paths/<src@dst>/hca_name

Defined on file *sysfs-class-rtrs-client*

RO, Contains the name of HCA the connection established on.

/sys/class/rtrs-client/<session-name>/paths/<src@dst>/hca_port

Defined on file *sysfs-class-rtrs-client*

RO, Contains the port number of active port traffic is going through.

/sys/class/rtrs-client/<session-name>/paths/<src@dst>/reconnect

Defined on file *sysfs-class-rtrs-client*

Write "1" to the file in order to reconnect the path. Operation is blocking and returns 0 if reconnect was successful.

/sys/class/rtrs-client/<session-name>/paths/<src@dst>/remove_path

Defined on file *sysfs-class-rtrs-client*

Write "1" to the file in order to disconnected and remove the path from the session. Operation blocks until the path is disconnected and removed from the session.

/sys/class/rtrs-client/<session-name>/paths/<src@dst>/src_addr

Defined on file *sysfs-class-rtrs-client*

RO, Contains the source address of the path

/sys/class/rtrs-client/<session-name>/paths/<src@dst>/state

Defined on file *sysfs-class-rtrs-client*

RO, Contains "connected" if the session is connected to the peer and fully functional. Otherwise the file contains "disconnected"

/sys/class/rtrs-client/<session-name>/paths/<src@dst>/stats/cpu_migration

Defined on file *sysfs-class-rtrs-client*

RTRS expects that each HCA IRQ is pinned to a separate CPU. If it's not the case, the processing of an I/O response could be processed on a different CPU than where it was originally submitted. This file shows how many interrupts where generated on a non expected CPU.

"from:"
is the CPU on which the IRQ was expected, but not generated.

"to:"
is the CPU on which the IRQ was generated, but not expected.

/sys/class/rtrs-client/<session-name>/paths/<src@dst>/stats/rdma

Defined on file *sysfs-class-rtrs-client*

Contains statistics regarding rdma operations and inflight operations. The output consists of 6 values:

```
<read-count> <read-total-size> <write-count> \
<write-total-size> <inflights> <failovered>
```

/sys/class/rtrs-client/<session-name>/paths/<src@dst>/stats/reconnects

Defined on file *sysfs-class-rtrs-client*

Contains 2 unsigned int values, the first one records number of successful reconnects in the path lifetime, the second one records number of failed reconnects in the path lifetime.

/sys/class/rtrs-client/<session-name>/paths/<src@dst>/stats/reset_all

Defined on file *sysfs-class-rtrs-client*

RW, Read will return usage help, write 0 will clear all the statistics.

/sys/class/rtrs-server

Defined on file *sysfs-class-rtrs-server*

When a user of RTRS API creates a new session on a client side, a directory entry with the name of that session is created in here.

/sys/class/rtrs-server/<session-name>/paths/

Defined on file *sysfs-class-rtrs-server*

When new path is created by writing to "add_path" entry on client side, a directory entry named as <source address>@<destination address> is created on server.

/sys/class/rtrs-server/<session-name>/paths/<src@dst>/disconnect

Defined on file *sysfs-class-rtrs-server*

When "1" is written to the file, the RTRS session is being disconnected. Operations is non-blocking and returns control immediately to the caller.

/sys/class/rtrs-server/<session-name>/paths/<src@dst>/dst_addr

Defined on file *sysfs-class-rtrs-server*

RO, Contains the destination address of the path

/sys/class/rtrs-server/<session-name>/paths/<src@dst>/hca_name

Defined on file *sysfs-class-rtrs-server*

RO, Contains the name of HCA the connection established on.

/sys/class/rtrs-server/<session-name>/paths/<src@dst>/hca_port

Defined on file *sysfs-class-rtrs-server*

RO, Contains the port number of active port traffic is going through.

/sys/class/rtrs-server/<session-name>/paths/<src@dst>/src_addr

Defined on file *sysfs-class-rtrs-server*

RO, Contains the source address of the path

/sys/class/rtrs-server/<session-name>/paths/<src@dst>/stats/rdma

Defined on file *sysfs-class-rtrs-server*

Contains statistics regarding rdma operations and inflight operations. The output consists of 5 values: <read-count> <read-total-size> <write-count> <write-total-size> <inflights>

/sys/class/scsi_device/*/device/dyn_cap_needed

Defined on file *sysfs-driver-ufs*

This file shows the amount of physical memory needed to be removed from the physical memory resources pool of the particular logical unit. The full information about the attribute could be found at UFS specifications 2.1.

The file is read only.

/sys/class/scsi_device/*/device/unit_descriptor/boot_lun_id

Defined on file *sysfs-driver-ufs*

This file shows boot LUN information. This is one of the UFS unit descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/class/scsi_device/*/device/unit_descriptor/context_capabilities

Defined on file *sysfs-driver-ufs*

This file shows the context capabilities. This is one of the UFS unit descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/class/scsi_device/*/device/unit_descriptor/data_reliability

Defined on file *sysfs-driver-ufs*

This file defines the device behavior when a power failure occurs during a write operation. This is one of the UFS unit descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/class/scsi_device/*/device/unit_descriptor/erase_block_size

Defined on file *sysfs-driver-ufs*

This file shows the erase block size. This is one of the UFS unit descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/class/scsi_device/*/device/unit_descriptor/large_unit_granularity

Defined on file *sysfs-driver-ufs*

This file shows the granularity of the LUN. This is one of the UFS unit descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/class/scsi_device/*/device/unit_descriptor/logical_block_count

Defined on file *sysfs-driver-ufs*

This file shows total number of addressable logical blocks. This is one of the UFS unit descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/class/scsi_device/*/device/unit_descriptor/logical_block_size

Defined on file *sysfs-driver-ufs*

This file shows the size of addressable logical blocks (calculated as an exponent with base 2). This is one of the UFS unit descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/class/scsi_device/*/device/unit_descriptor/lun_memory_type

Defined on file *sysfs-driver-ufs*

This file shows LUN memory type. This is one of the UFS unit descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/class/scsi_device/*/device/unit_descriptor/lun_queue_depth

Defined on file *sysfs-driver-ufs*

This file shows LUN queue depth. This is one of the UFS unit descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/class/scsi_device/*/device/unit_descriptor/lun_write_protect

Defined on file *sysfs-driver-ufs*

This file shows LUN write protection status. This is one of the UFS unit descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/class/scsi_device/*/device/unit_descriptor/physical_memory_resourse_count

Defined on file *sysfs-driver-ufs*

This file shows the total physical memory resources. This is one of the UFS unit descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/class/scsi_device/*/device/unit_descriptor/provisioning_type

Defined on file *sysfs-driver-ufs*

This file shows the thin provisioning type. This is one of the UFS unit descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/class/scsi_device/*/device/unit_descriptor/psa_sensitive

Defined on file *sysfs-driver-ufs*

This file shows PSA sensitivity. This is one of the UFS unit descriptor parameters. The full information about the descriptor could be found at UFS specifications 2.1.

The file is read only.

/sys/class/scsi_device/*/device/unit_descriptor/wb_buf_alloc_units

Defined on file *sysfs-driver-ufs*

This entry shows the configured size of WriteBooster buffer. 0400h corresponds to 4GB.

The file is read only.

/sys/class/scsi_host/hostX/acciopath_status

Defined on file *sysfs-class-scsi_host*

This file contains the current status of the "SSD Smart Path" feature of HP Smart Array RAID controllers using the hpsa driver. SSD Smart Path, when enabled permits the driver to send i/o requests directly to physical devices that are part of a logical drive, bypassing the controllers firmware RAID stack for a performance advantage when possible. A value of '1' indicates the feature is enabled, and the controller may use the direct i/o path to physical devices. A value of zero means the feature is disabled and the controller may not use the direct i/o path to physical devices. This setting is controller wide, affecting all configured logical drives on the controller. This file is readable and writable.

/sys/class/scsi_host/hostX/ahci_host_version

Defined on file *sysfs-class-scsi_host*

(RO) Display the version of the AHCI spec implemented by the host.

/sys/class/scsi_host/hostX/ahci_port_cmd
/sys/class/scsi_host/hostX/ahci_host_caps
/sys/class/scsi_host/hostX/ahci_host_cap2

Defined on file *sysfs-class-scsi_host*

[to be documented]

/sys/class/scsi_host/hostX/em_buffer

Defined on file *sysfs-class-scsi_host*

(RW) Allows access to AHCI EM (enclosure management) buffer directly if the host supports EM.

For eg. the AHCI driver supports SGPIO EM messages but the SATA/AHCI specs do not define the SGPIO message format of the EM buffer. Different hardware(HW) vendors may have different definitions. With the em_buffer attribute, this issue can be solved by allowing HW vendors to provide userland drivers and tools for their SGPIO initiators.

/sys/class/scsi_host/hostX/em_message_supported

Defined on file *sysfs-class-scsi_host*

(RO) Displays supported enclosure management message types.

/sys/class/scsi_host/hostX/em_message
/sys/class/scsi_host/hostX/em_message_type

Defined on file *sysfs-class-scsi_host*

em_message: (RW) Enclosure management support. For the LED protocol, writes and reads correspond to the LED message format as defined in the AHCI spec.

The user must turn sw_activity (under */sys/block/*/device/*) OFF it they wish to control the activity LED via the em_message file.

em_message_type: (RO) Displays the current enclosure management protocol that is being used by the driver (for eg. LED, SAF-TE, SES-2, SGPIO etc).

/sys/class/scsi_host/hostX/isci_id

Defined on file *sysfs-class-scsi_host*

This file contains the enumerated host ID for the Intel SCU controller. The Intel(R) C600 Series Chipset SATA/SAS Storage Control Unit embeds up to two 4-port controllers in a single PCI device. The controllers are enumerated in order which usually means the lowest number scsi_host corresponds with the first controller, but this association is not guaranteed. The 'isci_id' attribute unambiguously identifies the controller index: '0' for the first controller, '1' for the second.

/sys/class/scsi_host/hostX/link_power_management_policy

Defined on file *sysfs-class-scsi_host*

(RW) This parameter allows the user to read and set the link (interface) power management.

There are four possible options:

min_power: Tell the controller to try to make the link use the least possible power when possible. This may sacrifice some performance due to increased latency when coming out of lower power states.

max_performance: Generally, this means no power management. Tell the controller to have performance be a priority over power management.

medium_power: Tell the controller to enter a lower power state when possible, but do not enter the lowest power state, thus improving latency over min_power setting.

med_power_with_dipm: Identical to the existing medium_power setting except that it enables dipm (device initiated power management) on top, which makes it match the Windows IRST (Intel Rapid Storage Technology) driver settings. This setting is also close to min_power, except that:

a) It does not use host-initiated slumber mode, but it does allow device-initiated slumber

b) It does not enable low power device sleep mode (DevSlp).

/sys/class/scsi_tape/*/stats/in_flight

Defined on file *sysfs-class-scsi_tape*

Show the number of I/Os currently in-flight between the st module and the SCSI mid-layer.

/sys/class/scsi_tape/*/stats/io_ns

Defined on file *sysfs-class-scsi_tape*

Shows the total amount of time spent waiting for all I/O to and from the tape drive to complete. This includes all reads, writes, and other SCSI commands issued to the tape drive. An example of other SCSI commands would be tape movement such as a rewind when a rewind tape device is closed. This item is measured in nanoseconds.

To determine the amount of time spent waiting for other I/O to complete subtract read_ns and write_ns from this value.

/sys/class/scsi_tape/*/stats/other_cnt

Defined on file *sysfs-class-scsi_tape*

The number of I/O requests issued to the tape drive other than SCSI read/write requests.

/sys/class/scsi_tape/*/stats/read_byte_cnt

Defined on file *sysfs-class-scsi_tape*

Shows the total number of bytes requested from the tape drive. This value is presented in bytes because tape drives support variable length block sizes.

/sys/class/scsi_tape/*/stats/read_cnt

Defined on file *sysfs-class-scsi_tape*

Shows the total number of read requests issued to the tape drive.

/sys/class/scsi_tape/*/stats/read_ns

Defined on file *sysfs-class-scsi_tape*

Shows the total amount of time in nanoseconds waiting for read I/O requests to complete.

/sys/class/scsi_tape/*/stats/resid_cnt

Defined on file *sysfs-class-scsi_tape*

Shows the number of times we found that a residual >0 was found when the SCSI midlayer indicated that there was an error. For reads this may be a case of someone issuing reads greater than the block size.

/sys/class/scsi_tape/*/stats/write_byte_cnt

Defined on file *sysfs-class-scsi_tape*

Shows the total number of bytes written to the tape drive. This value is presented in bytes because tape drives support variable length block sizes.

/sys/class/scsi_tape/*/stats/write_cnt

Defined on file *sysfs-class-scsi_tape*

Shows the total number of write requests issued to the tape drive.

/sys/class/scsi_tape/*/stats/write_ms

Defined on file *sysfs-class-scsi_tape*

Shows the total amount of time in nanoseconds waiting for write I/O requests to complete.

/sys/class/spi_master/spi<bus>/spi<bus>.<dev>/fram

Defined on file *sysfs-class-spi-eeprom*

Contains the FRAM binary data. Same as EEPROM, just another file name to indicate that it employs ferroelectric process. It performs write operations at bus speed - no write delays.

/sys/class/spi_master/spi<bus>/spi<bus>.<dev>/sernum

Defined on file *sysfs-class-spi-eeprom*

Contains the serial number of the Cypress FRAM (FM25VN) if it is present. It will be displayed as a 8 byte hex string, as read from the device.

This is a read-only attribute.

/sys/class/stm/<stm>/channels

Defined on file *sysfs-class-stm*

Shows the number of channels per master on this STM device.

/sys/class/stm/<stm>/hw_override

Defined on file *sysfs-class-stm*

Reads as 0 if master numbers in the STP stream produced by this stm device will match the master numbers assigned by the software or 1 if the stm hardware overrides software assigned masters.

/sys/class/stm/<stm>/masters

Defined on file *sysfs-class-stm*

Shows first and last available to software master numbers on this STM device.

/sys/class/stm_source/<stm_source>/stm_source_link

Defined on file *sysfs-class-stm_source*

stm_source device linkage to stm device, where its tracing data is directed. Reads return an existing connection or "<none>" if this stm_source is not connected to any stm device yet. Write an existing (registered) stm device's name here to connect that device. If a device is already connected to this stm_source, it will first be disconnected.

/sys/class/switchtec

Defined on file *sysfs-class-switchtec*

The switchtec class subsystem folder. Each registered switchtec driver is represented by a switchtecX subfolder (X being an integer >= 0).

/sys/class/switchtec/switchtec[0-9]+/component_id

Defined on file *sysfs-class-switchtec*

Component identifier as stored in the hardware (eg. PM8543) (read only) Values: arbitrary string.

/sys/class/switchtec/switchtec[0-9]+/component_revision

Defined on file *sysfs-class-switchtec*

Component revision stored in the hardware (read only) Values: integer.

/sys/class/switchtec/switchtec[0-9]+/component_vendor

Defined on file *sysfs-class-switchtec*

Component vendor as stored in the hardware (eg. MICROSEM) (read only) Values: arbitrary string.

/sys/class/switchtec/switchtec[0-9]+/device_version

Defined on file *sysfs-class-switchtec*

Device version as stored in the hardware (read only) Values: integer.

/sys/class/switchtec/switchtec[0-9]+/fw_version

Defined on file *sysfs-class-switchtec*

Currently running firmware version (read only) Values: integer (in hexadecimal).

/sys/class/switchtec/switchtec[0-9]+/partition

Defined on file *sysfs-class-switchtec*

Partition number for this device in the switch (read only) Values: integer.

/sys/class/switchtec/switchtec[0-9]+/partition_count

Defined on file *sysfs-class-switchtec*

Total number of partitions in the switch (read only) Values: integer.

/sys/class/switchtec/switchtec[0-9]+/product_id

Defined on file *sysfs-class-switchtec*

Product identifier as stored in the hardware (eg. PSX 48XG3) (read only) Values: arbitrary string.

/sys/class/switchtec/switchtec[0-9]+/product_revision

Defined on file *sysfs-class-switchtec*

Product revision stored in the hardware (eg. RevB) (read only) Values: arbitrary string.

/sys/class/switchtec/switchtec[0-9]+/product_vendor

Defined on file *sysfs-class-switchtec*

Product vendor as stored in the hardware (eg. MICROSEM) (read only) Values: arbitrary string.

/sys/class/thermal/cooling_deviceX/cur_state

Defined on file *sysfs-class-thermal*

The current cooling state of this cooling device. The value can any integer numbers between 0 and max_state:

- cur_state == 0 means no cooling
- cur_state == max_state means the maximum cooling.

RW, Required

/sys/class/thermal/cooling_deviceX/max_state

Defined on file *sysfs-class-thermal*

The maximum permissible cooling state of this cooling device.

RO, Required

/sys/class/thermal/cooling_deviceX/stats/reset

Defined on file *sysfs-class-thermal*

Writing any value resets the cooling device's statistics.

WO, Required

/sys/class/thermal/cooling_deviceX/stats/time_in_state_ms:

Defined on file *sysfs-class-thermal*

The amount of time spent by the cooling device in various cooling states. The output will have "<state> <time>" pair in each line, which will mean this cooling device spent <time> msec of time at <state>.

Output will have one line for each of the supported states.

RO, Required

/sys/class/thermal/cooling_deviceX/stats/total_trans

Defined on file *sysfs-class-thermal*

A single positive value showing the total number of times the state of a cooling device is changed.

RO, Required

/sys/class/thermal/cooling_deviceX/stats/trans_table

Defined on file *sysfs-class-thermal*

This gives fine grained information about all the cooling state transitions. The cat output here is a two dimensional matrix, where an entry <i,j> (row i, column j) represents the number of transitions from State_i to State_j. If the transition table is bigger than PAGE_SIZE, reading this will return an -EFBIG error.

RO, Required

/sys/class/thermal/cooling_deviceX/type

Defined on file *sysfs-class-thermal*

String which represents the type of device, e.g:

- for generic ACPI: should be "Fan", "Processor" or "LCD"
- for memory controller device on intel_menlow platform: should be "Memory controller".

RO, Required

/sys/class/thermal/thermal_zoneX/available_policies

Defined on file *sysfs-class-thermal*

Available thermal governors which can be used for a particular zone.

RO, Required

/sys/class/thermal/thermal_zoneX/cdevY

Defined on file *sysfs-class-thermal*

Sysfs link to the thermal cooling device node where the sys I/F for cooling device throttling control represents.

RO, Optional

/sys/class/thermal/thermal_zoneX/cdevY_trip_point

Defined on file *sysfs-class-thermal*

The trip point in this thermal zone which *cdev[0-*]* is associated with; -1 means the cooling device is not associated with any trip point.

RO, Optional

/sys/class/thermal/thermal_zoneX/cdevY_weight

Defined on file *sysfs-class-thermal*

The influence of *cdev[0-*]* in this thermal zone. This value is relative to the rest of cooling devices in the thermal zone. For example, if a cooling device has a weight double than that of other, it's twice as effective in cooling the thermal zone.

RW, Optional

/sys/class/thermal/thermal_zoneX/emul_temp

Defined on file *sysfs-class-thermal*

Interface to set the emulated temperature method in thermal zone (sensor). After setting this temperature, the thermal zone may pass this temperature to platform emulation function if registered or cache it locally. This is useful in debugging different temperature threshold and its associated cooling action. This is write only node and writing 0 on this node should disable emulation.

Unit: millidegree Celsius

WO, Optional

WARNING:
Be careful while enabling this option on production systems, because userland can easily disable the thermal policy by simply flooding this sysfs node with low temperature values.

/sys/class/thermal/thermal_zoneX/integral_cutoff

Defined on file *sysfs-class-thermal*

Temperature offset from the desired temperature trip point above which the integral term of the power allocator governor's PID controller starts accumulating errors. For example, if integral_cutoff is 0, then the integral term only accumulates error when temperature is above the desired temperature trip point. For more information see /driver-api/thermal/power_allocator

Unit: millidegree Celsius

RW, Optional

/sys/class/thermal/thermal_zoneX/k_d

Defined on file *sysfs-class-thermal*

The derivative term of the power allocator governor's PID controller. For more information see /driver-api/thermal/power_allocator

RW, Optional

/sys/class/thermal/thermal_zoneX/k_i

Defined on file *sysfs-class-thermal*

The integral term of the power allocator governor's PID controller. This term allows the PID controller to compensate for long term drift. For more information see /driver-api/thermal/power_allocator

RW, Optional

/sys/class/thermal/thermal_zoneX/k_po

Defined on file *sysfs-class-thermal*

The proportional term of the power allocator governor's PID controller during temperature overshoot. Temperature overshoot is when the current temperature is above the "desired temperature" trip point. For more information see /driver-api/thermal/power_allocator

RW, Optional

/sys/class/thermal/thermal_zoneX/k_pu

Defined on file *sysfs-class-thermal*

The proportional term of the power allocator governor's PID controller during temperature undershoot. Temperature undershoot is when the current temperature is below the "desired temperature" trip point. For more information see /driver-api/thermal/power_allocator

RW, Optional

/sys/class/thermal/thermal_zoneX/mode

Defined on file *sysfs-class-thermal*

One of the predefined values in [enabled, disabled]. This file gives information about the algorithm that is currently managing the thermal zone. It can be either default kernel based algorithm or user space application.

enabled
: enable Kernel Thermal management.

disabled
: Preventing kernel thermal zone driver actions upon trip points so that user application can take full charge of the thermal management.

RW, Optional

/sys/class/thermal/thermal_zoneX/offset

Defined on file *sysfs-class-thermal*

The offset constant used in a linear extrapolation model to determine a hotspot temperature based off the sensor's raw readings. It is up to the device driver to determine the usage of these values.

RW, Optional

/sys/class/thermal/thermal_zoneX/policy

Defined on file *sysfs-class-thermal*

One of the various thermal governors used for a particular zone.

RW, Required

/sys/class/thermal/thermal_zoneX/slope

Defined on file *sysfs-class-thermal*

The slope constant used in a linear extrapolation model to determine a hotspot temperature based off the sensor's raw readings. It is up to the device driver to determine the usage of these values.

RW, Optional

/sys/class/thermal/thermal_zoneX/sustainable_power

Defined on file *sysfs-class-thermal*

An estimate of the sustained power that can be dissipated by the thermal zone. Used by the power allocator governor. For more information see /driver-api/thermal/power_allocator

Unit: milliwatts

RW, Optional

/sys/class/thermal/thermal_zoneX/temp

Defined on file *sysfs-class-thermal*

Current temperature as reported by thermal zone (sensor).

Unit: millidegree Celsius

RO, Required

/sys/class/thermal/thermal_zoneX/trip_point_Y_hyst

Defined on file *sysfs-class-thermal*

The hysteresis value for a trip point, represented as an integer.

Unit: Celsius

RW, Optional

/sys/class/thermal/thermal_zoneX/trip_point_Y_temp

Defined on file *sysfs-class-thermal*

The temperature above which trip point will be fired.

Unit: millidegree Celsius

RO, Optional

/sys/class/thermal/thermal_zoneX/trip_point_Y_type

Defined on file *sysfs-class-thermal*

Strings which indicate the type of the trip point.

E.g. it can be one of critical, hot, passive, *active[0-*]* for ACPI thermal zone.

RO, Optional

/sys/class/thermal/thermal_zoneX/type

Defined on file *sysfs-class-thermal*

Strings which represent the thermal zone type. This is given by thermal zone driver as part of registration. E.g: "acpitz" indicates it's an ACPI thermal device. In order to keep it consistent with hwmon sys attribute; this should be a short, lowercase string, not containing spaces nor dashes.

RO, Required

/sys/class/timecard/

Defined on file *sysfs-timecard*

This directory contains files and directories providing a standardized interface to the ancillary features of the OpenCompute timecard.

/sys/class/timecard/ocpN/

Defined on file *sysfs-timecard*

This directory contains the attributes of the Nth timecard registered.

/sys/class/timecard/ocpN/available_clock_sources

Defined on file *sysfs-timecard*

(RO) The list of available time sources that the PHC uses for clock adjustments.

NONE	no adjustments
PPS	adjustments come from the PPS1 selector (default)
TOD	adjustments from the GNSS/TOD module
IRIG	adjustments from external IRIG-B signal
DCF	adjustments from external DCF signal

/sys/class/timecard/ocpN/available_sma_inputs

Defined on file *sysfs-timecard*

(RO) Set of available destinations (sinks) for a SMA input signal.

10Mhz	signal is used as the 10Mhz reference clock
PPS1	signal is sent to the PPS1 selector
PPS2	signal is sent to the PPS2 selector
TS1	signal is sent to timestamper 1
TS2	signal is sent to timestamper 2
TS3	signal is sent to timestamper 3
TS4	signal is sent to timestamper 4
IRIG	signal is sent to the IRIG-B module
DCF	signal is sent to the DCF module
FREQ1	signal is sent to frequency counter 1
FREQ2	signal is sent to frequency counter 2
FREQ3	signal is sent to frequency counter 3
FREQ4	signal is sent to frequency counter 4
None	signal input is disabled

/sys/class/timecard/ocpN/available_sma_outputs

Defined on file *sysfs-timecard*

(RO) Set of available sources for a SMA output signal.

10Mhz	output is from the 10Mhz reference clock
PHC	output PPS is from the PHC clock
MAC	output PPS is from the Miniature Atomic Clock
GNSS1	output PPS is from the first GNSS module
GNSS2	output PPS is from the second GNSS module
IRIG	output is from the PHC, in IRIG-B format
DCF	output is from the PHC, in DCF format
GEN1	output is from frequency generator 1
GEN2	output is from frequency generator 2
GEN3	output is from frequency generator 3
GEN4	output is from frequency generator 4
GND	output is GND
VCC	output is VCC

/sys/class/timecard/ocpN/clock_source

Defined on file *sysfs-timecard*

(RW) Contains the current synchronization source used by the PHC. May be changed by writing one of the listed values from the available_clock_sources attribute set.

/sys/class/timecard/ocpN/clock_status_drift

Defined on file *sysfs-timecard*

(RO) Contains the current drift value used by the firmware for internal disciplining of the atomic clock.

/sys/class/timecard/ocpN/clock_status_offset

Defined on file *sysfs-timecard*

(RO) Contains the current offset value used by the firmware for internal disciplining of the atomic clock.

/sys/class/timecard/ocpN/freqX

Defined on file *sysfs-timecard*

(RO) Optional directory containing the sysfs nodes for frequency counter <X>.

/sys/class/timecard/ocpN/freqX/frequency

Defined on file *sysfs-timecard*

(RO) Contains the measured frequency over the specified measurement period.

/sys/class/timecard/ocpN/freqX/seconds

Defined on file *sysfs-timecard*

(RW) Specifies the number of seconds from 0-255 that the frequency should be measured over. Write 0 to disable.

/sys/class/timecard/ocpN/genX

Defined on file *sysfs-timecard*

(RO) Optional directory containing the sysfs nodes for frequency generator <X>.

/sys/class/timecard/ocpN/genX/duty

Defined on file *sysfs-timecard*

(RO) Specifies the signal duty cycle as a percentage from 1-99.

/sys/class/timecard/ocpN/genX/period

Defined on file *sysfs-timecard*

(RO) Specifies the signal period in nanoseconds.

/sys/class/timecard/ocpN/genX/phase

Defined on file *sysfs-timecard*

(RO) Specifies the signal phase offset in nanoseconds.

/sys/class/timecard/ocpN/genX/polarity

Defined on file *sysfs-timecard*

(RO) Specifies the signal polarity, either 1 or 0.

/sys/class/timecard/ocpN/genX/running

Defined on file *sysfs-timecard*

(RO) Either 0 or 1, showing if the signal generator is running.

/sys/class/timecard/ocpN/genX/signal

Defined on file *sysfs-timecard*

(RW) Used to start the signal generator, and summarize the current status.

The signal generator may be started by writing the signal period, followed by the optional signal values. If the optional values are not provided, they default to the current settings, which may be obtained from the other sysfs nodes.

> period [duty [phase [polarity]]]

echo 500000000 > signal # 1/2 second period echo 1000000 40 100 > signal echo 0 > signal # turn off generator

Period and phase are specified in nanoseconds. Duty cycle is a percentage from 1-99. Polarity is 1 or 0.

Reading this node will return:

> period duty phase polarity start_time

/sys/class/timecard/ocpN/genX/start

Defined on file *sysfs-timecard*

(RO) Shows the time in <sec>.<nsec> that the signal generator started running.

/sys/class/timecard/ocpN/gnss_sync

Defined on file *sysfs-timecard*

(RO) Indicates whether a valid GNSS signal is received, or when the signal was lost.

/sys/class/timecard/ocpN/i2c

Defined on file *sysfs-timecard*

This optional attribute links to the associated i2c device.

/sys/class/timecard/ocpN/irig_b_mode

Defined on file *sysfs-timecard*

(RW) An integer from 0-7 indicating the timecode format of the IRIG-B output signal: B00<n>

/sys/class/timecard/ocpN/pps

Defined on file *sysfs-timecard*

This optional attribute links to the associated PPS device.

/sys/class/timecard/ocpN/ptp

Defined on file *sysfs-timecard*

This attribute links to the associated PTP device.

/sys/class/timecard/ocpN/serialnum

Defined on file *sysfs-timecard*

(RO) Provides the serial number of the timecard.

/sys/class/timecard/ocpN/sma1
/sys/class/timecard/ocpN/sma2
/sys/class/timecard/ocpN/sma3
/sys/class/timecard/ocpN/sma4

Defined on file *sysfs-timecard*

(RW) These attributes specify the direction of the signal on the associated SMA connectors, and also the signal sink or source.

The display format of the attribute is a space separated list of signals, prefixed by the input/output direction.

The signal direction may be changed (if supported) by prefixing the signal list with either "in:" or "out:". If neither prefix is present, then the direction is unchanged.

The output signal may be changed by writing one of the listed values from the available_sma_outputs attribute set.

The input destinations may be changed by writing multiple values from the available_sma_inputs attribute set, separated by spaces. If there are duplicated input destinations between connectors, the lowest numbered SMA connector is given priority.

Note that not all input combinations may make sense.

The 10Mhz reference clock input is currently only valid on SMA1 and may not be combined with other destination sinks.

/sys/class/timecard/ocpN/tod_correction

Defined on file *sysfs-timecard*

(RW) The incoming GNSS signal is in UTC time, and the NMEA format messages do not provide a TAI offset. This sets the correction value for the incoming time.

If UBX_LS is enabled, this should be 0, and the offset is taken from the UBX-NAV-TIMELS message.

/sys/class/timecard/ocpN/ts_window_adjust

Defined on file *sysfs-timecard*

(RW) When retrieving the PHC with the PTP SYS_OFFSET_EXTENDED ioctl, a system timestamp is made before and after the PHC time is retrieved. The midpoint between the two system timestamps is usually taken to be the SYS time associated with the PHC time. This estimate may be wrong, as it depends on PCI latencies, and when the PHC time was latched

The attribute value reduces the end timestamp by the given number of nanoseconds, so the computed midpoint matches the retrieved PHC time.

The initial value is set based on measured PCI latency and the estimated point where the FPGA latches the PHC time. This value may be changed by writing an unsigned integer.

/sys/class/timecard/ocpN/ttyGNSS
/sys/class/timecard/ocpN/ttyGNSS2

Defined on file *sysfs-timecard*

These optional attributes link to the TTY serial ports associated with the GNSS devices.

/sys/class/timecard/ocpN/ttyMAC

Defined on file *sysfs-timecard*

This optional attribute links to the TTY serial port associated with the Miniature Atomic Clock.

/sys/class/timecard/ocpN/ttyNMEA

Defined on file *sysfs-timecard*

This optional attribute links to the TTY serial port which outputs the PHC time in NMEA ZDA format.

/sys/class/timecard/ocpN/utc_tai_offset

Defined on file *sysfs-timecard*

(RW) The DCF and IRIG output signals are in UTC, while the TimeCard operates on TAI. This attribute allows setting the offset in seconds, which is added to the TAI timebase for these formats.

The offset may be changed by writing an unsigned integer.

/sys/class/tpm/tpmX/ppi/

Defined on file *sysfs-driver-ppi*

This folder includes the attributes related with PPI (Physical Presence Interface). Only if TPM is supported by BIOS, this folder makes sense. The folder path can be got by command 'find /sys/ -name 'pcrs''. For the detail information of PPI, please refer to the PPI specification from

http://www.trustedcomputinggroup.org/

In Linux 4.2 ppi was moved to the character device directory. A symlink from tpmX/device/ppi to tpmX/ppi to provide backwards compatibility.

/sys/class/tpm/tpmX/ppi/request

Defined on file *sysfs-driver-ppi*

This attribute shows the request for an operation to be executed in the pre-OS environment. It is the only input from the OS to the pre-OS environment. The request should be an integer value range from 1 to 160, and 0 means no request. This file can be read and written.

/sys/class/tpm/tpmX/ppi/response

Defined on file *sysfs-driver-ppi*

This attribute shows the response to the most recent operation request it acted upon. The format is ”<request> <response num> : <response description>”. This file is readonly.

/sys/class/tpm/tpmX/ppi/tcg_operations

Defined on file *sysfs-driver-ppi*

This attribute shows whether it is allowed to request an operation to be executed in the pre-OS environment by the BIOS for the requests defined by TCG, i.e. requests from 1 to 22. The format is ”<request> <status num>: <status description>”. This attribute is only supported by PPI version 1.2+. This file is readonly.

/sys/class/tpm/tpmX/ppi/transition_action

Defined on file *sysfs-driver-ppi*

This attribute shows the platform-specific action that should take place in order to transition to the BIOS for execution of a requested operation. The format is ”<action num>: <action description>”. This file is readonly.

/sys/class/tpm/tpmX/ppi/version

Defined on file *sysfs-driver-ppi*

This attribute shows the version of the PPI supported by the platform. This file is readonly.

/sys/class/tpm/tpmX/ppi/vs_operations

Defined on file *sysfs-driver-ppi*

This attribute shows whether it is allowed to request an operation to be executed in the pre-OS environment by the BIOS for the verdor specific requests, i.e. requests from 128 to 255. The format is same with tcg_operations. This attribute is also only supported by PPI version 1.2+. This file is readonly.

/sys/class/tty/console/active

Defined on file *sysfs-tty*

Shows the list of currently configured console devices, like 'tty1 ttyS0'. The last entry in the file is the active device connected to /dev/console. The file supports poll() to detect virtual console switches.

/sys/class/tty/tty<x>/active

Defined on file *sysfs-tty*

Shows the currently active virtual console device, like 'tty1'. The file supports poll() to detect virtual console switches.

/sys/class/tty/ttyS<x>/close_delay

Defined on file *sysfs-tty*

Show the closing delay time for this port in centiseconds.

These sysfs values expose the TIOCGSERIAL interface via sysfs rather than via ioctls.

/sys/class/tty/ttyS<x>/closing_wait

Defined on file *sysfs-tty*

Show the close wait time for this port in centiseconds.

Waiting forever is represented as 0. If waiting on close is disabled then the value is 65535.

These sysfs values expose the TIOCGSERIAL interface via sysfs rather than via ioctls.

/sys/class/tty/ttyS<x>/console

Defined on file *sysfs-tty*

Allows user to detach or attach back the given device as kernel console. It shows and accepts a boolean variable.

/sys/class/tty/ttyS<x>/custom_divisor

Defined on file *sysfs-tty*

Show the custom divisor if any that is set on this port.

These sysfs values expose the TIOCGSERIAL interface via sysfs rather than via ioctls.

/sys/class/tty/ttyS<x>/flags

Defined on file *sysfs-tty*

Show the tty port status flags for this port.

These sysfs values expose the TIOCGSERIAL interface via sysfs rather than via ioctls.

/sys/class/tty/ttyS<x>/io_type

Defined on file *sysfs-tty*

Show the I/O type that is to be used with the iomem base address.

These sysfs values expose the TIOCGSERIAL interface via sysfs rather than via ioctls.

/sys/class/tty/ttyS<x>/iomem_base

Defined on file *sysfs-tty*

The I/O memory base for this port.

These sysfs values expose the TIOCGSERIAL interface via sysfs rather than via ioctls.

/sys/class/tty/ttyS<x>/iomem_reg_shift

Defined on file *sysfs-tty*

Show the register shift indicating the spacing to be used for accesses on this iomem address.

These sysfs values expose the TIOCGSERIAL interface via sysfs rather than via ioctls.

/sys/class/tty/ttyS<x>/irq

Defined on file *sysfs-tty*

Shows the current primary interrupt for this port.

These sysfs values expose the TIOCGSERIAL interface via sysfs rather than via ioctls.

/sys/class/tty/ttyS<x>/line

Defined on file *sysfs-tty*

Shows the current tty line number for this port.

These sysfs values expose the TIOCGSERIAL interface via sysfs rather than via ioctls.

/sys/class/tty/ttyS<x>/port

Defined on file *sysfs-tty*

Shows the current tty port I/O address for this port.

These sysfs values expose the TIOCGSERIAL interface via sysfs rather than via ioctls.

/sys/class/tty/ttyS<x>/rx_trig_bytes

Defined on file *sysfs-tty*

Shows current RX interrupt trigger bytes or sets the user specified value to change it for the FIFO buffer. Users can show or set this value regardless of opening the serial device file or not.

The RX trigger can be set one of four kinds of values for UART serials. When users input a meaning less value to this I/F, the RX trigger is changed to the nearest lower value for the device specification. For example, when user sets 7bytes on 16550A, which has 1/4/8/14 bytes trigger, the RX trigger is automatically changed to 4 bytes.

/sys/class/tty/ttyS<x>/type

Defined on file *sysfs-tty*

Shows the current tty type for this port.

These sysfs values expose the TIOCGSERIAL interface via sysfs rather than via ioctls.

/sys/class/tty/ttyS<x>/uartclk

Defined on file *sysfs-tty*

Shows the current uartclk value associated with the UART port in serial_core, that is bound to TTY like ttyS0. uartclk = 16 * baud_base

These sysfs values expose the TIOCGSERIAL interface via sysfs rather than via ioctls.

/sys/class/tty/ttyS<x>/xmit_fifo_size

Defined on file *sysfs-tty*

Show the transmit FIFO size for this port.

These sysfs values expose the TIOCGSERIAL interface via sysfs rather than via ioctls.

/sys/class/typec/<port>-<plug>/number_of_alternate_modes

Defined on file *sysfs-class-typec*

Shows the number of alternate modes which are advertised by the plug associated with a particular cable during Power Delivery discovery. This file remains hidden until a value greater than or equal to 0 is set by Type C port driver.

USB Type-C partner/cable Power Delivery Identity objects

NOTE: The following attributes will be applicable to both partner (e.g /sys/class/typec/port0-partner/) and cable (e.g /sys/class/typec/port0-cable/) devices. Consequently, the example file paths below are prefixed with "/sys/class/typec/<port>-{partner|cable}/" to reflect this.

/sys/class/typec/<port>-cable/plug_type

Defined on file *sysfs-class-typec*

Shows type of the plug on the cable:

- type-a - Standard A
- type-b - Standard B
- type-c
- captive

/sys/class/typec/<port>-cable/type

Defined on file *sysfs-class-typec*

USB Power Delivery Specification defines a set of product types for the cables. This file will show the product type of the cable if it is known. If the product type of the cable is not visible to the device driver, this file will not exist.

When the cable product type is detected, uvevent is also raised with PRODUCT_TYPE showing the product type of the cable.

Valid values:

undefined	•
active	Active Cable
passive	Passive Cable

/sys/class/typec/<port>-partner/accessory_mode

Defined on file *sysfs-class-typec*

Shows the Accessory Mode name when the partner is an Accessory. The Accessory Modes are defined in USB Type-C Specification.

/sys/class/typec/<port>-partner/identity/

Defined on file *sysfs-class-typec*

This directory appears only if the port device driver is capable of showing the result of Discover Identity USB power delivery command. That will not always be possible even when USB power delivery is supported, for example when USB power delivery communication for the port is mostly handled in firmware. If the directory exists, it will have an attribute file for every VDO in Discover Identity command result.

USB Type-C cable devices (eg. /sys/class/typec/port0-cable/)

Note: Electronically Marked Cables will have a device also for one cable plug (eg. /sys/class/typec/port0-plug0). If the cable is active and has also SOP Double Prime controller (USB Power Deliver specification ch. 2.4) it will have second device also for the other plug.

Both plugs may have alternate modes as described in USB Type-C and USB Power Delivery specifications.

/sys/class/typec/<port>-partner/number_of_alternate_modes

Defined on file *sysfs-class-typec*

Shows the number of alternate modes which are advertised by the partner during Power Delivery discovery. This file remains hidden until a value greater than or equal to 0 is set by Type C port driver.

/sys/class/typec/<port>-partner/supports_usb_power_delivery

Defined on file *sysfs-class-typec*

Shows if the partner supports USB Power Delivery communication: Valid values: yes, no

/sys/class/typec/<port>-partner/type

Defined on file *sysfs-class-typec*

USB Power Delivery Specification defines a set of product types for the partner devices. This file will show the product type of the partner if it is known. Dual-role capable partners will have both UFP and DFP product types defined, but only one that matches the current role will be active at the time. If the product type of the partner is not visible to the device driver, this file will not exist.

When the partner product type is detected, or changed with role swap, uvevent is also raised that contains PRODUCT_TYPE=<product type> (for example PRODUCT_TYPE=hub).

Valid values:

UFP / device role ========== ==========
undefined - hub PDUSB Hub peripheral PDUSB Peripheral psd Power Bank ama Alternate Mode Adapter ========== ==========

DFP / host role ========== ==========
undefined - hub PDUSB Hub host PDUSB Host power_brick Power Brick amc Alternate Mode Controller ========== ==========

/sys/class/typec/<port>-{partner|cable}/identity/

Defined on file *sysfs-class-typec*

This directory appears only if the port device driver is capable of showing the result of Discover Identity USB power delivery command. That will not always be possible even when USB power delivery is supported, for example when USB power delivery communication for the port is mostly handled in firmware. If the directory exists, it will have an attribute file for every VDO in Discover Identity command result.

/sys/class/typec/<port>-{partner|cable}/identity/cert_stat

Defined on file *sysfs-class-typec*

Cert Stat VDO part of Discover Identity command result. The value will show 0 until Discover Identity command result becomes available. The value can be polled.

/sys/class/typec/<port>-{partner|cable}/identity/id_header

Defined on file *sysfs-class-typec*

ID Header VDO part of Discover Identity command result. The value will show 0 until Discover Identity command result becomes available. The value can be polled.

/sys/class/typec/<port>-{partner|cable}/identity/product

Defined on file *sysfs-class-typec*

Product VDO part of Discover Identity command result. The value will show 0 until Discover Identity command result becomes available. The value can be polled.

/sys/class/typec/<port>-{partner|cable}/identity/product_type_vdo1

Defined on file *sysfs-class-typec*

1st Product Type VDO of Discover Identity command result. The value will show 0 until Discover Identity command result becomes available and a valid Product Type VDO is returned.

/sys/class/typec/<port>-{partner|cable}/identity/product_type_vdo2

Defined on file *sysfs-class-typec*

2nd Product Type VDO of Discover Identity command result. The value will show 0 until Discover Identity command result becomes available and a valid Product Type VDO is returned.

/sys/class/typec/<port>-{partner|cable}/identity/product_type_vdo3

Defined on file *sysfs-class-typec*

3rd Product Type VDO of Discover Identity command result. The value will show 0 until Discover Identity command result becomes available and a valid Product Type VDO is returned.

USB Type-C port alternate mode devices.

/sys/class/typec/<port>-{partner|cable}/usb_power_delivery_revision

Defined on file *sysfs-class-typec*

Revision number of the supported USB Power Delivery specification of the port partner or cable, or 0.0 when USB Power Delivery is not supported.

Example values: - ”2.0”: USB Power Delivery Release 2.0 - ”3.0”: USB Power Delivery Release 3.0 - ”3.1”: USB Power Delivery Release 3.1

/sys/class/typec/<port>/<alt mode>/supported_roles

Defined on file *sysfs-class-typec*

Space separated list of the supported roles.

Valid values: source, sink

/sys/class/typec/<port>/data_role

Defined on file *sysfs-class-typec*

The supported USB data roles. This attribute can be used for requesting data role swapping on the port. Swapping is supported as synchronous operation, so write(2) to the attribute will not return until the operation has finished. The attribute is notified about role changes so that poll(2) on the attribute wakes up. Change on the role will also generate uevent KOBJ_CHANGE on the port. The current role is show in brackets, for example "[host] device" when DRP port is in host mode.

Valid values: host, device

/sys/class/typec/<port>/orientation

Defined on file *sysfs-class-typec*

Indicates the active orientation of the Type-C connector. Valid values: - "normal": CC1 orientation - "reverse": CC2 orientation - "unknown": Orientation cannot be determined.

/sys/class/typec/<port>/port_type

Defined on file *sysfs-class-typec*

Indicates the type of the port. This attribute can be used for requesting a change in the port type. Port type change is supported as a synchronous operation, so write(2) to the attribute will not return until the operation has finished.

Valid values:

source	(The port will behave as source only DFP port)
sink	(The port will behave as sink only UFP port)
dual	(The port will behave as dual-role-data and dual-role-power port)

/sys/class/typec/<port>/power_operation_mode

Defined on file *sysfs-class-typec*

Shows the current power operational mode the port is in. The power operation mode means current level for VBUS. In case USB Power Delivery communication is used for negotiating the levels, power operation mode should show "usb_power_delivery".

Valid values:

- default
- 1.5A
- 3.0A

- usb_power_delivery

/sys/class/typec/<port>/power_role

Defined on file *sysfs-class-typec*

The supported power roles. This attribute can be used to request power role swap on the port. Swapping is supported as synchronous operation, so write(2) to the attribute will not return until the operation has finished. The attribute is notified about role changes so that poll(2) on the attribute wakes up. Change on the role will also generate uevent KOBJ_CHANGE. The current role is show in brackets, for example "[source] sink" when in source mode.

Valid values: source, sink

/sys/class/typec/<port>/preferred_role

Defined on file *sysfs-class-typec*

The user space can notify the driver about the preferred role. It should be handled as enabling of Try.SRC or Try.SNK, as defined in USB Type-C specification, in the port drivers. By default the preferred role should come from the platform.

Valid values: source, sink, none (to remove preference)

/sys/class/typec/<port>/select_usb_power_delivery

Defined on file *sysfs-class-typec*

Lists the USB Power Delivery Capabilities that the port can advertise to the partner. The currently used capabilities are in brackets. Selection happens by writing to the file.

USB Type-C partner devices (eg. /sys/class/typec/port0-partner/)

/sys/class/typec/<port>/supported_accessory_modes

Defined on file *sysfs-class-typec*

Space separated list of accessory modes, defined in the USB Type-C specification, the port supports.

/sys/class/typec/<port>/usb_power_delivery_revision

Defined on file *sysfs-class-typec*

Revision number of the supported USB Power Delivery specification, or 0.0 when USB Power Delivery is not supported.

Example values: - "2.0": USB Power Delivery Release 2.0 - "3.0": USB Power Delivery Release 3.0 - "3.1": USB Power Delivery Release 3.1

/sys/class/typec/<port>/usb_typec_revision

Defined on file *sysfs-class-typec*

Revision number of the supported USB Type-C specification.

/sys/class/typec/<port>/vconn_source

Defined on file *sysfs-class-typec*

Shows is the port VCONN Source. This attribute can be used to request VCONN swap to change the VCONN Source during connection when both the port and the partner support USB Power Delivery. Swapping is supported as synchronous operation, so write(2) to the attribute will not return until the operation has finished. The attribute is notified about VCONN source changes so that poll(2) on the attribute wakes up. Change on VCONN source also generates uevent KOBJ_CHANGE.

Valid values:

- ”no” when the port is not the VCONN Source
- ”yes” when the port is the VCONN Source

/sys/class/uacce/<dev_name>/algorithms

Defined on file *sysfs-driver-uacce*

Algorithms supported by this accelerator, separated by new line. Can be any string and up to userspace to parse.

/sys/class/uacce/<dev_name>/api

Defined on file *sysfs-driver-uacce*

Api of the device Can be any string and up to userspace to parse. Application use the api to match the correct driver

/sys/class/uacce/<dev_name>/available_instances

Defined on file *sysfs-driver-uacce*

Available instances left of the device Return -ENODEV if uacce_ops get_available_instances is not provided

/sys/class/uacce/<dev_name>/flags

Defined on file *sysfs-driver-uacce*

Attributes of the device, see UACCE_DEV_xxx flag defined in uacce.h

/sys/class/uacce/<dev_name>/isolate

Defined on file *sysfs-driver-uacce*

(R) A sysfs node that read the device isolated state. The value 1 means the device is unavailable. The 0 means the device is available.

/sys/class/uacce/<dev_name>/isolate_strategy

Defined on file *sysfs-driver-uacce*

(RW) A sysfs node that configure the error threshold for the hardware isolation strategy. This size is a configured integer value, which is the number of threshold for hardware errors occurred in one hour. The default is 0. 0 means never isolate the device. The maximum value is 65535. You can write a number of threshold based on your hardware.

/sys/class/uacce/<dev_name>/region_dus_size

Defined on file *sysfs-driver-uacce*

Size (bytes) of dus region queue file

/sys/class/uacce/<dev_name>/region_mmio_size

Defined on file *sysfs-driver-uacce*

Size (bytes) of mmio region queue file

/sys/class/udc/udc.name/device/gadget/video4linux/video.name/function_name

Defined on file *configfs-usb-gadget-uvc*

UVC configfs function instance name

/sys/class/usb_power_delivery

Defined on file *sysfs-class-usb_power_delivery*

Directory for USB Power Delivery devices.

/sys/class/usb_power_delivery/.../<capability>/1:fixed_supply/dual_role_data

Defined on file *sysfs-class-usb_power_delivery*

This file shows the value of the Dual-Role Data bit in vSafe5V Fixed Supply Object. Dual role data means ability act as both USB host and USB device.

/sys/class/usb_power_delivery/.../<capability>/1:fixed_supply/dual_role_power

Defined on file *sysfs-class-usb_power_delivery*

This file contains boolean value that tells does the device support both source and sink power roles.

/sys/class/usb_power_delivery/.../<capability>/1:fixed_supply/unchunked_extended_message

Defined on file *sysfs-class-usb_power_delivery*

This file shows the value of the Unchunked Extended Messages Supported bit in vSafe5V Fixed Supply Object.

/sys/class/usb_power_delivery/.../<capability>/1:fixed_supply/unconstrained_power

Defined on file *sysfs-class-usb_power_delivery*

This file shows the value of the Unconstrained Power bit in vSafe5V Fixed Supply Object. The bit is set when an external source of power, powerful enough to power the entire system on its own, is available for the device.

/sys/class/usb_power_delivery/.../<capability>/1:fixed_supply/usb_communication_capable

Defined on file *sysfs-class-usb_power_delivery*

This file shows the value of the USB Communication Capable bit in vSafe5V Fixed Supply Object.

/sys/class/usb_power_delivery/.../<capability>/<position>:battery

Defined on file *sysfs-class-usb_power_delivery*

Battery PDO.

/sys/class/usb_power_delivery/.../<capability>/<position>:battery/maximum_voltage

Defined on file *sysfs-class-usb_power_delivery*

Maximum Voltage in millivolts.

/sys/class/usb_power_delivery/.../<capability>/<position>:battery/minimum_voltage

Defined on file *sysfs-class-usb_power_delivery*

Minimum Voltage in millivolts.

/sys/class/usb_power_delivery/.../<capability>/<position>:fixed_supply

Defined on file *sysfs-class-usb_power_delivery*

Devices containing the attributes (the bit fields) defined for Fixed Supplies.

The device "1:fixed_supply" is special. USB Power Delivery Specification dictates that the first PDO (at object position 1), and the only mandatory PDO, is always the vSafe5V Fixed Supply Object. vSafe5V Object has additional fields defined for it that the other Fixed Supply Objects do not have and that are related to the USB capabilities rather than power capabilities.

/sys/class/usb_power_delivery/.../<capability>/<position>:fixed_supply/voltage

Defined on file *sysfs-class-usb_power_delivery*

The voltage the supply supports in millivolts.

/sys/class/usb_power_delivery/.../<capability>/<position>:programmable_supply

Defined on file *sysfs-class-usb_power_delivery*

Programmable Power Supply (PPS) Augmented PDO (APDO).

/sys/class/usb_power_delivery/.../<capability>/<position>:programmable_supply/maximum_

Defined on file *sysfs-class-usb_power_delivery*

Maximum Current in milliamperes.

/sys/class/usb_power_delivery/.../<capability>/<position>:programmable_supply/maximum_

Defined on file *sysfs-class-usb_power_delivery*

Maximum Voltage in millivolts.

/sys/class/usb_power_delivery/.../<capability>/<position>:programmable_supply/minimum_

Defined on file *sysfs-class-usb_power_delivery*

Minimum Voltage in millivolts.

/sys/class/usb_power_delivery/.../<capability>/<position>:variable_supply

Defined on file *sysfs-class-usb_power_delivery*

Variable Power Supply PDO.

/sys/class/usb_power_delivery/.../<capability>/<position>:variable_supply/maximum_voltag

Defined on file *sysfs-class-usb_power_delivery*

Maximum Voltage in millivolts.

/sys/class/usb_power_delivery/.../<capability>/<position>:variable_supply/minimum_voltag

Defined on file *sysfs-class-usb_power_delivery*

Minimum Voltage in millivolts.

/sys/class/usb_power_delivery/.../revision

Defined on file *sysfs-class-usb_power_delivery*

File showing the USB Power Delivery Specification Revision used in communication.

/sys/class/usb_power_delivery/.../sink-capabilities

Defined on file *sysfs-class-usb_power_delivery*

The sink capability message "Sink_Capabilities" contains a set of Power Data Objects (PDO) just like with source capabilities, but instead of describing the power capabilities, these objects describe the power requirements.

The order of the objects in the sink capability message is the same as with the source capabilities message.

Fixed Supplies

/sys/class/usb_power_delivery/.../sink-capabilities/1:fixed_supply/higher_capability

Defined on file *sysfs-class-usb_power_delivery*

This file shows the value of the Higher capability bit in vsafe5V Fixed Supply Object. If the bit is set, then the sink needs more than vsafe5V(eg. 12 V) to provide full functionality. Valid values: 0, 1

/sys/class/usb_power_delivery/.../sink-capabilities/<position>:battery/operational_power

Defined on file *sysfs-class-usb_power_delivery*

The operational power that the sink requires at the given voltage range.

Standard Power Range (SPR) Programmable Power Supplies

/sys/class/usb_power_delivery/.../sink-capabilities/<position>:fixed_supply/fast_role_swap_c

Defined on file *sysfs-class-usb_power_delivery*

This file contains the value of the "Fast Role Swap USB Type-C Current" field that tells the current level the sink requires after a Fast Role Swap. 0 - Fast Swap not supported" 1 - Default USB Power" 2 - 1.5A@5V" 3 - 3.0A@5V"

Variable Supplies

/sys/class/usb_power_delivery/.../sink-capabilities/<position>:fixed_supply/operational_curr

Defined on file *sysfs-class-usb_power_delivery*

Operational current of the sink in milliamperes.

/sys/class/usb_power_delivery/.../sink-capabilities/<position>:variable_supply/operational_c

Defined on file *sysfs-class-usb_power_delivery*

The operational current in milliamperes that the sink requires at the given Voltage range.

Battery Supplies

/sys/class/usb_power_delivery/.../source-capabilities

Defined on file *sysfs-class-usb_power_delivery*

The source capabilities message "Source_Capabilities" contains a set of Power Data Objects (PDO), each representing a type of power supply. The order of the PDO objects is defined in the USB Power Delivery Specification. Each PDO - power supply - will have its own device, and the PDO device name will start with the object position number as the first character followed by the power supply type name (":" as delimiter).

/sys/class/usb_power_delivery/.../source_capabilities/<position>:<type>

/sys/class/usb_power_delivery/.../source-capabilities/1:fixed_supply/usb_suspend_supported

Defined on file *sysfs-class-usb_power_delivery*

This file shows the value of the USB Suspend Supported bit in vSafe5V Fixed Supply Object. If the bit is set then the device will follow the USB 2.0 and USB 3.2 rules for suspend and resume.

/sys/class/usb_power_delivery/.../source-capabilities/<position>:battery/maximum_power

Defined on file *sysfs-class-usb_power_delivery*

Maximum allowable Power in milliwatts.

/sys/class/usb_power_delivery/.../source-capabilities/<position>:fixed_supply/maximum_cur

Defined on file *sysfs-class-usb_power_delivery*

Maximum current of the fixed source supply in milliamperes.

/sys/class/usb_power_delivery/.../source-capabilities/<position>:fixed_supply/peak_current

Defined on file *sysfs-class-usb_power_delivery*

This file shows the value of the Fixed Power Source Peak Current Capability field.

/sys/class/usb_power_delivery/.../source-capabilities/<position>:programmable_supply/pps_

Defined on file *sysfs-class-usb_power_delivery*

The PPS Power Limited bit indicates whether or not the source supply will exceed the rated output power if requested.

/sys/class/usb_power_delivery/.../source-capabilities/<position>:variable_supply/maximum_

Defined on file *sysfs-class-usb_power_delivery*

The maximum current in milliamperes that the source can supply at the given Voltage range.

/sys/class/usb_power_delivery/.../version

Defined on file *sysfs-class-usb_power_delivery*

This is an optional attribute file showing the version of the specific revision of the USB Power Delivery Specification. In most cases the specification version is not known and the file is not available.

/sys/class/usb_role/

Defined on file *sysfs-class-usb_role*

Place in sysfs for USB Role Switches. USB Role Switch is a device that can select the data role (host or device) for USB port.

/sys/class/usb_role/<switch>/role

Defined on file *sysfs-class-usb_role*

The current role of the switch. This attribute can be used for requesting role swapping with non-USB Type-C ports. With USB Type-C ports, the ABI defined for USB Type-C connector class must be used.

Valid values: - none - host - device

/sys/class/vduse/

Defined on file *sysfs-class-vduse*

The vduse/ class sub-directory belongs to the VDUSE framework and provides a sysfs interface for configuring VDUSE devices.

/sys/class/vduse/<device-name>/

Defined on file *sysfs-class-vduse*

This directory entry is created when a VDUSE device is created via the control device.

/sys/class/vduse/<device-name>/msg_timeout

Defined on file *sysfs-class-vduse*

(RW) The timeout (in seconds) for waiting for the control message's response from userspace. Default value is 30s. Writing a '0' to the file means to disable the timeout.

/sys/class/vduse/control/

Defined on file *sysfs-class-vduse*

This directory entry is created for the control device of VDUSE framework.

/sys/class/wakeup/

Defined on file *sysfs-class-wakeup*

The */sys/class/wakeup/* directory contains pointers to all wakeup sources in the kernel at that moment in time.

/sys/class/wakeup/.../active_count

Defined on file *sysfs-class-wakeup*

This file contains the number of times the wakeup source was activated.

/sys/class/wakeup/.../active_time_ms

Defined on file *sysfs-class-wakeup*

This file contains the amount of time the wakeup source has been continuously active, in milliseconds. If the wakeup source is not active, this file contains '0'.

/sys/class/wakeup/.../event_count

Defined on file *sysfs-class-wakeup*

This file contains the number of signaled wakeup events associated with the wakeup source.

/sys/class/wakeup/.../expire_count

Defined on file *sysfs-class-wakeup*

This file contains the number of times the wakeup source's timeout has expired.

/sys/class/wakeup/.../last_change_ms

Defined on file *sysfs-class-wakeup*

This file contains the monotonic clock time when the wakeup source was touched last time, in milliseconds.

/sys/class/wakeup/.../max_time_ms

Defined on file *sysfs-class-wakeup*

This file contains the maximum amount of time this wakeup source has been continuously active, in milliseconds.

/sys/class/wakeup/.../name

Defined on file *sysfs-class-wakeup*

This file contains the name of the wakeup source.

/sys/class/wakeup/.../prevent_suspend_time_ms

Defined on file *sysfs-class-wakeup*

The file contains the total amount of time this wakeup source has been preventing autosleep, in milliseconds.

/sys/class/wakeup/.../total_time_ms

Defined on file *sysfs-class-wakeup*

This file contains the total amount of time this wakeup source has been active, in milliseconds.

/sys/class/wakeup/.../wakeup_count

Defined on file *sysfs-class-wakeup*

This file contains the number of times the wakeup source might abort suspend.

/sys/class/watchdog/watchdog1/access_cs0

Defined on file *sysfs-class-watchdog*

It is a read/write file. This attribute exists only if the system has booted from the alternate flash chip due to expiration of a watchdog timer of AST2400/AST2500 when alternate boot function was enabled with 'aspeed,alt-boot' devicetree option for that watchdog or with an appropriate h/w strapping (for WDT2 only).

At alternate flash the 'access_cs0' sysfs node provides:

ast2400:
a way to get access to the primary SPI flash chip at CS0 after booting from the alternate chip at CS1.

ast2500:
a way to restore the normal address mapping from (CS0->CS1, CS1->CS0) to (CS0->CS0, CS1->CS1).

Clearing the boot code selection and timeout counter also resets to the initial state the chip select line mapping. When the SoC is in normal mapping state (i.e. booted from CS0), clearing those bits does nothing for both versions of the SoC. For alternate boot mode (booted from CS1 due to wdt2 expiration) the behavior differs as described above.

This option can be used with wdt2 (watchdog1) only.

When read, the current status of the boot code selection is shown. When written with any non-zero value, it clears the boot code selection and the timeout counter, which results in chipselect reset for AST2400/AST2500.

/sys/class/watchdog/watchdogn/bootstatus

Defined on file *sysfs-class-watchdog*

It is a read only file. It contains status of the watchdog device at boot. It is equivalent to WDIOC_GETBOOTSTATUS of ioctl interface.

/sys/class/watchdog/watchdogn/fw_version

Defined on file *sysfs-class-watchdog*

It is a read only file. It contains firmware version of watchdog device.

/sys/class/watchdog/watchdogn/identity

Defined on file *sysfs-class-watchdog*

It is a read only file. It contains identity string of watchdog device.

/sys/class/watchdog/watchdogn/nowayout

Defined on file *sysfs-class-watchdog*

It is a read/write file. While reading, it gives '1' if the device has the nowayout feature set, otherwise it gives '0'. Writing a '1' to the file enables the nowayout feature. Once set, the nowayout feature cannot be disabled, so writing a '0' either has no effect (if the feature was already disabled) or results in a permission error.

/sys/class/watchdog/watchdogn/options

Defined on file *sysfs-class-watchdog*

It is a read only file. It contains options of watchdog device.

/sys/class/watchdog/watchdogn/pretimeout

Defined on file *sysfs-class-watchdog*

It is a read only file. It specifies the time in seconds before timeout when the pretimeout interrupt is delivered. Pretimeout is an optional feature.

/sys/class/watchdog/watchdogn/pretimeout_avaialable_governors

Defined on file *sysfs-class-watchdog*

It is a read only file. It shows the pretimeout governors available for this watchdog.

/sys/class/watchdog/watchdogn/pretimeout_governor

Defined on file *sysfs-class-watchdog*

It is a read/write file. When read, the currently assigned pretimeout governor is returned. When written, it sets the pretimeout governor.

/sys/class/watchdog/watchdogn/state

Defined on file *sysfs-class-watchdog*

It is a read only file. It gives active/inactive status of watchdog device.

/sys/class/watchdog/watchdogn/status

Defined on file *sysfs-class-watchdog*

It is a read only file. It contains watchdog device's internal status bits. It is equivalent to WDIOC_GETSTATUS of ioctl interface.

/sys/class/watchdog/watchdogn/timeleft

Defined on file *sysfs-class-watchdog*

It is a read only file. It contains value of time left for reset generation. It is equivalent to WDIOC_GETTIMELEFT of ioctl interface.

/sys/class/watchdog/watchdogn/timeout

Defined on file *sysfs-class-watchdog*

It is a read only file. It is read to know about current value of timeout programmed.

/sys/class/zram-control/

Defined on file *sysfs-class-zram*

The zram-control/ class sub-directory belongs to zram device class

/sys/class/zram-control/hot_add

Defined on file *sysfs-class-zram*

RO attribute. Read operation will cause zram to add a new device and return its device id back to user (so one can use /dev/zram<id>), or error code.

/sys/class/zram-control/hot_remove

Defined on file *sysfs-class-zram*

WO attribute. Remove a specific /dev/zramX device, where X is a device_id provided by user.

5.2.33 Symbols under /sys/dev

/sys/dev

Defined on file *sysfs-dev*

The */sys/dev* tree provides a method to look up the sysfs path for a device using the information returned from stat(2). There are two directories, 'block' and 'char', beneath */sys/dev* containing symbolic links with names of the form "<major>:<minor>". These links point to the corresponding sysfs path for the given device.

Example:

```
$ readlink /sys/dev/block/8:32
../../block/sdc
```

Entries in /sys/dev/char and /sys/dev/block will be dynamically created and destroyed as devices enter and leave the system.

Users: mdadm <linux-raid@vger.kernel.org>

5.2.34 Symbols under /sys/devices

/sys/devices

Defined on file *sysfs-devices*

The */sys/devices* tree contains a snapshot of the internal state of the kernel device tree. Devices will be added and removed dynamically as the machine runs, and between different kernel versions, the layout of the devices within this tree will change.

Please do not rely on the format of this tree because of this. If a program wishes to find different things in the tree, please use the /sys/class structure and rely on the symlinks there to point to the proper location within the */sys/devices* tree of the individual devices. Or rely on the uevent messages to notify programs of devices being added and removed from this tree to find the location of those devices.

Note that sometimes not all devices along the directory chain will have emitted uevent messages, so userspace programs must be able to handle such occurrences.

Users:

udev <linux-hotplug-devel@lists.sourceforge.net>

/sys/devices/*/<our-device>/fuse

Defined on file *sysfs-driver-tegra-fuse*

read-only access to the efuses on Tegra20, Tegra30, Tegra114 and Tegra124 SoC's from NVIDIA. The efuses contain write once data programmed at the factory. The data is laid out in 32bit words in LSB first format. Each bit represents a single value as decoded from the fuse registers. Bits order/assignment exactly matches the HW registers, including any unused bits.

Users: any user space application which wants to read the efuses on Tegra SoC's

/sys/devices/*/<our-device>/nvmem

Defined on file *sysfs-driver-jz4780-efuse*

read-only access to the efuse on the Ingenic JZ4780 SoC The SoC has a one time programmable 8K efuse that is split into segments. The driver supports read only. The segments are:

0x000	64 bit	Random Number
0x008	128 bit	Ingenic Chip ID
0x018	128 bit	Customer ID
0x028	3520 bit	Reserved
0x1E0	8 bit	Protect Segment
0x1E1	2296 bit	HDMI Key
0x300	2048 bit	Security boot key

Users: any user space application which wants to read the Chip and Customer ID

/sys/devices/*/xenbus/event_channels

Defined on file *sysfs-devices-xenbus*

Number of Xen event channels associated with a kernel based paravirtualized device frontend or backend.

/sys/devices/*/xenbus/events

Defined on file *sysfs-devices-xenbus*

Total number of Xen events received for a Xen pv device frontend or backend.

/sys/devices/*/xenbus/jiffies_eoi_delayed

Defined on file *sysfs-devices-xenbus*

Summed up time in jiffies the EOI of an interrupt for a Xen pv device has been delayed in order to avoid stalls due to event storms. This value rising is a first sign for a rogue other end of the pv device.

/sys/devices/*/xenbus/spurious_events

Defined on file *sysfs-devices-xenbus*

Number of events received for a Xen pv device which did not require any action. Too many spurious events in a row will trigger delayed EOI processing.

/sys/devices/*/xenbus/spurious_threshold

Defined on file *sysfs-devices-xenbus*

Controls the tolerated number of subsequent spurious events before delayed EOI processing is triggered for a Xen pv device. Default is 1. This can be modified in case the other end of the pv device is issuing spurious events on a regular basis and is known not to be malicious on purpose. Raising the value for such cases can improve pv device performance.

/sys/devices/.../consumer:<consumer>

Defined on file *sysfs-devices-consumer*

The /sys/devices/.../consumer:<consumer> are symlinks to device links where this device is the supplier. <consumer> denotes the name of the consumer in that device link and is of the form bus:device name. There can be zero or more of these symlinks for a given device.

/sys/devices/.../coredump

Defined on file *sysfs-devices-coredump*

The /sys/devices/.../coredump attribute is only present when the device is bound to a driver, which provides the .coredump() callback. The attribute is write only. Anything written to this file will trigger the .coredump() callback.

Available when CONFIG_DEV_COREDUMP is enabled.

/sys/devices/.../firmware_node/

Defined on file *sysfs-devices-firmware_node*

The /sys/devices/.../firmware_node directory contains attributes allowing the user space to check and modify some firmware related properties of given device.

/sys/devices/.../firmware_node/description

Defined on file *sysfs-devices-firmware_node*

The /sys/devices/.../firmware/description attribute contains a string that describes the device as provided by the _STR method in the ACPI namespace. This attribute is read-only. If the device does not have an _STR method associated with it in the ACPI namespace, this attribute is not present.

/sys/devices/.../iommu

Defined on file *sysfs-class-iommu*

IOMMU drivers are able to link the IOMMU for a given device here to allow association of device to IOMMU.

/sys/devices/.../lpss_ltr/

Defined on file *sysfs-devices-lpss_ltr*

The /sys/devices/.../lpss_ltr/ directory is only present for devices included into the Intel Lynx-point Low Power Subsystem (LPSS). If present, it contains attributes containing the LTR mode and the values of LTR registers of the device.

/sys/devices/.../lpss_ltr/auto_ltr

Defined on file *sysfs-devices-lpss_ltr*

The /sys/devices/.../lpss_ltr/auto_ltr attribute contains the current value of the device's AUTO_LTR register (raw) represented as an 8-digit hexadecimal number.

This attribute is read-only. If the device's runtime PM status is not ”active”, attempts to read from this attribute cause -EAGAIN to be returned.

/sys/devices/.../lpss_ltr/ltr_mode

Defined on file *sysfs-devices-lpss_ltr*

The /sys/devices/.../lpss_ltr/ltr_mode attribute contains an integer number (0 or 1) indicating whether or not the devices' LTR functionality is working in the software mode (1).

This attribute is read-only. If the device's runtime PM status is not ”active”, attempts to read from this attribute cause -EAGAIN to be returned.

/sys/devices/.../lpss_ltr/sw_ltr

Defined on file *sysfs-devices-lpss_ltr*

The /sys/devices/.../lpss_ltr/auto_ltr attribute contains the current value of the device's SW_LTR register (raw) represented as an 8-digit hexadecimal number.

This attribute is read-only. If the device's runtime PM status is not "active", attempts to read from this attribute cause -EAGAIN to be returned.

/sys/devices/.../mmc_host/mmcX/mmcX:XXXX/enhanced_area_offset

Defined on file *sysfs-devices-mmc*

Enhanced area is a new feature defined in eMMC4.4 standard. eMMC4.4 or later card can support such feature. This kind of area can help to improve the card performance. If the feature is enabled, this attribute will indicate the start address of enhanced data area. If not, this attribute will be -EINVAL. Unit Byte. Format decimal.

/sys/devices/.../mmc_host/mmcX/mmcX:XXXX/enhanced_area_size

Defined on file *sysfs-devices-mmc*

Enhanced area is a new feature defined in eMMC4.4 standard. eMMC4.4 or later card can support such feature. This kind of area can help to improve the card performance. If the feature is enabled, this attribute will indicate the size of enhanced data area. If not, this attribute will be -EINVAL. Unit KByte. Format decimal.

/sys/devices/.../online

Defined on file *sysfs-devices-online*

The /sys/devices/.../online attribute is only present for devices whose bus types provide .online() and .offline() callbacks. The number read from it (0 or 1) reflects the value of the device's 'offline' field. If that number is 1 and '0' (or 'n', or 'N') is written to this file, the device bus type's .offline() callback is executed for the device and (if successful) its 'offline' field is updated accordingly. In turn, if that number is 0 and '1' (or 'y', or 'Y') is written to this file, the device bus type's .online() callback is executed for the device and (if successful) its 'offline' field is updated as appropriate.

After a successful execution of the bus type's .offline() callback the device cannot be used for any purpose until either it is removed (i.e. device_del() is called for it), or its bus type's .online() is executed successfully.

/sys/devices/.../physical_location

Defined on file *sysfs-devices-physical_location*

This directory contains information on physical location of the device connection point with respect to the system's housing.

/sys/devices/.../physical_location/dock

Defined on file *sysfs-devices-physical_location*

”Yes” if the device connection point resides in a docking station or a port replicator. ”No” otherwise.

/sys/devices/.../physical_location/horizontal_position

Defined on file *sysfs-devices-physical_location*

Describes horizontal position of the device connection point on the panel surface.

/sys/devices/.../physical_location/lid

Defined on file *sysfs-devices-physical_location*

”Yes” if the device connection point resides on the lid of laptop system. ”No” otherwise.

/sys/devices/.../physical_location/panel

Defined on file *sysfs-devices-physical_location*

Describes which panel surface of the system’s housing the device connection point resides on.

/sys/devices/.../physical_location/vertical_position

Defined on file *sysfs-devices-physical_location*

Describes vertical position of the device connection point on the panel surface.

/sys/devices/.../power/

Defined on file *sysfs-devices-power*

The /sys/devices/.../power directory contains attributes allowing the user space to check and modify some power management related properties of given device.

/sys/devices/.../power/async

Defined on file *sysfs-devices-power*

The /sys/devices/.../async attribute allows the user space to enable or diasble the device's suspend and resume callbacks to be executed asynchronously (ie. in separate threads, in parallel with the main suspend/resume thread) during system-wide power transitions (eg. suspend to RAM, hibernation).

All devices have one of the following two values for the power/async file:

- ”enabledn” to permit the asynchronous suspend/resume;
- ”disabledn” to forbid it;

The value of this attribute may be changed by writing either ”enabled”, or ”disabled” to it.

It generally is unsafe to permit the asynchronous suspend/resume of a device unless it is certain that all of the PM dependencies of the device are known to the PM core. However, for some

devices this attribute is set to ”enabled” by bus type code or device drivers and in that cases it should be safe to leave the default value.

/sys/devices/.../power/autosuspend_delay_ms

Defined on file *sysfs-devices-power*

The /sys/devices/.../power/autosuspend_delay_ms attribute contains the autosuspend delay value (in milliseconds). Some drivers do not want their device to suspend as soon as it becomes idle at run time; they want the device to remain inactive for a certain minimum period of time first. That period is called the autosuspend delay. Negative values will prevent the device from being suspended at run time (similar to writing ”on” to the power/control attribute). Values >= 1000 will cause the autosuspend timer expiration to be rounded up to the nearest second.

Not all drivers support this attribute. If it isn't supported, attempts to read or write it will yield I/O errors.

/sys/devices/.../power/control

Defined on file *sysfs-devices-power*

The /sys/devices/.../power/control attribute allows the user space to control the run-time power management of the device.

All devices have one of the following two values for the power/control file:

- ”auton” to allow the device to be power managed at run time;
- ”onn” to prevent the device from being power managed;

The default for all devices is ”auto”, which means that they may be subject to automatic power management, depending on their drivers. Changing this attribute to ”on” prevents the driver from power managing the device at run time. Doing that while the device is suspended causes it to be woken up.

/sys/devices/.../power/pm_qos_latency_tolerance_us

Defined on file *sysfs-devices-power*

The /sys/devices/.../power/pm_qos_latency_tolerance_us attribute contains the PM QoS active state latency tolerance limit for the given device in microseconds. That is the maximum memory access latency the device can suffer without any visible adverse effects on user space functionality. If that value is the string ”any”, the latency does not matter to user space at all, but hardware should not be allowed to set the latency tolerance for the device automatically.

Reading ”auto” from this file means that the maximum memory access latency for the device may be determined automatically by the hardware as needed. Writing ”auto” to it allows the hardware to be switched to this mode if there are no other latency tolerance requirements from the kernel side.

This attribute is only present if the feature controlled by it is supported by the hardware.

This attribute has no effect on runtime suspend and resume of devices and on system-wide suspend/resume and hibernation.

/sys/devices/.../power/pm_qos_no_power_off

Defined on file *sysfs-devices-power*

The /sys/devices/.../power/pm_qos_no_power_off attribute is used for manipulating the PM QoS "no power off" flag. If set, this flag indicates to the kernel that power should not be removed entirely from the device.

Not all drivers support this attribute. If it isn't supported, it is not present.

This attribute has no effect on system-wide suspend/resume and hibernation.

/sys/devices/.../power/pm_qos_resume_latency_us

Defined on file *sysfs-devices-power*

The /sys/devices/.../power/pm_qos_resume_latency_us attribute contains the PM QoS resume latency limit for the given device, which is the maximum allowed time it can take to resume the device, after it has been suspended at run time, from a resume request to the moment the device will be ready to process I/O, in microseconds. If it is equal to 0, however, this means that the PM QoS resume latency may be arbitrary and the special value "n/a" means that user space cannot accept any resume latency at all for the given device.

Not all drivers support this attribute. If it isn't supported, it is not present.

This attribute has no effect on system-wide suspend/resume and hibernation.

/sys/devices/.../power/runtime_active_kids

Defined on file *sysfs-devices-power*

Reports the runtime PM children usage count of a device, or 0 if the children will be ignored.

/sys/devices/.../power/runtime_active_time

Defined on file *sysfs-devices-power*

Reports the total time that the device has been active. Used for runtime PM statistics.

/sys/devices/.../power/runtime_enabled

Defined on file *sysfs-devices-power*

Is runtime PM enabled for this device? States are "enabled", "disabled", "forbidden" or a combination of the latter two.

/sys/devices/.../power/runtime_status

Defined on file *sysfs-devices-power*

The /sys/devices/.../power/runtime_status attribute contains the current runtime PM status of the device, which may be "suspended", "suspending", "resuming", "active", "error" (fatal error), or "unsupported" (runtime PM is disabled).

/sys/devices/.../power/runtime_suspended_time

Defined on file *sysfs-devices-power*

Reports total time that the device has been suspended. Used for runtime PM statistics.

/sys/devices/.../power/runtime_usage

Defined on file *sysfs-devices-power*

Reports the runtime PM usage count of a device.

/sys/devices/.../power/wakeup

Defined on file *sysfs-devices-power*

The /sys/devices/.../power/wakeup attribute allows the user space to check if the device is enabled to wake up the system from sleep states, such as the memory sleep state (suspend to RAM) and hibernation (suspend to disk), and to enable or disable it to do that as desired.

Some devices support "wakeup" events, which are hardware signals used to activate the system from a sleep state. Such devices have one of the following two values for the sysfs power/wakeup file:

- "enabledn" to issue the events;
- "disabledn" not to do so;

In that cases the user space can change the setting represented by the contents of this file by writing either "enabled", or "disabled" to it.

For the devices that are not capable of generating system wakeup events this file is not present. In that case the device cannot be enabled to wake up the system from sleep states.

/sys/devices/.../power/wakeup_abort_count

Defined on file *sysfs-devices-power*

The /sys/devices/.../wakeup_abort_count attribute contains the number of times the processing of a wakeup event associated with the device might have aborted system transition into a sleep state in progress. This attribute is read-only. If the device is not capable to wake up the system from sleep states, this attribute is not present. If the device is not enabled to wake up the system from sleep states, this attribute is empty.

/sys/devices/.../power/wakeup_active

Defined on file *sysfs-devices-power*

The /sys/devices/.../wakeup_active attribute contains either 1, or 0, depending on whether or not a wakeup event associated with the device is being processed (1). This attribute is read-only. If the device is not capable to wake up the system from sleep states, this attribute is not present. If the device is not enabled to wake up the system from sleep states, this attribute is empty.

/sys/devices/.../power/wakeup_active_count

Defined on file *sysfs-devices-power*

The /sys/devices/.../wakeup_active_count attribute contains the number of times the processing of wakeup events associated with the device was completed (at the kernel level). This attribute is read-only. If the device is not capable to wake up the system from sleep states, this attribute is not present. If the device is not enabled to wake up the system from sleep states, this attribute is empty.

/sys/devices/.../power/wakeup_count

Defined on file *sysfs-devices-power*

The /sys/devices/.../wakeup_count attribute contains the number of signaled wakeup events associated with the device. This attribute is read-only. If the device is not capable to wake up the system from sleep states, this attribute is not present. If the device is not enabled to wake up the system from sleep states, this attribute is empty.

/sys/devices/.../power/wakeup_expire_count

Defined on file *sysfs-devices-power*

The /sys/devices/.../wakeup_expire_count attribute contains the number of times a wakeup event associated with the device has been reported with a timeout that expired. This attribute is read-only. If the device is not capable to wake up the system from sleep states, this attribute is not present. If the device is not enabled to wake up the system from sleep states, this attribute is empty.

/sys/devices/.../power/wakeup_last_time_ms

Defined on file *sysfs-devices-power*

The /sys/devices/.../wakeup_last_time_ms attribute contains the value of the monotonic clock corresponding to the time of signaling the last wakeup event associated with the device, in milliseconds. This attribute is read-only. If the device is not enabled to wake up the system from sleep states, this attribute is not present. If the device is not enabled to wake up the system from sleep states, this attribute is empty.

/sys/devices/.../power/wakeup_max_time_ms

Defined on file *sysfs-devices-power*

The /sys/devices/.../wakeup_max_time_ms attribute contains the maximum time of processing a single wakeup event associated with the device, in milliseconds. This attribute is read-only. If the device is not capable to wake up the system from sleep states, this attribute is not present. If the device is not enabled to wake up the system from sleep states, this attribute is empty.

/sys/devices/.../power/wakeup_prevent_sleep_time_ms

Defined on file *sysfs-devices-power*

The /sys/devices/.../wakeup_prevent_sleep_time_ms attribute contains the total time the device has been preventing opportunistic transitions to sleep states from occurring. This attribute is read-only. If the device is not capable to wake up the system from sleep states, this attribute is not present. If the device is not enabled to wake up the system from sleep states, this attribute is empty.

/sys/devices/.../power/wakeup_total_time_ms

Defined on file *sysfs-devices-power*

The /sys/devices/.../wakeup_total_time_ms attribute contains the total time of processing wakeup events associated with the device, in milliseconds. This attribute is read-only. If the device is not capable to wake up the system from sleep states, this attribute is not present. If the device is not enabled to wake up the system from sleep states, this attribute is empty.

/sys/devices/.../power_resources_D0/

Defined on file *sysfs-devices-power_resources_D0*

The /sys/devices/.../power_resources_D0/ directory is only present for device objects representing ACPI device nodes that use ACPI power resources for power management.

If present, it contains symbolic links to device directories representing ACPI power resources that need to be turned on for the given device node to be in ACPI power state D0. The names of the links are the same as the names of the directories they point to.

/sys/devices/.../power_resources_D1/

Defined on file *sysfs-devices-power_resources_D1*

The /sys/devices/.../power_resources_D1/ directory is only present for device objects representing ACPI device nodes that use ACPI power resources for power management and support ACPI power state D1.

If present, it contains symbolic links to device directories representing ACPI power resources that need to be turned on for the given device node to be in ACPI power state D1. The names of the links are the same as the names of the directories they point to.

/sys/devices/.../power_resources_D2/

Defined on file *sysfs-devices-power_resources_D2*

The /sys/devices/.../power_resources_D2/ directory is only present for device objects representing ACPI device nodes that use ACPI power resources for power management and support ACPI power state D2.

If present, it contains symbolic links to device directories representing ACPI power resources that need to be turned on for the given device node to be in ACPI power state D2. The names of the links are the same as the names of the directories they point to.

/sys/devices/.../power_resources_D3hot/

Defined on file *sysfs-devices-power_resources_D3hot*

The /sys/devices/.../power_resources_D3hot/ directory is only present for device objects representing ACPI device nodes that use ACPI power resources for power management and support ACPI power state D3hot.

If present, it contains symbolic links to device directories representing ACPI power resources that need to be turned on for the given device node to be in ACPI power state D3hot. The names of the links are the same as the names of the directories they point to.

/sys/devices/.../power_resources_wakeup/

Defined on file *sysfs-devices-power_resources_wakeup*

The /sys/devices/.../power_resources_wakeup/ directory is only present for device objects representing ACPI device nodes that require ACPI power resources for wakeup signaling.

If present, it contains symbolic links to device directories representing ACPI power resources that need to be turned on for the given device node to be able to signal wakeup. The names of the links are the same as the names of the directories they point to.

/sys/devices/.../power_state

Defined on file *sysfs-devices-power_state*

The /sys/devices/.../power_state attribute is only present for device objects representing ACPI device nodes that provide power management methods.

If present, it contains a string representing the current ACPI power state of the given device node. Its possible values, "D0", "D1", "D2", "D3hot", and "D3cold", reflect the power state names defined by the ACPI specification (ACPI 4 and above).

If the device node uses shared ACPI power resources, this state determines a list of power resources required not to be turned off. However, some power resources needed by the device node in higher-power (lower-number) states may also be ON because of some other devices using them at the moment.

This attribute is read-only.

/sys/devices/.../real_power_state

Defined on file *sysfs-devices-real_power_state*

The /sys/devices/.../real_power_state attribute is only present for device objects representing ACPI device nodes that provide power management methods and use ACPI power resources for power management.

If present, it contains a string representing the real ACPI power state of the given device node as returned by the _PSC control method or inferred from the configuration of power resources. Its possible values, "D0", "D1", "D2", "D3hot", and "D3cold", reflect the power state names defined by the ACPI specification (ACPI 4 and above).

In some situations the value of this attribute may be different from the value of the /sys/devices/.../power_state attribute for the same device object. If that happens, some shared power resources used by the device node are only ON because of some other devices using them at the moment.

This attribute is read-only.

/sys/devices/.../removable

Defined on file *sysfs-devices-removable*

Information about whether a given device can be removed from the platform by the user. This is determined by its subsystem in a bus / platform-specific way. This attribute is only present for devices that can support determining such information:

"removable"	device can be removed from the platform by the user
"fixed"	device is fixed to the platform / cannot be removed by the user.
"unknown"	The information is unavailable / cannot be deduced.

Currently this is only supported by USB (which infers the information from a combination of hub descriptor bits and platform-specific data such as ACPI) and PCI (which gets this from ACPI / device tree).

/sys/devices/.../resource_in_use

Defined on file *sysfs-devices-resource_in_use*

The /sys/devices/.../resource_in_use attribute is only present for device objects representing ACPI power resources.

If present, it contains a number (0 or 1) representing the current status of the given power resource (0 means that the resource is not in use and therefore it has been turned off).

This attribute is read-only.

/sys/devices/.../software_node/

Defined on file *sysfs-devices-software_node*

This directory contains the details about the device that are assigned in kernel (i.e. software), as opposed to the firmware_node directory which contains the details that are assigned for the device in firmware. The main attributes in the directory will show the properties the device has, and the relationship it has to some of the other devices.

/sys/devices/.../state_synced

Defined on file *sysfs-devices-state_synced*

The /sys/devices/.../state_synced attribute is only present for devices whose bus types or driver provides the .sync_state() callback. The number read from it (0 or 1) reflects the value of the device's 'state_synced' field. A value of 0 means the .sync_state() callback hasn't been called yet. A value of 1 means the .sync_state() callback has been called.

Generally, if a device has sync_state() support and has some of the resources it provides enabled at the time the kernel starts (Eg: enabled by hardware reset or bootloader or anything that run before the kernel starts), then it'll keep those resources enabled and in a state that's compatible with the state they were in at the start of the kernel. The device will stop doing this only when

the sync_state() callback has been called -- which happens only when all its consumer devices are registered and have probed successfully. Resources that were left disabled at the time the kernel starts are not affected or limited in any way by sync_state() callbacks.

Writing "1" to this file will force a call to the device's sync_state() function if it hasn't been called already. The sync_state() call happens independent of the state of the consumer devices.

/sys/devices/.../sun

Defined on file *sysfs-devices-sun*

The file contains a Slot-unique ID which provided by the _SUN method in the ACPI namespace. The value is written in Advanced Configuration and Power Interface Specification as follows:

"The _SUN value is required to be unique among the slots of the same type. It is also recommended that this number match the slot number printed on the physical slot whenever possible."

So reading the sysfs file, we can identify a physical position of the slot in the system.

/sys/devices/.../supplier:<supplier>

Defined on file *sysfs-devices-supplier*

The /sys/devices/.../supplier:<supplier> are symlinks to device links where this device is the consumer. <supplier> denotes the name of the supplier in that device link and is of the form bus:device name. There can be zero or more of these symlinks for a given device.

/sys/devices/.../waiting_for_supplier

Defined on file *sysfs-devices-waiting_for_supplier*

The /sys/devices/.../waiting_for_supplier attribute is only present when fw_devlink kernel command line option is enabled and is set to something stricter than "permissive". It is removed once a device probes successfully (because the information is no longer relevant). The number read from it (0 or 1) reflects whether the device is waiting for one or more suppliers to be added and then linked to using device links before the device can probe.

A value of 0 means the device is not waiting for any suppliers to be added before it can probe. A value of 1 means the device is waiting for one or more suppliers to be added before it can probe.

/sys/devices/../../gisb_arb_timeout

Defined on file *sysfs-platform-brcmstb-gisb-arb*

Returns the currently configured raw timeout value of the Broadcom Set Top Box internal GISB bus arbiter. Minimum value is 1, and maximum value is 0xffffffff.

/sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS620A:00/protection_level

Defined on file *sysfs-driver-toshiba_haps*

This file controls the built-in accelerometer protection level, valid values are:

- 0 -> Disabled
- 1 -> Low
- 2 -> Medium
- 3 -> High

The default potection value is set to 2 (Medium).

Users: KToshiba

/sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS620A:00/reset_protection

Defined on file *sysfs-driver-toshiba_haps*

This file turns off the built-in accelerometer for a few seconds and then restore normal operation. Accepting 1 as the only parameter.

/sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/available_kbd_modes

Defined on file *sysfs-driver-toshiba_acpi*

This file shows the supported keyboard backlight modes the system supports, which can be:

- 0x1 -> FN-Z
- 0x2 -> AUTO (also called TIMER)
- 0x8 -> ON
- 0x10 -> OFF

Note that not all keyboard types support the listed modes. See the entry named "available_kbd_modes"

Users: KToshiba

/sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/cooling_method

Defined on file *sysfs-driver-toshiba_acpi*

This file controls the Cooling Method feature. Reading this file prints two values, the first is the actual cooling method and the second is the maximum cooling method supported. When the maximum cooling method is ONE, valid values are:

- 0 -> Maximum Performance
- 1 -> Battery Optimized

When the maximum cooling method is TWO, valid values are:

- 0 -> Maximum Performance
- 1 -> Performance
- 2 -> Battery Optimized

Users: KToshiba

/sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/fan

Defined on file *sysfs-driver-toshiba_acpi*

This file controls the state of the internal fan, valid values are:

- 0 -> OFF
- 1 -> ON

/sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/kbd_backlight_mode

Defined on file *sysfs-driver-toshiba_acpi*

This file controls the keyboard backlight operation mode, valid values are:

- 0x1 -> FN-Z
- 0x2 -> AUTO (also called TIMER)
- 0x8 -> ON
- 0x10 -> OFF

Note that from kernel 3.16 onwards this file accepts all listed parameters, kernel 3.15 only accepts the first two (FN-Z and AUTO). Also note that toggling this value on type 1 devices, requires a reboot for changes to take effect.

Users: KToshiba

/sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/kbd_backlight_timeou

Defined on file *sysfs-driver-toshiba_acpi*

This file controls the timeout of the keyboard backlight whenever the operation mode is set to AUTO (or TIMER), valid values range from 0-60. Note that the kernel 3.15 only had support for the first keyboard type, the kernel 3.16 added support for the second type and the range accepted for type 2 is 1-60. See the entry named ”kbd_type”

Users: KToshiba

/sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/kbd_function_keys

Defined on file *sysfs-driver-toshiba_acpi*

This file controls the Special Functions (hotkeys) operation mode, valid values are:

- 0 -> Normal Operation
- 1 -> Special Functions

In the ”Normal Operation” mode, the F{1-12} keys are as usual and the hotkeys are accessed via FN-F{1-12}. In the ”Special Functions” mode, the F{1-12} keys trigger the hotkey and the F{1-12} keys are accessed via FN-F{1-12}. Note that toggling this value requires a reboot for changes to take effect.

Users: KToshiba

/sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/kbd_type

Defined on file *sysfs-driver-toshiba_acpi*

This file shows the current keyboard backlight type, which can be:

- 1 -> Type 1, supporting modes FN-Z and AUTO
- 2 -> Type 2, supporting modes TIMER, ON and OFF

Users: KToshiba

/sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/panel_power_on

Defined on file *sysfs-driver-toshiba_acpi*

This file controls whether the laptop should turn ON whenever the LID is opened, valid values are:

- 0 -> Disabled
- 1 -> Enabled

Note that toggling this value requires a reboot for changes to take effect.

Users: KToshiba

/sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/position

Defined on file *sysfs-driver-toshiba_acpi*

This file shows the absolute position of the built-in accelereometer.

/sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/sleep_functions_on_b

Defined on file *sysfs-driver-toshiba_acpi*

This file controls the USB Sleep Functions under battery, and set the level at which point they will be disabled, accepted values can be:

- 0 -> Disabled
- 1-100 -> Battery level to disable sleep functions

Currently it prints two values, the first one indicates if the feature is enabled or disabled, while the second one shows the current battery level set. Note that when the value is set to disabled, the sleep function will only work when connected to power.

Users: KToshiba

/sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/touchpad

Defined on file *sysfs-driver-toshiba_acpi*

This files controls the status of the touchpad and pointing stick (if available), valid values are:

- 0 -> OFF
- 1 -> ON

Users: KToshiba

/sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/usb_rapid_charge

Defined on file *sysfs-driver-toshiba_acpi*

This file controls the USB Rapid Charge state, which can be:

- 0 -> Disabled
- 1 -> Enabled

Note that toggling this value requires a reboot for changes to take effect.

Users: KToshiba

/sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/usb_sleep_charge

Defined on file *sysfs-driver-toshiba_acpi*

This file controls the USB Sleep & Charge charging mode, which can be:

- 0 -> Disabled (0x00)
- 1 -> Alternate (0x09)
- 2 -> Auto (0x21)
- 3 -> Typical (0x11)

Note that from kernel 4.1 onwards this file accepts all listed values, kernel 4.0 only supports the first three. Note that this feature only works when connected to power, if you want to use it under battery, see the entry named "sleep_functions_on_battery"

Users: KToshiba

/sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/usb_sleep_music

Defined on file *sysfs-driver-toshiba_acpi*

This file controls the Sleep & Music state, which values can be:

- 0 -> Disabled
- 1 -> Enabled

Note that this feature only works when connected to power, if you want to use it under battery, see the entry named "sleep_functions_on_battery"

Users: KToshiba

/sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/usb_three

Defined on file *sysfs-driver-toshiba_acpi*

This file controls the USB 3 functionality, valid values are:

- 0 -> Disabled (Acts as a regular USB 2)
- 1 -> Enabled (Full USB 3 functionality)

Note that toggling this value requires a reboot for changes to take effect.

Users: KToshiba

/sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/version

Defined on file *sysfs-driver-toshiba_acpi*

This file shows the current version of the driver

Users: KToshiba

/sys/devices/cpu/events/

Defined on file *sysfs-bus-event_source-devices-events*

Generic performance monitoring events

A collection of performance monitoring events that may be supported by many/most CPUs. These events can be monitored using the 'perf(1)' tool.

The contents of each file would look like:

event=0xNNNN

where 'N' is a hex digit and the number '0xNNNN' shows the "raw code" for the perf event identified by the file's "basename".

/sys/devices/hisi_ptt<sicl_id>_<core_id>/requester_filters

Defined on file *sysfs-devices-hisi_ptt*

This directory contains the files providing the PCIe Requester filters information used for PTT trace. Each file is named after the supported Endpoint device name <domain>:<bus>:<device>.<function>.

See the description of the "filter" in /trace/hisi-ptt for more information.

/sys/devices/hisi_ptt<sicl_id>_<core_id>/requester_filters/<bdf>

Defined on file *sysfs-devices-hisi_ptt*

(Read) Indicates the filter value of this Requester filter, which can be used to control the TLP headers to trace by the PTT trace.

/sys/devices/hisi_ptt<sicl_id>_<core_id>/requester_filters/multiselect

Defined on file *sysfs-devices-hisi_ptt*

(Read) Indicates if this kind of filter can be selected at the same time as others filters, or must be used on it's own. 1 indicates the former case and 0 indicates the latter.

/sys/devices/hisi_ptt<sicl_id>_<core_id>/root_port_filters

Defined on file *sysfs-devices-hisi_ptt*

This directory contains the files providing the PCIe Root Port filters information used for PTT trace. Each file is named after the supported Root Port device name <domain>:<bus>:<device>.<function>.

See the description of the "filter" in /trace/hisi-ptt for more information.

/sys/devices/hisi_ptt<sicl_id>_<core_id>/root_port_filters/<bdf>

Defined on file *sysfs-devices-hisi_ptt*

(Read) Indicates the filter value of this Root Port filter, which can be used to control the TLP headers to trace by the PTT trace.

/sys/devices/hisi_ptt<sicl_id>_<core_id>/root_port_filters/multiselect

Defined on file *sysfs-devices-hisi_ptt*

(Read) Indicates if this kind of filter can be selected at the same time as others filters, or must be used on it's own. 1 indicates the former case and 0 indicates the latter.

/sys/devices/hisi_ptt<sicl_id>_<core_id>/tune

Defined on file *sysfs-devices-hisi_ptt*

This directory contains files for tuning the PCIe link parameters(events). Each file is named after the event of the PCIe link.

See /trace/hisi-ptt for more information.

/sys/devices/hisi_ptt<sicl_id>_<core_id>/tune/qos_tx_cpl

Defined on file *sysfs-devices-hisi_ptt*

(RW) Controls the weight of Tx completion TLPs, which influence the proportion of outbound completion TLPs on the PCIe link. The available tune data is [0, 1, 2]. Writing a negative value will return an error, and out of range values will be converted to 2. The value indicates a probable level of the event.

/sys/devices/hisi_ptt<sicl_id>_<core_id>/tune/qos_tx_np

Defined on file *sysfs-devices-hisi_ptt*

(RW) Controls the weight of Tx non-posted TLPs, which influence the proportion of outbound non-posted TLPs on the PCIe link. The available tune data is [0, 1, 2]. Writing a negative value will return an error, and out of range values will be converted to 2. The value indicates a probable level of the event.

/sys/devices/hisi_ptt<sicl_id>_<core_id>/tune/qos_tx_p

Defined on file *sysfs-devices-hisi_ptt*

(RW) Controls the weight of Tx posted TLPs, which influence the proportion of outbound posted TLPs on the PCIe link. The available tune data is [0, 1, 2]. Writing a negative value will return an error, and out of range values will be converted to 2. The value indicates a probable level of the event.

/sys/devices/hisi_ptt<sicl_id>_<core_id>/tune/rx_alloc_buf_level

Defined on file *sysfs-devices-hisi_ptt*

(RW) Control the allocated buffer watermark for inbound packets. The packets will be stored in the buffer first and then transmitted either when the watermark reached or when timed out. The available tune data is [0, 1, 2]. Writing a negative value will return an error, and out of range values will be converted to 2. The value indicates a probable level of the event.

/sys/devices/hisi_ptt<sicl_id>_<core_id>/tune/tx_alloc_buf_level

Defined on file *sysfs-devices-hisi_ptt*

(RW) Control the allocated buffer watermark of outbound packets. The packets will be stored in the buffer first and then transmitted either when the watermark reached or when timed out. The available tune data is [0, 1, 2]. Writing a negative value will return an error, and out of range values will be converted to 2. The value indicates a probable level of the event.

/sys/devices/hv_24x7/cpumask

Defined on file *sysfs-bus-event_source-devices-hv_24x7*

read only This sysfs file exposes the cpumask which is designated to make HCALLs to retrieve hv-24x7 pmu event counter data.

/sys/devices/hv_24x7/interface/chipspersocket

Defined on file *sysfs-bus-event_source-devices-hv_24x7*

read only This sysfs interface exposes the number of chips per socket present in the system.

/sys/devices/hv_24x7/interface/coresperchip

Defined on file *sysfs-bus-event_source-devices-hv_24x7*

read only This sysfs interface exposes the number of cores per chip present in the system.

/sys/devices/hv_24x7/interface/sockets

Defined on file *sysfs-bus-event_source-devices-hv_24x7*

read only This sysfs interface exposes the number of sockets present in the system.

/sys/devices/hv_gpci/cpumask

Defined on file *sysfs-bus-event_source-devices-hv_gpci*

read only This sysfs file exposes the cpumask which is designated to make HCALLs to retrieve hv-gpci pmu event counter data.

/sys/devices/hv_gpci/interface/affinity_domain_via_domain

Defined on file *sysfs-bus-event_source-devices-hv_gpci*

admin read only This sysfs file exposes the system topology information by making HCALL H_GET_PERF_COUNTER_INFO. The HCALL is made with counter request value AFFINITY_DOMAIN_INFORMATION_BY_DOMAIN(0xB0).

- This sysfs file will be created only for power10 and above platforms.
- User needs root privileges to read data from this sysfs file.
- This sysfs file will be created, only when the HCALL returns "H_SUCCESS", "H_AUTHORITY" or "H_PARAMETER" as the return type.

  HCALL with return error type "H_AUTHORITY" can be resolved during runtime by setting "Enable Performance Information Collection" option.
- The end user reading this sysfs file must decode the content as per underlying platform/firmware.

Possible error codes while reading this sysfs file:

- **"-EPERM"**
  [Partition is not permitted to retrieve performance information,] required to set "Enable Performance Information Collection" option.
- **"-EIO"**
  [Can't retrieve system information because of invalid buffer length/invalid address] or because of some hardware error. Refer to getPerfCountInfo documentation for more information.
- "-EFBIG" : System information exceeds PAGE_SIZE.

/sys/devices/hv_gpci/interface/affinity_domain_via_partition

Defined on file *sysfs-bus-event_source-devices-hv_gpci*

admin read only This sysfs file exposes the system topology information by making HCALL H_GET_PERF_COUNTER_INFO. The HCALL is made with counter request value AFFINITY_DOMAIN_INFORMATION_BY_PARTITION(0xB1).

- This sysfs file will be created only for power10 and above platforms.
- User needs root privileges to read data from this sysfs file.
- This sysfs file will be created, only when the HCALL returns "H_SUCCESS", "H_AUTHORITY" or "H_PARAMETER" as the return type.

  HCALL with return error type "H_AUTHORITY" can be resolved during runtime by setting "Enable Performance Information Collection" option.
- The end user reading this sysfs file must decode the content as per underlying platform/firmware.

Possible error codes while reading this sysfs file:

- **"-EPERM"**
  [Partition is not permitted to retrieve performance information,] required to set "Enable Performance Information Collection" option.
- **"-EIO"**
  [Can't retrieve system information because of invalid buffer length/invalid address] or because of some hardware error. Refer to getPerfCountInfo documentation for more information.
- "-EFBIG" : System information exceeds PAGE_SIZE.

/sys/devices/hv_gpci/interface/affinity_domain_via_virtual_processor

Defined on file *sysfs-bus-event_source-devices-hv_gpci*

admin read only This sysfs file exposes the system topology information by making HCALL H_GET_PERF_COUNTER_INFO. The HCALL is made with counter request value AFFINITY_DOMAIN_INFORMATION_BY_VIRTUAL_PROCESSOR(0xA0).

- This sysfs file will be created only for power10 and above platforms.
- User needs root privileges to read data from this sysfs file.
- This sysfs file will be created, only when the HCALL returns "H_SUCCESS", "H_AUTHORITY" or "H_PARAMETER" as the return type.

  HCALL with return error type "H_AUTHORITY" can be resolved during runtime by setting "Enable Performance Information Collection" option.
- The end user reading this sysfs file must decode the content as per underlying platform/firmware.

Possible error codes while reading this sysfs file:

- **"-EPERM"**
  [Partition is not permitted to retrieve performance information,] required to set "Enable Performance Information Collection" option.
- **"-EIO"**
  [Can't retrieve system information because of invalid buffer length/invalid address] or because of some hardware error. Refer to getPerfCountInfo documentation for more information.
- "-EFBIG" : System information exceeds PAGE_SIZE.

/sys/devices/hv_gpci/interface/processor_bus_topology

Defined on file *sysfs-bus-event_source-devices-hv_gpci*

admin read only This sysfs file exposes the system topology information by making HCALL H_GET_PERF_COUNTER_INFO. The HCALL is made with counter request value PROCESSOR_BUS_TOPOLOGY(0xD0).

- This sysfs file will be created only for power10 and above platforms.
- User needs root privileges to read data from this sysfs file.

- This sysfs file will be created, only when the HCALL returns "H_SUCCESS", "H_AUTHORITY" or "H_PARAMETER" as the return type.

  HCALL with return error type "H_AUTHORITY" can be resolved during runtime by setting "Enable Performance Information Collection" option.

- The end user reading this sysfs file must decode the content as per underlying platform/firmware.

Possible error codes while reading this sysfs file:

- **"-EPERM"**
  [Partition is not permitted to retrieve performance information,] required to set "Enable Performance Information Collection" option.
- **"-EIO"**
  [Can't retrieve system information because of invalid buffer length/invalid address] or because of some hardware error. Refer to getPerfCountInfo documentation for more information.
- "-EFBIG" : System information exceeds PAGE_SIZE.

/sys/devices/hv_gpci/interface/processor_config

Defined on file *sysfs-bus-event_source-devices-hv_gpci*

admin read only This sysfs file exposes the system topology information by making HCALL H_GET_PERF_COUNTER_INFO. The HCALL is made with counter request value PROCESSOR_CONFIG(0x90).

- This sysfs file will be created only for power10 and above platforms.
- User needs root privileges to read data from this sysfs file.
- This sysfs file will be created, only when the HCALL returns "H_SUCCESS", "H_AUTHORITY" or "H_PARAMETER" as the return type.

  HCALL with return error type "H_AUTHORITY" can be resolved during runtime by setting "Enable Performance Information Collection" option.

- The end user reading this sysfs file must decode the content as per underlying platform/firmware.

Possible error codes while reading this sysfs file:

- **"-EPERM"**
  [Partition is not permitted to retrieve performance information,] required to set "Enable Performance Information Collection" option.
- **"-EIO"**
  [Can't retrieve system information because of invalid buffer length/invalid address] or because of some hardware error. Refer to getPerfCountInfo documentation for more information.
- "-EFBIG" : System information exceeds PAGE_SIZE.

/sys/devices/pci0000:00/*/QEMU0001:00/capability for MMIO

Defined on file *sysfs-bus-pci-devices-pvpanic*

Read-only attribute. Capabilities of pvpanic device which are supported by QEMU.

Format: %x.

Detailed bit definition refers to section <Bit Definition> from pvpanic device specification: https://git.qemu.org/?p=qemu.git;a=blob_plain;f=docs/specs/pvpanic.txt

/sys/devices/pci0000:00/*/QEMU0001:00/events

Defined on file *sysfs-bus-pci-devices-pvpanic*

RW attribute. Set/get which features in-use. This attribute is used to enable/disable feature(s) of pvpanic device. Notice that this value should be a subset of capability.

Format: %x.

Also refer to pvpanic device specification.

/sys/devices/pci0000:00/<dev>/fw_info

Defined on file *sysfs-bus-pci-devices-catpt*

Detailed AudioDSP firmware build information including build hash and log-providers hash. This information is obtained during initial handshake with firmware. Format: %s.

/sys/devices/pci0000:00/<dev>/fw_version

Defined on file *sysfs-bus-pci-devices-catpt*

Version of AudioDSP firmware ASoC catpt driver is communicating with.

Format: %d.%d.%d.%d, type:major:minor:build.

/sys/devices/platform/*/cnqf_enable

Defined on file *sysfs-amd-pmf*

Reading this file tells if the AMD Platform Management(PMF) Cool n Quiet Framework(CnQF) feature is enabled or not.

This feature is not enabled by default and gets only turned on if OEM BIOS passes a ”flag” to PMF ACPI function (index 11 or 12) or in case the user writes ”on”.

To turn off CnQF user can write ”off” to the sysfs node. Note: Systems that support auto mode will not have this sysfs file available.

/sys/devices/platform/.../power_on_reason

Defined on file *sysfs-platform-power-on-reason*

Shows system power on reason. The following strings/reasons can be read (the list can be extended): ”regular power-up”, ”RTC wakeup”, ”watchdog timeout”, ”software reset”, ”reset button action”, ”CPU clock failure”, ”crystal oscillator failure”, ”brown-out reset”, ”unknown reason”.

The file is read only.

/sys/devices/platform/8086<x>:00/firmware_version

Defined on file *sysfs-platform-sst-atom*

LPE Firmware version for SST driver on all atom platforms (BYT/CHT/Merrifield/BSW). If the FW has never been loaded it will display:

```
"FW not yet loaded"
```

If FW has been loaded it will display:

```
"v01.aa.bb.cc"
```

aa: Major version is reflecting SoC version:

0d:	BYT FW
0b:	BSW FW
07:	Merrifield FW

bb: Minor version

cc: Build version

/sys/devices/platform/<i2c-demux-name>/available_masters

Defined on file *sysfs-platform-i2c-demux-pinctrl*

Reading the file will give you a list of masters which can be selected for a demultiplexed bus. The format is "<index>:<name>". Example from a Renesas Lager board:

```
0:/i2c@e6500000 1:/i2c@e6508000
```

/sys/devices/platform/<i2c-demux-name>/current_master

Defined on file *sysfs-platform-i2c-demux-pinctrl*

This file selects/shows the active I2C master for a demultiplexed bus. It uses the <index> value from the file 'available_masters'.

/sys/devices/platform/<phy-name>/role

Defined on file *sysfs-platform-phy-rcar-gen3-usb2*

This file can be read and write. The file can show/change the phy mode for role swap of usb.

Write the following strings to change the mode:

- "host" - switching mode from peripheral to host.
- "peripheral" - switching mode from host to peripheral.

Read the file, then it shows the following strings:

- ”host” - The mode is host now.
- ”peripheral” - The mode is peripheral now.

/sys/devices/platform/<platform>/camera

Defined on file *sysfs-platform-asus-wmi*

Control the camera. 1 means on, 0 means off.

/sys/devices/platform/<platform>/cardr

Defined on file *sysfs-platform-asus-wmi*

Control the card reader. 1 means on, 0 means off.

/sys/devices/platform/<platform>/charge_mode

Defined on file *sysfs-platform-asus-wmi*

Get the current charging mode being used:

- 1 - Barrel connected charger,
- 2 - USB-C charging
- 3 - Both connected, barrel used for charging

/sys/devices/platform/<platform>/cpufv

Defined on file *sysfs-platform-asus-wmi*

Change CPU clock configuration (write-only). There are three available clock configuration:

- 0 -> Super Performance Mode
- 1 -> High Performance Mode
- 2 -> Power Saving Mode

/sys/devices/platform/<platform>/dgpu_disable

Defined on file *sysfs-platform-asus-wmi*

Disable discrete GPU:

- 0 - Enable dGPU,
- 1 - Disable dGPU

/sys/devices/platform/<platform>/egpu_connected

Defined on file *sysfs-platform-asus-wmi*

Show if the egpu (XG Mobile) is correctly connected:

- 0 - False,

- 1 - True

/sys/devices/platform/<platform>/egpu_enable

Defined on file *sysfs-platform-asus-wmi*

Enable the external GPU paired with ROG X-Flow laptops. Toggling this setting will also trigger ACPI to disable the dGPU:

- 0 - Disable,
- 1 - Enable

/sys/devices/platform/<platform>/etr3

Defined on file *sysfs-platform-intel-pmc*

The file exposes "Extended Test Mode Register 3" global reset bits. The bits are used during an Intel platform manufacturing process to indicate that consequent reset of the platform is a "global reset". This type of reset is required in order for manufacturing configurations to take effect.

Display global reset setting bits for PMC.

- bit 31 - global reset is locked
- bit 20 - global reset is set

Writing bit 20 value to the etr3 will induce a platform "global reset" upon consequent platform reset, in case the register is not locked. The "global reset bit" should be locked on a production system and the file is in read-only mode.

/sys/devices/platform/<platform>/fan_boost_mode

Defined on file *sysfs-platform-asus-wmi*

Fan boost mode:

- 0 - normal,
- 1 - overboost,
- 2 - silent

/sys/devices/platform/<platform>/force_power

Defined on file *sysfs-platform-intel-wmi-thunderbolt*

Modify the platform force power state, influencing Thunderbolt controllers to turn on or off when no devices are connected (write-only) There are two available states:

- 0 -> Force power disabled
- 1 -> Force power enabled

/sys/devices/platform/<platform>/gpu_mux_mode

Defined on file *sysfs-platform-asus-wmi*

Switch the GPU hardware MUX mode. Laptops with this feature can can be toggled to boot with only the dGPU (discrete mode) or in standard Optimus/Hybrid mode. On switch a reboot is required:

- 0 - Discrete GPU,
- 1 - Optimus/Hybrid,

/sys/devices/platform/<platform>/lid_resume

Defined on file *sysfs-platform-asus-wmi*

Resume on lid open. 1 means on, 0 means off.

/sys/devices/platform/<platform>/mini_led_mode

Defined on file *sysfs-platform-asus-wmi*

Change the mini-LED mode:

- 0 - Single-zone,
- 1 - Multi-zone

/sys/devices/platform/<platform>/nv_dynamic_boost

Defined on file *sysfs-platform-asus-wmi*

Set the dynamic boost limit of the Nvidia dGPU:

- min=5, max=25

/sys/devices/platform/<platform>/nv_temp_target

Defined on file *sysfs-platform-asus-wmi*

Set the target temperature limit of the Nvidia dGPU:

- min=75, max=87

/sys/devices/platform/<platform>/panel_od

Defined on file *sysfs-platform-asus-wmi*

Enable an LCD response-time boost to reduce or remove ghosting:

- 0 - Disable,
- 1 - Enable

/sys/devices/platform/<platform>/ppt_apu_sppt

Defined on file *sysfs-platform-asus-wmi*

Set the APU SPPT limit. Shown on full AMD systems only:

- min=5, max=130

/sys/devices/platform/<platform>/ppt_fppt

Defined on file *sysfs-platform-asus-wmi*

Set the Fast Package Power Tracking Limit of CPU. AMD+Nvidia only:

- min=5, max=250

/sys/devices/platform/<platform>/ppt_pl1_spl

Defined on file *sysfs-platform-asus-wmi*

Set the Package Power Target total of CPU: PL1 on Intel, SPL on AMD. Shown on Intel+Nvidia or AMD+Nvidia based systems:

- min=5, max=250

/sys/devices/platform/<platform>/ppt_pl2_sppt

Defined on file *sysfs-platform-asus-wmi*

Set the Slow Package Power Tracking Limit of CPU: PL2 on Intel, SPPT, on AMD. Shown on Intel+Nvidia or AMD+Nvidia based systems:

- min=5, max=250

/sys/devices/platform/<platform>/ppt_platform_sppt

Defined on file *sysfs-platform-asus-wmi*

Set the platform SPPT limit. Shown on full AMD systems only:

- min=5, max=130

/sys/devices/platform/<platform>/throttle_thermal_policy

Defined on file *sysfs-platform-asus-wmi*

Throttle thermal policy mode:

- 0 - default,
- 1 - overboost,
- 2 - silent

/sys/devices/platform/<platform>/tokens/*

Defined on file *sysfs-platform-dell-smbios*

A read-only description of Dell platform tokens available on the machine.

Each token attribute is available as a pair of sysfs attributes readable by a process with CAP_SYS_ADMIN.

For example the token ID "5" would be available as the following attributes:

- 0005_location
- 0005_value

Tokens will vary from machine to machine, and only tokens available on that machine will be displayed.

/sys/devices/platform/<platform>/touchpad

Defined on file *sysfs-platform-asus-wmi*

Control the card touchpad. 1 means on, 0 means off.

/sys/devices/platform/<renesas_usb3's name>/role

Defined on file *sysfs-platform-renesas_usb3*

This file can be read and write. The file can show/change the drd mode of usb.

Write the following string to change the mode:

- "host" - switching mode from peripheral to host.
- "peripheral" - switching mode from host to peripheral.

Read the file, then it shows the following strings:

- "host" - The mode is host now.
- "peripheral" - The mode is peripheral now.

/sys/devices/platform/HISI04Bx:00/chipX/all_linked
/sys/devices/platform/HISI04Bx:00/chipX/linked_full_lane
/sys/devices/platform/HISI04Bx:00/chipX/crc_err_cnt

Defined on file *sysfs-devices-platform-kunpeng_hccs*

The /sys/devices/platform/HISI04Bx:00/chipX/ directory contains read-only attributes exposing some summarization information of all HCCS ports under a specified chip. The X in 'chipX' indicates the Xth chip on platform.

There are following attributes in this directory:

all_linked:	(RO)	if all enabled ports on this chip are linked (bool).
linked_full_lane:	(RO)	if all linked ports on this chip are full lane (bool).
crc_err_cnt:	(RO)	total CRC err count for all ports on this chip.

/sys/devices/platform/HISI04Bx:00/chipX/dieY/all_linked
/sys/devices/platform/HISI04Bx:00/chipX/dieY/linked_full_lane
/sys/devices/platform/HISI04Bx:00/chipX/dieY/crc_err_cnt

Defined on file *sysfs-devices-platform-kunpeng_hccs*

The /sys/devices/platform/HISI04Bx:00/chipX/dieY/ directory contains read-only attributes exposing some summarization information of all HCCS ports under a specified die. The Y in 'dieY' indicates the hardware id of the die on chip who has chip id X.

There are following attributes in this directory:

all_linked:	(RO)	if all enabled ports on this die are linked (bool).
linked_full_lane:	(RO)	if all linked ports on this die are full lane (bool).
crc_err_cnt:	(RO)	total CRC err count for all ports on this die.

/sys/devices/platform/HISI04Bx:00/chipX/dieY/hccsN/type
/sys/devices/platform/HISI04Bx:00/chipX/dieY/hccsN/lane_mode
/sys/devices/platform/HISI04Bx:00/chipX/dieY/hccsN/enable
/sys/devices/platform/HISI04Bx:00/chipX/dieY/hccsN/cur_lane_num
/sys/devices/platform/HISI04Bx:00/chipX/dieY/hccsN/link_fsm
/sys/devices/platform/HISI04Bx:00/chipX/dieY/hccsN/lane_mask
/sys/devices/platform/HISI04Bx:00/chipX/dieY/hccsN/crc_err_cnt

Defined on file *sysfs-devices-platform-kunpeng_hccs*

The /sys/devices/platform/HISI04Bx/chipX/dieX/hccsN/ directory contains read-only attributes exposing information about a HCCS port. The N value in 'hccsN' indicates this port id. The X in 'chipX' indicates the ID of the chip to which the HCCS port belongs. For example, X ranges from to 'n - 1' if the chip number on platform is n. The Y in 'dieY' indicates the hardware id of the die to which the hccs port belongs. Note: type, lane_mode and enable are fixed attributes on running platform.

The HCCS port have the following attributes:

type:	(RO)	port type (string), e.g. HCCS-v1 -> H32
lane_mode:	(RO)	the lane mode of this port (string), e.g. x8
enable:	(RO)	indicate if this port is enabled (bool).
cur_lane_num:	(RO)	current lane number of this port.
link_fsm:	(RO)	link finite state machine of this port.
lane_mask:	(RO)	current lane mask of this port, every bit indicates a lane.
crc_err_cnt:	(RO)	CRC err count on this port.

/sys/devices/platform/_UDC_/gadget/gadget-lunX/nofua

Defined on file *sysfs-devices-platform-_UDC_-gadget*

Show or set the reaction on the FUA (Force Unit Access) bit in the SCSI WRITE(10,12) commands when a gadget in USB Mass Storage mode.

Possible values are:

- 1 -> ignore the FUA flag
- 0 -> obey the FUA flag

/sys/devices/platform/_UDC_/gadget/suspended

Defined on file *sysfs-devices-platform-_UDC_-gadget*

Show the suspend state of an USB composite gadget.

- 1 -> suspended
- 0 -> resumed

(_UDC_ is the name of the USB Device Controller driver)

/sys/devices/platform/amba_pl/<dev>/dis_block_break

Defined on file *sysfs-driver-xilinx-tmr-manager*

Write any value to it, This control file enables the break signal. This file is write only. Example: # echo <any value> > /sys/devices/platform/amba_pl/44a10000.tmr_manager/dis_block_break

/sys/devices/platform/amba_pl/<dev>/errcnt

Defined on file *sysfs-driver-xilinx-tmr-manager*

This control file provides the fault detection count. This file cannot be written. Example: # cat /sys/devices/platform/amba_pl/44a10000.tmr_manager/errcnt 1

/sys/devices/platform/asus_laptop/bluetooth

Defined on file *sysfs-platform-asus-laptop*

Control the bluetooth device. 1 means on, 0 means off. This may control the led, the device or both.

Users: Lapsus

/sys/devices/platform/asus_laptop/display

Defined on file *sysfs-platform-asus-laptop*

This file allows display switching. The value is composed by 4 bits and defined as follow:

```
4321
|||`- LCD
||`-- CRT
|`--- TV
`---- DVI
```

Ex:

- 0 (0000b) means no display
- 3 (0011b) CRT+LCD.

/sys/devices/platform/asus_laptop/gps

Defined on file *sysfs-platform-asus-laptop*

Control the gps device. 1 means on, 0 means off.

Users: Lapsus

/sys/devices/platform/asus_laptop/ledd

Defined on file *sysfs-platform-asus-laptop*

Some models like the W1N have a LED display that can be used to display several items of information. To control the LED display, use the following:

```
echo 0x0T000DDD > /sys/devices/platform/asus_laptop/
```

where T control the 3 letters display, and DDD the 3 digits display. The DDD table can be found in *Asus Laptop Extras*

/sys/devices/platform/asus_laptop/wimax

Defined on file *sysfs-platform-asus-laptop*

Control the wimax device. 1 means on, 0 means off.

/sys/devices/platform/asus_laptop/wlan

Defined on file *sysfs-platform-asus-laptop*

Control the wlan device. 1 means on, 0 means off. This may control the led, the device or both.

Users: Lapsus

/sys/devices/platform/asus_laptop/wwan

Defined on file *sysfs-platform-asus-laptop*

Control the wwan (3G) device. 1 means on, 0 means off.

/sys/devices/platform/at91_can/net/<iface>/mb0_id

Defined on file *sysfs-platform-at91*

Value representing the can_id of mailbox 0.

Default: 0x7ff (standard frame)

Due to a chip bug (errata 50.2.6.3 & 50.3.5.3 in "AT91SAM9263 Preliminary 6249H-ATARM-27-Jul-09") the contents of mailbox 0 may be send under certain conditions (even if disabled or in rx mode).

The workaround in the errata suggests not to use the mailbox and load it with an unused identifier.

In order to use an extended can_id add the CAN_EFF_FLAG (0x80000000U) to the can_id. Example:

- standard id 0x7ff:

```
echo 0x7ff      > /sys/class/net/can0/mb0_id
```

- extended id 0x1fffffff:

```
echo 0x9fffffff > /sys/class/net/can0/mb0_id
```

/sys/devices/platform/dmi-ipmi-ssif.*/hosed
/sys/devices/platform/dmi-ipmi-ssif.*/alerts
/sys/devices/platform/dmi-ipmi-ssif.*/sent_messages
/sys/devices/platform/dmi-ipmi-ssif.*/sent_messages_parts
/sys/devices/platform/dmi-ipmi-ssif.*/received_messages
/sys/devices/platform/dmi-ipmi-ssif.*/received_message_parts
/sys/devices/platform/dmi-ipmi-ssif.*/events
/sys/devices/platform/dmi-ipmi-ssif.*/watchdog_pretimeouts
/sys/devices/platform/dmi-ipmi-ssif.*/flag_fetches
/sys/devices/platform/dmi-ipmi-ssif.*/send_retries
/sys/devices/platform/dmi-ipmi-ssif.*/receive_retries
/sys/devices/platform/dmi-ipmi-ssif.*/send_errors
/sys/devices/platform/dmi-ipmi-ssif.*/receive_errors

Defined on file *sysfs-devices-platform-ipmi*

hosed	(RO) Number of times the hardware didn't follow the state machine.
alerts	(RO) Number of alerts received.
sent_messages	(RO) Number of total messages sent.
sent_message_parts	(RO) Number of message parts sent. Messages may be broken into parts if they are long.
received_messages	(RO) Number of message responses received.
received_message_parts	(RO) Number of message fragments received.
events	(RO) Number of received events.
watchdog_pretimeouts	(RO) Number of watchdog pretimeouts.
flag_fetches	(RO) Number of times a flag fetch was requested.
send_retries	(RO) Number of time a message was retried.
receive_retries	(RO) Number of times the receive of a message was retried.
send_errors	(RO) Number of times the send of a message failed.
receive_errors	(RO) Number of errors in receiving messages.

/sys/devices/platform/dmi-ipmi-ssif.*/type

Defined on file *sysfs-devices-platform-ipmi*

(RO) Shows the IMPI device interface type - ”ssif” here.

/sys/devices/platform/docg3/f[0-3]_dps[01]_is_keylocked

Defined on file *sysfs-devices-platform-docg3*

Show whether the floor (0 to 4), protection area (0 or 1) is keylocked. Each docg3 chip (or floor) has 2 protection areas, which can cover any part of it, block aligned, called DPS. The protection has information embedded whether it blocks reads, writes or both. The result is:

- 0 -> the DPS is not keylocked
- 1 -> the DPS is keylocked

Users: None identified so far.

/sys/devices/platform/docg3/f[0-3]_dps[01]_protection_key

Defined on file *sysfs-devices-platform-docg3*

Enter the protection key for the floor (0 to 4), protection area (0 or 1). Each docg3 chip (or floor) has 2 protection areas, which can cover any part of it, block aligned, called DPS. The protection has information embedded whether it blocks reads, writes or both. The protection key is a string of 8 bytes (value 0-255). Entering the correct value toggle the lock, and can be observed through f[0-3]_dps[01]_is_keylocked. Possible values are:

- 8 bytes

Typical values are:

- "00000000"
- "12345678"

Users: None identified so far.

/sys/devices/platform/dock.<N>/docked

Defined on file *sysfs-devices-platform-dock*

(RO) Value 1 or 0 indicates whether the software believes the laptop is docked in a docking station.

/sys/devices/platform/dock.<N>/flags

Defined on file *sysfs-devices-platform-dock*

(RO) Show dock station flags, useful for checking if undock request has been made by the user (from the immediate_undock option).

/sys/devices/platform/dock.<N>/type

Defined on file *sysfs-devices-platform-dock*

(RO) Display the dock station type- dock_station, ata_bay or battery_bay.

/sys/devices/platform/dock.<N>/uid

Defined on file *sysfs-devices-platform-dock*

(RO) Displays the docking station the laptop is docked to.

/sys/devices/platform/dock.<N>/undock

Defined on file *sysfs-devices-platform-dock*

(WO) Writing to this file causes the software to initiate an undock request to the firmware.

/sys/devices/platform/eeepc/available_cpufv

Defined on file *sysfs-platform-eeepc-laptop*

List available cpufv modes.

/sys/devices/platform/eeepc/camera

Defined on file *sysfs-platform-eeepc-laptop*

Control the camera. 1 means on, 0 means off.

/sys/devices/platform/eeepc/cardr

Defined on file *sysfs-platform-eeepc-laptop*

Control the card reader. 1 means on, 0 means off.

/sys/devices/platform/eeepc/cpufv

Defined on file *sysfs-platform-eeepc-laptop*

Change CPU clock configuration. On the Eee PC 1000H there are three available clock configuration:

- 0 -> Super Performance Mode
- 1 -> High Performance Mode
- 2 -> Power Saving Mode

On Eee PC 701 there is only 2 available clock configurations. Available configuration are listed in available_cpufv file. Reading this file will show the raw hexadecimal value which is defined as follow:

```
| 8 bit | 8 bit |
    |       `---- Current mode
    `------------ Availables modes
```

For example, 0x301 means: mode 1 selected, 3 available modes.

/sys/devices/platform/eeepc/disp

Defined on file *sysfs-platform-eeepc-laptop*

This file allows display switching.

- 1 = LCD
- 2 = CRT
- 3 = LCD+CRT

If you run X11, you should use xrandr instead.

/sys/devices/platform/hidma-*/chid

Defined on file *sysfs-platform-hidma*

Contains the ID of the channel within the HIDMA instance. It is used to associate a given HIDMA channel with the priority and weight calls in the management interface.

/sys/devices/platform/hidma-mgmt*/chanops/chan*/priority

Defined on file *sysfs-platform-hidma-mgmt*

Contains either 0 or 1 and indicates if the DMA channel is a low priority (0) or high priority (1) channel.

/sys/devices/platform/hidma-mgmt*/chanops/chan*/weight

Defined on file *sysfs-platform-hidma-mgmt*

Contains 0..15 and indicates the weight of the channel among equal priority channels during round robin scheduling.

/sys/devices/platform/hidma-mgmt*/chreset_timeout_cycles

Defined on file *sysfs-platform-hidma-mgmt*

Contains the platform specific cycle value to wait after a reset command is issued. If the value is chosen too short, then the HW will issue a reset failure interrupt. The value is platform specific and should not be changed without consultance.

/sys/devices/platform/hidma-mgmt*/dma_channels

Defined on file *sysfs-platform-hidma-mgmt*

Contains the number of dma channels supported by one instance of HIDMA hardware. The value may change from chip to chip.

/sys/devices/platform/hidma-mgmt*/hw_version_major

Defined on file *sysfs-platform-hidma-mgmt*

Version number major for the hardware.

/sys/devices/platform/hidma-mgmt*/hw_version_minor

Defined on file *sysfs-platform-hidma-mgmt*

Version number minor for the hardware.

/sys/devices/platform/hidma-mgmt*/max_rd_xactions

Defined on file *sysfs-platform-hidma-mgmt*

Contains a value between 0 and 31. Maximum number of read transactions that can be issued back to back. Choosing a higher number gives better performance but can also cause performance reduction to other peripherals sharing the same bus.

/sys/devices/platform/hidma-mgmt*/max_read_request

Defined on file *sysfs-platform-hidma-mgmt*

Size of each read request. The value needs to be a power of two and can be between 128 and 1024.

/sys/devices/platform/hidma-mgmt*/max_wr_xactions

Defined on file *sysfs-platform-hidma-mgmt*

Contains a value between 0 and 31. Maximum number of write transactions that can be issued back to back. Choosing a higher number gives better performance but can also cause performance reduction to other peripherals sharing the same bus.

/sys/devices/platform/hidma-mgmt*/max_write_request

Defined on file *sysfs-platform-hidma-mgmt*

Size of each write request. The value needs to be a power of two and can be between 128 and 1024.

/sys/devices/platform/i8042/.../draghys

Defined on file *sysfs-devices-platform-trackpoint*

(RW) The drag hysteresis controls how hard it is to drag with z-axis pressed.

/sys/devices/platform/i8042/.../drift_time

Defined on file *sysfs-devices-platform-trackpoint*

(RW) This parameter controls the period of time to test for a 'hands off' condition (i.e. when no force is applied) before a drift (noise) calibration occurs.

IBM Trackpoints have a feature to compensate for drift by recalibrating themselves periodically. By default, if for 0.5 seconds there is no change in position, it's used as the new zero. This duration is too low. Often, the calibration happens when the trackpoint is in fact being used.

/sys/devices/platform/i8042/.../ext_dev

Defined on file *sysfs-devices-platform-trackpoint*

(RW) Disable (0) or enable (1) external pointing device.

/sys/devices/platform/i8042/.../intertia

Defined on file *sysfs-devices-platform-trackpoint*

(RW) Negative inertia factor. High values cause the cursor to snap backward when the trackpoint is released.

/sys/devices/platform/i8042/.../jenks

Defined on file *sysfs-devices-platform-trackpoint*

(RW) Minimum curvature in degrees required to generate a double click without a release.

/sys/devices/platform/i8042/.../mindrag

Defined on file *sysfs-devices-platform-trackpoint*

(RW) Minimum amount of force needed to trigger dragging.

/sys/devices/platform/i8042/.../press_to_select

Defined on file *sysfs-devices-platform-trackpoint*

(RW) Writing a value of 1 to this file will enable the Press to Select functions like tapping the control stick to simulate a left click, and writing 0 will disable it.

/sys/devices/platform/i8042/.../reach

Defined on file *sysfs-devices-platform-trackpoint*

(RW) Backup range for z-axis press.

/sys/devices/platform/i8042/.../sensitivity

Defined on file *sysfs-devices-platform-trackpoint*

(RW) Trackpoint sensitivity.

/sys/devices/platform/i8042/.../skipback

Defined on file *sysfs-devices-platform-trackpoint*

(RW) When the skipback bit is set, backup cursor movement during releases from drags will be suppressed. The default value for this bit is 0.

/sys/devices/platform/i8042/.../speed

Defined on file *sysfs-devices-platform-trackpoint*

(RW) Speed of the trackpoint cursor.

/sys/devices/platform/i8042/.../thresh

Defined on file *sysfs-devices-platform-trackpoint*

(RW) Minimum value for z-axis force required to trigger a press or release, relative to the running average.

/sys/devices/platform/i8042/.../upthresh

Defined on file *sysfs-devices-platform-trackpoint*

(RW) The offset from the running average required to generate a select (click) on z-axis on release.

/sys/devices/platform/i8042/.../ztime

Defined on file *sysfs-devices-platform-trackpoint*

(RW) This attribute determines how sharp a press has to be in order to be recognized.

/sys/devices/platform/ipmi_bmc.*/additional_device_support

Defined on file *sysfs-devices-platform-ipmi*

(RO) Lists the IPMI 'logical device' commands and functions that the controller supports that are in addition to the mandatory IPM and Application commands.

/sys/devices/platform/ipmi_bmc.*/aux_firmware_revision

Defined on file *sysfs-devices-platform-ipmi*

(RO) Holds additional information about the firmware revision, such as boot block or internal data structure version numbers. The meanings of the numbers are specific to the vendor identified by Manufacturer ID.

/sys/devices/platform/ipmi_bmc.*/device_id

Defined on file *sysfs-devices-platform-ipmi*

(RO) Device id is specified by the manufacturer identified by the Manufacturer ID field. This field allows controller specific software to identify the unique application command, OEM fields, and functionality that are provided by the controller

/sys/devices/platform/ipmi_bmc.*/firmware_revision

Defined on file *sysfs-devices-platform-ipmi*

(RO) The major and minor revision of the firmware.

/sys/devices/platform/ipmi_bmc.*/guid

Defined on file *sysfs-devices-platform-ipmi*

(RO) A GUID (Globally Unique ID), also referred to as a UUID (Universally Unique Identifier), for the management controller, as described in section 20.8 'Get Device GUID Command' of the IPMI specification v2.0.

/sys/devices/platform/ipmi_bmc.*/ipmi_version

Defined on file *sysfs-devices-platform-ipmi*

(RO) Displays the IPMI Command Specification Version.

/sys/devices/platform/ipmi_bmc.*/manufacturer_id

Defined on file *sysfs-devices-platform-ipmi*

(RO) Identifies the manufacturer responsible for the specification of functionality of the vendor (OEM)-specific commands, codes, and interfaces used in the controller.

/sys/devices/platform/ipmi_bmc.*/product_id

Defined on file *sysfs-devices-platform-ipmi*

(RO) Displays a number that identifies a particular system, module, add-in card, or board set. The number is specified according to the manufacturer given by Manufacturer ID.

For detailed definitions of the above attributes, refer to section 20.1 'Get Device ID Command' of the IPMI specification v2.0.

/sys/devices/platform/ipmi_bmc.*/provides_device_sdrs

Defined on file *sysfs-devices-platform-ipmi*

(RO) Indicates whether device provides device sensor data records (1) or not (0).

/sys/devices/platform/ipmi_bmc.*/revision

Defined on file *sysfs-devices-platform-ipmi*

(RO) Device revision. Useful for identifying if significant hardware changes have been made to the implementation of the management controller.

/sys/devices/platform/ipmi_si.*/idles
/sys/devices/platform/ipmi_si.*/watchdog_pretimeouts
/sys/devices/platform/ipmi_si.*/complete_transactions
/sys/devices/platform/ipmi_si.*/events
/sys/devices/platform/ipmi_si.*/interrupts
/sys/devices/platform/ipmi_si.*/hosed_count
/sys/devices/platform/ipmi_si.*/long_timeouts
/sys/devices/platform/ipmi_si.*/flag_fetches
/sys/devices/platform/ipmi_si.*/attentions
/sys/devices/platform/ipmi_si.*/incoming_messages
/sys/devices/platform/ipmi_si.*/short_timeouts

Defined on file *sysfs-devices-platform-ipmi*

idles	(RO) Number of times the interface was idle while being polled.
watchdog_pretimeouts	(RO) Number of watchdog pretimeouts.
complete_transactions	(RO) Number of completed messages.
events	(RO) Number of IPMI events received from the hardware.
interrupts	(RO) Number of interrupts the driver handled.
hosed_count	(RO) Number of times the hardware didn't follow the state machine.
long_timeouts	(RO) Number of times the driver requested a timer while nothing was in progress.
flag_fetches	(RO) Number of times the driver requested flags from the hardware.
attentions	(RO) Number of time the driver got an ATTN from the hardware.
incoming_messages	(RO) Number of asynchronous messages received.
short_timeouts	(RO) Number of times the driver requested a timer while an operation was in progress.

/sys/devices/platform/ipmi_si.*/interrupts_enabled

Defined on file *sysfs-devices-platform-ipmi*

(RO) Indicates whether interrupts are enabled or not. The driver disables interrupts when it gets into a situation where it cannot handle messages due to lack of memory. Once that situation clears up, it will re-enable interrupts.

/sys/devices/platform/ipmi_si.*/params

Defined on file *sysfs-devices-platform-ipmi*

[to be documented]

/sys/devices/platform/ipmi_si.*/type

Defined on file *sysfs-devices-platform-ipmi*

(RO) The device interface for IPMI ”kcs”, ”smic”, ”bt” or ”invalid”

/sys/devices/platform/kim/baud_rate

Defined on file *sysfs-platform-kim*

The maximum reliable baud-rate the host can support. Different platforms tend to have different high-speed UART configurations, so the baud-rate needs to be set locally and also sent across to the WL128x via a HCI-VS command. The entry is read and made use by the user-space daemon when the ldisc install is requested.

/sys/devices/platform/kim/dev_name

Defined on file *sysfs-platform-kim*

Name of the UART device at which the WL128x chip is connected. example: ”/dev/ttyS0”.

The device name flows down to architecture specific board initialization file from the ATAGS bootloader firmware. The name exposed is read from the user-space daemon and opens the device when install is requested.

/sys/devices/platform/kim/flow_cntrl

Defined on file *sysfs-platform-kim*

The WL128x makes use of flow control mechanism, and this entry most often should be 1, the host's UART is required to have the capability of flow-control, or else this entry can be made use of for exceptions.

/sys/devices/platform/kim/install

Defined on file *sysfs-platform-kim*

When one of the protocols Bluetooth, FM or GPS wants to make use of the shared UART transport, it registers to the shared transport driver, which will signal the user-space for opening, configuring baud and install line discipline via this sysfs entry. This entry would be polled upon by the user-space daemon managing the UART, and is notified about the change by the sysfs_notify. The value would be '1' when UART needs to be opened/ldisc installed, and would be '0' when UART is no more required and needs to be closed.

/sys/devices/platform/lg-laptop/battery_care_limit

Defined on file *sysfs-platform-lg-laptop*

Deprecated use /sys/class/power_supply/CMB0/charge_control_end_threshold Maximal battery charge level. Accepted values are 80 or 100.

/sys/devices/platform/lg-laptop/fan_mode

Defined on file *sysfs-platform-lg-laptop*

Control fan mode. 1 for performance mode, 0 for silent mode.

/sys/devices/platform/lg-laptop/fn_lock

Defined on file *sysfs-platform-lg-laptop*

Control FN lock mode. 1 means on, 0 means off.

/sys/devices/platform/lg-laptop/reader_mode

Defined on file *sysfs-platform-lg-laptop*

Control reader mode. 1 means on, 0 means off.

/sys/devices/platform/lg-laptop/usb_charge

Defined on file *sysfs-platform-lg-laptop*

Control USB port charging when device is turned off. 1 means on, 0 means off.

/sys/devices/platform/msi-laptop-pf/auto_brightness

Defined on file *sysfs-platform-msi-laptop*

Enable automatic brightness control: contains either 0 or 1. If set to 1 the hardware adjusts the screen brightness automatically when the power cord is plugged/unplugged.

/sys/devices/platform/msi-laptop-pf/auto_fan

Defined on file *sysfs-platform-msi-laptop*

Contains either 0 or 1 and indicates if fan speed is controlled automatically (1) or fan runs at maximal speed (0). Can be toggled in software.

/sys/devices/platform/msi-laptop-pf/bluetooth

Defined on file *sysfs-platform-msi-laptop*

Bluetooth subsystem enabled: contains either 0 or 1. Please note that this file is constantly 0 if no Bluetooth hardware is available.

/sys/devices/platform/msi-laptop-pf/eco_mode

Defined on file *sysfs-platform-msi-laptop*

Contains either 0 or 1 and indicates if ECO mode is turned on. In ECO mode power LED is green and userspace should do some powersaving actions. ECO mode is available only on battery power. ECO mode can only be toggled by pressing Fn+F10.

/sys/devices/platform/msi-laptop-pf/lcd_level

Defined on file *sysfs-platform-msi-laptop*

Screen brightness: contains a single integer in the range 0..8.

/sys/devices/platform/msi-laptop-pf/touchpad

Defined on file *sysfs-platform-msi-laptop*

Contains either 0 or 1 and indicates if touchpad is turned on. Touchpad state can only be toggled by pressing Fn+F3.

/sys/devices/platform/msi-laptop-pf/turbo_cooldown

Defined on file *sysfs-platform-msi-laptop*

Contains value in range 0..3:

- 0 -> Turbo mode is off
- 1 -> Turbo mode is on, cannot be turned off yet

- 2 -> Turbo mode is off, cannot be turned on yet
- 3 -> Turbo mode is on

/sys/devices/platform/msi-laptop-pf/turbo_mode

Defined on file *sysfs-platform-msi-laptop*

Contains either 0 or 1 and indicates if turbo mode is turned on. In turbo mode power LED is orange and processor is overclocked. Turbo mode is available only if charging. It is only possible to toggle turbo mode state by pressing Fn+F10, and there is a few seconds cooldown between subsequent toggles. If user presses Fn+F10 too frequent, turbo mode state is not changed.

/sys/devices/platform/msi-laptop-pf/wlan

Defined on file *sysfs-platform-msi-laptop*

WLAN subsystem enabled: contains either 0 or 1.

/sys/devices/platform/samsung/battery_life_extender

Defined on file *sysfs-driver-samsung-laptop*

Max battery charge level can be modified, battery cycle life can be extended by reducing the max battery charge level.

- 0 means normal battery mode (100% charge)
- 1 means battery life extender mode (80% charge)

/sys/devices/platform/samsung/lid_handling

Defined on file *sysfs-driver-samsung-laptop*

Some Samsung laptops handle lid closing quicker and only handle lid opening with this mode enabled. 1 means enabled, 0 means disabled.

/sys/devices/platform/samsung/performance_level

Defined on file *sysfs-driver-samsung-laptop*

Some Samsung laptops have different ”performance levels” that can be modified by a function key, and by this sysfs file. These values don't always make a whole lot of sense, but some users like to modify them to keep their fans quiet at all costs. Reading from this file will show the current performance level. Writing to the file can change this value.

Valid options:

- ”silent”
- ”normal”
- ”overclock”

Note that not all laptops support all of these options. Specifically, not all support the "overclock" option, and it's still unknown if this value even changes anything, other than making the user feel a bit better.

/sys/devices/platform/samsung/usb_charge

Defined on file *sysfs-driver-samsung-laptop*

Use your USB ports to charge devices, even when your laptop is powered off. 1 means enabled, 0 means disabled.

/sys/devices/platform/sh_mobile_lcdc_fb.[0-3]/graphics/fb[0-9]/ovl_alpha

Defined on file *sysfs-devices-platform-sh_mobile_lcdc_fb*

This file is only available on fb[0-9] devices corresponding to overlay planes.

Stores the alpha blending value for the overlay. Values range from 0 (transparent) to 255 (opaque). The value is ignored if the mode is not set to Alpha Blending.

/sys/devices/platform/sh_mobile_lcdc_fb.[0-3]/graphics/fb[0-9]/ovl_mode

Defined on file *sysfs-devices-platform-sh_mobile_lcdc_fb*

This file is only available on fb[0-9] devices corresponding to overlay planes.

Selects the composition mode for the overlay. Possible values are:

- 0 - Alpha Blending
- 1 - ROP3

/sys/devices/platform/sh_mobile_lcdc_fb.[0-3]/graphics/fb[0-9]/ovl_position

Defined on file *sysfs-devices-platform-sh_mobile_lcdc_fb*

This file is only available on fb[0-9] devices corresponding to overlay planes.

Stores the x,y overlay position on the display in pixels. The position format is *[0-9]+,[0-9]+*.

/sys/devices/platform/sh_mobile_lcdc_fb.[0-3]/graphics/fb[0-9]/ovl_rop3

Defined on file *sysfs-devices-platform-sh_mobile_lcdc_fb*

This file is only available on fb[0-9] devices corresponding to overlay planes.

Stores the raster operation (ROP3) for the overlay. Values range from 0 to 255. The value is ignored if the mode is not set to ROP3.

/sys/devices/platform/silicom-platform/efuse_status

Defined on file *sysfs-platform-silicom*

This file is read only. It returns the current OTP status:

0 - not programmed. 1 - programmed.

/sys/devices/platform/silicom-platform/power_cycle

Defined on file *sysfs-platform-silicom*

This file allow user to power cycle the platform. Default value is 0; when set to 1, it powers down the platform, waits 5 seconds, then powers on the device. It returns to default value after power cycle.

0 - default value.

/sys/devices/platform/silicom-platform/uc_version

Defined on file *sysfs-platform-silicom*

This file allows to read microcontroller firmware version of current platform.

/sys/devices/platform/soc@X/XXXXXXX.ipa/

Defined on file *sysfs-devices-platform-soc-ipa*

The /sys/devices/platform/soc@X/XXXXXXX.ipa/ directory contains read-only attributes exposing information about an IPA device. The X values could vary, but are typically "soc@0/1e40000.ipa".

/sys/devices/platform/stratix10-rsu.0/current_image

Defined on file *sysfs-devices-platform-stratix10-rsu*

(RO) the address in flash of currently running image.

/sys/devices/platform/stratix10-rsu.0/dcmf0

Defined on file *sysfs-devices-platform-stratix10-rsu*

(RO) Decision firmware copy 0 version information.

/sys/devices/platform/stratix10-rsu.0/dcmf1

Defined on file *sysfs-devices-platform-stratix10-rsu*

(RO) Decision firmware copy 1 version information.

/sys/devices/platform/stratix10-rsu.0/dcmf2

Defined on file *sysfs-devices-platform-stratix10-rsu*

(RO) Decision firmware copy 2 version information.

/sys/devices/platform/stratix10-rsu.0/dcmf3

Defined on file *sysfs-devices-platform-stratix10-rsu*

(RO) Decision firmware copy 3 version information.

/sys/devices/platform/stratix10-rsu.0/error_details

Defined on file *sysfs-devices-platform-stratix10-rsu*

(RO) error code.

/sys/devices/platform/stratix10-rsu.0/error_location

Defined on file *sysfs-devices-platform-stratix10-rsu*

(RO) the error offset inside the image that failed.

/sys/devices/platform/stratix10-rsu.0/fail_image

Defined on file *sysfs-devices-platform-stratix10-rsu*

(RO) the address in flash of failed image.

/sys/devices/platform/stratix10-rsu.0/max_retry

Defined on file *sysfs-devices-platform-stratix10-rsu*

(RO) max retry parameter is stored in the firmware decision IO section, as a byte located at offset 0x18c.

/sys/devices/platform/stratix10-rsu.0/notify

Defined on file *sysfs-devices-platform-stratix10-rsu*

(WO) client to notify firmware with different actions.

b[15:0]
inform firmware the current software execution stage.

0	the first stage bootloader didn't run or didn't reach the point of launching second stage bootloader.
1	failed in second bootloader or didn't get to the point of launching the operating system.
2	both first and second stage bootloader ran and the operating system launch was attempted.

b[16]

1	firmware to reset current image retry counter.
0	no action.

b[17]

1	firmware to clear RSU log
0	no action.

b[18]
this is negative logic

1	no action
0	firmware record the notify code defined in b[15:0].

/sys/devices/platform/stratix10-rsu.0/reboot_image

Defined on file *sysfs-devices-platform-stratix10-rsu*

(WO) the address in flash of image to be loaded on next reboot command.

/sys/devices/platform/stratix10-rsu.0/retry_counter

Defined on file *sysfs-devices-platform-stratix10-rsu*

(RO) the current image's retry counter, which is used by user to know how many times the images is still allowed to reload itself before giving up and starting RSU fail-over flow.

/sys/devices/platform/stratix10-rsu.0/state

Defined on file *sysfs-devices-platform-stratix10-rsu*

(RO) the state of RSU system. The state field has two parts: major error code in upper 16 bits and minor error code in lower 16 bits.

b[15:0]
Currently used only when major error is 0xF006 (CPU watchdog timeout), in which case the minor error code is the value reported by CPU to firmware through the RSU notify command before the watchdog timeout occurs.

b[31:16]
0xF001 bitstream error 0xF002 hardware access failure 0xF003 bitstream corruption 0xF004 internal error 0xF005 device error 0xF006 CPU watchdog timeout 0xF007 internal unknown error

/sys/devices/platform/stratix10-rsu.0/version

Defined on file *sysfs-devices-platform-stratix10-rsu*

(RO) the version number of RSU firmware. 19.3 or late version includes information about the firmware which reported the error.

pre 19.3:

b[31:0]
0x0 version number

19.3 or late:

b[15:0]
0x1 version number

b[31:16]
0x0 no error 0x0DCF Decision CMF error 0x0ACF Application CMF error

/sys/devices/platform/ts5500/adc

Defined on file *sysfs-platform-ts5500*

Indicates the presence of an A/D Converter. If it is present, it will display "1", otherwise "0".

/sys/devices/platform/ts5500/ereset

Defined on file *sysfs-platform-ts5500*

Indicates the presence of an external reset. If it is present, it will display "1", otherwise "0".

/sys/devices/platform/ts5500/id

Defined on file *sysfs-platform-ts5500*

Product ID of the TS board. TS-5500 ID is 0x60.

/sys/devices/platform/ts5500/jumpers

Defined on file *sysfs-platform-ts5500*

Bitfield showing the jumpers' state. If a jumper is present, the corresponding bit is set. For instance, 0x0e means jumpers 2, 3 and 4 are set.

/sys/devices/platform/ts5500/name

Defined on file *sysfs-platform-ts5500*

Model name of the TS board, e.g. "TS-5500".

/sys/devices/platform/ts5500/rs485

Defined on file *sysfs-platform-ts5500*

Indicates the presence of the RS485 option. If it is present, it will display "1", otherwise "0".

/sys/devices/platform/ts5500/sram

Defined on file *sysfs-platform-ts5500*

Indicates the presence of the SRAM option. If it is present, it will display "1", otherwise "0".

/sys/devices/platform/usbip-vudc.%d/dev_desc

Defined on file *sysfs-platform-usbip-vudc*

This file allows to read device descriptor of gadget driver which is currently bound to this controller. It is possible to read this file only if gadget driver is bound, otherwise error is returned.

/sys/devices/platform/usbip-vudc.%d/usbip_sockfd

Defined on file *sysfs-platform-usbip-vudc*

This file allows to export usb device to connection peer. It is done by writing to this file socket fd (as a string for example "8") associated with a connection to remote peer who would like to use this device. It is possible to close the connection by writing -1 instead of socked fd.

/sys/devices/platform/usbip-vudc.%d/usbip_status

Defined on file *sysfs-platform-usbip-vudc*

Current status of the device. Allowed values:

1	Device is available and can be exported
2	Device is currently exported
3	Fatal error occurred during communication with peer

/sys/devices/socX

Defined on file *sysfs-devices-soc*

The /sys/devices/ directory contains a sub-directory for each System-on-Chip (SoC) device on a running platform. Information regarding each SoC can be obtained by reading sysfs files. This functionality is only available if implemented by the platform.

The directory created for each SoC will also house information about devices which are commonly contained in /sys/devices/platform. It has been agreed that if an SoC device exists, its supported devices would be better suited to appear as children of that SoC.

/sys/devices/socX/family

Defined on file *sysfs-devices-soc*

Read-only attribute common to all SoCs. Contains SoC family name (e.g. DB8500).

On many of ARM based silicon with SMCCC v1.2+ compliant firmware this will contain the JEDEC JEP106 manufacturer's identification code. The format is "jep106:XXYY" where XX is identity code and YY is continuation code.

This manufacturer's identification code is defined by one or more eight (8) bit fields, each consisting of seven (7) data bits plus one (1) odd parity bit. It is a single field, limiting the possible number of vendors to 126. To expand the maximum number of identification codes, a continuation scheme has been defined.

The specified mechanism is that an identity code of 0x7F represents the "continuation code" and implies the presence of an additional identity code field, and this mechanism may be extended to multiple continuation codes followed by the manufacturer's identity code.

For example, ARM has identity code 0x7F 0x7F 0x7F 0x7F 0x3B, which is code 0x3B on the fifth 'page'. This is shortened as JEP106 identity code of 0x3B and a continuation code of 0x4 to represent the four continuation codes preceding the identity code.

/sys/devices/socX/machine

Defined on file *sysfs-devices-soc*

Read-only attribute common to all SoCs. Contains the SoC machine name (e.g. Ux500).

/sys/devices/socX/process

Defined on file *sysfs-devices-soc*

Read-only attribute supported ST-Ericsson's silicon. Contains the the process by which the silicon chip was manufactured.

/sys/devices/socX/revision

Defined on file *sysfs-devices-soc*

Read-only attribute supported by most SoCs. Contains the SoC's manufacturing revision number.

/sys/devices/socX/serial_number

Defined on file *sysfs-devices-soc*

Read-only attribute supported by most SoCs. Contains the SoC's serial number, if available.

/sys/devices/socX/soc_id

Defined on file *sysfs-devices-soc*

Read-only attribute supported by most SoCs. In the case of ST-Ericsson's chips this contains the SoC serial number.

On many of ARM based silicon with SMCCC v1.2+ compliant firmware this will contain the SOC ID appended to the family attribute to ensure there is no conflict in this namespace across various vendors. The format is "jep106:XXYY:ZZZZ" where XX is identity code, YY is continuation code and ZZZZ is the SOC ID.

/sys/devices/system/cpu/

Defined on file *sysfs-devices-system-cpu*

A collection of both global and individual CPU attributes

Individual CPU attributes are contained in subdirectories named by the kernel's logical CPU number, e.g.:

/sys/devices/system/cpu/cpuX/

/sys/devices/system/cpu/aarch32_el0

Defined on file *sysfs-devices-system-cpu*

Identifies the subset of CPUs in the system that can execute AArch32 (32-bit ARM) applications. If present, the same format as /sys/devices/system/cpu/{offline,online,possible,present} is used. If absent, then all or none of the CPUs can execute AArch32 applications and execve() will behave accordingly.

/sys/devices/system/cpu/cpu*/cache/index*/<set_of_attributes_mentioned_below>

Defined on file *sysfs-devices-system-cpu*

Parameters for the CPU cache attributes

allocation_policy:

- **WriteAllocate:**
  allocate a memory location to a cache line on a cache miss because of a write
- **ReadAllocate:**
  allocate a memory location to a cache line on a cache miss because of a read
- **ReadWriteAllocate:**
  both writeallocate and readallocate

coherency_line_size:
the minimum amount of data in bytes that gets transferred from memory to cache

level:
the cache hierarchy in the multi-level cache configuration

number_of_sets:
total number of sets in the cache, a set is a collection of cache lines with the same cache index

physical_line_partition:
number of physical cache line per cache tag

shared_cpu_list:
the list of logical cpus sharing the cache

shared_cpu_map:
logical cpu mask containing the list of cpus sharing the cache

size:
the total cache size in kB

type:

- Instruction: cache that only holds instructions
- Data: cache that only caches data

- Unified: cache that holds both data and instructions

ways_of_associativity:
degree of freedom in placing a particular block of memory in the cache

write_policy:

- **WriteThrough:**
  data is written to both the cache line and to the block in the lower-level memory
- **WriteBack:**
  data is written only to the cache line and the modified cache line is written to main memory only when it is replaced

/sys/devices/system/cpu/cpu*/cache/index*/id

Defined on file *sysfs-devices-system-cpu*

Cache id

The id provides a unique number for a specific instance of a cache of a particular type. E.g. there may be a level 3 unified cache on each socket in a server and we may assign them ids 0, 1, 2, ...

Note that id value can be non-contiguous. E.g. level 1 caches typically exist per core, but there may not be a power of two cores on a socket, so these caches may be numbered 0, 1, 2, 3, 4, 5, 8, 9, 10, ...

/sys/devices/system/cpu/cpu*/cache/index3/cache_disable_{0,1}

Defined on file *sysfs-devices-system-cpu*

Disable L3 cache indices

These files exist in every CPU's cache/index3 directory. Each cache_disable_{0,1} file corresponds to one disable slot which can be used to disable a cache index. Reading from these files on a processor with this functionality will return the currently disabled index for that node. There is one L3 structure per node, or per internal node on MCM machines. Writing a valid index to one of these files will cause the specified cache index to be disabled.

All AMD processors with L3 caches provide this functionality. For details, see BKDGs at https://www.amd.com/en/support/tech-docs?keyword=bios+kernel

/sys/devices/system/cpu/cpuX/cpu_capacity

Defined on file *sysfs-devices-system-cpu*

information about CPUs heterogeneity.

cpu_capacity: capacity of cpuX.

/sys/devices/system/cpu/cpuX/cpufreq/*

Defined on file *sysfs-devices-system-cpu*

Discover and change clock speed of CPUs

Clock scaling allows you to change the clock speed of the CPUs on the fly. This is a nice method to save battery power, because the lower the clock speed, the less power the CPU consumes.

There are many knobs to tweak in this directory.

See files in Documentation/cpu-freq/ for more information.

/sys/devices/system/cpu/cpuX/cpufreq/freqdomain_cpus

Defined on file *sysfs-devices-system-cpu*

Discover CPUs in the same CPU frequency coordination domain

freqdomain_cpus is the list of CPUs (online+offline) that share the same clock/freq domain (possibly at the hardware level). That information may be hidden from the cpufreq core and the value of related_cpus may be different from freqdomain_cpus. This attribute is useful for user space DVFS controllers to get better power/performance results for platforms using acpi-cpufreq.

This file is only present if the acpi-cpufreq or the cppc-cpufreq drivers are in use.

/sys/devices/system/cpu/cpuX/cpufreq/throttle_stats

Defined on file *sysfs-devices-system-cpu*

POWERNV CPUFreq driver's frequency throttle stats directory and attributes

'cpuX/cpufreq/throttle_stats' directory contains the CPU frequency throttle stat attributes for the chip. The throttle stats of a cpu is common across all the cpus belonging to a chip. Below are the throttle attributes exported in the 'throttle_stats' directory:

- turbo_stat : This file gives the total number of times the max frequency is throttled to lower frequency in turbo (at and above nominal frequency) range of frequencies.
- sub_turbo_stat : This file gives the total number of times the max frequency is throttled to lower frequency in sub-turbo(below nominal frequency) range of frequencies.
- unthrottle : This file gives the total number of times the max frequency is unthrottled after being throttled.
- powercap : This file gives the total number of times the max frequency is throttled due to 'Power Capping'.
- overtemp : This file gives the total number of times the max frequency is throttled due to 'CPU Over Temperature'.
- supply_fault : This file gives the total number of times the max frequency is throttled due to 'Power Supply Failure'.
- overcurrent : This file gives the total number of times the max frequency is throttled due to 'Overcurrent'.
- occ_reset : This file gives the total number of times the max frequency is throttled due to 'OCC Reset'.

The sysfs attributes representing different throttle reasons like powercap, overtemp, supply_fault, overcurrent and occ_reset map to the reasons provided by OCC firmware for throttling the frequency.

/sys/devices/system/cpu/cpuX/cpuidle/state<N>/default_status

Defined on file *sysfs-devices-system-cpu*

(RO) The default status of this state, "enabled" or "disabled".

/sys/devices/system/cpu/cpuX/cpuidle/state<N>/desc

Defined on file *sysfs-devices-system-cpu*

(RO) A small description about the idle state (string).

/sys/devices/system/cpu/cpuX/cpuidle/state<N>/disable

Defined on file *sysfs-devices-system-cpu*

(RW) Option to disable this idle state (bool). The behavior and the effect of the disable variable depends on the implementation of a particular governor. In the ladder governor, for example, it is not coherent, i.e. if one is disabling a light state, then all deeper states are disabled as well, but the disable variable does not reflect it. Likewise, if one enables a deep state but a lighter state still is disabled, then this has no effect.

/sys/devices/system/cpu/cpuX/cpuidle/state<N>/name

Defined on file *sysfs-devices-system-cpu*

The directory /sys/devices/system/cpu/cpuX/cpuidle contains per logical CPU specific cpuidle information for each online cpu X. The processor idle states which are available for use have the following attributes:

name:	(RO)	Name of the idle state (string).
latency:	(RO)	The latency to exit out of this idle state (in microseconds).
power:	(RO)	The power consumed while in this idle state (in milliwatts).
time:	(RO)	The total time spent in this idle state (in microseconds).
usage:	(RO)	Number of times this state was entered (a count).
above:	(RO)	Number of times this state was entered, but the observed CPU idle duration was too short for it (a count).
below:	(RO)	Number of times this state was entered, but the observed CPU idle duration was too long for it (a count).

/sys/devices/system/cpu/cpuX/cpuidle/state<N>/residency

Defined on file *sysfs-devices-system-cpu*

(RO) Display the target residency i.e. the minimum amount of time (in microseconds) this cpu should spend in this idle state to make the transition worth the effort.

/sys/devices/system/cpu/cpuX/cpuidle/state<N>/s2idle/

Defined on file *sysfs-devices-system-cpu*

Idle state usage statistics related to suspend-to-idle.

This attribute group is only present for states that can be used in suspend-to-idle with suspended timekeeping.

/sys/devices/system/cpu/cpuX/cpuidle/state<N>/s2idle/time

Defined on file *sysfs-devices-system-cpu*

Total time spent by the CPU in suspend-to-idle (with scheduler tick suspended) after requesting this state.

/sys/devices/system/cpu/cpuX/cpuidle/state<N>/s2idle/usage

Defined on file *sysfs-devices-system-cpu*

Total number of times this state has been requested by the CPU while entering suspend-to-idle.

/sys/devices/system/cpu/cpuX/crash_notes

Defined on file *sysfs-devices-system-cpu*

address and size of the percpu note.

crash_notes: the physical address of the memory that holds the note of cpuX.

crash_notes_size: size of the note of cpuX.

/sys/devices/system/cpu/cpuX/idle_purr

Defined on file *sysfs-devices-system-cpu*

PURR ticks for cpuX when it was idle.

This sysfs interface exposes the number of PURR ticks for cpuX when it was idle.

/sys/devices/system/cpu/cpuX/idle_spurr

Defined on file *sysfs-devices-system-cpu*

SPURR ticks for cpuX when it was idle.

This sysfs interface exposes the number of SPURR ticks for cpuX when it was idle.

/sys/devices/system/cpu/cpuX/mte_tcf_preferred

Defined on file *sysfs-devices-system-cpu*

Preferred MTE tag checking mode

When a user program specifies more than one MTE tag checking mode, this sysfs node is used to specify which mode should be preferred when scheduling a task on that CPU. Possible values:

”sync”	Prefer synchronous mode
”asymm”	Prefer asymmetric mode
”async”	Prefer asynchronous mode

See also: /arch/arm64/memory-tagging-extension

/sys/devices/system/cpu/cpuX/node

Defined on file *sysfs-devices-system-cpu*

Discover NUMA node a CPU belongs to

When CONFIG_NUMA is enabled, a symbolic link that points to the corresponding NUMA node directory.

For example, the following symlink is created for cpu42 in NUMA node 2:

/sys/devices/system/cpu/cpu42/node2 -> ../../node/node2

/sys/devices/system/cpu/cpuX/power/energy_perf_bias

Defined on file *sysfs-devices-system-cpu*

Intel Energy and Performance Bias Hint (EPB)

EPB for the given CPU in a sliding scale 0 - 15, where a value of 0 corresponds to a hint preference for highest performance and a value of 15 corresponds to the maximum energy savings.

In order to change the EPB value for the CPU, write either a number in the 0 - 15 sliding scale above, or one of the strings: ”performance”, ”balance-performance”, ”normal”, ”balance-power”, ”power” (that represent values reflected by their meaning), to this attribute.

This attribute is present for all online CPUs supporting the Intel EPB feature.

/sys/devices/system/cpu/cpuX/purr

Defined on file *sysfs-devices-system-cpu*

PURR ticks for this CPU since the system boot.

The Processor Utilization Resources Register (PURR) is a 64-bit counter which provides an estimate of the resources used by the CPU thread. The contents of this register increases monotonically. This sysfs interface exposes the number of PURR ticks for cpuX.

/sys/devices/system/cpu/cpuX/regs/

Defined on file *sysfs-devices-system-cpu*

AArch64 CPU registers

'identification' directory exposes the CPU ID registers for identifying model and revision of the CPU and SMCU.

/sys/devices/system/cpu/cpuX/spurr

Defined on file *sysfs-devices-system-cpu*

SPURR ticks for this CPU since the system boot.

The Scaled Processor Utilization Resources Register (SPURR) is a 64-bit counter that provides a frequency invariant estimate of the resources used by the CPU thread. The contents of this register increases monotonically. This sysfs interface exposes the number of SPURR ticks for cpuX.

/sys/devices/system/cpu/cpuX/topology/core_siblings

Defined on file *sysfs-devices-system-cpu*

CPU topology files that describe a logical CPU's relationship to other cores and threads in the same physical package.

One cpuX directory is created per logical CPU in the system, e.g. /sys/devices/system/cpu/cpu42/.

Briefly, the files above are:

core_siblings: internal kernel map of cpuX's hardware threads within the same physical_package_id.

core_siblings_list: human-readable list of the logical CPU numbers within the same physical_package_id as cpuX.

physical_package_id: physical package id of cpuX. Typically corresponds to a physical socket number, but the actual value is architecture and platform dependent.

thread_siblings: internal kernel map of cpuX's hardware threads within the same core as cpuX

thread_siblings_list: human-readable list of cpuX's hardware threads within the same core as cpuX

ppin: human-readable Protected Processor Identification Number of the socket the cpu# belongs to. There should be one per physical_package_id. File is readable only to admin.

See *How CPU topology info is exported via sysfs* for more information.

/sys/devices/system/cpu/cpufreq/boost

Defined on file *sysfs-devices-system-cpu*

Processor frequency boosting control

This switch controls the boost setting for the whole system. Boosting allows the CPU and the firmware to run at a frequency beyond its nominal limit.

More details can be found in *CPU Performance Scaling*

/sys/devices/system/cpu/cpufreq/policyX/throttle_stats

Defined on file *sysfs-devices-system-cpu*

POWERNV CPUFreq driver's frequency throttle stats directory and attributes

'policyX/throttle_stats' directory and all the attributes are same as the */sys/devices/system/cpu/cpuX/cpufreq/throttle_stats* directory and attributes which give the frequency throttle information of the chip.

/sys/devices/system/cpu/cpuidle/available_governors

Defined on file *sysfs-devices-system-cpu*

Discover cpuidle policy and mechanism

Various CPUs today support multiple idle levels that are differentiated by varying exit latencies and power consumption during idle.

Idle policy (governor) is differentiated from idle mechanism (driver).

available_governors: (RO) displays a space separated list of available governors.

current_driver: (RO) displays current idle mechanism.

current_governor: (RW) displays current idle policy. Users can switch the governor at runtime by writing to this file.

current_governor_ro: (RO) displays current idle policy.

See *CPU Idle Time Management* and /driver-api/pm/cpuidle for more information.

/sys/devices/system/cpu/crash_hotplug

Defined on file *sysfs-devices-system-cpu*

(RO) indicates whether or not the kernel directly supports modifying the crash elfcorehdr for CPU hot un/plug and/or on/offline changes.

/sys/devices/system/cpu/intel_pstate/max_perf_pct

Defined on file *sysfs-devices-system-cpu*

Parameters for the Intel P-state driver

Logic for selecting the current P-state in Intel Sandybridge+ processors. The three knobs control limits for the P-state that will be requested by the driver.

max_perf_pct: limits the maximum P state that will be requested by the driver stated as a percentage of the available performance.

min_perf_pct: limits the minimum P state that will be requested by the driver stated as a percentage of the available performance.

no_turbo: limits the driver to selecting P states below the turbo frequency range.

More details can be found in *intel_pstate CPU Performance Scaling Driver*

/sys/devices/system/cpu/isolated

Defined on file *sysfs-devices-system-cpu*

(RO) the list of CPUs that are isolated and don't participate in load balancing. These CPUs are set by boot parameter ”isolcpus=”.

/sys/devices/system/cpu/kernel_max

Defined on file *sysfs-devices-system-cpu*

CPU topology files that describe kernel limits related to hotplug. Briefly:

kernel_max: the maximum cpu index allowed by the kernel configuration.

offline: cpus that are not online because they have been HOTPLUGGED off or exceed the limit of cpus allowed by the kernel configuration (kernel_max above).

online: cpus that are online and being scheduled.

possible: cpus that have been allocated resources and can be brought online if they are present.

present: cpus that have been identified as being present in the system.

See *How CPU topology info is exported via sysfs* for more information.

/sys/devices/system/cpu/nohz_full

Defined on file *sysfs-devices-system-cpu*

(RO) the list of CPUs that are in nohz_full mode. These CPUs are set by boot parameter "nohz_full=".

/sys/devices/system/cpu/probe

Defined on file *sysfs-devices-system-cpu*

Dynamic addition and removal of CPU's. This is not hotplug removal, this is meant complete removal/addition of the CPU from the system.

probe: writes to this file will dynamically add a CPU to the system. Information written to the file to add CPU's is architecture specific.

release: writes to this file dynamically remove a CPU from the system. Information written to the file to remove CPU's is architecture specific.

/sys/devices/system/cpu/smt

Defined on file *sysfs-devices-system-cpu*

Control Symmetric Multi Threading (SMT)

active: Tells whether SMT is active (enabled and siblings online)

control: Read/write interface to control SMT. Possible
values:

"on"	SMT is enabled
"off"	SMT is disabled
"<N>"	SMT is enabled with N threads per core.
"forceoff"	SMT is force disabled. Cannot be changed.
"notsupported"	SMT is not supported by the CPU
"notimplemented"	SMT runtime toggling is not implemented for the architecture

If control status is "forceoff" or "notsupported" writes are rejected.

/sys/devices/system/cpu/svm

Defined on file *sysfs-devices-system-cpu*

Secure Virtual Machine

If 1, it means the system is using the Protected Execution Facility in POWER9 and newer processors. i.e., it is a Secure Virtual Machine.

/sys/devices/system/cpu/umwait_control

Defined on file *sysfs-devices-system-cpu*

Umwait control

enable_c02: Read/write interface to control umwait C0.2 state

Read returns C0.2 state status:
0: C0.2 is disabled 1: C0.2 is enabled

Write 'y' or '1' or 'on' to enable C0.2 state. Write 'n' or '0' or 'off' to disable C0.2 state.

The interface is case insensitive.

max_time: Read/write interface to control umwait maximum time
in TSC-quanta that the CPU can reside in either C0.1 or C0.2 state. The time is an unsigned 32-bit number. Note that a value of zero means there is no limit. Low order two bits must be zero.

/sys/devices/system/cpu/vulnerabilities

Defined on file *sysfs-devices-system-cpu*

Information about CPU vulnerabilities

The files are named after the code names of CPU vulnerabilities. The output of those files reflects the state of the CPUs in the system. Possible output values:

"Not affected"	CPU is not affected by the vulnerability
"Vulnerable"	CPU is affected and no mitigation in effect
"Mitigation: $M"	CPU is affected and mitigation $M is in effect

See also: *Hardware vulnerabilities*

/sys/devices/system/edac/mc/mc*/(dimm|rank)*/dimm_ce_count

Defined on file *sysfs-devices-edac*

This attribute file displays the total count of correctable errors that have occurred on this DIMM. This count is very important to examine. CEs provide early indications that a DIMM is beginning to fail. This count field should be monitored for non-zero values and report such information to the system administrator.

/sys/devices/system/edac/mc/mc*/(dimm|rank)*/dimm_dev_type

Defined on file *sysfs-devices-edac*

This attribute file will display what type of DRAM device is being utilized on this DIMM (x1, x2, x4, x8, ...).

/sys/devices/system/edac/mc/mc*/(dimm|rank)*/dimm_edac_mode

Defined on file *sysfs-devices-edac*

This attribute file will display what type of Error detection and correction is being utilized. For example: S4ECD4ED would mean a Chipkill with x4 DRAM.

/sys/devices/system/edac/mc/mc*/(dimm|rank)*/dimm_label

Defined on file *sysfs-devices-edac*

This control file allows this DIMM to have a label assigned to it. With this label in the module, when errors occur the output can provide the DIMM label in the system log. This becomes vital for panic events to isolate the cause of the UE event. DIMM Labels must be assigned after booting, with information that correctly identifies the physical slot with its silk screen label. This information is currently very motherboard specific and determination of this information must occur in userland at this time.

/sys/devices/system/edac/mc/mc*/(dimm|rank)*/dimm_location

Defined on file *sysfs-devices-edac*

This attribute file will display the location (csrow/channel, branch/channel/slot or channel/slot) of the dimm or rank.

/sys/devices/system/edac/mc/mc*/(dimm|rank)*/dimm_mem_type

Defined on file *sysfs-devices-edac*

This attribute file will display what type of memory is currently on this csrow. Normally, either buffered or unbuffered memory (for example, Unbuffered-DDR3).

/sys/devices/system/edac/mc/mc*/(dimm|rank)*/dimm_ue_count

Defined on file *sysfs-devices-edac*

This attribute file displays the total count of uncorrectable errors that have occurred on this DIMM. If panic_on_ue is set, this counter will not have a chance to increment, since EDAC will panic the system

/sys/devices/system/edac/mc/mc*/(dimm|rank)*/size

Defined on file *sysfs-devices-edac*

This attribute file will display the size of dimm or rank. For dimm*/size, this is the size, in MB of the DIMM memory stick. For rank*/size, this is the size, in MB for one rank of the DIMM memory stick. On single rank memories (1R), this is also the total size of the dimm. On dual rank (2R) memories, this is half the size of the total DIMM memories.

/sys/devices/system/edac/mc/mc*/ce_count

Defined on file *sysfs-devices-edac*

This attribute file displays the total count of correctable errors that have occurred on this memory controller. This count is very important to examine. CEs provide early indications that a DIMM is beginning to fail. This count field should be monitored for non-zero values and report such information to the system administrator.

/sys/devices/system/edac/mc/mc*/ce_noinfo_count

Defined on file *sysfs-devices-edac*

This attribute file displays the number of CEs that have occurred on this memory controller wherewith no information as to which DIMM slot is having errors. Memory is handicapped, but operational, yet no information is available to indicate which slot the failing memory is in. This count field should be also be monitored for non-zero values.

/sys/devices/system/edac/mc/mc*/max_location

Defined on file *sysfs-devices-edac*

This attribute file displays the information about the last available memory slot in this memory controller. It is used by userspace tools in order to display the memory filling layout.

/sys/devices/system/edac/mc/mc*/mc_name

Defined on file *sysfs-devices-edac*

This attribute file displays the type of memory controller that is being utilized.

/sys/devices/system/edac/mc/mc*/reset_counters

Defined on file *sysfs-devices-edac*

This write-only control file will zero all the statistical counters for UE and CE errors on the given memory controller. Zeroing the counters will also reset the timer indicating how long since the last counter were reset. This is useful for computing errors/time. Since the counters are always reset at driver initialization time, no module/kernel parameter is available.

/sys/devices/system/edac/mc/mc*/sdram_scrub_rate

Defined on file *sysfs-devices-edac*

Read/Write attribute file that controls memory scrubbing. The scrubbing rate used by the memory controller is set by writing a minimum bandwidth in bytes/sec to the attribute file. The rate will be translated to an internal value that gives at least the specified rate. Reading the file will

return the actual scrubbing rate employed. If configuration fails or memory scrubbing is not implemented, the value of the attribute file will be -1.

/sys/devices/system/edac/mc/mc*/seconds_since_reset

Defined on file *sysfs-devices-edac*

This attribute file displays how many seconds have elapsed since the last counter reset. This can be used with the error counters to measure error rates.

/sys/devices/system/edac/mc/mc*/size_mb

Defined on file *sysfs-devices-edac*

This attribute file displays, in count of megabytes, of memory that this memory controller manages.

/sys/devices/system/edac/mc/mc*/ue_count

Defined on file *sysfs-devices-edac*

This attribute file displays the total count of uncorrectable errors that have occurred on this memory controller. If panic_on_ue is set, this counter will not have a chance to increment, since EDAC will panic the system

/sys/devices/system/edac/mc/mc*/ue_noinfo_count

Defined on file *sysfs-devices-edac*

This attribute file displays the number of UEs that have occurred on this memory controller with no information as to which DIMM slot is having errors.

/sys/devices/system/ibm_rtl/state

Defined on file *sysfs-devices-system-ibm-rtl*

The state file allows a means by which to change in and out of Premium Real-Time Mode (PRTM), as well as the ability to query the current state.

- 0 => PRTM off
- 1 => PRTM enabled

Users: The ibm-prtm userspace daemon uses this interface.

/sys/devices/system/ibm_rtl/version

Defined on file *sysfs-devices-system-ibm-rtl*

The version file provides a means by which to query the RTL table version that lives in the Extended BIOS Data Area (EBDA).

Users: The ibm-prtm userspace daemon uses this interface.

/sys/devices/system/machinecheck/machinecheckX/

Defined on file *sysfs-mce*

(X = CPU number)

Machine checks report internal hardware error conditions detected by the CPU. Uncorrected errors typically cause a machine check (often with panic), corrected ones cause a machine check log entry.

For more details about the x86 machine check architecture see the Intel and AMD architecture manuals from their developer websites.

For more details about the architecture see http://one.firstfloor.org/~andi/mce.pdf

Each CPU has its own directory.

/sys/devices/system/machinecheck/machinecheckX/bank<Y>

Defined on file *sysfs-mce*

(Y bank number)

64bit Hex bitmask enabling/disabling specific subevents for bank Y.

When a bit in the bitmask is zero then the respective subevent will not be reported.

By default all events are enabled.

Note that BIOS maintain another mask to disable specific events per bank. This is not visible here

/sys/devices/system/machinecheck/machinecheckX/check_interval

Defined on file *sysfs-mce*

The entries appear for each CPU, but they are truly shared between all CPUs.

How often to poll for corrected machine check errors, in seconds (Note output is hexadecimal). Default 5 minutes. When the poller finds MCEs it triggers an exponential speedup (poll more often) on the polling interval. When the poller stops finding MCEs, it triggers an exponential backoff (poll less often) on the polling interval. The check_interval variable is both the initial and maximum polling interval. 0 means no polling for corrected machine check errors (but some corrected errors might be still reported in other ways)

/sys/devices/system/machinecheck/machinecheckX/cmci_disabled

Defined on file *sysfs-mce*

Disables the CMCI feature.

/sys/devices/system/machinecheck/machinecheckX/dont_log_ce

Defined on file *sysfs-mce*

Disables logging for corrected errors. All reported corrected errors will be cleared silently.

This option will be useful if you never care about corrected errors.

/sys/devices/system/machinecheck/machinecheckX/ignore_ce

Defined on file *sysfs-mce*

Disables polling and CMCI for corrected errors. All corrected events are not cleared and kept in bank MSRs.

/sys/devices/system/machinecheck/machinecheckX/monarch_timeout

Defined on file *sysfs-mce*

How long to wait for the other CPUs to machine check too on a exception. 0 to disable waiting for other CPUs.

Unit: us

/sys/devices/system/machinecheck/machinecheckX/trigger

Defined on file *sysfs-mce*

The entries appear for each CPU, but they are truly shared between all CPUs.

Program to run when a machine check event is detected. This is an alternative to running mcelog regularly from cron and allows to detect events faster.

/sys/devices/system/memory

Defined on file *sysfs-devices-memory*

The */sys/devices/system/memory* contains a snapshot of the internal state of the kernel memory blocks. Files could be added or removed dynamically to represent hot-add/remove operations.

Users: hotplug memory add/remove tools http://www.ibm.com/developerworks/wikis/display/LinuxP/powerpc-utils

/sys/devices/system/memory/crash_hotplug

Defined on file *sysfs-devices-memory*

(RO) indicates whether or not the kernel directly supports modifying the crash elfcorehdr for memory hot un/plug and/or on/offline changes.

/sys/devices/system/memory/hard_offline_page

Defined on file *sysfs-memory-page-offline*

Hard-offline the memory page containing the physical address written into this file. Input is a hex number specifying the physical address of the page. The kernel will then attempt to hard-offline the page, by trying to drop the page or killing any owner or triggering IO errors if needed. Note this may kill any processes owning the page. The kernel will avoid to access this page assuming it's poisoned by the hardware.

The offlining is done in kernel specific granularity. Normally it's the base page size of the kernel, but this might change.

Return value is the size of the number, or a error when the offlining failed. Reading the file is not allowed.

/sys/devices/system/memory/memoryX/phys_device

Defined on file *sysfs-devices-memory*

The file */sys/devices/system/memory/memoryX/phys_device* is read-only; it is a legacy interface only ever used on s390x to expose the covered storage increment.

Users: Legacy s390-tools lsmem/chmem

/sys/devices/system/memory/memoryX/phys_index

Defined on file *sysfs-devices-memory*

The file */sys/devices/system/memory/memoryX/phys_index* is read-only and contains the section ID in hexadecimal which is equivalent to decimal X contained in the memory section directory name.

/sys/devices/system/memory/memoryX/removable

Defined on file *sysfs-devices-memory*

The file */sys/devices/system/memory/memoryX/removable* is a legacy interface used to indicated whether a memory block is likely to be offlineable or not. Newer kernel versions return "1" if and only if the kernel supports memory offlining.

Users: hotplug memory remove tools http://www.ibm.com/developerworks/wikis/display/LinuxP/powerpc-utils lsmem/chmem part of util-linux

/sys/devices/system/memory/memoryX/state

Defined on file *sysfs-devices-memory*

The file */sys/devices/system/memory/memoryX/state* is read-write. When read, it returns the online/offline state of the memory block. When written, root can toggle the online/offline state of a memory block using the following commands:

```
# echo online > /sys/devices/system/memory/memoryX/state
# echo offline > /sys/devices/system/memory/memoryX/state
```

On newer kernel versions, advanced states can be specified when onlining to select a target zone: "online_movable" selects the movable zone. "online_kernel" selects the applicable kernel zone (DMA, DMA32, or Normal). However, after successfully setting one of the advanced states, reading the file will return "online"; the zone information can be obtained via "valid_zones" instead.

While onlining is unlikely to fail, there are no guarantees that offlining will succeed. Offlining is more likely to succeed if "valid_zones" indicates "Movable".

Users: hotplug memory remove tools http://www.ibm.com/developerworks/wikis/display/LinuxP/powerpc-utils

/sys/devices/system/memory/memoryX/valid_zones

Defined on file *sysfs-devices-memory*

The file */sys/devices/system/memory/memoryX/valid_zones* is read-only.

For online memory blocks, it returns in which zone memory provided by a memory block is managed. If multiple zones apply (not applicable for hotplugged memory), "None" is returned and the memory block cannot be offlined.

For offline memory blocks, it returns by which zone memory provided by a memory block can be managed when onlining. The first returned zone ("default") will be used when setting the state of an offline memory block to "online". Only one of the kernel zones (DMA, DMA32, Normal) is applicable for a single memory block.

/sys/devices/system/memory/soft_offline_page

Defined on file *sysfs-memory-page-offline*

Soft-offline the memory page containing the physical address written into this file. Input is a hex number specifying the physical address of the page. The kernel will then attempt to soft-offline it, by moving the contents elsewhere or dropping it if possible. The kernel will then be placed on the bad page list and never be reused.

The offlining is done in kernel specific granularity. Normally it's the base page size of the kernel, but this might change.

The page must be still accessible, not poisoned. The kernel will never kill anything for this, but rather fail the offline. Return value is the size of the number, or a error when the offlining failed. Reading the file is not allowed.

/sys/devices/system/memoryX/nodeY

Defined on file *sysfs-devices-memory*

When CONFIG_NUMA is enabled, a symbolic link that points to the corresponding NUMA node directory.

For example, the following symbolic link is created for memory section 9 on node0:

/sys/devices/system/memory/memory9/node0 -> ../../node/node0

/sys/devices/system/node/nodeX/memoryY

Defined on file *sysfs-devices-memory*

When CONFIG_NUMA is enabled */sys/devices/system/node/nodeX/memoryY* is a symbolic link that points to the corresponding /sys/devices/system/memory/memoryY memory section directory. For example, the following symbolic link is created for memory section 9 on node0.

/sys/devices/system/node/node0/memory9 -> ../../memory/memory9

/sys/devices/system/xen_cpu/

Defined on file *sysfs-devices-system-xen_cpu*

A collection of global/individual Xen physical cpu attributes

Individual physical cpu attributes are contained in subdirectories named by the Xen's logical cpu number, e.g.: /sys/devices/system/xen_cpu/xen_cpu#/

/sys/devices/system/xen_cpu/xen_cpu#/online

Defined on file *sysfs-devices-system-xen_cpu*

Interface to online/offline Xen physical cpus

When running under Xen platform, it provide user interface to online/offline physical cpus, except cpu0 due to several logic restrictions and assumptions.

/sys/devices/uncore_iio_x/dieX

Defined on file *sysfs-devices-mapping*

Each IIO stack (PCIe root port) has its own IIO PMON block, so each dieX file (where X is die number) holds "Segment:Root Bus" for PCIe root port, which can be monitored by that IIO PMON block. For example, on 4-die Xeon platform with up to 6 IIO stacks per die and, therefore, 6 IIO PMON blocks per die, the mapping of IIO PMON block 0 exposes as the following:

```
$ ls /sys/devices/uncore_iio_0/die*
-r--r--r-- /sys/devices/uncore_iio_0/die0
-r--r--r-- /sys/devices/uncore_iio_0/die1
-r--r--r-- /sys/devices/uncore_iio_0/die2
-r--r--r-- /sys/devices/uncore_iio_0/die3

$ tail /sys/devices/uncore_iio_0/die*
==> /sys/devices/uncore_iio_0/die0 <==
0000:00
==> /sys/devices/uncore_iio_0/die1 <==
0000:40
==> /sys/devices/uncore_iio_0/die2 <==
0000:80
==> /sys/devices/uncore_iio_0/die3 <==
0000:c0
```

Which means:

```
IIO PMU 0 on die 0 belongs to PCI RP on bus 0x00, domain 0x0000
IIO PMU 0 on die 1 belongs to PCI RP on bus 0x40, domain 0x0000
IIO PMU 0 on die 2 belongs to PCI RP on bus 0x80, domain 0x0000
IIO PMU 0 on die 3 belongs to PCI RP on bus 0xc0, domain 0x0000
```

/sys/devices/uncore_upi_x/dieX

Defined on file *sysfs-devices-mapping*

Each /sys/devices/uncore_upi_X/dieY file holds "upi_Z,die_W" value that means UPI link number X on die Y is connected to UPI link Z on die W and this link between sockets can be monitored by UPI PMON block. For example, 4-die Sapphire Rapids platform has the following UPI 0 topology:

```
# tail /sys/devices/uncore_upi_0/die*
==> /sys/devices/uncore_upi_0/die0 <==
upi_1,die_1
==> /sys/devices/uncore_upi_0/die1 <==
upi_0,die_3
==> /sys/devices/uncore_upi_0/die2 <==
upi_1,die_3
==> /sys/devices/uncore_upi_0/die3 <==
upi_0,die_1
```

Which means:

```
UPI link 0 on die 0 is connected to UPI link 1 on die 1
UPI link 0 on die 1 is connected to UPI link 0 on die 3
UPI link 0 on die 2 is connected to UPI link 1 on die 3
UPI link 0 on die 3 is connected to UPI link 0 on die 1
```

/sys/devices/virtual/memory_tiering/

Defined on file *sysfs-kernel-mm-memory-tiers*

A collection of all the memory tiers allocated.

Individual memory tier details are contained in subdirectories named by the abstract distance of the memory tier.

/sys/devices/virtual/memory_tiering/memory_tierN/

/sys/devices/virtual/memory_tiering/memory_tierN/

Defined on file *sysfs-kernel-mm-memory-tiers*

Directory with details of a specific memory tier

This is the directory containing information about a particular memory tier, memtierN, where N is derived based on abstract distance.

A smaller value of N implies a higher (faster) memory tier in the hierarchy.

nodelist: NUMA nodes that are part of this memory tier.

/sys/devices/virtual/misc/intel_ifs_<N>/current_batch

Defined on file *sysfs-platform-intel-ifs*

Write a number less than or equal to 0xff to load an IFS test image. The number written treated as the 2 digit suffix in the following file name: /lib/firmware/intel/ifs_<N>/ff-mm-ss-02x.scan

Reading the file will provide the suffix of the currently loaded IFS test image. This file is present only for device instances where a test image is applicable. Devices: intel_ifs_0

/sys/devices/virtual/misc/intel_ifs_<N>/details

Defined on file *sysfs-platform-intel-ifs*

Additional information regarding the last test. The details file reports the hex value of the STATUS MSR for this test. Note that the error_code field may contain driver defined software code not defined in the Intel SDM. Devices: all

/sys/devices/virtual/misc/intel_ifs_<N>/image_version

Defined on file *sysfs-platform-intel-ifs*

Version (hexadecimal) of loaded IFS test image. If no test image is loaded reports "none". Only present for device instances where a test image is applicable. Devices: intel_ifs_0

/sys/devices/virtual/misc/intel_ifs_<N>/run_test

Defined on file *sysfs-platform-intel-ifs*

Write <cpu#> to trigger IFS test for one online core. Note that the test is per core. The cpu# can be for any thread on the core. Running on one thread completes the test for the core containing that thread. Example: to test the core containing cpu5: echo 5 > */sys/devices/virtual/misc/intel_ifs_<N>/run_test* Devices: all

/sys/devices/virtual/misc/intel_ifs_<N>/status

Defined on file *sysfs-platform-intel-ifs*

The status of the last test. It can be one of "pass", "fail" or "untested". Devices: all

5.2.35 Symbols under /sys/firmware

/sys/firmware/acpi/bgrt/

Defined on file *sysfs-firmware-acpi*

The BGRT is an ACPI 5.0 feature that allows the OS to obtain a copy of the firmware boot splash and some associated metadata. This is intended to be used by boot splash applications in order to interact with the firmware boot splash in order to avoid jarring transitions.

image: The image bitmap. Currently a 32-bit BMP. status: 1 if the image is valid, 0 if firmware invalidated it. type: 0 indicates image is in BMP format.

version:	The version of the BGRT. Currently 1.
xoffset:	The number of pixels between the left of the screen and the left edge of the image.
yoffset:	The number of pixels between the top of the screen and the top edge of the image.

/sys/firmware/acpi/fpdt/

Defined on file *sysfs-firmware-acpi*

ACPI Firmware Performance Data Table (FPDT) provides information for firmware performance data for system boot, S3 suspend and S3 resume. This sysfs entry contains the performance data retrieved from the FPDT.

boot:

firmware_start_ns: Timer value logged at the beginning
of firmware image execution. In nanoseconds.

bootloader_load_ns: Timer value logged just prior to
loading the OS boot loader into memory. In nanoseconds.

bootloader_launch_ns: Timer value logged just prior to
launching the currently loaded OS boot loader image. In nanoseconds.

exitbootservice_start_ns: Timer value logged at the
point when the OS loader calls the ExitBootServices function for UEFI compatible firmware. In nanoseconds.

exitbootservice_end_ns: Timer value logged at the point
just prior to the OS loader gaining control back from the ExitBootServices function for UEFI compatible firmware. In nanoseconds.

suspend:

suspend_start_ns: Timer value recorded at the previous
OS write to SLP_TYP upon entry to S3. In nanoseconds.

suspend_end_ns: Timer value recorded at the previous
firmware write to SLP_TYP used to trigger hardware entry to S3. In nanoseconds.

resume:

resume_count: A count of the number of S3 resume cycles
since the last full boot sequence.

resume_avg_ns: Average timer value of all resume cycles
logged since the last full boot sequence, including the most recent resume. In nanoseconds.

resume_prev_ns: Timer recorded at the end of the previous
platform runtime firmware S3 resume, just prior to handoff to the OS waking vector. In nanoseconds.

/sys/firmware/acpi/hotplug/

Defined on file *sysfs-firmware-acpi*

There are separate hotplug profiles for different classes of devices supported by ACPI, such as containers, memory modules, processors, PCI root bridges etc. A hotplug profile for a given class of devices is a collection of settings defining the way that class of devices will be handled by the ACPI core hotplug code. Those profiles are represented in sysfs as subdirectories of */sys/firmware/acpi/hotplug/*.

The following setting is available to user space for each hotplug profile:

enabled:	If set, the ACPI core will handle notifications of hotplug events associated with the given class of devices and will allow those devices to be ejected with the help of the _EJ0 control method. Unsetting it effectively disables hotplug for the corresponding class of devices.

The value of the above attribute is an integer number: 1 (set) or 0 (unset). Attempts to write any other values to it will cause -EINVAL to be returned.

/sys/firmware/acpi/interrupts/

Defined on file *sysfs-firmware-acpi*

All ACPI interrupts are handled via a single IRQ, the System Control Interrupt (SCI), which appears as "acpi" in /proc/interrupts.

However, one of the main functions of ACPI is to make the platform understand random hardware without special driver support. So while the SCI handles a few well known (fixed feature) interrupts sources, such as the power button, it can also handle a variable number of a "General Purpose Events" (GPE).

A GPE vectors to a specified handler in AML, which can do a anything the BIOS writer wants from OS context. GPE 0x12, for example, would vector to a level or edge handler called _L12 or _E12. The handler may do its business and return. Or the handler may send send a Notify event to a Linux device driver registered on an ACPI device, such as a battery, or a processor.

To figure out where all the SCI's are coming from, /sys/firmware/acpi/interrupts contains a file listing every possible source, and the count of how many times it has triggered:

```
$ cd /sys/firmware/acpi/interrupts
$ grep . *
error:             0
ff_gbl_lock:       0   enable
ff_pmtimer:        0  invalid
ff_pwr_btn:        0   enable
ff_rt_clk:         2  disable
ff_slp_btn:        0  invalid
gpe00:             0  invalid
gpe01:             0   enable
gpe02:           108   enable
gpe03:             0  invalid
gpe04:             0  invalid
gpe05:             0  invalid
gpe06:             0   enable
gpe07:             0   enable
gpe08:             0  invalid
gpe09:             0  invalid
gpe0A:             0  invalid
gpe0B:             0  invalid
gpe0C:             0  invalid
gpe0D:             0  invalid
```

```
gpe0E:             0  invalid
gpe0F:             0  invalid
gpe10:             0  invalid
gpe11:             0  invalid
gpe12:             0  invalid
gpe13:             0  invalid
gpe14:             0  invalid
gpe15:             0  invalid
gpe16:             0  invalid
gpe17:          1084   enable
gpe18:             0   enable
gpe19:             0  invalid
gpe1A:             0  invalid
gpe1B:             0  invalid
gpe1C:             0  invalid
gpe1D:             0  invalid
gpe1E:             0  invalid
gpe1F:             0  invalid
gpe_all:        1192
sci:            1194
sci_not:           0
```

sci	The number of times the ACPI SCI has been called and claimed an interrupt.
sci_not	The number of times the ACPI SCI has been called and NOT claimed an interrupt.
gpe_all	count of SCI caused by GPEs.
gpeXX	count for individual GPE source
ff_gbl_lock	Global Lock
ff_pmtimer	PM Timer
ff_pwr_btn	Power Button
ff_rt_clk	Real Time Clock
ff_slp_btn	Sleep Button
error	an interrupt that can't be accounted for above.
invalid	it's either a GPE or a Fixed Event that doesn't have an event handler.
disable	the GPE/Fixed Event is valid but disabled.
enable	the GPE/Fixed Event is valid and enabled.

Root has permission to clear any of these counters. Eg.:

```
# echo 0 > gpe11
```

All counters can be cleared by clearing the total ”sci”:

```
# echo 0 > sci
```

None of these counters has an effect on the function of the system, they are simply statistics.

Besides this, user can also write specific strings to these files to enable/disable/clear ACPI interrupts in user space, which can be used to debug some ACPI interrupt storm issues.

Note that only writing to VALID GPE/Fixed Event is allowed, i.e. user can only change the status

of runtime GPE and Fixed Event with event handler installed.

Let's take power button fixed event for example, please kill acpid and other user space applications so that the machine won't shutdown when pressing the power button:

```
# cat ff_pwr_btn
0     enabled
# press the power button for 3 times;
# cat ff_pwr_btn
3     enabled
# echo disable > ff_pwr_btn
# cat ff_pwr_btn
3     disabled
# press the power button for 3 times;
# cat ff_pwr_btn
3     disabled
# echo enable > ff_pwr_btn
# cat ff_pwr_btn
4     enabled
/*
 * this is because the status bit is set even if the enable
 * bit is cleared, and it triggers an ACPI fixed event when
 * the enable bit is set again
 */
# press the power button for 3 times;
# cat ff_pwr_btn
7     enabled
# echo disable > ff_pwr_btn
# press the power button for 3 times;
# echo clear > ff_pwr_btn     /* clear the status bit */
# echo disable > ff_pwr_btn
# cat ff_pwr_btn
7     enabled
```

/sys/firmware/acpi/platform_profile

Defined on file *sysfs-platform_profile*

Reading this file gives the current selected profile for this device. Writing this file with one of the strings from platform_profile_choices changes the profile to the new value.

This file can be monitored for changes by polling for POLLPRI, POLLPRI will be signalled on any changes, independent of those changes coming from a userspace write; or coming from another source such as e.g. a hotkey triggered profile change handled either directly by the embedded-controller or fully handled inside the kernel.

/sys/firmware/acpi/platform_profile_choices

Defined on file *sysfs-platform_profile*

This file contains a space-separated list of profiles supported for this device.

Drivers must use the following standard profile-names:

low-power	Low power consumption
cool	Cooler operation
quiet	Quieter operation
balanced	Balance between low power consumption and performance
balanced-performance	Balance between performance and low power consumption with a slight bias towards performance
performance	High performance operation

Userspace may expect drivers to offer more than one of these standard profile names.

/sys/firmware/devicetree/*

Defined on file *sysfs-firmware-ofw*

When using OpenFirmware or a Flattened Device Tree to enumerate hardware, the device tree structure will be exposed in this directory.

It is possible for multiple device-tree directories to exist. Some device drivers use a separate detached device tree which have no attachment to the system tree and will appear in a different subdirectory under /sys/firmware/devicetree.

Userspace must not use the /sys/firmware/devicetree/base path directly, but instead should follow /proc/device-tree symlink. It is possible that the absolute path will change in the future, but the symlink is the stable ABI.

The /proc/device-tree symlink replaces the devicetree /proc filesystem support, and has largely the same semantics and should be compatible with existing userspace.

The contents of /sys/firmware/devicetree/ is a hierarchy of directories, one per device tree node. The directory name is the resolved path component name (node name plus address). Properties are represented as files in the directory. The contents of each file is the exact binary data from the device tree.

/sys/firmware/dmi/entries/

Defined on file *sysfs-firmware-dmi-entries*

Many machines' firmware (x86 and arm64) export DMI / SMBIOS tables to the operating system. Getting at this information is often valuable to userland, especially in cases where there are OEM extensions used.

The kernel itself does not rely on the majority of the information in these tables being correct. It equally cannot ensure that the data as exported to userland is without error either.

DMI is structured as a large table of entries, where each entry has a common header indicating the type and length of the entry, as well as a firmware-provided 'handle' that is supposed to be unique amongst all entries.

Some entries are required by the specification, but many others are optional. In general though, users should never expect to find a specific entry type on their system unless they know for certain what their firmware is doing. Machine to machine experiences will vary.

Multiple entries of the same type are allowed. In order to handle these duplicate entry types, each entry is assigned by the operating system an 'instance', which is derived from an entry

type's ordinal position. That is to say, if there are 'N' multiple entries with the same type 'T' in the DMI tables (adjacent or spread apart, it doesn't matter), they will be represented in sysfs as entries ”T-0” through ”T-(N-1)”:

Example entry directories:

```
/sys/firmware/dmi/entries/17-0
/sys/firmware/dmi/entries/17-1
/sys/firmware/dmi/entries/17-2
/sys/firmware/dmi/entries/17-3
...
```

Instance numbers are used in lieu of the firmware assigned entry handles as the kernel itself makes no guarantees that handles as exported are unique, and there are likely firmware images that get this wrong in the wild.

Each DMI entry in sysfs has the common header values exported as attributes:

handle	The 16bit 'handle' that is assigned to this entry by the firmware. This handle may be referred to by other entries.
length	The length of the entry, as presented in the entry itself. Note that this is _not the total count of bytes associated with the entry. This value represents the length of the ”formatted” portion of the entry. This ”formatted” region is sometimes followed by the ”unformatted” region composed of nul terminated strings, with termination signalled by a two nul characters in series.
raw	The raw bytes of the entry. This includes the ”formatted” portion of the entry, the ”unformatted” strings portion of the entry, and the two terminating nul characters.
type	The type of the entry. This value is the same as found in the directory name. It indicates how the rest of the entry should be interpreted.
instance	The instance ordinal of the entry for the given type. This value is the same as found in the parent directory name.
position	The ordinal position (zero-based) of the entry within the entirety of the DMI entry table.

Entry Specialization

Some entry types may have other information available in sysfs. Not all types are specialized.

Type 15 - System Event Log

This entry allows the firmware to export a log of events the system has taken. This information is typically backed by nvram, but the implementation details are abstracted by this table. This entry's data is exported in the directory:

```
/sys/firmware/dmi/entries/15-0/system_event_log
```

and has the following attributes (documented in the SMBIOS / DMI specification under ”System Event Log (Type 15)”:

- area_length
- header_start_offset
- data_start_offset
- access_method

- status
- change_token
- access_method_address
- header_format
- per_log_type_descriptor_length
- type_descriptors_supported_count

As well, the kernel exports the binary attribute:

raw_event_log	The raw binary bits of the event log as described by the DMI entry.

/sys/firmware/dmi/tables/

Defined on file *sysfs-firmware-dmi-tables*

The firmware provides DMI structures as a packed list of data referenced by a SMBIOS table entry point. The SMBIOS entry point contains general information, like SMBIOS version, DMI table size, etc. The structure, content and size of SMBIOS entry point is dependent on SMBIOS version. The format of SMBIOS entry point and DMI structures can be read in SMBIOS specification.

The dmi/tables provides raw SMBIOS entry point and DMI tables through sysfs as an alternative to utilities reading them from /dev/mem. The raw SMBIOS entry point and DMI table are presented as binary attributes and are accessible via:

/sys/firmware/dmi/tables/smbios_entry_point /sys/firmware/dmi/tables/DMI

The complete DMI information can be obtained using these two tables.

/sys/firmware/efi/config_table

Defined on file *sysfs-firmware-efi*

It shows the physical address of config table entry in the EFI system table.

Users: Kexec

/sys/firmware/efi/esrt/

Defined on file *sysfs-firmware-efi-esrt*

Provides userland access to read the EFI System Resource Table (ESRT), a catalog of firmware for which can be updated with the UEFI UpdateCapsule mechanism described in section 7.5 of the UEFI Standard.

Users: fwupdate - https://github.com/rhinstaller/fwupdate

/sys/firmware/efi/esrt/entries/entry<N>/

Defined on file *sysfs-firmware-efi-esrt*

Each ESRT entry is identified by a GUID, and each gets a subdirectory under entries/ . example: /sys/firmware/efi/esrt/entries/entry0/

/sys/firmware/efi/esrt/entries/entry<N>/capsule_flags

Defined on file *sysfs-firmware-efi-esrt*

Flags that must be passed to UpdateCapsule()

/sys/firmware/efi/esrt/entries/entry<N>/fw_class

Defined on file *sysfs-firmware-efi-esrt*

This is the entry's guid, and will match the directory name.

/sys/firmware/efi/esrt/entries/entry<N>/fw_type

Defined on file *sysfs-firmware-efi-esrt*

What kind of firmware entry this is:

0	Unknown
1	System Firmware
2	Device Firmware
3	UEFI Driver

/sys/firmware/efi/esrt/entries/entry<N>/fw_version

Defined on file *sysfs-firmware-efi-esrt*

The version of the firmware currently installed. This is a 32-bit unsigned integer.

/sys/firmware/efi/esrt/entries/entry<N>/last_attempt_status

Defined on file *sysfs-firmware-efi-esrt*

The result of the last firmware update attempt for the firmware resource entry.

0	Success
1	Insufficient resources
2	Incorrect version
3	Invalid format
4	Authentication error
5	AC power event
6	Battery power event

/sys/firmware/efi/esrt/entries/entry<N>/last_attempt_version

Defined on file *sysfs-firmware-efi-esrt*

The last firmware version for which an update was attempted.

/sys/firmware/efi/esrt/entries/entry<N>/lowest_supported_fw_version

Defined on file *sysfs-firmware-efi-esrt*

The lowest version of the firmware that can be installed.

/sys/firmware/efi/esrt/fw_resource_count

Defined on file *sysfs-firmware-efi-esrt*

The number of entries in the ESRT

/sys/firmware/efi/esrt/fw_resource_count_max

Defined on file *sysfs-firmware-efi-esrt*

The maximum number of entries that /could/ be registered in the allocation the table is currently in. This is really only useful to the system firmware itself.

/sys/firmware/efi/esrt/fw_resource_version

Defined on file *sysfs-firmware-efi-esrt*

The version of the ESRT structure provided by the firmware.

/sys/firmware/efi/fw_vendor

Defined on file *sysfs-firmware-efi*

It shows the physical address of firmware vendor field in the EFI system table.

Users: Kexec

/sys/firmware/efi/runtime

Defined on file *sysfs-firmware-efi*

It shows the physical address of runtime service table entry in the EFI system table.

Users: Kexec

/sys/firmware/efi/runtime-map/

Defined on file *sysfs-firmware-efi-runtime-map*

Switching efi runtime services to virtual mode requires that all efi memory ranges which have the runtime attribute bit set to be mapped to virtual addresses.

The efi runtime services can only be switched to virtual mode once without rebooting. The kexec kernel must maintain the same physical to virtual address mappings as the first kernel.

The mappings are exported to sysfs so userspace tools can reassemble them and pass them into the kexec kernel.

/sys/firmware/efi/runtime-map/ is the directory the kernel exports that information in.

subdirectories are named with the number of the memory range:

```
/sys/firmware/efi/runtime-map/0
/sys/firmware/efi/runtime-map/1
/sys/firmware/efi/runtime-map/2
/sys/firmware/efi/runtime-map/3
...
```

Each subdirectory contains five files:

attribute	The attributes of the memory range.
num_pages	The size of the memory range in pages.
phys_addr	The physical address of the memory range.
type	The type of the memory range.
virt_addr	The virtual address of the memory range.

Above values are all hexadecimal numbers with the '0x' prefix.

Users: Kexec

/sys/firmware/efi/systab

Defined on file *sysfs-firmware-efi*

Displays the physical addresses of all EFI Configuration Tables found via the EFI System Table. The order in which the tables are printed forms an ABI and newer versions are always printed first, i.e. ACPI20 comes before ACPI.

Users: dmidecode

/sys/firmware/efi/tables/rci2

Defined on file *sysfs-firmware-efi*

Displays the content of the Runtime Configuration Interface Table version 2 on Dell EMC PowerEdge systems in binary format

Users: It is used by Dell EMC OpenManage Server Administrator tool to populate BIOS setup page.

/sys/firmware/fdt

Defined on file *sysfs-firmware-ofw*

Exports the FDT blob that was passed to the kernel by the bootloader. This allows userland applications such as kexec to access the raw binary. This blob is also useful when debugging since it contains any changes made to the blob by the bootloader.

The fact that this node does not reside under /sys/firmware/device-tree is deliberate: FDT is also used on arm64 UEFI/ACPI systems to communicate just the UEFI and ACPI entry points, but the FDT is never unflattened and used to configure the system.

A CRC32 checksum is calculated over the entire FDT blob, and verified at late_initcall time. The sysfs entry is instantiated only if the checksum is valid, i.e., if the FDT blob has not been modified in the mean time. Otherwise, a warning is printed.

Users: kexec, debugging

/sys/firmware/gsmi

Defined on file *sysfs-firmware-gsmi*

Some servers used internally at Google have firmware that provides callback functionality via explicit SMI triggers. Some of the callbacks are similar to those provided by the EFI runtime services page, but due to historical reasons this different entry-point has been used.

The gsmi driver implements the kernel's abstraction for these firmware callbacks. Currently, this functionality is limited to handling the system event log and getting access to EFI-style variables stored in nvram.

Layout:

/sys/firmware/gsmi/vars:

> This directory has the same layout (and underlying implementation as /sys/firmware/efi/vars. See *Documentation/ABI/*/sysfs-firmware-efi-vars* for more information on how to interact with this structure.

/sys/firmware/gsmi/append_to_eventlog - write-only:

> This file takes a binary blob and passes it onto the firmware to be timestamped and appended to the system eventlog. The binary format is interpreted by the firmware and may change from platform to platform. The only kernel-enforced requirement is that the blob be prefixed with a 32bit host-endian type used as part of the firmware call.

/sys/firmware/gsmi/clear_config - write-only:

> Writing any value to this file will cause the entire firmware configuration to be reset to ”factory defaults”. Callers should assume that a reboot is required for the configuration to be cleared.

/sys/firmware/gsmi/clear_eventlog - write-only:

> This file is used to clear out a portion/the whole of the system event log. Values written should be values between 1 and 100 inclusive (in ASCII) representing the fraction of the log to clear. Not all platforms support fractional clearing though, and this writes to this file will error out if the firmware doesn't like your submitted fraction.
>
> Callers should assume that a reboot is needed for this operation to complete.

/sys/firmware/ibft/acpi_header

Defined on file *sysfs-ibft*

The */sys/firmware/ibft/acpi_header* directory will contain files that expose the SIGNATURE, OEM_ID, and OEM_TABLE_ID fields of the acpi table header of the iBFT structure. This will allow for identification of the creator of the table which is useful in determining quirks associated with some adapters when used in hardware vs software iscsi initiator mode.

/sys/firmware/ibft/ethernetX

Defined on file *sysfs-ibft*

The */sys/firmware/ibft/ethernetX* directory will contain files that expose the iSCSI Boot Firmware Table NIC data. Usually this contains the IP address, MAC, and gateway of the NIC.

/sys/firmware/ibft/initiator

Defined on file *sysfs-ibft*

The */sys/firmware/ibft/initiator* directory will contain files that expose the iSCSI Boot Firmware Table initiator data. Usually this contains the Initiator name.

/sys/firmware/ibft/targetX

Defined on file *sysfs-ibft*

The */sys/firmware/ibft/targetX* directory will contain files that expose the iSCSI Boot Firmware Table target data. Usually this contains the target's IP address, boot LUN, target name, and what NIC it is associated with. It can also contain the CHAP name (and password), the reverse CHAP name (and password)

/sys/firmware/initrd

Defined on file *sysfs-firmware-initrd*

When the kernel was booted with an initrd and the ”retain_initrd” option is set on the kernel command line, */sys/firmware/initrd* contains the contents of the initrd that the kernel was booted with.

/sys/firmware/lefi/boardinfo

Defined on file *sysfs-firmware-lefi-boardinfo*

Get mainboard and BIOS info easily on the Loongson platform, this is useful to point out the current used mainboard type and BIOS version when there exists problems related with hardware or firmware.

The related structures are already defined in the interface specification about firmware and kernel which are common requirement and specific for Loongson64, so only add a new boardinfo.c file in arch/mips/loongson64.

For example:

[loongson@linux ~]$ cat */sys/firmware/lefi/boardinfo* Board Info Manufacturer : LEMOTE Board Name : LEMOTE-LS3A4000-7A1000-1w-V01-pc Family : LOONGSON3

BIOS Info Vendor : Kunlun Version : Kunlun-A1901-V4.1.3-20200414093938 ROM Size : 4 KB Release Date : 2020-04-14

By the way, using dmidecode command can get the similar info if there exists SMBIOS in firmware, but the fact is that there is no SMBIOS on some machines, we can see nothing when execute dmidecode, like this:

[root@linux loongson]# dmidecode # dmidecode 2.12 # No SMBIOS nor DMI entry point found, sorry.

/sys/firmware/log

Defined on file *sysfs-firmware-log*

The */sys/firmware/log* is a binary file that represents a read-only copy of the firmware's log if one is available.

/sys/firmware/memmap/

Defined on file *sysfs-firmware-memmap*

On all platforms, the firmware provides a memory map which the kernel reads. The resources from that memory map are registered in the kernel resource tree and exposed to userspace via /proc/iomem (together with other resources).

However, on most architectures that firmware-provided memory map is modified afterwards by the kernel itself, either because the kernel merges that memory map with other information or just because the user overwrites that memory map via command line.

kexec needs the raw firmware-provided memory map to setup the parameter segment of the kernel that should be booted with kexec. Also, the raw memory map is useful for debugging. For that reason, /sys/firmware/memmap is an interface that provides the raw memory map to userspace.

The structure is as follows: Under /sys/firmware/memmap there are subdirectories with the number of the entry as their name:

```
/sys/firmware/memmap/0
/sys/firmware/memmap/1
/sys/firmware/memmap/2
/sys/firmware/memmap/3
...
```

The maximum depends on the number of memory map entries provided by the firmware. The order is just the order that the firmware provides.

Each directory contains three files:

start	The start address (as hexadecimal number with the '0x' prefix).
end	The end address, inclusive (regardless whether the firmware provides inclusive or exclusive ranges).
type	Type of the entry as string. See below for a list of valid types.

So, for example:

```
/sys/firmware/memmap/0/start
/sys/firmware/memmap/0/end
/sys/firmware/memmap/0/type
/sys/firmware/memmap/1/start
...
```

Currently following types exist:

- System RAM
- ACPI Tables
- ACPI Non-volatile Storage
- Unusable memory
- Persistent Memory (legacy)
- Persistent Memory
- Soft Reserved
- reserved

Following shell snippet can be used to display that memory map in a human-readable format:

```
#!/bin/bash
cd /sys/firmware/memmap
for dir in * ; do
    start=$(cat $dir/start)
    end=$(cat $dir/end)
    type=$(cat $dir/type)
    printf "%016x-%016x (%s)\n" $start $[ $end +1] "$type"
done
```

/sys/firmware/opal/powercap

Defined on file *sysfs-firmware-opal-powercap*

Powercap directory for Powernv (P8, P9) servers

Each folder in this directory contains a power-cappable component.

/sys/firmware/opal/powercap/system-powercap

Defined on file *sysfs-firmware-opal-powercap*

System powercap directory and attributes applicable for Powernv (P8, P9) servers

This directory provides powercap information. It contains below sysfs attributes:

- powercap-min : This file provides the minimum possible powercap in Watt units
- powercap-max : This file provides the maximum possible powercap in Watt units
- powercap-current : This file provides the current powercap set on the system. Writing to this file creates a request for setting a new-powercap. The powercap requested must be between powercap-min and powercap-max.

/sys/firmware/opal/psr

Defined on file *sysfs-firmware-opal-psr*

Power-Shift-Ratio directory for Powernv P9 servers

Power-Shift-Ratio allows to provide hints the firmware to shift/throttle power between different entities in the system. Each attribute in this directory indicates a settable PSR.

/sys/firmware/opal/psr/cpu_to_gpu_X

Defined on file *sysfs-firmware-opal-psr*

PSR sysfs attributes for Powernv P9 servers

Power-Shift-Ratio between CPU and GPU for a given chip with chip-id X. This file gives the ratio (0-100) which is used by OCC for power-capping.

/sys/firmware/opal/sensor_groups

Defined on file *sysfs-firmware-opal-sensor-groups*

Sensor groups directory for POWER9 powernv servers

Each folder in this directory contains a sensor group which are classified based on type of the sensor like power, temperature, frequency, current, etc. They can also indicate the group of sensors belonging to different owners like CSM, Profiler, Job-Scheduler

/sys/firmware/opal/sensor_groups/<sensor_group_name>/clear

Defined on file *sysfs-firmware-opal-sensor-groups*

Sysfs file to clear the min-max of all the sensors belonging to the group.

Writing 1 to this file will clear the minimum and maximum values of all the sensors in the group. In POWER9, the min-max of a sensor is the historical minimum and maximum value of the sensor cached by OCC.

/sys/firmware/papr/energy_scale_info

Defined on file *sysfs-firmware-papr-energy-scale-info*

Directory hosting a set of platform attributes like energy/frequency on Linux running as a PAPR guest.

Each file in a directory contains a platform attribute hierarchy pertaining to performance/energy-savings mode and processor frequency.

/sys/firmware/papr/energy_scale_info/<id>

Defined on file *sysfs-firmware-papr-energy-scale-info*

Energy, frequency attributes directory for POWERVM servers

/sys/firmware/papr/energy_scale_info/<id>/desc

Defined on file *sysfs-firmware-papr-energy-scale-info*

String description of the energy attribute of <id>

/sys/firmware/papr/energy_scale_info/<id>/value

Defined on file *sysfs-firmware-papr-energy-scale-info*

Numeric value of the energy attribute of <id>

/sys/firmware/papr/energy_scale_info/<id>/value_desc

Defined on file *sysfs-firmware-papr-energy-scale-info*

String value of the energy attribute of <id>

/sys/firmware/qemu_fw_cfg/

Defined on file *sysfs-firmware-qemu_fw_cfg*

Several different architectures supported by QEMU (x86, arm, sun4*, ppc/mac) are provisioned with a firmware configuration (fw_cfg) device, originally intended as a way for the host to provide configuration data to the guest firmware. Starting with QEMU v2.4, arbitrary fw_cfg file entries may be specified by the user on the command line, which makes fw_cfg additionally useful as an out-of-band, asynchronous mechanism for providing configuration data to the guest userspace.

The authoritative guest-side hardware interface documentation to the fw_cfg device can be found in "docs/specs/fw_cfg.rst" in the QEMU source tree, or online at: https://qemu-project.gitlab.io/qemu/specs/fw_cfg.html

SysFS fw_cfg Interface

The fw_cfg sysfs interface described in this document is only intended to display discoverable blobs (i.e., those registered with the file directory), as there is no way to determine the presence or size of "legacy" blobs (with selector keys between 0x0002 and 0x0018) programmatically.

All fw_cfg information is shown under:

/sys/firmware/qemu_fw_cfg/

The only legacy blob displayed is the fw_cfg device revision:

/sys/firmware/qemu_fw_cfg/rev

Discoverable fw_cfg blobs by selector key

All discoverable blobs listed in the fw_cfg file directory are displayed as entries named after their unique selector key value, e.g.:

/sys/firmware/qemu_fw_cfg/by_key/32 /sys/firmware/qemu_fw_cfg/by_key/33 /sys/firmware/qemu_fw_cfg/by_key/34 ...

Each such fw_cfg sysfs entry has the following values exported as attributes:

name	The 56-byte nul-terminated ASCII string used as the blob's 'file name' in the fw_cfg directory.
size	The length of the blob, as given in the fw_cfg directory.
key	The value of the blob's selector key as given in the fw_cfg directory. This value is the same as used in the parent directory name.
raw	The raw bytes of the blob, obtained by selecting the entry via the control register, and reading a number of bytes equal to the blob size from the data register.

Listing fw_cfg blobs by file name

While the fw_cfg device does not impose any specific naming convention on the blobs registered in the file directory, QEMU developers have traditionally used path name semantics to give each blob a descriptive name. For example:

```
"bootorder"
"genroms/kvmvapic.bin"
"etc/e820"
"etc/boot-fail-wait"
"etc/system-states"
"etc/table-loader"
"etc/acpi/rsdp"
"etc/acpi/tables"
"etc/smbios/smbios-tables"
"etc/smbios/smbios-anchor"
...
```

In addition to the listing by unique selector key described above, the fw_cfg sysfs driver also attempts to build a tree of directories matching the path name components of fw_cfg blob names, ending in symlinks to the by_key entry for each ”basename”, as illustrated below (assume current directory is /sys/firmware):

```
qemu_fw_cfg/by_name/bootorder -> ../by_key/38
qemu_fw_cfg/by_name/etc/e820 -> ../../by_key/35
qemu_fw_cfg/by_name/etc/acpi/rsdp -> ../../../by_key/41
...
```

Construction of the directory tree and symlinks is done on a ”best-effort” basis, as there is no guarantee that components of fw_cfg blob names are always ”well behaved”. I.e., there is the possibility that a symlink (basename) will conflict with a dirname component of another fw_cfg blob, in which case the creation of the offending /sys/firmware/qemu_fw_cfg/by_name entry will be skipped.

The authoritative list of entries will continue to be found under the /sys/firmware/qemu_fw_cfg/by_key directory.

/sys/firmware/secvar

Defined on file *sysfs-secvar*

This directory is created if the POWER firmware supports OS secureboot, thereby secure variables. It exposes interface for reading/writing the secure variables

/sys/firmware/secvar/config

Defined on file *sysfs-secvar*

This optional directory contains read-only config attributes as defined by the secure variable implementation. All data is in ASCII format. The directory is only created if the backing implementation provides variables to populate it, which at present is only PLPKS on the pseries platform.

/sys/firmware/secvar/config/max_object_size

Defined on file *sysfs-secvar*

Maximum allowed size of objects in the keystore in bytes, represented in ASCII decimal format.

This is not necessarily the same as the max size that can be written to an update file as writes can contain more than object data, you should use the size of the update file for that purpose.

Currently only provided by PLPKS on the pseries platform.

/sys/firmware/secvar/config/signed_update_algorithms

Defined on file *sysfs-secvar*

Bitmask of flags indicating which algorithms the hypervisor supports for signed update of objects, represented as a 16 byte hexadecimal ASCII string. Consult the hypervisor documentation for what these flags mean.

Currently only provided by PLPKS on the pseries platform.

/sys/firmware/secvar/config/supported_policies

Defined on file *sysfs-secvar*

Bitmask of supported policy flags by the hypervisor, represented as an 8 byte hexadecimal ASCII string. Consult the hypervisor documentation for what these flags are.

Currently only provided by PLPKS on the pseries platform.

/sys/firmware/secvar/config/total_size

Defined on file *sysfs-secvar*

Total size of the PLPKS in bytes, represented in ASCII decimal format.

Currently only provided by PLPKS on the pseries platform.

/sys/firmware/secvar/config/used_space

Defined on file *sysfs-secvar*

Current space consumed by the key store, in bytes, represented in ASCII decimal format.

Currently only provided by PLPKS on the pseries platform.

/sys/firmware/secvar/config/version

Defined on file *sysfs-secvar*

Config version as reported by the hypervisor in ASCII decimal format.

Currently only provided by PLPKS on the pseries platform.

/sys/firmware/secvar/format

Defined on file *sysfs-secvar*

A string indicating which backend is in use by the firmware. This determines the format of the variable and the accepted format of variable updates.

On powernv/OPAL, this value is provided by the OPAL firmware and is expected to be "ibm,edk2-compat-v1".

On pseries/PLPKS, this is generated by the kernel based on the version number in the SB_VERSION variable in the keystore, and has the form "ibm,plpks-sb-v<version>", or "ibm,plpks-sb-unknown" if there is no SB_VERSION variable.

/sys/firmware/secvar/vars

Defined on file *sysfs-secvar*

This directory lists all the secure variables that are supported by the firmware.

/sys/firmware/secvar/vars/<variable name>

Defined on file *sysfs-secvar*

Each secure variable is represented as a directory named as <variable_name>. The variable name is unique and is in ASCII representation. The data and size can be determined by reading their respective attribute files.

/sys/firmware/secvar/vars/<variable_name>/data

Defined on file *sysfs-secvar*

A read-only file containing the value of the variable. The size of the file represents the maximum size of the variable data.

/sys/firmware/secvar/vars/<variable_name>/size

Defined on file *sysfs-secvar*

An integer representation of the size of the content of the variable. In other words, it represents the size of the data.

/sys/firmware/secvar/vars/<variable_name>/update

Defined on file *sysfs-secvar*

A write-only file that is used to submit the new value for the variable. The size of the file represents the maximum size of the variable data that can be written.

/sys/firmware/sgi_uv/

Defined on file *sysfs-firmware-sgi_uv*

The /sys/firmware/sgi_uv directory contains information about the UV platform.

Under that directory are a number of read-only attributes:

```
archtype
hub_type
hubless
partition_id
coherence_id
uv_type
```

The archtype entry contains the UV architecture type that is used to select arch-dependent addresses and features. It can be set via the OEM_ID in the ACPI MADT table or by UVsystab entry both passed from UV BIOS.

The hub_type entry is used to select the type of hub which is similar to uv_type but encoded in a binary format. Include the file uv_hub.h to get the definitions.

The hubless entry basically is present and set only if there is no hub. In this case the hub_type entry is not present.

The partition_id entry contains the partition id. UV systems can be partitioned into multiple physical machines, which each partition running a unique copy of the operating system. Each partition will have a unique partition id.

The coherence_id entry contains the coherence id. A partitioned UV system can have one or more coherence domains. The coherence id indicates which coherence domain this partition is in.

The uv_type entry contains the hub revision number. This value can be used to identify the UV system version:

```
"0.*" = Hubless UV ('*' is subtype)
"3.0" = UV2
"5.0" = UV3
"7.0" = UV4
"7.1" = UV4a
"9.0" = UV5
```

The /sys/firmware/sgi_uv directory also contains two directories:

```
hubs/
pcibuses/
```

The hubs directory contains a number of hub objects, each representing a UV Hub visible to the BIOS. Each hub object's name is appended by a unique ordinal value (ex. /sys/firmware/sgi_uv/hubs/hub_5)

Each hub object directory contains a number of read-only attributes:

```
cnode
location
name
nasid
shared
this_partition
```

The cnode entry contains the cnode number of the corresponding hub. If a cnode value is not applicable, the value returned will be -1.

The location entry contains the location string of the corresponding hub. This value is used to physically identify a hub within a system.

The name entry contains the name of the corresponding hub. This name can be two variants:

```
"UVHub x.x" = A 'node' ASIC, connecting a CPU to the interconnect
fabric. The 'x.x' value represents the ASIC revision.
(ex. 'UVHub 5.0')

"NLxRouter" = A 'router ASIC, only connecting other ASICs to
the interconnect fabric. The 'x' value representing
the fabric technology version. (ex. 'NL8Router')
```

The nasid entry contains the nasid number of the corresponding hub. If a nasid value is not applicable, the value returned will be -1.

The shared entry contains a boolean value describing whether the corresponding hub is shared between system partitions.

The this_partition entry contains a boolean value describing whether the corresponding hub is local to the current partition.

Each hub object directory also contains a number of port objects, each representing a fabric port on the corresponding hub. A port object's name is appended by a unique ordinal value (ex. /sys/firmware/sgi_uv/hubs/hub_5/port_3)

Each port object directory contains a number of read-only attributes:

```
conn_hub
conn_port
```

The conn_hub entry contains a value representing the unique ordinal value of the hub on the other end of the fabric cable plugged into the port. If the port is disconnected, the value returned will be -1.

The conn_port entry contains a value representing the unique ordinal value of the port on the other end of the fabric cable plugged into the port. If the port is disconnected, the value returned will be -1.

Ex:

A value of '3' is read from:
/sys/firmware/sgi_uv/hubs/hub_5/port_3/conn_hub

and a value of '6' is read from:
/sys/firmware/sgi_uv/hubs/hub_5/port_3/conn_port

representing that this port is connected to:
/sys/firmware/sgi_uv/hubs/hub_3/port_6

The pcibuses directory contains a number of PCI bus objects. Each PCI bus object's name is appended by its PCI bus address. (ex. pcibus_0003:80)

Each pcibus object has a number of possible read-only attributes:

```
type
location
slot
ppb_addr
iio_stack
```

The type entry contains a value describing the type of IO at the corresponding PCI bus address. Known possible values across all UV versions are:

```
BASE IO
PCIe IO
PCIe SLOT
NODE IO
Riser
PPB
```

The location entry contains the location string of the UV Hub of the CPU physically connected to the corresponding PCI bus.

The slot entry contains the physical slot number of the corresponding PCI bus. This value is used to physically locate PCI cards within a system.

The ppb_addr entry contains the PCI address string of the bridged PCI bus. This entry is only present when the PCI bus object type is 'PPB'.

The iio_stack entry contains a value describing the IIO stack number that the corresponding PCI bus object is connected to.

/sys/firmware/turris-mox-rwtm/board_version

Defined on file *sysfs-firmware-turris-mox-rwtm*

(Read) Board version burned into eFuses of this Turris Mox board. Format: %i

/sys/firmware/turris-mox-rwtm/mac_address*

Defined on file *sysfs-firmware-turris-mox-rwtm*

(Read) MAC addresses burned into eFuses of this Turris Mox board. Format: %pM

/sys/firmware/turris-mox-rwtm/pubkey

Defined on file *sysfs-firmware-turris-mox-rwtm*

(Read) ECDSA public key (in pubkey hex compressed form) computed as pair to the ECDSA private key burned into eFuses of this Turris Mox Board. Format: string

/sys/firmware/turris-mox-rwtm/ram_size

Defined on file *sysfs-firmware-turris-mox-rwtm*

(Read) RAM size in MiB of this Turris Mox board as was detected during manufacturing and burned into eFuses. Can be 512 or 1024. Format: %i

/sys/firmware/turris-mox-rwtm/serial_number

Defined on file *sysfs-firmware-turris-mox-rwtm*

(Read) Serial number burned into eFuses of this Turris Mox device. Format: %016X

5.2.36 Symbols under /sys/fs

/sys/fs/erofs/<disk>/sync_decompress

Defined on file *sysfs-fs-erofs*

Control strategy of sync decompression:

- 0 (default, auto): enable for readpage, and enable for readahead on atomic contexts only.
- 1 (force on): enable for readpage and readahead.
- 2 (force off): disable for all situations.

/sys/fs/erofs/features/

Defined on file *sysfs-fs-erofs*

Shows all enabled kernel features. Supported features: zero_padding, compr_cfgs, big_pcluster, chunked_file, device_table, compr_head2, sb_chksum, ztailpacking, dedupe, fragments.

/sys/fs/ext4/<disk>/delayed_allocation_blocks

Defined on file *sysfs-fs-ext4*

This file is read-only and shows the number of blocks that are dirty in the page cache, but which do not have their location in the filesystem allocated yet.

/sys/fs/ext4/<disk>/extent_max_zeroout_kb

Defined on file *sysfs-fs-ext4*

The maximum number of kilobytes which will be zeroed out in preference to creating a new uninitialized extent when manipulating an inode's extent tree. Note that using a larger value

will increase the variability of time necessary to complete a random write operation (since a 4k random write might turn into a much larger write due to the zeroout operation).

/sys/fs/ext4/<disk>/inode_goal

Defined on file *sysfs-fs-ext4*

Tuning parameter which (if non-zero) controls the goal inode used by the inode allocator in preference to all other allocation heuristics. This is intended for debugging use only, and should be 0 on production systems.

/sys/fs/ext4/<disk>/inode_readahead_blks

Defined on file *sysfs-fs-ext4*

Tuning parameter which controls the maximum number of inode table blocks that ext4's inode table readahead algorithm will pre-read into the buffer cache

/sys/fs/ext4/<disk>/journal_task

Defined on file *sysfs-fs-ext4*

This file is read-only and shows the pid of journal thread in current pid-namespace or 0 if task is unreachable.

/sys/fs/ext4/<disk>/lifetime_write_kbytes

Defined on file *sysfs-fs-ext4*

This file is read-only and shows the number of kilobytes of data that have been written to this filesystem since it was created.

/sys/fs/ext4/<disk>/max_writeback_mb_bump

Defined on file *sysfs-fs-ext4*

The maximum number of megabytes the writeback code will try to write out before move on to another inode.

/sys/fs/ext4/<disk>/mb_group_prealloc

Defined on file *sysfs-fs-ext4*

The multiblock allocator will round up allocation requests to a multiple of this tuning parameter if the stripe size is not set in the ext4 superblock

/sys/fs/ext4/<disk>/mb_max_to_scan

Defined on file *sysfs-fs-ext4*

The maximum number of extents the multiblock allocator will search to find the best extent

/sys/fs/ext4/<disk>/mb_min_to_scan

Defined on file *sysfs-fs-ext4*

The minimum number of extents the multiblock allocator will search to find the best extent

/sys/fs/ext4/<disk>/mb_order2_req

Defined on file *sysfs-fs-ext4*

Tuning parameter which controls the minimum size for requests (as a power of 2) where the buddy cache is used

/sys/fs/ext4/<disk>/mb_stats

Defined on file *sysfs-fs-ext4*

Controls whether the multiblock allocator should collect statistics, which are shown during the unmount. 1 means to collect statistics, 0 means not to collect statistics

/sys/fs/ext4/<disk>/mb_stream_req

Defined on file *sysfs-fs-ext4*

Files which have fewer blocks than this tunable parameter will have their blocks allocated out of a block group specific preallocation pool, so that small files are packed closely together. Each large file will have its blocks allocated out of its own unique preallocation pool.

/sys/fs/ext4/<disk>/session_write_kbytes

Defined on file *sysfs-fs-ext4*

This file is read-only and shows the number of kilobytes of data that have been written to this filesystem since it was mounted.

/sys/fs/f2fs/<disk>/atgc_age_threshold

Defined on file *sysfs-fs-f2fs*

When ATGC is on, it controls age threshold to bypass GCing young candidates whose age is not beyond the threshold, by default it was initialized as 604800 seconds (equals to 7 days).

/sys/fs/f2fs/<disk>/atgc_age_weight

Defined on file *sysfs-fs-f2fs*

When ATGC is on, it controls age weight to balance weight proportion in between aging and valid blocks, the value should be in range of [0, 100], by default it was initialized as 60(%).

/sys/fs/f2fs/<disk>/atgc_candidate_count

Defined on file *sysfs-fs-f2fs*

When ATGC is on, it controls candidate count in order to limit total number of potential victim in all candidates, by default it was initialized as 10 (sections).

/sys/fs/f2fs/<disk>/atgc_candidate_ratio

Defined on file *sysfs-fs-f2fs*

When ATGC is on, it controls candidate ratio in order to limit total number of potential victim in all candidates, the value should be in range of [0, 100], by default it was initialized as 20(%).

/sys/fs/f2fs/<disk>/avg_vblocks

Defined on file *sysfs-fs-f2fs*

Average number of valid blocks. Available when CONFIG_F2FS_STAT_FS=y.

/sys/fs/f2fs/<disk>/ckpt_thread_ioprio

Defined on file *sysfs-fs-f2fs*

Give a way to change checkpoint merge daemon's io priority. Its default value is "be,3", which means "BE" I/O class and I/O priority "3". We can select the class between "rt" and "be", and set the I/O priority within valid range of it. "," delimiter is necessary in between I/O class and priority number.

/sys/fs/f2fs/<disk>/committed_atomic_block

Defined on file *sysfs-fs-f2fs*

Show the accumulated total committed atomic write block count after boot. If you write "0" here, you can initialize to "0".

/sys/fs/f2fs/<disk>/compr_new_inode

Defined on file *sysfs-fs-f2fs*

Show the count of inode newly enabled for compression since mount. Note that when the compression is disabled for the files, this count doesn't decrease. If you write "0" here, you can initialize compr_new_inode to "0".

/sys/fs/f2fs/<disk>/compr_saved_block

Defined on file *sysfs-fs-f2fs*

Show the saved block count with compression since mount. Note that when the compressed blocks are deleted, this count doesn't decrease. If you write "0" here, you can initialize compr_written_block and compr_saved_block to "0".

/sys/fs/f2fs/<disk>/compr_written_block

Defined on file *sysfs-fs-f2fs*

Show the block count written after compression since mount. Note that when the compressed blocks are deleted, this count doesn't decrease. If you write "0" here, you can initialize compr_written_block and compr_saved_block to "0".

/sys/fs/f2fs/<disk>/compress_percent

Defined on file *sysfs-fs-f2fs*

When compress cache is on, it controls cached page percent(compress pages / free_ram) in order to limit caching compress page. If cached page percent exceed threshold, then deny caching compress page. The value should be in range of (0, 100], by default it was initialized as 20(%).

/sys/fs/f2fs/<disk>/compress_watermark

Defined on file *sysfs-fs-f2fs*

When compress cache is on, it controls free memory watermark in order to limit caching compress page. If free memory is lower than watermark, then deny caching compress page. The value should be in range of (0, 100], by default it was initialized as 20(%).

/sys/fs/f2fs/<disk>/cp_background_calls

Defined on file *sysfs-fs-f2fs*

Number of checkpoint operations performed in the background to free segments. Available when CONFIG_F2FS_STAT_FS=y.

/sys/fs/f2fs/<disk>/cp_foreground_calls

Defined on file *sysfs-fs-f2fs*

Number of checkpoint operations performed on demand. Available when CONFIG_F2FS_STAT_FS=y.

/sys/fs/f2fs/<disk>/cp_interval

Defined on file *sysfs-fs-f2fs*

Controls the checkpoint timing, set to 60 seconds by default.

/sys/fs/f2fs/<disk>/current_atomic_write

Defined on file *sysfs-fs-f2fs*

Show the total current atomic write block count, which is not committed yet. This is a read-only entry.

/sys/fs/f2fs/<disk>/current_reserved_blocks

Defined on file *sysfs-fs-f2fs*

Shows current reserved blocks in system, it may be temporarily smaller than target_reserved_blocks, but will gradually increase to target_reserved_blocks when more free blocks are freed by user later.

/sys/fs/f2fs/<disk>/data_io_flag

Defined on file *sysfs-fs-f2fs*

Give a way to attach REQ_META|FUA to data writes given temperature-based bits. Now the bits indicate:

REQ_META			REQ_FUA		
5	4	3	2	1	0
Cold	Warm	Hot	Cold	Warm	Hot

/sys/fs/f2fs/<disk>/dir_level

Defined on file *sysfs-fs-f2fs*

Controls the directory level for large directory. If a directory has a number of files, it can reduce the file lookup latency by increasing this dir_level value. Otherwise, it needs to decrease this value to reduce the space overhead. The default value is 0.

/sys/fs/f2fs/<disk>/dirty_nats_ratio

Defined on file *sysfs-fs-f2fs*

Controls dirty nat entries ratio threshold, if current ratio exceeds configured threshold, checkpoint will be triggered for flushing dirty nat entries.

/sys/fs/f2fs/<disk>/dirty_segments

Defined on file *sysfs-fs-f2fs*

Shows the number of dirty segments.

/sys/fs/f2fs/<disk>/discard_granularity

Defined on file *sysfs-fs-f2fs*

Controls discard granularity of inner discard thread. Inner thread will not issue discards with size that is smaller than granularity. The unit size is one block(4KB), now only support configuring in range of [1, 512]. Default value is 16. For small devices, default value is 1.

/sys/fs/f2fs/<disk>/discard_idle_interval

Defined on file *sysfs-fs-f2fs*

Controls the idle timing of discard thread given this time interval. Default is 5 secs.

/sys/fs/f2fs/<disk>/discard_io_aware

Defined on file *sysfs-fs-f2fs*

It controls to enable/disable IO aware feature for background discard. By default, the value is 1 which indicates IO aware is on.

/sys/fs/f2fs/<disk>/discard_io_aware_gran

Defined on file *sysfs-fs-f2fs*

Controls background discard granularity of inner discard thread when is not in idle. Inner thread will not issue discards with size that is smaller than granularity. The unit size is one block(4KB), now only support configuring in range of [0, 512]. Default: 512

/sys/fs/f2fs/<disk>/discard_urgent_util

Defined on file *sysfs-fs-f2fs*

When space utilization exceeds this, do background DISCARD aggressively. Does DISCARD forcibly in a period of given min_discard_issue_time when the number of discards is not 0 and set discard granularity to 1. Default: 80

/sys/fs/f2fs/<disk>/encoding

Defined on file *sysfs-fs-f2fs*

Displays name and version of the encoding set for the filesystem. If no encoding is set, displays (none)

/sys/fs/f2fs/<disk>/extension_list

Defined on file *sysfs-fs-f2fs*

Used to control configure extension list: - Query: cat */sys/fs/f2fs/<disk>/extension_list* - Add: echo '[h/c]extension' > */sys/fs/f2fs/<disk>/extension_list* - Del: echo '[h/c]!extension' > */sys/fs/f2fs/<disk>/extension_list* - [h] means add/del hot file extension - [c] means add/del cold file extension

/sys/fs/f2fs/<disk>/fault_rate

Defined on file *sysfs-fs-f2fs*

Enable fault injection in all supported types with specified injection rate.

/sys/fs/f2fs/<disk>/fault_type

Defined on file *sysfs-fs-f2fs*

Support configuring fault injection type, should be enabled with fault_injection option, fault type value is shown below, it supports single or combined type.

Type_Name	Type_Value
FAULT_KMALLOC	0x000000001
FAULT_KVMALLOC	0x000000002
FAULT_PAGE_ALLOC	0x000000004
FAULT_PAGE_GET	0x000000008
FAULT_ALLOC_BIO	0x000000010 (obsolete)
FAULT_ALLOC_NID	0x000000020
FAULT_ORPHAN	0x000000040
FAULT_BLOCK	0x000000080
FAULT_DIR_DEPTH	0x000000100
FAULT_EVICT_INODE	0x000000200
FAULT_TRUNCATE	0x000000400
FAULT_READ_IO	0x000000800
FAULT_CHECKPOINT	0x000001000
FAULT_DISCARD	0x000002000
FAULT_WRITE_IO	0x000004000
FAULT_SLAB_ALLOC	0x000008000
FAULT_DQUOT_INIT	0x000010000
FAULT_LOCK_OP	0x000020000
FAULT_BLKADDR	0x000040000

/sys/fs/f2fs/<disk>/feature_list/

Defined on file *sysfs-fs-f2fs*

Expand */sys/fs/f2fs/<disk>/features* to meet sysfs rule. Supported on-disk features: encryption, block_zoned (aka blkzoned), extra_attr, project_quota (aka projquota), inode_checksum, flexible_inline_xattr, quota_ino, inode_crtime, lost_found, verity, sb_checksum, casefold, readonly, compression. Note that, pin_file is moved into */sys/fs/f2fs/features/*.

/sys/fs/f2fs/<disk>/features

Defined on file *sysfs-fs-f2fs*

<deprecated: should use /sys/fs/f2fs/<disk>/feature_list/> Shows all enabled features in current device. Supported features: encryption, blkzoned, extra_attr, projquota, inode_checksum, flexible_inline_xattr, quota_ino, inode_crtime, lost_found, verity, sb_checksum, casefold, readonly, compression, pin_file.

/sys/fs/f2fs/<disk>/free_segments

Defined on file *sysfs-fs-f2fs*

Number of free segments in disk.

/sys/fs/f2fs/<disk>/gc_background_calls

Defined on file *sysfs-fs-f2fs*

Number of garbage collection operations triggered in background. Available when CONFIG_F2FS_STAT_FS=y.

/sys/fs/f2fs/<disk>/gc_foreground_calls

Defined on file *sysfs-fs-f2fs*

Number of garbage collection operations performed on demand. Available when CONFIG_F2FS_STAT_FS=y.

/sys/fs/f2fs/<disk>/gc_idle

Defined on file *sysfs-fs-f2fs*

Controls the victim selection policy for garbage collection. Setting gc_idle = 0(default) will disable this option. Setting:

gc_idle = 1	will select the Cost Benefit approach & setting
gc_idle = 2	will select the greedy approach & setting
gc_idle = 3	will select the age-threshold based approach.

/sys/fs/f2fs/<disk>/gc_idle_interval

Defined on file *sysfs-fs-f2fs*

Controls the idle timing for gc path. Set to 5 seconds by default.

/sys/fs/f2fs/<disk>/gc_max_sleep_time

Defined on file *sysfs-fs-f2fs*

Controls the maximum sleep time for gc_thread. Time is in milliseconds.

/sys/fs/f2fs/<disk>/gc_min_sleep_time

Defined on file *sysfs-fs-f2fs*

Controls the minimum sleep time for gc_thread. Time is in milliseconds.

/sys/fs/f2fs/<disk>/gc_mode

Defined on file *sysfs-fs-f2fs*

Show the current gc_mode as a string. This is a read-only entry.

/sys/fs/f2fs/<disk>/gc_no_gc_sleep_time

Defined on file *sysfs-fs-f2fs*

Controls the default sleep time for gc_thread. Time is in milliseconds.

/sys/fs/f2fs/<disk>/gc_pin_file_thresh

Defined on file *sysfs-fs-f2fs*

This indicates how many GC can be failed for the pinned file. If it exceeds this, F2FS doesn't guarantee its pinning state. 2048 trials is set by default.

/sys/fs/f2fs/<disk>/gc_reclaimed_segments

Defined on file *sysfs-fs-f2fs*

Show how many segments have been reclaimed by GC during a specific GC mode (0: GC normal, 1: GC idle CB, 2: GC idle greedy, 3: GC idle AT, 4: GC urgent high, 5: GC urgent low 6: GC urgent mid) You can re-initialize this value to "0".

/sys/fs/f2fs/<disk>/gc_remaining_trials

Defined on file *sysfs-fs-f2fs*

You can set the trial count limit for GC urgent and idle mode with this value. If GC thread gets to the limit, the mode will turn back to GC normal mode. By default, the value is zero, which means there is no limit like before.

/sys/fs/f2fs/<disk>/gc_segment_mode

Defined on file *sysfs-fs-f2fs*

You can control for which gc mode the "gc_reclaimed_segments" node shows. Refer to the description of the modes in "gc_reclaimed_segments".

/sys/fs/f2fs/<disk>/gc_urgent

Defined on file *sysfs-fs-f2fs*

Do background GC aggressively when set. Set to 0 by default. gc urgent high(1): does GC forcibly in a period of given gc_urgent_sleep_time and ignores I/O idling check. uses greedy GC approach and turns SSR mode on. gc urgent low(2): lowers the bar of checking I/O idling in order to process outstanding discard commands and GC a little bit aggressively. uses cost benefit GC approach. gc urgent mid(3): does GC forcibly in a period of given gc_urgent_sleep_time and executes a mid level of I/O idling check. uses cost benefit GC approach.

/sys/fs/f2fs/<disk>/gc_urgent_sleep_time

Defined on file *sysfs-fs-f2fs*

Controls sleep time of GC urgent mode. Set to 500ms by default.

/sys/fs/f2fs/<disk>/hot_data_age_threshold

Defined on file *sysfs-fs-f2fs*

When DATA SEPARATION is on, it controls the age threshold to indicate the data blocks as hot. By default it was initialized as 262144 blocks (equals to 1GB).

/sys/fs/f2fs/<disk>/idle_interval

Defined on file *sysfs-fs-f2fs*

Controls the idle timing of system, if there is no FS operation during given interval. Set to 5 seconds by default.

/sys/fs/f2fs/<disk>/inject_rate

Defined on file *sysfs-fs-f2fs*

Controls the injection rate of arbitrary faults.

/sys/fs/f2fs/<disk>/inject_type

Defined on file *sysfs-fs-f2fs*

Controls the injection type of arbitrary faults.

/sys/fs/f2fs/<disk>/iostat_enable

Defined on file *sysfs-fs-f2fs*

Controls to enable/disable IO stat.

/sys/fs/f2fs/<disk>/iostat_period_ms

Defined on file *sysfs-fs-f2fs*

Give a way to change iostat_period time. 3secs by default. The new iostat trace gives stats gap given the period.

/sys/fs/f2fs/<disk>/ipu_policy

Defined on file *sysfs-fs-f2fs*

Controls the in-place-update policy. updates in f2fs. User can set:

value	policy	description
0x00	DISABLE	disable IPU(=default option in LFS mode)
0x01	FORCE	all the time
0x02	SSR	if SSR mode is activated
0x04	UTIL	if FS utilization is over threshold
0x08	SSR_UTIL	if SSR mode is activated and FS utilization is over threshold
0x10	FSYNC	activated in fsync path only for high performance flash storages. IPU will be triggered only if the # of dirty pages over min_fsync_blocks. (=default option)
0x20	ASYNC	do IPU given by asynchronous write requests
0x40	NOCACHE	disable IPU bio cache
0x80	HONOR_OPU_WRITE	use OPU write prior to IPU write if inode has FI_OPU_WRITE flag

Refer segment.h for details.

/sys/fs/f2fs/<disk>/last_age_weight

Defined on file *sysfs-fs-f2fs*

When DATA SEPARATION is on, it controls the weight of last data block age.

/sys/fs/f2fs/<disk>/lifetime_write_kbytes

Defined on file *sysfs-fs-f2fs*

Shows total written kbytes issued to disk.

/sys/fs/f2fs/<disk>/main_blkaddr

Defined on file *sysfs-fs-f2fs*

Shows first block address of MAIN area.

/sys/fs/f2fs/<disk>/max_discard_issue_time

Defined on file *sysfs-fs-f2fs*

Controls the interval the discard thread will wait when there are no discard operations to be issued.

/sys/fs/f2fs/<disk>/max_discard_request

Defined on file *sysfs-fs-f2fs*

Controls the number of discards a thread will issue at a time. Higher number will allow the discard thread to finish its work faster, at the cost of higher latency for incoming I/O.

/sys/fs/f2fs/<disk>/max_fragment_chunk

Defined on file *sysfs-fs-f2fs*

With "mode=fragment:block" mount options, we can scatter block allocation. f2fs will allocate 1..<max_fragment_chunk> blocks in a chunk and make a hole in the length of 1..<max_fragment_hole> by turns. This value can be set between 1..512 and the default value is 4.

/sys/fs/f2fs/<disk>/max_fragment_hole

Defined on file *sysfs-fs-f2fs*

With "mode=fragment:block" mount options, we can scatter block allocation. f2fs will allocate 1..<max_fragment_chunk> blocks in a chunk and make a hole in the length of 1..<max_fragment_hole> by turns. This value can be set between 1..512 and the default value is 4.

/sys/fs/f2fs/<disk>/max_io_bytes

Defined on file *sysfs-fs-f2fs*

This gives a control to limit the bio size in f2fs. Default is zero, which will follow underlying block layer limit, whereas, if it has a certain bytes value, f2fs won't submit a bio larger than that size.

/sys/fs/f2fs/<disk>/max_ordered_discard

Defined on file *sysfs-fs-f2fs*

Controls the maximum ordered discard, the unit size is one block(4KB). Set it to 16 by default.

/sys/fs/f2fs/<disk>/max_roll_forward_node_blocks

Defined on file *sysfs-fs-f2fs*

Controls max # of node block writes to be used for roll forward recovery. This can limit the roll forward recovery time.

/sys/fs/f2fs/<disk>/max_small_discards

Defined on file *sysfs-fs-f2fs*

Controls the issue rate of discard commands that consist of small blocks less than 2MB. The candidates to be discarded are cached during checkpoint, and issued by issue_discard thread after checkpoint. It is enabled by default.

/sys/fs/f2fs/<disk>/max_victim_search

Defined on file *sysfs-fs-f2fs*

Controls the number of trials to find a victim segment when conducting SSR and cleaning operations. The default value is 4096 which covers 8GB block address range.

/sys/fs/f2fs/<disk>/mid_discard_issue_time

Defined on file *sysfs-fs-f2fs*

Controls the interval the discard thread will wait between issuing discard requests when there are discards to be issued and an I/O aware interruption occurs.

/sys/fs/f2fs/<disk>/migration_granularity

Defined on file *sysfs-fs-f2fs*

Controls migration granularity of garbage collection on large section, it can let GC move partial segment{s} of one section in one GC cycle, so that dispersing heavy overhead GC to multiple lightweight one.

/sys/fs/f2fs/<disk>/min_discard_issue_time

Defined on file *sysfs-fs-f2fs*

Controls the interval the discard thread will wait between issuing discard requests when there are discards to be issued and no I/O aware interruptions occur.

/sys/fs/f2fs/<disk>/min_fsync_blocks

Defined on file *sysfs-fs-f2fs*

Controls the dirty page count condition for the in-place-update policies.

/sys/fs/f2fs/<disk>/min_hot_blocks

Defined on file *sysfs-fs-f2fs*

Controls the dirty page count condition for redefining hot data.

/sys/fs/f2fs/<disk>/min_ipu_util

Defined on file *sysfs-fs-f2fs*

Controls the FS utilization condition for the in-place-update policies. It is used by F2FS_IPU_UTIL and F2FS_IPU_SSR_UTIL policies.

/sys/fs/f2fs/<disk>/min_seq_blocks

Defined on file *sysfs-fs-f2fs*

Controls the dirty page count condition for batched sequential writes in writepages.

/sys/fs/f2fs/<disk>/min_ssr_sections

Defined on file *sysfs-fs-f2fs*

Controls the free section threshold to trigger SSR allocation. If this is large, SSR mode will be enabled early.

/sys/fs/f2fs/<disk>/mounted_time_sec

Defined on file *sysfs-fs-f2fs*

Show the mounted time in secs of this partition.

/sys/fs/f2fs/<disk>/moved_blocks_background

Defined on file *sysfs-fs-f2fs*

Number of blocks moved by garbage collection in background. Available when CONFIG_F2FS_STAT_FS=y.

/sys/fs/f2fs/<disk>/moved_blocks_foreground

Defined on file *sysfs-fs-f2fs*

Number of blocks moved by garbage collection in foreground. Available when CONFIG_F2FS_STAT_FS=y.

/sys/fs/f2fs/<disk>/node_io_flag

Defined on file *sysfs-fs-f2fs*

Give a way to attach REQ_META|FUA to node writes given temperature-based bits. Now the bits indicate:

REQ_META			REQ_FUA		
5	4	3	2	1	0
Cold	Warm	Hot	Cold	Warm	Hot

/sys/fs/f2fs/<disk>/ovp_segments

Defined on file *sysfs-fs-f2fs*

Shows the number of overprovision segments.

/sys/fs/f2fs/<disk>/peak_atomic_write

Defined on file *sysfs-fs-f2fs*

Show the peak value of total current atomic write block count after boot. If you write "0" here, you can initialize to "0".

/sys/fs/f2fs/<disk>/pending_discard

Defined on file *sysfs-fs-f2fs*

Shows the number of pending discard commands in the queue.

/sys/fs/f2fs/<disk>/ra_nid_pages

Defined on file *sysfs-fs-f2fs*

Controls the count of nid pages to be readaheaded. When building free nids, F2FS reads NAT blocks ahead for speed up. Default is 0.

/sys/fs/f2fs/<disk>/ram_thresh

Defined on file *sysfs-fs-f2fs*

Controls the memory footprint used by free nids and cached nat entries. By default, 1 is set, which indicates 10 MB / 1 GB RAM.

/sys/fs/f2fs/<disk>/readdir_ra

Defined on file *sysfs-fs-f2fs*

Controls readahead inode block in readdir. Enabled by default.

/sys/fs/f2fs/<disk>/reclaim_segments

Defined on file *sysfs-fs-f2fs*

This parameter controls the number of prefree segments to be reclaimed. If the number of prefree segments is larger than the number of segments in the proportion to the percentage over total volume size, f2fs tries to conduct checkpoint to reclaim the prefree segments to free segments. By default, 5% over total # of segments.

/sys/fs/f2fs/<disk>/reserved_blocks

Defined on file *sysfs-fs-f2fs*

Controls target reserved blocks in system, the threshold is soft, it could exceed current available user space.

/sys/fs/f2fs/<disk>/revoked_atomic_block

Defined on file *sysfs-fs-f2fs*

Show the accumulated total revoked atomic write block count after boot. If you write ”0” here, you can initialize to ”0”.

/sys/fs/f2fs/<disk>/seq_file_ra_mul

Defined on file *sysfs-fs-f2fs*

You can control the multiplier value of bdi device readahead window size between 2 (default) and 256 for POSIX_FADV_SEQUENTIAL advise option.

/sys/fs/f2fs/<disk>/stat/cp_status

Defined on file *sysfs-fs-f2fs*

Show status of f2fs checkpoint in real time.

cp flag	value
CP_UMOUNT_FLAG	0x00000001
CP_ORPHAN_PRESENT_FLAG	0x00000002
CP_COMPACT_SUM_FLAG	0x00000004
CP_ERROR_FLAG	0x00000008
CP_FSCK_FLAG	0x00000010
CP_FASTBOOT_FLAG	0x00000020
CP_CRC_RECOVERY_FLAG	0x00000040
CP_NAT_BITS_FLAG	0x00000080
CP_TRIMMED_FLAG	0x00000100
CP_NOCRC_RECOVERY_FLAG	0x00000200
CP_LARGE_NAT_BITMAP_FLAG	0x00000400
CP_QUOTA_NEED_FSCK_FLAG	0x00000800
CP_DISABLED_FLAG	0x00001000
CP_DISABLED_QUICK_FLAG	0x00002000
CP_RESIZEFS_FLAG	0x00004000

/sys/fs/f2fs/<disk>/stat/issued_discard

Defined on file *sysfs-fs-f2fs*

Shows the number of issued discard.

/sys/fs/f2fs/<disk>/stat/queued_discard

Defined on file *sysfs-fs-f2fs*

Shows the number of queued discard.

/sys/fs/f2fs/<disk>/stat/sb_status

Defined on file *sysfs-fs-f2fs*

Show status of f2fs superblock in real time.

value	sb status macro	description
0x1	SBI_IS_DIRTY	dirty flag for checkpoint
0x2	SBI_IS_CLOSE	specify unmounting
0x4	SBI_NEED_FSCK	need fsck.f2fs to fix
0x8	SBI_POR_DOING	recovery is doing or not
0x10	SBI_NEED_SB_WRITE	need to recover superblock
0x20	SBI_NEED_CP	need to checkpoint
0x40	SBI_IS_SHUTDOWN	shutdown by ioctl
0x80	SBI_IS_RECOVERED	recovered orphan/data
0x100	SBI_CP_DISABLED	CP was disabled last mount
0x200	SBI_CP_DISABLED_QUICK	CP was disabled quickly
0x400	SBI_QUOTA_NEED_FLUSH	need to flush quota info in CP
0x800	SBI_QUOTA_SKIP_FLUSH	skip flushing quota in current CP
0x1000	SBI_QUOTA_NEED_REPAIR	quota file may be corrupted
0x2000	SBI_IS_RESIZEFS	resizefs is in process
0x4000	SBI_IS_FREEZING	freefs is in process

/sys/fs/f2fs/<disk>/stat/undiscard_blks

Defined on file *sysfs-fs-f2fs*

Shows the total number of undiscard blocks.

/sys/fs/f2fs/<disk>/umount_discard_timeout

Defined on file *sysfs-fs-f2fs*

Set timeout to issue discard commands during umount. Default: 5 secs

/sys/fs/f2fs/<disk>/unusable

Defined on file *sysfs-fs-f2fs*

If checkpoint=disable, it displays the number of blocks that are unusable. If checkpoint=enable it displays the number of blocks that would be unusable if checkpoint=disable were to be set.

/sys/fs/f2fs/<disk>/unusable_blocks_per_sec

Defined on file *sysfs-fs-f2fs*

Shows the number of unusable blocks in a section which was defined by the zone capacity reported by underlying zoned device.

/sys/fs/f2fs/<disk>/warm_data_age_threshold

Defined on file *sysfs-fs-f2fs*

When DATA SEPARATION is on, it controls the age threshold to indicate the data blocks as warm. By default it was initialized as 2621440 blocks (equals to 10GB).

/sys/fs/f2fs/features/

Defined on file *sysfs-fs-f2fs*

Shows all enabled kernel features. Supported features: encryption, block_zoned, extra_attr, project_quota, inode_checksum, flexible_inline_xattr, quota_ino, inode_crtime, lost_found, verity, sb_checksum, casefold, readonly, compression, test_dummy_encryption_v2, atomic_write, pin_file, encrypted_casefold.

/sys/fs/nilfs2/<device>/README

Defined on file *sysfs-fs-nilfs2*

Describe attributes of /sys/fs/nilfs2/<device> group.

/sys/fs/nilfs2/<device>/blocksize

Defined on file *sysfs-fs-nilfs2*

Show volume's block size in bytes.

/sys/fs/nilfs2/<device>/checkpoints/README

Defined on file *sysfs-fs-nilfs2*

Describe attributes of /sys/fs/nilfs2/<device>/checkpoints group.

/sys/fs/nilfs2/<device>/checkpoints/checkpoints_number

Defined on file *sysfs-fs-nilfs2*

Show number of checkpoints on volume.

/sys/fs/nilfs2/<device>/checkpoints/last_seg_checkpoint

Defined on file *sysfs-fs-nilfs2*

Show checkpoint number of the latest segment.

/sys/fs/nilfs2/<device>/checkpoints/next_checkpoint

Defined on file *sysfs-fs-nilfs2*

Show next checkpoint number.

/sys/fs/nilfs2/<device>/checkpoints/snapshots_number

Defined on file *sysfs-fs-nilfs2*

Show number of snapshots on volume.

/sys/fs/nilfs2/<device>/device_size

Defined on file *sysfs-fs-nilfs2*

Show volume size in bytes.

/sys/fs/nilfs2/<device>/free_blocks

Defined on file *sysfs-fs-nilfs2*

Show count of free blocks on volume.

/sys/fs/nilfs2/<device>/mounted_snapshots/<id>/README

Defined on file *sysfs-fs-nilfs2*

Describe attributes of /sys/fs/nilfs2/<device>/mounted_snapshots/<id> group.

/sys/fs/nilfs2/<device>/mounted_snapshots/<id>/blocks_count

Defined on file *sysfs-fs-nilfs2*

Show number of blocks for snapshot.

/sys/fs/nilfs2/<device>/mounted_snapshots/<id>/inodes_count

Defined on file *sysfs-fs-nilfs2*

Show number of inodes for snapshot.

/sys/fs/nilfs2/<device>/mounted_snapshots/README

Defined on file *sysfs-fs-nilfs2*

Describe content of /sys/fs/nilfs2/<device>/mounted_snapshots group.

/sys/fs/nilfs2/<device>/revision

Defined on file *sysfs-fs-nilfs2*

Show NILFS file system revision on volume. This value informs about metadata structures' revision on mounted volume.

/sys/fs/nilfs2/<device>/segctor/README

Defined on file *sysfs-fs-nilfs2*

Describe attributes of /sys/fs/nilfs2/<device>/segctor group.

/sys/fs/nilfs2/<device>/segctor/current_last_full_seg

Defined on file *sysfs-fs-nilfs2*

Show index number of the latest full segment.

/sys/fs/nilfs2/<device>/segctor/current_seg_sequence

Defined on file *sysfs-fs-nilfs2*

Show segment sequence counter.

/sys/fs/nilfs2/<device>/segctor/dirty_data_blocks_count

Defined on file *sysfs-fs-nilfs2*

Show number of dirty data blocks.

/sys/fs/nilfs2/<device>/segctor/last_nongc_write_time

Defined on file *sysfs-fs-nilfs2*

Show write time of the last segment not for cleaner operation in human-readable format.

/sys/fs/nilfs2/<device>/segctor/last_nongc_write_time_secs

Defined on file *sysfs-fs-nilfs2*

Show write time of the last segment not for cleaner operation in seconds.

/sys/fs/nilfs2/<device>/segctor/last_pseg_block

Defined on file *sysfs-fs-nilfs2*

Show start block number of the latest segment.

/sys/fs/nilfs2/<device>/segctor/last_seg_checkpoint

Defined on file *sysfs-fs-nilfs2*

Show checkpoint number of the latest segment.

/sys/fs/nilfs2/<device>/segctor/last_seg_sequence

Defined on file *sysfs-fs-nilfs2*

Show sequence value of the latest segment.

/sys/fs/nilfs2/<device>/segctor/last_seg_write_time

Defined on file *sysfs-fs-nilfs2*

Show write time of the last segment in human-readable format.

/sys/fs/nilfs2/<device>/segctor/last_seg_write_time_secs

Defined on file *sysfs-fs-nilfs2*

Show write time of the last segment in seconds.

/sys/fs/nilfs2/<device>/segctor/next_checkpoint

Defined on file *sysfs-fs-nilfs2*

Show next checkpoint number.

/sys/fs/nilfs2/<device>/segctor/next_full_seg

Defined on file *sysfs-fs-nilfs2*

Show index number of the full segment index to be used next.

/sys/fs/nilfs2/<device>/segctor/next_pseg_offset

Defined on file *sysfs-fs-nilfs2*

Show offset of next partial segment in the current full segment.

/sys/fs/nilfs2/<device>/segments/README

Defined on file *sysfs-fs-nilfs2*

Describe attributes of /sys/fs/nilfs2/<device>/segments group.

/sys/fs/nilfs2/<device>/segments/blocks_per_segment

Defined on file *sysfs-fs-nilfs2*

Show number of blocks in segment.

/sys/fs/nilfs2/<device>/segments/clean_segments

Defined on file *sysfs-fs-nilfs2*

Show count of clean segments.

/sys/fs/nilfs2/<device>/segments/dirty_segments

Defined on file *sysfs-fs-nilfs2*

Show count of dirty segments.

/sys/fs/nilfs2/<device>/segments/segments_number

Defined on file *sysfs-fs-nilfs2*

Show number of segments on a volume.

/sys/fs/nilfs2/<device>/superblock/README

Defined on file *sysfs-fs-nilfs2*

Describe attributes of /sys/fs/nilfs2/<device>/superblock group.

/sys/fs/nilfs2/<device>/superblock/sb_update_frequency

Defined on file *sysfs-fs-nilfs2*

Show/Set interval of periodical update of superblock (in seconds).

/sys/fs/nilfs2/<device>/superblock/sb_write_count

Defined on file *sysfs-fs-nilfs2*

Show current write count of super block.

/sys/fs/nilfs2/<device>/superblock/sb_write_time

Defined on file *sysfs-fs-nilfs2*

Show last write time of super block in human-readable format.

/sys/fs/nilfs2/<device>/superblock/sb_write_time_secs

Defined on file *sysfs-fs-nilfs2*

Show last write time of super block in seconds.

/sys/fs/nilfs2/<device>/uuid

Defined on file *sysfs-fs-nilfs2*

Show volume's UUID (Universally Unique Identifier).

/sys/fs/nilfs2/<device>/volume_name

Defined on file *sysfs-fs-nilfs2*

Show volume's label.

/sys/fs/nilfs2/features/README

Defined on file *sysfs-fs-nilfs2*

Describe attributes of /sys/fs/nilfs2/features group.

/sys/fs/nilfs2/features/revision

Defined on file *sysfs-fs-nilfs2*

Show current revision of NILFS file system driver. This value informs about file system revision that driver is ready to support.

/sys/fs/ocfs2/

Defined on file *sysfs-ocfs2*

The /sys/fs/ocfs2 directory contains knobs used by the ocfs2-tools to interact with the filesystem.

/sys/fs/ocfs2/active_cluster_plugin

Defined on file *sysfs-ocfs2*

The */sys/fs/ocfs2/active_cluster_plugin* displays which cluster plugin is currently in use by the filesystem. The active plugin will appear in the loaded_cluster_plugins file as well. Only one plugin can be used at a time.

Reading from this file returns the name of the active plugin on a single line.

This file is read-only. Which plugin is active depends on the cluster stack in use. The contents may change when all filesystems are unmounted and the cluster stack is changed.

/sys/fs/ocfs2/cluster_stack

Defined on file *sysfs-ocfs2*

The */sys/fs/ocfs2/cluster_stack* file contains the name of current ocfs2 cluster stack. This value is set by userspace tools when bringing the cluster stack online.

Cluster stack names are 4 characters in length.

When the 'o2cb' cluster stack is used, the 'o2cb' cluster plugin is active. All other cluster stacks use the 'user' cluster plugin.

Reading from this file returns the name of the current cluster stack on a single line.

Writing a new stack name to this file changes the current cluster stack unless there are mounted ocfs2 filesystems. If there are mounted filesystems, attempts to change the stack return an error.

Users:

ocfs2-tools <ocfs2-tools-devel@lists.linux.dev>

/sys/fs/ocfs2/loaded_cluster_plugins

Defined on file *sysfs-ocfs2*

The */sys/fs/ocfs2/loaded_cluster_plugins* file describes the available plugins to support ocfs2 cluster operation. A cluster plugin is required to use ocfs2 in a cluster. There are currently two available plugins:

- **'o2cb' - The classic o2cb cluster stack that ocfs2 has**
  used since its inception.
- **'user' - A plugin supporting userspace cluster software**
  in conjunction with fs/dlm.

Reading from this file returns the names of all loaded plugins, one per line.

This file is read-only. Its contents may change as plugins are loaded or removed.

/sys/fs/ocfs2/max_locking_protocol

Defined on file *sysfs-ocfs2*

The */sys/fs/ocfs2/max_locking_protocol* file displays version of ocfs2 locking supported by the filesystem. This version covers how ocfs2 uses distributed locking between cluster nodes.

The protocol version has a major and minor number. Two cluster nodes can interoperate if they have an identical major number and an overlapping minor number - thus, a node with version 1.10 can interoperate with a node sporting version 1.8, as long as both use the 1.8 protocol.

Reading from this file returns a single line, the major number and minor number joined by a period, eg "1.10".

This file is read-only. The value is compiled into the driver.

/sys/fs/pstore/..
/dev/pstore/..

Defined on file *pstore*

Generic interface to platform dependent persistent storage.

Platforms that provide a mechanism to preserve some data across system reboots can register with this driver to provide a generic interface to show records captured in the dying moments. In the case of a panic the last part of the console log is captured, but other interesting data can also be saved:

```
# mount -t pstore -o kmsg_bytes=8000 - /sys/fs/pstore

$ ls -l /sys/fs/pstore/
total 0
-r--r--r-- 1 root root 7896 Nov 30 15:38 dmesg-erst-1
```

Different users of this interface will result in different filename prefixes. Currently two are defined:

- "dmesg" - saved console log
- "mce" - architecture dependent data from fatal h/w error

Once the information in a file has been read, removing the file will signal to the underlying persistent storage device that it can reclaim the space for later re-use:

```
$ rm /sys/fs/pstore/dmesg-erst-1
```

The expectation is that all files in /sys/fs/pstore/ will be saved elsewhere and erased from persistent store soon after boot to free up space ready for the next catastrophe.

The 'kmsg_bytes' mount option changes the target amount of data saved on each oops/panic. Pstore saves (possibly multiple) files based on the record size of the underlying persistent storage until at least this amount is reached. Default is 10 Kbytes.

Pstore only supports one backend at a time. If multiple backends are available, the preferred backend may be set by passing the pstore.backend= argument to the kernel at boot time.

/sys/fs/ubifsX_Y/error_crc

Defined on file *sysfs-fs-ubifs*

Exposes crc errors: every node embeds a crc checksum.

This counter keeps track of the number of accesses of nodes with a bad crc checksum.

The counter is reset to 0 with a remount.

/sys/fs/ubifsX_Y/error_magic

Defined on file *sysfs-fs-ubifs*

Exposes magic errors: every node starts with a magic number.

This counter keeps track of the number of accesses of nodes with a corrupted magic number.

The counter is reset to 0 with a remount.

/sys/fs/ubifsX_Y/error_node

Defined on file *sysfs-fs-ubifs*

Exposes node errors. Every node embeds its type.

This counter keeps track of the number of accesses of nodes with a corrupted node type.

The counter is reset to 0 with a remount.

/sys/fs/xfs/<disk>/log/log_head_lsn

Defined on file *sysfs-fs-xfs*

The log sequence number (LSN) of the current head of the log. The LSN is exported in "cycle:basic block" format.

Users: xfstests

/sys/fs/xfs/<disk>/log/log_tail_lsn

Defined on file *sysfs-fs-xfs*

The log sequence number (LSN) of the current tail of the log. The LSN is exported in "cycle:basic block" format.

/sys/fs/xfs/<disk>/log/reserve_grant_head

Defined on file *sysfs-fs-xfs*

The current state of the log reserve grant head. It represents the total log reservation of all currently outstanding transactions. The grant head is exported in "cycle:bytes" format.

Users: xfstests

/sys/fs/xfs/<disk>/log/write_grant_head

Defined on file *sysfs-fs-xfs*

The current state of the log write grant head. It represents the total log reservation of all currently outstanding transactions, including regrants due to rolling transactions. The grant head is exported in "cycle:bytes" format.

Users: xfstests

5.2.37 Symbols under /sys/hypervisor

/sys/hypervisor/guest_type

Defined on file *sysfs-hypervisor-xen*

If running under Xen: Type of guest: "Xen": standard guest type on arm "HVM": fully virtualized guest (x86) "PV": paravirtualized guest (x86) "PVH": fully virtualized guest without legacy emulation (x86)

/sys/hypervisor/pmu/pmu_features

Defined on file *sysfs-hypervisor-xen*

If running under Xen: Describes Xen PMU features (as an integer). A set bit indicates that the corresponding feature is enabled. See include/xen/interface/xenpmu.h for available features

/sys/hypervisor/pmu/pmu_mode

Defined on file *sysfs-hypervisor-xen*

If running under Xen: Describes mode that Xen's performance-monitoring unit (PMU) uses. Accepted values are:

"off"	PMU is disabled
"self"	The guest can profile itself
"hv"	The guest can profile itself and, if it is privileged (e.g. dom0), the hypervisor
"all"	The guest can profile itself, the hypervisor and all other guests. Only available to privileged guests.

/sys/hypervisor/properties/buildid

Defined on file *sysfs-hypervisor-xen*

If running under Xen: Build id of the hypervisor, needed for hypervisor live patching. Might return "<denied>" in case of special security settings in the hypervisor.

5.2.38 Symbols under /sys/kernel

/sys/kernel/address_bit

Defined on file *sysfs-kernel-address_bits*

The address size of the running kernel in bits.

Access: Read

Users: util-linux

/sys/kernel/boot_params

Defined on file *sysfs-kernel-boot_params*

The */sys/kernel/boot_params* directory contains two files: "data" and "version" and one subdirectory "setup_data". It is used to export the kernel boot parameters of an x86 platform to userspace for kexec and debugging purpose.

If there's no setup_data in boot_params the subdirectory will not be created.

"data" file is the binary representation of struct boot_params.

"version" file is the string representation of boot protocol version.

"setup_data" subdirectory contains the setup_data data structure in boot_params. setup_data is maintained in kernel as a link list. In "setup_data" subdirectory there's one subdirectory for each link list node named with the number of the list nodes. The list node subdirectory contains two files "type" and "data". "type" file is the string representation of setup_data type. "data" file is the binary representation of setup_data payload.

The whole boot_params directory structure is like below:

```
/sys/kernel/boot_params
|__ data
|__ setup_data
|   |__ 0
|   |   |__ data
|   |   |__ type
|   |__ 1
|       |__ data
|       |__ type
|__ version
```

Users: Kexec

/sys/kernel/btf

Defined on file *sysfs-kernel-btf*

Contains BTF type information and related data for kernel and kernel modules.

/sys/kernel/btf/<module-name>

Defined on file *sysfs-kernel-btf*

Read-only binary attribute exposing kernel module's BTF type information as an add-on to the kernel's BTF (*/sys/kernel/btf/vmlinux*).

/sys/kernel/btf/vmlinux

Defined on file *sysfs-kernel-btf*

Read-only binary attribute exposing kernel's own BTF type information with description of all internal kernel types. See /bpf/btf for detailed description of format itself.

/sys/kernel/config/most_<component>

Defined on file *configfs-most*

Interface is used to configure and connect device channels to component drivers.

Attributes are visible only when configfs is mounted. To mount configfs in /sys/kernel/config directory use: # mount -t configfs none /sys/kernel/config/

/sys/kernel/config/most_cdev/<link>

Defined on file *configfs-most*

The attributes:

buffer_size
: configure the buffer size for this channel

subbuffer_size
: configure the sub-buffer size for this channel (needed for synchronous and isochronous data)

num_buffers
: configure number of buffers used for this channel

datatype
: configure type of data that will travel over this channel

direction
: configure whether this link will be an input or output

dbr_size
: configure DBR data buffer size (this is used for MediaLB communication only)

packets_per_xact
configure the number of packets that will be collected from the network before being transmitted via USB (this is used for USB communication only)

device
name of the device the link is to be attached to

channel
name of the channel the link is to be attached to

comp_params
pass parameters needed by some components

create_link
write '1' to this attribute to trigger the creation of the link. In case of speculative configuration, the creation is post-poned until a physical device is being attached to the bus.

destroy_link
write '1' to this attribute to destroy an active link

/sys/kernel/config/most_net/<link>

Defined on file *configfs-most*

The attributes:

buffer_size
configure the buffer size for this channel

subbuffer_size
configure the sub-buffer size for this channel (needed for synchronous and isochronous data)

num_buffers
configure number of buffers used for this channel

datatype
configure type of data that will travel over this channel

direction
configure whether this link will be an input or output

dbr_size
configure DBR data buffer size (this is used for MediaLB communication only)

packets_per_xact
configure the number of packets that will be collected from the network before being transmitted via USB (this is used for USB communication only)

device
name of the device the link is to be attached to

channel
name of the channel the link is to be attached to

comp_params
pass parameters needed by some components

create_link
write '1' to this attribute to trigger the creation of the link. In case of speculative configuration, the creation is post-poned until a physical device is being attached to the bus.

destroy_link
write '1' to this attribute to destroy an active link

/sys/kernel/config/most_sound/<card>

Defined on file *configfs-most*

The attributes:

create_card
write '1' to this attribute to trigger the registration of the sound card with the ALSA subsystem.

/sys/kernel/config/most_sound/<card>/<link>

Defined on file *configfs-most*

The attributes:

buffer_size
configure the buffer size for this channel

subbuffer_size
configure the sub-buffer size for this channel (needed for synchronous and isochronous data)

num_buffers
configure number of buffers used for this channel

datatype
configure type of data that will travel over this channel

direction
configure whether this link will be an input or output

dbr_size
configure DBR data buffer size (this is used for MediaLB communication only)

packets_per_xact
configure the number of packets that will be collected from the network before being transmitted via USB (this is used for USB communication only)

device
name of the device the link is to be attached to

channel
name of the channel the link is to be attached to

comp_params
pass parameters needed by some components

create_link
write '1' to this attribute to trigger the creation of the link. In case of speculative configuration, the creation is post-poned until a physical device is being attached to the bus.

destroy_link
write '1' to this attribute to destroy an active link

/sys/kernel/config/most_video/<link>

Defined on file *configfs-most*

The attributes:

buffer_size
configure the buffer size for this channel

subbuffer_size
configure the sub-buffer size for this channel (needed for synchronous and isochronous data)

num_buffers
configure number of buffers used for this channel

datatype
configure type of data that will travel over this channel

direction
configure whether this link will be an input or output

dbr_size
configure DBR data buffer size (this is used for MediaLB communication only)

packets_per_xact
configure the number of packets that will be collected from the network before being transmitted via USB (this is used for USB communication only)

device
name of the device the link is to be attached to

channel
name of the channel the link is to be attached to

comp_params
pass parameters needed by some components

create_link
write '1' to this attribute to trigger the creation of the link. In case of speculative configuration, the creation is post-poned until a physical device is being attached to the bus.

destroy_link
write '1' to this attribute to destroy an active link

/sys/kernel/config/tsm/report/$name/auxblob

Defined on file *configfs-tsm*

(RO) Optional supplemental data that a TSM may emit, visibility of this attribute depends on TSM, and may be empty if no auxiliary data is available.

When @provider is ”sev_guest” this file contains the ”cert_table” from SEV-ES Guest-Hypervisor Communication Block Standardization v2.03 Section 4.1.8.1 MSG_REPORT_REQ. https://www.amd.com/content/dam/amd/en/documents/epyc-technical-docs/specifications/56421.pdf

/sys/kernel/config/tsm/report/$name/generation

Defined on file *configfs-tsm*

(RO) The value in this attribute increments each time @inblob or any option is written. Userspace can detect conflicts by checking generation before writing to any attribute and making sure the number of writes matches expectations after reading @outblob, or it can prevent conflicts by creating a report instance per requesting context.

/sys/kernel/config/tsm/report/$name/inblob

Defined on file *configfs-tsm*

(WO) Up to 64 bytes of user specified binary data. For replay protection this should include a nonce, but the kernel does not place any restrictions on the content.

/sys/kernel/config/tsm/report/$name/outblob

Defined on file *configfs-tsm*

(RO) Binary attestation report generated from @inblob and other options The format of the report is implementation specific where the implementation is conveyed via the @provider attribute.

/sys/kernel/config/tsm/report/$name/privlevel

Defined on file *configfs-tsm*

(WO) Attribute is visible if a TSM implementation provider supports the concept of attestation reports for TVMs running at different privilege levels, like SEV-SNP "VMPL", specify the privilege level via this attribute. The minimum acceptable value is conveyed via @privlevel_floor and the maximum acceptable value is TSM_PRIVLEVEL_MAX (3).

/sys/kernel/config/tsm/report/$name/privlevel_floor

Defined on file *configfs-tsm*

(RO) Indicates the minimum permissible value that can be written to @privlevel.

/sys/kernel/config/tsm/report/$name/provider

Defined on file *configfs-tsm*

(RO) A name for the format-specification of @outblob like "sev_guest" [1] or "tdx_guest" [2] in the near term, or a common standard format in the future.

[1]: SEV Secure Nested Paging Firmware ABI Specification Revision 1.55 Table 22 https://www.amd.com/content/dam/amd/en/documents/epyc-technical-docs/specifications/56860.pdf

[2]: Intel® Trust Domain Extensions Data Center Attestation Primitives : Quote Generation Library and Quote Verification Library Revision 0.8 Appendix 4,5 https://download.01.org/intel-sgx/latest/dcap-latest/linux/docs/Intel_TDX_DCAP_Quoting_Library_API.pdf

/sys/kernel/cpu_byteorder

Defined on file *sysfs-kernel-cpu_byteorder*

The endianness of the running kernel.

Access: Read

Valid values:
"little", "big"

Users: util-linux

/sys/kernel/debug/<cros-ec-device>/console_log

Defined on file *debugfs-cros-ec*

If the EC supports the CONSOLE_READ command type, this file can be used to grab the EC logs. The kernel polls for the log and keeps its own buffer but userspace should grab this and write it out to some logs.

/sys/kernel/debug/<cros-ec-device>/last_resume_result

Defined on file *debugfs-cros-ec*

Some ECs have a feature where they will track transitions to the (Intel) processor's SLP_S0 line, in order to detect cases where a system failed to go into S0ix. When the system resumes, an EC with this feature will return a summary of SLP_S0 transitions that occurred. The last_resume_result file returns the most recent response from the AP's resume message to the EC.

The bottom 31 bits contain a count of the number of SLP_S0 transitions that occurred since the suspend message was received. Bit 31 is set if the EC attempted to wake the system due to a timeout when watching for SLP_S0 transitions. Callers can use this to detect a wake from the EC due to S0ix timeouts. The result will be zero if no suspend transitions have been attempted, or the EC does not support this feature.

Output will be in the format: "0x%08xn".

/sys/kernel/debug/<cros-ec-device>/panicinfo

Defined on file *debugfs-cros-ec*

This file dumps the EC panic information from the previous reboot. This file will only exist if the PANIC_INFO command type is supported by the EC.

/sys/kernel/debug/<cros-ec-device>/pdinfo

Defined on file *debugfs-cros-ec*

This file provides the port role, muxes and power debug information for all the USB PD/type-C ports available. If the are no ports available, this file will be just an empty file.

/sys/kernel/debug/<cros-ec-device>/suspend_timeout_ms

Defined on file *debugfs-cros-ec*

Some ECs have a feature where they will track transitions of a hardware-controlled sleep line, such as Intel's SLP_S0 line, in order to detect cases where a system failed to go into deep sleep states. The suspend_timeout_ms file controls the amount of time in milliseconds the EC will wait before declaring a sleep timeout event and attempting to wake the system.

Supply 0 to use the default value coded into EC firmware. Supply 65535 (EC_HOST_SLEEP_TIMEOUT_INFINITE) to disable the EC sleep failure detection mechanism. Values in between 0 and 65535 indicate the number of milliseconds the EC should wait after a sleep transition before declaring a timeout. This includes both the duration after a sleep command was received but before the hardware line changed, as well as the duration between when the hardware line changed and the kernel sent an EC resume command.

Output will be in the format: "%un".

/sys/kernel/debug/<cros-ec-device>/uptime

Defined on file *debugfs-cros-ec*

A u32 providing the time since EC booted in ms. This is is used for synchronizing the AP host time with the EC log. An error is returned if the command is not supported by the EC or there is a communication problem.

/sys/kernel/debug/accel/<parent_device>/addr

Defined on file *debugfs-driver-habanalabs*

Sets the device address to be used for read or write through PCI bar, or the device VA of a host mapped memory to be read or written directly from the host. The latter option is allowed only when the IOMMU is disabled. The acceptable value is a string that starts with "0x"

/sys/kernel/debug/accel/<parent_device>/clk_gate

Defined on file *debugfs-driver-habanalabs*

This setting is now deprecated as clock gating is handled solely by the f/w

/sys/kernel/debug/accel/<parent_device>/command_buffers

Defined on file *debugfs-driver-habanalabs*

Displays a list with information about the currently allocated command buffers

/sys/kernel/debug/accel/<parent_device>/command_submission

Defined on file *debugfs-driver-habanalabs*

Displays a list with information about the currently active command submissions

/sys/kernel/debug/accel/<parent_device>/command_submission_jobs

Defined on file *debugfs-driver-habanalabs*

Displays a list with detailed information about each JOB (CB) of each active command submission

/sys/kernel/debug/accel/<parent_device>/data32

Defined on file *debugfs-driver-habanalabs*

Allows the root user to read or write directly through the device's PCI bar. Writing to this file generates a write transaction while reading from the file generates a read transaction. This custom interface is needed (instead of using the generic Linux user-space PCI mapping) because the DDR bar is very small compared to the DDR memory and only the driver can move the bar before and after the transaction.

If the IOMMU is disabled, it also allows the root user to read or write from the host a device VA of a host mapped memory

/sys/kernel/debug/accel/<parent_device>/data64

Defined on file *debugfs-driver-habanalabs*

Allows the root user to read or write 64 bit data directly through the device's PCI bar. Writing to this file generates a write transaction while reading from the file generates a read transaction. This custom interface is needed (instead of using the generic Linux user-space PCI mapping) because the DDR bar is very small compared to the DDR memory and only the driver can move the bar before and after the transaction.

If the IOMMU is disabled, it also allows the root user to read or write from the host a device VA of a host mapped memory

/sys/kernel/debug/accel/<parent_device>/data_dma

Defined on file *debugfs-driver-habanalabs*

Allows the root user to read from the device's internal memory (DRAM/SRAM) through a DMA engine. This property is a binary blob that contains the result of the DMA transfer. This custom interface is needed (instead of using the generic Linux user-space PCI mapping) because the amount of internal memory is huge (>32GB) and reading it via the PCI bar will take a very long time. This interface doesn't support concurrency in the same device. In GAUDI and GOYA, this action can cause undefined behavior in case it is done while the device is executing user workloads. Only supported on GAUDI at this stage.

/sys/kernel/debug/accel/<parent_device>/device

Defined on file *debugfs-driver-habanalabs*

Enables the root user to set the device to specific state. Valid values are ”disable”, ”enable”, ”suspend”, ”resume”. User can read this property to see the valid values

/sys/kernel/debug/accel/<parent_device>/device_release_watchdog_timeout

Defined on file *debugfs-driver-habanalabs*

The watchdog timeout value in seconds for a device release upon certain error cases, after which the device is reset.

/sys/kernel/debug/accel/<parent_device>/dma_size

Defined on file *debugfs-driver-habanalabs*

Specify the size of the DMA transaction when using DMA to read from the device's internal memory. The value can not be larger than 128MB. Writing to this value initiates the DMA transfer. When the write is finished, the user can read the "data_dma" blob

/sys/kernel/debug/accel/<parent_device>/dump_razwi_events

Defined on file *debugfs-driver-habanalabs*

Dumps all razwi events to dmesg if exist. After reading the status register of an existing event the routine will clear the status register. Usage: cat dump_razwi_events

/sys/kernel/debug/accel/<parent_device>/dump_security_violations

Defined on file *debugfs-driver-habanalabs*

Dumps all security violations to dmesg. This will also ack all security violations meanings those violations will not be dumped next time user calls this API

/sys/kernel/debug/accel/<parent_device>/engines

Defined on file *debugfs-driver-habanalabs*

Displays the status registers values of the device engines and their derived idle status

/sys/kernel/debug/accel/<parent_device>/i2c_addr

Defined on file *debugfs-driver-habanalabs*

Sets I2C device address for I2C transaction that is generated by the device's CPU, Not available when device is loaded with secured firmware

/sys/kernel/debug/accel/<parent_device>/i2c_bus

Defined on file *debugfs-driver-habanalabs*

Sets I2C bus address for I2C transaction that is generated by the device's CPU, Not available when device is loaded with secured firmware

/sys/kernel/debug/accel/<parent_device>/i2c_data

Defined on file *debugfs-driver-habanalabs*

Triggers an I2C transaction that is generated by the device's CPU. Writing to this file generates a write transaction while reading from the file generates a read transaction, Not available when device is loaded with secured firmware

/sys/kernel/debug/accel/<parent_device>/i2c_len

Defined on file *debugfs-driver-habanalabs*

Sets I2C length in bytes for I2C transaction that is generated by the device's CPU, Not available when device is loaded with secured firmware

/sys/kernel/debug/accel/<parent_device>/i2c_reg

Defined on file *debugfs-driver-habanalabs*

Sets I2C register id for I2C transaction that is generated by the device's CPU, Not available when device is loaded with secured firmware

/sys/kernel/debug/accel/<parent_device>/led0

Defined on file *debugfs-driver-habanalabs*

Sets the state of the first S/W led on the device, Not available when device is loaded with secured firmware

/sys/kernel/debug/accel/<parent_device>/led1

Defined on file *debugfs-driver-habanalabs*

Sets the state of the second S/W led on the device, Not available when device is loaded with secured firmware

/sys/kernel/debug/accel/<parent_device>/led2

Defined on file *debugfs-driver-habanalabs*

Sets the state of the third S/W led on the device, Not available when device is loaded with secured firmware

/sys/kernel/debug/accel/<parent_device>/memory_scrub

Defined on file *debugfs-driver-habanalabs*

Allows the root user to scrub the dram memory. The scrubbing value can be set using the debugfs file memory_scrub_val.

/sys/kernel/debug/accel/<parent_device>/memory_scrub_val

Defined on file *debugfs-driver-habanalabs*

The value to which the dram will be set to when the user scrubs the dram using 'memory_scrub' debugfs file and the scrubbing value when using module param 'memory_scrub'

/sys/kernel/debug/accel/<parent_device>/mmu

Defined on file *debugfs-driver-habanalabs*

Displays the hop values and physical address for a given ASID and virtual address. The user should write the ASID and VA into the file and then read the file to get the result. e.g. to display info about VA 0x1000 for ASID 1 you need to do: echo "1 0x1000" > /sys/kernel/debug/accel/0/mmu

/sys/kernel/debug/accel/<parent_device>/mmu_error

Defined on file *debugfs-driver-habanalabs*

Check and display page fault or access violation mmu errors for all MMUs specified in mmu_cap_mask. e.g. to display error info for MMU hw cap bit 9, you need to do: echo "0x200" > /sys/kernel/debug/accel/0/mmu_error cat /sys/kernel/debug/accel/0/mmu_error

/sys/kernel/debug/accel/<parent_device>/monitor_dump

Defined on file *debugfs-driver-habanalabs*

Allows the root user to dump monitors status from the device's protected config space. This property is a binary blob that contains the result of the monitors registers dump. This custom interface is needed (instead of using the generic Linux user-space PCI mapping) because this space is protected and cannot be accessed using PCI read. This interface doesn't support concurrency in the same device. Only supported on GAUDI.

/sys/kernel/debug/accel/<parent_device>/monitor_dump_trig

Defined on file *debugfs-driver-habanalabs*

Triggers dump of monitor data. The value to trigger the operation must be 1. Triggering the monitor dump operation initiates dump of current registers values of all monitors. When the write is finished, the user can read the "monitor_dump" blob

/sys/kernel/debug/accel/<parent_device>/set_power_state

Defined on file *debugfs-driver-habanalabs*

Sets the PCI power state. Valid values are "1" for D0 and "2" for D3Hot

/sys/kernel/debug/accel/<parent_device>/skip_reset_on_timeout

Defined on file *debugfs-driver-habanalabs*

Sets the skip reset on timeout option for the device. Value of "0" means device will be reset in case some CS has timed out, otherwise it will not be reset.

/sys/kernel/debug/accel/<parent_device>/state_dump

Defined on file *debugfs-driver-habanalabs*

Gets the state dump occurring on a CS timeout or failure. State dump is used for debug and is created each time in case of a problem in a CS execution, before reset. Reading from the node

returns the newest state dump available. Writing an integer X discards X state dumps, so that the next read would return X+1-st newest state dump.

/sys/kernel/debug/accel/<parent_device>/stop_on_err

Defined on file *debugfs-driver-habanalabs*

Sets the stop-on_error option for the device engines. Value of "0" is for disable, otherwise enable. Relevant only for GOYA and GAUDI.

/sys/kernel/debug/accel/<parent_device>/timeout_locked

Defined on file *debugfs-driver-habanalabs*

Sets the command submission timeout value in seconds.

/sys/kernel/debug/accel/<parent_device>/userptr

Defined on file *debugfs-driver-habanalabs*

Displays a list with information about the current user pointers (user virtual addresses) that are pinned and mapped to DMA addresses

/sys/kernel/debug/accel/<parent_device>/userptr_lookup

Defined on file *debugfs-driver-habanalabs*

Allows to search for specific user pointers (user virtual addresses) that are pinned and mapped to DMA addresses, and see their resolution to the specific dma address.

/sys/kernel/debug/accel/<parent_device>/vm

Defined on file *debugfs-driver-habanalabs*

Displays a list with information about all the active virtual address mappings per ASID and all user mappings of HW blocks

/sys/kernel/debug/cec/*/error-inj

Defined on file *debugfs-cec-error-inj*

The CEC Framework allows for CEC error injection commands through debugfs. Drivers that support this will create an error-inj file through which the error injection commands can be given.

The basic syntax is as follows:

Leading spaces/tabs are ignored. If the next character is a '#' or the end of the line was reached, then the whole line is ignored. Otherwise a command is expected.

It is up to the driver to decide what commands to implement. The only exception is that the command 'clear' without any arguments must be implemented and that it will remove all current error injection commands.

This ensures that you can always do 'echo clear >error-inj' to clear any error injections without having to know the details of the driver-specific commands.

Note that the output of 'error-inj' shall be valid as input to 'error-inj'. So this must work:

```
$ cat error-inj >einj.txt
$ cat einj.txt >error-inj
```

Other than these basic rules described above this ABI is not considered stable and may change in the future.

Drivers that implement this functionality must document the commands as part of the CEC documentation and must keep that documentation up to date when changes are made.

The following CEC error injection implementations exist:

- /userspace-api/media/cec/cec-pin-error-inj

/sys/kernel/debug/cxl/memX/inject_poison

Defined on file *debugfs-cxl*

(WO) When a Device Physical Address (DPA) is written to this attribute, the memdev driver sends an inject poison command to the device for the specified address. The DPA must be 64-byte aligned and the length of the injected poison is 64-bytes. If successful, the device returns poison when the address is accessed through the CXL.mem bus. Injecting poison adds the address to the device's Poison List and the error source is set to Injected. In addition, the device adds a poison creation event to its internal Informational Event log, updates the Event Status register, and if configured, interrupts the host. It is not an error to inject poison into an address that already has poison present and no error is returned. The inject_poison attribute is only visible for devices supporting the capability.

/sys/kernel/debug/dcc/.../[list-number]/config

Defined on file *debugfs-driver-dcc*

This stores the addresses of the registers which can be read in case of a hardware crash or manual software triggers. The input addresses type can be one of following dcc instructions: read, write, read-write, and loop type. The lists need to be configured sequentially and not in a overlapping manner; e.g. users can jump to list x only after list y is configured and enabled. The input format for each type is as follows:

i) Read instruction

```
echo R <addr> <n> <bus> >/sys/kernel/debug/dcc/../[list-number]/config
```

where:

<addr>
The address to be read.

<n>
The addresses word count, starting from address <1>. Each word is 32 bits (4 bytes). If omitted, defaulted to 1.

<bus type>
The bus type, which can be either 'apb' or 'ahb'. The default is 'ahb' if leaved out.

ii) Write instruction

```
echo W <addr> <n> <bus type> > /sys/kernel/debug/dcc/../[list-number]/
↪config
```

where:

<addr>
The address to be written.

<n>
The value to be written at <addr>.

<bus type>
The bus type, which can be either 'apb' or 'ahb'.

iii) Read-write instruction

```
echo RW <addr> <n> <mask> > /sys/kernel/debug/dcc/../[list-number]/config
```

where:

<addr>
The address to be read and written.

<n>
The value to be written at <addr>.

<mask>
The value mask.

iv) Loop instruction

```
echo L <loop count> <address count> <address>... > /sys/kernel/debug/dcc/..
↪/[list-number]/config
```

where:

<loop count>
Number of iterations

<address count>
total number of addresses to be written

<address>
Space-separated list of addresses.

/sys/kernel/debug/dcc/.../[list-number]/enable

Defined on file *debugfs-driver-dcc*

This debugfs interface is used for enabling the the dcc hardware. A file named "enable" is in the directory list number where users can enable/disable the specific list by writing boolean (1 or 0) to the file.

On enabling the dcc, all the addresses specified by the user for the corresponding list is written into dcc sram which is read by the dcc hardware on manual or crash induced triggers. Lists must be configured and enabled sequentially, e.g. list 2 can only be enabled when list 1 have so.

/sys/kernel/debug/dcc/.../config_reset

Defined on file *debugfs-driver-dcc*

This file is used to reset the configuration of a dcc driver to the default configuration. When '1' is written to the file, all the previous addresses stored in the driver gets removed and users need to reconfigure addresses again.

/sys/kernel/debug/dcc/.../ready

Defined on file *debugfs-driver-dcc*

This file is used to check the status of the dcc hardware if it's ready to receive user configurations. A 'Y' here indicates dcc is ready.

/sys/kernel/debug/dcc/.../trigger

Defined on file *debugfs-driver-dcc*

This is the debugfs interface for manual software triggers. The trigger can be invoked by writing '1' to the file.

/sys/kernel/debug/dell-wmi-ddv-<wmi_device_name>/fan_sensor_information

Defined on file *debugfs-dell-wmi-ddv*

This file contains the contents of the fan sensor information buffer, which contains fan sensor entries and a terminating character (0xFF).

Each fan sensor entry contains:

- fan type (single byte)
- fan speed in RPM (two bytes, little endian)

See /wmi/devices/dell-wmi-ddv for details.

/sys/kernel/debug/dell-wmi-ddv-<wmi_device_name>/thermal_sensor_information

Defined on file *debugfs-dell-wmi-ddv*

This file contains the contents of the thermal sensor information buffer, which contains thermal sensor entries and a terminating character (0xFF).

Each thermal sensor entry contains:

- thermal type (single byte)
- current temperature (single byte)
- min. temperature (single byte)

- max. temperature (single byte)
- unknown field (single byte)

See /wmi/devices/dell-wmi-ddv for details.

/sys/kernel/debug/ec/*/{gpe,use_global_lock,io}

Defined on file *debugfs-ec*

General information like which GPE is assigned to the EC and whether the global lock should get used. Knowing the EC GPE one can watch the amount of HW events related to the EC here (XY -> GPE number from */sys/kernel/debug/ec/*/gpe*): /sys/firmware/acpi/interrupts/gpeXY

The io file is binary and a userspace tool located here: ftp://ftp.suse.com/pub/people/trenn/sources/ec/ should get used to read out the 256 Embedded Controller registers or writing to them.

CAUTION:

Do not write to the Embedded Controller if you don't know what you are doing! Rebooting afterwards also is a good idea. This can influence the way your machine is cooled and fans may not get switched on again after you did a wrong write.

/sys/kernel/debug/genwqe/genwqe<n>_card/curr_dbg_uid0

Defined on file *debugfs-driver-genwqe*

Internal chip state of UID0 (unit id 0). Only available for PF.

/sys/kernel/debug/genwqe/genwqe<n>_card/curr_dbg_uid1

Defined on file *debugfs-driver-genwqe*

Internal chip state of UID1. Only available for PF.

/sys/kernel/debug/genwqe/genwqe<n>_card/curr_dbg_uid2

Defined on file *debugfs-driver-genwqe*

Internal chip state of UID2. Only available for PF.

/sys/kernel/debug/genwqe/genwqe<n>_card/curr_regs

Defined on file *debugfs-driver-genwqe*

Dump of the current error registers. Only available for PF.

/sys/kernel/debug/genwqe/genwqe<n>_card/ddcb_info

Defined on file *debugfs-driver-genwqe*

DDCB queue dump used for debugging queueing problems.

/sys/kernel/debug/genwqe/genwqe<n>_card/err_inject

Defined on file *debugfs-driver-genwqe*

Possibility to inject error cases to ensure that the drivers error handling code works well.

/sys/kernel/debug/genwqe/genwqe<n>_card/info

Defined on file *debugfs-driver-genwqe*

Comprehensive summary of bitstream version and software version. Used bitstream and bitstream clocking information.

/sys/kernel/debug/genwqe/genwqe<n>_card/jobtimer

Defined on file *debugfs-driver-genwqe*

Dump job timeout register values for PF and VFs. Only available for PF.

/sys/kernel/debug/genwqe/genwqe<n>_card/prev_dbg_uid0

Defined on file *debugfs-driver-genwqe*

Internal chip state of UID0 before card was reset. Only available for PF.

/sys/kernel/debug/genwqe/genwqe<n>_card/prev_dbg_uid1

Defined on file *debugfs-driver-genwqe*

Internal chip state of UID1 before card was reset. Only available for PF.

/sys/kernel/debug/genwqe/genwqe<n>_card/prev_dbg_uid2

Defined on file *debugfs-driver-genwqe*

Internal chip state of UID2 before card was reset. Only available for PF.

/sys/kernel/debug/genwqe/genwqe<n>_card/prev_regs

Defined on file *debugfs-driver-genwqe*

Dump of the error registers before the last reset of the card occurred. Only available for PF.

/sys/kernel/debug/genwqe/genwqe<n>_card/queue_working_time

Defined on file *debugfs-driver-genwqe*

Dump queue working time register values for PF and VFs. Only available for PF.

/sys/kernel/debug/genwqe/genwqe<n>_card/vf<0..14>_jobtimeout_msec

Defined on file *debugfs-driver-genwqe*

Default VF timeout 250ms. Testing might require 1000ms. Using 0 will use the cards default value (whatever that is).

The timeout depends on the max number of available cards in the system and the maximum allowed queue size.

The driver ensures that the settings are done just before the VFs get enabled. Changing the timeouts in flight is not possible. Only available for PF.

/sys/kernel/debug/hisi_hpre/<bdf>/alg_qos

Defined on file *debugfs-hisi-hpre*

The <bdf> is related the function for PF and VF. HPRE driver supports to configure each function's QoS, the driver supports to write <bdf> value to alg_qos in the host. Such as "echo <bdf> value > alg_qos". The qos value is 1~1000, means 1/1000~1000/1000 of total QoS. The driver reading alg_qos to get related QoS in the host and VM, Such as "cat alg_qos".

/sys/kernel/debug/hisi_hpre/<bdf>/cluster[0-3]/cluster_ctrl

Defined on file *debugfs-hisi-hpre*

Write the HPRE core selection in the cluster into this file, and then we can read the debug information of the core. Only available for PF.

/sys/kernel/debug/hisi_hpre/<bdf>/cluster[0-3]/regs

Defined on file *debugfs-hisi-hpre*

Dump debug registers from the HPRE cluster. Only available for PF.

/sys/kernel/debug/hisi_hpre/<bdf>/current_qm

Defined on file *debugfs-hisi-hpre*

One HPRE controller has one PF and multiple VFs, each function has a QM. Select the QM which below qm refers to. Only available for PF.

/sys/kernel/debug/hisi_hpre/<bdf>/hpre_dfx/diff_regs

Defined on file *debugfs-hisi-hpre*

HPRE debug registers(regs) read hardware register value. This node is used to show the change of the register values. This node can be help users to check the change of register values.

/sys/kernel/debug/hisi_hpre/<bdf>/hpre_dfx/invalid_req_cnt

Defined on file *debugfs-hisi-hpre*

Dump the total number of invalid requests being received. Available for both PF and VF, and take no other effect on HPRE.

/sys/kernel/debug/hisi_hpre/<bdf>/hpre_dfx/over_thrhld_cnt

Defined on file *debugfs-hisi-hpre*

Dump the total number of time out requests. Available for both PF and VF, and take no other effect on HPRE.

/sys/kernel/debug/hisi_hpre/<bdf>/hpre_dfx/overtime_thrhld

Defined on file *debugfs-hisi-hpre*

Set the threshold time for counting the request which is processed longer than the threshold. 0: disable(default), 1: 1 microsecond. Available for both PF and VF, and take no other effect on HPRE.

/sys/kernel/debug/hisi_hpre/<bdf>/hpre_dfx/recv_cnt

Defined on file *debugfs-hisi-hpre*

Dump the total number of received requests. Available for both PF and VF, and take no other effect on HPRE.

/sys/kernel/debug/hisi_hpre/<bdf>/hpre_dfx/send_busy_cnt

Defined on file *debugfs-hisi-hpre*

Dump the total number of requests sent with returning busy. Available for both PF and VF, and take no other effect on HPRE.

/sys/kernel/debug/hisi_hpre/<bdf>/hpre_dfx/send_cnt

Defined on file *debugfs-hisi-hpre*

Dump the total number of sent requests. Available for both PF and VF, and take no other effect on HPRE.

/sys/kernel/debug/hisi_hpre/<bdf>/hpre_dfx/send_fail_cnt

Defined on file *debugfs-hisi-hpre*

Dump the total number of completed but error requests. Available for both PF and VF, and take no other effect on HPRE.

/sys/kernel/debug/hisi_hpre/<bdf>/qm/abnormal_irq

Defined on file *debugfs-hisi-hpre*

Dump the number of interrupts for QM abnormal event. Available for both PF and VF, and take no other effect on HPRE.

/sys/kernel/debug/hisi_hpre/<bdf>/qm/aeq_irq

Defined on file *debugfs-hisi-hpre*

Dump the number of QM async event queue interrupts. Available for both PF and VF, and take no other effect on HPRE.

/sys/kernel/debug/hisi_hpre/<bdf>/qm/clear_enable

Defined on file *debugfs-hisi-hpre*

QM debug registers(regs) read clear control. 1 means enable register read clear, otherwise 0. Writing to this file has no functional effect, only enable or disable counters clear after reading of these registers. Only available for PF.

/sys/kernel/debug/hisi_hpre/<bdf>/qm/create_qp_err

Defined on file *debugfs-hisi-hpre*

Dump the number of queue allocation errors. Available for both PF and VF, and take no other effect on HPRE.

/sys/kernel/debug/hisi_hpre/<bdf>/qm/current_q

Defined on file *debugfs-hisi-hpre*

One QM may contain multiple queues. Select specific queue to show its debug registers in above regs. Only available for PF.

/sys/kernel/debug/hisi_hpre/<bdf>/qm/diff_regs

Defined on file *debugfs-hisi-hpre*

QM debug registers(regs) read hardware register value. This node is used to show the change of the qm register values. This node can be help users to check the change of register values.

/sys/kernel/debug/hisi_hpre/<bdf>/qm/err_irq

Defined on file *debugfs-hisi-hpre*

Dump the number of invalid interrupts for QM task completion. Available for both PF and VF, and take no other effect on HPRE.

/sys/kernel/debug/hisi_hpre/<bdf>/qm/mb_err

Defined on file *debugfs-hisi-hpre*

Dump the number of failed QM mailbox commands. Available for both PF and VF, and take no other effect on HPRE.

/sys/kernel/debug/hisi_hpre/<bdf>/qm/regs

Defined on file *debugfs-hisi-hpre*

Dump debug registers from the QM. Available for PF and VF in host. VF in guest currently only has one debug register.

/sys/kernel/debug/hisi_hpre/<bdf>/qm/status

Defined on file *debugfs-hisi-hpre*

Dump the status of the QM. Two states: work, stop. Available for both PF and VF, and take no other effect on HPRE.

/sys/kernel/debug/hisi_hpre/<bdf>/rdclr_en

Defined on file *debugfs-hisi-hpre*

HPRE cores debug registers read clear control. 1 means enable register read clear, otherwise 0. Writing to this file has no functional effect, only enable or disable counters clear after reading of these registers. Only available for PF.

/sys/kernel/debug/hisi_hpre/<bdf>/regs

Defined on file *debugfs-hisi-hpre*

Dump debug registers from the HPRE. Only available for PF.

/sys/kernel/debug/hisi_sec2/<bdf>/alg_qos

Defined on file *debugfs-hisi-sec*

The <bdf> is related the function for PF and VF. SEC driver supports to configure each function's QoS, the driver supports to write <bdf> value to alg_qos in the host. Such as "echo <bdf> value > alg_qos". The qos value is 1~1000, means 1/1000~1000/1000 of total QoS. The driver reading alg_qos to get related QoS in the host and VM, Such as "cat alg_qos".

/sys/kernel/debug/hisi_sec2/<bdf>/clear_enable

Defined on file *debugfs-hisi-sec*

Enabling/disabling of clear action after reading the SEC debug registers. 0: disable, 1: enable. Only available for PF, and take no other effect on SEC.

/sys/kernel/debug/hisi_sec2/<bdf>/current_qm

Defined on file *debugfs-hisi-sec*

One SEC controller has one PF and multiple VFs, each function has a QM. This file can be used to select the QM which below qm refers to. Only available for PF.

/sys/kernel/debug/hisi_sec2/<bdf>/qm/abnormal_irq

Defined on file *debugfs-hisi-sec*

Dump the number of interrupts for QM abnormal event. Available for both PF and VF, and take no other effect on SEC.

/sys/kernel/debug/hisi_sec2/<bdf>/qm/aeq_irq

Defined on file *debugfs-hisi-sec*

Dump the number of QM async event queue interrupts. Available for both PF and VF, and take no other effect on SEC.

/sys/kernel/debug/hisi_sec2/<bdf>/qm/clear_enable

Defined on file *debugfs-hisi-sec*

Enabling/disabling of clear action after reading the SEC's QM debug registers. 0: disable, 1: enable. Only available for PF, and take no other effect on SEC.

/sys/kernel/debug/hisi_sec2/<bdf>/qm/create_qp_err

Defined on file *debugfs-hisi-sec*

Dump the number of queue allocation errors. Available for both PF and VF, and take no other effect on SEC.

/sys/kernel/debug/hisi_sec2/<bdf>/qm/current_q

Defined on file *debugfs-hisi-sec*

One QM of SEC may contain multiple queues. Select specific queue to show its debug registers in above 'regs'. Only available for PF.

/sys/kernel/debug/hisi_sec2/<bdf>/qm/diff_regs

Defined on file *debugfs-hisi-sec*

QM debug registers(regs) read hardware register value. This node is used to show the change of the qm register values. This node can be help users to check the change of register values.

/sys/kernel/debug/hisi_sec2/<bdf>/qm/err_irq

Defined on file *debugfs-hisi-sec*

Dump the number of invalid interrupts for QM task completion. Available for both PF and VF, and take no other effect on SEC.

/sys/kernel/debug/hisi_sec2/<bdf>/qm/mb_err

Defined on file *debugfs-hisi-sec*

Dump the number of failed QM mailbox commands. Available for both PF and VF, and take no other effect on SEC.

/sys/kernel/debug/hisi_sec2/<bdf>/qm/qm_regs

Defined on file *debugfs-hisi-sec*

Dump of QM related debug registers. Available for PF and VF in host. VF in guest currently only has one debug register.

/sys/kernel/debug/hisi_sec2/<bdf>/qm/status

Defined on file *debugfs-hisi-sec*

Dump the status of the QM. Two states: work, stop. Available for both PF and VF, and take no other effect on SEC.

/sys/kernel/debug/hisi_sec2/<bdf>/sec_dfx/diff_regs

Defined on file *debugfs-hisi-sec*

SEC debug registers(regs) read hardware register value. This node is used to show the change of the register values. This node can be help users to check the change of register values.

/sys/kernel/debug/hisi_sec2/<bdf>/sec_dfx/done_flag_cnt

Defined on file *debugfs-hisi-sec*

Dump the total number of completed but marked error requests to be received. Available for both PF and VF, and take no other effect on SEC.

/sys/kernel/debug/hisi_sec2/<bdf>/sec_dfx/err_bd_cnt

Defined on file *debugfs-hisi-sec*

Dump the total number of BD type error requests to be received. Available for both PF and VF, and take no other effect on SEC.

/sys/kernel/debug/hisi_sec2/<bdf>/sec_dfx/invalid_req_cnt

Defined on file *debugfs-hisi-sec*

Dump the total number of invalid requests being received. Available for both PF and VF, and take no other effect on SEC.

/sys/kernel/debug/hisi_sec2/<bdf>/sec_dfx/recv_cnt

Defined on file *debugfs-hisi-sec*

Dump the total number of received requests. Available for both PF and VF, and take no other effect on SEC.

/sys/kernel/debug/hisi_sec2/<bdf>/sec_dfx/send_busy_cnt

Defined on file *debugfs-hisi-sec*

Dump the total number of requests sent with returning busy. Available for both PF and VF, and take no other effect on SEC.

/sys/kernel/debug/hisi_sec2/<bdf>/sec_dfx/send_cnt

Defined on file *debugfs-hisi-sec*

Dump the total number of sent requests. Available for both PF and VF, and take no other effect on SEC.

/sys/kernel/debug/hisi_zip/<bdf>/alg_qos

Defined on file *debugfs-hisi-zip*

The <bdf> is related the function for PF and VF. ZIP driver supports to configure each function's QoS, the driver supports to write <bdf> value to alg_qos in the host. Such as "echo <bdf> value > alg_qos". The qos value is 1~1000, means 1/1000~1000/1000 of total QoS. The driver reading alg_qos to get related QoS in the host and VM, Such as "cat alg_qos".

/sys/kernel/debug/hisi_zip/<bdf>/clear_enable

Defined on file *debugfs-hisi-zip*

Compression/decompression core debug registers read clear control. 1 means enable register read clear, otherwise 0. Writing to this file has no functional effect, only enable or disable counters clear after reading of these registers. Only available for PF.

/sys/kernel/debug/hisi_zip/<bdf>/comp_core[01]/regs

Defined on file *debugfs-hisi-zip*

Dump of compression cores related debug registers. Only available for PF.

/sys/kernel/debug/hisi_zip/<bdf>/current_qm

Defined on file *debugfs-hisi-zip*

One ZIP controller has one PF and multiple VFs, each function has a QM. Select the QM which below qm refers to. Only available for PF.

/sys/kernel/debug/hisi_zip/<bdf>/decomp_core[0-5]/regs

Defined on file *debugfs-hisi-zip*

Dump of decompression cores related debug registers. Only available for PF.

/sys/kernel/debug/hisi_zip/<bdf>/qm/abnormal_irq

Defined on file *debugfs-hisi-zip*

Dump the number of interrupts for QM abnormal event. Available for both PF and VF, and take no other effect on ZIP.

/sys/kernel/debug/hisi_zip/<bdf>/qm/aeq_irq

Defined on file *debugfs-hisi-zip*

Dump the number of QM async event queue interrupts. Available for both PF and VF, and take no other effect on ZIP.

/sys/kernel/debug/hisi_zip/<bdf>/qm/clear_enable

Defined on file *debugfs-hisi-zip*

QM debug registers(regs) read clear control. 1 means enable register read clear, otherwise 0. Writing to this file has no functional effect, only enable or disable counters clear after reading of these registers. Only available for PF.

/sys/kernel/debug/hisi_zip/<bdf>/qm/create_qp_err

Defined on file *debugfs-hisi-zip*

Dump the number of queue allocation errors. Available for both PF and VF, and take no other effect on ZIP.

/sys/kernel/debug/hisi_zip/<bdf>/qm/current_q

Defined on file *debugfs-hisi-zip*

One QM may contain multiple queues. Select specific queue to show its debug registers in above regs. Only available for PF.

/sys/kernel/debug/hisi_zip/<bdf>/qm/diff_regs

Defined on file *debugfs-hisi-zip*

QM debug registers(regs) read hardware register value. This node is used to show the change of the qm registers value. This node can be help users to check the change of register values.

/sys/kernel/debug/hisi_zip/<bdf>/qm/err_irq

Defined on file *debugfs-hisi-zip*

Dump the number of invalid interrupts for QM task completion. Available for both PF and VF, and take no other effect on ZIP.

/sys/kernel/debug/hisi_zip/<bdf>/qm/mb_err

Defined on file *debugfs-hisi-zip*

Dump the number of failed QM mailbox commands. Available for both PF and VF, and take no other effect on ZIP.

/sys/kernel/debug/hisi_zip/<bdf>/qm/regs

Defined on file *debugfs-hisi-zip*

Dump of QM related debug registers. Available for PF and VF in host. VF in guest currently only has one debug register.

/sys/kernel/debug/hisi_zip/<bdf>/qm/status

Defined on file *debugfs-hisi-zip*

Dump the status of the QM. Two states: work, stop. Available for both PF and VF, and take no other effect on ZIP.

/sys/kernel/debug/hisi_zip/<bdf>/zip_dfx/diff_regs

Defined on file *debugfs-hisi-zip*

ZIP debug registers(regs) read hardware register value. This node is used to show the change of the registers value. this node can be help users to check the change of register values.

/sys/kernel/debug/hisi_zip/<bdf>/zip_dfx/err_bd_cnt

Defined on file *debugfs-hisi-zip*

Dump the total number of BD type error requests to be received. Available for both PF and VF, and take no other effect on ZIP.

/sys/kernel/debug/hisi_zip/<bdf>/zip_dfx/recv_cnt

Defined on file *debugfs-hisi-zip*

Dump the total number of received requests. Available for both PF and VF, and take no other effect on ZIP.

/sys/kernel/debug/hisi_zip/<bdf>/zip_dfx/send_busy_cnt

Defined on file *debugfs-hisi-zip*

Dump the total number of requests received with returning busy. Available for both PF and VF, and take no other effect on ZIP.

/sys/kernel/debug/hisi_zip/<bdf>/zip_dfx/send_cnt

Defined on file *debugfs-hisi-zip*

Dump the total number of sent requests. Available for both PF and VF, and take no other effect on ZIP.

/sys/kernel/debug/hyperv/<UUID>/delay/fuzz_test_buffer_interrupt_delay

Defined on file *debugfs-hyperv*

Fuzz testing buffer interrupt delay value between 0 - 1000
microseconds (inclusive).

Users: Debugging tools

/sys/kernel/debug/hyperv/<UUID>/delay/fuzz_test_message_delay

Defined on file *debugfs-hyperv*

Fuzz testing message delay value between 0 - 1000 microseconds
(inclusive).

Users: Debugging tools

/sys/kernel/debug/hyperv/<UUID>/fuzz_test_state

Defined on file *debugfs-hyperv*

Fuzz testing status of a vmbus device, whether its in an ON state or a OFF state

Users: Debugging tools

/sys/kernel/debug/ideapad/cfg

Defined on file *debugfs-ideapad*

cfg shows the return value of _CFG method in VPC2004 device. It tells machine capability and what graphic component within the machine.

/sys/kernel/debug/ideapad/status

Defined on file *debugfs-ideapad*

status shows infos we can read and tells its meaning and value.

/sys/kernel/debug/memX/clear_poison

Defined on file *debugfs-cxl*

(WO) When a Device Physical Address (DPA) is written to this attribute, the memdev driver sends a clear poison command to the device for the specified address. Clearing poison removes the address from the device's Poison List and writes 0 (zero) for 64 bytes starting at address. It is not an error to clear poison from an address that does not have poison set. If the device cannot clear poison from the address, -ENXIO is returned. The clear_poison attribute is only visible for devices supporting the capability.

/sys/kernel/debug/moxtet/input

Defined on file *debugfs-moxtet*

(Read) Read input from the shift registers, in hexadecimal. Returns N+1 bytes, where N is the number of Moxtet connected modules. The first byte is from the CPU board itself.

Example:

```
101214
```

10	CPU board with SD card
12	2 = PCIe module, 1 = IRQ not active
14	4 = Peridot module, 1 = IRQ not active

/sys/kernel/debug/moxtet/output

Defined on file *debugfs-moxtet*

(RW) Read last written value to the shift registers, in hexadecimal, or write values to the shift registers, also in hexadecimal.

Example:

```
0102
```

01	01 was last written, or is to be written, to the first module's shift register
02	the same for second module

/sys/kernel/debug/nx-crypto/*

Defined on file *debugfs-pfo-nx-crypto*

These debugfs interfaces are built by the nx-crypto driver, built in arch/powerpc/crypto/nx.

Error Detection

errors:
A u32 providing a total count of errors since the driver was loaded. The only errors counted here are those returned from the hcall, H_COP_OP.

last_error:
The most recent non-zero return code from the H_COP_OP hcall. -EBUSY is not recorded here (the hcall will retry until -EBUSY goes away).

last_error_pid:
The process ID of the process who received the most recent error from the hcall.

Device Use

aes_bytes:
The total number of bytes encrypted using AES in any of the driver's supported modes.

aes_ops:
The total number of AES operations submitted to the hardware.

sha256_bytes:
The total number of bytes hashed by the hardware using SHA-256.

sha256_ops:
The total number of SHA-256 operations submitted to the hardware.

sha512_bytes:
The total number of bytes hashed by the hardware using SHA-512.

sha512_ops:
The total number of SHA-512 operations submitted to the hardware.

/sys/kernel/debug/olpc-ec/cmd

Defined on file *debugfs-olpc*

A generic interface for executing OLPC Embedded Controller commands and reading their responses.

To execute a command, write data with the format: CC:N A A A A CC is the (hex) command, N is the count of expected reply bytes, and A A A A are optional (hex) arguments.

To read the response (if any), read from the generic node after executing a command. Hex reply bytes will be returned, *whether or not* they came from the immediately previous command.

/sys/kernel/debug/pktcdvd/pktcdvd[0-7]

Defined on file *debugfs-pktcdvd*

The pktcdvd module (packet writing driver) creates these files in debugfs:

/sys/kernel/debug/pktcdvd/pktcdvd[0-7]/

info	0444	Lots of driver statistics and infos.

Example:

```
cat /sys/kernel/debug/pktcdvd/pktcdvd0/info
```

/sys/kernel/debug/powerpc/memtrace

Defined on file *ppc-memtrace*

This folder contains the relevant debugfs files for the hardware trace macro to use. CONFIG_PPC64_HARDWARE_TRACING must be set.

/sys/kernel/debug/powerpc/memtrace/<node-id>

Defined on file *ppc-memtrace*

This directory contains information about the removed memory from the specific NUMA node.

/sys/kernel/debug/powerpc/memtrace/<node-id>/size

Defined on file *ppc-memtrace*

This contains the size of the memory removed from the node.

/sys/kernel/debug/powerpc/memtrace/<node-id>/start

Defined on file *ppc-memtrace*

This contains the start address of the removed memory.

/sys/kernel/debug/powerpc/memtrace/<node-id>/trace

Defined on file *ppc-memtrace*

This is where the hardware trace macro will output the trace it generates.

/sys/kernel/debug/powerpc/memtrace/enable

Defined on file *ppc-memtrace*

Write an integer containing the size in bytes of the memory you want removed from each NUMA node to this file - it must be aligned to the memblock size. This amount of RAM will be removed from each NUMA node in the kernel mappings and the following debugfs files will be created. Once memory is successfully removed from each node, the following files are created. To re-add memory to the kernel, echo 0 into this file (it will be automatically onlined).

/sys/kernel/debug/qat_<device>_<BDF>/cnv_errors

Defined on file *debugfs-driver-qat*

(RO) Read returns, for each Acceleration Engine (AE), the number of errors and the type of the last error detected by the device when performing verified compression. Reported counters:

```
<N>: Number of Compress and Verify (CnV) errors and type
     of the last CnV error detected by Acceleration
     Engine N.
```

/sys/kernel/debug/qat_<device>_<BDF>/fw_counters

Defined on file *debugfs-driver-qat*

(RO) Read returns the number of requests sent to the FW and the number of responses received from the FW for each Acceleration Engine Reported firmware counters:

```
<N>: Number of requests sent from Acceleration Engine N to FW and responses
     Acceleration Engine N received from FW
```

/sys/kernel/debug/qat_<device>_<BDF>/heartbeat/config

Defined on file *debugfs-driver-qat*

(RW) Read returns value of the Heartbeat update period. Write to the file changes this period value.

This period should reflect planned polling interval of device health status. High frequency Heartbeat monitoring wastes CPU cycles but minimizes the customer's system downtime. Also, if there are large service requests that take some time to complete, high frequency Heartbeat

monitoring could result in false reports of unresponsiveness and in those cases, period needs to be increased.

This parameter is effective only for c3xxx, c62x, dh895xcc devices. 4xxx has this value internally fixed to 200ms.

Default value is set to 500. Minimal allowed value is 200. All values are expressed in milliseconds.

/sys/kernel/debug/qat_<device>_<BDF>/heartbeat/queries_failed

Defined on file *debugfs-driver-qat*

(RO) Read returns the number of times the device became unresponsive.

Attribute returns value of the counter which is incremented when status query results negative.

/sys/kernel/debug/qat_<device>_<BDF>/heartbeat/queries_sent

Defined on file *debugfs-driver-qat*

(RO) Read returns the number of times the control process checked if the device is responsive.

Attribute returns value of the counter which is incremented on every status query.

/sys/kernel/debug/qat_<device>_<BDF>/heartbeat/status

Defined on file *debugfs-driver-qat*

(RO) Read returns the device health status.

Returns 0 when device is healthy or -1 when is unresponsive or the query failed to send.

The driver does not monitor for Heartbeat. It is left for a user to poll the status periodically.

/sys/kernel/debug/qat_<device>_<BDF>/pm_status

Defined on file *debugfs-driver-qat*

(RO) Read returns power management information specific to the QAT device.

This attribute is only available for qat_4xxx devices.

/sys/kernel/debug/qat_<device>_<BDF>/telemetry/control

Defined on file *debugfs-driver-qat_telemetry*

(RW) Enables/disables the reporting of telemetry metrics.

Allowed values to write:

- 0: disable telemetry
- 1: enable telemetry
- 2, 3, 4: enable telemetry and calculate minimum, maximum and average for each counter over 2, 3 or 4 samples

Returned values:

- 1-4: telemetry is enabled and running
- 0: telemetry is disabled

Example.

Writing '3' to this file starts the collection of telemetry metrics. Samples are collected every second and stored in a circular buffer of size 3. These values are then used to calculate the minimum, maximum and average for each counter. After enabling, counters can be retrieved through the `device_data` file:

```
echo 3 > /sys/kernel/debug/qat_4xxx_0000:6b:00.0/telemetry/control
```

Writing '0' to this file stops the collection of telemetry metrics:

```
echo 0 > /sys/kernel/debug/qat_4xxx_0000:6b:00.0/telemetry/control
```

This attribute is only available for qat_4xxx devices.

/sys/kernel/debug/qat_<device>_<BDF>/telemetry/device_data

Defined on file *debugfs-driver-qat_telemetry*

(RO) Reports device telemetry counters. Reads report metrics about performance and utilization of a QAT device:

Field	Description
sample_cnt	number of acquisitions of telemetry data from the device. Reads are performed every 1000 ms.
pci_trans_cnt	number of PCIe partial transactions
max_rd_lat	maximum logged read latency [ns] (could be any read operation)
rd_lat_acc_avg	average read latency [ns]
max_gp_lat	max get to put latency [ns] (only takes samples for AE0)
gp_lat_acc_avg	average get to put latency [ns]
bw_in	PCIe, write bandwidth [Mbps]
bw_out	PCIe, read bandwidth [Mbps]
at_page_req_lat_avg	Address Translator(AT), average page request latency [ns]
at_trans_lat_avg	AT, average page translation latency [ns]
at_max_tlb_used	AT, maximum uTLB used
util_cpr<N>	utilization of Compression slice N [%]
exec_cpr<N>	execution count of Compression slice N
util_xlt<N>	utilization of Translator slice N [%]
exec_xlt<N>	execution count of Translator slice N
util_dcpr<N>	utilization of Decompression slice N [%]
exec_dcpr<N>	execution count of Decompression slice N
util_pke<N>	utilization of PKE N [%]
exec_pke<N>	execution count of PKE N
util_ucs<N>	utilization of UCS slice N [%]
exec_ucs<N>	execution count of UCS slice N
util_wat<N>	utilization of Wireless Authentication slice N [%]
exec_wat<N>	execution count of Wireless Authentication slice N
util_wcp<N>	utilization of Wireless Cipher slice N [%]
exec_wcp<N>	execution count of Wireless Cipher slice N
util_cph<N>	utilization of Cipher slice N [%]
exec_cph<N>	execution count of Cipher slice N
util_ath<N>	utilization of Authentication slice N [%]
exec_ath<N>	execution count of Authentication slice N

The telemetry report file can be read with the following command:

```
cat /sys/kernel/debug/qat_4xxx_0000:6b:00.0/telemetry/device_data
```

If `control` is set to 1, only the current values of the counters are displayed:

```
<counter_name> <current>
```

If `control` is 2, 3 or 4, counters are displayed in the following format:

```
<counter_name> <current> <min> <max> <avg>
```

If a device lacks of a specific accelerator, the corresponding attribute is not reported.

This attribute is only available for qat_4xxx devices.

/sys/kernel/debug/qat_<device>_<BDF>/telemetry/rp_<A/B/C/D>_data

Defined on file *debugfs-driver-qat_telemetry*

(RW) Selects up to 4 Ring Pairs (RP) to monitor, one per file, and report telemetry counters related to each.

Allowed values to write:

- 0 to `<num_rps - 1>`: Ring pair to be monitored. The value of `num_rps` can be retrieved through `/sys/bus/pci/devices/<BDF>/qat/num_rps`. See *testing/sysfs-driver-qat*.

Reads report metrics about performance and utilization of the selected RP:

Field	Description
sample_cnt	number of acquisitions of telemetry data from the device. Reads are performed every 1000 ms
rp_num	RP number associated with slot <A/B/C/D>
service_type	service associated to the RP
pci_trans_cnt	number of PCIe partial transactions
gp_lat_acc_avg	average get to put latency [ns]
bw_in	PCIe, write bandwidth [Mbps]
bw_out	PCIe, read bandwidth [Mbps]
at_glob_devtlb_hit	Message descriptor DevTLB hit rate
at_glob_devtlb_miss	Message descriptor DevTLB miss rate
tl_at_payld_devtlb_hit	Payload DevTLB hit rate
tl_at_payld_devtlb_miss	Payload DevTLB miss rate

Example.

Writing the value '32' to the file `rp_C_data` starts the collection of telemetry metrics for ring pair 32:

```
echo 32 > /sys/kernel/debug/qat_4xxx_0000:6b:00.0/telemetry/rp_C_data
```

Once a ring pair is selected, statistics can be read accessing the file:

```
cat /sys/kernel/debug/qat_4xxx_0000:6b:00.0/telemetry/rp_C_data
```

If `control` is set to 1, only the current values of the counters are displayed:

```
<counter_name> <current>
```

If `control` is 2, 3 or 4, counters are displayed in the following format:

```
<counter_name> <current> <min> <max> <avg>
```

On QAT GEN4 devices there are 64 RPs on a PF, so the allowed values are 0..63. This number is absolute to the device. If Virtual Functions (VF) are used, the ring pair number can be derived from the Bus, Device, Function of the VF:

PCI BDF/VF	RP0	RP1	RP2	RP3
0000:6b:0.1	RP 0	RP 1	RP 2	RP 3
0000:6b:0.2	RP 4	RP 5	RP 6	RP 7
0000:6b:0.3	RP 8	RP 9	RP 10	RP 11
0000:6b:0.4	RP 12	RP 13	RP 14	RP 15
0000:6b:0.5	RP 16	RP 17	RP 18	RP 19
0000:6b:0.6	RP 20	RP 21	RP 22	RP 23
0000:6b:0.7	RP 24	RP 25	RP 26	RP 27
0000:6b:1.0	RP 28	RP 29	RP 30	RP 31
0000:6b:1.1	RP 32	RP 33	RP 34	RP 35
0000:6b:1.2	RP 36	RP 37	RP 38	RP 39
0000:6b:1.3	RP 40	RP 41	RP 42	RP 43
0000:6b:1.4	RP 44	RP 45	RP 46	RP 47
0000:6b:1.5	RP 48	RP 49	RP 50	RP 51
0000:6b:1.6	RP 52	RP 53	RP 54	RP 55
0000:6b:1.7	RP 56	RP 57	RP 58	RP 59
0000:6b:2.0	RP 60	RP 61	RP 62	RP 63

The mapping is only valid for the BDFs of VFs on the host.

The service provided on a ring-pair varies depending on the configuration. The configuration for a given device can be queried and set using `cfg_services`. See *testing/sysfs-driver-qat* for details.

The following table reports how ring pairs are mapped to VFs on the PF 0000:6b:0.0 configured for *sym;asym* or *asym;sym*:

PCI BDF/VF	RP0/service	RP1/service	RP2/service	RP3/service
0000:6b:0.1	RP 0 asym	RP 1 sym	RP 2 asym	RP 3 sym
0000:6b:0.2	RP 4 asym	RP 5 sym	RP 6 asym	RP 7 sym
0000:6b:0.3	RP 8 asym	RP 9 sym	RP10 asym	RP11 sym
...	...	...	...	...

All VFs follow the same pattern.

The following table reports how ring pairs are mapped to VFs on the PF 0000:6b:0.0 configured for *dc*:

PCI BDF/VF	RP0/service	RP1/service	RP2/service	RP3/service
0000:6b:0.1	RP 0 dc	RP 1 dc	RP 2 dc	RP 3 dc
0000:6b:0.2	RP 4 dc	RP 5 dc	RP 6 dc	RP 7 dc
0000:6b:0.3	RP 8 dc	RP 9 dc	RP10 dc	RP11 dc
...	...	...	...	...

The mapping of a RP to a service can be retrieved using `rp2srv` from sysfs. See *testing/sysfs-driver-qat* for details.

This attribute is only available for qat_4xxx devices.

/sys/kernel/debug/scmi/<n>/atomic_threshold_us

Defined on file *debugfs-scmi*

An optional time value, expressed in microseconds, representing, on this SCMI instance <n>, the threshold above which any SCMI command, advertised to have an higher-than-threshold execution latency, should not be considered for atomic mode of operation, even if requested.

Users: Debugging, any userspace test suite

/sys/kernel/debug/scmi/<n>/instance_name

Defined on file *debugfs-scmi*

The name of the underlying SCMI instance <n> described by all the debugfs accessors rooted at /sys/kernel/debug/scmi/<n>, expressed as the full name of the top DT SCMI node under which this SCMI instance is rooted.

Users: Debugging, any userspace test suite

/sys/kernel/debug/scmi/<n>/raw/channels/<m>/message

Defined on file *debugfs-scmi-raw*

SCMI Raw synchronous message injection/snooping facility; write a complete SCMI synchronous command message (header included) in little-endian binary format to have it sent to the configured backend SCMI server for instance <n> through the <m> transport channel. Any subsequently received response can be read from this same entry if it arrived on channel <m> within the configured timeout. Each write to the entry causes one command request to be built and sent while the replies are read back one message at time (receiving an EOF at each message boundary). Channel identifier <m> matches the SCMI protocol number which has been associated with this transport channel in the DT description, with base protocol number 0x10 being the default channel for this instance. Note that these per-channel entries rooted at <..>/channels exist only if the transport is configured to have more than one default channel.

Users: Debugging, any userspace test suite

/sys/kernel/debug/scmi/<n>/raw/channels/<m>/message_async

Defined on file *debugfs-scmi-raw*

SCMI Raw asynchronous message injection/snooping facility; write a complete SCMI asynchronous command message (header included) in little-endian binary format to have it sent to the configured backend SCMI server for instance <n> through the <m> transport channel. Any subsequently received response can be read from this same entry if it arrived on channel <m> within the configured timeout. Any additional delayed response received afterwards can be read from this same entry too if it arrived within the configured timeout. Each write to the entry causes one command request to be built and sent while the replies are read back one message at time (receiving an EOF at each message boundary). Channel identifier <m> matches the SCMI protocol number which has been associated with this transport channel in the DT description, with base protocol number 0x10 being the default channel for this instance. Note that these per-channel entries rooted at <..>/channels exist only if the transport is configured to have more than one default channel.

Users: Debugging, any userspace test suite

/sys/kernel/debug/scmi/<n>/raw/errors

Defined on file *debugfs-scmi-raw*

SCMI Raw message errors facility; any kind of timed-out or generally unexpectedly received SCMI message, for instance <n>, can be read from this entry. Each read gives back one message at time (receiving an EOF at each message boundary).

Users: Debugging, any userspace test suite

/sys/kernel/debug/scmi/<n>/raw/message

Defined on file *debugfs-scmi-raw*

SCMI Raw synchronous message injection/snooping facility; write a complete SCMI synchronous command message (header included) in little-endian binary format to have it sent to the configured backend SCMI server for instance <n>. Any subsequently received response can be read from this same entry if it arrived within the configured timeout. Each write to the entry causes one command request to be built and sent while the replies are read back one message at time (receiving an EOF at each message boundary).

Users: Debugging, any userspace test suite

/sys/kernel/debug/scmi/<n>/raw/message_async

Defined on file *debugfs-scmi-raw*

SCMI Raw asynchronous message injection/snooping facility; write a complete SCMI asynchronous command message (header included) in little-endian binary format to have it sent to the configured backend SCMI server for instance <n>. Any subsequently received response can be read from this same entry if it arrived within the configured timeout. Any additional delayed response received afterwards can be read from this same entry too if it arrived within the configured timeout. Each write to the entry causes one command request to be built and sent while the replies are read back one message at time (receiving an EOF at each message boundary).

Users: Debugging, any userspace test suite

/sys/kernel/debug/scmi/<n>/raw/notification

Defined on file *debugfs-scmi-raw*

SCMI Raw notification snooping facility; any notification emitted by the backend SCMI server, for instance <n>, can be read from this entry. Each read gives back one message at time (receiving an EOF at each message boundary).

Users: Debugging, any userspace test suite

/sys/kernel/debug/scmi/<n>/raw/reset

Defined on file *debugfs-scmi-raw*

SCMI Raw stack reset facility; writing a value to this entry causes the internal queues of any kind of received message, still pending to be read out for instance <n>, to be immediately flushed. Can be used to reset and clean the SCMI Raw stack between to different test-run.

Users: Debugging, any userspace test suite

/sys/kernel/debug/scmi/<n>/transport/is_atomic

Defined on file *debugfs-scmi*

A boolean stating if the transport configured on the underlying SCMI instance <n> is capable of atomic mode of operation.

Users: Debugging, any userspace test suite

/sys/kernel/debug/scmi/<n>/transport/max_msg_size

Defined on file *debugfs-scmi*

Max message size of allowed SCMI messages for the currently configured SCMI transport for instance <n>.

Users: Debugging, any userspace test suite

/sys/kernel/debug/scmi/<n>/transport/max_rx_timeout_ms

Defined on file *debugfs-scmi*

Timeout in milliseconds allowed for SCMI synchronous replies for the currently configured SCMI transport for instance <n>.

Users: Debugging, any userspace test suite

/sys/kernel/debug/scmi/<n>/transport/rx_max_msg

Defined on file *debugfs-scmi*

Max number of concurrently allowed in-flight SCMI messages for the currently configured SCMI transport for instance <n> on the RX channels.

Users: Debugging, any userspace test suite

/sys/kernel/debug/scmi/<n>/transport/tx_max_msg

Defined on file *debugfs-scmi*

Max number of concurrently allowed in-flight SCMI messages for the currently configured SCMI transport for instance <n> on the TX channels.

Users: Debugging, any userspace test suite

/sys/kernel/debug/scmi/<n>/transport/type

Defined on file *debugfs-scmi*

A string representing the type of transport configured for this SCMI instance <n>.

Users: Debugging, any userspace test suite

/sys/kernel/debug/tpmi-<n>/pfs_dump

Defined on file *debugfs-tpmi*

The PFS (PM Feature Structure) table, shows details of each power management feature. This includes: tpmi_id, number of entries, entry size, offset, vsec offset, lock status and disabled status.

Users: Debugging, any user space test suite

/sys/kernel/debug/tpmi-<n>/tpmi-id-<n>/mem_dump

Defined on file *debugfs-tpmi*

Shows the memory dump of the MMIO region for a TPMI ID.

Users: Debugging, any user space test suite

/sys/kernel/debug/tpmi-<n>/tpmi-id-<n>/mem_write

Defined on file *debugfs-tpmi*

Allows to write at any offset. It doesn't check for Read/Write access as hardware will not allow to write at read-only memory. This write is at offset multiples of 4. The format is instance,offset,contents. Example: echo 0,0x20,0xff > mem_write echo 1,64,64 > mem_write

Users: Debugging, any user space test suite

/sys/kernel/debug/turris-mox-rwtm/do_sign

Defined on file *debugfs-turris-mox-rwtm*

(Write)	Message to sign with the ECDSA private key stored in device's OTP. The message must be exactly 64 bytes (since this is intended for SHA-512 hashes).
(Read)	The resulting signature, 136 bytes. This contains the R and S values of the ECDSA signature, both in big-endian format.

/sys/kernel/debug/vfio

Defined on file *debugfs-vfio*

This debugfs file directory is used for debugging of vfio devices, it's a common directory for all vfio devices. Vfio core will create a device subdirectory under this directory.

/sys/kernel/debug/vfio/<device>/migration

Defined on file *debugfs-vfio*

This debugfs file directory is used for debugging of vfio devices that support live migration. The debugfs of each vfio device that supports live migration could be created under this directory.

/sys/kernel/debug/vfio/<device>/migration/state

Defined on file *debugfs-vfio*

Read the live migration status of the vfio device. The contents of the state file reflects the migration state relative to those defined in the vfio_device_mig_state enum

/sys/kernel/debug/wilco_ec/h1_gpio

Defined on file *debugfs-wilco-ec*

As part of Chrome OS's FAFT (Fully Automated Firmware Testing) tests, we need to ensure that the H1 chip is properly setting some GPIO lines. The h1_gpio attribute exposes the state of the lines: - ENTRY_TO_FACT_MODE in BIT(0) - SPI_CHROME_SEL in BIT(1)

Output will formatted with "0x%02xn".

/sys/kernel/debug/wilco_ec/raw

Defined on file *debugfs-wilco-ec*

Write and read raw mailbox commands to the EC.

You can write a hexadecimal sentence to raw, and that series of bytes will be sent to the EC. Then, you can read the bytes of response by reading from raw.

For writing, bytes 0-1 indicate the message type, one of enum wilco_ec_msg_type. Byte 2+ consist of the data passed in the request, starting at MBOX[0]. At least three bytes are required for writing, two for the type and at least a single byte of data.

Example:

```
// Request EC info type 3 (EC firmware build date)
// Corresponds with sending type 0x00f0 with
// MBOX = [38, 00, 03, 00]
$ echo 00 f0 38 00 03 00 > /sys/kernel/debug/wilco_ec/raw
// View the result. The decoded ASCII result "12/21/18" is
// included after the raw hex.
// Corresponds with MBOX = [00, 00, 31, 32, 2f, 32, 31, 38, ...]
$ cat /sys/kernel/debug/wilco_ec/raw
00 00 31 32 2f 32 31 2f 31 38 00 38 00 01 00 2f 00  ..12/21/18.8...
```

Note that the first 16 bytes of the received MBOX[] will be printed, even if some of the data is junk, and skipping bytes 17 to 32. It is up to you to know how many of the first bytes of data are the actual response.

/sys/kernel/dmabuf/buffers

Defined on file *sysfs-kernel-dmabuf-buffers*

The */sys/kernel/dmabuf/buffers* directory contains a snapshot of the internal state of every DMA-BUF. /sys/kernel/dmabuf/buffers/<inode_number> will contain the statistics for the DMA-BUF with the unique inode number <inode_number>

Users: kernel memory tuning/debugging tools

/sys/kernel/dmabuf/buffers/<inode_number>/exporter_name

Defined on file *sysfs-kernel-dmabuf-buffers*

This file is read-only and contains the name of the exporter of the DMA-BUF.

/sys/kernel/dmabuf/buffers/<inode_number>/size

Defined on file *sysfs-kernel-dmabuf-buffers*

This file is read-only and specifies the size of the DMA-BUF in bytes.

/sys/kernel/fadump/*

Defined on file *sysfs-kernel-fadump*

The /sys/kernel/fadump/* is a collection of FADump sysfs file provide information about the configuration status of Firmware Assisted Dump (FADump).

/sys/kernel/fadump/enabled

Defined on file *sysfs-kernel-fadump*

read only Primarily used to identify whether the FADump is enabled in the kernel or not. User: Kdump service

/sys/kernel/fadump/mem_reserved

Defined on file *sysfs-kernel-fadump*

read only Provide information about the amount of memory reserved by FADump to save the crash dump in bytes.

/sys/kernel/fadump/registered

Defined on file *sysfs-kernel-fadump*

read/write Helps to control the dump collect feature from userspace. Setting 1 to this file enables the system to collect the dump and 0 to disable it. User: Kdump service

/sys/kernel/fadump/release_mem

Defined on file *sysfs-kernel-fadump*

write only This is a special sysfs file and only available when the system is booted to capture the vmcore using FADump. It is used to release the memory reserved by FADump to save the crash dump.

/sys/kernel/fscaps

Defined on file *sysfs-kernel-fscaps*

Shows whether file system capabilities are honored when executing a binary

/sys/kernel/iommu_groups/

Defined on file *sysfs-kernel-iommu_groups*

/sys/kernel/iommu_groups/ contains a number of sub- directories, each representing an IOMMU group. The name of the sub-directory matches the iommu_group_id() for the group, which is an integer value. Within each subdirectory is another directory named "devices" with links to the sysfs devices contained in this group. The group directory also optionally contains a "name" file if the IOMMU driver has chosen to register a more common name for the group.

/sys/kernel/iommu_groups/<grp_id>/type

Defined on file *sysfs-kernel-iommu_groups*

/sys/kernel/iommu_groups/<grp_id>/type shows the type of default domain in use by iommu for this group. See include/linux/iommu.h for possible read values. A privileged user could request kernel to change the group type by writing to this file. Valid write values:

DMA	All the DMA transactions from the device in this group are translated by the iommu.
DMA-FQ	As above, but using batched invalidation to lazily remove translations after use. This may offer reduced overhead at the cost of reduced memory protection.
identity	All the DMA transactions from the device in this group are not translated by the iommu. Maximum performance but zero protection.
auto	Change to the type the device was booted with.

The default domain type of a group may be modified only when

- The device in the group is not bound to any device driver. So, the users must unbind the appropriate driver before changing the default domain type.

Unbinding a device driver will take away the driver's control over the device and if done on devices that host root file system could lead to catastrophic effects (the users might need to reboot the machine to get it to normal state). So, it's expected that the users understand what they're doing.

/sys/kernel/iommu_groups/reserved_regions

Defined on file *sysfs-kernel-iommu_groups*

/sys/kernel/iommu_groups/reserved_regions list IOVA regions that are reserved. Not necessarily all reserved regions are listed. This is typically used to output direct-mapped, MSI, non mappable regions. Each region is described on a single line: the 1st field is the base IOVA, the second is the end IOVA and the third field describes the type of the region.

Since kernel 5.3, in case an RMRR is used only by graphics or USB devices it is now exposed as "direct-relaxable" instead of "direct". In device assignment use case, for instance, those RMRR are considered to be relaxable and safe.

/sys/kernel/irq

Defined on file *sysfs-kernel-irq*

Directory containing information about the system's IRQs. Specifically, data from the associated struct irq_desc. The information here is similar to that in /proc/interrupts but in a more machine-friendly format. This directory contains one subdirectory for each Linux IRQ number.

/sys/kernel/irq/<irq>/actions

Defined on file *sysfs-kernel-irq*

The IRQ action chain. A comma-separated list of zero or more device names associated with this interrupt.

/sys/kernel/irq/<irq>/chip_name

Defined on file *sysfs-kernel-irq*

Human-readable chip name supplied by the associated device driver.

/sys/kernel/irq/<irq>/hwirq

Defined on file *sysfs-kernel-irq*

When interrupt translation domains are used, this file contains the underlying hardware IRQ number used for this Linux IRQ.

/sys/kernel/irq/<irq>/name

Defined on file *sysfs-kernel-irq*

Human-readable flow handler name as defined by the irq chip driver.

/sys/kernel/irq/<irq>/per_cpu_count

Defined on file *sysfs-kernel-irq*

The number of times the interrupt has fired since boot. This is a comma-separated list of counters; one per CPU in CPU id order. NOTE: This file consistently shows counters for all CPU ids. This differs from the behavior of /proc/interrupts which only shows counters for online CPUs.

/sys/kernel/irq/<irq>/type

Defined on file *sysfs-kernel-irq*

The type of the interrupt. Either the string 'level' or 'edge'.

/sys/kernel/irq/<irq>/wakeup

Defined on file *sysfs-kernel-irq*

The wakeup state of the interrupt. Either the string 'enabled' or 'disabled'.

/sys/kernel/livepatch

Defined on file *sysfs-kernel-livepatch*

Interface for kernel live patching

The */sys/kernel/livepatch* directory contains subdirectories for each loaded live patch module.

/sys/kernel/livepatch/<patch>

Defined on file *sysfs-kernel-livepatch*

The patch directory contains subdirectories for each kernel object (vmlinux or a module) in which it patched functions.

/sys/kernel/livepatch/<patch>/<object>

Defined on file *sysfs-kernel-livepatch*

The object directory contains subdirectories for each function that is patched within the object.

/sys/kernel/livepatch/<patch>/<object>/<function,sympos>

Defined on file *sysfs-kernel-livepatch*

The function directory contains attributes regarding the properties and state of the patched function.

The directory name contains the patched function name and a sympos number corresponding to the nth occurrence of the symbol name in kallsyms for the patched object.

There are currently no such attributes.

/sys/kernel/livepatch/<patch>/<object>/patched

Defined on file *sysfs-kernel-livepatch*

An attribute which indicates whether the object is currently patched.

/sys/kernel/livepatch/<patch>/enabled

Defined on file *sysfs-kernel-livepatch*

A writable attribute that indicates whether the patched code is currently applied. Writing 0 will disable the patch while writing 1 will re-enable the patch.

/sys/kernel/livepatch/<patch>/force

Defined on file *sysfs-kernel-livepatch*

A writable attribute that allows administrator to affect the course of an existing transition. Writing 1 clears TIF_PATCH_PENDING flag of all tasks and thus forces the tasks to the patched or unpatched state. Administrator should not use this feature without a clearance from a patch distributor. Removal (rmmod) of patch modules is permanently disabled when the feature is used. See /livepatch/livepatch for more information.

/sys/kernel/livepatch/<patch>/transition

Defined on file *sysfs-kernel-livepatch*

An attribute which indicates whether the patch is currently in transition.

/sys/kernel/mm

Defined on file *sysfs-kernel-mm*

/sys/kernel/mm/ should contain any and all VM related information in /sys/kernel/.

/sys/kernel/mm/cma/

Defined on file *sysfs-kernel-mm-cma*

/sys/kernel/mm/cma/ contains a subdirectory for each CMA heap name (also sometimes called CMA areas).

Each CMA heap subdirectory (that is, each /sys/kernel/mm/cma/<cma-heap-name> directory) contains the following items:

alloc_pages_success alloc_pages_fail

/sys/kernel/mm/cma/<cma-heap-name>/alloc_pages_fail

Defined on file *sysfs-kernel-mm-cma*

the number of pages CMA API failed to allocate

/sys/kernel/mm/cma/<cma-heap-name>/alloc_pages_success

Defined on file *sysfs-kernel-mm-cma*

the number of pages CMA API succeeded to allocate

/sys/kernel/mm/damon/

Defined on file *sysfs-kernel-mm-damon*

Interface for Data Access MONitoring (DAMON). Contains files for controlling DAMON. For more details on DAMON itself, please refer to *DAMON: Data Access MONitor*.

/sys/kernel/mm/damon/admin/

Defined on file *sysfs-kernel-mm-damon*

Interface for privileged users of DAMON. Contains files for controlling DAMON that aimed to be used by privileged users.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/avail_operations

Defined on file *sysfs-kernel-mm-damon*

Reading this file returns the available monitoring operations sets on the currently running kernel.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/monitoring_attrs/intervals/agg

Defined on file *sysfs-kernel-mm-damon*

Writing a value to this file sets the aggregation interval of the DAMON context in microseconds as the value. Reading this file returns the value.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/monitoring_attrs/intervals/san

Defined on file *sysfs-kernel-mm-damon*

Writing a value to this file sets the sampling interval of the DAMON context in microseconds as the value. Reading this file returns the value.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/monitoring_attrs/intervals/upd

Defined on file *sysfs-kernel-mm-damon*

Writing a value to this file sets the update interval of the DAMON context in microseconds as the value. Reading this file returns the value.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/monitoring_attrs/nr_regions/m

Defined on file *sysfs-kernel-mm-damon*

Writing a value to this file sets the maximum number of monitoring regions of the DAMON context as the value. Reading this file returns the value.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/monitoring_attrs/nr_regions/m

Defined on file *sysfs-kernel-mm-damon*

Writing a value to this file sets the minimum number of monitoring regions of the DAMON context as the value. Reading this file returns the value.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/operations

Defined on file *sysfs-kernel-mm-damon*

Writing a keyword for a monitoring operations set ('vaddr' for virtual address spaces monitoring, 'fvaddr' for fixed virtual address ranges monitoring, and 'paddr' for the physical address space monitoring) to this file makes the context to use the operations set. Reading the file returns the keyword for the operations set the context is set to use.

Note that only the operations sets that listed in 'avail_operations' file are valid inputs.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/access_pattern/a

Defined on file *sysfs-kernel-mm-damon*

Writing to and reading from this file sets and gets the maximum 'age' of the scheme's target regions.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/access_pattern/a

Defined on file *sysfs-kernel-mm-damon*

Writing to and reading from this file sets and gets the minimum 'age' of the scheme's target regions.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/access_pattern/n

Defined on file *sysfs-kernel-mm-damon*

Writing to and reading from this file sets and gets the maximum 'nr_accesses' of the scheme's target regions.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/access_pattern/n

Defined on file *sysfs-kernel-mm-damon*

Writing to and reading from this file sets and gets the manimum 'nr_accesses' of the scheme's target regions.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/access_pattern/s

Defined on file *sysfs-kernel-mm-damon*

Writing to and reading from this file sets and gets the maximum size of the scheme's target regions in bytes.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/access_pattern/s

Defined on file *sysfs-kernel-mm-damon*

Writing to and reading from this file sets and gets the minimum size of the scheme's target regions in bytes.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/action

Defined on file *sysfs-kernel-mm-damon*

Writing to and reading from this file sets and gets the action of the scheme.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/apply_interval_u

Defined on file *sysfs-kernel-mm-damon*

Writing a value to this file sets the action apply interval of the scheme in microseconds. Reading this file returns the value.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/filters/<F>/addr

Defined on file *sysfs-kernel-mm-damon*

If 'addr' is written to the 'type' file, writing to or reading from this file sets or gets the end address of the address range for the filter.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/filters/<F>/addr

Defined on file *sysfs-kernel-mm-damon*

If 'addr' is written to the 'type' file, writing to or reading from this file sets or gets the start address of the address range for the filter.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/filters/<F>/matc

Defined on file *sysfs-kernel-mm-damon*

Writing 'Y' or 'N' to this file sets whether to filter out pages that do or do not match to the 'type' and 'memcg_path', respectively. Filter out means the action of the scheme will not be applied to.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/filters/<F>/mem

Defined on file *sysfs-kernel-mm-damon*

If 'memcg' is written to the 'type' file, writing to and reading from this file sets and gets the path to the memory cgroup of the interest.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/filters/<F>/targe

Defined on file *sysfs-kernel-mm-damon*

If 'target' is written to the 'type' file, writing to or reading from this file sets or gets the index of the DAMON monitoring target of the interest.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/filters/<F>/type

Defined on file *sysfs-kernel-mm-damon*

Writing to and reading from this file sets and gets the type of the memory of the interest. 'anon' for anonymous pages, 'memcg' for specific memory cgroup, 'addr' for address range (an open-ended interval), or 'target' for DAMON monitoring target can be written and read.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/filters/nr_filters

Defined on file *sysfs-kernel-mm-damon*

Writing a number 'N' to this file creates the number of directories for setting filters of the scheme named '0' to 'N-1' under the filters/ directory.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/quotas/bytes

Defined on file *sysfs-kernel-mm-damon*

Writing to and reading from this file sets and gets the size quota of the scheme in bytes.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/quotas/goals/<G

Defined on file *sysfs-kernel-mm-damon*

Writing to and reading from this file sets and gets the current value of the goal metric.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/quotas/goals/<G

Defined on file *sysfs-kernel-mm-damon*

Writing to and reading from this file sets and gets the target value of the goal metric.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/quotas/goals/nr_

Defined on file *sysfs-kernel-mm-damon*

Writing a number 'N' to this file creates the number of directories for setting automatic tuning of the scheme's aggressiveness named '0' to 'N-1' under the goals/ directory.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/quotas/ms

Defined on file *sysfs-kernel-mm-damon*

Writing to and reading from this file sets and gets the time quota of the scheme in milliseconds.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/quotas/reset_int

Defined on file *sysfs-kernel-mm-damon*

Writing to and reading from this file sets and gets the quotas charge reset interval of the scheme in milliseconds.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/quotas/weights/

Defined on file *sysfs-kernel-mm-damon*

Writing to and reading from this file sets and gets the under-quota limit regions prioritization weight for 'age' in permil.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/quotas/weights/

Defined on file *sysfs-kernel-mm-damon*

Writing to and reading from this file sets and gets the under-quota limit regions prioritization weight for 'nr_accesses' in permil.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/quotas/weights/

Defined on file *sysfs-kernel-mm-damon*

Writing to and reading from this file sets and gets the under-quota limit regions prioritization weight for 'size' in permil.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/stats/nr_applied

Defined on file *sysfs-kernel-mm-damon*

Reading this file returns the number of regions that the action of the scheme has successfully applied.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/stats/nr_tried

Defined on file *sysfs-kernel-mm-damon*

Reading this file returns the number of regions that the action of the scheme has tried to be applied.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/stats/qt_exceeds

Defined on file *sysfs-kernel-mm-damon*

Reading this file returns the number of the exceed events of the scheme's quotas.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/stats/sz_applied

Defined on file *sysfs-kernel-mm-damon*

Reading this file returns the total size of regions that the action of the scheme has successfully applied in bytes.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/stats/sz_tried

Defined on file *sysfs-kernel-mm-damon*

Reading this file returns the total size of regions that the action of the scheme has tried to be applied in bytes.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/tried_regions/<

Defined on file *sysfs-kernel-mm-damon*

Reading this file returns the 'age' of a memory region that corresponding DAMON-based Operation Scheme's action has tried to be applied.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/tried_regions/<I

Defined on file *sysfs-kernel-mm-damon*

Reading this file returns the end address of a memory region that corresponding DAMON-based Operation Scheme's action has tried to be applied.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/tried_regions/<I

Defined on file *sysfs-kernel-mm-damon*

Reading this file returns the 'nr_accesses' of a memory region that corresponding DAMON-based Operation Scheme's action has tried to be applied.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/tried_regions/<I

Defined on file *sysfs-kernel-mm-damon*

Reading this file returns the start address of a memory region that corresponding DAMON-based Operation Scheme's action has tried to be applied.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/tried_regions/to

Defined on file *sysfs-kernel-mm-damon*

Reading this file returns the total amount of memory that corresponding DAMON-based Operation Scheme's action has tried to be applied.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/watermarks/hig

Defined on file *sysfs-kernel-mm-damon*

Writing to and reading from this file sets and gets the high watermark of the scheme in permil.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/watermarks/inte

Defined on file *sysfs-kernel-mm-damon*

Writing to and reading from this file sets and gets the metric check interval of the watermarks for the scheme in microseconds.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/watermarks/low

Defined on file *sysfs-kernel-mm-damon*

Writing to and reading from this file sets and gets the low watermark of the scheme in permil.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/watermarks/met

Defined on file *sysfs-kernel-mm-damon*

Writing to and reading from this file sets and gets the metric of the watermarks for the scheme. The writable/readable keywords for this file are 'none' for disabling the watermarks feature, or 'free_mem_rate' for the system's global free memory rate in permil.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/watermarks/mid

Defined on file *sysfs-kernel-mm-damon*

Writing to and reading from this file sets and gets the mid watermark of the scheme in permil.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/nr_schemes

Defined on file *sysfs-kernel-mm-damon*

Writing a number 'N' to this file creates the number of directories for controlling each DAMON-based operation scheme of the context named '0' to 'N-1' under the schemes/ directory.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/targets/<T>/pid_target

Defined on file *sysfs-kernel-mm-damon*

Writing to and reading from this file sets and gets the pid of the target process if the context is for virtual address spaces monitoring, respectively.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/targets/<T>/regions/<R>/end

Defined on file *sysfs-kernel-mm-damon*

Writing to and reading from this file sets and gets the end address of the monitoring region.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/targets/<T>/regions/<R>/star

Defined on file *sysfs-kernel-mm-damon*

Writing to and reading from this file sets and gets the start address of the monitoring region.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/targets/<T>/regions/nr_region

Defined on file *sysfs-kernel-mm-damon*

Writing a number 'N' to this file creates the number of directories for setting each DAMON target memory region of the context named '0' to 'N-1' under the regions/ directory. In case of the virtual address space monitoring, DAMON automatically sets the target memory region based on the target processes' mappings.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/targets/nr_targets

Defined on file *sysfs-kernel-mm-damon*

Writing a number 'N' to this file creates the number of directories for controlling each DAMON target of the context named '0' to 'N-1' under the contexts/ directory.

/sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/nr_contexts

Defined on file *sysfs-kernel-mm-damon*

Writing a number 'N' to this file creates the number of directories for controlling each DAMON context named '0' to 'N-1' under the contexts/ directory.

/sys/kernel/mm/damon/admin/kdamonds/<K>/pid

Defined on file *sysfs-kernel-mm-damon*

Reading this file returns the pid of the kdamond if it is running.

/sys/kernel/mm/damon/admin/kdamonds/<K>/state

Defined on file *sysfs-kernel-mm-damon*

Writing 'on' or 'off' to this file makes the kdamond starts or stops, respectively. Reading the file returns the keywords based on the current status. Writing 'commit' to this file makes the kdamond reads the user inputs in the sysfs files except 'state' again. Writing 'commit_schemes_quota_goals' to this file makes the kdamond reads the quota goal files again. Writing 'update_schemes_stats' to the file updates contents of schemes stats files of the kdamond. Writing 'update_schemes_tried_regions' to the file updates contents of 'tried_regions' directory of every scheme directory of this kdamond. Writing 'update_schemes_tried_bytes' to the file updates only '.../tried_regions/total_bytes' files of this kdamond. Writing 'clear_schemes_tried_regions' to the file removes contents of the 'tried_regions' directory.

/sys/kernel/mm/damon/admin/kdamonds/nr_kdamonds

Defined on file *sysfs-kernel-mm-damon*

Writing a number 'N' to this file creates the number of directories for controlling each DAMON worker thread (kdamond) named '0' to 'N-1' under the kdamonds/ directory.

/sys/kernel/mm/hugepages/

Defined on file *sysfs-kernel-mm-hugepages*

/sys/kernel/mm/hugepages/ contains a number of subdirectories of the form hugepages-<size>kB, where <size> is the page size of the hugepages supported by the kernel/CPU combination.

Under these directories are a number of files:

- nr_hugepages
- nr_overcommit_hugepages
- free_hugepages

- surplus_hugepages
- resv_hugepages

See *HugeTLB Pages* for details.

/sys/kernel/mm/ksm

Defined on file *sysfs-kernel-mm-ksm*

Interface for Kernel Samepage Merging (KSM)

/sys/kernel/mm/ksm/full_scans
/sys/kernel/mm/ksm/pages_shared
/sys/kernel/mm/ksm/pages_sharing
/sys/kernel/mm/ksm/pages_to_scan
/sys/kernel/mm/ksm/pages_unshared
/sys/kernel/mm/ksm/pages_volatile
/sys/kernel/mm/ksm/run
/sys/kernel/mm/ksm/sleep_millisecs

Defined on file *sysfs-kernel-mm-ksm*

Kernel Samepage Merging daemon sysfs interface

full_scans: how many times all mergeable areas have been scanned.

pages_shared: how many shared pages are being used.

pages_sharing: how many more sites are sharing them i.e. how much saved.

pages_to_scan: how many present pages to scan before ksmd goes to sleep.

pages_unshared: how many pages unique but repeatedly checked for merging.

pages_volatile: how many pages changing too fast to be placed in a tree.

run: write 0 to disable ksm, read 0 while ksm is disabled.

- write 1 to run ksm, read 1 while ksm is running.
- write 2 to disable ksm and unmerge all its pages.

sleep_millisecs: how many milliseconds ksm should sleep between scans.

See /mm/ksm for more information.

/sys/kernel/mm/ksm/general_profit

Defined on file *sysfs-kernel-mm-ksm*

Measure how effective KSM is. general_profit: how effective is KSM. The formula for the calculation is in *Kernel Samepage Merging*.

/sys/kernel/mm/ksm/merge_across_nodes

Defined on file *sysfs-kernel-mm-ksm*

Control merging pages across different NUMA nodes.

When it is set to 0 only pages from the same node are merged, otherwise pages from all nodes can be merged together (default).

/sys/kernel/mm/numa/

Defined on file *sysfs-kernel-mm-numa*

Interface for NUMA

/sys/kernel/mm/numa/demotion_enabled

Defined on file *sysfs-kernel-mm-numa*

Enable/disable demoting pages during reclaim

Page migration during reclaim is intended for systems with tiered memory configurations. These systems have multiple types of memory with varied performance characteristics instead of plain NUMA systems where the same kind of memory is found at varied distances. Allowing page migration during reclaim enables these systems to migrate pages from fast tiers to slow tiers when the fast tier is under pressure. This migration is performed before swap. It may move data to a NUMA node that does not fall into the cpuset of the allocating process which might be construed to violate the guarantees of cpusets. This should not be enabled on systems which need strict cpuset location guarantees.

/sys/kernel/mm/swap/

Defined on file *sysfs-kernel-mm-swap*

Interface for swapping

/sys/kernel/mm/swap/vma_ra_enabled

Defined on file *sysfs-kernel-mm-swap*

Enable/disable VMA based swap readahead.

If set to true, the VMA based swap readahead algorithm will be used for swappable anonymous pages mapped in a VMA, and the global swap readahead algorithm will be still used for tmpfs etc. other users. If set to false, the global swap readahead algorithm will be used for all swappable pages.

/sys/kernel/oops_count

Defined on file *sysfs-kernel-oops_count*

Shows how many times the system has Oopsed since last boot.

/sys/kernel/profiling

Defined on file *sysfs-profiling*

/sys/kernel/profiling is the runtime equivalent of the boot-time profile= option.

You can get the same effect running:

```
echo 2 > /sys/kernel/profiling
```

as you would by issuing profile=2 on the boot command line.

/sys/kernel/reboot

Defined on file *sysfs-kernel-reboot*

Interface to set the kernel reboot behavior, similarly to what can be done via the reboot= cmd-line option. (see *The kernel's command-line parameters*)

/sys/kernel/reboot/cpu

Defined on file *sysfs-kernel-reboot*

CPU number to use to reboot.

/sys/kernel/reboot/force

Defined on file *sysfs-kernel-reboot*

Don't wait for any other CPUs on reboot and avoid anything that could hang.

/sys/kernel/reboot/mode

Defined on file *sysfs-kernel-reboot*

Reboot mode. Valid values are: cold warm hard soft gpio

/sys/kernel/reboot/type

Defined on file *sysfs-kernel-reboot*

Reboot type. Valid values are: bios acpi kbd triple efi pci

/sys/kernel/security/*/evm/evm_xattrs

Defined on file *evm*

Shows the set of extended attributes used to calculate or validate the EVM signature, and allows additional attributes to be added at runtime. Any signatures generated after additional attributes are added (and on files possessing those additional attributes) will only be valid if the same additional attributes are configured on system boot. Writing a single period (.) will lock the xattr list from any further modification.

/sys/kernel/security/*/ima/policy

Defined on file *ima_policy*

The Trusted Computing Group(TCG) runtime Integrity Measurement Architecture(IMA) maintains a list of hash values of executables and other sensitive system files loaded into the run-time of this system. At runtime, the policy can be constrained based on LSM specific data. Policies are loaded into the securityfs file ima/policy by opening the file, writing the rules one at a time and then closing the file. The new policy takes effect after the file ima/policy is closed.

IMA appraisal, if configured, uses these file measurements for local measurement appraisal.

```
rule format: action [condition ...]

action: measure | dont_measure | appraise | dont_appraise |
        audit | hash | dont_hash
condition:= base | lsm  [option]
      base:   [[func=] [mask=] [fsmagic=] [fsuuid=] [fsname=]
              [uid=] [euid=] [gid=] [egid=]
              [fowner=] [fgroup=]]
      lsm:    [[subj_user=] [subj_role=] [subj_type=]
               [obj_user=] [obj_role=] [obj_type=]]
      option: [digest_type=] [template=] [permit_directio]
              [appraise_type=] [appraise_flag=]
              [appraise_algos=] [keyrings=]
base:
      func:= [BPRM_CHECK][MMAP_CHECK][CREDS_CHECK][FILE_CHECK][MODULE_CHECK]
              [FIRMWARE_CHECK]
              [KEXEC_KERNEL_CHECK] [KEXEC_INITRAMFS_CHECK]
              [KEXEC_CMDLINE] [KEY_CHECK] [CRITICAL_DATA]
              [SETXATTR_CHECK][MMAP_CHECK_REQPROT]
      mask:= [[^]MAY_READ] [[^]MAY_WRITE] [[^]MAY_APPEND]
             [[^]MAY_EXEC]
      fsmagic:= hex value
      fsuuid:= file system UUID (e.g 8bcbe394-4f13-4144-be8e-5aa9ea2ce2f6)
      uid:= decimal value
      euid:= decimal value
      gid:= decimal value
      egid:= decimal value
      fowner:= decimal value
      fgroup:= decimal value
lsm:  are LSM specific
option:
      appraise_type:= [imasig] | [imasig|modsig] | [sigv3]
          where 'imasig' is the original or the signature
              format v2.
          where 'modsig' is an appended signature,
          where 'sigv3' is the signature format v3. (Currently
              limited to fsverity digest based signatures
              stored in security.ima xattr. Requires
              specifying "digest_type=verity" first.)

      appraise_flag:= [check_blacklist] (deprecated)
      Setting the check_blacklist flag is no longer necessary.
      All appraisal functions set it by default.
      digest_type:= verity
```

```
          Require fs-verity's file digest instead of the
          regular IMA file hash.
      keyrings:= list of keyrings
      (eg, .builtin_trusted_keys|.ima). Only valid
      when action is "measure" and func is KEY_CHECK.
      template:= name of a defined IMA template type
      (eg, ima-ng). Only valid when action is "measure".
      pcr:= decimal value
      label:= [selinux]|[kernel_info]|[data_label]
      data_label:= a unique string used for grouping and limiting critical␣
↪data.
      For example, "selinux" to measure critical data for SELinux.
      appraise_algos:= comma-separated list of hash algorithms
      For example, "sha256,sha512" to only accept to appraise
      files where the security.ima xattr was hashed with one
      of these two algorithms.

default policy:
      # PROC_SUPER_MAGIC
      dont_measure fsmagic=0x9fa0
      dont_appraise fsmagic=0x9fa0
      # SYSFS_MAGIC
      dont_measure fsmagic=0x62656572
      dont_appraise fsmagic=0x62656572
      # DEBUGFS_MAGIC
      dont_measure fsmagic=0x64626720
      dont_appraise fsmagic=0x64626720
      # TMPFS_MAGIC
      dont_measure fsmagic=0x01021994
      dont_appraise fsmagic=0x01021994
      # RAMFS_MAGIC
      dont_appraise fsmagic=0x858458f6
      # DEVPTS_SUPER_MAGIC
      dont_measure fsmagic=0x1cd1
      dont_appraise fsmagic=0x1cd1
      # BINFMTFS_MAGIC
      dont_measure fsmagic=0x42494e4d
      dont_appraise fsmagic=0x42494e4d
      # SECURITYFS_MAGIC
      dont_measure fsmagic=0x73636673
      dont_appraise fsmagic=0x73636673
      # SELINUX_MAGIC
      dont_measure fsmagic=0xf97cff8c
      dont_appraise fsmagic=0xf97cff8c
      # CGROUP_SUPER_MAGIC
      dont_measure fsmagic=0x27e0eb
      dont_appraise fsmagic=0x27e0eb
      # NSFS_MAGIC
      dont_measure fsmagic=0x6e736673
      dont_appraise fsmagic=0x6e736673
```

```
measure func=BPRM_CHECK
measure func=FILE_MMAP mask=MAY_EXEC
measure func=FILE_CHECK mask=MAY_READ uid=0
measure func=MODULE_CHECK
measure func=FIRMWARE_CHECK
appraise fowner=0
```

The default policy measures all executables in bprm_check, all files mmapped executable in file_mmap, and all files open for read by root in do_filp_open. The default appraisal policy appraises all files owned by root.

Examples of LSM specific definitions:

SELinux:

```
dont_measure obj_type=var_log_t
dont_appraise obj_type=var_log_t
dont_measure obj_type=auditd_log_t
dont_appraise obj_type=auditd_log_t
measure subj_user=system_u func=FILE_CHECK mask=MAY_READ
measure subj_role=system_r func=FILE_CHECK mask=MAY_READ
```

Smack:

```
measure subj_user=_ func=FILE_CHECK mask=MAY_READ
```

Example of measure rules using alternate PCRs:

```
measure func=KEXEC_KERNEL_CHECK pcr=4
measure func=KEXEC_INITRAMFS_CHECK pcr=5
```

Example of appraise rule allowing modsig appended signatures:

appraise func=KEXEC_KERNEL_CHECK appraise_type=imasig|modsig

Example of measure rule using KEY_CHECK to measure all keys:

measure func=KEY_CHECK

Example of measure rule using KEY_CHECK to only measure keys added to .builtin_trusted_keys or .ima keyring:

measure func=KEY_CHECK keyrings=.builtin_trusted_keys|.ima

Example of the special SETXATTR_CHECK appraise rule, that restricts the hash algorithms allowed when writing to the security.ima xattr of a file:

appraise func=SETXATTR_CHECK appraise_algos=sha256,sha384,sha512

Example of a 'measure' rule requiring fs-verity's digests with indication of type of digest in the measurement list.

measure func=FILE_CHECK digest_type=verity
template=ima-ngv2

Example of 'measure' and 'appraise' rules requiring fs-verity signatures (format version 3) stored in security.ima xattr.

The 'measure' rule specifies the 'ima-sigv3' template option, which includes the indication of type of digest and the file signature in the measurement list.

measure func=BPRM_CHECK digest_type=verity
template=ima-sigv3

The 'appraise' rule specifies the type and signature format version (sigv3) required.

appraise func=BPRM_CHECK digest_type=verity
appraise_type=sigv3

All of these policy rules could, for example, be constrained either based on a filesystem's UUID (fsuuid) or based on LSM labels.

/sys/kernel/security/evm
/sys/kernel/security/*/evm

Defined on file *evm*

EVM protects a file's security extended attributes(xattrs) against integrity attacks. The initial method maintains an HMAC-sha1 value across the extended attributes, storing the value as the extended attribute 'security.evm'.

EVM supports two classes of security.evm. The first is an HMAC-sha1 generated locally with a trusted/encrypted key stored in the Kernel Key Retention System. The second is a digital signature generated either locally or remotely using an asymmetric key. These keys are loaded onto root's keyring using keyctl, and EVM is then enabled by echoing a value to <securityfs>/evm made up of the following bits:

Bit	Effect
0	Enable HMAC validation and creation
1	Enable digital signature validation
2	Permit modification of EVM-protected metadata at runtime. Not supported if HMAC validation and creation is enabled (deprecated).
31	Disable further runtime modification of EVM policy

For example:

```
echo 1 ><securityfs>/evm
```

will enable HMAC validation and creation

```
echo 0x80000003 ><securityfs>/evm
```

will enable HMAC and digital signature validation and HMAC creation and disable all further modification of policy.

```
echo 0x80000006 ><securityfs>/evm
```

will enable digital signature validation, permit modification of EVM-protected metadata and disable all further modification of policy. This option is now deprecated in favor of:

```
echo 0x80000002 ><securityfs>/evm
```

as the outstanding issues that prevent the usage of EVM portable signatures have been solved.

Echoing a value is additive, the new value is added to the existing initialization flags.

For example, after:

```
echo 2 ><securityfs>/evm
```

another echo can be performed:

```
echo 1 ><securityfs>/evm
```

and the resulting value will be 3.

Note that once an HMAC key has been loaded, it will no longer be possible to enable metadata modification. Signaling that an HMAC key has been loaded will clear the corresponding flag. For example, if the current value is 6 (2 and 4 set):

```
echo 1 ><securityfs>/evm
```

will set the new value to 3 (4 cleared).

Loading an HMAC key is the only way to disable metadata modification.

Until key loading has been signaled EVM can not create or validate the 'security.evm' xattr, but returns INTEGRITY_UNKNOWN. Loading keys and signaling EVM should be done as early as possible. Normally this is done in the initramfs, which has already been measured as part of the trusted boot. For more information on creating and loading existing trusted/encrypted keys, refer to: /security/keys/trusted-encrypted. Both dracut (via 97masterkey and 98integrity) and systemd (via core/ima-setup) have support for loading keys at boot time.

/sys/kernel/slab

Defined on file *sysfs-kernel-slab*

The */sys/kernel/slab* directory contains a snapshot of the internal state of the SLUB allocator for each cache. Certain files may be modified to change the behavior of the cache (and any cache it aliases, if any).

Users: kernel memory tuning tools

/sys/kernel/slab/<cache>/aliases

Defined on file *sysfs-kernel-slab*

The aliases file is read-only and specifies how many caches have merged into this cache.

/sys/kernel/slab/<cache>/align

Defined on file *sysfs-kernel-slab*

The align file is read-only and specifies the cache's object alignment in bytes.

/sys/kernel/slab/<cache>/alloc_calls

Defined on file *sysfs-kernel-slab*

The alloc_calls file is read-only and lists the kernel code locations from which allocations for this cache were performed. The alloc_calls file only contains information if debugging is enabled for that cache (see /mm/slub).

/sys/kernel/slab/<cache>/alloc_fastpath

Defined on file *sysfs-kernel-slab*

The alloc_fastpath file shows how many objects have been allocated using the fast path. It can be written to clear the current count. Available when CONFIG_SLUB_STATS is enabled.

/sys/kernel/slab/<cache>/alloc_from_partial

Defined on file *sysfs-kernel-slab*

The alloc_from_partial file shows how many times a cpu slab has been full and it has been refilled by using a slab from the list of partially used slabs. It can be written to clear the current count. Available when CONFIG_SLUB_STATS is enabled.

/sys/kernel/slab/<cache>/alloc_refill

Defined on file *sysfs-kernel-slab*

The alloc_refill file shows how many times the per-cpu freelist was empty but there were objects available as the result of remote cpu frees. It can be written to clear the current count. Available when CONFIG_SLUB_STATS is enabled.

/sys/kernel/slab/<cache>/alloc_slab

Defined on file *sysfs-kernel-slab*

The alloc_slab file is shows how many times a new slab had to be allocated from the page allocator. It can be written to clear the current count. Available when CONFIG_SLUB_STATS is enabled.

/sys/kernel/slab/<cache>/alloc_slowpath

Defined on file *sysfs-kernel-slab*

The alloc_slowpath file shows how many objects have been allocated using the slow path because of a refill or allocation from a partial or new slab. It can be written to clear the current count. Available when CONFIG_SLUB_STATS is enabled.

/sys/kernel/slab/<cache>/cache_dma

Defined on file *sysfs-kernel-slab*

The cache_dma file is read-only and specifies whether objects are from ZONE_DMA. Available when CONFIG_ZONE_DMA is enabled.

/sys/kernel/slab/<cache>/cpu_partial

Defined on file *sysfs-kernel-slab*

This read-only file shows the number of per cpu partial pages to keep around.

/sys/kernel/slab/<cache>/cpu_slabs

Defined on file *sysfs-kernel-slab*

The cpu_slabs file is read-only and displays how many cpu slabs are active and their NUMA locality.

/sys/kernel/slab/<cache>/cpuslab_flush

Defined on file *sysfs-kernel-slab*

The file cpuslab_flush shows how many times a cache's cpu slabs have been flushed as the result of destroying or shrinking a cache, a cpu going offline, or as the result of forcing an allocation from a certain node. It can be written to clear the current count. Available when CONFIG_SLUB_STATS is enabled.

/sys/kernel/slab/<cache>/ctor

Defined on file *sysfs-kernel-slab*

The ctor file is read-only and specifies the cache's object constructor function, which is invoked for each object when a new slab is allocated.

/sys/kernel/slab/<cache>/deactivate_empty

Defined on file *sysfs-kernel-slab*

The deactivate_empty file shows how many times an empty cpu slab was deactivated. It can be written to clear the current count. Available when CONFIG_SLUB_STATS is enabled.

/sys/kernel/slab/<cache>/deactivate_full

Defined on file *sysfs-kernel-slab*

The deactivate_full file shows how many times a full cpu slab was deactivated. It can be written to clear the current count. Available when CONFIG_SLUB_STATS is enabled.

/sys/kernel/slab/<cache>/deactivate_remote_frees

Defined on file *sysfs-kernel-slab*

The deactivate_remote_frees file shows how many times a cpu slab has been deactivated and contained free objects that were freed remotely. It can be written to clear the current count. Available when CONFIG_SLUB_STATS is enabled.

/sys/kernel/slab/<cache>/deactivate_to_head

Defined on file *sysfs-kernel-slab*

The deactivate_to_head file shows how many times a partial cpu slab was deactivated and added to the head of its node's partial list. It can be written to clear the current count. Available when CONFIG_SLUB_STATS is enabled.

/sys/kernel/slab/<cache>/deactivate_to_tail

Defined on file *sysfs-kernel-slab*

The deactivate_to_tail file shows how many times a partial cpu slab was deactivated and added to the tail of its node's partial list. It can be written to clear the current count. Available when CONFIG_SLUB_STATS is enabled.

/sys/kernel/slab/<cache>/destroy_by_rcu

Defined on file *sysfs-kernel-slab*

The destroy_by_rcu file is read-only and specifies whether slabs (not objects) are freed by rcu.

/sys/kernel/slab/<cache>/free_add_partial

Defined on file *sysfs-kernel-slab*

The free_add_partial file shows how many times an object has been freed in a full slab so that it had to added to its node's partial list. It can be written to clear the current count. Available when CONFIG_SLUB_STATS is enabled.

/sys/kernel/slab/<cache>/free_calls

Defined on file *sysfs-kernel-slab*

The free_calls file is read-only and lists the locations of object frees if slab debugging is enabled (see /mm/slub).

/sys/kernel/slab/<cache>/free_fastpath

Defined on file *sysfs-kernel-slab*

The free_fastpath file shows how many objects have been freed using the fast path because it was an object from the cpu slab. It can be written to clear the current count. Available when CONFIG_SLUB_STATS is enabled.

/sys/kernel/slab/<cache>/free_frozen

Defined on file *sysfs-kernel-slab*

The free_frozen file shows how many objects have been freed to a frozen slab (i.e. a remote cpu slab). It can be written to clear the current count. Available when CONFIG_SLUB_STATS is enabled.

/sys/kernel/slab/<cache>/free_remove_partial

Defined on file *sysfs-kernel-slab*

The free_remove_partial file shows how many times an object has been freed to a now-empty slab so that it had to be removed from its node's partial list. It can be written to clear the current count. Available when CONFIG_SLUB_STATS is enabled.

/sys/kernel/slab/<cache>/free_slab

Defined on file *sysfs-kernel-slab*

The free_slab file shows how many times an empty slab has been freed back to the page allocator. It can be written to clear the current count. Available when CONFIG_SLUB_STATS is enabled.

/sys/kernel/slab/<cache>/free_slowpath

Defined on file *sysfs-kernel-slab*

The free_slowpath file shows how many objects have been freed using the slow path (i.e. to a full or partial slab). It can be written to clear the current count. Available when CONFIG_SLUB_STATS is enabled.

/sys/kernel/slab/<cache>/hwcache_align

Defined on file *sysfs-kernel-slab*

The hwcache_align file is read-only and specifies whether objects are aligned on cachelines.

/sys/kernel/slab/<cache>/min_partial

Defined on file *sysfs-kernel-slab*

The min_partial file specifies how many empty slabs shall remain on a node's partial list to avoid the overhead of allocating new slabs. Such slabs may be reclaimed by utilizing the shrink file.

/sys/kernel/slab/<cache>/object_size

Defined on file *sysfs-kernel-slab*

The object_size file is read-only and specifies the cache's object size.

/sys/kernel/slab/<cache>/objects

Defined on file *sysfs-kernel-slab*

The objects file is read-only and displays how many objects are active and from which nodes they are from.

/sys/kernel/slab/<cache>/objects_partial

Defined on file *sysfs-kernel-slab*

The objects_partial file is read-only and displays how many objects are on partial slabs and from which nodes they are from.

/sys/kernel/slab/<cache>/objs_per_slab

Defined on file *sysfs-kernel-slab*

The file objs_per_slab is read-only and specifies how many objects may be allocated from a single slab of the order specified in */sys/kernel/slab/<cache>/order*.

/sys/kernel/slab/<cache>/order

Defined on file *sysfs-kernel-slab*

The order file specifies the page order at which new slabs are allocated. It is writable and can be changed to increase the number of objects per slab. If a slab cannot be allocated because of fragmentation, SLUB will retry with the minimum order possible depending on its characteristics.

When debug_guardpage_minorder=N (N > 0) parameter is specified (see *The kernel's command-line parameters*), the minimum possible order is used and this sysfs entry can not be used to change the order at run time.

/sys/kernel/slab/<cache>/order_fallback

Defined on file *sysfs-kernel-slab*

The order_fallback file shows how many times an allocation of a new slab has not been possible at the cache's order and instead fallen back to its minimum possible order. It can be written to clear the current count.

Available when CONFIG_SLUB_STATS is enabled.

/sys/kernel/slab/<cache>/partial

Defined on file *sysfs-kernel-slab*

The partial file is read-only and displays how long many partial slabs there are and how long each node's list is.

/sys/kernel/slab/<cache>/poison

Defined on file *sysfs-kernel-slab*

The poison file specifies whether objects should be poisoned when a new slab is allocated.

/sys/kernel/slab/<cache>/reclaim_account

Defined on file *sysfs-kernel-slab*

The reclaim_account file specifies whether the cache's objects are reclaimable (and grouped by their mobility).

/sys/kernel/slab/<cache>/red_zone

Defined on file *sysfs-kernel-slab*

The red_zone file specifies whether the cache's objects are red zoned.

/sys/kernel/slab/<cache>/remote_node_defrag_ratio

Defined on file *sysfs-kernel-slab*

The file remote_node_defrag_ratio specifies the percentage of times SLUB will attempt to refill the cpu slab with a partial slab from a remote node as opposed to allocating a new slab on the local node. This reduces the amount of wasted memory over the entire system but can be expensive.

Available when CONFIG_NUMA is enabled.

/sys/kernel/slab/<cache>/sanity_checks

Defined on file *sysfs-kernel-slab*

The sanity_checks file specifies whether expensive checks should be performed on free and, at minimum, enables double free checks. Caches that enable sanity_checks cannot be merged with caches that do not.

/sys/kernel/slab/<cache>/shrink

Defined on file *sysfs-kernel-slab*

The shrink file is used to reclaim unused slab cache memory from a cache. Empty per-cpu or partial slabs are freed and the partial list is sorted so the slabs with the fewest available objects are used first. It only accepts a value of "1" on write for shrinking the cache. Other input values are considered invalid. Shrinking slab caches might be expensive and can adversely impact other running applications. So it should be used with care.

/sys/kernel/slab/<cache>/slab_size

Defined on file *sysfs-kernel-slab*

The slab_size file is read-only and specifies the object size with metadata (debugging information and alignment) in bytes.

/sys/kernel/slab/<cache>/slabs

Defined on file *sysfs-kernel-slab*

The slabs file is read-only and displays how long many slabs there are (both cpu and partial) and from which nodes they are from.

/sys/kernel/slab/<cache>/slabs_cpu_partial

Defined on file *sysfs-kernel-slab*

This read-only file shows the number of partialli allocated frozen slabs.

/sys/kernel/slab/<cache>/store_user

Defined on file *sysfs-kernel-slab*

The store_user file specifies whether the location of allocation or free should be tracked for a cache.

/sys/kernel/slab/<cache>/total_objects

Defined on file *sysfs-kernel-slab*

The total_objects file is read-only and displays how many total objects a cache has and from which nodes they are from.

/sys/kernel/slab/<cache>/trace

Defined on file *sysfs-kernel-slab*

The trace file specifies whether object allocations and frees should be traced.

/sys/kernel/slab/<cache>/usersize

Defined on file *sysfs-kernel-slab*

The usersize file is read-only and contains the usercopy region size.

/sys/kernel/slab/<cache>/validate

Defined on file *sysfs-kernel-slab*

Writing to the validate file causes SLUB to traverse all of its cache's objects and check the validity of metadata.

/sys/kernel/vmcoreinfo

Defined on file *sysfs-kernel-vmcoreinfo*

Shows physical address and size of vmcoreinfo ELF note. First value contains physical address of note in hex and second value contains the size of note in hex. This ELF note info is parsed by second kernel and exported to user space as part of ELF note in /proc/vmcore file. This note contains various information like struct size, symbol values, page size etc.

/sys/kernel/warn_count

Defined on file *sysfs-kernel-warn_count*

Shows how many times the system has Warned since last boot.

5.2.39 Symbols under /sys/module

/sys/module/*/initstate

Defined on file *sysfs-module*

Show the initialization state(live, coming, going) of the module.

/sys/module/*/taint

Defined on file *sysfs-module*

Module taint flags:

P	proprietary module
O	out-of-tree module
F	force-loaded module
C	staging driver module
E	unsigned module

/sys/module/*/{coresize,initsize}

Defined on file *sysfs-module*

Module size in bytes.

/sys/module/ehci_hcd/drivers/.../uframe_periodic_max

Defined on file *sysfs-module*

Maximum time allowed for periodic transfers per microframe (µs)

Note:

USB 2.0 sets maximum allowed time for periodic transfers per microframe to be 80%, that is 100 microseconds out of 125 microseconds (full microframe).

However there are cases, when 80% max isochronous bandwidth is too limiting. For example two video streams could require 110 microseconds of isochronous bandwidth per microframe to work together.

Through this setting it is possible to raise the limit so that the host controller would allow allocating more than 100 microseconds of periodic bandwidth per microframe.

Beware, non-standard modes are usually not thoroughly tested by hardware designers, and the hardware can malfunction when this setting differ from default 100.

/sys/module/grant_table/parameters/free_per_iteration

Defined on file *sysfs-module*

Read and write number of grant entries to attempt to free per iteration.

Note: Future versions of Xen and Linux may provide a better interface for controlling the rate of deferred grant reclaim or may not need it at all.

Users: Qubes OS (https://www.qubes-os.org)

/sys/module/pch_phub/drivers/.../pch_firmware

Defined on file *sysfs-module*

Write/read Option ROM data.

/sys/module/pch_phub/drivers/.../pch_mac

Defined on file *sysfs-module*

Write/read GbE MAC address.

/sys/module/xen_blkback/parameters/buffer_squeeze_duration_ms

Defined on file *sysfs-driver-xen-blkback*

When memory pressure is reported to blkback this option controls the duration in milliseconds that blkback will not cache any page not backed by a grant mapping. The default is 10ms.

/sys/module/xen_blkback/parameters/feature_persistent

Defined on file *sysfs-driver-xen-blkback*

Whether to enable the persistent grants feature or not. Note that this option only takes effect on newly connected backends. The default is Y (enable).

/sys/module/xen_blkback/parameters/max_buffer_pages

Defined on file *sysfs-driver-xen-blkback*

Maximum number of free pages to keep in each block backend buffer.

/sys/module/xen_blkback/parameters/max_persistent_grants

Defined on file *sysfs-driver-xen-blkback*

Maximum number of grants to map persistently in blkback. If the frontend tries to use more than max_persistent_grants, the LRU kicks in and starts removing 5% of max_persistent_grants every 100ms.

/sys/module/xen_blkback/parameters/persistent_grant_unused_seconds

Defined on file *sysfs-driver-xen-blkback*

How long a persistent grant is allowed to remain allocated without being in use. The time is in seconds, 0 means indefinitely long. The default is 60 seconds.

/sys/module/xen_blkfront/parameters/feature_persistent

Defined on file *sysfs-driver-xen-blkfront*

Whether to enable the persistent grants feature or not. Note that this option only takes effect on newly connected frontends. The default is Y (enable).

/sys/module/xen_blkfront/parameters/max_indirect_segments

Defined on file *sysfs-driver-xen-blkfront*

Maximum number of segments that the frontend will negotiate with the backend for indirect descriptors. The default value is 32 - higher value means more potential throughput but more memory usage. The backend picks the minimum of the frontend and its default backend value.

5.2.40 Symbols under /sys/power

/sys/power/

Defined on file *sysfs-power*

The /sys/power directory will contain files that will provide a unified interface to the power management subsystem.

/sys/power/autosleep

Defined on file *sysfs-power*

The */sys/power/autosleep* file can be written one of the strings returned by reads from */sys/power/state*. If that happens, a work item attempting to trigger a transition of the system to the sleep state represented by that string is queued up. This attempt will only succeed if there are no active wakeup sources in the system at that time. After every execution, regardless of whether or not the attempt to put the system to sleep has succeeded, the work item requeues itself until user space writes "off" to */sys/power/autosleep*.

Reading from this file causes the last string successfully written to it to be returned.

/sys/power/disk

Defined on file *sysfs-power*

The */sys/power/disk* file controls the operating mode of the suspend-to-disk mechanism. Reading from this file returns the name of the method by which the system will be put to sleep on the next suspend. There are four methods supported:

'firmware' - means that the memory image will be saved to disk by some firmware, in which case we also assume that the firmware will handle the system suspend.

'platform' - the memory image will be saved by the kernel and the system will be put to sleep by the platform driver (e.g. ACPI or other PM registers).

'shutdown' - the memory image will be saved by the kernel and the system will be powered off.

'reboot' - the memory image will be saved by the kernel and the system will be rebooted.

Additionally, */sys/power/disk* can be used to turn on one of the two testing modes of the suspend-to-disk mechanism: 'testproc' or 'test'. If the suspend-to-disk mechanism is in the 'testproc' mode, writing 'disk' to */sys/power/state* will cause the kernel to disable nonboot CPUs and freeze tasks, wait for 5 seconds, unfreeze tasks and enable nonboot CPUs. If it is in the 'test' mode, writing 'disk' to */sys/power/state* will cause the kernel to disable nonboot CPUs and freeze tasks, shrink memory, suspend devices, wait for 5 seconds, resume devices, unfreeze tasks and enable nonboot CPUs. Then, we are able to look in the log messages and work out, for example, which code is being slow and which device drivers are misbehaving.

The suspend-to-disk method may be chosen by writing to this file one of the accepted strings:

- 'firmware'
- 'platform'
- 'shutdown'
- 'reboot'
- 'testproc'
- 'test'

It will only change to 'firmware' or 'platform' if the system supports that.

/sys/power/image_size

Defined on file *sysfs-power*

The */sys/power/image_size* file controls the size of the image created by the suspend-to-disk mechanism. It can be written a string representing a non-negative integer that will be used as an upper limit of the image size, in bytes. The kernel's suspend-to-disk code will do its best to ensure the image size will not exceed this number. However, if it turns out to be impossible, the kernel will try to suspend anyway using the smallest image possible. In particular, if ”0” is written to this file, the suspend image will be as small as possible.

Reading from this file will display the current image size limit, which is set to around 2/5 of available RAM by default.

/sys/power/mem_sleep

Defined on file *sysfs-power*

The */sys/power/mem_sleep* file controls the operating mode of system suspend. Reading from it returns the available modes as ”s2idle” (always present), ”shallow” and ”deep” (present if supported). The mode that will be used on subsequent attempts to suspend the system (by writing ”mem” to the */sys/power/state* file described above) is enclosed in square brackets.

Writing one of the above strings to this file causes the mode represented by it to be used on subsequent attempts to suspend the system.

See *System Sleep States* for more information.

/sys/power/pm_async

Defined on file *sysfs-power*

The */sys/power/pm_async* file controls the switch allowing the user space to enable or disable asynchronous suspend and resume of devices. If enabled, this feature will cause some device drivers' suspend and resume callbacks to be executed in parallel with each other and with the main suspend thread. It is enabled if this file contains "1", which is the default. It may be disabled by writing "0" to this file, in which case all devices will be suspended and resumed synchronously.

/sys/power/pm_debug_messages

Defined on file *sysfs-power*

The */sys/power/pm_debug_messages* file controls the printing of debug messages from the system suspend/hiberbation infrastructure to the kernel log.

Writing a "1" to this file enables the debug messages and writing a "0" (default) to it disables them. Reads from this file return the current value.

/sys/power/pm_print_times

Defined on file *sysfs-power*

The */sys/power/pm_print_times* file allows user space to control whether the time taken by devices to suspend and resume is printed. These prints are useful for hunting down devices that take too long to suspend or resume.

Writing a "1" enables this printing while writing a "0" disables it. The default value is "0". Reading from this file will display the current value.

/sys/power/pm_trace

Defined on file *sysfs-power*

The */sys/power/pm_trace* file controls the code which saves the last PM event point in the RTC across reboots, so that you can debug a machine that just hangs during suspend (or more commonly, during resume). Namely, the RTC is only used to save the last PM event point if this file contains '1'. Initially it contains '0' which may be changed to '1' by writing a string representing a nonzero integer into it.

To use this debugging feature you should attempt to suspend the machine, then reboot it and run:

```
dmesg -s 1000000 | grep 'hash matches'
```

If you do not get any matches (or they appear to be false positives), it is possible that the last PM event point referred to a device created by a loadable kernel module. In this case cat /sys/power/pm_trace_dev_match (see below) after your system is started up and the kernel modules are loaded.

CAUTION: Using it will cause your machine's real-time (CMOS) clock to be set to a random invalid time after a resume.

What; /sys/power/pm_trace_dev_match The /sys/power/pm_trace_dev_match file contains the name of the device associated with the last PM event point saved in the RTC across reboots when pm_trace has been used. More precisely it contains the list of current devices (including those registered by loadable kernel modules since boot) which match the device hash in the RTC at boot, with a newline after each one.

The advantage of this file over the hash matches printed to the kernel log (see */sys/power/pm_trace*), is that it includes devices created after boot by loadable kernel modules.

Due to the small hash size necessary to fit in the RTC, it is possible that more than one device matches the hash, in which case further investigation is required to determine which device is causing the problem. Note that genuine RTC clock values (such as when pm_trace has not been used), can still match a device and output its name here.

/sys/power/pm_wakeup_irq

Defined on file *sysfs-power*

The */sys/power/pm_wakeup_irq* file reports to user space the IRQ number of the first wakeup interrupt (that is, the first interrupt from an IRQ line armed for system wakeup) seen by the kernel during the most recent system suspend/resume cycle.

This output is useful for system wakeup diagnostics of spurious wakeup interrupts.

/sys/power/reserved_size

Defined on file *sysfs-power*

The */sys/power/reserved_size* file allows user space to control the amount of memory reserved for allocations made by device drivers during the ”device freeze” stage of hibernation. It can be written a string representing a non-negative integer that will be used as the amount of memory to reserve for allocations made by device drivers' ”freeze” callbacks, in bytes.

Reading from this file will display the current value, which is set to 1 MB by default.

/sys/power/resume_offset

Defined on file *sysfs-power*

This file is used for telling the kernel an offset into a disk to use when hibernating the system such as with a swap file.

Reads from this file will display the current offset the kernel will be using on the next hibernation attempt.

Using this sysfs file will override any values that were set using the kernel command line for disk offset.

/sys/power/state

Defined on file *sysfs-power*

The */sys/power/state* file controls system sleep states. Reading from this file returns the available sleep state labels, which may be "mem" (suspend), "standby" (power-on suspend), "freeze" (suspend-to-idle) and "disk" (hibernation).

Writing one of the above strings to this file causes the system to transition into the corresponding state, if available.

See *System Sleep States* for more information.

/sys/power/suspend_stats

Defined on file *sysfs-power*

The */sys/power/suspend_stats* directory contains suspend related statistics.

/sys/power/suspend_stats/fail

Defined on file *sysfs-power*

The */sys/power/suspend_stats/fail* file contains the number of times entering system sleep state failed.

/sys/power/suspend_stats/failed_freeze

Defined on file *sysfs-power*

The */sys/power/suspend_stats/failed_freeze* file contains the number of times freezing processes failed.

/sys/power/suspend_stats/failed_prepare

Defined on file *sysfs-power*

The */sys/power/suspend_stats/failed_prepare* file contains the number of times preparing all non-sysdev devices for a system PM transition failed.

/sys/power/suspend_stats/failed_resume

Defined on file *sysfs-power*

The */sys/power/suspend_stats/failed_resume* file contains the number of times executing "resume" callbacks of non-sysdev devices failed.

/sys/power/suspend_stats/failed_resume_early

Defined on file *sysfs-power*

The */sys/power/suspend_stats/failed_resume_early* file contains the number of times executing "early resume" callbacks of devices failed.

/sys/power/suspend_stats/failed_resume_noirq

Defined on file *sysfs-power*

The */sys/power/suspend_stats/failed_resume_noirq* file contains the number of times executing "noirq resume" callbacks of devices failed.

/sys/power/suspend_stats/failed_suspend

Defined on file *sysfs-power*

The */sys/power/suspend_stats/failed_suspend* file contains the number of times executing "suspend" callbacks of all non-sysdev devices failed.

/sys/power/suspend_stats/failed_suspend_late

Defined on file *sysfs-power*

The */sys/power/suspend_stats/failed_suspend_late* file contains the number of times executing "late suspend" callbacks of all devices failed.

/sys/power/suspend_stats/failed_suspend_noirq

Defined on file *sysfs-power*

The */sys/power/suspend_stats/failed_suspend_noirq* file contains the number of times executing "noirq suspend" callbacks of all devices failed.

/sys/power/suspend_stats/last_failed_dev

Defined on file *sysfs-power*

The */sys/power/suspend_stats/last_failed_dev* file contains the last device for which a suspend/resume callback failed.

/sys/power/suspend_stats/last_failed_errno

Defined on file *sysfs-power*

The */sys/power/suspend_stats/last_failed_errno* file contains the errno of the last failed attempt at entering system sleep state.

/sys/power/suspend_stats/last_failed_step

Defined on file *sysfs-power*

The */sys/power/suspend_stats/last_failed_step* file contains the last failed step in the suspend/resume path.

/sys/power/suspend_stats/last_hw_sleep

Defined on file *sysfs-power*

The */sys/power/suspend_stats/last_hw_sleep* file contains the duration of time spent in a hardware sleep state in the most recent system suspend-resume cycle. This number is measured in microseconds.

/sys/power/suspend_stats/max_hw_sleep

Defined on file *sysfs-power*

The */sys/power/suspend_stats/max_hw_sleep* file contains the maximum amount of time that the hardware can report for time spent in a hardware sleep state. When sleep cycles are longer than this time, the values for 'total_hw_sleep' and 'last_hw_sleep' may not be accurate. This number is measured in microseconds.

/sys/power/suspend_stats/success

Defined on file *sysfs-power*

The */sys/power/suspend_stats/success* file contains the number of times entering system sleep state succeeded.

/sys/power/suspend_stats/total_hw_sleep

Defined on file *sysfs-power*

The */sys/power/suspend_stats/total_hw_sleep* file contains the aggregate of time spent in a hardware sleep state since the kernel was booted. This number is measured in microseconds.

/sys/power/sync_on_suspend

Defined on file *sysfs-power*

This file controls whether or not the kernel will sync() filesystems during system suspend (after freezing user space and before suspending devices).

Writing a ”1” to this file enables the sync() and writing a ”0” disables it. Reads from the file return the current value. The default is ”1” if the build-time ”SUSPEND_SKIP_SYNC” config flag is unset, or ”0” otherwise.

/sys/power/wake_lock

Defined on file *sysfs-power*

The */sys/power/wake_lock* file allows user space to create wakeup source objects and activate them on demand (if one of those wakeup sources is active, reads from the */sys/power/wakeup_count* file block or return false). When a string without white space is written to */sys/power/wake_lock*, it will be assumed to represent a wakeup source name. If there is a wakeup source object with that name, it will be activated (unless active already). Otherwise, a new wakeup source object will be registered, assigned the given name and activated. If a string written to */sys/power/wake_lock* contains white space, the part of the string preceding the white space will be regarded as a wakeup source name and handled as descrived above. The other part of the string will be regarded as a timeout (in nanoseconds) such that the wakeup

source will be automatically deactivated after it has expired. The timeout, if present, is set regardless of the current state of the wakeup source object in question.

Reads from this file return a string consisting of the names of wakeup sources created with the help of it that are active at the moment, separated with spaces.

/sys/power/wake_unlock

Defined on file *sysfs-power*

The */sys/power/wake_unlock* file allows user space to deactivate wakeup sources created with the help of */sys/power/wake_lock*. When a string is written to */sys/power/wake_unlock*, it will be assumed to represent the name of a wakeup source to deactivate.

If a wakeup source object of that name exists and is active at the moment, it will be deactivated.

Reads from this file return a string consisting of the names of wakeup sources created with the help of */sys/power/wake_lock* that are inactive at the moment, separated with spaces.

/sys/power/wakeup_count

Defined on file *sysfs-power*

The */sys/power/wakeup_count* file allows user space to put the system into a sleep state while taking into account the concurrent arrival of wakeup events. Reading from it returns the current number of registered wakeup events and it blocks if some wakeup events are being processed at the time the file is read from. Writing to it will only succeed if the current number of wakeup events is equal to the written value and, if successful, will make the kernel abort a subsequent transition to a sleep state if any wakeup events are reported after the write has returned.

5.2.41 Raise a uevent when a USB Host Controller has died

Raise a uevent when a USB Host Controller has died

Defined on file *usb-uevent*

When the USB Host Controller has entered a state where it is no longer functional a uevent will be raised. The uevent will contain ACTION=offline and ERROR=DEAD.

Here is an example taken using udevadm monitor -p:

```
KERNEL[130.428945] offline  /devices/pci0000:00/0000:00:10.0/usb2 (usb)
ACTION=offline
BUSNUM=002
DEVNAME=/dev/bus/usb/002/001
DEVNUM=001
DEVPATH=/devices/pci0000:00/0000:00:10.0/usb2
DEVTYPE=usb_device
DRIVER=usb
ERROR=DEAD
MAJOR=189
MINOR=128
```

```
PRODUCT=1d6b/2/414
SEQNUM=2168
SUBSYSTEM=usb
TYPE=9/0/1
```

Users: chromium-os-dev@chromium.org

5.2.42 Raise a uevent when a USB charger is inserted or removed

Raise a uevent when a USB charger is inserted or removed

Defined on file *usb-charger-uevent*

There are two USB charger states:

- USB_CHARGER_ABSENT
- USB_CHARGER_PRESENT

There are five USB charger types:

USB_CHARGER_UNKNOWN_TYPE	Charger type is unknown
USB_CHARGER_SDP_TYPE	Standard Downstream Port
USB_CHARGER_CDP_TYPE	Charging Downstream Port
USB_CHARGER_DCP_TYPE	Dedicated Charging Port
USB_CHARGER_ACA_TYPE	Accessory Charging Adapter

https://www.usb.org/document-library/battery-charging-v12-spec-and-adopters-agreement

Here are two examples taken using `udevadm monitor -p` when USB charger is online:

```
UDEV  change   /devices/soc0/usbphynop1 (platform)
ACTION=change
DEVPATH=/devices/soc0/usbphynop1
DRIVER=usb_phy_generic
MODALIAS=of:Nusbphynop1T(null)Cusb-nop-xceiv
OF_COMPATIBLE_0=usb-nop-xceiv
OF_COMPATIBLE_N=1
OF_FULLNAME=/usbphynop1
OF_NAME=usbphynop1
SEQNUM=2493
SUBSYSTEM=platform
USB_CHARGER_STATE=USB_CHARGER_PRESENT
USB_CHARGER_TYPE=USB_CHARGER_SDP_TYPE
USEC_INITIALIZED=227422826
```

USB charger is offline:

```
KERNEL change   /devices/soc0/usbphynop1 (platform)
ACTION=change
DEVPATH=/devices/soc0/usbphynop1
```

```
DRIVER=usb_phy_generic
MODALIAS=of:Nusbphynop1T(null)Cusb-nop-xceiv
OF_COMPATIBLE_0=usb-nop-xceiv
OF_COMPATIBLE_N=1
OF_FULLNAME=/usbphynop1
OF_NAME=usbphynop1
SEQNUM=2494
SUBSYSTEM=platform
USB_CHARGER_STATE=USB_CHARGER_ABSENT
USB_CHARGER_TYPE=USB_CHARGER_UNKNOWN_TYPE
```

5.2.43 nvdimm

nvdimm

Defined on file *sysfs-bus-nvdimm*

The libnvdimm sub-system implements a common sysfs interface for platform nvdimm resources. See Documentation/driver-api/nvdimm/.

5.2.44 Symbols under security/secrets

security/secrets/coco

Defined on file *securityfs-secrets-coco*

Exposes confidential computing (coco) EFI secrets to userspace via securityfs.

EFI can declare memory area used by confidential computing platforms (such as AMD SEV and SEV-ES) for secret injection by the Guest Owner during VM's launch. The secrets are encrypted by the Guest Owner and decrypted inside the trusted enclave, and therefore are not readable by the untrusted host.

The efi_secret module exposes the secrets to userspace. Each secret appears as a file under <securityfs>/secrets/coco, where the filename is the GUID of the entry in the secrets table. This module is loaded automatically by the EFI driver if the EFI secret area is populated.

Two operations are supported for the files: read and unlink. Reading the file returns the content of secret entry. Unlinking the file overwrites the secret data with zeroes and removes the entry from the filesystem. A secret cannot be read after it has been unlinked.

For example, listing the available secrets:

```
# modprobe efi_secret
# ls -l /sys/kernel/security/secrets/coco
-r--r----- 1 root root 0 Jun 28 11:54 736870e5-84f0-4973-92ec-06879ce3da0b
-r--r----- 1 root root 0 Jun 28 11:54 83c83f7f-1356-4975-8b7e-d3a0b54312c6
-r--r----- 1 root root 0 Jun 28 11:54 9553f55d-3da2-43ee-ab5d-ff17f78864d2
-r--r----- 1 root root 0 Jun 28 11:54 e6f5a162-d67f-4750-a67c-5d065f2a9910
```

Reading the secret data by reading a file:

```
# cat /sys/kernel/security/secrets/coco/e6f5a162-d67f-4750-a67c-5d065f2a9910
the-content-of-the-secret-data
```

Wiping a secret by unlinking a file:

```
# rm /sys/kernel/security/secrets/coco/e6f5a162-d67f-4750-a67c-5d065f2a9910
# ls -l /sys/kernel/security/secrets/coco
-r--r----- 1 root root 0 Jun 28 11:54 736870e5-84f0-4973-92ec-06879ce3da0b
-r--r----- 1 root root 0 Jun 28 11:54 83c83f7f-1356-4975-8b7e-d3a0b54312c6
-r--r----- 1 root root 0 Jun 28 11:54 9553f55d-3da2-43ee-ab5d-ff17f78864d2
```

Note: The binary format of the secrets table injected by the Guest Owner is described in drivers/virt/coco/efi_secret/efi_secret.c under "Structure of the EFI secret area".

5.2.45 File testing/configfs-acpi

Has the following ABI:

- */config/acpi*
- */config/acpi/table*

5.2.46 File testing/configfs-iio

Has the following ABI:

- */config/iio*
- */config/iio/triggers*
- */config/iio/triggers/hrtimers*
- */config/iio/devices*
- */config/iio/devices/dummy*

5.2.47 File testing/configfs-most

Has the following ABI:

- */sys/kernel/config/most_<component>*
- */sys/kernel/config/most_cdev/<link>*
- */sys/kernel/config/most_video/<link>*
- */sys/kernel/config/most_net/<link>*
- */sys/kernel/config/most_sound/<card>*
- */sys/kernel/config/most_sound/<card>/<link>*

5.2.48 File testing/configfs-rdma_cm

Has the following ABI:

- */config/rdma_cm*
- */config/rdma_cm/<hca>/ports/<port-num>/default_roce_mode*
- */config/rdma_cm/<hca>/ports/<port-num>/default_roce_tos*

5.2.49 File testing/configfs-spear-pcie-gadget

Has the following ABI:

- */config/pcie-gadget*

5.2.50 File testing/configfs-stp-policy

Has the following ABI:

- */config/stp-policy*
- */config/stp-policy/<device>.<policy>*
- */config/stp-policy/<device>.<policy>/device*
- */config/stp-policy/<device>.<policy>/<node>*
- */config/stp-policy/<device>.<policy>/<node>/masters*
- */config/stp-policy/<device>.<policy>/<node>/channels*

5.2.51 File testing/configfs-stp-policy-p_sys-t

Has the following ABI:

- */config/stp-policy/<device>:p_sys-t.<policy>/<node>/uuid*
- */config/stp-policy/<device>:p_sys-t.<policy>/<node>/do_len*
- */config/stp-policy/<device>:p_sys-t.<policy>/<node>/ts_interval*
- */config/stp-policy/<device>:p_sys-t.<policy>/<node>/clocksync_interval*

5.2.52 File testing/configfs-tsm

Has the following ABI:

- */sys/kernel/config/tsm/report/$name/inblob*
- */sys/kernel/config/tsm/report/$name/outblob*
- */sys/kernel/config/tsm/report/$name/auxblob*
- */sys/kernel/config/tsm/report/$name/provider*
- */sys/kernel/config/tsm/report/$name/generation*
- */sys/kernel/config/tsm/report/$name/privlevel*

- */sys/kernel/config/tsm/report/$name/privlevel_floor*

5.2.53 File testing/configfs-usb-gadget

Has the following ABI:

- */config/usb-gadget*
- */config/usb-gadget/gadget*
- */config/usb-gadget/gadget/configs*
- */config/usb-gadget/gadget/configs/config*
- */config/usb-gadget/gadget/configs/config/strings*
- */config/usb-gadget/gadget/configs/config/strings/language*
- */config/usb-gadget/gadget/functions*
- */config/usb-gadget/gadget/functions/<func>.<inst>/interface.<n>*
- */config/usb-gadget/gadget/functions/<func>.<inst>/interface.<n>/<property>*
- */config/usb-gadget/gadget/strings*
- */config/usb-gadget/gadget/strings/language*
- */config/usb-gadget/gadget/os_desc*
- */config/usb-gadget/gadget/webusb*

5.2.54 File testing/configfs-usb-gadget-acm

Has the following ABI:

- */config/usb-gadget/gadget/functions/acm.name*

5.2.55 File testing/configfs-usb-gadget-ecm

Has the following ABI:

- */config/usb-gadget/gadget/functions/ecm.name*

5.2.56 File testing/configfs-usb-gadget-eem

Has the following ABI:

- */config/usb-gadget/gadget/functions/eem.name*

5.2.57 File testing/configfs-usb-gadget-ffs

Has the following ABI:

- */config/usb-gadget/gadget/functions/ffs.name*

5.2.58 File testing/configfs-usb-gadget-hid

Has the following ABI:

- */config/usb-gadget/gadget/functions/hid.name*

5.2.59 File testing/configfs-usb-gadget-loopback

Has the following ABI:

- */config/usb-gadget/gadget/functions/Loopback.name*

5.2.60 File testing/configfs-usb-gadget-mass-storage

Has the following ABI:

- */config/usb-gadget/gadget/functions/mass_storage.name*
- */config/usb-gadget/gadget/functions/mass_storage.name/lun.name*

5.2.61 File testing/configfs-usb-gadget-midi

Has the following ABI:

- */config/usb-gadget/gadget/functions/midi.name*

5.2.62 File testing/configfs-usb-gadget-midi2

Has the following ABI:

- */config/usb-gadget/gadget/functions/midi2.name*
- */config/usb-gadget/gadget/functions/midi2.name/ep.number*
- */config/usb-gadget/gadget/functions/midi2.name/ep.number/block.number*

5.2.63 File testing/configfs-usb-gadget-ncm

Has the following ABI:

- */config/usb-gadget/gadget/functions/ncm.name*

5.2.64 File testing/configfs-usb-gadget-obex

Has the following ABI:

- */config/usb-gadget/gadget/functions/obex.name*

5.2.65 File testing/configfs-usb-gadget-phonet

Has the following ABI:

- */config/usb-gadget/gadget/functions/phonet.name*

5.2.66 File testing/configfs-usb-gadget-printer

Has the following ABI:

- */config/usb-gadget/gadget/functions/printer.name*

5.2.67 File testing/configfs-usb-gadget-rndis

Has the following ABI:

- */config/usb-gadget/gadget/functions/rndis.name*

5.2.68 File testing/configfs-usb-gadget-serial

Has the following ABI:

- */config/usb-gadget/gadget/functions/gser.name*

5.2.69 File testing/configfs-usb-gadget-sourcesink

Has the following ABI:

- */config/usb-gadget/gadget/functions/SourceSink.name*

5.2.70 File testing/configfs-usb-gadget-subset

Has the following ABI:

- */config/usb-gadget/gadget/functions/geth.name*

5.2.71 File testing/configfs-usb-gadget-tcm

Has the following ABI:

- */config/usb-gadget/gadget/functions/tcm.name*

5.2.72 File testing/configfs-usb-gadget-uac1

Has the following ABI:

- */config/usb-gadget/gadget/functions/uac1.name*

5.2.73 File testing/configfs-usb-gadget-uac1_legacy

Has the following ABI:

- */config/usb-gadget/gadget/functions/uac1_legacy.name*

5.2.74 File testing/configfs-usb-gadget-uac2

Has the following ABI:

- */config/usb-gadget/gadget/functions/uac2.name*

5.2.75 File testing/configfs-usb-gadget-uvc

Has the following ABI:

- */config/usb-gadget/gadget/functions/uvc.name*
- */config/usb-gadget/gadget/functions/uvc.name/control*
- */config/usb-gadget/gadget/functions/uvc.name/control/class*
- */config/usb-gadget/gadget/functions/uvc.name/control/class/ss*
- */config/usb-gadget/gadget/functions/uvc.name/control/class/fs*
- */config/usb-gadget/gadget/functions/uvc.name/control/terminal*
- */config/usb-gadget/gadget/functions/uvc.name/control/terminal/output*
- */config/usb-gadget/gadget/functions/uvc.name/control/terminal/output/default*
- */config/usb-gadget/gadget/functions/uvc.name/control/terminal/camera*
- */config/usb-gadget/gadget/functions/uvc.name/control/terminal/camera/default*
- */config/usb-gadget/gadget/functions/uvc.name/control/processing*
- */config/usb-gadget/gadget/functions/uvc.name/control/processing/default*
- */config/usb-gadget/gadget/functions/uvc.name/control/extensions*
- */config/usb-gadget/gadget/functions/uvc.name/control/extensions/name*
- */config/usb-gadget/gadget/functions/uvc.name/control/header*
- */config/usb-gadget/gadget/functions/uvc.name/control/header/name*

- */config/usb-gadget/gadget/functions/uvc.name/streaming*
- */config/usb-gadget/gadget/functions/uvc.name/streaming/class*
- */config/usb-gadget/gadget/functions/uvc.name/streaming/class/ss*
- */config/usb-gadget/gadget/functions/uvc.name/streaming/class/hs*
- */config/usb-gadget/gadget/functions/uvc.name/streaming/class/fs*
- */config/usb-gadget/gadget/functions/uvc.name/streaming/color_matching*
- */config/usb-gadget/gadget/functions/uvc.name/streaming/color_matching/default*
- */config/usb-gadget/gadget/functions/uvc.name/streaming/color_matching/name*
- */config/usb-gadget/gadget/functions/uvc.name/streaming/mjpeg*
- */config/usb-gadget/gadget/functions/uvc.name/streaming/mjpeg/name*
- */config/usb-gadget/gadget/functions/uvc.name/streaming/mjpeg/name/name*
- */config/usb-gadget/gadget/functions/uvc.name/streaming/uncompressed*
- */config/usb-gadget/gadget/functions/uvc.name/streaming/uncompressed/name*
- */config/usb-gadget/gadget/functions/uvc.name/streaming/uncompressed/name/name*
- */config/usb-gadget/gadget/functions/uvc.name/streaming/header*
- */config/usb-gadget/gadget/functions/uvc.name/streaming/header/name*
- */sys/class/udc/udc.name/device/gadget/video4linux/video.name/function_name*

5.2.76 File testing/debugfs-cec-error-inj

Has the following ABI:

- */sys/kernel/debug/cec/*/error-inj*

5.2.77 File testing/debugfs-cros-ec

Has the following ABI:

- */sys/kernel/debug/<cros-ec-device>/console_log*
- */sys/kernel/debug/<cros-ec-device>/panicinfo*
- */sys/kernel/debug/<cros-ec-device>/pdinfo*
- */sys/kernel/debug/<cros-ec-device>/uptime*
- */sys/kernel/debug/<cros-ec-device>/last_resume_result*
- */sys/kernel/debug/<cros-ec-device>/suspend_timeout_ms*

5.2.78 File testing/debugfs-cxl

Has the following ABI:

- */sys/kernel/debug/cxl/memX/inject_poison*
- */sys/kernel/debug/memX/clear_poison*

5.2.79 File testing/debugfs-dell-wmi-ddv

Has the following ABI:

- */sys/kernel/debug/dell-wmi-ddv-<wmi_device_name>/fan_sensor_information*
- */sys/kernel/debug/dell-wmi-ddv-<wmi_device_name>/thermal_sensor_information*

5.2.80 File testing/debugfs-driver-dcc

Has the following ABI:

- */sys/kernel/debug/dcc/.../ready*
- */sys/kernel/debug/dcc/.../trigger*
- */sys/kernel/debug/dcc/.../config_reset*
- */sys/kernel/debug/dcc/.../[list-number]/config*
- */sys/kernel/debug/dcc/.../[list-number]/enable*

5.2.81 File testing/debugfs-driver-genwqe

Has the following ABI:

- */sys/kernel/debug/genwqe/genwqe<n>_card/ddcb_info*
- */sys/kernel/debug/genwqe/genwqe<n>_card/curr_regs*
- */sys/kernel/debug/genwqe/genwqe<n>_card/curr_dbg_uid0*
- */sys/kernel/debug/genwqe/genwqe<n>_card/curr_dbg_uid1*
- */sys/kernel/debug/genwqe/genwqe<n>_card/curr_dbg_uid2*
- */sys/kernel/debug/genwqe/genwqe<n>_card/prev_regs*
- */sys/kernel/debug/genwqe/genwqe<n>_card/prev_dbg_uid0*
- */sys/kernel/debug/genwqe/genwqe<n>_card/prev_dbg_uid1*
- */sys/kernel/debug/genwqe/genwqe<n>_card/prev_dbg_uid2*
- */sys/kernel/debug/genwqe/genwqe<n>_card/info*
- */sys/kernel/debug/genwqe/genwqe<n>_card/err_inject*
- */sys/kernel/debug/genwqe/genwqe<n>_card/vf<0..14>_jobtimeout_msec*
- */sys/kernel/debug/genwqe/genwqe<n>_card/jobtimer*
- */sys/kernel/debug/genwqe/genwqe<n>_card/queue_working_time*

5.2.82 File testing/debugfs-driver-habanalabs

Has the following ABI:

- */sys/kernel/debug/accel/<parent_device>/addr*
- */sys/kernel/debug/accel/<parent_device>/clk_gate*
- */sys/kernel/debug/accel/<parent_device>/command_buffers*
- */sys/kernel/debug/accel/<parent_device>/command_submission*
- */sys/kernel/debug/accel/<parent_device>/command_submission_jobs*
- */sys/kernel/debug/accel/<parent_device>/data32*
- */sys/kernel/debug/accel/<parent_device>/data64*
- */sys/kernel/debug/accel/<parent_device>/data_dma*
- */sys/kernel/debug/accel/<parent_device>/device*
- */sys/kernel/debug/accel/<parent_device>/device_release_watchdog_timeout*
- */sys/kernel/debug/accel/<parent_device>/dma_size*
- */sys/kernel/debug/accel/<parent_device>/dump_razwi_events*
- */sys/kernel/debug/accel/<parent_device>/dump_security_violations*
- */sys/kernel/debug/accel/<parent_device>/engines*
- */sys/kernel/debug/accel/<parent_device>/i2c_addr*
- */sys/kernel/debug/accel/<parent_device>/i2c_bus*
- */sys/kernel/debug/accel/<parent_device>/i2c_data*
- */sys/kernel/debug/accel/<parent_device>/i2c_len*
- */sys/kernel/debug/accel/<parent_device>/i2c_reg*
- */sys/kernel/debug/accel/<parent_device>/led0*
- */sys/kernel/debug/accel/<parent_device>/led1*
- */sys/kernel/debug/accel/<parent_device>/led2*
- */sys/kernel/debug/accel/<parent_device>/memory_scrub*
- */sys/kernel/debug/accel/<parent_device>/memory_scrub_val*
- */sys/kernel/debug/accel/<parent_device>/mmu*
- */sys/kernel/debug/accel/<parent_device>/mmu_error*
- */sys/kernel/debug/accel/<parent_device>/monitor_dump*
- */sys/kernel/debug/accel/<parent_device>/monitor_dump_trig*
- */sys/kernel/debug/accel/<parent_device>/set_power_state*
- */sys/kernel/debug/accel/<parent_device>/skip_reset_on_timeout*
- */sys/kernel/debug/accel/<parent_device>/state_dump*
- */sys/kernel/debug/accel/<parent_device>/stop_on_err*

- */sys/kernel/debug/accel/<parent_device>/timeout_locked*
- */sys/kernel/debug/accel/<parent_device>/userptr*
- */sys/kernel/debug/accel/<parent_device>/userptr_lookup*
- */sys/kernel/debug/accel/<parent_device>/vm*

5.2.83 File testing/debugfs-driver-qat

Has the following ABI:

- */sys/kernel/debug/qat_<device>_<BDF>/fw_counters*
- */sys/kernel/debug/qat_<device>_<BDF>/heartbeat/config*
- */sys/kernel/debug/qat_<device>_<BDF>/heartbeat/queries_failed*
- */sys/kernel/debug/qat_<device>_<BDF>/heartbeat/queries_sent*
- */sys/kernel/debug/qat_<device>_<BDF>/heartbeat/status*
- */sys/kernel/debug/qat_<device>_<BDF>/pm_status*
- */sys/kernel/debug/qat_<device>_<BDF>/cnv_errors*

5.2.84 File testing/debugfs-driver-qat_telemetry

Has the following ABI:

- */sys/kernel/debug/qat_<device>_<BDF>/telemetry/control*
- */sys/kernel/debug/qat_<device>_<BDF>/telemetry/device_data*
- */sys/kernel/debug/qat_<device>_<BDF>/telemetry/rp_<A/B/C/D>_data*

5.2.85 File testing/debugfs-ec

Has the following ABI:

- */sys/kernel/debug/ec/*/{gpe,use_global_lock,io}*

5.2.86 File testing/debugfs-hisi-hpre

Has the following ABI:

- */sys/kernel/debug/hisi_hpre/<bdf>/cluster[0-3]/regs*
- */sys/kernel/debug/hisi_hpre/<bdf>/cluster[0-3]/cluster_ctrl*
- */sys/kernel/debug/hisi_hpre/<bdf>/rdclr_en*
- */sys/kernel/debug/hisi_hpre/<bdf>/current_qm*
- */sys/kernel/debug/hisi_hpre/<bdf>/alg_qos*
- */sys/kernel/debug/hisi_hpre/<bdf>/regs*
- */sys/kernel/debug/hisi_hpre/<bdf>/qm/regs*

- */sys/kernel/debug/hisi_hpre/<bdf>/qm/current_q*
- */sys/kernel/debug/hisi_hpre/<bdf>/qm/clear_enable*
- */sys/kernel/debug/hisi_hpre/<bdf>/qm/err_irq*
- */sys/kernel/debug/hisi_hpre/<bdf>/qm/aeq_irq*
- */sys/kernel/debug/hisi_hpre/<bdf>/qm/abnormal_irq*
- */sys/kernel/debug/hisi_hpre/<bdf>/qm/create_qp_err*
- */sys/kernel/debug/hisi_hpre/<bdf>/qm/mb_err*
- */sys/kernel/debug/hisi_hpre/<bdf>/qm/status*
- */sys/kernel/debug/hisi_hpre/<bdf>/qm/diff_regs*
- */sys/kernel/debug/hisi_hpre/<bdf>/hpre_dfx/diff_regs*
- */sys/kernel/debug/hisi_hpre/<bdf>/hpre_dfx/send_cnt*
- */sys/kernel/debug/hisi_hpre/<bdf>/hpre_dfx/recv_cnt*
- */sys/kernel/debug/hisi_hpre/<bdf>/hpre_dfx/send_busy_cnt*
- */sys/kernel/debug/hisi_hpre/<bdf>/hpre_dfx/send_fail_cnt*
- */sys/kernel/debug/hisi_hpre/<bdf>/hpre_dfx/invalid_req_cnt*
- */sys/kernel/debug/hisi_hpre/<bdf>/hpre_dfx/overtime_thrhld*
- */sys/kernel/debug/hisi_hpre/<bdf>/hpre_dfx/over_thrhld_cnt*

5.2.87 File testing/debugfs-hisi-sec

Has the following ABI:

- */sys/kernel/debug/hisi_sec2/<bdf>/clear_enable*
- */sys/kernel/debug/hisi_sec2/<bdf>/current_qm*
- */sys/kernel/debug/hisi_sec2/<bdf>/alg_qos*
- */sys/kernel/debug/hisi_sec2/<bdf>/qm/qm_regs*
- */sys/kernel/debug/hisi_sec2/<bdf>/qm/current_q*
- */sys/kernel/debug/hisi_sec2/<bdf>/qm/clear_enable*
- */sys/kernel/debug/hisi_sec2/<bdf>/qm/err_irq*
- */sys/kernel/debug/hisi_sec2/<bdf>/qm/aeq_irq*
- */sys/kernel/debug/hisi_sec2/<bdf>/qm/abnormal_irq*
- */sys/kernel/debug/hisi_sec2/<bdf>/qm/create_qp_err*
- */sys/kernel/debug/hisi_sec2/<bdf>/qm/mb_err*
- */sys/kernel/debug/hisi_sec2/<bdf>/qm/status*
- */sys/kernel/debug/hisi_sec2/<bdf>/qm/diff_regs*
- */sys/kernel/debug/hisi_sec2/<bdf>/sec_dfx/diff_regs*

- */sys/kernel/debug/hisi_sec2/<bdf>/sec_dfx/send_cnt*
- */sys/kernel/debug/hisi_sec2/<bdf>/sec_dfx/recv_cnt*
- */sys/kernel/debug/hisi_sec2/<bdf>/sec_dfx/send_busy_cnt*
- */sys/kernel/debug/hisi_sec2/<bdf>/sec_dfx/err_bd_cnt*
- */sys/kernel/debug/hisi_sec2/<bdf>/sec_dfx/invalid_req_cnt*
- */sys/kernel/debug/hisi_sec2/<bdf>/sec_dfx/done_flag_cnt*

5.2.88 File testing/debugfs-hisi-zip

Has the following ABI:

- */sys/kernel/debug/hisi_zip/<bdf>/comp_core[01]/regs*
- */sys/kernel/debug/hisi_zip/<bdf>/decomp_core[0-5]/regs*
- */sys/kernel/debug/hisi_zip/<bdf>/clear_enable*
- */sys/kernel/debug/hisi_zip/<bdf>/current_qm*
- */sys/kernel/debug/hisi_zip/<bdf>/alg_qos*
- */sys/kernel/debug/hisi_zip/<bdf>/qm/regs*
- */sys/kernel/debug/hisi_zip/<bdf>/qm/current_q*
- */sys/kernel/debug/hisi_zip/<bdf>/qm/clear_enable*
- */sys/kernel/debug/hisi_zip/<bdf>/qm/err_irq*
- */sys/kernel/debug/hisi_zip/<bdf>/qm/aeq_irq*
- */sys/kernel/debug/hisi_zip/<bdf>/qm/abnormal_irq*
- */sys/kernel/debug/hisi_zip/<bdf>/qm/create_qp_err*
- */sys/kernel/debug/hisi_zip/<bdf>/qm/mb_err*
- */sys/kernel/debug/hisi_zip/<bdf>/qm/status*
- */sys/kernel/debug/hisi_zip/<bdf>/qm/diff_regs*
- */sys/kernel/debug/hisi_zip/<bdf>/zip_dfx/diff_regs*
- */sys/kernel/debug/hisi_zip/<bdf>/zip_dfx/send_cnt*
- */sys/kernel/debug/hisi_zip/<bdf>/zip_dfx/recv_cnt*
- */sys/kernel/debug/hisi_zip/<bdf>/zip_dfx/send_busy_cnt*
- */sys/kernel/debug/hisi_zip/<bdf>/zip_dfx/err_bd_cnt*

5.2.89 File testing/debugfs-hyperv

Has the following ABI:

- */sys/kernel/debug/hyperv/<UUID>/fuzz_test_state*
- */sys/kernel/debug/hyperv/<UUID>/delay/fuzz_test_buffer_interrupt_delay*
- */sys/kernel/debug/hyperv/<UUID>/delay/fuzz_test_message_delay*

5.2.90 File testing/debugfs-ideapad

Has the following ABI:

- */sys/kernel/debug/ideapad/cfg*
- */sys/kernel/debug/ideapad/status*

5.2.91 File testing/debugfs-moxtet

Has the following ABI:

- */sys/kernel/debug/moxtet/input*
- */sys/kernel/debug/moxtet/output*

5.2.92 File testing/debugfs-olpc

Has the following ABI:

- */sys/kernel/debug/olpc-ec/cmd*

5.2.93 File testing/debugfs-pfo-nx-crypto

Has the following ABI:

- */sys/kernel/debug/nx-crypto/**

5.2.94 File testing/debugfs-pktcdvd

Has the following ABI:

- */sys/kernel/debug/pktcdvd/pktcdvd[0-7]*

5.2.95 File testing/debugfs-scmi

Has the following ABI:

- */sys/kernel/debug/scmi/<n>/instance_name*
- */sys/kernel/debug/scmi/<n>/atomic_threshold_us*
- */sys/kernel/debug/scmi/<n>/transport/type*
- */sys/kernel/debug/scmi/<n>/transport/is_atomic*
- */sys/kernel/debug/scmi/<n>/transport/max_rx_timeout_ms*
- */sys/kernel/debug/scmi/<n>/transport/max_msg_size*
- */sys/kernel/debug/scmi/<n>/transport/tx_max_msg*
- */sys/kernel/debug/scmi/<n>/transport/rx_max_msg*

5.2.96 File testing/debugfs-scmi-raw

Has the following ABI:

- */sys/kernel/debug/scmi/<n>/raw/message*
- */sys/kernel/debug/scmi/<n>/raw/message_async*
- */sys/kernel/debug/scmi/<n>/raw/errors*
- */sys/kernel/debug/scmi/<n>/raw/notification*
- */sys/kernel/debug/scmi/<n>/raw/reset*
- */sys/kernel/debug/scmi/<n>/raw/channels/<m>/message*
- */sys/kernel/debug/scmi/<n>/raw/channels/<m>/message_async*

5.2.97 File testing/debugfs-tpmi

Has the following ABI:

- */sys/kernel/debug/tpmi-<n>/pfs_dump*
- */sys/kernel/debug/tpmi-<n>/tpmi-id-<n>/mem_dump*
- */sys/kernel/debug/tpmi-<n>/tpmi-id-<n>/mem_write*

5.2.98 File testing/debugfs-turris-mox-rwtm

Has the following ABI:

- */sys/kernel/debug/turris-mox-rwtm/do_sign*

5.2.99 File testing/debugfs-vfio

Has the following ABI:

- */sys/kernel/debug/vfio*
- */sys/kernel/debug/vfio/<device>/migration*
- */sys/kernel/debug/vfio/<device>/migration/state*

5.2.100 File testing/debugfs-wilco-ec

Has the following ABI:

- */sys/kernel/debug/wilco_ec/h1_gpio*
- */sys/kernel/debug/wilco_ec/raw*

5.2.101 File testing/dell-smbios-wmi

Has the following ABI:

- */dev/wmi/dell-smbios*

5.2.102 File testing/dev-kmsg

Has the following ABI:

- */dev/kmsg*

5.2.103 File testing/devlink-resource-mlxsw

Has the following ABI:

- */kvd/*
- */kvd/linear*
- */kvd/hash_single*
- */kvd/hash_double*

5.2.104 File testing/evm

Has the following ABI:

- */sys/kernel/security/evm*
- */sys/kernel/security/*/evm*
- */sys/kernel/security/*/evm/evm_xattrs*

5.2.105 File testing/gpio-cdev

Has the following ABI:

- */dev/gpiochip[0-9]+*

5.2.106 File testing/ima_policy

Has the following ABI:

- */sys/kernel/security/*/ima/policy*

5.2.107 File testing/ppc-memtrace

Has the following ABI:

- */sys/kernel/debug/powerpc/memtrace*
- */sys/kernel/debug/powerpc/memtrace/enable*
- */sys/kernel/debug/powerpc/memtrace/<node-id>*
- */sys/kernel/debug/powerpc/memtrace/<node-id>/size*
- */sys/kernel/debug/powerpc/memtrace/<node-id>/start*
- */sys/kernel/debug/powerpc/memtrace/<node-id>/trace*

5.2.108 File testing/procfs-attr-current

Has the following ABI:

- */proc/*/attr/current*

5.2.109 File testing/procfs-attr-exec

Has the following ABI:

- */proc/*/attr/exec*

5.2.110 File testing/procfs-attr-prev

Has the following ABI:

- */proc/*/attr/prev*

5.2.111 File testing/procfs-diskstats

Has the following ABI:

- */proc/diskstats*

5.2.112 File testing/procfs-smaps_rollup

Has the following ABI:

- */proc/pid/smaps_rollup*

5.2.113 File testing/pstore

Has the following ABI:

- */sys/fs/pstore/..*
- */dev/pstore/..*

5.2.114 File testing/rtc-cdev

Has the following ABI:

- */dev/rtcX*

5.2.115 File testing/securityfs-secrets-coco

Has the following ABI:

- *security/secrets/coco*

5.2.116 File testing/sysfs-amd-pmc

Has the following ABI:

- */sys/bus/platform/drivers/amd_pmc/*/smu_fw_version*
- */sys/bus/platform/drivers/amd_pmc/*/smu_program*

5.2.117 File testing/sysfs-amd-pmf

Has the following ABI:

- */sys/devices/platform/*/cnqf_enable*

5.2.118 File testing/sysfs-ata

Has the following ABI:

- */sys/class/ata_**
- */sys/class/ata_port/ataX/nr_pmp_links*
- */sys/class/ata_port/ataX/idle_irq*
- */sys/class/ata_port/ataX/port_no*
- */sys/class/ata_link/linkX[.Y]/hw_sata_spd_limit*
- */sys/class/ata_link/linkX[.Y]/sata_spd_limit*
- */sys/class/ata_link/linkX[.Y]/sata_spd*
- */sys/class/ata_device/devX[.Y].Z/spdn_cnt*
- */sys/class/ata_device/devX[.Y].Z/gscr*
- */sys/class/ata_device/devX[.Y].Z/ering*
- */sys/class/ata_device/devX[.Y].Z/id*
- */sys/class/ata_device/devX[.Y].Z/pio_mode*
- */sys/class/ata_device/devX[.Y].Z/xfer_mode*
- */sys/class/ata_device/devX[.Y].Z/dma_mode*
- */sys/class/ata_device/devX[.Y].Z/class*
- */sys/class/ata_device/devX[.Y].Z/trim*

5.2.119 File testing/sysfs-block-aoe

Has the following ABI:

- */sys/block/etherd*/mac*
- */sys/block/etherd*/netif*
- */sys/block/etherd*/state*
- */sys/block/etherd*/firmware-version*
- */sys/block/etherd*/payload*

5.2.120 File testing/sysfs-block-bcache

Has the following ABI:

- */sys/block/<disk>/bcache/unregister*
- */sys/block/<disk>/bcache/clear_stats*
- */sys/block/<disk>/bcache/cache*
- */sys/block/<disk>/bcache/cache_hits*
- */sys/block/<disk>/bcache/cache_misses*

- */sys/block/<disk>/bcache/cache_hit_ratio*
- */sys/block/<disk>/bcache/sequential_cutoff*
- */sys/block/<disk>/bcache/bypassed*
- */sys/block/<disk>/bcache/writeback*
- */sys/block/<disk>/bcache/writeback_running*
- */sys/block/<disk>/bcache/writeback_delay*
- */sys/block/<disk>/bcache/writeback_percent*
- */sys/block/<disk>/bcache/synchronous*
- */sys/block/<disk>/bcache/discard*
- */sys/block/<disk>/bcache/bucket_size*
- */sys/block/<disk>/bcache/nbuckets*
- */sys/block/<disk>/bcache/tree_depth*
- */sys/block/<disk>/bcache/btree_cache_size*
- */sys/block/<disk>/bcache/written*
- */sys/block/<disk>/bcache/btree_written*

5.2.121 File testing/sysfs-block-device

Has the following ABI:

- */sys/block/*/device/sw_activity*
- */sys/block/*/device/unload_heads*
- */sys/block/*/device/ncq_prio_enable*
- */sys/block/*/device/sas_ncq_prio_enable*
- */sys/block/*/device/ncq_prio_supported*
- */sys/block/*/device/sas_ncq_prio_supported*
- */sys/block/*/device/cdl_supported*
- */sys/block/*/device/cdl_enable*

5.2.122 File testing/sysfs-block-dm

Has the following ABI:

- */sys/block/dm-<num>/dm/name*
- */sys/block/dm-<num>/dm/uuid*
- */sys/block/dm-<num>/dm/suspended*
- */sys/block/dm-<num>/dm/rq_based_seq_io_merge_deadline*
- */sys/block/dm-<num>/dm/use_blk_mq*

5.2.123 File testing/sysfs-block-loop

Has the following ABI:

- */sys/block/loopX/loop/autoclear*
- */sys/block/loopX/loop/backing_file*
- */sys/block/loopX/loop/offset*
- */sys/block/loopX/loop/sizelimit*
- */sys/block/loopX/loop/partscan*
- */sys/block/loopX/loop/dio*

5.2.124 File testing/sysfs-block-rnbd

Has the following ABI:

- */sys/block/rnbd<N>/rnbd/unmap_device*
- */sys/block/rnbd<N>/rnbd/state*
- */sys/block/rnbd<N>/rnbd/session*
- */sys/block/rnbd<N>/rnbd/mapping_path*
- */sys/block/rnbd<N>/rnbd/access_mode*
- */sys/block/rnbd<N>/rnbd/resize*
- */sys/block/rnbd<N>/rnbd/remap_device*
- */sys/block/rnbd<N>/rnbd/nr_poll_queues*

5.2.125 File testing/sysfs-block-rssd

Has the following ABI:

- */sys/block/rssd*/status*

5.2.126 File testing/sysfs-block-zram

Has the following ABI:

- */sys/block/zram<id>/disksize*
- */sys/block/zram<id>/initstate*
- */sys/block/zram<id>/reset*
- */sys/block/zram<id>/max_comp_streams*
- */sys/block/zram<id>/comp_algorithm*
- */sys/block/zram<id>/mem_used_max*
- */sys/block/zram<id>/mem_limit*
- */sys/block/zram<id>/compact*

- */sys/block/zram<id>/io_stat*
- */sys/block/zram<id>/mm_stat*
- */sys/block/zram<id>/debug_stat*
- */sys/block/zram<id>/backing_dev*
- */sys/block/zram<id>/idle*
- */sys/block/zram<id>/writeback*
- */sys/block/zram<id>/bd_stat*
- */sys/block/zram<id>/writeback_limit_enable*
- */sys/block/zram<id>/writeback_limit*
- */sys/block/zram<id>/recomp_algorithm*
- */sys/block/zram<id>/recompress*

5.2.127 File testing/sysfs-bus-acpi

Has the following ABI:

- */sys/bus/acpi/devices/.../path*
- */sys/bus/acpi/devices/.../modalias*
- */sys/bus/acpi/devices/.../hid*
- */sys/bus/acpi/devices/.../description*
- */sys/bus/acpi/devices/.../adr*
- */sys/bus/acpi/devices/.../uid*
- */sys/bus/acpi/devices/.../eject*
- */sys/bus/acpi/devices/.../status*
- */sys/bus/acpi/devices/.../hrv*

5.2.128 File testing/sysfs-bus-amba

Has the following ABI:

- */sys/bus/amba/devices/.../driver_override*

5.2.129 File testing/sysfs-bus-bcma

Has the following ABI:

- */sys/bus/bcma/devices/.../manuf*
- */sys/bus/bcma/devices/.../id*
- */sys/bus/bcma/devices/.../rev*
- */sys/bus/bcma/devices/.../class*

5.2.130 File testing/sysfs-bus-cdx

Has the following ABI:

- */sys/bus/cdx/rescan*
- */sys/bus/cdx/devices/.../vendor*
- */sys/bus/cdx/devices/.../device*
- */sys/bus/cdx/devices/.../subsystem_vendor*
- */sys/bus/cdx/devices/.../subsystem_device*
- */sys/bus/cdx/devices/.../class*
- */sys/bus/cdx/devices/.../revision*
- */sys/bus/cdx/devices/.../enable*
- */sys/bus/cdx/devices/.../reset*
- */sys/bus/cdx/devices/.../remove*
- */sys/bus/cdx/devices/.../resource<N>*
- */sys/bus/cdx/devices/.../modalias*

5.2.131 File testing/sysfs-bus-coreboot

Has the following ABI:

- */sys/bus/coreboot*
- */sys/bus/coreboot/devices/cbmem-<id>*
- */sys/bus/coreboot/devices/cbmem-<id>/address*
- */sys/bus/coreboot/devices/cbmem-<id>/size*
- */sys/bus/coreboot/devices/cbmem-<id>/mem*

5.2.132 File testing/sysfs-bus-coresight-devices-cti

Has the following ABI:

- */sys/bus/coresight/devices/<cti-name>/enable*
- */sys/bus/coresight/devices/<cti-name>/powered*
- */sys/bus/coresight/devices/<cti-name>/ctmid*
- */sys/bus/coresight/devices/<cti-name>/nr_trigger_cons*
- */sys/bus/coresight/devices/<cti-name>/triggers<N>/name*
- */sys/bus/coresight/devices/<cti-name>/triggers<N>/in_signals*
- */sys/bus/coresight/devices/<cti-name>/triggers<N>/in_types*
- */sys/bus/coresight/devices/<cti-name>/triggers<N>/out_signals*
- */sys/bus/coresight/devices/<cti-name>/triggers<N>/out_types*
- */sys/bus/coresight/devices/<cti-name>/regs/inout_sel*
- */sys/bus/coresight/devices/<cti-name>/regs/inen*
- */sys/bus/coresight/devices/<cti-name>/regs/outen*
- */sys/bus/coresight/devices/<cti-name>/regs/gate*
- */sys/bus/coresight/devices/<cti-name>/regs/asicctl*
- */sys/bus/coresight/devices/<cti-name>/regs/intack*
- */sys/bus/coresight/devices/<cti-name>/regs/appset*
- */sys/bus/coresight/devices/<cti-name>/regs/appclear*
- */sys/bus/coresight/devices/<cti-name>/regs/apppulse*
- */sys/bus/coresight/devices/<cti-name>/regs/chinstatus*
- */sys/bus/coresight/devices/<cti-name>/regs/choutstatus*
- */sys/bus/coresight/devices/<cti-name>/regs/triginstatus*
- */sys/bus/coresight/devices/<cti-name>/regs/trigoutstatus*
- */sys/bus/coresight/devices/<cti-name>/channels/trigin_attach*
- */sys/bus/coresight/devices/<cti-name>/channels/trigin_detach*
- */sys/bus/coresight/devices/<cti-name>/channels/trigout_attach*
- */sys/bus/coresight/devices/<cti-name>/channels/trigout_detach*
- */sys/bus/coresight/devices/<cti-name>/channels/chan_gate_enable*
- */sys/bus/coresight/devices/<cti-name>/channels/chan_gate_disable*
- */sys/bus/coresight/devices/<cti-name>/channels/chan_set*
- */sys/bus/coresight/devices/<cti-name>/channels/chan_clear*
- */sys/bus/coresight/devices/<cti-name>/channels/chan_pulse*
- */sys/bus/coresight/devices/<cti-name>/channels/trigout_filtered*

- */sys/bus/coresight/devices/<cti-name>/channels/trig_filter_enable*
- */sys/bus/coresight/devices/<cti-name>/channels/chan_inuse*
- */sys/bus/coresight/devices/<cti-name>/channels/chan_free*
- */sys/bus/coresight/devices/<cti-name>/channels/chan_xtrigs_sel*
- */sys/bus/coresight/devices/<cti-name>/channels/chan_xtrigs_in*
- */sys/bus/coresight/devices/<cti-name>/channels/chan_xtrigs_out*
- */sys/bus/coresight/devices/<cti-name>/channels/chan_xtrigs_reset*

5.2.133 File testing/sysfs-bus-coresight-devices-etb10

Has the following ABI:

- */sys/bus/coresight/devices/<memory_map>.etb/enable_sink*
- */sys/bus/coresight/devices/<memory_map>.etb/trigger_cntr*
- */sys/bus/coresight/devices/<memory_map>.etb/mgmt/rdp*
- */sys/bus/coresight/devices/<memory_map>.etb/mgmt/sts*
- */sys/bus/coresight/devices/<memory_map>.etb/mgmt/rrp*
- */sys/bus/coresight/devices/<memory_map>.etb/mgmt/rwp*
- */sys/bus/coresight/devices/<memory_map>.etb/mgmt/trg*
- */sys/bus/coresight/devices/<memory_map>.etb/mgmt/ctl*
- */sys/bus/coresight/devices/<memory_map>.etb/mgmt/ffsr*
- */sys/bus/coresight/devices/<memory_map>.etb/mgmt/ffcr*

5.2.134 File testing/sysfs-bus-coresight-devices-etm3x

Has the following ABI:

- */sys/bus/coresight/devices/<memory_map>.[etm|ptm]/enable_source*
- */sys/bus/coresight/devices/<memory_map>.[etm|ptm]/addr_idx*
- */sys/bus/coresight/devices/<memory_map>.[etm|ptm]/addr_acctype*
- */sys/bus/coresight/devices/<memory_map>.[etm|ptm]/addr_range*
- */sys/bus/coresight/devices/<memory_map>.[etm|ptm]/addr_single*
- */sys/bus/coresight/devices/<memory_map>.[etm|ptm]/addr_start*
- */sys/bus/coresight/devices/<memory_map>.[etm|ptm]/addr_stop*
- */sys/bus/coresight/devices/<memory_map>.[etm|ptm]/cntr_idx*
- */sys/bus/coresight/devices/<memory_map>.[etm|ptm]/cntr_event*
- */sys/bus/coresight/devices/<memory_map>.[etm|ptm]/cntr_val*
- */sys/bus/coresight/devices/<memory_map>.[etm|ptm]/cntr_rld_val*

- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/cntr_rld_event
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/ctxid_idx
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/ctxid_mask
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/ctxid_pid
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/enable_event
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/etmsr
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/fifofull_level
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/mode
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/nr_addr_cmp
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/nr_cntr
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/nr_ctxid_cmp
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/reset
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/seq_12_event
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/seq_13_event
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/seq_21_event
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/seq_23_event
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/seq_31_event
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/seq_32_event
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/curr_seq_state
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/sync_freq
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/timestamp_event
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/traceid
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/trigger_event
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/cpu
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/mgmt/etmccr
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/mgmt/etmccer
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/mgmt/etmscr
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/mgmt/etmidr
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/mgmt/etmcr
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/mgmt/etmtraceidr
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/mgmt/etmteevr
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/mgmt/etmtsscr
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/mgmt/etmtecr1
- /sys/bus/coresight/devices/<memory_map>.[etm|ptm]/mgmt/etmtecr2

5.2.135 File testing/sysfs-bus-coresight-devices-etm4x

Has the following ABI:

- */sys/bus/coresight/devices/etm<N>/enable_source*
- */sys/bus/coresight/devices/etm<N>/cpu*
- */sys/bus/coresight/devices/etm<N>/nr_pe_cmp*
- */sys/bus/coresight/devices/etm<N>/nr_addr_cmp*
- */sys/bus/coresight/devices/etm<N>/nr_cntr*
- */sys/bus/coresight/devices/etm<N>/nr_ext_inp*
- */sys/bus/coresight/devices/etm<N>/numcidc*
- */sys/bus/coresight/devices/etm<N>/numvmidc*
- */sys/bus/coresight/devices/etm<N>/nrseqstate*
- */sys/bus/coresight/devices/etm<N>/nr_resource*
- */sys/bus/coresight/devices/etm<N>/nr_ss_cmp*
- */sys/bus/coresight/devices/etm<N>/reset*
- */sys/bus/coresight/devices/etm<N>/mode*
- */sys/bus/coresight/devices/etm<N>/pe*
- */sys/bus/coresight/devices/etm<N>/event*
- */sys/bus/coresight/devices/etm<N>/event_instren*
- */sys/bus/coresight/devices/etm<N>/event_ts*
- */sys/bus/coresight/devices/etm<N>/syncfreq*
- */sys/bus/coresight/devices/etm<N>/cyc_threshold*
- */sys/bus/coresight/devices/etm<N>/bb_ctrl*
- */sys/bus/coresight/devices/etm<N>/event_vinst*
- */sys/bus/coresight/devices/etm<N>/s_exlevel_vinst*
- */sys/bus/coresight/devices/etm<N>/ns_exlevel_vinst*
- */sys/bus/coresight/devices/etm<N>/addr_idx*
- */sys/bus/coresight/devices/etm<N>/addr_instdatatype*
- */sys/bus/coresight/devices/etm<N>/addr_single*
- */sys/bus/coresight/devices/etm<N>/addr_range*
- */sys/bus/coresight/devices/etm<N>/seq_idx*
- */sys/bus/coresight/devices/etm<N>/seq_state*
- */sys/bus/coresight/devices/etm<N>/seq_event*
- */sys/bus/coresight/devices/etm<N>/seq_reset_event*
- */sys/bus/coresight/devices/etm<N>/cntr_idx*

- /sys/bus/coresight/devices/etm<N>/cntrldvr
- /sys/bus/coresight/devices/etm<N>/cntr_val
- /sys/bus/coresight/devices/etm<N>/cntr_ctrl
- /sys/bus/coresight/devices/etm<N>/res_idx
- /sys/bus/coresight/devices/etm<N>/res_ctrl
- /sys/bus/coresight/devices/etm<N>/ctxid_idx
- /sys/bus/coresight/devices/etm<N>/ctxid_pid
- /sys/bus/coresight/devices/etm<N>/ctxid_masks
- /sys/bus/coresight/devices/etm<N>/vmid_idx
- /sys/bus/coresight/devices/etm<N>/vmid_val
- /sys/bus/coresight/devices/etm<N>/vmid_masks
- /sys/bus/coresight/devices/etm<N>/addr_exlevel_s_ns
- /sys/bus/coresight/devices/etm<N>/vinst_pe_cmp_start_stop
- /sys/bus/coresight/devices/etm<N>/addr_cmp_view
- /sys/bus/coresight/devices/etm<N>/sshot_idx
- /sys/bus/coresight/devices/etm<N>/sshot_ctrl
- /sys/bus/coresight/devices/etm<N>/sshot_status
- /sys/bus/coresight/devices/etm<N>/sshot_pe_ctrl
- /sys/bus/coresight/devices/etm<N>/mgmt/trcoslsr
- /sys/bus/coresight/devices/etm<N>/mgmt/trcpdcr
- /sys/bus/coresight/devices/etm<N>/mgmt/trcpdsr
- /sys/bus/coresight/devices/etm<N>/mgmt/trclsr
- /sys/bus/coresight/devices/etm<N>/mgmt/trcauthstatus
- /sys/bus/coresight/devices/etm<N>/mgmt/trcdevid
- /sys/bus/coresight/devices/etm<N>/mgmt/trcdevarch
- /sys/bus/coresight/devices/etm<N>/mgmt/trcdevtype
- /sys/bus/coresight/devices/etm<N>/mgmt/trcpidr0
- /sys/bus/coresight/devices/etm<N>/mgmt/trcpidr1
- /sys/bus/coresight/devices/etm<N>/mgmt/trcpidr2
- /sys/bus/coresight/devices/etm<N>/mgmt/trcpidr3
- /sys/bus/coresight/devices/etm<N>/mgmt/trcconfig
- /sys/bus/coresight/devices/etm<N>/mgmt/trctraceid
- /sys/bus/coresight/devices/etm<N>/trcidr/trcidr0
- /sys/bus/coresight/devices/etm<N>/trcidr/trcidr1

- */sys/bus/coresight/devices/etm<N>/trcidr/trcidr2*
- */sys/bus/coresight/devices/etm<N>/trcidr/trcidr3*
- */sys/bus/coresight/devices/etm<N>/trcidr/trcidr4*
- */sys/bus/coresight/devices/etm<N>/trcidr/trcidr5*
- */sys/bus/coresight/devices/etm<N>/trcidr/trcidr8*
- */sys/bus/coresight/devices/etm<N>/trcidr/trcidr9*
- */sys/bus/coresight/devices/etm<N>/trcidr/trcidr10*
- */sys/bus/coresight/devices/etm<N>/trcidr/trcidr11*
- */sys/bus/coresight/devices/etm<N>/trcidr/trcidr12*
- */sys/bus/coresight/devices/etm<N>/trcidr/trcidr13*
- */sys/bus/coresight/devices/etm<N>/ts_source*

5.2.136 File testing/sysfs-bus-coresight-devices-funnel

Has the following ABI:

- */sys/bus/coresight/devices/<memory_map>.funnel/funnel_ctrl*
- */sys/bus/coresight/devices/<memory_map>.funnel/priority*

5.2.137 File testing/sysfs-bus-coresight-devices-stm

Has the following ABI:

- */sys/bus/coresight/devices/<memory_map>.stm/enable_source*
- */sys/bus/coresight/devices/<memory_map>.stm/hwevent_enable*
- */sys/bus/coresight/devices/<memory_map>.stm/hwevent_select*
- */sys/bus/coresight/devices/<memory_map>.stm/port_enable*
- */sys/bus/coresight/devices/<memory_map>.stm/port_select*
- */sys/bus/coresight/devices/<memory_map>.stm/status*
- */sys/bus/coresight/devices/<memory_map>.stm/traceid*

5.2.138 File testing/sysfs-bus-coresight-devices-tmc

Has the following ABI:

- */sys/bus/coresight/devices/<memory_map>.tmc/trigger_cntr*
- */sys/bus/coresight/devices/<memory_map>.tmc/mgmt/rsz*
- */sys/bus/coresight/devices/<memory_map>.tmc/mgmt/sts*
- */sys/bus/coresight/devices/<memory_map>.tmc/mgmt/rrp*
- */sys/bus/coresight/devices/<memory_map>.tmc/mgmt/rwp*

- */sys/bus/coresight/devices/<memory_map>.tmc/mgmt/trg*
- */sys/bus/coresight/devices/<memory_map>.tmc/mgmt/ctl*
- */sys/bus/coresight/devices/<memory_map>.tmc/mgmt/ffsr*
- */sys/bus/coresight/devices/<memory_map>.tmc/mgmt/ffcr*
- */sys/bus/coresight/devices/<memory_map>.tmc/mgmt/mode*
- */sys/bus/coresight/devices/<memory_map>.tmc/mgmt/devid*
- */sys/bus/coresight/devices/<memory_map>.tmc/buffer_size*
- */sys/bus/coresight/devices/<memory_map>.tmc/buf_modes_available*
- */sys/bus/coresight/devices/<memory_map>.tmc/buf_mode_preferred*

5.2.139 File testing/sysfs-bus-coresight-devices-tpdm

Has the following ABI:

- */sys/bus/coresight/devices/<tpdm-name>/integration_test*
- */sys/bus/coresight/devices/<tpdm-name>/reset_dataset*
- */sys/bus/coresight/devices/<tpdm-name>/dsb_trig_type*
- */sys/bus/coresight/devices/<tpdm-name>/dsb_trig_ts*
- */sys/bus/coresight/devices/<tpdm-name>/dsb_mode*
- */sys/bus/coresight/devices/<tpdm-name>/dsb_edge/ctrl_idx*
- */sys/bus/coresight/devices/<tpdm-name>/dsb_edge/ctrl_val*
- */sys/bus/coresight/devices/<tpdm-name>/dsb_edge/ctrl_mask*
- */sys/bus/coresight/devices/<tpdm-name>/dsb_edge/edcr[0:15]*
- */sys/bus/coresight/devices/<tpdm-name>/dsb_edge/edcmr[0:7]*
- */sys/bus/coresight/devices/<tpdm-name>/dsb_trig_patt/xpr[0:7]*
- */sys/bus/coresight/devices/<tpdm-name>/dsb_trig_patt/xpmr[0:7]*
- */sys/bus/coresight/devices/<tpdm-name>/dsb_patt/tpr[0:7]*
- */sys/bus/coresight/devices/<tpdm-name>/dsb_patt/tpmr[0:7]*
- */sys/bus/coresight/devices/<tpdm-name>/dsb_patt/enable_ts*
- */sys/bus/coresight/devices/<tpdm-name>/dsb_patt/set_type*
- */sys/bus/coresight/devices/<tpdm-name>/dsb_msr/msr[0:31]*

5.2.140 File testing/sysfs-bus-coresight-devices-trbe

Has the following ABI:

- */sys/bus/coresight/devices/trbe<cpu>/align*
- */sys/bus/coresight/devices/trbe<cpu>/flag*

5.2.141 File testing/sysfs-bus-coresight-devices-ultra_smb

Has the following ABI:

- */sys/bus/coresight/devices/ultra_smb<N>/enable_sink*
- */sys/bus/coresight/devices/ultra_smb<N>/mgmt/buf_size*
- */sys/bus/coresight/devices/ultra_smb<N>/mgmt/buf_status*
- */sys/bus/coresight/devices/ultra_smb<N>/mgmt/read_pos*
- */sys/bus/coresight/devices/ultra_smb<N>/mgmt/write_pos*

5.2.142 File testing/sysfs-bus-counter

Has the following ABI:

- */sys/bus/counter/devices/counterX/cascade_counts_enable*
- */sys/bus/counter/devices/counterX/external_input_phase_clock_select*
- */sys/bus/counter/devices/counterX/external_input_phase_clock_select_available*
- */sys/bus/counter/devices/counterX/countY/count*
- */sys/bus/counter/devices/counterX/countY/capture*
- */sys/bus/counter/devices/counterX/countY/ceiling*
- */sys/bus/counter/devices/counterX/countY/floor*
- */sys/bus/counter/devices/counterX/countY/count_mode*
- */sys/bus/counter/devices/counterX/countY/count_mode_available*
- */sys/bus/counter/devices/counterX/countY/error_noise_available*
- */sys/bus/counter/devices/counterX/countY/function_available*
- */sys/bus/counter/devices/counterX/countY/prescaler_available*
- */sys/bus/counter/devices/counterX/countY/signalZ_action_available*
- */sys/bus/counter/devices/counterX/countY/direction*
- */sys/bus/counter/devices/counterX/countY/enable*
- */sys/bus/counter/devices/counterX/countY/error_noise*
- */sys/bus/counter/devices/counterX/countY/function*
- */sys/bus/counter/devices/counterX/countY/name*
- */sys/bus/counter/devices/counterX/countY/prescaler*

- /sys/bus/counter/devices/counterX/countY/preset
- /sys/bus/counter/devices/counterX/countY/preset_enable
- /sys/bus/counter/devices/counterX/countY/signalZ_action
- /sys/bus/counter/devices/counterX/countY/num_overflows
- /sys/bus/counter/devices/counterX/cascade_counts_enable_component_id
- /sys/bus/counter/devices/counterX/external_input_phase_clock_select_component_id
- /sys/bus/counter/devices/counterX/countY/capture_component_id
- /sys/bus/counter/devices/counterX/countY/ceiling_component_id
- /sys/bus/counter/devices/counterX/countY/floor_component_id
- /sys/bus/counter/devices/counterX/countY/count_mode_component_id
- /sys/bus/counter/devices/counterX/countY/direction_component_id
- /sys/bus/counter/devices/counterX/countY/enable_component_id
- /sys/bus/counter/devices/counterX/countY/error_noise_component_id
- /sys/bus/counter/devices/counterX/countY/prescaler_component_id
- /sys/bus/counter/devices/counterX/countY/preset_component_id
- /sys/bus/counter/devices/counterX/countY/preset_enable_component_id
- /sys/bus/counter/devices/counterX/countY/signalZ_action_component_id
- /sys/bus/counter/devices/counterX/countY/num_overflows_component_id
- /sys/bus/counter/devices/counterX/signalY/cable_fault_component_id
- /sys/bus/counter/devices/counterX/signalY/cable_fault_enable_component_id
- /sys/bus/counter/devices/counterX/signalY/filter_clock_prescaler_component_id
- /sys/bus/counter/devices/counterX/signalY/index_polarity_component_id
- /sys/bus/counter/devices/counterX/signalY/polarity_component_id
- /sys/bus/counter/devices/counterX/signalY/synchronous_mode_component_id
- /sys/bus/counter/devices/counterX/signalY/frequency_component_id
- /sys/bus/counter/devices/counterX/countY/spike_filter_ns
- /sys/bus/counter/devices/counterX/events_queue_size
- /sys/bus/counter/devices/counterX/name
- /sys/bus/counter/devices/counterX/num_counts
- /sys/bus/counter/devices/counterX/num_signals
- /sys/bus/counter/devices/counterX/signalY/cable_fault
- /sys/bus/counter/devices/counterX/signalY/cable_fault_enable
- /sys/bus/counter/devices/counterX/signalY/filter_clock_prescaler
- /sys/bus/counter/devices/counterX/signalY/index_polarity

- */sys/bus/counter/devices/counterX/signalY/index_polarity_available*
- */sys/bus/counter/devices/counterX/signalY/synchronous_mode_available*
- */sys/bus/counter/devices/counterX/signalY/polarity*
- */sys/bus/counter/devices/counterX/signalY/name*
- */sys/bus/counter/devices/counterX/signalY/signal*
- */sys/bus/counter/devices/counterX/signalY/synchronous_mode*
- */sys/bus/counter/devices/counterX/signalY/frequency*

5.2.143 File testing/sysfs-bus-css

Has the following ABI:

- */sys/bus/css/devices/.../type*
- */sys/bus/css/devices/.../modalias*
- */sys/bus/css/drivers/io_subchannel/.../chpids*
- */sys/bus/css/drivers/io_subchannel/.../pimpampom*
- */sys/bus/css/devices/.../driver_override*

5.2.144 File testing/sysfs-bus-cxl

Has the following ABI:

- */sys/bus/cxl/flush*
- */sys/bus/cxl/devices/memX/firmware_version*
- */sys/bus/cxl/devices/memX/ram/size*
- */sys/bus/cxl/devices/memX/ram/qos_class*
- */sys/bus/cxl/devices/memX/pmem/size*
- */sys/bus/cxl/devices/memX/pmem/qos_class*
- */sys/bus/cxl/devices/memX/serial*
- */sys/bus/cxl/devices/memX/numa_node*
- */sys/bus/cxl/devices/memX/security/state*
- */sys/bus/cxl/devices/memX/security/sanitize*
- */sys/bus/cxl/devices/memX/firmware/*
- */sys/bus/cxl/devices/*/devtype*
- */sys/bus/cxl/devices/*/modalias*
- */sys/bus/cxl/devices/portX/uport*
- */sys/bus/cxl/devices/{port,endpoint}X/parent_dport*
- */sys/bus/cxl/devices/portX/dportY*

- */sys/bus/cxl/devices/portX/decoders_committed*
- */sys/bus/cxl/devices/decoderX.Y*
- */sys/bus/cxl/devices/decoderX.Y/{start,size}*
- */sys/bus/cxl/devices/decoderX.Y/locked*
- */sys/bus/cxl/devices/decoderX.Y/target_list*
- */sys/bus/cxl/devices/decoderX.Y/cap_{pmem,ram,type2,type3}*
- */sys/bus/cxl/devices/decoderX.Y/target_type*
- */sys/bus/cxl/devices/endpointX/CDAT*
- */sys/bus/cxl/devices/decoderX.Y/mode*
- */sys/bus/cxl/devices/decoderX.Y/dpa_resource*
- */sys/bus/cxl/devices/decoderX.Y/dpa_size*
- */sys/bus/cxl/devices/decoderX.Y/interleave_ways*
- */sys/bus/cxl/devices/decoderX.Y/interleave_granularity*
- */sys/bus/cxl/devices/decoderX.Y/create_{pmem,ram}_region*
- */sys/bus/cxl/devices/decoderX.Y/delete_region*
- */sys/bus/cxl/devices/decoderX.Y/qos_class*
- */sys/bus/cxl/devices/regionZ/uuid*
- */sys/bus/cxl/devices/regionZ/interleave_granularity*
- */sys/bus/cxl/devices/regionZ/interleave_ways*
- */sys/bus/cxl/devices/regionZ/size*
- */sys/bus/cxl/devices/regionZ/mode*
- */sys/bus/cxl/devices/regionZ/resource*
- */sys/bus/cxl/devices/regionZ/target[0..N]*
- */sys/bus/cxl/devices/regionZ/commit*
- */sys/bus/cxl/devices/memX/trigger_poison_list*

5.2.145 File testing/sysfs-bus-dfl

Has the following ABI:

- */sys/bus/dfl/devices/dfl_dev.X/type*
- */sys/bus/dfl/devices/dfl_dev.X/feature_id*

5.2.146 File testing/sysfs-bus-dfl-devices-emif

Has the following ABI:

- */sys/bus/dfl/devices/dfl_dev.X/infX_cal_fail*
- */sys/bus/dfl/devices/dfl_dev.X/infX_init_done*
- */sys/bus/dfl/devices/dfl_dev.X/infX_clear*

5.2.147 File testing/sysfs-bus-dfl-devices-n3000-nios

Has the following ABI:

- */sys/bus/dfl/devices/dfl_dev.X/fec_mode*
- */sys/bus/dfl/devices/dfl_dev.X/retimer_A_mode*
- */sys/bus/dfl/devices/dfl_dev.X/retimer_B_mode*
- */sys/bus/dfl/devices/dfl_dev.X/nios_fw_version*

5.2.148 File testing/sysfs-bus-event_source-devices-caps

Has the following ABI:

- */sys/bus/event_source/devices/<dev>/caps*

5.2.149 File testing/sysfs-bus-event_source-devices-dfl_fme

Has the following ABI:

- */sys/bus/event_source/devices/dfl_fmeX/format*
- */sys/bus/event_source/devices/dfl_fmeX/cpumask*
- */sys/bus/event_source/devices/dfl_fmeX/events*

5.2.150 File testing/sysfs-bus-event_source-devices-dsa

Has the following ABI:

- */sys/bus/event_source/devices/dsa*/format*
- */sys/bus/event_source/devices/dsa*/cpumask*

5.2.151 File testing/sysfs-bus-event_source-devices-events

Has the following ABI:

- */sys/devices/cpu/events/*
- */sys/bus/event_source/devices/<pmu>/events/<event>*
- */sys/bus/event_source/devices/<pmu>/events/<event>.unit*
- */sys/bus/event_source/devices/<pmu>/events/<event>.scale*

5.2.152 File testing/sysfs-bus-event_source-devices-format

Has the following ABI:

- */sys/bus/event_source/devices/<dev>/format*

5.2.153 File testing/sysfs-bus-event_source-devices-hv_24x7

Has the following ABI:

- */sys/bus/event_source/devices/hv_24x7/format*
- */sys/bus/event_source/devices/hv_24x7/interface/catalog*
- */sys/bus/event_source/devices/hv_24x7/interface/catalog_length*
- */sys/bus/event_source/devices/hv_24x7/interface/catalog_version*
- */sys/devices/hv_24x7/interface/sockets*
- */sys/devices/hv_24x7/interface/chipspersocket*
- */sys/devices/hv_24x7/interface/coresperchip*
- */sys/devices/hv_24x7/cpumask*
- */sys/bus/event_source/devices/hv_24x7/event_descs/<event-name>*
- */sys/bus/event_source/devices/hv_24x7/event_long_descs/<event-name>*

5.2.154 File testing/sysfs-bus-event_source-devices-hv_gpci

Has the following ABI:

- */sys/bus/event_source/devices/hv_gpci/format*
- */sys/bus/event_source/devices/hv_gpci/interface/collect_privileged*
- */sys/bus/event_source/devices/hv_gpci/interface/ga*
- */sys/bus/event_source/devices/hv_gpci/interface/expanded*
- */sys/bus/event_source/devices/hv_gpci/interface/lab*
- */sys/bus/event_source/devices/hv_gpci/interface/version*
- */sys/bus/event_source/devices/hv_gpci/interface/kernel_version*
- */sys/devices/hv_gpci/cpumask*

- */sys/devices/hv_gpci/interface/processor_bus_topology*
- */sys/devices/hv_gpci/interface/processor_config*
- */sys/devices/hv_gpci/interface/affinity_domain_via_virtual_processor*
- */sys/devices/hv_gpci/interface/affinity_domain_via_domain*
- */sys/devices/hv_gpci/interface/affinity_domain_via_partition*

5.2.155 File testing/sysfs-bus-event_source-devices-iommu

Has the following ABI:

- */sys/bus/event_source/devices/dmar*/format*
- */sys/bus/event_source/devices/dmar*/cpumask*

5.2.156 File testing/sysfs-bus-event_source-devices-uncore

Has the following ABI:

- */sys/bus/event_source/devices/uncore_*/alias*

5.2.157 File testing/sysfs-bus-fcoe

Has the following ABI:

- */sys/bus/fcoe/*
- */sys/bus/fcoe/devices/ctlr_X*
- */sys/bus/fcoe/devices/fcf_X*

5.2.158 File testing/sysfs-bus-fsi

Has the following ABI:

- */sys/bus/platform/devices/../fsi-master/fsi0/rescan*
- */sys/bus/platform/devices/../fsi-master/fsi0/break*
- */sys/bus/platform/devices/../fsi-master/fsi0/slave@00:00/term*
- */sys/bus/platform/devices/../fsi-master/fsi0/slave@00:00/raw*
- */sys/bus/platform/devices/../cfam_reset*

5.2.159 File testing/sysfs-bus-fsi-devices-sbefifo

Has the following ABI:

- */sys/bus/fsi/devices/XX.XX.00:06/sbefifoX/timeout*

5.2.160 File testing/sysfs-bus-fsl-mc

Has the following ABI:

- */sys/bus/fsl-mc/drivers/.../bind*
- */sys/bus/fsl-mc/drivers/.../unbind*

5.2.161 File testing/sysfs-bus-hsi

Has the following ABI:

- */sys/bus/hsi*
- */sys/bus/hsi/devices/.../modalias*

5.2.162 File testing/sysfs-bus-i2c-devices-bq32k

Has the following ABI:

- */sys/bus/i2c/devices/.../trickle_charge_bypass*

5.2.163 File testing/sysfs-bus-i2c-devices-fsa9480

Has the following ABI:

- */sys/bus/i2c/devices/.../device*
- */sys/bus/i2c/devices/.../switch*

5.2.164 File testing/sysfs-bus-i2c-devices-hm6352

Has the following ABI:

- */sys/bus/i2c/devices/.../heading0_input*
- */sys/bus/i2c/devices/.../power_state*
- */sys/bus/i2c/devices/.../calibration*

5.2.165 File testing/sysfs-bus-i2c-devices-lm3533

Has the following ABI:

- */sys/bus/i2c/devices/.../output_hvled[n]*
- */sys/bus/i2c/devices/.../output_lvled[n]*

5.2.166 File testing/sysfs-bus-i2c-devices-pca954x

Has the following ABI:

- */sys/bus/i2c/.../idle_state*

5.2.167 File testing/sysfs-bus-i3c

Has the following ABI:

- */sys/bus/i3c/devices/i3c-<bus-id>*
- */sys/bus/i3c/devices/i3c-<bus-id>/current_master*
- */sys/bus/i3c/devices/i3c-<bus-id>/mode*
- */sys/bus/i3c/devices/i3c-<bus-id>/i3c_scl_frequency*
- */sys/bus/i3c/devices/i3c-<bus-id>/i2c_scl_frequency*
- */sys/bus/i3c/devices/i3c-<bus-id>/dynamic_address*
- */sys/bus/i3c/devices/i3c-<bus-id>/bcr*
- */sys/bus/i3c/devices/i3c-<bus-id>/dcr*
- */sys/bus/i3c/devices/i3c-<bus-id>/pid*
- */sys/bus/i3c/devices/i3c-<bus-id>/hdrcap*
- */sys/bus/i3c/devices/i3c-<bus-id>/hotjoin*
- */sys/bus/i3c/devices/i3c-<bus-id>/<bus-id>-<device-pid>*
- */sys/bus/i3c/devices/i3c-<bus-id>/<bus-id>-<device-pid>/dynamic_address*
- */sys/bus/i3c/devices/i3c-<bus-id>/<bus-id>-<device-pid>/bcr*
- */sys/bus/i3c/devices/i3c-<bus-id>/<bus-id>-<device-pid>/dcr*
- */sys/bus/i3c/devices/i3c-<bus-id>/<bus-id>-<device-pid>/pid*
- */sys/bus/i3c/devices/i3c-<bus-id>/<bus-id>-<device-pid>/hdrcap*
- */sys/bus/i3c/devices/<bus-id>-<device-pid>*

5.2.168 File testing/sysfs-bus-iio

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX*
- */sys/bus/iio/devices/triggerX*
- */sys/bus/iio/devices/iio:deviceX/buffer*
- */sys/bus/iio/devices/iio:deviceX/name*
- */sys/bus/iio/devices/iio:deviceX/label*
- */sys/bus/iio/devices/iio:deviceX/current_timestamp_clock*
- */sys/bus/iio/devices/iio:deviceX/sampling_frequency*
- */sys/bus/iio/devices/iio:deviceX/in_intensity_sampling_frequency*
- */sys/bus/iio/devices/iio:deviceX/buffer/sampling_frequency*
- */sys/bus/iio/devices/triggerX/sampling_frequency*
- */sys/bus/iio/devices/iio:deviceX/sampling_frequency_available*
- */sys/bus/iio/devices/iio:deviceX/in_intensity_sampling_frequency_available*
- */sys/bus/iio/devices/iio:deviceX/in_proximity_sampling_frequency_available*
- */sys/.../iio:deviceX/buffer/sampling_frequency_available*
- */sys/bus/iio/devices/triggerX/sampling_frequency_available*
- */sys/bus/iio/devices/iio:deviceX/oversampling_ratio*
- */sys/bus/iio/devices/iio:deviceX/oversampling_ratio_available*
- */sys/bus/iio/devices/iio:deviceX/in_voltageY_raw*
- */sys/bus/iio/devices/iio:deviceX/in_voltageY_supply_raw*
- */sys/bus/iio/devices/iio:deviceX/in_voltageY_i_raw*
- */sys/bus/iio/devices/iio:deviceX/in_voltageY_q_raw*
- */sys/bus/iio/devices/iio:deviceX/in_voltageY-voltageZ_raw*
- */sys/bus/iio/devices/iio:deviceX/in_currentY_raw*
- */sys/bus/iio/devices/iio:deviceX/in_currentY_supply_raw*
- */sys/bus/iio/devices/iio:deviceX/in_powerY_raw*
- */sys/bus/iio/devices/iio:deviceX/in_capacitanceY_raw*
- */sys/.../iio:deviceX/in_capacitanceY-capacitanceZ_raw*
- */sys/.../iio:deviceX/in_capacitanceY-capacitanceZ_zeropoint*
- */sys/bus/iio/devices/iio:deviceX/in_temp_raw*
- */sys/bus/iio/devices/iio:deviceX/in_tempX_raw*
- */sys/bus/iio/devices/iio:deviceX/in_temp_x_raw*
- */sys/bus/iio/devices/iio:deviceX/in_temp_y_raw*

- */sys/bus/iio/devices/iio:deviceX/in_temp_ambient_raw*
- */sys/bus/iio/devices/iio:deviceX/in_temp_object_raw*
- */sys/bus/iio/devices/iio:deviceX/in_tempX_input*
- */sys/bus/iio/devices/iio:deviceX/in_accel_x_raw*
- */sys/bus/iio/devices/iio:deviceX/in_accel_y_raw*
- */sys/bus/iio/devices/iio:deviceX/in_accel_z_raw*
- */sys/bus/iio/devices/iio:deviceX/in_accel_linear_x_raw*
- */sys/bus/iio/devices/iio:deviceX/in_accel_linear_y_raw*
- */sys/bus/iio/devices/iio:deviceX/in_accel_linear_z_raw*
- */sys/bus/iio/devices/iio:deviceX/in_gravity_x_raw*
- */sys/bus/iio/devices/iio:deviceX/in_gravity_y_raw*
- */sys/bus/iio/devices/iio:deviceX/in_gravity_z_raw*
- */sys/bus/iio/devices/iio:deviceX/in_deltaangl_x_raw*
- */sys/bus/iio/devices/iio:deviceX/in_deltaangl_y_raw*
- */sys/bus/iio/devices/iio:deviceX/in_deltaangl_z_raw*
- */sys/bus/iio/devices/iio:deviceX/in_deltavelocity_x_raw*
- */sys/bus/iio/devices/iio:deviceX/in_deltavelocity_y_raw*
- */sys/bus/iio/devices/iio:deviceX/in_deltavelocity_z_raw*
- */sys/bus/iio/devices/iio:deviceX/in_angl_raw*
- */sys/bus/iio/devices/iio:deviceX/in_anglY_raw*
- */sys/bus/iio/devices/iio:deviceX/in_positionrelative_x_raw*
- */sys/bus/iio/devices/iio:deviceX/in_positionrelative_y_raw*
- */sys/bus/iio/devices/iio:deviceX/in_anglvel_x_raw*
- */sys/bus/iio/devices/iio:deviceX/in_anglvel_y_raw*
- */sys/bus/iio/devices/iio:deviceX/in_anglvel_z_raw*
- */sys/bus/iio/devices/iio:deviceX/in_incli_x_raw*
- */sys/bus/iio/devices/iio:deviceX/in_incli_y_raw*
- */sys/bus/iio/devices/iio:deviceX/in_incli_z_raw*
- */sys/bus/iio/devices/iio:deviceX/in_magn_x_raw*
- */sys/bus/iio/devices/iio:deviceX/in_magn_y_raw*
- */sys/bus/iio/devices/iio:deviceX/in_magn_z_raw*
- */sys/bus/iio/devices/iio:deviceX/in_accel_x_peak_raw*
- */sys/bus/iio/devices/iio:deviceX/in_accel_y_peak_raw*
- */sys/bus/iio/devices/iio:deviceX/in_accel_z_peak_raw*

- /sys/bus/iio/devices/iio:deviceX/in_humidityrelative_peak_raw
- /sys/bus/iio/devices/iio:deviceX/in_temp_peak_raw
- /sys/bus/iio/devices/iio:deviceX/in_humidityrelative_trough_raw
- /sys/bus/iio/devices/iio:deviceX/in_temp_trough_raw
- /sys/bus/iio/devices/iio:deviceX/in_accel_xyz_squared_peak_raw
- /sys/bus/iio/devices/iio:deviceX/in_pressureY_raw
- /sys/bus/iio/devices/iio:deviceX/in_pressure_raw
- /sys/bus/iio/devices/iio:deviceX/in_pressureY_input
- /sys/bus/iio/devices/iio:deviceX/in_pressure_input
- /sys/bus/iio/devices/iio:deviceX/in_humidityrelative_raw
- /sys/bus/iio/devices/iio:deviceX/in_humidityrelative_input
- /sys/bus/iio/devices/iio:deviceX/in_X_mean_raw
- /sys/bus/iio/devices/iio:deviceX/in_accel_offset
- /sys/bus/iio/devices/iio:deviceX/in_accel_x_offset
- /sys/bus/iio/devices/iio:deviceX/in_accel_y_offset
- /sys/bus/iio/devices/iio:deviceX/in_accel_z_offset
- /sys/bus/iio/devices/iio:deviceX/in_voltageY_offset
- /sys/bus/iio/devices/iio:deviceX/in_voltage_offset
- /sys/bus/iio/devices/iio:deviceX/in_voltageY_i_offset
- /sys/bus/iio/devices/iio:deviceX/in_voltageY_q_offset
- /sys/bus/iio/devices/iio:deviceX/in_voltage_q_offset
- /sys/bus/iio/devices/iio:deviceX/in_voltage_i_offset
- /sys/bus/iio/devices/iio:deviceX/in_currentY_offset
- /sys/bus/iio/devices/iio:deviceX/in_current_offset
- /sys/bus/iio/devices/iio:deviceX/in_currentY_i_offset
- /sys/bus/iio/devices/iio:deviceX/in_currentY_q_offset
- /sys/bus/iio/devices/iio:deviceX/in_current_q_offset
- /sys/bus/iio/devices/iio:deviceX/in_current_i_offset
- /sys/bus/iio/devices/iio:deviceX/in_tempY_offset
- /sys/bus/iio/devices/iio:deviceX/in_temp_offset
- /sys/bus/iio/devices/iio:deviceX/in_pressureY_offset
- /sys/bus/iio/devices/iio:deviceX/in_pressure_offset
- /sys/bus/iio/devices/iio:deviceX/in_humidityrelative_offset
- /sys/bus/iio/devices/iio:deviceX/in_magn_offset

- */sys/bus/iio/devices/iio:deviceX/in_rot_offset*
- */sys/bus/iio/devices/iio:deviceX/in_angl_offset*
- */sys/bus/iio/devices/iio:deviceX/in_capacitanceX_offset*
- */sys/bus/iio/devices/iio:deviceX/in_voltageY_scale*
- */sys/bus/iio/devices/iio:deviceX/in_voltageY_i_scale*
- */sys/bus/iio/devices/iio:deviceX/in_voltageY_q_scale*
- */sys/bus/iio/devices/iio:deviceX/in_voltageY_supply_scale*
- */sys/bus/iio/devices/iio:deviceX/in_voltage_scale*
- */sys/bus/iio/devices/iio:deviceX/in_voltage_i_scale*
- */sys/bus/iio/devices/iio:deviceX/in_voltage_q_scale*
- */sys/bus/iio/devices/iio:deviceX/in_voltage-voltage_scale*
- */sys/bus/iio/devices/iio:deviceX/out_voltageY_scale*
- */sys/bus/iio/devices/iio:deviceX/out_altvoltageY_scale*
- */sys/bus/iio/devices/iio:deviceX/in_currentY_scale*
- */sys/bus/iio/devices/iio:deviceX/in_currentY_supply_scale*
- */sys/bus/iio/devices/iio:deviceX/in_current_scale*
- */sys/bus/iio/devices/iio:deviceX/in_currentY_i_scale*
- */sys/bus/iio/devices/iio:deviceX/in_currentY_q_scale*
- */sys/bus/iio/devices/iio:deviceX/in_current_i_scale*
- */sys/bus/iio/devices/iio:deviceX/in_current_q_scale*
- */sys/bus/iio/devices/iio:deviceX/in_accel_scale*
- */sys/bus/iio/devices/iio:deviceX/in_accel_peak_scale*
- */sys/bus/iio/devices/iio:deviceX/in_anglvel_scale*
- */sys/bus/iio/devices/iio:deviceX/in_energy_scale*
- */sys/bus/iio/devices/iio:deviceX/in_distance_scale*
- */sys/bus/iio/devices/iio:deviceX/in_magn_scale*
- */sys/bus/iio/devices/iio:deviceX/in_magn_x_scale*
- */sys/bus/iio/devices/iio:deviceX/in_magn_y_scale*
- */sys/bus/iio/devices/iio:deviceX/in_magn_z_scale*
- */sys/bus/iio/devices/iio:deviceX/in_rot_from_north_magnetic_scale*
- */sys/bus/iio/devices/iio:deviceX/in_rot_from_north_true_scale*
- */sys/bus/iio/devices/iio:deviceX/in_rot_from_north_magnetic_tilt_comp_scale*
- */sys/bus/iio/devices/iio:deviceX/in_rot_from_north_true_tilt_comp_scale*
- */sys/bus/iio/devices/iio:deviceX/in_pressureY_scale*

- /sys/bus/iio/devices/iio:deviceX/in_pressure_scale
- /sys/bus/iio/devices/iio:deviceX/in_humidityrelative_scale
- /sys/bus/iio/devices/iio:deviceX/in_velocity_sqrt(x^2+y^2+z^2)_scale
- /sys/bus/iio/devices/iio:deviceX/in_illuminance_scale
- /sys/bus/iio/devices/iio:deviceX/in_countY_scale
- /sys/bus/iio/devices/iio:deviceX/in_deltaangl_scale
- /sys/bus/iio/devices/iio:deviceX/in_deltavelocity_scale
- /sys/bus/iio/devices/iio:deviceX/in_angl_scale
- /sys/bus/iio/devices/iio:deviceX/in_intensity_x_scale
- /sys/bus/iio/devices/iio:deviceX/in_intensity_y_scale
- /sys/bus/iio/devices/iio:deviceX/in_intensity_z_scale
- /sys/bus/iio/devices/iio:deviceX/in_concentration_co2_scale
- /sys/bus/iio/devices/iio:deviceX/in_accel_x_calibbias
- /sys/bus/iio/devices/iio:deviceX/in_accel_y_calibbias
- /sys/bus/iio/devices/iio:deviceX/in_accel_z_calibbias
- /sys/bus/iio/devices/iio:deviceX/in_anglvel_x_calibbias
- /sys/bus/iio/devices/iio:deviceX/in_anglvel_y_calibbias
- /sys/bus/iio/devices/iio:deviceX/in_anglvel_z_calibbias
- /sys/bus/iio/devices/iio:deviceX/in_illuminance0_calibbias
- /sys/bus/iio/devices/iio:deviceX/in_proximity0_calibbias
- /sys/bus/iio/devices/iio:deviceX/in_pressureY_calibbias
- /sys/bus/iio/devices/iio:deviceX/in_pressure_calibbias
- /sys/bus/iio/devices/iio:deviceX/in_accel_calibbias_available
- /sys/bus/iio/devices/iio:deviceX/in_anglvel_calibbias_available
- /sys/bus/iio/devices/iio:deviceX/in_voltageY_calibscale
- /sys/bus/iio/devices/iio:deviceX/in_voltageY_supply_calibscale
- /sys/bus/iio/devices/iio:deviceX/in_voltageY_i_calibscale
- /sys/bus/iio/devices/iio:deviceX/in_voltageY_q_calibscale
- /sys/bus/iio/devices/iio:deviceX/in_voltage_i_calibscale
- /sys/bus/iio/devices/iio:deviceX/in_voltage_q_calibscale
- /sys/bus/iio/devices/iio:deviceX/in_altvoltage_calibscale
- /sys/bus/iio/devices/iio:deviceX/in_voltage_calibscale
- /sys/bus/iio/devices/iio:deviceX/in_accel_x_calibscale
- /sys/bus/iio/devices/iio:deviceX/in_accel_y_calibscale

- */sys/bus/iio/devices/iio:deviceX/in_accel_z_calibscale*
- */sys/bus/iio/devices/iio:deviceX/in_anglvel_x_calibscale*
- */sys/bus/iio/devices/iio:deviceX/in_anglvel_y_calibscale*
- */sys/bus/iio/devices/iio:deviceX/in_anglvel_z_calibscale*
- */sys/bus/iio/devices/iio:deviceX/in_illuminance0_calibscale*
- */sys/bus/iio/devices/iio:deviceX/in_proximity0_calibscale*
- */sys/bus/iio/devices/iio:deviceX/in_pressureY_calibscale*
- */sys/bus/iio/devices/iio:deviceX/in_pressure_calibscale*
- */sys/bus/iio/devices/iio:deviceX/in_illuminance_calibscale*
- */sys/bus/iio/devices/iio:deviceX/in_activity_calibgender*
- */sys/bus/iio/devices/iio:deviceX/in_energy_calibgender*
- */sys/bus/iio/devices/iio:deviceX/in_distance_calibgender*
- */sys/bus/iio/devices/iio:deviceX/in_velocity_calibgender*
- */sys/bus/iio/devices/iio:deviceX/in_activity_calibgender_available*
- */sys/bus/iio/devices/iio:deviceX/in_energy_calibgender_available*
- */sys/bus/iio/devices/iio:deviceX/in_distance_calibgender_available*
- */sys/bus/iio/devices/iio:deviceX/in_velocity_calibgender_available*
- */sys/bus/iio/devices/iio:deviceX/in_activity_calibheight*
- */sys/bus/iio/devices/iio:deviceX/in_energy_calibheight*
- */sys/bus/iio/devices/iio:deviceX/in_distance_calibheight*
- */sys/bus/iio/devices/iio:deviceX/in_velocity_calibheight*
- */sys/bus/iio/devices/iio:deviceX/in_energy_calibweight*
- */sys/bus/iio/devices/iio:deviceX/in_accel_scale_available*
- */sys/.../iio:deviceX/in_anglvel_scale_available*
- */sys/.../iio:deviceX/in_magn_scale_available*
- */sys/.../iio:deviceX/in_illuminance_scale_available*
- */sys/.../iio:deviceX/in_intensity_scale_available*
- */sys/.../iio:deviceX/in_proximity_scale_available*
- */sys/.../iio:deviceX/in_voltageX_scale_available*
- */sys/.../iio:deviceX/in_voltage-voltage_scale_available*
- */sys/.../iio:deviceX/out_voltageX_scale_available*
- */sys/.../iio:deviceX/out_altvoltageX_scale_available*
- */sys/.../iio:deviceX/in_capacitance_scale_available*
- */sys/.../iio:deviceX/in_pressure_scale_available*

- */sys/.../iio:deviceX/in_pressureY_scale_available*
- */sys/bus/iio/devices/iio:deviceX/out_voltageY_hardwaregain*
- */sys/bus/iio/devices/iio:deviceX/in_intensity_hardwaregain*
- */sys/bus/iio/devices/iio:deviceX/in_intensity_red_hardwaregain*
- */sys/bus/iio/devices/iio:deviceX/in_intensity_green_hardwaregain*
- */sys/bus/iio/devices/iio:deviceX/in_intensity_blue_hardwaregain*
- */sys/bus/iio/devices/iio:deviceX/in_intensity_clear_hardwaregain*
- */sys/bus/iio/devices/iio:deviceX/in_intensity_hardwaregain_available*
- */sys/.../in_accel_filter_low_pass_3db_frequency*
- */sys/.../in_magn_filter_low_pass_3db_frequency*
- */sys/.../in_anglvel_filter_low_pass_3db_frequency*
- */sys/.../in_accel_filter_high_pass_3db_frequency*
- */sys/.../in_anglvel_filter_high_pass_3db_frequency*
- */sys/.../in_magn_filter_high_pass_3db_frequency*
- */sys/bus/iio/devices/iio:deviceX/out_voltageY_raw*
- */sys/bus/iio/devices/iio:deviceX/out_altvoltageY_raw*
- */sys/bus/iio/devices/iio:deviceX/out_voltageY&Z_raw*
- */sys/bus/iio/devices/iio:deviceX/out_altvoltageY&Z_raw*
- */sys/bus/iio/devices/iio:deviceX/out_voltageY_powerdown_mode*
- */sys/bus/iio/devices/iio:deviceX/out_voltage_powerdown_mode*
- */sys/bus/iio/devices/iio:deviceX/out_altvoltageY_powerdown_mode*
- */sys/bus/iio/devices/iio:deviceX/out_altvoltage_powerdown_mode*
- */sys/.../iio:deviceX/out_voltageY_powerdown_mode_available*
- */sys/.../iio:deviceX/out_voltage_powerdown_mode_available*
- */sys/.../iio:deviceX/out_altvoltageY_powerdown_mode_available*
- */sys/.../iio:deviceX/out_altvoltage_powerdown_mode_available*
- */sys/bus/iio/devices/iio:deviceX/out_voltageY_powerdown*
- */sys/bus/iio/devices/iio:deviceX/out_voltage_powerdown*
- */sys/bus/iio/devices/iio:deviceX/out_altvoltageY_powerdown*
- */sys/bus/iio/devices/iio:deviceX/out_altvoltage_powerdown*
- */sys/bus/iio/devices/iio:deviceX/out_altvoltageY_frequency*
- */sys/bus/iio/devices/iio:deviceX/out_altvoltageY_phase*
- */sys/bus/iio/devices/iio:deviceX/out_currentY_raw*
- */sys/bus/iio/devices/iio:deviceX/events*

- */sys/.../iio:deviceX/events/in_accel_x_thresh_rising_en*
- */sys/.../iio:deviceX/events/in_accel_x_thresh_falling_en*
- */sys/.../iio:deviceX/events/in_accel_y_thresh_rising_en*
- */sys/.../iio:deviceX/events/in_accel_y_thresh_falling_en*
- */sys/.../iio:deviceX/events/in_accel_z_thresh_rising_en*
- */sys/.../iio:deviceX/events/in_accel_z_thresh_falling_en*
- */sys/.../iio:deviceX/events/in_anglvel_x_thresh_rising_en*
- */sys/.../iio:deviceX/events/in_anglvel_x_thresh_falling_en*
- */sys/.../iio:deviceX/events/in_anglvel_y_thresh_rising_en*
- */sys/.../iio:deviceX/events/in_anglvel_y_thresh_falling_en*
- */sys/.../iio:deviceX/events/in_anglvel_z_thresh_rising_en*
- */sys/.../iio:deviceX/events/in_anglvel_z_thresh_falling_en*
- */sys/.../iio:deviceX/events/in_magn_x_thresh_rising_en*
- */sys/.../iio:deviceX/events/in_magn_x_thresh_falling_en*
- */sys/.../iio:deviceX/events/in_magn_y_thresh_rising_en*
- */sys/.../iio:deviceX/events/in_magn_y_thresh_falling_en*
- */sys/.../iio:deviceX/events/in_magn_z_thresh_rising_en*
- */sys/.../iio:deviceX/events/in_magn_z_thresh_falling_en*
- */sys/.../iio:deviceX/events/in_rot_from_north_magnetic_thresh_rising_en*
- */sys/.../iio:deviceX/events/in_rot_from_north_magnetic_thresh_falling_en*
- */sys/.../iio:deviceX/events/in_rot_from_north_true_thresh_rising_en*
- */sys/.../iio:deviceX/events/in_rot_from_north_true_thresh_falling_en*
- */sys/.../iio:deviceX/events/in_rot_from_north_magnetic_tilt_comp_thresh_rising_en*
- */sys/.../iio:deviceX/events/in_rot_from_north_magnetic_tilt_comp_thresh_falling_en*
- */sys/.../iio:deviceX/events/in_rot_from_north_true_tilt_comp_thresh_rising_en*
- */sys/.../iio:deviceX/events/in_rot_from_north_true_tilt_comp_thresh_falling_en*
- */sys/.../iio:deviceX/events/in_voltageY_supply_thresh_rising_en*
- */sys/.../iio:deviceX/events/in_voltageY_supply_thresh_falling_en*
- */sys/.../iio:deviceX/events/in_voltageY_thresh_rising_en*
- */sys/.../iio:deviceX/events/in_voltageY_thresh_falling_en*
- */sys/.../iio:deviceX/events/in_voltageY_thresh_either_en*
- */sys/.../iio:deviceX/events/in_tempY_thresh_rising_en*
- */sys/.../iio:deviceX/events/in_tempY_thresh_falling_en*
- */sys/.../iio:deviceX/events/in_capacitanceY_thresh_rising_en*

- */sys/.../iio:deviceX/events/in_capacitanceY_thresh_falling_en*
- */sys/.../iio:deviceX/events/in_accel_x_roc_rising_en*
- */sys/.../iio:deviceX/events/in_accel_x_roc_falling_en*
- */sys/.../iio:deviceX/events/in_accel_y_roc_rising_en*
- */sys/.../iio:deviceX/events/in_accel_y_roc_falling_en*
- */sys/.../iio:deviceX/events/in_accel_z_roc_rising_en*
- */sys/.../iio:deviceX/events/in_accel_z_roc_falling_en*
- */sys/.../iio:deviceX/events/in_anglvel_x_roc_rising_en*
- */sys/.../iio:deviceX/events/in_anglvel_x_roc_falling_en*
- */sys/.../iio:deviceX/events/in_anglvel_y_roc_rising_en*
- */sys/.../iio:deviceX/events/in_anglvel_y_roc_falling_en*
- */sys/.../iio:deviceX/events/in_anglvel_z_roc_rising_en*
- */sys/.../iio:deviceX/events/in_anglvel_z_roc_falling_en*
- */sys/.../iio:deviceX/events/in_magn_x_roc_rising_en*
- */sys/.../iio:deviceX/events/in_magn_x_roc_falling_en*
- */sys/.../iio:deviceX/events/in_magn_y_roc_rising_en*
- */sys/.../iio:deviceX/events/in_magn_y_roc_falling_en*
- */sys/.../iio:deviceX/events/in_magn_z_roc_rising_en*
- */sys/.../iio:deviceX/events/in_magn_z_roc_falling_en*
- */sys/.../iio:deviceX/events/in_rot_from_north_magnetic_roc_rising_en*
- */sys/.../iio:deviceX/events/in_rot_from_north_magnetic_roc_falling_en*
- */sys/.../iio:deviceX/events/in_rot_from_north_true_roc_rising_en*
- */sys/.../iio:deviceX/events/in_rot_from_north_true_roc_falling_en*
- */sys/.../iio:deviceX/events/in_rot_from_north_magnetic_tilt_comp_roc_rising_en*
- */sys/.../iio:deviceX/events/in_rot_from_north_magnetic_tilt_comp_roc_falling_en*
- */sys/.../iio:deviceX/events/in_rot_from_north_true_tilt_comp_roc_rising_en*
- */sys/.../iio:deviceX/events/in_rot_from_north_true_tilt_comp_roc_falling_en*
- */sys/.../iio:deviceX/events/in_voltageY_supply_roc_rising_en*
- */sys/.../iio:deviceX/events/in_voltageY_supply_roc_falling_en*
- */sys/.../iio:deviceX/events/in_voltageY_roc_rising_en*
- */sys/.../iio:deviceX/events/in_voltageY_roc_falling_en*
- */sys/.../iio:deviceX/events/in_tempY_roc_rising_en*
- */sys/.../iio:deviceX/events/in_tempY_roc_falling_en*
- */sys/.../events/in_capacitanceY_adaptive_thresh_rising_en*

- */sys/.../events/in_capacitanceY_adaptive_thresh_falling_en*
- */sys/.../in_capacitanceY_adaptive_thresh_rising_timeout*
- */sys/.../in_capacitanceY_adaptive_thresh_falling_timeout*
- */sys/.../events/in_accel_thresh_rising_value*
- */sys/.../events/in_accel_thresh_falling_value*
- */sys/.../events/in_accel_x_raw_thresh_rising_value*
- */sys/.../events/in_accel_x_raw_thresh_falling_value*
- */sys/.../events/in_accel_y_raw_thresh_rising_value*
- */sys/.../events/in_accel_y_raw_thresh_falling_value*
- */sys/.../events/in_accel_z_raw_thresh_rising_value*
- */sys/.../events/in_accel_z_raw_thresh_falling_value*
- */sys/.../events/in_anglvel_x_raw_thresh_rising_value*
- */sys/.../events/in_anglvel_x_raw_thresh_falling_value*
- */sys/.../events/in_anglvel_y_raw_thresh_rising_value*
- */sys/.../events/in_anglvel_y_raw_thresh_falling_value*
- */sys/.../events/in_anglvel_z_raw_thresh_rising_value*
- */sys/.../events/in_anglvel_z_raw_thresh_falling_value*
- */sys/.../events/in_magn_x_raw_thresh_rising_value*
- */sys/.../events/in_magn_x_raw_thresh_falling_value*
- */sys/.../events/in_magn_y_raw_thresh_rising_value*
- */sys/.../events/in_magn_y_raw_thresh_falling_value*
- */sys/.../events/in_magn_z_raw_thresh_rising_value*
- */sys/.../events/in_magn_z_raw_thresh_falling_value*
- */sys/.../events/in_rot_from_north_magnetic_raw_thresh_rising_value*
- */sys/.../events/in_rot_from_north_magnetic_raw_thresh_falling_value*
- */sys/.../events/in_rot_from_north_true_raw_thresh_rising_value*
- */sys/.../events/in_rot_from_north_true_raw_thresh_falling_value*
- */sys/.../events/in_rot_from_north_magnetic_tilt_comp_raw_thresh_rising_value*
- */sys/.../events/in_rot_from_north_magnetic_tilt_comp_raw_thresh_falling_value*
- */sys/.../events/in_rot_from_north_true_tilt_comp_raw_thresh_rising_value*
- */sys/.../events/in_rot_from_north_true_tilt_comp_raw_thresh_falling_value*
- */sys/.../events/in_voltageY_supply_raw_thresh_rising_value*
- */sys/.../events/in_voltageY_supply_raw_thresh_falling_value*
- */sys/.../events/in_voltageY_raw_thresh_rising_value*

- */sys/.../events/in_voltageY_raw_thresh_falling_value*
- */sys/.../events/in_tempY_raw_thresh_rising_value*
- */sys/.../events/in_tempY_raw_thresh_falling_value*
- */sys/.../events/in_illuminance0_thresh_falling_value*
- */sys/.../events/in_illuminance0_thresh_rising_value*
- */sys/.../events/in_proximity0_thresh_falling_value*
- */sys/.../events/in_proximity0_thresh_rising_value*
- */sys/.../events/in_illuminance_thresh_rising_value*
- */sys/.../events/in_illuminance_thresh_falling_value*
- */sys/.../events/in_capacitanceY_thresh_rising_value*
- */sys/.../events/in_capacitanceY_thresh_falling_value*
- */sys/.../events/in_capacitanceY_thresh_adaptive_rising_value*
- */sys/.../events/in_capacitanceY_thresh_falling_rising_value*
- */sys/.../events/in_accel_scale*
- */sys/.../events/in_accel_peak_scale*
- */sys/.../events/in_anglvel_scale*
- */sys/.../events/in_magn_scale*
- */sys/.../events/in_rot_from_north_magnetic_scale*
- */sys/.../events/in_rot_from_north_true_scale*
- */sys/.../events/in_voltage_scale*
- */sys/.../events/in_voltage_supply_scale*
- */sys/.../events/in_temp_scale*
- */sys/.../events/in_illuminance_scale*
- */sys/.../events/in_proximity_scale*
- */sys/.../events/in_accel_x_thresh_rising_hysteresis*
- */sys/.../events/in_accel_x_thresh_falling_hysteresis*
- */sys/.../events/in_accel_x_thresh_either_hysteresis*
- */sys/.../events/in_accel_y_thresh_rising_hysteresis*
- */sys/.../events/in_accel_y_thresh_falling_hysteresis*
- */sys/.../events/in_accel_y_thresh_either_hysteresis*
- */sys/.../events/in_accel_z_thresh_rising_hysteresis*
- */sys/.../events/in_accel_z_thresh_falling_hysteresis*
- */sys/.../events/in_accel_z_thresh_either_hysteresis*
- */sys/.../events/in_anglvel_x_thresh_rising_hysteresis*

- */sys/.../events/in_anglvel_x_thresh_falling_hysteresis*
- */sys/.../events/in_anglvel_x_thresh_either_hysteresis*
- */sys/.../events/in_anglvel_y_thresh_rising_hysteresis*
- */sys/.../events/in_anglvel_y_thresh_falling_hysteresis*
- */sys/.../events/in_anglvel_y_thresh_either_hysteresis*
- */sys/.../events/in_anglvel_z_thresh_rising_hysteresis*
- */sys/.../events/in_anglvel_z_thresh_falling_hysteresis*
- */sys/.../events/in_anglvel_z_thresh_either_hysteresis*
- */sys/.../events/in_magn_x_thresh_rising_hysteresis*
- */sys/.../events/in_magn_x_thresh_falling_hysteresis*
- */sys/.../events/in_magn_x_thresh_either_hysteresis*
- */sys/.../events/in_magn_y_thresh_rising_hysteresis*
- */sys/.../events/in_magn_y_thresh_falling_hysteresis*
- */sys/.../events/in_magn_y_thresh_either_hysteresis*
- */sys/.../events/in_magn_z_thresh_rising_hysteresis*
- */sys/.../events/in_magn_z_thresh_falling_hysteresis*
- */sys/.../events/in_magn_z_thresh_either_hysteresis*
- */sys/.../events/in_rot_from_north_magnetic_thresh_rising_hysteresis*
- */sys/.../events/in_rot_from_north_magnetic_thresh_falling_hysteresis*
- */sys/.../events/in_rot_from_north_magnetic_thresh_either_hysteresis*
- */sys/.../events/in_rot_from_north_true_thresh_rising_hysteresis*
- */sys/.../events/in_rot_from_north_true_thresh_falling_hysteresis*
- */sys/.../events/in_rot_from_north_true_thresh_either_hysteresis*
- */sys/.../events/in_rot_from_north_magnetic_tilt_comp_thresh_rising_hysteresis*
- */sys/.../events/in_rot_from_north_magnetic_tilt_comp_thresh_falling_hysteresis*
- */sys/.../events/in_rot_from_north_magnetic_tilt_comp_thresh_either_hysteresis*
- */sys/.../events/in_rot_from_north_true_tilt_comp_thresh_rising_hysteresis*
- */sys/.../events/in_rot_from_north_true_tilt_comp_thresh_falling_hysteresis*
- */sys/.../events/in_rot_from_north_true_tilt_comp_thresh_either_hysteresis*
- */sys/.../events/in_voltageY_thresh_rising_hysteresis*
- */sys/.../events/in_voltageY_thresh_falling_hysteresis*
- */sys/.../events/in_voltageY_thresh_either_hysteresis*
- */sys/.../events/in_tempY_thresh_rising_hysteresis*
- */sys/.../events/in_tempY_thresh_falling_hysteresis*

- */sys/.../events/in_tempY_thresh_either_hysteresis*
- */sys/.../events/in_illuminance0_thresh_falling_hysteresis*
- */sys/.../events/in_illuminance0_thresh_rising_hysteresis*
- */sys/.../events/in_illuminance0_thresh_either_hysteresis*
- */sys/.../events/in_proximity0_thresh_falling_hysteresis*
- */sys/.../events/in_proximity0_thresh_rising_hysteresis*
- */sys/.../events/in_proximity0_thresh_either_hysteresis*
- */sys/.../events/in_accel_x_raw_roc_rising_value*
- */sys/.../events/in_accel_x_raw_roc_falling_value*
- */sys/.../events/in_accel_y_raw_roc_rising_value*
- */sys/.../events/in_accel_y_raw_roc_falling_value*
- */sys/.../events/in_accel_z_raw_roc_rising_value*
- */sys/.../events/in_accel_z_raw_roc_falling_value*
- */sys/.../events/in_anglvel_x_raw_roc_rising_value*
- */sys/.../events/in_anglvel_x_raw_roc_falling_value*
- */sys/.../events/in_anglvel_y_raw_roc_rising_value*
- */sys/.../events/in_anglvel_y_raw_roc_falling_value*
- */sys/.../events/in_anglvel_z_raw_roc_rising_value*
- */sys/.../events/in_anglvel_z_raw_roc_falling_value*
- */sys/.../events/in_magn_x_raw_roc_rising_value*
- */sys/.../events/in_magn_x_raw_roc_falling_value*
- */sys/.../events/in_magn_y_raw_roc_rising_value*
- */sys/.../events/in_magn_y_raw_roc_falling_value*
- */sys/.../events/in_magn_z_raw_roc_rising_value*
- */sys/.../events/in_magn_z_raw_roc_falling_value*
- */sys/.../events/in_rot_from_north_magnetic_raw_roc_rising_value*
- */sys/.../events/in_rot_from_north_magnetic_raw_roc_falling_value*
- */sys/.../events/in_rot_from_north_true_raw_roc_rising_value*
- */sys/.../events/in_rot_from_north_true_raw_roc_falling_value*
- */sys/.../events/in_rot_from_north_magnetic_tilt_comp_raw_roc_rising_value*
- */sys/.../events/in_rot_from_north_magnetic_tilt_comp_raw_roc_falling_value*
- */sys/.../events/in_rot_from_north_true_tilt_comp_raw_roc_rising_value*
- */sys/.../events/in_rot_from_north_true_tilt_comp_raw_roc_falling_value*
- */sys/.../events/in_voltageY_supply_raw_roc_rising_value*

- */sys/.../events/in_voltageY_supply_raw_roc_falling_value*
- */sys/.../events/in_voltageY_raw_roc_rising_value*
- */sys/.../events/in_voltageY_raw_roc_falling_value*
- */sys/.../events/in_tempY_raw_roc_rising_value*
- */sys/.../events/in_tempY_raw_roc_falling_value*
- */sys/.../events/in_accel_x_thresh_rising_period*
- */sys/.../events/in_accel_x_thresh_falling_period*
- */sys/.../events/in_accel_x_roc_rising_period*
- */sys/.../events/in_accel_x_roc_falling_period*
- */sys/.../events/in_accel_y_thresh_rising_period*
- */sys/.../events/in_accel_y_thresh_falling_period*
- */sys/.../events/in_accel_y_roc_rising_period*
- */sys/.../events/in_accel_y_roc_falling_period*
- */sys/.../events/in_accel_z_thresh_rising_period*
- */sys/.../events/in_accel_z_thresh_falling_period*
- */sys/.../events/in_accel_z_roc_rising_period*
- */sys/.../events/in_accel_z_roc_falling_period*
- */sys/.../events/in_anglvel_x_thresh_rising_period*
- */sys/.../events/in_anglvel_x_thresh_falling_period*
- */sys/.../events/in_anglvel_x_roc_rising_period*
- */sys/.../events/in_anglvel_x_roc_falling_period*
- */sys/.../events/in_anglvel_y_thresh_rising_period*
- */sys/.../events/in_anglvel_y_thresh_falling_period*
- */sys/.../events/in_anglvel_y_roc_rising_period*
- */sys/.../events/in_anglvel_y_roc_falling_period*
- */sys/.../events/in_anglvel_z_thresh_rising_period*
- */sys/.../events/in_anglvel_z_thresh_falling_period*
- */sys/.../events/in_anglvel_z_roc_rising_period*
- */sys/.../events/in_anglvel_z_roc_falling_period*
- */sys/.../events/in_magn_x_thresh_rising_period*
- */sys/.../events/in_magn_x_thresh_falling_period*
- */sys/.../events/in_magn_x_roc_rising_period*
- */sys/.../events/in_magn_x_roc_falling_period*
- */sys/.../events/in_magn_y_thresh_rising_period*

- */sys/.../events/in_magn_y_thresh_falling_period*
- */sys/.../events/in_magn_y_roc_rising_period*
- */sys/.../events/in_magn_y_roc_falling_period*
- */sys/.../events/in_magn_z_thresh_rising_period*
- */sys/.../events/in_magn_z_thresh_falling_period*
- */sys/.../events/in_magn_z_roc_rising_period*
- */sys/.../events/in_magn_z_roc_falling_period*
- */sys/.../events/in_rot_from_north_magnetic_thresh_rising_period*
- */sys/.../events/in_rot_from_north_magnetic_thresh_falling_period*
- */sys/.../events/in_rot_from_north_magnetic_roc_rising_period*
- */sys/.../events/in_rot_from_north_magnetic_roc_falling_period*
- */sys/.../events/in_rot_from_north_true_thresh_rising_period*
- */sys/.../events/in_rot_from_north_true_thresh_falling_period*
- */sys/.../events/in_rot_from_north_true_roc_rising_period*
- */sys/.../events/in_rot_from_north_true_roc_falling_period*
- */sys/.../events/in_rot_from_north_magnetic_tilt_comp_thresh_rising_period*
- */sys/.../events/in_rot_from_north_magnetic_tilt_comp_thresh_falling_period*
- */sys/.../events/in_rot_from_north_magnetic_tilt_comp_roc_rising_period*
- */sys/.../events/in_rot_from_north_magnetic_tilt_comp_roc_falling_period*
- */sys/.../events/in_rot_from_north_true_tilt_comp_thresh_rising_period*
- */sys/.../events/in_rot_from_north_true_tilt_comp_thresh_falling_period*
- */sys/.../events/in_rot_from_north_true_tilt_comp_roc_rising_period*
- */sys/.../events/in_rot_from_north_true_tilt_comp_roc_falling_period*
- */sys/.../events/in_voltageY_supply_thresh_rising_period*
- */sys/.../events/in_voltageY_supply_thresh_falling_period*
- */sys/.../events/in_voltageY_supply_roc_rising_period*
- */sys/.../events/in_voltageY_supply_roc_falling_period*
- */sys/.../events/in_voltageY_thresh_rising_period*
- */sys/.../events/in_voltageY_thresh_falling_period*
- */sys/.../events/in_voltageY_roc_rising_period*
- */sys/.../events/in_voltageY_roc_falling_period*
- */sys/.../events/in_tempY_thresh_rising_period*
- */sys/.../events/in_tempY_thresh_falling_period*
- */sys/.../events/in_tempY_roc_rising_period*

- */sys/.../events/in_tempY_roc_falling_period*
- */sys/.../events/in_accel_x&y&z_mag_falling_period*
- */sys/.../events/in_intensity0_thresh_period*
- */sys/.../events/in_proximity0_thresh_period*
- */sys/.../events/in_activity_still_thresh_rising_period*
- */sys/.../events/in_activity_still_thresh_falling_period*
- */sys/.../events/in_activity_walking_thresh_rising_period*
- */sys/.../events/in_activity_walking_thresh_falling_period*
- */sys/.../events/in_activity_jogging_thresh_rising_period*
- */sys/.../events/in_activity_jogging_thresh_falling_period*
- */sys/.../events/in_activity_running_thresh_rising_period*
- */sys/.../events/in_activity_running_thresh_falling_period*
- */sys/.../events/in_illuminance_thresh_either_period*
- */sys/.../events/in_accel_thresh_rising_low_pass_filter_3db*
- */sys/.../events/in_anglvel_thresh_rising_low_pass_filter_3db*
- */sys/.../events/in_magn_thresh_rising_low_pass_filter_3db*
- */sys/.../events/in_accel_thresh_rising_high_pass_filter_3db*
- */sys/.../events/in_anglvel_thresh_rising_high_pass_filter_3db*
- */sys/.../events/in_magn_thresh_rising_high_pass_filter_3db*
- */sys/.../events/in_activity_still_thresh_rising_en*
- */sys/.../events/in_activity_still_thresh_falling_en*
- */sys/.../events/in_activity_walking_thresh_rising_en*
- */sys/.../events/in_activity_walking_thresh_falling_en*
- */sys/.../events/in_activity_jogging_thresh_rising_en*
- */sys/.../events/in_activity_jogging_thresh_falling_en*
- */sys/.../events/in_activity_running_thresh_rising_en*
- */sys/.../events/in_activity_running_thresh_falling_en*
- */sys/.../events/in_activity_still_thresh_rising_value*
- */sys/.../events/in_activity_still_thresh_falling_value*
- */sys/.../events/in_activity_walking_thresh_rising_value*
- */sys/.../events/in_activity_walking_thresh_falling_value*
- */sys/.../events/in_activity_jogging_thresh_rising_value*
- */sys/.../events/in_activity_jogging_thresh_falling_value*
- */sys/.../events/in_activity_running_thresh_rising_value*

- */sys/.../events/in_activity_running_thresh_falling_value*
- */sys/.../iio:deviceX/events/in_accel_mag_en*
- */sys/.../iio:deviceX/events/in_accel_mag_rising_en*
- */sys/.../iio:deviceX/events/in_accel_mag_falling_en*
- */sys/.../iio:deviceX/events/in_accel_x_mag_en*
- */sys/.../iio:deviceX/events/in_accel_x_mag_rising_en*
- */sys/.../iio:deviceX/events/in_accel_x_mag_falling_en*
- */sys/.../iio:deviceX/events/in_accel_y_mag_en*
- */sys/.../iio:deviceX/events/in_accel_y_mag_rising_en*
- */sys/.../iio:deviceX/events/in_accel_y_mag_falling_en*
- */sys/.../iio:deviceX/events/in_accel_z_mag_en*
- */sys/.../iio:deviceX/events/in_accel_z_mag_rising_en*
- */sys/.../iio:deviceX/events/in_accel_z_mag_falling_en*
- */sys/.../iio:deviceX/events/in_accel_x&y&z_mag_rising_en*
- */sys/.../iio:deviceX/events/in_accel_x&y&z_mag_falling_en*
- */sys/.../events/in_accel_raw_mag_value*
- */sys/.../events/in_accel_x_raw_mag_rising_value*
- */sys/.../events/in_accel_y_raw_mag_rising_value*
- */sys/.../events/in_accel_z_raw_mag_rising_value*
- */sys/.../iio:deviceX/events/in_accel_mag_referenced_en*
- */sys/.../iio:deviceX/events/in_accel_mag_referenced_rising_en*
- */sys/.../iio:deviceX/events/in_accel_mag_referenced_falling_en*
- */sys/.../iio:deviceX/events/in_accel_y_mag_referenced_en*
- */sys/.../iio:deviceX/events/in_accel_y_mag_referenced_rising_en*
- */sys/.../iio:deviceX/events/in_accel_y_mag_referenced_falling_en*
- */sys/.../iio:deviceX/events/in_accel_mag_referenced_value*
- */sys/.../iio:deviceX/events/in_accel_mag_referenced_rising_value*
- */sys/.../iio:deviceX/events/in_accel_mag_referenced_falling_value*
- */sys/.../iio:deviceX/events/in_accel_y_mag_referenced_value*
- */sys/.../iio:deviceX/events/in_accel_y_mag_referenced_rising_value*
- */sys/.../iio:deviceX/events/in_accel_y_mag_referenced_falling_value*
- */sys/.../events/in_steps_change_en*
- */sys/.../events/in_steps_change_value*
- */sys/bus/iio/devices/iio:deviceX/trigger/current_trigger*

- */sys/bus/iio/devices/iio:deviceX/bufferY/length*
- */sys/bus/iio/devices/iio:deviceX/bufferY/enable*
- */sys/bus/iio/devices/iio:deviceX/bufferY*
- */sys/.../iio:deviceX/bufferY/in_accel_x_en*
- */sys/.../iio:deviceX/bufferY/in_accel_y_en*
- */sys/.../iio:deviceX/bufferY/in_accel_z_en*
- */sys/.../iio:deviceX/bufferY/in_deltaangl_x_en*
- */sys/.../iio:deviceX/bufferY/in_deltaangl_y_en*
- */sys/.../iio:deviceX/bufferY/in_deltaangl_z_en*
- */sys/.../iio:deviceX/bufferY/in_deltavelocity_x_en*
- */sys/.../iio:deviceX/bufferY/in_deltavelocity_y_en*
- */sys/.../iio:deviceX/bufferY/in_deltavelocity_z_en*
- */sys/.../iio:deviceX/bufferY/in_anglvel_x_en*
- */sys/.../iio:deviceX/bufferY/in_anglvel_y_en*
- */sys/.../iio:deviceX/bufferY/in_anglvel_z_en*
- */sys/.../iio:deviceX/bufferY/in_magn_x_en*
- */sys/.../iio:deviceX/bufferY/in_magn_y_en*
- */sys/.../iio:deviceX/bufferY/in_magn_z_en*
- */sys/.../iio:deviceX/bufferY/in_rot_from_north_magnetic_en*
- */sys/.../iio:deviceX/bufferY/in_rot_from_north_true_en*
- */sys/.../iio:deviceX/bufferY/in_rot_from_north_magnetic_tilt_comp_en*
- */sys/.../iio:deviceX/bufferY/in_rot_from_north_true_tilt_comp_en*
- */sys/.../iio:deviceX/bufferY/in_timestamp_en*
- */sys/.../iio:deviceX/bufferY/in_voltageY_supply_en*
- */sys/.../iio:deviceX/bufferY/in_voltageY_en*
- */sys/.../iio:deviceX/bufferY/in_voltageY-voltageZ_en*
- */sys/.../iio:deviceX/bufferY/in_voltageY_i_en*
- */sys/.../iio:deviceX/bufferY/in_voltageY_q_en*
- */sys/.../iio:deviceX/bufferY/in_voltage_i_en*
- */sys/.../iio:deviceX/bufferY/in_voltage_q_en*
- */sys/.../iio:deviceX/bufferY/in_incli_x_en*
- */sys/.../iio:deviceX/bufferY/in_incli_y_en*
- */sys/.../iio:deviceX/bufferY/in_pressureY_en*
- */sys/.../iio:deviceX/bufferY/in_pressure_en*

- */sys/.../iio:deviceX/bufferY/in_rot_quaternion_en*
- */sys/.../iio:deviceX/bufferY/in_proximity_en*
- */sys/.../iio:deviceX/bufferY/in_accel_type*
- */sys/.../iio:deviceX/bufferY/in_deltaangl_type*
- */sys/.../iio:deviceX/bufferY/in_deltavelocity_type*
- */sys/.../iio:deviceX/bufferY/in_anglvel_type*
- */sys/.../iio:deviceX/bufferY/in_magn_type*
- */sys/.../iio:deviceX/bufferY/in_incli_type*
- */sys/.../iio:deviceX/bufferY/in_voltageY_type*
- */sys/.../iio:deviceX/bufferY/in_voltage_type*
- */sys/.../iio:deviceX/bufferY/in_voltageY_supply_type*
- */sys/.../iio:deviceX/bufferY/in_voltageY_i_type*
- */sys/.../iio:deviceX/bufferY/in_voltageY_q_type*
- */sys/.../iio:deviceX/bufferY/in_voltage_i_type*
- */sys/.../iio:deviceX/bufferY/in_voltage_q_type*
- */sys/.../iio:deviceX/bufferY/in_timestamp_type*
- */sys/.../iio:deviceX/bufferY/in_pressureY_type*
- */sys/.../iio:deviceX/bufferY/in_pressure_type*
- */sys/.../iio:deviceX/bufferY/in_rot_quaternion_type*
- */sys/.../iio:deviceX/bufferY/in_proximity_type*
- */sys/.../iio:deviceX/scan_elements/in_accel_type_available*
- */sys/.../iio:deviceX/bufferY/in_voltageY_index*
- */sys/.../iio:deviceX/bufferY/in_voltageY_supply_index*
- */sys/.../iio:deviceX/bufferY/in_voltageY_i_index*
- */sys/.../iio:deviceX/bufferY/in_voltageY_q_index*
- */sys/.../iio:deviceX/bufferY/in_voltage_i_index*
- */sys/.../iio:deviceX/bufferY/in_voltage_q_index*
- */sys/.../iio:deviceX/bufferY/in_accel_x_index*
- */sys/.../iio:deviceX/bufferY/in_accel_y_index*
- */sys/.../iio:deviceX/bufferY/in_accel_z_index*
- */sys/.../iio:deviceX/bufferY/in_deltaangl_x_index*
- */sys/.../iio:deviceX/bufferY/in_deltaangl_y_index*
- */sys/.../iio:deviceX/bufferY/in_deltaangl_z_index*
- */sys/.../iio:deviceX/bufferY/in_deltavelocity_x_index*

- */sys/.../iio:deviceX/bufferY/in_deltavelocity_y_index*
- */sys/.../iio:deviceX/bufferY/in_deltavelocity_z_index*
- */sys/.../iio:deviceX/bufferY/in_anglvel_x_index*
- */sys/.../iio:deviceX/bufferY/in_anglvel_y_index*
- */sys/.../iio:deviceX/bufferY/in_anglvel_z_index*
- */sys/.../iio:deviceX/bufferY/in_magn_x_index*
- */sys/.../iio:deviceX/bufferY/in_magn_y_index*
- */sys/.../iio:deviceX/bufferY/in_magn_z_index*
- */sys/.../iio:deviceX/bufferY/in_rot_from_north_magnetic_index*
- */sys/.../iio:deviceX/bufferY/in_rot_from_north_true_index*
- */sys/.../iio:deviceX/bufferY/in_rot_from_north_magnetic_tilt_comp_index*
- */sys/.../iio:deviceX/bufferY/in_rot_from_north_true_tilt_comp_index*
- */sys/.../iio:deviceX/bufferY/in_incli_x_index*
- */sys/.../iio:deviceX/bufferY/in_incli_y_index*
- */sys/.../iio:deviceX/bufferY/in_timestamp_index*
- */sys/.../iio:deviceX/bufferY/in_pressureY_index*
- */sys/.../iio:deviceX/bufferY/in_pressure_index*
- */sys/.../iio:deviceX/bufferY/in_rot_quaternion_index*
- */sys/.../iio:deviceX/bufferY/in_proximity_index*
- */sys/.../iio:deviceX/in_activity_still_input*
- */sys/.../iio:deviceX/in_activity_walking_input*
- */sys/.../iio:deviceX/in_activity_jogging_input*
- */sys/.../iio:deviceX/in_activity_running_input*
- */sys/.../iio:deviceX/in_anglvel_z_quadrature_correction_raw*
- */sys/.../iio:deviceX/in_accelX_power_mode*
- */sys/.../iio:deviceX/in_energy_input*
- */sys/.../iio:deviceX/in_energy_raw*
- */sys/.../iio:deviceX/in_distance_input*
- */sys/.../iio:deviceX/in_distance_raw*
- */sys/bus/iio/devices/iio:deviceX/store_eeprom*
- */sys/.../iio:deviceX/in_proximity_raw*
- */sys/.../iio:deviceX/in_proximity_input*
- */sys/.../iio:deviceX/in_proximityY_raw*
- */sys/.../iio:deviceX/in_illuminance_input*

- /sys/.../iio:deviceX/in_illuminance_raw
- /sys/.../iio:deviceX/in_illuminanceY_input
- /sys/.../iio:deviceX/in_illuminanceY_raw
- /sys/.../iio:deviceX/in_illuminanceY_mean_raw
- /sys/.../iio:deviceX/in_illuminance_ir_raw
- /sys/.../iio:deviceX/in_illuminance_clear_raw
- /sys/.../iio:deviceX/in_intensityY_raw
- /sys/.../iio:deviceX/in_intensityY_ir_raw
- /sys/.../iio:deviceX/in_intensityY_both_raw
- /sys/.../iio:deviceX/in_intensityY_uv_raw
- /sys/.../iio:deviceX/in_intensityY_uva_raw
- /sys/.../iio:deviceX/in_intensityY_uvb_raw
- /sys/.../iio:deviceX/in_intensityY_duv_raw
- /sys/.../iio:deviceX/in_uvindex_input
- /sys/.../iio:deviceX/in_intensity_integration_time
- /sys/.../iio:deviceX/in_intensity_red_integration_time
- /sys/.../iio:deviceX/in_intensity_green_integration_time
- /sys/.../iio:deviceX/in_intensity_blue_integration_time
- /sys/.../iio:deviceX/in_intensity_clear_integration_time
- /sys/.../iio:deviceX/in_illuminance_integration_time
- /sys/.../iio:deviceX/in_velocity_sqrt(x^2+y^2+z^2)_integration_time
- /sys/bus/iio/devices/iio:deviceX/in_rot_quaternion_raw
- /sys/bus/iio/devices/iio:deviceX/in_rot_from_north_magnetic_tilt_comp_raw
- /sys/bus/iio/devices/iio:deviceX/in_rot_from_north_true_tilt_comp_raw
- /sys/bus/iio/devices/iio:deviceX/in_rot_from_north_magnetic_raw
- /sys/bus/iio/devices/iio:deviceX/in_rot_from_north_true_raw
- /sys/bus/iio/devices/iio:deviceX/in_currentX_raw
- /sys/bus/iio/devices/iio:deviceX/in_currentX_i_raw
- /sys/bus/iio/devices/iio:deviceX/in_currentX_q_raw
- /sys/.../iio:deviceX/in_energy_en
- /sys/.../iio:deviceX/in_distance_en
- /sys/.../iio:deviceX/in_velocity_sqrt(x^2+y^2+z^2)_en
- /sys/.../iio:deviceX/in_steps_en
- /sys/.../iio:deviceX/in_steps_input

- */sys/.../iio:deviceX/in_velocity_sqrt(x^2+y^2+z^2)_input*
- */sys/.../iio:deviceX/in_velocity_sqrt(x^2+y^2+z^2)_raw*
- */sys/.../iio:deviceX/in_steps_debounce_count*
- */sys/.../iio:deviceX/in_steps_debounce_time*
- */sys/bus/iio/devices/iio:deviceX/bufferY/watermark*
- */sys/bus/iio/devices/iio:deviceX/bufferY/data_available*
- */sys/bus/iio/devices/iio:deviceX/buffer/hwfifo_enabled*
- */sys/bus/iio/devices/iio:device*/buffer/hwfifo_timeout*
- */sys/bus/iio/devices/iio:deviceX/buffer/hwfifo_watermark*
- */sys/bus/iio/devices/iio:deviceX/buffer/hwfifo_watermark_min*
- */sys/bus/iio/devices/iio:deviceX/buffer/hwfifo_watermark_max*
- */sys/bus/iio/devices/iio:deviceX/buffer/hwfifo_watermark_available*
- */sys/bus/iio/devices/iio:deviceX/in_temp_calibemissivity*
- */sys/bus/iio/devices/iio:deviceX/in_tempX_calibemissivity*
- */sys/bus/iio/devices/iio:deviceX/in_temp_object_calibemissivity*
- */sys/bus/iio/devices/iio:deviceX/in_tempX_object_calibemissivity*
- */sys/bus/iio/devices/iio:deviceX/in_magn_x_oversampling_ratio*
- */sys/bus/iio/devices/iio:deviceX/in_magn_y_oversampling_ratio*
- */sys/bus/iio/devices/iio:deviceX/in_magn_z_oversampling_ratio*
- */sys/bus/iio/devices/iio:deviceX/in_concentration_raw*
- */sys/bus/iio/devices/iio:deviceX/in_concentrationX_raw*
- */sys/bus/iio/devices/iio:deviceX/in_concentration_co2_raw*
- */sys/bus/iio/devices/iio:deviceX/in_concentrationX_co2_raw*
- */sys/bus/iio/devices/iio:deviceX/in_concentration_ethanol_raw*
- */sys/bus/iio/devices/iio:deviceX/in_concentrationX_ethanol_raw*
- */sys/bus/iio/devices/iio:deviceX/in_concentration_h2_raw*
- */sys/bus/iio/devices/iio:deviceX/in_concentrationX_h2_raw*
- */sys/bus/iio/devices/iio:deviceX/in_concentration_o2_raw*
- */sys/bus/iio/devices/iio:deviceX/in_concentrationX_o2_raw*
- */sys/bus/iio/devices/iio:deviceX/in_concentration_voc_raw*
- */sys/bus/iio/devices/iio:deviceX/in_concentrationX_voc_raw*
- */sys/bus/iio/devices/iio:deviceX/in_resistance_raw*
- */sys/bus/iio/devices/iio:deviceX/in_resistanceX_raw*
- */sys/bus/iio/devices/iio:deviceX/out_resistance_raw*

- /sys/bus/iio/devices/iio:deviceX/out_resistanceX_raw
- /sys/bus/iio/devices/iio:deviceX/heater_enable
- /sys/bus/iio/devices/iio:deviceX/in_ph_raw
- /sys/bus/iio/devices/iio:deviceX/mount_matrix
- /sys/bus/iio/devices/iio:deviceX/in_mount_matrix
- /sys/bus/iio/devices/iio:deviceX/out_mount_matrix
- /sys/bus/iio/devices/iio:deviceX/in_anglvel_mount_matrix
- /sys/bus/iio/devices/iio:deviceX/in_accel_mount_matrix
- /sys/bus/iio/devices/iio:deviceX/in_electricalconductivity_raw
- /sys/bus/iio/devices/iio:deviceX/in_countY_raw
- /sys/bus/iio/devices/iio:deviceX/in_indexY_raw
- /sys/bus/iio/devices/iio:deviceX/in_count_count_direction_available
- /sys/bus/iio/devices/iio:deviceX/in_countY_count_direction
- /sys/bus/iio/devices/iio:deviceX/in_voltageY_label
- /sys/bus/iio/devices/iio:deviceX/out_voltageY_label
- /sys/bus/iio/devices/iio:deviceX/in_phaseY_raw
- /sys/bus/iio/devices/iio:deviceX/in_massconcentration_pm1_input
- /sys/bus/iio/devices/iio:deviceX/in_massconcentrationY_pm1_input
- /sys/bus/iio/devices/iio:deviceX/in_massconcentration_pm2p5_input
- /sys/bus/iio/devices/iio:deviceX/in_massconcentrationY_pm2p5_input
- /sys/bus/iio/devices/iio:deviceX/in_massconcentration_pm4_input
- /sys/bus/iio/devices/iio:deviceX/in_massconcentrationY_pm4_input
- /sys/bus/iio/devices/iio:deviceX/in_massconcentration_pm10_input
- /sys/bus/iio/devices/iio:deviceX/in_massconcentrationY_pm10_input
- /sys/bus/iio/devices/iio:deviceX/events/in_illuminance_period_available
- /sys/bus/iio/devices/iio:deviceX/in_filter_notch_center_frequency
- /sys/bus/iio/devices/iio:deviceX/in_temp_thermocouple_type
- /sys/bus/iio/devices/iio:deviceX/in_temp_object_calibambient
- /sys/bus/iio/devices/iio:deviceX/in_tempX_object_calibambient
- /sys/bus/iio/devices/iio:deviceX/in_intensity_x_raw
- /sys/bus/iio/devices/iio:deviceX/in_intensity_y_raw
- /sys/bus/iio/devices/iio:deviceX/in_intensity_z_raw
- /sys/bus/iio/devices/iio:deviceX/in_anglY_label
- /sys/bus/iio/devices/iio:deviceX/in_illuminance_hysteresis_relative

- /sys/bus/iio/devices/iio:deviceX/in_intensity_hysteresis_relative
- /sys/bus/iio/devices/iio:deviceX/calibration_auto_enable
- /sys/bus/iio/devices/iio:deviceX/calibration_forced_value
- /sys/bus/iio/devices/iio:deviceX/calibration_forced_value_available
- /sys/bus/iio/devices/iio:deviceX/in_voltageX_sampling_frequency
- /sys/bus/iio/devices/iio:deviceX/in_powerY_sampling_frequency
- /sys/bus/iio/devices/iio:deviceX/in_currentZ_sampling_frequency
- /sys/.../events/in_accel_gesture_singletap_en
- /sys/.../events/in_accel_gesture_doubletap_en
- /sys/.../events/in_accel_gesture_singletap_value
- /sys/.../events/in_accel_gesture_doubletap_value
- /sys/.../events/in_accel_gesture_tap_value_available
- /sys/.../events/in_accel_gesture_singletap_reset_timeout
- /sys/.../events/in_accel_gesture_doubletap_reset_timeout
- /sys/.../events/in_accel_gesture_tap_reset_timeout_available
- /sys/.../events/in_accel_gesture_doubletap_tap2_min_delay
- /sys/.../events/in_accel_gesture_doubletap_tap2_min_delay_available
- /sys/.../events/in_accel_gesture_tap_maxtomin_time
- /sys/.../events/in_accel_gesture_tap_maxtomin_time_available
- /sys/bus/iio/devices/iio:deviceX/in_rot_yaw_raw
- /sys/bus/iio/devices/iio:deviceX/in_rot_pitch_raw
- /sys/bus/iio/devices/iio:deviceX/in_rot_roll_raw
- /sys/bus/iio/devices/iio:deviceX/serialnumber
- /sys/.../events/in_proximity_thresh_either_runningperiod
- /sys/.../events/in_proximity_thresh_either_runningcount
- /sys/bus/iio/devices/iio:deviceX/in_colortemp_raw
- /sys/bus/iio/devices/iio:deviceX/in_chromaticity_x_raw
- /sys/bus/iio/devices/iio:deviceX/in_chromaticity_y_raw
- /sys/bus/iio/devices/iio:deviceX/events/in_altvoltageY_mag_either_label
- /sys/bus/iio/devices/iio:deviceX/events/in_altvoltageY_mag_rising_label
- /sys/bus/iio/devices/iio:deviceX/events/in_altvoltageY_thresh_falling_label
- /sys/bus/iio/devices/iio:deviceX/events/in_altvoltageY_thresh_rising_label
- /sys/bus/iio/devices/iio:deviceX/events/in_anglvelY_mag_rising_label
- /sys/bus/iio/devices/iio:deviceX/events/in_anglY_thresh_rising_label

- */sys/bus/iio/devices/iio:deviceX/events/in_phaseY_mag_rising_label*
- */sys/.../events/in_accel_gesture_tap_wait_timeout*
- */sys/.../events/in_accel_gesture_tap_wait_dur*
- */sys/.../events/in_accel_gesture_tap_wait_dur_available*

5.2.169 File testing/sysfs-bus-iio-accel-adxl372

Has the following ABI:

- */sys/bus/iio/devices/triggerX/name = "adxl372-devX-peak"*

5.2.170 File testing/sysfs-bus-iio-accel-bmc150

Has the following ABI:

- */sys/bus/iio/devices/triggerX/name = "bmc150_accel-any-motion-devX"*

5.2.171 File testing/sysfs-bus-iio-adc-ad4130

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/in_voltage-voltage_filter_mode_available*
- */sys/bus/iio/devices/iio:deviceX/in_voltageY-voltageZ_filter_mode*

5.2.172 File testing/sysfs-bus-iio-adc-ad7192

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/ac_excitation_en*
- */sys/bus/iio/devices/iio:deviceX/bridge_switch_en*
- */sys/bus/iio/devices/iio:deviceX/in_voltagex_sys_calibration*
- */sys/bus/iio/devices/iio:deviceX/in_voltage2-voltage2_shorted_raw*
- */sys/bus/iio/devices/iio:deviceX/in_voltagex_sys_calibration_mode_available*
- */sys/bus/iio/devices/iio:deviceX/in_voltagex_sys_calibration_mode*

5.2.173 File testing/sysfs-bus-iio-adc-ad7280a

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/in_voltageY-voltageZ_balance_switch_en*
- */sys/bus/iio/devices/iio:deviceX/in_voltageY-voltageZ_balance_switch_timer*

5.2.174 File testing/sysfs-bus-iio-adc-envelope-detector

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/in_altvoltageY_invert*
- */sys/bus/iio/devices/iio:deviceX/in_altvoltageY_compare_interval*

5.2.175 File testing/sysfs-bus-iio-adc-hi8435

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/in_voltageY_sensing_mode*
- */sys/bus/iio/devices/iio:deviceX/events/in_voltageY_thresh_falling_value*
- */sys/bus/iio/devices/iio:deviceX/events/in_voltageY_thresh_rising_value*

5.2.176 File testing/sysfs-bus-iio-adc-max11410

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/in_voltage_filterY_notch_en*
- */sys/bus/iio/devices/iio:deviceX/in_voltage_filterY_notch_center*

5.2.177 File testing/sysfs-bus-iio-adc-max9611

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/in_power_shunt_resistor*
- */sys/bus/iio/devices/iio:deviceX/in_current_shunt_resistor*

5.2.178 File testing/sysfs-bus-iio-adc-mcp3564

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/boost_current_gain*
- */sys/bus/iio/devices/iio:deviceX/boost_current_gain_available*
- */sys/bus/iio/devices/iio:deviceX/auto_zeroing_mux_enable*
- */sys/bus/iio/devices/iio:deviceX/auto_zeroing_ref_enable*

5.2.179 File testing/sysfs-bus-iio-adc-mt6360

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/in_voltage0_raw*
- */sys/bus/iio/devices/iio:deviceX/in_voltage1_raw*
- */sys/bus/iio/devices/iio:deviceX/in_voltage2_raw*
- */sys/bus/iio/devices/iio:deviceX/in_voltage3_raw*
- */sys/bus/iio/devices/iio:deviceX/in_voltage4_raw*
- */sys/bus/iio/devices/iio:deviceX/in_current5_raw*
- */sys/bus/iio/devices/iio:deviceX/in_current6_raw*
- */sys/bus/iio/devices/iio:deviceX/in_current7_raw*
- */sys/bus/iio/devices/iio:deviceX/in_temp8_raw*
- */sys/bus/iio/devices/iio:deviceX/in_voltage9_raw*
- */sys/bus/iio/devices/iio:deviceX/in_voltage10_raw*

5.2.180 File testing/sysfs-bus-iio-adc-stm32

Has the following ABI:

- */sys/bus/iio/devices/triggerX/trigger_polarity*
- */sys/bus/iio/devices/triggerX/trigger_polarity_available*

5.2.181 File testing/sysfs-bus-iio-bno055

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/in_accel_raw_range*
- */sys/bus/iio/devices/iio:deviceX/in_anglvel_raw_range*
- */sys/bus/iio/devices/iio:deviceX/in_accel_raw_range_available*
- */sys/bus/iio/devices/iio:deviceX/in_anglvel_raw_range_available*
- */sys/bus/iio/devices/iio:deviceX/in_magn_calibration_fast_enable*
- */sys/bus/iio/devices/iio:deviceX/fusion_enable*
- */sys/bus/iio/devices/iio:deviceX/calibration_data*
- */sys/bus/iio/devices/iio:deviceX/in_accel_calibration_auto_status*
- */sys/bus/iio/devices/iio:deviceX/in_gyro_calibration_auto_status*
- */sys/bus/iio/devices/iio:deviceX/in_magn_calibration_auto_status*
- */sys/bus/iio/devices/iio:deviceX/sys_calibration_auto_status*

5.2.182 File testing/sysfs-bus-iio-cdc-ad7746

Has the following ABI:

- */sys/.../iio:deviceX/in_capacitableY_calibbias_calibration*
- */sys/.../iio:deviceX/in_capacitableY_calibscale_calibration*

5.2.183 File testing/sysfs-bus-iio-chemical-sgp40

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/out_temp_raw*
- */sys/bus/iio/devices/iio:deviceX/out_humidityrelative_raw*
- */sys/bus/iio/devices/iio:deviceX/in_resistance_calibbias*

5.2.184 File testing/sysfs-bus-iio-chemical-sunrise-co2

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/in_concentration_co2_calibration_factory*
- */sys/bus/iio/devices/iio:deviceX/in_concentration_co2_calibration_background*
- */sys/bus/iio/devices/iio:deviceX/error_status_available*
- */sys/bus/iio/devices/iio:deviceX/error_status*

5.2.185 File testing/sysfs-bus-iio-chemical-vz89x

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/in_concentration_VOC_short_raw*

5.2.186 File testing/sysfs-bus-iio-cros-ec

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/calibrate*
- */sys/bus/iio/devices/iio:deviceX/location*
- */sys/bus/iio/devices/iio:deviceX/id*

5.2.187 File testing/sysfs-bus-iio-dac-ad5766

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/in_voltageY_dither_enable*
- */sys/bus/iio/devices/iio:deviceX/in_voltageY_dither_invert*
- */sys/bus/iio/devices/iio:deviceX/in_voltageY_dither_scale_available*
- */sys/bus/iio/devices/iio:deviceX/in_voltageY_dither_scale*
- */sys/bus/iio/devices/iio:deviceX/in_voltageY_dither_source*

5.2.188 File testing/sysfs-bus-iio-dac-dpot-dac

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/out_voltageY_raw_available*

5.2.189 File testing/sysfs-bus-iio-dac-ltc2688

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/out_voltageY_dither_en*
- */sys/bus/iio/devices/iio:deviceX/out_voltageY_dither_raw*
- */sys/bus/iio/devices/iio:deviceX/out_voltageY_dither_raw_available*
- */sys/bus/iio/devices/iio:deviceX/out_voltageY_dither_offset*
- */sys/bus/iio/devices/iio:deviceX/out_voltageY_dither_frequency*
- */sys/bus/iio/devices/iio:deviceX/out_voltageY_dither_frequency_available*
- */sys/bus/iio/devices/iio:deviceX/out_voltageY_dither_phase*
- */sys/bus/iio/devices/iio:deviceX/out_voltageY_dither_phase_available*
- */sys/bus/iio/devices/iio:deviceX/out_voltageY_toggle_en*
- */sys/bus/iio/devices/iio:deviceX/out_voltageY_raw0*
- */sys/bus/iio/devices/iio:deviceX/out_voltageY_raw1*
- */sys/bus/iio/devices/iio:deviceX/out_voltageY_symbol*

5.2.190 File testing/sysfs-bus-iio-dfsdm-adc-stm32

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/in_voltage_spi_clk_freq*

5.2.191 File testing/sysfs-bus-iio-distance-srf08

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/sensor_max_range*

5.2.192 File testing/sysfs-bus-iio-dma-buffer

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/buffer/length_align_bytes*

5.2.193 File testing/sysfs-bus-iio-filter-admv8818

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/filter_mode_available*
- */sys/bus/iio/devices/iio:deviceX/filter_mode*

5.2.194 File testing/sysfs-bus-iio-frequency-ad9523

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/pll2_feedback_clk_present*
- */sys/bus/iio/devices/iio:deviceX/pll2_reference_clk_present*
- */sys/bus/iio/devices/iio:deviceX/pll1_reference_clk_a_present*
- */sys/bus/iio/devices/iio:deviceX/pll1_reference_clk_b_present*
- */sys/bus/iio/devices/iio:deviceX/pll1_reference_clk_test_present*
- */sys/bus/iio/devices/iio:deviceX/vcxo_clk_present*
- */sys/bus/iio/devices/iio:deviceX/pllY_locked*
- */sys/bus/iio/devices/iio:deviceX/sync_dividers*

5.2.195 File testing/sysfs-bus-iio-frequency-adf4350

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/out_altvoltageY_frequency_resolution*
- */sys/bus/iio/devices/iio:deviceX/out_altvoltageY_refin_frequency*

5.2.196 File testing/sysfs-bus-iio-frequency-adf4371

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/out_altvoltageY_name*

5.2.197 File testing/sysfs-bus-iio-frequency-admv1013

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/in_altvoltage0-1_i_calibphase*
- */sys/bus/iio/devices/iio:deviceX/in_altvoltage0-1_q_calibphase*
- */sys/bus/iio/devices/iio:deviceX/in_altvoltage0_i_calibbias*
- */sys/bus/iio/devices/iio:deviceX/in_altvoltage0_q_calibbias*
- */sys/bus/iio/devices/iio:deviceX/in_altvoltage1_i_calibbias*
- */sys/bus/iio/devices/iio:deviceX/in_altvoltage1_q_calibbias*

5.2.198 File testing/sysfs-bus-iio-frequency-admv1014

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/in_altvoltage0_i_calibscale_coarse*
- */sys/bus/iio/devices/iio:deviceX/in_altvoltage0_q_calibscale_coarse*
- */sys/bus/iio/devices/iio:deviceX/in_altvoltage0_i_calibscale_fine*
- */sys/bus/iio/devices/iio:deviceX/in_altvoltage0_q_calibscale_fine*

5.2.199 File testing/sysfs-bus-iio-gyro-bmg160

Has the following ABI:

- */sys/bus/iio/devices/triggerX/name = "bmg160-any-motion-devX"*

5.2.200 File testing/sysfs-bus-iio-health-afe440x

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/in_intensityY_raw*
- */sys/bus/iio/devices/iio:deviceX/in_intensityY_offset*
- */sys/bus/iio/devices/iio:deviceX/in_intensityY_resistance*
- */sys/bus/iio/devices/iio:deviceX/in_intensityY_capacitance*

5.2.201 File testing/sysfs-bus-iio-humidity

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/out_current_heater_raw*
- */sys/bus/iio/devices/iio:deviceX/out_current_heater_raw_available*

5.2.202 File testing/sysfs-bus-iio-impedance-analyzer-ad5933

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/out_altvoltageY_frequency_start*
- */sys/bus/iio/devices/iio:deviceX/out_altvoltageY_frequency_increment*
- */sys/bus/iio/devices/iio:deviceX/out_altvoltageY_frequency_points*
- */sys/bus/iio/devices/iio:deviceX/out_altvoltageY_settling_cycles*

5.2.203 File testing/sysfs-bus-iio-ina2xx-adc

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/in_allow_async_readout*
- */sys/bus/iio/devices/iio:deviceX/in_shunt_resistor*

5.2.204 File testing/sysfs-bus-iio-isl29501

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/in_proximity0_agc_gain*
- */sys/bus/iio/devices/iio:deviceX/in_proximity0_agc_gain_bias*
- */sys/bus/iio/devices/iio:deviceX/in_proximity0_calib_phase_temp_a*
- */sys/bus/iio/devices/iio:deviceX/in_proximity0_calib_phase_temp_b*
- */sys/bus/iio/devices/iio:deviceX/in_proximity0_calib_phase_light_a*
- */sys/bus/iio/devices/iio:deviceX/in_proximity0_calib_phase_light_b*

5.2.205 File testing/sysfs-bus-iio-light-isl29018

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/proximity_on_chip_ambient_infrared_suppression*

5.2.206 File testing/sysfs-bus-iio-light-lm3533-als

Has the following ABI:

- */sys/.../events/in_illuminance0_thresh_either_en*
- */sys/.../events/in_illuminance0_threshY_hysteresis*
- */sys/.../events/illuminance_threshY_falling_value*
- */sys/.../events/illuminance_threshY_raising_value*
- */sys/bus/iio/devices/iio:deviceX/in_illuminance0_zone*
- */sys/bus/iio/devices/iio:deviceX/out_currentY_currentZ_raw*

5.2.207 File testing/sysfs-bus-iio-light-si1133

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/in_intensity_ir_small_raw*
- */sys/bus/iio/devices/iio:deviceX/in_intensity_ir_large_raw*
- */sys/bus/iio/devices/iio:deviceX/in_intensity_large_raw*

5.2.208 File testing/sysfs-bus-iio-light-tsl2583

Has the following ABI:

- */sys/bus/iio/devices/device[n]/in_illuminance_calibrate*
- */sys/bus/iio/devices/device[n]/in_illuminance_lux_table*
- */sys/bus/iio/devices/device[n]/in_illuminance_input_target*

5.2.209 File testing/sysfs-bus-iio-light-tsl2772

Has the following ABI:

- */sys/bus/iio/devices/device[n]/in_illuminance0_calibrate*
- */sys/bus/iio/devices/device[n]/in_proximity0_calibrate*

5.2.210 File testing/sysfs-bus-iio-magnetometer-hmc5843

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/meas_conf*
- */sys/bus/iio/devices/iio:deviceX/meas_conf_available*

5.2.211 File testing/sysfs-bus-iio-meas-spec

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/battery_low*

5.2.212 File testing/sysfs-bus-iio-mpu6050

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/in_gyro_matrix*
- */sys/bus/iio/devices/iio:deviceX/in_accel_matrix*
- */sys/bus/iio/devices/iio:deviceX/in_magn_matrix*

5.2.213 File testing/sysfs-bus-iio-potentiometer-mcp4531

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/out_resistance_raw_available*

5.2.214 File testing/sysfs-bus-iio-proximity

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/in_proximity_nearlevel*
- */sys/bus/iio/devices/iio:deviceX/sensor_sensitivity*

5.2.215 File testing/sysfs-bus-iio-proximity-as3935

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/in_proximity_input*

5.2.216 File testing/sysfs-bus-iio-resolver-ad2s1210

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/events/in_altvoltage0_mag_rising_reset_max*
- */sys/bus/iio/devices/iio:deviceX/events/in_altvoltage0_mag_rising_reset_max_available*
- */sys/bus/iio/devices/iio:deviceX/events/in_altvoltage0_mag_rising_reset_min*
- */sys/bus/iio/devices/iio:deviceX/events/in_altvoltage0_mag_rising_reset_min_available*

5.2.217 File testing/sysfs-bus-iio-sps30

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/start_cleaning*
- */sys/bus/iio/devices/iio:deviceX/cleaning_period*
- */sys/bus/iio/devices/iio:deviceX/cleaning_period_available*

5.2.218 File testing/sysfs-bus-iio-sx9310

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/in_proximity3_comb_raw*

5.2.219 File testing/sysfs-bus-iio-sx9324

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/in_proximity<id>_setup*

5.2.220 File testing/sysfs-bus-iio-thermocouple

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/fault_ovuv*
- */sys/bus/iio/devices/iio:deviceX/fault_oc*

5.2.221 File testing/sysfs-bus-iio-timer-stm32

Has the following ABI:

- */sys/bus/iio/devices/triggerX/master_mode_available*
- */sys/bus/iio/devices/triggerX/master_mode*
- */sys/bus/iio/devices/iio:deviceX/in_count0_preset*
- */sys/bus/iio/devices/iio:deviceX/in_count_enable_mode_available*
- */sys/bus/iio/devices/iio:deviceX/in_count0_enable_mode*
- */sys/bus/iio/devices/iio:deviceX/in_count_trigger_mode_available*
- */sys/bus/iio/devices/iio:deviceX/in_count0_trigger_mode*

5.2.222 File testing/sysfs-bus-iio-trigger-sysfs

Has the following ABI:

- */sys/bus/iio/devices/triggerX/trigger_now*
- */sys/bus/iio/devices/triggerX/name*
- */sys/bus/iio/devices/iio_sysfs_trigger/add_trigger*
- */sys/bus/iio/devices/iio_sysfs_trigger/remove_trigger*

5.2.223 File testing/sysfs-bus-iio-vf610

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/in_conversion_mode*
- */sys/bus/iio/devices/iio:deviceX/out_conversion_mode*

5.2.224 File testing/sysfs-bus-intel_th-devices-gth

Has the following ABI:

- */sys/bus/intel_th/devices/<intel_th_id>-gth/masters/**
- */sys/bus/intel_th/devices/<intel_th_id>-gth/outputs/[0-7]_port*
- */sys/bus/intel_th/devices/<intel_th_id>-gth/outputs/[0-7]_drop*
- */sys/bus/intel_th/devices/<intel_th_id>-gth/outputs/[0-7]_null*
- */sys/bus/intel_th/devices/<intel_th_id>-gth/outputs/[0-7]_flush*
- */sys/bus/intel_th/devices/<intel_th_id>-gth/outputs/[0-7]_reset*
- */sys/bus/intel_th/devices/<intel_th_id>-gth/outputs/[0-7]_smcfreq*

5.2.225 File testing/sysfs-bus-intel_th-devices-msc

Has the following ABI:

- */sys/bus/intel_th/devices/<intel_th_id>-msc<msc-id>/wrap*
- */sys/bus/intel_th/devices/<intel_th_id>-msc<msc-id>/mode*
- */sys/bus/intel_th/devices/<intel_th_id>-msc<msc-id>/nr_pages*
- */sys/bus/intel_th/devices/<intel_th_id>-msc<msc-id>/win_switch*
- */sys/bus/intel_th/devices/<intel_th_id>-msc<msc-id>/stop_on_full*

5.2.226 File testing/sysfs-bus-intel_th-devices-pti

Has the following ABI:

- */sys/bus/intel_th/devices/<intel_th_id>-pti/mode*
- */sys/bus/intel_th/devices/<intel_th_id>-pti/freerunning_clock*
- */sys/bus/intel_th/devices/<intel_th_id>-pti/clock_divider*

5.2.227 File testing/sysfs-bus-intel_th-output-devices

Has the following ABI:

- */sys/bus/intel_th/devices/<intel_th_id>-<device><id>/active*
- */sys/bus/intel_th/devices/<intel_th_id>-msc<msc-id>/port*

5.2.228 File testing/sysfs-bus-mcb

Has the following ABI:

- */sys/bus/mcb/devices/mcb:X*
- */sys/bus/mcb/devices/mcb:X/revision*
- */sys/bus/mcb/devices/mcb:X/minor*
- */sys/bus/mcb/devices/mcb:X/model*
- */sys/bus/mcb/devices/mcb:X/name*

5.2.229 File testing/sysfs-bus-mdio

Has the following ABI:

- */sys/bus/mdio_bus/devices/.../statistics/*
- */sys/class/mdio_bus/.../statistics/*
- */sys/bus/mdio_bus/devices/.../statistics/transfers*
- */sys/class/mdio_bus/.../transfers*
- */sys/bus/mdio_bus/devices/.../statistics/errors*
- */sys/class/mdio_bus/.../statistics/errors*
- */sys/bus/mdio_bus/devices/.../statistics/writes*
- */sys/class/mdio_bus/.../statistics/writes*
- */sys/bus/mdio_bus/devices/.../statistics/reads*
- */sys/class/mdio_bus/.../statistics/reads*
- */sys/bus/mdio_bus/devices/.../statistics/transfers_<addr>*
- */sys/class/mdio_bus/.../statistics/transfers_<addr>*
- */sys/bus/mdio_bus/devices/.../statistics/errors_<addr>*

- */sys/class/mdio_bus/.../statistics/errors_<addr>*
- */sys/bus/mdio_bus/devices/.../statistics/writes_<addr>*
- */sys/class/mdio_bus/.../statistics/writes_<addr>*
- */sys/bus/mdio_bus/devices/.../statistics/reads_<addr>*
- */sys/class/mdio_bus/.../statistics/reads_<addr>*

5.2.230 File testing/sysfs-bus-media

Has the following ABI:

- */sys/bus/media/devices/.../model*

5.2.231 File testing/sysfs-bus-mei

Has the following ABI:

- */sys/bus/mei/devices/.../modalias*
- */sys/bus/mei/devices/.../name*
- */sys/bus/mei/devices/.../uuid*
- */sys/bus/mei/devices/.../version*
- */sys/bus/mei/devices/.../max_conn*
- */sys/bus/mei/devices/.../fixed*
- */sys/bus/mei/devices/.../vtag*
- */sys/bus/mei/devices/.../max_len*

5.2.232 File testing/sysfs-bus-mmc

Has the following ABI:

- */sys/bus/mmc/devices/.../rev*

5.2.233 File testing/sysfs-bus-most

Has the following ABI:

- */sys/bus/most/devices/<dev>/description*
- */sys/bus/most/devices/<dev>/interface*
- */sys/bus/most/devices/<dev>/dci*
- */sys/bus/most/devices/<dev>/dci/arb_address*
- */sys/bus/most/devices/<dev>/dci/arb_value*
- */sys/bus/most/devices/<dev>/dci/mep_eui48_hi*
- */sys/bus/most/devices/<dev>/dci/mep_eui48_lo*

- /sys/bus/most/devices/<dev>/dci/mep_eui48_mi
- /sys/bus/most/devices/<dev>/dci/mep_filter
- /sys/bus/most/devices/<dev>/dci/mep_hash0
- /sys/bus/most/devices/<dev>/dci/mep_hash1
- /sys/bus/most/devices/<dev>/dci/mep_hash2
- /sys/bus/most/devices/<dev>/dci/mep_hash3
- /sys/bus/most/devices/<dev>/dci/ni_state
- /sys/bus/most/devices/<dev>/dci/node_address
- /sys/bus/most/devices/<dev>/dci/node_position
- /sys/bus/most/devices/<dev>/dci/packet_bandwidth
- /sys/bus/most/devices/<dev>/dci/sync_ep
- /sys/bus/most/devices/<dev>/<channel>/
- /sys/bus/most/devices/<dev>/<channel>/available_datatypes
- /sys/bus/most/devices/<dev>/<channel>/available_directions
- /sys/bus/most/devices/<dev>/<channel>/number_of_packet_buffers
- /sys/bus/most/devices/<dev>/<channel>/number_of_stream_buffers
- /sys/bus/most/devices/<dev>/<channel>/size_of_packet_buffer
- /sys/bus/most/devices/<dev>/<channel>/size_of_stream_buffer
- /sys/bus/most/devices/<dev>/<channel>/set_number_of_buffers
- /sys/bus/most/devices/<dev>/<channel>/set_buffer_size
- /sys/bus/most/devices/<dev>/<channel>/set_direction
- /sys/bus/most/devices/<dev>/<channel>/set_datatype
- /sys/bus/most/devices/<dev>/<channel>/set_subbuffer_size
- /sys/bus/most/devices/<dev>/<channel>/set_packets_per_xact
- /sys/bus/most/devices/<dev>/<channel>/channel_starving
- /sys/bus/most/drivers/most_core/components
- /sys/bus/most/drivers/most_core/links

5.2.234 File testing/sysfs-bus-moxtet-devices

Has the following ABI:

- /sys/bus/moxtet/devices/moxtet-<name>.<addr>/module_description
- /sys/bus/moxtet/devices/moxtet-<name>.<addr>/module_id
- /sys/bus/moxtet/devices/moxtet-<name>.<addr>/module_name

5.2.235 File testing/sysfs-bus-nfit

For all of the nmem device attributes under `nfit/*`, see the 'NVDIMM Firmware Interface Table (NFIT)' section in the ACPI specification (http://www.uefi.org/specifications) for more details.

Has the following ABI:

- */sys/bus/nd/devices/nmemX/nfit/serial*
- */sys/bus/nd/devices/nmemX/nfit/handle*
- */sys/bus/nd/devices/nmemX/nfit/device*
- */sys/bus/nd/devices/nmemX/nfit/rev_id*
- */sys/bus/nd/devices/nmemX/nfit/phys_id*
- */sys/bus/nd/devices/nmemX/nfit/flags*
- */sys/bus/nd/devices/nmemX/nfit/format*
- */sys/bus/nd/devices/nmemX/nfit/format1*
- */sys/bus/nd/devices/nmemX/nfit/formats*
- */sys/bus/nd/devices/nmemX/nfit/vendor*
- */sys/bus/nd/devices/nmemX/nfit/dsm_mask*
- */sys/bus/nd/devices/nmemX/nfit/family*
- */sys/bus/nd/devices/nmemX/nfit/id*
- */sys/bus/nd/devices/nmemX/nfit/subsystem_vendor*
- */sys/bus/nd/devices/nmemX/nfit/subsystem_rev_id*
- */sys/bus/nd/devices/nmemX/nfit/subsystem_device*
- */sys/bus/nd/devices/ndbusX/nfit/revision*
- */sys/bus/nd/devices/ndbusX/nfit/scrub*
- */sys/bus/nd/devices/ndbusX/nfit/hw_error_scrub*
- */sys/bus/nd/devices/ndbusX/nfit/dsm_mask*
- */sys/bus/nd/devices/ndbusX/nfit/firmware_activate_noidle*
- */sys/bus/nd/devices/regionX/nfit/range_index*

5.2.236 File testing/sysfs-bus-nvdimm

Has the following ABI:

- *nvdimm*
- */sys/bus/event_source/devices/nmemX/format*
- */sys/bus/event_source/devices/nmemX/events*
- */sys/bus/event_source/devices/nmemX/cpumask*
- */sys/bus/nd/devices/nmemX/cxl/id*

- */sys/bus/nd/devices/nmemX/cxl/provider*

5.2.237 File testing/sysfs-bus-optee-devices

Has the following ABI:

- */sys/bus/tee/devices/optee-ta-<uuid>/*
- */sys/bus/tee/devices/optee-ta-<uuid>/need_supplicant*

5.2.238 File testing/sysfs-bus-papr-pmem

Has the following ABI:

- */sys/bus/nd/devices/nmemX/papr/flags*
- */sys/bus/nd/devices/nmemX/papr/perf_stats*
- */sys/bus/nd/devices/nmemX/papr/health_bitmap_inject*

5.2.239 File testing/sysfs-bus-pci

Has the following ABI:

- */sys/bus/pci/drivers/.../bind*
- */sys/devices/pciX/.../bind*
- */sys/bus/pci/drivers/.../unbind*
- */sys/devices/pciX/.../unbind*
- */sys/bus/pci/drivers/.../new_id*
- */sys/devices/pciX/.../new_id*
- */sys/bus/pci/drivers/.../remove_id*
- */sys/devices/pciX/.../remove_id*
- */sys/bus/pci/rescan*
- */sys/bus/pci/devices/.../msi_bus*
- */sys/bus/pci/devices/.../msi_irqs/*
- */sys/bus/pci/devices/.../msi_irqs/<N>*
- */sys/bus/pci/devices/.../irq*
- */sys/bus/pci/devices/.../remove*
- */sys/bus/pci/devices/.../pci_bus/.../rescan*
- */sys/bus/pci/devices/.../rescan*
- */sys/bus/pci/devices/.../reset_method*
- */sys/bus/pci/devices/.../reset*
- */sys/bus/pci/devices/.../vpd*

- */sys/bus/pci/devices/.../virtfn<N>*
- */sys/bus/pci/devices/.../dep_link*
- */sys/bus/pci/devices/.../physfn*
- */sys/bus/pci/devices/.../modalias*
- */sys/bus/pci/slots/.../module*
- */sys/bus/pci/devices/.../label*
- */sys/bus/pci/devices/.../index*
- */sys/bus/pci/devices/.../acpi_index*
- */sys/bus/pci/devices/.../d3cold_allowed*
- */sys/bus/pci/devices/.../sriov_totalvfs*
- */sys/bus/pci/devices/.../sriov_numvfs*
- */sys/bus/pci/devices/.../driver_override*
- */sys/bus/pci/devices/.../numa_node*
- */sys/bus/pci/devices/.../revision*
- */sys/bus/pci/devices/.../sriov_drivers_autoprobe*
- */sys/bus/pci/devices/.../p2pmem/size*
- */sys/bus/pci/devices/.../p2pmem/available*
- */sys/bus/pci/devices/.../p2pmem/published*
- */sys/bus/pci/devices/.../p2pmem/allocate*
- */sys/bus/pci/devices/.../link/clkpm*
- */sys/bus/pci/devices/.../power_state*
- */sys/bus/pci/devices/.../sriov_vf_total_msix*
- */sys/bus/pci/devices/.../sriov_vf_msix_count*
- */sys/bus/pci/devices/.../resourceN_resize*

5.2.240 File testing/sysfs-bus-pci-devices-aer_stats

PCIe Device AER statistics

These attributes show up under all the devices that are AER capable. These statistical counters indicate the errors "as seen/reported by the device". Note that this may mean that if an endpoint is causing problems, the AER counters may increment at its link partner (e.g. root port) because the errors may be "seen" / reported by the link partner and not the problematic endpoint itself (which may report all counters as 0 as it never saw any problems).

Has the following ABI:

- */sys/bus/pci/devices/<dev>/aer_dev_correctable*
- */sys/bus/pci/devices/<dev>/aer_dev_fatal*
- */sys/bus/pci/devices/<dev>/aer_dev_nonfatal*

- */sys/bus/pci/devices/<dev>/aer_stats/aer_rootport_total_err_cor*
- */sys/bus/pci/devices/<dev>/aer_stats/aer_rootport_total_err_fatal*
- */sys/bus/pci/devices/<dev>/aer_stats/aer_rootport_total_err_nonfatal*

5.2.241 File testing/sysfs-bus-pci-devices-catpt

Has the following ABI:

- */sys/devices/pci0000:00/<dev>/fw_version*
- */sys/devices/pci0000:00/<dev>/fw_info*

5.2.242 File testing/sysfs-bus-pci-devices-cciss

Has the following ABI:

- */sys/bus/pci/devices/<dev>/ccissX/cXdY/model*
- */sys/bus/pci/devices/<dev>/ccissX/cXdY/rev*
- */sys/bus/pci/devices/<dev>/ccissX/cXdY/unique_id*
- */sys/bus/pci/devices/<dev>/ccissX/cXdY/vendor*
- */sys/bus/pci/devices/<dev>/ccissX/cXdY/block:cciss!cXdY*
- */sys/bus/pci/devices/<dev>/ccissX/rescan*
- */sys/bus/pci/devices/<dev>/ccissX/cXdY/lunid*
- */sys/bus/pci/devices/<dev>/ccissX/cXdY/raid_level*
- */sys/bus/pci/devices/<dev>/ccissX/cXdY/usage_count*
- */sys/bus/pci/devices/<dev>/ccissX/resettable*
- */sys/bus/pci/devices/<dev>/ccissX/transport_mode*

5.2.243 File testing/sysfs-bus-pci-devices-pvpanic

Has the following ABI:

- */sys/devices/pci0000:00/*/QEMU0001:00/capability for MMIO*
- */sys/devices/pci0000:00/*/QEMU0001:00/events*

5.2.244 File testing/sysfs-bus-pci-drivers-ehci_hcd

Has the following ABI:

- */sys/bus/pci/drivers/ehci_hcd/.../companion*

5.2.245 File testing/sysfs-bus-pci-drivers-janz-cmodio

Has the following ABI:

- */sys/bus/pci/drivers/janz-cmodio/.../modulbus_number*

5.2.246 File testing/sysfs-bus-pci-drivers-xhci_hcd

Has the following ABI:

- */sys/bus/pci/drivers/xhci_hcd/.../dbc*
- */sys/bus/pci/drivers/xhci_hcd/.../dbc_idVendor*
- */sys/bus/pci/drivers/xhci_hcd/.../dbc_idProduct*
- */sys/bus/pci/drivers/xhci_hcd/.../dbc_bcdDevice*
- */sys/bus/pci/drivers/xhci_hcd/.../dbc_bInterfaceProtocol*

5.2.247 File testing/sysfs-bus-peci

Has the following ABI:

- */sys/bus/peci/rescan*
- */sys/bus/peci/devices/<controller_id>-<device_addr>/remove*

5.2.248 File testing/sysfs-bus-platform

Has the following ABI:

- */sys/bus/platform/devices/.../driver_override*
- */sys/bus/platform/devices/.../numa_node*
- */sys/bus/platform/devices/.../msi_irqs/*
- */sys/bus/platform/devices/.../msi_irqs/<N>*
- */sys/bus/platform/devices/.../modalias*

5.2.249 File testing/sysfs-bus-platform-devices-ampere-smpro

Has the following ABI:

- */sys/bus/platform/devices/smpro-errmon.*/error_[core|mem|pcie|other]_[ce|ue]*
- */sys/bus/platform/devices/smpro-errmon.*/overflow_[core|mem|pcie|other]_[ce|ue]*
- */sys/bus/platform/devices/smpro-errmon.*/[error|warn]_[smpro|pmpro]*
- */sys/bus/platform/devices/smpro-errmon.*/event_[vrd_warn_fault|vrd_hot|dimm_hot|dimm_2x_re*
- */sys/bus/platform/devices/smpro-errmon.*/event_dimm[0-15]_syndrome*
- */sys/bus/platform/devices/smpro-misc.*/boot_progress*
- */sys/bus/platform/devices/smpro-misc*/soc_power_limit*

5.2.250 File testing/sysfs-bus-platform-devices-occ-hwmon

Has the following ABI:

- */sys/bus/platform/devices/occ-hwmon.X/ffdc*

5.2.251 File testing/sysfs-bus-platform-onboard-usb-hub

Has the following ABI:

- */sys/bus/platform/devices/<dev>/always_powered_in_suspend*

5.2.252 File testing/sysfs-bus-rapidio

Has the following ABI:

- */sys/bus/rapidio/devices/<nn>:<d>:<iiii>*
- */sys/bus/rapidio/devices/<nn>:<d>:<iiii>/did*
- */sys/bus/rapidio/devices/<nn>:<d>:<iiii>/vid*
- */sys/bus/rapidio/devices/<nn>:<d>:<iiii>/device_rev*
- */sys/bus/rapidio/devices/<nn>:<d>:<iiii>/asm_did*
- */sys/bus/rapidio/devices/<nn>:<d>:<iiii>/asm_rev*
- */sys/bus/rapidio/devices/<nn>:<d>:<iiii>/asm_vid*
- */sys/bus/rapidio/devices/<nn>:<d>:<iiii>/destid*
- */sys/bus/rapidio/devices/<nn>:<d>:<iiii>/lprev*
- */sys/bus/rapidio/devices/<nn>:<d>:<iiii>/modalias*
- */sys/bus/rapidio/devices/<nn>:<d>:<iiii>/config*
- */sys/bus/rapidio/devices/<nn>:<s>:<iiii>/routes*
- */sys/bus/rapidio/devices/<nn>:<s>:<iiii>/destid*
- */sys/bus/rapidio/devices/<nn>:<s>:<iiii>/hopcount*

- */sys/bus/rapidio/devices/<nn>:<s>:<iiii>/lnext*
- */sys/bus/rapidio/devices/<nn>:<s>:<iiii>/errlog*
- */sys/bus/rapidio/scan*

5.2.253 File testing/sysfs-bus-rbd

Has the following ABI:

- */sys/bus/rbd/add*
- */sys/bus/rbd/remove*
- */sys/bus/rbd/add_single_major*
- */sys/bus/rbd/remove_single_major*
- */sys/bus/rbd/supported_features*
- */sys/bus/rbd/devices/<dev-id>/size*
- */sys/bus/rbd/devices/<dev-id>/major*
- */sys/bus/rbd/devices/<dev-id>/client_id*
- */sys/bus/rbd/devices/<dev-id>/pool*
- */sys/bus/rbd/devices/<dev-id>/name*
- */sys/bus/rbd/devices/<dev-id>/refresh*
- */sys/bus/rbd/devices/<dev-id>/current_snap*
- */sys/bus/rbd/devices/<dev-id>/pool_id*
- */sys/bus/rbd/devices/<dev-id>/image_id*
- */sys/bus/rbd/devices/<dev-id>/features*
- */sys/bus/rbd/devices/<dev-id>/parent*
- */sys/bus/rbd/devices/<dev-id>/minor*
- */sys/bus/rbd/devices/<dev-id>/snap_id*
- */sys/bus/rbd/devices/<dev-id>/config_info*
- */sys/bus/rbd/devices/<dev-id>/cluster_fsid*
- */sys/bus/rbd/devices/<dev-id>/client_addr*

5.2.254 File testing/sysfs-bus-rpmsg

Has the following ABI:

- */sys/bus/rpmsg/devices/.../name*
- */sys/bus/rpmsg/devices/.../src*
- */sys/bus/rpmsg/devices/.../dst*
- */sys/bus/rpmsg/devices/.../announce*

- */sys/bus/rpmsg/devices/.../driver_override*

5.2.255 File testing/sysfs-bus-siox

Has the following ABI:

- */sys/bus/siox/devices/siox-X/active*
- */sys/bus/siox/devices/siox-X/device_add*
- */sys/bus/siox/devices/siox-X/device_remove*
- */sys/bus/siox/devices/siox-X/poll_interval_ns*
- */sys/bus/siox/devices/siox-X-Y/connected*
- */sys/bus/siox/devices/siox-X-Y/inbytes*
- */sys/bus/siox/devices/siox-X-Y/status_errors*
- */sys/bus/siox/devices/siox-X-Y/type*
- */sys/bus/siox/devices/siox-X-Y/watchdog*
- */sys/bus/siox/devices/siox-X-Y/watchdog_errors*
- */sys/bus/siox/devices/siox-X-Y/outbytes*

5.2.256 File testing/sysfs-bus-soundwire-master

Has the following ABI:

- */sys/bus/soundwire/devices/sdw-master-<N>/revision*

5.2.257 File testing/sysfs-bus-soundwire-slave

Has the following ABI:

- */sys/bus/soundwire/devices/sdw:.../status*
- */sys/bus/soundwire/devices/sdw:.../dev-properties/mipi_revision*
- */sys/bus/soundwire/devices/sdw:.../dp0/max_word*
- */sys/bus/soundwire/devices/sdw:.../dp<N>_src/max_word*

5.2.258 File testing/sysfs-bus-spi-devices-spi-nor

Has the following ABI:

- */sys/bus/spi/devices/.../spi-nor/jedec_id*
- */sys/bus/spi/devices/.../spi-nor/manufacturer*
- */sys/bus/spi/devices/.../spi-nor/partname*
- */sys/bus/spi/devices/.../spi-nor/sfdp*

5.2.259 File testing/sysfs-bus-surface_aggregator-tabletsw

Has the following ABI:

- */sys/bus/surface_aggregator/devices/01:0e:01:00:01/state*
- */sys/bus/surface_aggregator/devices/01:26:01:00:01/state*

5.2.260 File testing/sysfs-bus-thunderbolt

Has the following ABI:

- */sys/bus/thunderbolt/devices/.../domainX/boot_acl*
- */sys/bus/thunderbolt/devices/.../domainX/deauthorization*
- */sys/bus/thunderbolt/devices/.../domainX/iommu_dma_protection*
- */sys/bus/thunderbolt/devices/.../domainX/security*
- */sys/bus/thunderbolt/devices/.../authorized*
- */sys/bus/thunderbolt/devices/.../boot*
- */sys/bus/thunderbolt/devices/.../generation*
- */sys/bus/thunderbolt/devices/.../key*
- */sys/bus/thunderbolt/devices/.../device*
- */sys/bus/thunderbolt/devices/.../device_name*
- */sys/bus/thunderbolt/devices/.../maxhopid*
- */sys/bus/thunderbolt/devices/.../rx_speed*
- */sys/bus/thunderbolt/devices/.../rx_lanes*
- */sys/bus/thunderbolt/devices/.../tx_speed*
- */sys/bus/thunderbolt/devices/.../tx_lanes*
- */sys/bus/thunderbolt/devices/.../vendor*
- */sys/bus/thunderbolt/devices/.../vendor_name*
- */sys/bus/thunderbolt/devices/.../unique_id*
- */sys/bus/thunderbolt/devices/.../nvm_version*
- */sys/bus/thunderbolt/devices/.../nvm_authenticate*
- */sys/bus/thunderbolt/devices/.../nvm_authenticate_on_disconnect*
- */sys/bus/thunderbolt/devices/<xdomain>.<service>/key*
- */sys/bus/thunderbolt/devices/<xdomain>.<service>/modalias*
- */sys/bus/thunderbolt/devices/<xdomain>.<service>/prtcid*
- */sys/bus/thunderbolt/devices/<xdomain>.<service>/prtcvers*
- */sys/bus/thunderbolt/devices/<xdomain>.<service>/prtcrevs*
- */sys/bus/thunderbolt/devices/<xdomain>.<service>/prtcstns*

- */sys/bus/thunderbolt/devices/usb4_portX/connector*
- */sys/bus/thunderbolt/devices/usb4_portX/link*
- */sys/bus/thunderbolt/devices/usb4_portX/offline*
- */sys/bus/thunderbolt/devices/usb4_portX/rescan*
- */sys/bus/thunderbolt/devices/<device>:<port>.<index>/device*
- */sys/bus/thunderbolt/devices/<device>:<port>.<index>/nvm_authenticate*
- */sys/bus/thunderbolt/devices/<device>:<port>.<index>/nvm_version*
- */sys/bus/thunderbolt/devices/<device>:<port>.<index>/vendor*

5.2.261 File testing/sysfs-bus-typec

Has the following ABI:

- */sys/bus/typec/devices/.../active*
- */sys/bus/typec/devices/.../description*
- */sys/bus/typec/devices/.../mode*
- */sys/bus/typec/devices/.../svid*
- */sys/bus/typec/devices/.../vdo*

5.2.262 File testing/sysfs-bus-usb

Has the following ABI:

- */sys/bus/usb/devices/<INTERFACE>/authorized*
- */sys/bus/usb/devices/usbX/interface_authorized_default*
- */sys/bus/usb/device/.../authorized*
- */sys/bus/usb/drivers/.../new_id*
- */sys/bus/usb-serial/drivers/.../new_id*
- */sys/bus/usb/drivers/.../remove_id*
- */sys/bus/usb/devices/.../power/usb2_hardware_lpm*
- */sys/bus/usb/devices/.../power/usb3_hardware_lpm_u1*
- */sys/bus/usb/devices/.../ltm_capable*
- */sys/bus/usb/devices/<INTERFACE>/wireless_status*
- */sys/bus/usb/devices/.../<hub_interface>/port<X>*
- */sys/bus/usb/devices/.../<hub_interface>/port<X>/connect_type*
- */sys/bus/usb/devices/.../<hub_interface>/port<X>/location*
- */sys/bus/usb/devices/.../<hub_interface>/port<X>/quirks*
- */sys/bus/usb/devices/.../<hub_interface>/port<X>/over_current_count*

- */sys/bus/usb/devices/.../<hub_interface>/port<X>/usb3_lpm_permit*
- */sys/bus/usb/devices/.../<hub_interface>/port<X>/connector*
- */sys/bus/usb/devices/.../<hub_interface>/port<X>/disable*
- */sys/bus/usb/devices/.../<hub_interface>/port<X>/early_stop*
- */sys/bus/usb/devices/.../<hub_interface>/port<X>/state*
- */sys/bus/usb/devices/.../power/usb2_lpm_l1_timeout*
- */sys/bus/usb/devices/.../power/usb2_lpm_besl*
- */sys/bus/usb/devices/.../rx_lanes*
- */sys/bus/usb/devices/.../tx_lanes*
- */sys/bus/usb/devices/.../typec*
- */sys/bus/usb/devices/usbX/bAlternateSetting*
- */sys/bus/usb/devices/usbX/bcdDevice*
- */sys/bus/usb/devices/usbX/bConfigurationValue*
- */sys/bus/usb/devices/usbX/bDeviceClass*
- */sys/bus/usb/devices/usbX/bDeviceProtocol*
- */sys/bus/usb/devices/usbX/bDeviceSubClass*
- */sys/bus/usb/devices/usbX/bInterfaceClass*
- */sys/bus/usb/devices/usbX/bInterfaceNumber*
- */sys/bus/usb/devices/usbX/bInterfaceProtocol*
- */sys/bus/usb/devices/usbX/bInterfaceSubClass*
- */sys/bus/usb/devices/usbX/bmAttributes*
- */sys/bus/usb/devices/usbX/bMaxPacketSize0*
- */sys/bus/usb/devices/usbX/bMaxPower*
- */sys/bus/usb/devices/usbX/bNumConfigurations*
- */sys/bus/usb/devices/usbX/bNumEndpoints*
- */sys/bus/usb/devices/usbX/bNumInterfaces*
- */sys/bus/usb/devices/usbX/busnum*
- */sys/bus/usb/devices/usbX/configuration*
- */sys/bus/usb/devices/usbX/descriptors*
- */sys/bus/usb/devices/usbX/idProduct*
- */sys/bus/usb/devices/usbX/idVendor*
- */sys/bus/usb/devices/usbX/devspec*
- */sys/bus/usb/devices/usbX/avoid_reset_quirk*
- */sys/bus/usb/devices/usbX/devnum*

- */sys/bus/usb/devices/usbX/devpath*
- */sys/bus/usb/devices/usbX/manufacturer*
- */sys/bus/usb/devices/usbX/maxchild*
- */sys/bus/usb/devices/usbX/persist*
- */sys/bus/usb/devices/usbX/product*
- */sys/bus/usb/devices/usbX/speed*
- */sys/bus/usb/devices/usbX/supports_autosuspend*
- */sys/bus/usb/devices/usbX/urbnum*
- */sys/bus/usb/devices/usbX/version*
- */sys/bus/usb/devices/usbX/power/autosuspend*
- */sys/bus/usb/devices/usbX/power/active_duration*
- */sys/bus/usb/devices/usbX/power/connected_duration*
- */sys/bus/usb/devices/usbX/power/level*
- */sys/bus/usb/devices/usbX/ep_<N>/bEndpointAddress*
- */sys/bus/usb/devices/usbX/ep_<N>/bInterval*
- */sys/bus/usb/devices/usbX/ep_<N>/bLength*
- */sys/bus/usb/devices/usbX/ep_<N>/bmAttributes*
- */sys/bus/usb/devices/usbX/ep_<N>/direction*
- */sys/bus/usb/devices/usbX/ep_<N>/interval*
- */sys/bus/usb/devices/usbX/ep_<N>/type*
- */sys/bus/usb/devices/usbX/ep_<N>/wMaxPacketSize*

5.2.263 File testing/sysfs-bus-usb-devices-usbsevseg

Has the following ABI:

- */sys/bus/usb/.../powered*
- */sys/bus/usb/.../mode_msb*
- */sys/bus/usb/.../mode_lsb*
- */sys/bus/usb/.../textmode*
- */sys/bus/usb/.../text*
- */sys/bus/usb/.../decimals*

5.2.264 File testing/sysfs-bus-usb-lvstest

Link Layer Validation Device is a standard device for testing of Super Speed Link Layer tests. These nodes are available in sysfs only when lvs driver is bound with root hub device.

Has the following ABI:

- */sys/bus/usb/devices/.../get_dev_desc*
- */sys/bus/usb/devices/.../u1_timeout*
- */sys/bus/usb/devices/.../u2_timeout*
- */sys/bus/usb/devices/.../hot_reset*
- */sys/bus/usb/devices/.../u3_entry*
- */sys/bus/usb/devices/.../u3_exit*
- */sys/bus/usb/devices/.../enable_compliance*
- */sys/bus/usb/devices/.../warm_reset*

5.2.265 File testing/sysfs-bus-vdpa

Has the following ABI:

- */sys/bus/vdpa/drivers_autoprobe*
- */sys/bus/vdpa/driver_probe*
- */sys/bus/vdpa/drivers/.../bind*
- */sys/bus/vdpa/drivers/.../unbind*
- */sys/bus/vdpa/devices/.../driver_override*

5.2.266 File testing/sysfs-bus-vfio-mdev

Has the following ABI:

- */sys/.../<device>/mdev_supported_types/*
- */sys/.../<device>/mdev_supported_types/<type-id>/*
- */sys/.../mdev_supported_types/<type-id>/create*
- */sys/.../mdev_supported_types/<type-id>/devices/*
- */sys/.../mdev_supported_types/<type-id>/available_instances*
- */sys/.../mdev_supported_types/<type-id>/device_api*
- */sys/.../mdev_supported_types/<type-id>/name*
- */sys/.../mdev_supported_types/<type-id>/description*
- */sys/.../<device>/<UUID>/*
- */sys/.../<device>/<UUID>/mdev_type*
- */sys/.../<device>/<UUID>/remove*

5.2.267 File testing/sysfs-bus-vmbus

Has the following ABI:

- */sys/bus/vmbus/devices/.../driver_override*

5.2.268 File testing/sysfs-c2port

Has the following ABI:

- */sys/class/c2port/*
- */sys/class/c2port/c2portX*
- */sys/class/c2port/c2portX/access*
- */sys/class/c2port/c2portX/dev_id*
- */sys/class/c2port/c2portX/flash_access*
- */sys/class/c2port/c2portX/flash_block_size*
- */sys/class/c2port/c2portX/flash_blocks_num*
- */sys/class/c2port/c2portX/flash_data*
- */sys/class/c2port/c2portX/flash_erase*
- */sys/class/c2port/c2portX/reset*
- */sys/class/c2port/c2portX/rev_id*

5.2.269 File testing/sysfs-cfq-target-latency

Has the following ABI:

- */sys/block/<device>/iosched/target_latency*

5.2.270 File testing/sysfs-class

Has the following ABI:

- */sys/class/*

5.2.271 File testing/sysfs-class-backlight

Has the following ABI:

- */sys/class/backlight/<backlight>/scale*
- */sys/class/backlight/<backlight>/ambient_light_level*
- */sys/class/backlight/<backlight>/ambient_light_zone*
- */sys/class/backlight/<backlight>/<ambient light zone>_max*
- */sys/class/backlight/<backlight>/<ambient light zone>_dim*

5.2.272 File testing/sysfs-class-backlight-driver-lm3533

Has the following ABI:

- */sys/class/backlight/<backlight>/als_channel*
- */sys/class/backlight/<backlight>/als_en*
- */sys/class/backlight/<backlight>/id*
- */sys/class/backlight/<backlight>/linear*
- */sys/class/backlight/<backlight>/pwm*

5.2.273 File testing/sysfs-class-backlight-lm3639

sysfs interface for Texas Instruments lm3639 backlight + flash led driver chip

Has the following ABI:

- */sys/class/backlight/<backlight>/bled_mode*

5.2.274 File testing/sysfs-class-bdi

Has the following ABI:

- */sys/class/bdi/<bdi>/*
- */sys/class/bdi/<bdi>/read_ahead_kb*
- */sys/class/bdi/<bdi>/min_ratio*
- */sys/class/bdi/<bdi>/min_ratio_fine*
- */sys/class/bdi/<bdi>/max_ratio*
- */sys/class/bdi/<bdi>/max_ratio_fine*
- */sys/class/bdi/<bdi>/min_bytes*
- */sys/class/bdi/<bdi>/max_bytes*
- */sys/class/bdi/<bdi>/strict_limit*
- */sys/class/bdi/<bdi>/stable_pages_required*

5.2.275 File testing/sysfs-class-bsr

Has the following ABI:

- */sys/class/bsr/bsr*/bsr_size*
- */sys/class/bsr/bsr*/bsr_length*
- */sys/class/bsr/bsr*/bsr_stride*

5.2.276 File testing/sysfs-class-chromeos

Has the following ABI:

- */sys/class/chromeos/<ec-device-name>/flashinfo*
- */sys/class/chromeos/<ec-device-name>/kb_wake_angle*
- */sys/class/chromeos/<ec-device-name>/reboot*
- */sys/class/chromeos/<ec-device-name>/version*

5.2.277 File testing/sysfs-class-chromeos-driver-cros-ec-lightbar

Has the following ABI:

- */sys/class/chromeos/<ec-device-name>/lightbar/brightness*
- */sys/class/chromeos/<ec-device-name>/lightbar/interval_msec*
- */sys/class/chromeos/<ec-device-name>/lightbar/led_rgb*
- */sys/class/chromeos/<ec-device-name>/lightbar/program*
- */sys/class/chromeos/<ec-device-name>/lightbar/sequence*
- */sys/class/chromeos/<ec-device-name>/lightbar/userspace_control*
- */sys/class/chromeos/<ec-device-name>/lightbar/version*

5.2.278 File testing/sysfs-class-chromeos-driver-cros-ec-vbc

Has the following ABI:

- */sys/class/chromeos/<ec-device-name>/vbc/vboot_context*

5.2.279 File testing/sysfs-class-cxl

Please note that attributes that are shared between devices are stored in the directory pointed to by the symlink device/. For example, the real path of the attribute /sys/class/cxl/afu0.0s/irqs_max is /sys/class/cxl/afu0.0s/device/irqs_max, i.e. /sys/class/cxl/afu0.0/irqs_max.

Slave contexts (eg. /sys/class/cxl/afu0.0s):

Has the following ABI:

- */sys/class/cxl/<afu>/afu_err_buf*
- */sys/class/cxl/<afu>/irqs_max*
- */sys/class/cxl/<afu>/irqs_min*
- */sys/class/cxl/<afu>/mmio_size*
- */sys/class/cxl/<afu>/modes_supported*
- */sys/class/cxl/<afu>/mode*
- */sys/class/cxl/<afu>/prefault_mode*

- */sys/class/cxl/<afu>/reset*
- */sys/class/cxl/<afu>/api_version*
- */sys/class/cxl/<afu>/api_version_compatible*
- */sys/class/cxl/<afu>/cr<config num>/vendor*
- */sys/class/cxl/<afu>/cr<config num>/device*
- */sys/class/cxl/<afu>/cr<config num>/class*
- */sys/class/cxl/<afu>/cr<config num>/config*
- */sys/class/cxl/<afu>m/mmio_size*
- */sys/class/cxl/<afu>m/pp_mmio_len*
- */sys/class/cxl/<afu>m/pp_mmio_off*
- */sys/class/cxl/<card>/caia_version*
- */sys/class/cxl/<card>/psl_revision*
- */sys/class/cxl/<card>/base_image*
- */sys/class/cxl/<card>/image_loaded*
- */sys/class/cxl/<card>/load_image_on_perst*
- */sys/class/cxl/<card>/reset*
- */sys/class/cxl/<card>/perst_reloads_same_image*
- */sys/class/cxl/<card>/psl_timebase_synced*
- */sys/class/cxl/<card>/tunneled_ops_supported*

5.2.280 File testing/sysfs-class-devfreq

Has the following ABI:

- */sys/class/devfreq/.../*
- */sys/class/devfreq/.../name*
- */sys/class/devfreq/.../governor*
- */sys/class/devfreq/.../cur_freq*
- */sys/class/devfreq/.../target_freq*
- */sys/class/devfreq/.../trans_stat*
- */sys/class/devfreq/.../available_frequencies*
- */sys/class/devfreq/.../available_governors*
- */sys/class/devfreq/.../min_freq*
- */sys/class/devfreq/.../max_freq*
- */sys/class/devfreq/.../polling_interval*
- */sys/class/devfreq/.../userspace/set_freq*

- */sys/class/devfreq/.../timer*

5.2.281 File testing/sysfs-class-devfreq-event

Has the following ABI:

- */sys/class/devfreq-event/event<x>/*
- */sys/class/devfreq-event/event<x>/name*
- */sys/class/devfreq-event/event<x>/enable_count*

5.2.282 File testing/sysfs-class-devlink

Has the following ABI:

- */sys/class/devlink/.../*
- */sys/class/devlink/.../auto_remove_on*
- */sys/class/devlink/.../consumer*
- */sys/class/devlink/.../runtime_pm*
- */sys/class/devlink/.../status*
- */sys/class/devlink/.../supplier*
- */sys/class/devlink/.../sync_state_only*

5.2.283 File testing/sysfs-class-extcon

Has the following ABI:

- */sys/class/extcon/.../*
- */sys/class/extcon/.../name*
- */sys/class/extcon/.../state*
- */sys/class/extcon/.../cable.X/name*
- */sys/class/extcon/.../cable.X/state*
- */sys/class/extcon/.../mutually_exclusive/..*

5.2.284 File testing/sysfs-class-fc

Has the following ABI:

- */sys/class/fc/fc_udev_device/appid_store*

5.2.285 File testing/sysfs-class-fc_host

Has the following ABI:

- */sys/class/fc_host/hostX/statistics/fpin_cn_yyy*
- */sys/class/fc_host/hostX/statistics/fpin_li_yyy*
- */sys/class/fc_host/hostX/statistics/fpin_dn_yyy*

5.2.286 File testing/sysfs-class-fc_remote_ports

Has the following ABI:

- */sys/class/fc_remote_ports/rport-X:Y-Z/statistics/fpin_cn_yyy*
- */sys/class/fc_remote_ports/rport-X:Y-Z/statistics/fpin_li_yyy*
- */sys/class/fc_remote_ports/rport-X:Y-Z/statistics/fpin_dn_yyy*

5.2.287 File testing/sysfs-class-firmware

Has the following ABI:

- */sys/class/firmware/.../data*
- */sys/class/firmware/.../cancel*
- */sys/class/firmware/.../error*
- */sys/class/firmware/.../loading*
- */sys/class/firmware/.../remaining_size*
- */sys/class/firmware/.../status*
- */sys/class/firmware/.../timeout*

5.2.288 File testing/sysfs-class-firmware-attributes

Has the following ABI:

- */sys/class/firmware-attributes/*/attributes/*/*
- */sys/class/firmware-attributes/*/authentication/*
- */sys/class/firmware-attributes/*/attributes/pending_reboot*
- */sys/class/firmware-attributes/*/attributes/reset_bios*
- */sys/class/firmware-attributes/*/attributes/save_settings*
- */sys/class/firmware-attributes/*/attributes/debug_cmd*
- */sys/class/firmware-attributes/*/authentication/SPM/kek*
- */sys/class/firmware-attributes/*/authentication/SPM/sk*
- */sys/class/firmware-attributes/*/authentication/SPM/status*
- */sys/class/firmware-attributes/*/attributes/Sure_Start/audit_log_entries*

- */sys/class/firmware-attributes/*/attributes/Sure_Start/audit_log_entry_count*

5.2.289 File testing/sysfs-class-fpga-bridge

Has the following ABI:

- */sys/class/fpga_bridge/<bridge>/name*
- */sys/class/fpga_bridge/<bridge>/state*

5.2.290 File testing/sysfs-class-fpga-manager

Has the following ABI:

- */sys/class/fpga_manager/<fpga>/name*
- */sys/class/fpga_manager/<fpga>/state*
- */sys/class/fpga_manager/<fpga>/status*

5.2.291 File testing/sysfs-class-fpga-region

Has the following ABI:

- */sys/class/fpga_region/<region>/compat_id*

5.2.292 File testing/sysfs-class-gnss

Has the following ABI:

- */sys/class/gnss/gnss<N>/type*

5.2.293 File testing/sysfs-class-hwmon

Has the following ABI:

- */sys/class/hwmon/hwmonX/name*
- */sys/class/hwmon/hwmonX/label*
- */sys/class/hwmon/hwmonX/update_interval*
- */sys/class/hwmon/hwmonX/inY_min*
- */sys/class/hwmon/hwmonX/inY_lcrit*
- */sys/class/hwmon/hwmonX/inY_max*
- */sys/class/hwmon/hwmonX/inY_crit*
- */sys/class/hwmon/hwmonX/inY_input*
- */sys/class/hwmon/hwmonX/inY_average*
- */sys/class/hwmon/hwmonX/inY_lowest*
- */sys/class/hwmon/hwmonX/inY_highest*

- */sys/class/hwmon/hwmonX/inY_reset_history*
- */sys/class/hwmon/hwmonX/in_reset_history*
- */sys/class/hwmon/hwmonX/inY_label*
- */sys/class/hwmon/hwmonX/inY_enable*
- */sys/class/hwmon/hwmonX/cpuY_vid*
- */sys/class/hwmon/hwmonX/vrm*
- */sys/class/hwmon/hwmonX/inY_rated_min*
- */sys/class/hwmon/hwmonX/inY_rated_max*
- */sys/class/hwmon/hwmonX/fanY_min*
- */sys/class/hwmon/hwmonX/fanY_max*
- */sys/class/hwmon/hwmonX/fanY_input*
- */sys/class/hwmon/hwmonX/fanY_div*
- */sys/class/hwmon/hwmonX/fanY_pulses*
- */sys/class/hwmon/hwmonX/fanY_target*
- */sys/class/hwmon/hwmonX/fanY_label*
- */sys/class/hwmon/hwmonX/fanY_enable*
- */sys/class/hwmon/hwmonX/fanY_fault*
- */sys/class/hwmon/hwmonX/pwmY*
- */sys/class/hwmon/hwmonX/pwmY_enable*
- */sys/class/hwmon/hwmonX/pwmY_mode*
- */sys/class/hwmon/hwmonX/pwmY_freq*
- */sys/class/hwmon/hwmonX/pwmY_auto_channels_temp*
- */sys/class/hwmon/hwmonX/pwmY_auto_pointZ_pwm*
- */sys/class/hwmon/hwmonX/pwmY_auto_pointZ_temp*
- */sys/class/hwmon/hwmonX/pwmY_auto_pointZ_temp_hyst*
- */sys/class/hwmon/hwmonX/tempY_auto_pointZ_pwm*
- */sys/class/hwmon/hwmonX/tempY_auto_pointZ_temp*
- */sys/class/hwmon/hwmonX/tempY_auto_pointZ_temp_hyst*
- */sys/class/hwmon/hwmonX/tempY_type*
- */sys/class/hwmon/hwmonX/tempY_max*
- */sys/class/hwmon/hwmonX/tempY_max_alarm*
- */sys/class/hwmon/hwmonX/tempY_min*
- */sys/class/hwmon/hwmonX/tempY_min_alarm*
- */sys/class/hwmon/hwmonX/tempY_max_hyst*

- */sys/class/hwmon/hwmonX/tempY_min_hyst*
- */sys/class/hwmon/hwmonX/tempY_input*
- */sys/class/hwmon/hwmonX/tempY_crit*
- */sys/class/hwmon/hwmonX/tempY_crit_alarm*
- */sys/class/hwmon/hwmonX/tempY_crit_hyst*
- */sys/class/hwmon/hwmonX/tempY_emergency*
- */sys/class/hwmon/hwmonX/tempY_emergency_alarm*
- */sys/class/hwmon/hwmonX/tempY_emergency_hyst*
- */sys/class/hwmon/hwmonX/tempY_lcrit*
- */sys/class/hwmon/hwmonX/tempY_lcrit_hyst*
- */sys/class/hwmon/hwmonX/tempY_offset*
- */sys/class/hwmon/hwmonX/tempY_label*
- */sys/class/hwmon/hwmonX/tempY_lowest*
- */sys/class/hwmon/hwmonX/tempY_highest*
- */sys/class/hwmon/hwmonX/tempY_reset_history*
- */sys/class/hwmon/hwmonX/temp_reset_history*
- */sys/class/hwmon/hwmonX/tempY_enable*
- */sys/class/hwmon/hwmonX/tempY_rated_min*
- */sys/class/hwmon/hwmonX/tempY_rated_max*
- */sys/class/hwmon/hwmonX/currY_max*
- */sys/class/hwmon/hwmonX/currY_min*
- */sys/class/hwmon/hwmonX/currY_lcrit*
- */sys/class/hwmon/hwmonX/currY_crit*
- */sys/class/hwmon/hwmonX/currY_input*
- */sys/class/hwmon/hwmonX/currY_average*
- */sys/class/hwmon/hwmonX/currY_lowest*
- */sys/class/hwmon/hwmonX/currY_highest*
- */sys/class/hwmon/hwmonX/currY_reset_history*
- */sys/class/hwmon/hwmonX/curr_reset_history*
- */sys/class/hwmon/hwmonX/currY_enable*
- */sys/class/hwmon/hwmonX/currY_rated_min*
- */sys/class/hwmon/hwmonX/currY_rated_max*
- */sys/class/hwmon/hwmonX/powerY_average*
- */sys/class/hwmon/hwmonX/powerY_average_interval*

- */sys/class/hwmon/hwmonX/powerY_average_interval_max*
- */sys/class/hwmon/hwmonX/powerY_average_interval_min*
- */sys/class/hwmon/hwmonX/powerY_average_highest*
- */sys/class/hwmon/hwmonX/powerY_average_lowest*
- */sys/class/hwmon/hwmonX/powerY_average_max*
- */sys/class/hwmon/hwmonX/powerY_average_min*
- */sys/class/hwmon/hwmonX/powerY_input*
- */sys/class/hwmon/hwmonX/powerY_input_highest*
- */sys/class/hwmon/hwmonX/powerY_input_lowest*
- */sys/class/hwmon/hwmonX/powerY_reset_history*
- */sys/class/hwmon/hwmonX/powerY_accuracy*
- */sys/class/hwmon/hwmonX/powerY_cap*
- */sys/class/hwmon/hwmonX/powerY_cap_hyst*
- */sys/class/hwmon/hwmonX/powerY_cap_max*
- */sys/class/hwmon/hwmonX/powerY_cap_min*
- */sys/class/hwmon/hwmonX/powerY_max*
- */sys/class/hwmon/hwmonX/powerY_crit*
- */sys/class/hwmon/hwmonX/powerY_enable*
- */sys/class/hwmon/hwmonX/powerY_rated_min*
- */sys/class/hwmon/hwmonX/powerY_rated_max*
- */sys/class/hwmon/hwmonX/energyY_input*
- */sys/class/hwmon/hwmonX/energyY_enable*
- */sys/class/hwmon/hwmonX/humidityY_alarm*
- */sys/class/hwmon/hwmonX/humidityY_enable*
- */sys/class/hwmon/hwmonX/humidityY_fault*
- */sys/class/hwmon/hwmonX/humidityY_input*
- */sys/class/hwmon/hwmonX/humidityY_label*
- */sys/class/hwmon/hwmonX/humidityY_max*
- */sys/class/hwmon/hwmonX/humidityY_max_hyst*
- */sys/class/hwmon/hwmonX/humidityY_min*
- */sys/class/hwmon/hwmonX/humidityY_min_hyst*
- */sys/class/hwmon/hwmonX/humidityY_rated_min*
- */sys/class/hwmon/hwmonX/humidityY_rated_max*
- */sys/class/hwmon/hwmonX/intrusionY_alarm*

- */sys/class/hwmon/hwmonX/intrusionY_beep*
- */sys/class/hwmon/hwmonX/device/pec*

5.2.294 File testing/sysfs-class-intel_pmt

Has the following ABI:

- */sys/class/intel_pmt/*
- */sys/class/intel_pmt/telem<x>*
- */sys/class/intel_pmt/telem<x>/telem*
- */sys/class/intel_pmt/telem<x>/guid*
- */sys/class/intel_pmt/telem<x>/size*
- */sys/class/intel_pmt/telem<x>/offset*
- */sys/class/intel_pmt/crashlog<x>*
- */sys/class/intel_pmt/crashlog<x>/crashlog*
- */sys/class/intel_pmt/crashlog<x>/guid*
- */sys/class/intel_pmt/crashlog<x>/size*
- */sys/class/intel_pmt/crashlog<x>/offset*
- */sys/class/intel_pmt/crashlog<x>/enable*
- */sys/class/intel_pmt/crashlog<x>/trigger*

5.2.295 File testing/sysfs-class-iommu

Has the following ABI:

- */sys/class/iommu/<iommu>/devices/*
- */sys/devices/.../iommu*

5.2.296 File testing/sysfs-class-iommu-amd-iommu

Has the following ABI:

- */sys/class/iommu/<iommu>/amd-iommu/cap*
- */sys/class/iommu/<iommu>/amd-iommu/features*

5.2.297 File testing/sysfs-class-iommu-intel-iommu

Has the following ABI:

- */sys/class/iommu/<iommu>/intel-iommu/address*
- */sys/class/iommu/<iommu>/intel-iommu/cap*
- */sys/class/iommu/<iommu>/intel-iommu/ecap*
- */sys/class/iommu/<iommu>/intel-iommu/version*

5.2.298 File testing/sysfs-class-lcd

Has the following ABI:

- */sys/class/lcd/<lcd>/lcd_power*
- */sys/class/lcd/<lcd>/contrast*
- */sys/class/lcd/<lcd>/max_contrast*

5.2.299 File testing/sysfs-class-led

Has the following ABI:

- */sys/class/leds/<led>/brightness*
- */sys/class/leds/<led>/max_brightness*
- */sys/class/leds/<led>/brightness_hw_changed*
- */sys/class/leds/<led>/trigger*
- */sys/class/leds/<led>/inverted*

5.2.300 File testing/sysfs-class-led-driver-aw200xx

Has the following ABI:

- */sys/class/leds/<led>/dim*

5.2.301 File testing/sysfs-class-led-driver-lm3533

Has the following ABI:

- */sys/class/leds/<led>/als_channel*
- */sys/class/leds/<led>/als_en*
- */sys/class/leds/<led>/falltime*
- */sys/class/leds/<led>/risetime*
- */sys/class/leds/<led>/id*
- */sys/class/leds/<led>/linear*
- */sys/class/leds/<led>/pwm*

5.2.302 File testing/sysfs-class-led-driver-turris-omnia

Has the following ABI:

- */sys/class/leds/<led>/device/brightness*
- */sys/class/leds/<led>/device/gamma_correction*

5.2.303 File testing/sysfs-class-led-flash

Has the following ABI:

- */sys/class/leds/<led>/flash_brightness*
- */sys/class/leds/<led>/max_flash_brightness*
- */sys/class/leds/<led>/flash_timeout*
- */sys/class/leds/<led>/max_flash_timeout*
- */sys/class/leds/<led>/flash_strobe*
- */sys/class/leds/<led>/flash_fault*

5.2.304 File testing/sysfs-class-led-multicolor

Has the following ABI:

- */sys/class/leds/<led>/multi_index*
- */sys/class/leds/<led>/multi_intensity*

5.2.305 File testing/sysfs-class-led-trigger-netdev

Has the following ABI:

- */sys/class/leds/<led>/device_name*
- */sys/class/leds/<led>/interval*
- */sys/class/leds/<led>/link*
- */sys/class/leds/<led>/tx*
- */sys/class/leds/<led>/rx*
- */sys/class/leds/<led>/offloaded*
- */sys/class/leds/<led>/link_10*
- */sys/class/leds/<led>/link_100*
- */sys/class/leds/<led>/link_1000*
- */sys/class/leds/<led>/link_2500*
- */sys/class/leds/<led>/link_5000*
- */sys/class/leds/<led>/link_10000*
- */sys/class/leds/<led>/half_duplex*

- */sys/class/leds/<led>/full_duplex*

5.2.306 File testing/sysfs-class-led-trigger-oneshot

Has the following ABI:

- */sys/class/leds/<led>/delay_on*
- */sys/class/leds/<led>/delay_off*
- */sys/class/leds/<led>/invert*
- */sys/class/leds/<led>/shot*

5.2.307 File testing/sysfs-class-led-trigger-pattern

Has the following ABI:

- */sys/class/leds/<led>/pattern*
- */sys/class/leds/<led>/hw_pattern*
- */sys/class/leds/<led>/repeat*

5.2.308 File testing/sysfs-class-led-trigger-tty

Has the following ABI:

- */sys/class/leds/<led>/ttyname*
- */sys/class/leds/<led>/rx*
- */sys/class/leds/<led>/tx*
- */sys/class/leds/<led>/cts*
- */sys/class/leds/<led>/dsr*
- */sys/class/leds/<led>/dcd*
- */sys/class/leds/<led>/rng*

5.2.309 File testing/sysfs-class-led-trigger-usbport

Has the following ABI:

- */sys/class/leds/<led>/ports/<port>*

5.2.310 File testing/sysfs-class-leds-gt683r

Has the following ABI:

- */sys/class/leds/<led>/gt683r/mode*

5.2.311 File testing/sysfs-class-mei

Has the following ABI:

- */sys/class/mei/*
- */sys/class/mei/mei<N>/*
- */sys/class/mei/mei<N>/fw_status*
- */sys/class/mei/mei<N>/hbm_ver*
- */sys/class/mei/mei<N>/hbm_ver_drv*
- */sys/class/mei/mei<N>/tx_queue_limit*
- */sys/class/mei/mei<N>/fw_ver*
- */sys/class/mei/mei<N>/dev_state*
- */sys/class/mei/mei<N>/trc*
- */sys/class/mei/mei<N>/kind*

5.2.312 File testing/sysfs-class-mic

Has the following ABI:

- */sys/class/mic/*
- */sys/class/mic/mic<X>*
- */sys/class/mic/mic<X>/family*
- */sys/class/mic/mic<X>/stepping*
- */sys/class/mic/mic<X>/state*
- */sys/class/mic/mic<X>/shutdown_status*
- */sys/class/mic/mic<X>/cmdline*
- */sys/class/mic/mic<X>/firmware*
- */sys/class/mic/mic<X>/ramdisk*
- */sys/class/mic/mic<X>/bootmode*
- */sys/class/mic/mic<X>/log_buf_addr*
- */sys/class/mic/mic<X>/log_buf_len*
- */sys/class/mic/mic<X>/heartbeat_enable*

5.2.313 File testing/sysfs-class-mtd

Has the following ABI:

- */sys/class/mtd/*
- */sys/class/mtd/mtdX/*
- */sys/class/mtd/mtdXro/*
- */sys/class/mtd/mtdX/dev*
- */sys/class/mtd/mtdXro/dev*
- */sys/class/mtd/mtdX/erasesize*
- */sys/class/mtd/mtdX/flags*
- */sys/class/mtd/mtdX/name*
- */sys/class/mtd/mtdX/numeraseregions*
- */sys/class/mtd/mtdX/oobsize*
- */sys/class/mtd/mtdX/size*
- */sys/class/mtd/mtdX/type*
- */sys/class/mtd/mtdX/writesize*
- */sys/class/mtd/mtdX/ecc_strength*
- */sys/class/mtd/mtdX/bitflip_threshold*
- */sys/class/mtd/mtdX/ecc_step_size*
- */sys/class/mtd/mtdX/ecc_failures*
- */sys/class/mtd/mtdX/corrected_bits*
- */sys/class/mtd/mtdX/bad_blocks*
- */sys/class/mtd/mtdX/bbt_blocks*
- */sys/class/mtd/mtdX/offset*
- */sys/class/mtd/mtdX/oobavail*

5.2.314 File testing/sysfs-class-mux

Has the following ABI:

- */sys/class/mux/*
- */sys/class/mux/muxchip<N>/*

5.2.315 File testing/sysfs-class-net

Has the following ABI:

- */sys/class/net/<iface>/name_assign_type*
- */sys/class/net/<iface>/addr_assign_type*
- */sys/class/net/<iface>/addr_len*
- */sys/class/net/<iface>/address*
- */sys/class/net/<bridge iface>/bridge/group_fwd_mask*
- */sys/class/net/<iface>/broadcast*
- */sys/class/net/<iface>/carrier*
- */sys/class/net/<iface>/dev_id*
- */sys/class/net/<iface>/dev_port*
- */sys/class/net/<iface>/dormant*
- */sys/class/net/<iface>/testing*
- */sys/class/net/<iface>/duplex*
- */sys/class/net/<iface>/flags*
- */sys/class/net/<iface>/ifalias*
- */sys/class/net/<iface>/ifindex*
- */sys/class/net/<iface>/iflink*
- */sys/class/net/<iface>/link_mode*
- */sys/class/net/<iface>/mtu*
- */sys/class/net/<iface>/netdev_group*
- */sys/class/net/<iface>/operstate*
- */sys/class/net/<iface>/phys_port_id*
- */sys/class/net/<iface>/phys_port_name*
- */sys/class/net/<iface>/speed*
- */sys/class/net/<iface>/tx_queue_len*
- */sys/class/net/<iface>/type*
- */sys/class/net/<iface>/phys_switch_id*
- */sys/class/net/<iface>/phydev*
- */sys/class/net/<iface>/carrier_changes*
- */sys/class/net/<iface>/carrier_up_count*
- */sys/class/net/<iface>/carrier_down_count*
- */sys/class/net/<iface>/threaded*

5.2.316 File testing/sysfs-class-net-cdc_ncm

Has the following ABI:

- */sys/class/net/<iface>/cdc_ncm/min_tx_pkt*
- */sys/class/net/<iface>/cdc_ncm/ndp_to_end*
- */sys/class/net/<iface>/cdc_ncm/rx_max*
- */sys/class/net/<iface>/cdc_ncm/tx_max*
- */sys/class/net/<iface>/cdc_ncm/tx_timer_usecs*
- */sys/class/net/<iface>/cdc_ncm/bmNtbFormatsSupported*
- */sys/class/net/<iface>/cdc_ncm/dwNtbInMaxSize*
- */sys/class/net/<iface>/cdc_ncm/wNdpInDivisor*
- */sys/class/net/<iface>/cdc_ncm/wNdpInPayloadRemainder*
- */sys/class/net/<iface>/cdc_ncm/wNdpInAlignment*
- */sys/class/net/<iface>/cdc_ncm/dwNtbOutMaxSize*
- */sys/class/net/<iface>/cdc_ncm/wNdpOutDivisor*
- */sys/class/net/<iface>/cdc_ncm/wNdpOutPayloadRemainder*
- */sys/class/net/<iface>/cdc_ncm/wNdpOutAlignment*
- */sys/class/net/<iface>/cdc_ncm/wNtbOutMaxDatagrams*

5.2.317 File testing/sysfs-class-net-dsa

Has the following ABI:

- */sys/class/net/<iface>/dsa/tagging*

5.2.318 File testing/sysfs-class-net-grcan

Has the following ABI:

- */sys/class/net/<iface>/grcan/enable0*
- */sys/class/net/<iface>/grcan/enable1*
- */sys/class/net/<iface>/grcan/select*

5.2.319 File testing/sysfs-class-net-janz-ican3

Has the following ABI:

- */sys/class/net/<iface>/termination*
- */sys/class/net/<iface>/fwinfo*

5.2.320 File testing/sysfs-class-net-peak_usb

Has the following ABI:

- */sys/class/net/<iface>/peak_usb/can_channel_id*

5.2.321 File testing/sysfs-class-net-phydev

Has the following ABI:

- */sys/class/mdio_bus/<bus>/<device>/attached_dev*
- */sys/class/mdio_bus/<bus>/<device>/phy_has_fixups*
- */sys/class/mdio_bus/<bus>/<device>/phy_id*
- */sys/class/mdio_bus/<bus>/<device>/phy_interface*
- */sys/class/mdio_bus/<bus>/<device>/phy_standalone*
- */sys/class/mdio_bus/<bus>/<device>/phy_dev_flags*

5.2.322 File testing/sysfs-class-net-qmi

Has the following ABI:

- */sys/class/net/<iface>/qmi/raw_ip*
- */sys/class/net/<iface>/qmi/add_mux*
- */sys/class/net/<iface>/qmi/del_mux*
- */sys/class/net/<qmimux iface>/qmap/mux_id*
- */sys/class/net/<iface>/qmi/pass_through*

5.2.323 File testing/sysfs-class-net-queues

Has the following ABI:

- */sys/class/net/<iface>/queues/rx-<queue>/rps_cpus*
- */sys/class/net/<iface>/queues/rx-<queue>/rps_flow_cnt*
- */sys/class/net/<iface>/queues/tx-<queue>/tx_timeout*
- */sys/class/net/<iface>/queues/tx-<queue>/tx_maxrate*
- */sys/class/net/<iface>/queues/tx-<queue>/xps_cpus*
- */sys/class/net/<iface>/queues/tx-<queue>/xps_rxqs*

- */sys/class/net/<iface>/queues/tx-<queue>/byte_queue_limits/hold_time*
- */sys/class/net/<iface>/queues/tx-<queue>/byte_queue_limits/inflight*
- */sys/class/net/<iface>/queues/tx-<queue>/byte_queue_limits/limit*
- */sys/class/net/<iface>/queues/tx-<queue>/byte_queue_limits/limit_max*
- */sys/class/net/<iface>/queues/tx-<queue>/byte_queue_limits/limit_min*

5.2.324 File testing/sysfs-class-net-statistics

Has the following ABI:

- */sys/class/net/<iface>/statistics/collisions*
- */sys/class/net/<iface>/statistics/multicast*
- */sys/class/net/<iface>/statistics/rx_bytes*
- */sys/class/net/<iface>/statistics/rx_compressed*
- */sys/class/net/<iface>/statistics/rx_crc_errors*
- */sys/class/net/<iface>/statistics/rx_dropped*
- */sys/class/net/<iface>/statistics/rx_errors*
- */sys/class/net/<iface>/statistics/rx_fifo_errors*
- */sys/class/net/<iface>/statistics/rx_frame_errors*
- */sys/class/net/<iface>/statistics/rx_length_errors*
- */sys/class/net/<iface>/statistics/rx_missed_errors*
- */sys/class/net/<iface>/statistics/rx_nohandler*
- */sys/class/net/<iface>/statistics/rx_over_errors*
- */sys/class/net/<iface>/statistics/rx_packets*
- */sys/class/net/<iface>/statistics/tx_aborted_errors*
- */sys/class/net/<iface>/statistics/tx_bytes*
- */sys/class/net/<iface>/statistics/tx_carrier_errors*
- */sys/class/net/<iface>/statistics/tx_compressed*
- */sys/class/net/<iface>/statistics/tx_dropped*
- */sys/class/net/<iface>/statistics/tx_errors*
- */sys/class/net/<iface>/statistics/tx_fifo_errors*
- */sys/class/net/<iface>/statistics/tx_heartbeat_errors*
- */sys/class/net/<iface>/statistics/tx_packets*
- */sys/class/net/<iface>/statistics/tx_window_errors*

5.2.325 File testing/sysfs-class-ocxl

Has the following ABI:

- */sys/class/ocxl/<afu name>/afu_version*
- */sys/class/ocxl/<afu name>/contexts*
- */sys/class/ocxl/<afu name>/pp_mmio_size*
- */sys/class/ocxl/<afu name>/global_mmio_size*
- */sys/class/ocxl/<afu name>/global_mmio_area*
- */sys/class/ocxl/<afu name>/reload_on_reset*

5.2.326 File testing/sysfs-class-pktcdvd

sysfs interface

The pktcdvd module (packet writing driver) creates the following files in the sysfs: (<devid> is in the format major:minor)

Has the following ABI:

- */sys/class/pktcdvd/add*
- */sys/class/pktcdvd/remove*
- */sys/class/pktcdvd/device_map*
- */sys/class/pktcdvd/pktcdvd[0-7]/dev*
- */sys/class/pktcdvd/pktcdvd[0-7]/uevent*
- */sys/class/pktcdvd/pktcdvd[0-7]/stat/packets_started*
- */sys/class/pktcdvd/pktcdvd[0-7]/stat/packets_finished*
- */sys/class/pktcdvd/pktcdvd[0-7]/stat/kb_written*
- */sys/class/pktcdvd/pktcdvd[0-7]/stat/kb_read*
- */sys/class/pktcdvd/pktcdvd[0-7]/stat/kb_read_gather*
- */sys/class/pktcdvd/pktcdvd[0-7]/stat/reset*
- */sys/class/pktcdvd/pktcdvd[0-7]/write_queue/size*
- */sys/class/pktcdvd/pktcdvd[0-7]/write_queue/congestion_off*
- */sys/class/pktcdvd/pktcdvd[0-7]/write_queue/congestion_on*

5.2.327 File testing/sysfs-class-power

General Properties

Has the following ABI:

- */sys/class/power_supply/<supply_name>/manufacturer*
- */sys/class/power_supply/<supply_name>/model_name*
- */sys/class/power_supply/<supply_name>/serial_number*
- */sys/class/power_supply/<supply_name>/type*
- */sys/class/power_supply/<supply_name>/current_avg*
- */sys/class/power_supply/<supply_name>/current_max*
- */sys/class/power_supply/<supply_name>/current_now*
- */sys/class/power_supply/<supply_name>/temp*
- */sys/class/power_supply/<supply_name>/temp_alert_max*
- */sys/class/power_supply/<supply_name>/temp_alert_min*
- */sys/class/power_supply/<supply_name>/temp_max*
- */sys/class/power_supply/<supply_name>/temp_min*
- */sys/class/power_supply/<supply_name>/voltage_max*
- */sys/class/power_supply/<supply_name>/voltage_min*
- */sys/class/power_supply/<supply_name>/voltage_now*
- */sys/class/power_supply/<supply_name>/capacity*
- */sys/class/power_supply/<supply_name>/capacity_alert_max*
- */sys/class/power_supply/<supply_name>/capacity_alert_min*
- */sys/class/power_supply/<supply_name>/capacity_error_margin*
- */sys/class/power_supply/<supply_name>/capacity_level*
- */sys/class/power_supply/<supply_name>/charge_control_limit*
- */sys/class/power_supply/<supply_name>/charge_control_limit_max*
- */sys/class/power_supply/<supply_name>/charge_control_start_threshold*
- */sys/class/power_supply/<supply_name>/charge_control_end_threshold*
- */sys/class/power_supply/<supply_name>/charge_type*
- */sys/class/power_supply/<supply_name>/charge_term_current*
- */sys/class/power_supply/<supply_name>/health*
- */sys/class/power_supply/<supply_name>/precharge_current*
- */sys/class/power_supply/<supply_name>/present*
- */sys/class/power_supply/<supply_name>/status*
- */sys/class/power_supply/<supply_name>/charge_behaviour*

- */sys/class/power_supply/<supply_name>/technology*
- */sys/class/power_supply/<supply_name>/voltage_avg*
- */sys/class/power_supply/<supply_name>/cycle_count*
- */sys/class/power_supply/<supply_name>/input_current_limit*
- */sys/class/power_supply/<supply_name>/input_voltage_limit*
- */sys/class/power_supply/<supply_name>/input_power_limit*
- */sys/class/power_supply/<supply_name>/online*
- */sys/class/power_supply/<supply_name>/usb_type*
- */sys/class/power/ds2760-battery.*/charge_now*
- */sys/class/power/ds2760-battery.*/charge_full*
- */sys/class/power_supply/max14577-charger/device/fast_charge_timer*
- */sys/class/power_supply/max77693-charger/device/fast_charge_timer*
- */sys/class/power_supply/max77693-charger/device/top_off_threshold_current*
- */sys/class/power_supply/max77693-charger/device/top_off_timer*
- */sys/class/power_supply/bq24257-charger/ovp_voltage*
- */sys/class/power_supply/bq24257-charger/in_dpm_voltage*
- */sys/class/power_supply/bq24257-charger/high_impedance_enable*
- */sys/class/power_supply/bq24257-charger/sysoff_enable*
- */sys/class/power_supply/<supply_name>/manufacture_year*
- */sys/class/power_supply/<supply_name>/manufacture_month*
- */sys/class/power_supply/<supply_name>/manufacture_day*

5.2.328 File testing/sysfs-class-power-ltc4162l

Has the following ABI:

- */sys/class/power_supply/ltc4162-l/charge_status*
- */sys/class/power_supply/ltc4162-l/ibat*
- */sys/class/power_supply/ltc4162-l/vbat*
- */sys/class/power_supply/ltc4162-l/vbat_avg*
- */sys/class/power_supply/ltc4162-l/force_telemetry*
- */sys/class/power_supply/ltc4162-l/arm_ship_mode*

5.2.329 File testing/sysfs-class-power-mp2629

Has the following ABI:

- */sys/class/power_supply/mp2629_battery/batt_impedance_compen*

5.2.330 File testing/sysfs-class-power-rt9467

Has the following ABI:

- */sys/class/power_supply/rt9467-*/sysoff_enable*

5.2.331 File testing/sysfs-class-power-rt9471

Has the following ABI:

- */sys/class/power_supply/rt9471-*/sysoff_enable*
- */sys/class/power_supply/rt9471-*/port_detect_enable*

5.2.332 File testing/sysfs-class-power-surface

Has the following ABI:

- */sys/class/power_supply/<supply_name>/alarm*

5.2.333 File testing/sysfs-class-power-twl4030

Has the following ABI:

- */sys/class/power_supply/twl4030_usb/mode*
- */sys/class/power_supply/twl4030_ac/mode*

5.2.334 File testing/sysfs-class-power-wilco

Has the following ABI:

- */sys/class/power_supply/wilco-charger/charge_type*
- */sys/class/power_supply/wilco-charger/charge_control_start_threshold*
- */sys/class/power_supply/wilco-charger/charge_control_end_threshold*

5.2.335 File testing/sysfs-class-powercap

Has the following ABI:

- */sys/class/powercap/*
- */sys/class/powercap/<control type>*
- */sys/class/powercap/<control type>/enabled*
- */sys/class/powercap/<control type>/<power zone>*
- */sys/class/powercap/<control type>/<power zone>/<child power zone>*
- */sys/class/powercap/.../<power zone>/name*
- */sys/class/powercap/.../<power zone>/energy_uj*
- */sys/class/powercap/.../<power zone>/max_energy_range_uj*
- */sys/class/powercap/.../<power zone>/power_uw*
- */sys/class/powercap/.../<power zone>/max_power_range_uw*
- */sys/class/powercap/.../<power zone>/constraint_X_name*
- */sys/class/powercap/.../<power zone>/constraint_X_power_limit_uw*
- */sys/class/powercap/.../<power zone>/constraint_X_time_window_us*
- */sys/class/powercap/<control type>/.../constraint_X_max_power_uw*
- */sys/class/powercap/<control type>/.../constraint_X_min_power_uw*
- */sys/class/powercap/.../<power zone>/constraint_X_max_time_window_us*
- */sys/class/powercap/.../<power zone>/constraint_X_min_time_window_us*
- */sys/class/powercap/.../<power zone>/enabled*

5.2.336 File testing/sysfs-class-pwm

Has the following ABI:

- */sys/class/pwm/*
- */sys/class/pwm/pwmchip<N>/*
- */sys/class/pwm/pwmchip<N>/npwm*
- */sys/class/pwm/pwmchip<N>/export*
- */sys/class/pwm/pwmchip<N>/unexport*
- */sys/class/pwm/pwmchip<N>/pwmX*
- */sys/class/pwm/pwmchip<N>/pwmX/period*
- */sys/class/pwm/pwmchip<N>/pwmX/duty_cycle*
- */sys/class/pwm/pwmchip<N>/pwmX/polarity*
- */sys/class/pwm/pwmchip<N>/pwmX/enable*
- */sys/class/pwm/pwmchip<N>/pwmX/capture*

5.2.337 File testing/sysfs-class-rapidio

Has the following ABI:

- */sys/class/rapidio_port*
- */sys/class/rapidio_port/rapidio<N>/sys_size*
- */sys/class/rapidio_port/rapidio<N>/port_destid*

5.2.338 File testing/sysfs-class-rc

Has the following ABI:

- */sys/class/rc/*
- */sys/class/rc/rc<N>/*
- */sys/class/rc/rc<N>/protocols*
- */sys/class/rc/rc<N>/filter*
- */sys/class/rc/rc<N>/filter_mask*
- */sys/class/rc/rc<N>/wakeup_protocols*
- */sys/class/rc/rc<N>/wakeup_filter*
- */sys/class/rc/rc<N>/wakeup_filter_mask*

5.2.339 File testing/sysfs-class-rc-nuvoton

Has the following ABI:

- */sys/class/rc/rc<N>/wakeup_data*

5.2.340 File testing/sysfs-class-regulator

Has the following ABI:

- */sys/class/regulator/.../state*
- */sys/class/regulator/.../status*
- */sys/class/regulator/.../type*
- */sys/class/regulator/.../microvolts*
- */sys/class/regulator/.../microamps*
- */sys/class/regulator/.../opmode*
- */sys/class/regulator/.../min_microvolts*
- */sys/class/regulator/.../max_microvolts*
- */sys/class/regulator/.../min_microamps*
- */sys/class/regulator/.../max_microamps*
- */sys/class/regulator/.../name*

- */sys/class/regulator/.../num_users*
- */sys/class/regulator/.../requested_microamps*
- */sys/class/regulator/.../parent*
- */sys/class/regulator/.../suspend_mem_microvolts*
- */sys/class/regulator/.../suspend_disk_microvolts*
- */sys/class/regulator/.../suspend_standby_microvolts*
- */sys/class/regulator/.../suspend_mem_mode*
- */sys/class/regulator/.../suspend_disk_mode*
- */sys/class/regulator/.../suspend_standby_mode*
- */sys/class/regulator/.../suspend_mem_state*
- */sys/class/regulator/.../suspend_disk_state*
- */sys/class/regulator/.../suspend_standby_state*
- */sys/class/regulator/.../bypass*
- */sys/class/regulator/.../under_voltage*
- */sys/class/regulator/.../over_current*
- */sys/class/regulator/.../regulation_out*
- */sys/class/regulator/.../fail*
- */sys/class/regulator/.../over_temp*
- */sys/class/regulator/.../under_voltage_warn*
- */sys/class/regulator/.../over_current_warn*
- */sys/class/regulator/.../over_voltage_warn*
- */sys/class/regulator/.../over_temp_warn*

5.2.341 File testing/sysfs-class-remoteproc

Has the following ABI:

- */sys/class/remoteproc/.../firmware*
- */sys/class/remoteproc/.../state*
- */sys/class/remoteproc/.../name*
- */sys/class/remoteproc/.../coredump*
- */sys/class/remoteproc/.../recovery*

5.2.342 File testing/sysfs-class-rnbd-client

Has the following ABI:

- */sys/class/rnbd-client*
- */sys/class/rnbd-client/ctl/map_device*
- */sys/class/rnbd-client/ctl/devices/*

5.2.343 File testing/sysfs-class-rnbd-server

Has the following ABI:

- */sys/class/rnbd-server*
- */sys/class/rnbd-server/ctl/*
- */sys/class/rnbd-server/ctl/devices/<device_name>/block_dev*
- */sys/class/rnbd-server/ctl/devices/<device_name>/sessions/*
- */sys/class/rnbd-server/ctl/devices/<device_name>/sessions/<session-name>/read_only*
- */sys/class/rnbd-server/ctl/devices/<device_name>/sessions/<session-name>/mapping_path*
- */sys/class/rnbd-server/ctl/devices/<device_name>/sessions/<session-name>/access_mode*
- */sys/class/rnbd-server/ctl/devices/<device_name>/sessions/<session-name>/force_close*

5.2.344 File testing/sysfs-class-rtc

Has the following ABI:

- */sys/class/rtc/*
- */sys/class/rtc/rtcX/*
- */sys/class/rtc/rtcX/date*
- */sys/class/rtc/rtcX/hctosys*
- */sys/class/rtc/rtcX/max_user_freq*
- */sys/class/rtc/rtcX/name*
- */sys/class/rtc/rtcX/range*
- */sys/class/rtc/rtcX/since_epoch*
- */sys/class/rtc/rtcX/time*
- */sys/class/rtc/rtcX/offset*
- */sys/class/rtc/rtcX/wakealarm*

5.2.345 File testing/sysfs-class-rtc-rtc0-device-rtc_calibration

Has the following ABI:

- */sys/class/rtc/rtc0/device/rtc_calibration*

5.2.346 File testing/sysfs-class-rtrs-client

Has the following ABI:

- */sys/class/rtrs-client*
- */sys/class/rtrs-client/<session-name>/add_path*
- */sys/class/rtrs-client/<session-name>/max_reconnect_attempts*
- */sys/class/rtrs-client/<session-name>/mp_policy*
- */sys/class/rtrs-client/<session-name>/paths/*
- */sys/class/rtrs-client/<session-name>/paths/<src@dst>/state*
- */sys/class/rtrs-client/<session-name>/paths/<src@dst>/reconnect*
- */sys/class/rtrs-client/<session-name>/paths/<src@dst>/disconnect*
- */sys/class/rtrs-client/<session-name>/paths/<src@dst>/remove_path*
- */sys/class/rtrs-client/<session-name>/paths/<src@dst>/hca_name*
- */sys/class/rtrs-client/<session-name>/paths/<src@dst>/hca_port*
- */sys/class/rtrs-client/<session-name>/paths/<src@dst>/src_addr*
- */sys/class/rtrs-client/<session-name>/paths/<src@dst>/dst_addr*
- */sys/class/rtrs-client/<session-name>/paths/<src@dst>/cur_latency*
- */sys/class/rtrs-client/<session-name>/paths/<src@dst>/stats/reset_all*
- */sys/class/rtrs-client/<session-name>/paths/<src@dst>/stats/cpu_migration*
- */sys/class/rtrs-client/<session-name>/paths/<src@dst>/stats/reconnects*
- */sys/class/rtrs-client/<session-name>/paths/<src@dst>/stats/rdma*

5.2.347 File testing/sysfs-class-rtrs-server

Has the following ABI:

- */sys/class/rtrs-server*
- */sys/class/rtrs-server/<session-name>/paths/*
- */sys/class/rtrs-server/<session-name>/paths/<src@dst>/disconnect*
- */sys/class/rtrs-server/<session-name>/paths/<src@dst>/hca_name*
- */sys/class/rtrs-server/<session-name>/paths/<src@dst>/hca_port*
- */sys/class/rtrs-server/<session-name>/paths/<src@dst>/src_addr*
- */sys/class/rtrs-server/<session-name>/paths/<src@dst>/dst_addr*

- */sys/class/rtrs-server/<session-name>/paths/<src@dst>/stats/rdma*

5.2.348 File testing/sysfs-class-scsi_host

Has the following ABI:

- */sys/class/scsi_host/hostX/isci_id*
- */sys/class/scsi_host/hostX/acciopath_status*
- */sys/class/scsi_host/hostX/link_power_management_policy*
- */sys/class/scsi_host/hostX/em_message*
- */sys/class/scsi_host/hostX/em_message_type*
- */sys/class/scsi_host/hostX/ahci_port_cmd*
- */sys/class/scsi_host/hostX/ahci_host_caps*
- */sys/class/scsi_host/hostX/ahci_host_cap2*
- */sys/class/scsi_host/hostX/ahci_host_version*
- */sys/class/scsi_host/hostX/em_buffer*
- */sys/class/scsi_host/hostX/em_message_supported*

5.2.349 File testing/sysfs-class-scsi_tape

Has the following ABI:

- */sys/class/scsi_tape/*/stats/in_flight*
- */sys/class/scsi_tape/*/stats/io_ns*
- */sys/class/scsi_tape/*/stats/other_cnt*
- */sys/class/scsi_tape/*/stats/read_byte_cnt*
- */sys/class/scsi_tape/*/stats/read_cnt*
- */sys/class/scsi_tape/*/stats/read_ns*
- */sys/class/scsi_tape/*/stats/write_byte_cnt*
- */sys/class/scsi_tape/*/stats/write_cnt*
- */sys/class/scsi_tape/*/stats/write_ms*
- */sys/class/scsi_tape/*/stats/resid_cnt*

5.2.350 File testing/sysfs-class-spi-eeprom

Has the following ABI:

- */sys/class/spi_master/spi<bus>/spi<bus>.<dev>/fram*
- */sys/class/spi_master/spi<bus>/spi<bus>.<dev>/sernum*

5.2.351 File testing/sysfs-class-stm

Has the following ABI:

- */sys/class/stm/<stm>/masters*
- */sys/class/stm/<stm>/channels*
- */sys/class/stm/<stm>/hw_override*

5.2.352 File testing/sysfs-class-stm_source

Has the following ABI:

- */sys/class/stm_source/<stm_source>/stm_source_link*

5.2.353 File testing/sysfs-class-switchtec

switchtec - Microsemi Switchtec PCI Switch Management Endpoint

For details on this subsystem look at /driver-api/switchtec.

Has the following ABI:

- */sys/class/switchtec*
- */sys/class/switchtec/switchtec[0-9]+/component_id*
- */sys/class/switchtec/switchtec[0-9]+/component_revision*
- */sys/class/switchtec/switchtec[0-9]+/component_vendor*
- */sys/class/switchtec/switchtec[0-9]+/device_version*
- */sys/class/switchtec/switchtec[0-9]+/fw_version*
- */sys/class/switchtec/switchtec[0-9]+/partition*
- */sys/class/switchtec/switchtec[0-9]+/partition_count*
- */sys/class/switchtec/switchtec[0-9]+/product_id*
- */sys/class/switchtec/switchtec[0-9]+/product_revision*
- */sys/class/switchtec/switchtec[0-9]+/product_vendor*

5.2.354 File testing/sysfs-class-thermal

Has the following ABI:

- */sys/class/thermal/thermal_zoneX/type*
- */sys/class/thermal/thermal_zoneX/temp*
- */sys/class/thermal/thermal_zoneX/mode*
- */sys/class/thermal/thermal_zoneX/policy*
- */sys/class/thermal/thermal_zoneX/available_policies*
- */sys/class/thermal/thermal_zoneX/trip_point_Y_temp*
- */sys/class/thermal/thermal_zoneX/trip_point_Y_type*
- */sys/class/thermal/thermal_zoneX/trip_point_Y_hyst*
- */sys/class/thermal/thermal_zoneX/cdevY*
- */sys/class/thermal/thermal_zoneX/cdevY_trip_point*
- */sys/class/thermal/thermal_zoneX/cdevY_weight*
- */sys/class/thermal/thermal_zoneX/emul_temp*
- */sys/class/thermal/thermal_zoneX/k_d*
- */sys/class/thermal/thermal_zoneX/k_i*
- */sys/class/thermal/thermal_zoneX/k_po*
- */sys/class/thermal/thermal_zoneX/k_pu*
- */sys/class/thermal/thermal_zoneX/integral_cutoff*
- */sys/class/thermal/thermal_zoneX/slope*
- */sys/class/thermal/thermal_zoneX/offset*
- */sys/class/thermal/thermal_zoneX/sustainable_power*
- */sys/class/thermal/cooling_deviceX/type*
- */sys/class/thermal/cooling_deviceX/max_state*
- */sys/class/thermal/cooling_deviceX/cur_state*
- */sys/class/thermal/cooling_deviceX/stats/reset*
- */sys/class/thermal/cooling_deviceX/stats/time_in_state_ms:*
- */sys/class/thermal/cooling_deviceX/stats/total_trans*
- */sys/class/thermal/cooling_deviceX/stats/trans_table*

5.2.355 File testing/sysfs-class-typec

USB Type-C port devices (eg. /sys/class/typec/port0/)

Has the following ABI:

- */sys/class/typec/<port>/data_role*
- */sys/class/typec/<port>/power_role*
- */sys/class/typec/<port>/port_type*
- */sys/class/typec/<port>/vconn_source*
- */sys/class/typec/<port>/power_operation_mode*
- */sys/class/typec/<port>/preferred_role*
- */sys/class/typec/<port>/supported_accessory_modes*
- */sys/class/typec/<port>/usb_power_delivery_revision*
- */sys/class/typec/<port>-{partner|cable}/usb_power_delivery_revision*
- */sys/class/typec/<port>/usb_typec_revision*
- */sys/class/typec/<port>/orientation*
- */sys/class/typec/<port>/select_usb_power_delivery*
- */sys/class/typec/<port>-partner/accessory_mode*
- */sys/class/typec/<port>-partner/supports_usb_power_delivery*
- */sys/class/typec/<port>-partner/number_of_alternate_modes*
- */sys/class/typec/<port>-partner/type*
- */sys/class/typec/<port>-partner/identity/*
- */sys/class/typec/<port>-cable/type*
- */sys/class/typec/<port>-cable/plug_type*
- */sys/class/typec/<port>-<plug>/number_of_alternate_modes*
- */sys/class/typec/<port>-{partner|cable}/identity/*
- */sys/class/typec/<port>-{partner|cable}/identity/id_header*
- */sys/class/typec/<port>-{partner|cable}/identity/cert_stat*
- */sys/class/typec/<port>-{partner|cable}/identity/product*
- */sys/class/typec/<port>-{partner|cable}/identity/product_type_vdo1*
- */sys/class/typec/<port>-{partner|cable}/identity/product_type_vdo2*
- */sys/class/typec/<port>-{partner|cable}/identity/product_type_vdo3*
- */sys/class/typec/<port>/<alt mode>/supported_roles*

5.2.356 File testing/sysfs-class-usb_power_delivery

Has the following ABI:

- */sys/class/usb_power_delivery*
- */sys/class/usb_power_delivery/.../revision*
- */sys/class/usb_power_delivery/.../version*
- */sys/class/usb_power_delivery/.../source-capabilities*
- */sys/class/usb_power_delivery/.../sink-capabilities*
- */sys/class/usb_power_delivery/.../<capability>/<position>:fixed_supply*
- */sys/class/usb_power_delivery/.../<capability>/1:fixed_supply/dual_role_power*
- */sys/class/usb_power_delivery/.../source-capabilities/1:fixed_supply/usb_suspend_supported*
- */sys/class/usb_power_delivery/.../sink-capabilities/1:fixed_supply/higher_capability*
- */sys/class/usb_power_delivery/.../<capability>/1:fixed_supply/unconstrained_power*
- */sys/class/usb_power_delivery/.../<capability>/1:fixed_supply/usb_communication_capable*
- */sys/class/usb_power_delivery/.../<capability>/1:fixed_supply/dual_role_data*
- */sys/class/usb_power_delivery/.../<capability>/1:fixed_supply/unchunked_extended_messages_su*
- */sys/class/usb_power_delivery/.../<capability>/<position>:fixed_supply/voltage*
- */sys/class/usb_power_delivery/.../source-capabilities/<position>:fixed_supply/peak_current*
- */sys/class/usb_power_delivery/.../source-capabilities/<position>:fixed_supply/maximum_current*
- */sys/class/usb_power_delivery/.../sink-capabilities/<position>:fixed_supply/operational_current*
- */sys/class/usb_power_delivery/.../sink-capabilities/<position>:fixed_supply/fast_role_swap_curre*
- */sys/class/usb_power_delivery/.../<capability>/<position>:variable_supply*
- */sys/class/usb_power_delivery/.../<capability>/<position>:variable_supply/maximum_voltage*
- */sys/class/usb_power_delivery/.../<capability>/<position>:variable_supply/minimum_voltage*
- */sys/class/usb_power_delivery/.../source-capabilities/<position>:variable_supply/maximum_curre*
- */sys/class/usb_power_delivery/.../sink-capabilities/<position>:variable_supply/operational_curre*
- */sys/class/usb_power_delivery/.../<capability>/<position>:battery*
- */sys/class/usb_power_delivery/.../<capability>/<position>:battery/maximum_voltage*
- */sys/class/usb_power_delivery/.../<capability>/<position>:battery/minimum_voltage*
- */sys/class/usb_power_delivery/.../source-capabilities/<position>:battery/maximum_power*
- */sys/class/usb_power_delivery/.../sink-capabilities/<position>:battery/operational_power*
- */sys/class/usb_power_delivery/.../<capability>/<position>:programmable_supply*
- */sys/class/usb_power_delivery/.../<capability>/<position>:programmable_supply/maximum_volt*
- */sys/class/usb_power_delivery/.../<capability>/<position>:programmable_supply/minimum_volta*
- */sys/class/usb_power_delivery/.../<capability>/<position>:programmable_supply/maximum_curr*

- */sys/class/usb_power_delivery/.../source-capabilities/<position>:programmable_supply/pps_pow*

5.2.357 File testing/sysfs-class-usb_role

Has the following ABI:

- */sys/class/usb_role/*
- */sys/class/usb_role/<switch>/role*

5.2.358 File testing/sysfs-class-vduse

Has the following ABI:

- */sys/class/vduse/*
- */sys/class/vduse/control/*
- */sys/class/vduse/<device-name>/*
- */sys/class/vduse/<device-name>/msg_timeout*

5.2.359 File testing/sysfs-class-wakeup

Has the following ABI:

- */sys/class/wakeup/*
- */sys/class/wakeup/.../name*
- */sys/class/wakeup/.../active_count*
- */sys/class/wakeup/.../event_count*
- */sys/class/wakeup/.../wakeup_count*
- */sys/class/wakeup/.../expire_count*
- */sys/class/wakeup/.../active_time_ms*
- */sys/class/wakeup/.../total_time_ms*
- */sys/class/wakeup/.../max_time_ms*
- */sys/class/wakeup/.../last_change_ms*
- */sys/class/wakeup/.../prevent_suspend_time_ms*

5.2.360 File testing/sysfs-class-watchdog

Has the following ABI:

- */sys/class/watchdog/watchdogn/bootstatus*
- */sys/class/watchdog/watchdogn/options*
- */sys/class/watchdog/watchdogn/fw_version*
- */sys/class/watchdog/watchdogn/identity*
- */sys/class/watchdog/watchdogn/nowayout*
- */sys/class/watchdog/watchdogn/state*
- */sys/class/watchdog/watchdogn/status*
- */sys/class/watchdog/watchdogn/timeleft*
- */sys/class/watchdog/watchdogn/timeout*
- */sys/class/watchdog/watchdogn/pretimeout*
- */sys/class/watchdog/watchdogn/pretimeout_avaialable_governors*
- */sys/class/watchdog/watchdogn/pretimeout_governor*
- */sys/class/watchdog/watchdog1/access_cs0*

5.2.361 File testing/sysfs-class-zram

Has the following ABI:

- */sys/class/zram-control/*
- */sys/class/zram-control/hot_add*
- */sys/class/zram-control/hot_remove*

5.2.362 File testing/sysfs-dev

Has the following ABI:

- */sys/dev*

5.2.363 File testing/sysfs-devices

Has the following ABI:

- */sys/devices*

5.2.364 File testing/sysfs-devices-consumer

Has the following ABI:

- */sys/devices/.../consumer:<consumer>*

5.2.365 File testing/sysfs-devices-coredump

Has the following ABI:

- */sys/devices/.../coredump*

5.2.366 File testing/sysfs-devices-edac

Has the following ABI:

- */sys/devices/system/edac/mc/mc*/reset_counters*
- */sys/devices/system/edac/mc/mc*/seconds_since_reset*
- */sys/devices/system/edac/mc/mc*/mc_name*
- */sys/devices/system/edac/mc/mc*/size_mb*
- */sys/devices/system/edac/mc/mc*/ue_count*
- */sys/devices/system/edac/mc/mc*/ue_noinfo_count*
- */sys/devices/system/edac/mc/mc*/ce_count*
- */sys/devices/system/edac/mc/mc*/ce_noinfo_count*
- */sys/devices/system/edac/mc/mc*/sdram_scrub_rate*
- */sys/devices/system/edac/mc/mc*/max_location*
- */sys/devices/system/edac/mc/mc*/(dimm|rank)*/size*
- */sys/devices/system/edac/mc/mc*/(dimm|rank)*/dimm_dev_type*
- */sys/devices/system/edac/mc/mc*/(dimm|rank)*/dimm_edac_mode*
- */sys/devices/system/edac/mc/mc*/(dimm|rank)*/dimm_label*
- */sys/devices/system/edac/mc/mc*/(dimm|rank)*/dimm_location*
- */sys/devices/system/edac/mc/mc*/(dimm|rank)*/dimm_mem_type*
- */sys/devices/system/edac/mc/mc*/(dimm|rank)*/dimm_ce_count*
- */sys/devices/system/edac/mc/mc*/(dimm|rank)*/dimm_ue_count*

5.2.367 File testing/sysfs-devices-firmware_node

Has the following ABI:

- */sys/devices/.../firmware_node/*
- */sys/devices/.../firmware_node/description*

5.2.368 File testing/sysfs-devices-hisi_ptt

Has the following ABI:

- */sys/devices/hisi_ptt<sicl_id>_<core_id>/tune*
- */sys/devices/hisi_ptt<sicl_id>_<core_id>/tune/qos_tx_cpl*
- */sys/devices/hisi_ptt<sicl_id>_<core_id>/tune/qos_tx_np*
- */sys/devices/hisi_ptt<sicl_id>_<core_id>/tune/qos_tx_p*
- */sys/devices/hisi_ptt<sicl_id>_<core_id>/tune/rx_alloc_buf_level*
- */sys/devices/hisi_ptt<sicl_id>_<core_id>/tune/tx_alloc_buf_level*
- */sys/devices/hisi_ptt<sicl_id>_<core_id>/root_port_filters*
- */sys/devices/hisi_ptt<sicl_id>_<core_id>/root_port_filters/multiselect*
- */sys/devices/hisi_ptt<sicl_id>_<core_id>/root_port_filters/<bdf>*
- */sys/devices/hisi_ptt<sicl_id>_<core_id>/requester_filters*
- */sys/devices/hisi_ptt<sicl_id>_<core_id>/requester_filters/multiselect*
- */sys/devices/hisi_ptt<sicl_id>_<core_id>/requester_filters/<bdf>*

5.2.369 File testing/sysfs-devices-lpss_ltr

Has the following ABI:

- */sys/devices/.../lpss_ltr/*
- */sys/devices/.../lpss_ltr/ltr_mode*
- */sys/devices/.../lpss_ltr/auto_ltr*
- */sys/devices/.../lpss_ltr/sw_ltr*

5.2.370 File testing/sysfs-devices-mapping

Has the following ABI:

- */sys/devices/uncore_iio_x/dieX*
- */sys/devices/uncore_upi_x/dieX*

5.2.371 File testing/sysfs-devices-memory

Has the following ABI:

- */sys/devices/system/memory*
- */sys/devices/system/memory/memoryX/removable*
- */sys/devices/system/memory/memoryX/phys_device*
- */sys/devices/system/memory/memoryX/phys_index*
- */sys/devices/system/memory/memoryX/state*
- */sys/devices/system/memory/memoryX/valid_zones*
- */sys/devices/system/memoryX/nodeY*
- */sys/devices/system/node/nodeX/memoryY*
- */sys/devices/system/memory/crash_hotplug*

5.2.372 File testing/sysfs-devices-mmc

Has the following ABI:

- */sys/devices/.../mmc_host/mmcX/mmcX:XXXX/enhanced_area_offset*
- */sys/devices/.../mmc_host/mmcX/mmcX:XXXX/enhanced_area_size*

5.2.373 File testing/sysfs-devices-online

Has the following ABI:

- */sys/devices/.../online*

5.2.374 File testing/sysfs-devices-physical_location

Has the following ABI:

- */sys/devices/.../physical_location*
- */sys/devices/.../physical_location/panel*
- */sys/devices/.../physical_location/vertical_position*
- */sys/devices/.../physical_location/horizontal_position*
- */sys/devices/.../physical_location/dock*
- */sys/devices/.../physical_location/lid*

5.2.375 File testing/sysfs-devices-platform-ACPI-TAD

ACPI Time and Alarm (TAD) device attributes.

Has the following ABI:

- */sys/bus/platform/devices/ACPI000E:00/caps*
- */sys/bus/platform/devices/ACPI000E:00/ac_alarm*
- */sys/bus/platform/devices/ACPI000E:00/ac_policy*
- */sys/bus/platform/devices/ACPI000E:00/ac_status*
- */sys/bus/platform/devices/ACPI000E:00/dc_alarm*
- */sys/bus/platform/devices/ACPI000E:00/dc_policy*
- */sys/bus/platform/devices/ACPI000E:00/dc_status*

5.2.376 File testing/sysfs-devices-platform-_UDC_-gadget

Has the following ABI:

- */sys/devices/platform/_UDC_/gadget/suspended*
- */sys/devices/platform/_UDC_/gadget/gadget-lunX/nofua*

5.2.377 File testing/sysfs-devices-platform-docg3

Has the following ABI:

- */sys/devices/platform/docg3/f[0-3]_dps[01]_is_keylocked*
- */sys/devices/platform/docg3/f[0-3]_dps[01]_protection_key*

5.2.378 File testing/sysfs-devices-platform-dock

Has the following ABI:

- */sys/devices/platform/dock.<N>/docked*
- */sys/devices/platform/dock.<N>/undock*
- */sys/devices/platform/dock.<N>/uid*
- */sys/devices/platform/dock.<N>/flags*
- */sys/devices/platform/dock.<N>/type*

5.2.379 File testing/sysfs-devices-platform-ipmi

Has the following ABI:

- */sys/devices/platform/ipmi_bmc.*/firmware_revision*
- */sys/devices/platform/ipmi_bmc.*/aux_firmware_revision*
- */sys/devices/platform/ipmi_bmc.*/revision*
- */sys/devices/platform/ipmi_bmc.*/provides_device_sdrs*
- */sys/devices/platform/ipmi_bmc.*/device_id*
- */sys/devices/platform/ipmi_bmc.*/additional_device_support*
- */sys/devices/platform/ipmi_bmc.*/ipmi_version*
- */sys/devices/platform/ipmi_bmc.*/manufacturer_id*
- */sys/devices/platform/ipmi_bmc.*/product_id*
- */sys/devices/platform/ipmi_bmc.*/guid*
- */sys/devices/platform/ipmi_si.*/type*
- */sys/devices/platform/ipmi_si.*/idles*
- */sys/devices/platform/ipmi_si.*/watchdog_pretimeouts*
- */sys/devices/platform/ipmi_si.*/complete_transactions*
- */sys/devices/platform/ipmi_si.*/events*
- */sys/devices/platform/ipmi_si.*/interrupts*
- */sys/devices/platform/ipmi_si.*/hosed_count*
- */sys/devices/platform/ipmi_si.*/long_timeouts*
- */sys/devices/platform/ipmi_si.*/flag_fetches*
- */sys/devices/platform/ipmi_si.*/attentions*
- */sys/devices/platform/ipmi_si.*/incoming_messages*
- */sys/devices/platform/ipmi_si.*/short_timeouts*
- */sys/devices/platform/ipmi_si.*/interrupts_enabled*
- */sys/devices/platform/ipmi_si.*/params*
- */sys/devices/platform/dmi-ipmi-ssif.*/type*
- */sys/devices/platform/dmi-ipmi-ssif.*/hosed*
- */sys/devices/platform/dmi-ipmi-ssif.*/alerts*
- */sys/devices/platform/dmi-ipmi-ssif.*/sent_messages*
- */sys/devices/platform/dmi-ipmi-ssif.*/sent_messages_parts*
- */sys/devices/platform/dmi-ipmi-ssif.*/received_messages*
- */sys/devices/platform/dmi-ipmi-ssif.*/received_message_parts*
- */sys/devices/platform/dmi-ipmi-ssif.*/events*

- */sys/devices/platform/dmi-ipmi-ssif.*/watchdog_pretimeouts*
- */sys/devices/platform/dmi-ipmi-ssif.*/flag_fetches*
- */sys/devices/platform/dmi-ipmi-ssif.*/send_retries*
- */sys/devices/platform/dmi-ipmi-ssif.*/receive_retries*
- */sys/devices/platform/dmi-ipmi-ssif.*/send_errors*
- */sys/devices/platform/dmi-ipmi-ssif.*/receive_errors*

5.2.380 File testing/sysfs-devices-platform-kunpeng_hccs

Has the following ABI:

- */sys/devices/platform/HISI04Bx:00/chipX/all_linked*
- */sys/devices/platform/HISI04Bx:00/chipX/linked_full_lane*
- */sys/devices/platform/HISI04Bx:00/chipX/crc_err_cnt*
- */sys/devices/platform/HISI04Bx:00/chipX/dieY/all_linked*
- */sys/devices/platform/HISI04Bx:00/chipX/dieY/linked_full_lane*
- */sys/devices/platform/HISI04Bx:00/chipX/dieY/crc_err_cnt*
- */sys/devices/platform/HISI04Bx:00/chipX/dieY/hccsN/type*
- */sys/devices/platform/HISI04Bx:00/chipX/dieY/hccsN/lane_mode*
- */sys/devices/platform/HISI04Bx:00/chipX/dieY/hccsN/enable*
- */sys/devices/platform/HISI04Bx:00/chipX/dieY/hccsN/cur_lane_num*
- */sys/devices/platform/HISI04Bx:00/chipX/dieY/hccsN/link_fsm*
- */sys/devices/platform/HISI04Bx:00/chipX/dieY/hccsN/lane_mask*
- */sys/devices/platform/HISI04Bx:00/chipX/dieY/hccsN/crc_err_cnt*

5.2.381 File testing/sysfs-devices-platform-sh_mobile_lcdc_fb

Has the following ABI:

- */sys/devices/platform/sh_mobile_lcdc_fb.[0-3]/graphics/fb[0-9]/ovl_alpha*
- */sys/devices/platform/sh_mobile_lcdc_fb.[0-3]/graphics/fb[0-9]/ovl_mode*
- */sys/devices/platform/sh_mobile_lcdc_fb.[0-3]/graphics/fb[0-9]/ovl_position*
- */sys/devices/platform/sh_mobile_lcdc_fb.[0-3]/graphics/fb[0-9]/ovl_rop3*

5.2.382 File testing/sysfs-devices-platform-soc-ipa

Has the following ABI:

- */sys/devices/platform/soc@X/XXXXXXX.ipa/*
- */.../XXXXXXX.ipa/version*
- */.../XXXXXXX.ipa/feature/*
- */.../XXXXXXX.ipa/feature/rx_offload*
- */.../XXXXXXX.ipa/feature/tx_offload*
- */.../XXXXXXX.ipa/endpoint_id/*
- */.../XXXXXXX.ipa/endpoint_id/modem_rx*
- */.../XXXXXXX.ipa/endpoint_id/modem_tx*
- */.../XXXXXXX.ipa/endpoint_id/monitor_rx*
- */.../XXXXXXX.ipa/modem/*
- */.../XXXXXXX.ipa/modem/rx_endpoint_id*
- */.../XXXXXXX.ipa/modem/tx_endpoint_id*

5.2.383 File testing/sysfs-devices-platform-stratix10-rsu

Intel Stratix10 Remote System Update (RSU) device attributes

Has the following ABI:

- */sys/devices/platform/stratix10-rsu.0/current_image*
- */sys/devices/platform/stratix10-rsu.0/fail_image*
- */sys/devices/platform/stratix10-rsu.0/state*
- */sys/devices/platform/stratix10-rsu.0/version*
- */sys/devices/platform/stratix10-rsu.0/error_location*
- */sys/devices/platform/stratix10-rsu.0/error_details*
- */sys/devices/platform/stratix10-rsu.0/retry_counter*
- */sys/devices/platform/stratix10-rsu.0/reboot_image*
- */sys/devices/platform/stratix10-rsu.0/notify*
- */sys/devices/platform/stratix10-rsu.0/dcmf0*
- */sys/devices/platform/stratix10-rsu.0/dcmf1*
- */sys/devices/platform/stratix10-rsu.0/dcmf2*
- */sys/devices/platform/stratix10-rsu.0/dcmf3*
- */sys/devices/platform/stratix10-rsu.0/max_retry*

5.2.384 File testing/sysfs-devices-platform-trackpoint

Has the following ABI:

- */sys/devices/platform/i8042/.../sensitivity*
- */sys/devices/platform/i8042/.../intertia*
- */sys/devices/platform/i8042/.../reach*
- */sys/devices/platform/i8042/.../draghys*
- */sys/devices/platform/i8042/.../mindrag*
- */sys/devices/platform/i8042/.../speed*
- */sys/devices/platform/i8042/.../thresh*
- */sys/devices/platform/i8042/.../upthresh*
- */sys/devices/platform/i8042/.../ztime*
- */sys/devices/platform/i8042/.../jenks*
- */sys/devices/platform/i8042/.../skipback*
- */sys/devices/platform/i8042/.../ext_dev*
- */sys/devices/platform/i8042/.../press_to_select*
- */sys/devices/platform/i8042/.../drift_time*

5.2.385 File testing/sysfs-devices-power

Has the following ABI:

- */sys/devices/.../power/*
- */sys/devices/.../power/wakeup*
- */sys/devices/.../power/control*
- */sys/devices/.../power/async*
- */sys/devices/.../power/wakeup_count*
- */sys/devices/.../power/wakeup_active_count*
- */sys/devices/.../power/wakeup_abort_count*
- */sys/devices/.../power/wakeup_expire_count*
- */sys/devices/.../power/wakeup_active*
- */sys/devices/.../power/wakeup_total_time_ms*
- */sys/devices/.../power/wakeup_max_time_ms*
- */sys/devices/.../power/wakeup_last_time_ms*
- */sys/devices/.../power/wakeup_prevent_sleep_time_ms*
- */sys/devices/.../power/autosuspend_delay_ms*
- */sys/devices/.../power/pm_qos_resume_latency_us*

- */sys/devices/.../power/pm_qos_latency_tolerance_us*
- */sys/devices/.../power/pm_qos_no_power_off*
- */sys/devices/.../power/runtime_status*
- */sys/devices/.../power/runtime_active_time*
- */sys/devices/.../power/runtime_suspended_time*
- */sys/devices/.../power/runtime_usage*
- */sys/devices/.../power/runtime_enabled*
- */sys/devices/.../power/runtime_active_kids*

5.2.386 File testing/sysfs-devices-power_resources_D0

Has the following ABI:

- */sys/devices/.../power_resources_D0/*

5.2.387 File testing/sysfs-devices-power_resources_D1

Has the following ABI:

- */sys/devices/.../power_resources_D1/*

5.2.388 File testing/sysfs-devices-power_resources_D2

Has the following ABI:

- */sys/devices/.../power_resources_D2/*

5.2.389 File testing/sysfs-devices-power_resources_D3hot

Has the following ABI:

- */sys/devices/.../power_resources_D3hot/*

5.2.390 File testing/sysfs-devices-power_resources_wakeup

Has the following ABI:

- */sys/devices/.../power_resources_wakeup/*

5.2.391 File testing/sysfs-devices-power_state

Has the following ABI:

- */sys/devices/.../power_state*

5.2.392 File testing/sysfs-devices-real_power_state

Has the following ABI:

- */sys/devices/.../real_power_state*

5.2.393 File testing/sysfs-devices-removable

Has the following ABI:

- */sys/devices/.../removable*

5.2.394 File testing/sysfs-devices-resource_in_use

Has the following ABI:

- */sys/devices/.../resource_in_use*

5.2.395 File testing/sysfs-devices-soc

Has the following ABI:

- */sys/devices/socX*
- */sys/devices/socX/machine*
- */sys/devices/socX/family*
- */sys/devices/socX/serial_number*
- */sys/devices/socX/soc_id*
- */sys/devices/socX/revision*
- */sys/devices/socX/process*
- */sys/bus/soc*

5.2.396 File testing/sysfs-devices-software_node

Has the following ABI:

- */sys/devices/.../software_node/*

5.2.397 File testing/sysfs-devices-state_synced

Has the following ABI:

- */sys/devices/.../state_synced*

5.2.398 File testing/sysfs-devices-sun

Has the following ABI:

- */sys/devices/.../sun*

5.2.399 File testing/sysfs-devices-supplier

Has the following ABI:

- */sys/devices/.../supplier:<supplier>*

5.2.400 File testing/sysfs-devices-system-cpu

Has the following ABI:

- */sys/devices/system/cpu/*
- */sys/devices/system/cpu/kernel_max*
- */sys/devices/system/cpu/probe*
- */sys/devices/system/cpu/cpuX/node*
- */sys/devices/system/cpu/cpuX/topology/core_siblings*
- */sys/devices/system/cpu/cpuidle/available_governors*
- */sys/devices/system/cpu/cpuX/cpuidle/state<N>/name*
- */sys/devices/system/cpu/cpuX/cpuidle/state<N>/desc*
- */sys/devices/system/cpu/cpuX/cpuidle/state<N>/disable*
- */sys/devices/system/cpu/cpuX/cpuidle/state<N>/default_status*
- */sys/devices/system/cpu/cpuX/cpuidle/state<N>/residency*
- */sys/devices/system/cpu/cpuX/cpuidle/state<N>/s2idle/*
- */sys/devices/system/cpu/cpuX/cpuidle/state<N>/s2idle/time*
- */sys/devices/system/cpu/cpuX/cpuidle/state<N>/s2idle/usage*
- */sys/devices/system/cpu/cpuX/cpufreq/**
- */sys/devices/system/cpu/cpuX/cpufreq/freqdomain_cpus*
- */sys/devices/system/cpu/cpu*/cache/index3/cache_disable_{0,1}*
- */sys/devices/system/cpu/cpufreq/boost*
- */sys/devices/system/cpu/cpuX/crash_notes*
- */sys/devices/system/cpu/intel_pstate/max_perf_pct*

- */sys/devices/system/cpu/cpu*/cache/index*/<set_of_attributes_mentioned_below>*
- */sys/devices/system/cpu/cpu*/cache/index*/id*
- */sys/devices/system/cpu/cpuX/cpufreq/throttle_stats*
- */sys/devices/system/cpu/cpufreq/policyX/throttle_stats*
- */sys/devices/system/cpu/cpuX/regs/*
- */sys/devices/system/cpu/aarch32_el0*
- */sys/devices/system/cpu/cpuX/cpu_capacity*
- */sys/devices/system/cpu/vulnerabilities*
- */sys/devices/system/cpu/smt*
- */sys/devices/system/cpu/cpuX/power/energy_perf_bias*
- */sys/devices/system/cpu/umwait_control*
- */sys/devices/system/cpu/svm*
- */sys/devices/system/cpu/cpuX/purr*
- */sys/devices/system/cpu/cpuX/spurr*
- */sys/devices/system/cpu/cpuX/idle_purr*
- */sys/devices/system/cpu/cpuX/idle_spurr*
- */sys/devices/system/cpu/cpuX/mte_tcf_preferred*
- */sys/devices/system/cpu/nohz_full*
- */sys/devices/system/cpu/isolated*
- */sys/devices/system/cpu/crash_hotplug*

5.2.401 File testing/sysfs-devices-system-ibm-rtl

Has the following ABI:

- */sys/devices/system/ibm_rtl/state*
- */sys/devices/system/ibm_rtl/version*

5.2.402 File testing/sysfs-devices-system-xen_cpu

Has the following ABI:

- */sys/devices/system/xen_cpu/*
- */sys/devices/system/xen_cpu/xen_cpu#/online*

5.2.403 File testing/sysfs-devices-vfio-dev

Has the following ABI:

- */sys/.../<device>/vfio-dev/vfioX/*

5.2.404 File testing/sysfs-devices-waiting_for_supplier

Has the following ABI:

- */sys/devices/.../waiting_for_supplier*

5.2.405 File testing/sysfs-devices-xenbus

Has the following ABI:

- */sys/devices/*/xenbus/event_channels*
- */sys/devices/*/xenbus/events*
- */sys/devices/*/xenbus/jiffies_eoi_delayed*
- */sys/devices/*/xenbus/spurious_events*
- */sys/devices/*/xenbus/spurious_threshold*

5.2.406 File testing/sysfs-driver-altera-cvp

Has the following ABI:

- */sys/bus/pci/drivers/altera-cvp/chkcfg*

5.2.407 File testing/sysfs-driver-aspeed-uart-routing

Has the following ABI:

- */sys/bus/platform/drivers/aspeed-uart-routing/*/uart**
- */sys/bus/platform/drivers/aspeed-uart-routing/*/io**

5.2.408 File testing/sysfs-driver-bd9571mwv-regulator

Has the following ABI:

- */sys/bus/i2c/devices/.../bd9571mwv-regulator.*.auto/backup_mode*

5.2.409 File testing/sysfs-driver-ccp

Has the following ABI:

- */sys/bus/pci/devices/<BDF>/fused_part*
- */sys/bus/pci/devices/<BDF>/debug_lock_on*
- */sys/bus/pci/devices/<BDF>/tsme_status*
- */sys/bus/pci/devices/<BDF>/anti_rollback_status*
- */sys/bus/pci/devices/<BDF>/rpmc_production_enabled*
- */sys/bus/pci/devices/<BDF>/rpmc_spirom_available*
- */sys/bus/pci/devices/<BDF>/hsp_tpm_available*
- */sys/bus/pci/devices/<BDF>/rom_armor_enforced*
- */sys/bus/pci/devices/<BDF>/bootloader_version*
- */sys/bus/pci/devices/<BDF>/tee_version*

5.2.410 File testing/sysfs-driver-chromeos-acpi

Has the following ABI:

- */sys/bus/platform/devices/GGL0001:*/BINF.2*
- */sys/bus/platform/devices/GGL0001:*/BINF.3*
- */sys/bus/platform/devices/GGL0001:*/CHSW*
- */sys/bus/platform/devices/GGL0001:*/FMAP*
- */sys/bus/platform/devices/GGL0001:*/FRID*
- */sys/bus/platform/devices/GGL0001:*/FWID*
- */sys/bus/platform/devices/GGL0001:*/GPIO.X/GPIO.0*
- */sys/bus/platform/devices/GGL0001:*/GPIO.X/GPIO.1*
- */sys/bus/platform/devices/GGL0001:*/GPIO.X/GPIO.2*
- */sys/bus/platform/devices/GGL0001:*/GPIO.X/GPIO.3*
- */sys/bus/platform/devices/GGL0001:*/HWID*
- */sys/bus/platform/devices/GGL0001:*/MECK*
- */sys/bus/platform/devices/GGL0001:*/VBNV.0*
- */sys/bus/platform/devices/GGL0001:*/VBNV.1*
- */sys/bus/platform/devices/GGL0001:*/VDAT*

5.2.411 File testing/sysfs-driver-eud

Has the following ABI:

- */sys/bus/platform/drivers/qcom_eud/.../enable*

5.2.412 File testing/sysfs-driver-fsi-master-gpio

Has the following ABI:

- */sys/bus/platform/devices/[..]/fsi-master-gpio/external_mode*

5.2.413 File testing/sysfs-driver-ge-achc

Has the following ABI:

- */sys/bus/spi/<dev>/update_firmware*
- */sys/bus/spi/<dev>/reset*

5.2.414 File testing/sysfs-driver-genwqe

Has the following ABI:

- */sys/class/genwqe/genwqe<n>_card/version*
- */sys/class/genwqe/genwqe<n>_card/appid*
- */sys/class/genwqe/genwqe<n>_card/type*
- */sys/class/genwqe/genwqe<n>_card/curr_bitstream*
- */sys/class/genwqe/genwqe<n>_card/next_bitstream*
- */sys/class/genwqe/genwqe<n>_card/reload_bitstream*
- */sys/class/genwqe/genwqe<n>_card/tempsens*
- */sys/class/genwqe/genwqe<n>_card/freerunning_timer*
- */sys/class/genwqe/genwqe<n>_card/queue_working_time*
- */sys/class/genwqe/genwqe<n>_card/state*
- */sys/class/genwqe/genwqe<n>_card/base_clock*
- */sys/class/genwqe/genwqe<n>_card/device/sriov_numvfs*

5.2.415 File testing/sysfs-driver-habanalabs

Has the following ABI:

- */sys/class/accel/accel<n>/device/armcp_kernel_ver*
- */sys/class/accel/accel<n>/device/armcp_ver*
- */sys/class/accel/accel<n>/device/clk_max_freq_mhz*
- */sys/class/accel/accel<n>/device/clk_cur_freq_mhz*
- */sys/class/accel/accel<n>/device/cpld_ver*
- */sys/class/accel/accel<n>/device/cpucp_kernel_ver*
- */sys/class/accel/accel<n>/device/cpucp_ver*
- */sys/class/accel/accel<n>/device/device_type*
- */sys/class/accel/accel<n>/device/eeprom*
- */sys/class/accel/accel<n>/device/fuse_ver*
- */sys/class/accel/accel<n>/device/fw_os_ver*
- */sys/class/accel/accel<n>/device/hard_reset*
- */sys/class/accel/accel<n>/device/hard_reset_cnt*
- */sys/class/accel/accel<n>/device/high_pll*
- */sys/class/accel/accel<n>/device/ic_clk*
- */sys/class/accel/accel<n>/device/ic_clk_curr*
- */sys/class/accel/accel<n>/device/infineon_ver*
- */sys/class/accel/accel<n>/device/max_power*
- */sys/class/accel/accel<n>/device/mme_clk*
- */sys/class/accel/accel<n>/device/mme_clk_curr*
- */sys/class/accel/accel<n>/device/module_id*
- */sys/class/accel/accel<n>/device/parent_device*
- */sys/class/accel/accel<n>/device/pci_addr*
- */sys/class/accel/accel<n>/device/pm_mng_profile*
- */sys/class/accel/accel<n>/device/preboot_btl_ver*
- */sys/class/accel/accel<n>/device/security_enabled*
- */sys/class/accel/accel<n>/device/soft_reset*
- */sys/class/accel/accel<n>/device/soft_reset_cnt*
- */sys/class/accel/accel<n>/device/status*
- */sys/class/accel/accel<n>/device/thermal_ver*
- */sys/class/accel/accel<n>/device/tpc_clk*
- */sys/class/accel/accel<n>/device/tpc_clk_curr*

- */sys/class/accel/accel<n>/device/uboot_ver*
- */sys/class/accel/accel<n>/device/vrm_ver*

5.2.416 File testing/sysfs-driver-hid

Has the following ABI:

- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/report_descriptor*
- */sys/class/bluetooth/hci<addr>/<hid-bus>:<vendor-id>:<product-id>.<num>/report_descriptor*
- */sys/class/hidraw/hidraw<num>/device/report_descriptor*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/country*
- */sys/class/bluetooth/hci<addr>/<hid-bus>:<vendor-id>:<product-id>.<num>/country*
- */sys/class/hidraw/hidraw<num>/device/country*

5.2.417 File testing/sysfs-driver-hid-corsair

Has the following ABI:

- */sys/bus/drivers/corsair/<dev>/macro_mode*
- */sys/bus/drivers/corsair/<dev>/current_profile*

5.2.418 File testing/sysfs-driver-hid-lenovo

Has the following ABI:

- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/press_to_select*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/dragging*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/release_to_select*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/select_right*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/sensitivity*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/press_speed*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/fn_lock*

5.2.419 File testing/sysfs-driver-hid-logitech-hidpp

Has the following ABI:

- */sys/bus/hid/drivers/logitech-hidpp-device/<dev>/range*
- */sys/bus/hid/drivers/logitech-hidpp-device/<dev>/builtin_power_supply*

5.2.420 File testing/sysfs-driver-hid-logitech-lg4ff

Has the following ABI:

- */sys/bus/hid/drivers/logitech/<dev>/range*
- */sys/bus/hid/drivers/logitech/<dev>/alternate_modes*
- */sys/bus/hid/drivers/logitech/<dev>/real_id*
- */sys/bus/hid/drivers/logitech/<dev>/combine_pedals*

5.2.421 File testing/sysfs-driver-hid-multitouch

Has the following ABI:

- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/quirks*

5.2.422 File testing/sysfs-driver-hid-ntrig

Has the following ABI:

- */sys/bus/hid/drivers/ntrig/<dev>/activate_slack*
- */sys/bus/hid/drivers/ntrig/<dev>/decativate_slack*
- */sys/bus/hid/drivers/ntrig/<dev>/activation_width*
- */sys/bus/hid/drivers/ntrig/<dev>/activation_height*
- */sys/bus/hid/drivers/ntrig/<dev>/min_width*
- */sys/bus/hid/drivers/ntrig/<dev>/min_height*
- */sys/bus/hid/drivers/ntrig/<dev>/sensor_physical_width*
- */sys/bus/hid/drivers/ntrig/<dev>/sensor_physical_height*
- */sys/bus/hid/drivers/ntrig/<dev>/sensor_logical_width*
- */sys/bus/hid/drivers/ntrig/<dev>/sensor_logical_height*

5.2.423 File testing/sysfs-driver-hid-picolcd

Has the following ABI:

- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/operation_mode*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/operation_mode_delay*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/fb_update_rate*

5.2.424 File testing/sysfs-driver-hid-prodikeys

Has the following ABI:

- */sys/bus/hid/drivers/prodikeys/.../channel*
- */sys/bus/hid/drivers/prodikeys/.../sustain*
- */sys/bus/hid/drivers/prodikeys/.../octave*

5.2.425 File testing/sysfs-driver-hid-roccat-kone

Has the following ABI:

- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kone/roccatkone<minor>/actual_dpi*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kone/roccatkone<minor>/actual_profile*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kone/roccatkone<minor>/firmware_version*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kone/roccatkone<minor>/profile[1-5]*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kone/roccatkone<minor>/settings*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kone/roccatkone<minor>/startup_profile*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kone/roccatkone<minor>/tcu*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kone/roccatkone<minor>/weight*

5.2.426 File testing/sysfs-driver-hid-srws1

Has the following ABI:

- */sys/class/leds/SRWS1::<serial>::RPM1*
- */sys/class/leds/SRWS1::<serial>::RPM2*
- */sys/class/leds/SRWS1::<serial>::RPM3*
- */sys/class/leds/SRWS1::<serial>::RPM4*
- */sys/class/leds/SRWS1::<serial>::RPM5*
- */sys/class/leds/SRWS1::<serial>::RPM6*
- */sys/class/leds/SRWS1::<serial>::RPM7*
- */sys/class/leds/SRWS1::<serial>::RPM8*
- */sys/class/leds/SRWS1::<serial>::RPM9*
- */sys/class/leds/SRWS1::<serial>::RPM10*
- */sys/class/leds/SRWS1::<serial>::RPM11*
- */sys/class/leds/SRWS1::<serial>::RPM12*
- */sys/class/leds/SRWS1::<serial>::RPM13*
- */sys/class/leds/SRWS1::<serial>::RPM14*
- */sys/class/leds/SRWS1::<serial>::RPM15*
- */sys/class/leds/SRWS1::<serial>::RPMALL*

5.2.427 File testing/sysfs-driver-hid-wiimote

Has the following ABI:

- */sys/bus/hid/drivers/wiimote/<dev>/led1*
- */sys/bus/hid/drivers/wiimote/<dev>/led2*
- */sys/bus/hid/drivers/wiimote/<dev>/led3*
- */sys/bus/hid/drivers/wiimote/<dev>/led4*
- */sys/bus/hid/drivers/wiimote/<dev>/extension*
- */sys/bus/hid/drivers/wiimote/<dev>/devtype*
- */sys/bus/hid/drivers/wiimote/<dev>/bboard_calib*
- */sys/bus/hid/drivers/wiimote/<dev>/pro_calib*

5.2.428 File testing/sysfs-driver-input-axp-pek

Has the following ABI:

- */sys/class/input/input(x)/device/startup*
- */sys/class/input/input(x)/device/shutdown*

5.2.429 File testing/sysfs-driver-input-cros-ec-keyb

Has the following ABI:

- */sys/class/input/input(x)/device/function_row_physmap*

5.2.430 File testing/sysfs-driver-input-exc3000

Has the following ABI:

- */sys/bus/i2c/devices/xxx/fw_version*
- */sys/bus/i2c/devices/xxx/model*
- */sys/bus/i2c/devices/xxx/type*

5.2.431 File testing/sysfs-driver-intc_sar

Has the following ABI:

- */sys/bus/platform/devices/INTC1092:00/intc_reg*
- */sys/bus/platform/devices/INTC1092:00/intc_data*

5.2.432 File testing/sysfs-driver-intel-i915-hwmon

Has the following ABI:

- */sys/bus/pci/drivers/i915/.../hwmon/hwmon<i>/in0_input*
- */sys/bus/pci/drivers/i915/.../hwmon/hwmon<i>/power1_max*
- */sys/bus/pci/drivers/i915/.../hwmon/hwmon<i>/power1_rated_max*
- */sys/bus/pci/drivers/i915/.../hwmon/hwmon<i>/power1_max_interval*
- */sys/bus/pci/drivers/i915/.../hwmon/hwmon<i>/power1_crit*
- */sys/bus/pci/drivers/i915/.../hwmon/hwmon<i>/curr1_crit*
- */sys/bus/pci/drivers/i915/.../hwmon/hwmon<i>/energy1_input*

5.2.433 File testing/sysfs-driver-intel-m10-bmc

Has the following ABI:

- */sys/bus/.../drivers/intel-m10-bmc/.../bmc_version*
- */sys/bus/.../drivers/intel-m10-bmc/.../bmcfw_version*
- */sys/bus/.../drivers/intel-m10-bmc/.../mac_address*
- */sys/bus/.../drivers/intel-m10-bmc/.../mac_count*

5.2.434 File testing/sysfs-driver-intel-m10-bmc-sec-update

Has the following ABI:

- */sys/bus/platform/drivers/intel-m10bmc-sec-update/.../security/sr_root_entry_hash*
- */sys/bus/platform/drivers/intel-m10bmc-sec-update/.../security/pr_root_entry_hash*
- */sys/bus/platform/drivers/intel-m10bmc-sec-update/.../security/bmc_root_entry_hash*
- */sys/bus/platform/drivers/intel-m10bmc-sec-update/.../security/sr_canceled_csks*
- */sys/bus/platform/drivers/intel-m10bmc-sec-update/.../security/pr_canceled_csks*
- */sys/bus/platform/drivers/intel-m10bmc-sec-update/.../security/bmc_canceled_csks*
- */sys/bus/platform/drivers/intel-m10bmc-sec-update/.../security/flash_count*

5.2.435 File testing/sysfs-driver-intel-rapid-start

Has the following ABI:

- */sys/bus/acpi/intel-rapid-start/wakeup_events*
- */sys/bus/acpi/intel-rapid-start/wakeup_time*

5.2.436 File testing/sysfs-driver-intel-xe-hwmon

Has the following ABI:

- */sys/bus/pci/drivers/xe/.../hwmon/hwmon<i>/power1_max*
- */sys/bus/pci/drivers/xe/.../hwmon/hwmon<i>/power1_rated_max*
- */sys/bus/pci/drivers/xe/.../hwmon/hwmon<i>/power1_crit*
- */sys/bus/pci/drivers/xe/.../hwmon/hwmon<i>/curr1_crit*
- */sys/bus/pci/drivers/xe/.../hwmon/hwmon<i>/in0_input*
- */sys/bus/pci/drivers/xe/.../hwmon/hwmon<i>/energy1_input*
- */sys/bus/pci/drivers/xe/.../hwmon/hwmon<i>/power1_max_interval*

5.2.437 File testing/sysfs-driver-intel_sdsi

Has the following ABI:

- */sys/bus/auxiliary/devices/intel_vsec.sdsi.X*
- */sys/bus/auxiliary/devices/intel_vsec.sdsi.X/guid*
- */sys/bus/auxiliary/devices/intel_vsec.sdsi.X/registers*
- */sys/bus/auxiliary/devices/intel_vsec.sdsi.X/provision_akc*
- */sys/bus/auxiliary/devices/intel_vsec.sdsi.X/provision_cap*
- */sys/bus/auxiliary/devices/intel_vsec.sdsi.X/meter_certificate*
- */sys/bus/auxiliary/devices/intel_vsec.sdsi.X/state_certificate*

5.2.438 File testing/sysfs-driver-jz4780-efuse

Has the following ABI:

- */sys/devices/*/<our-device>/nvmem*

5.2.439 File testing/sysfs-driver-pciback

Has the following ABI:

- */sys/bus/pci/drivers/pciback/quirks*
- */sys/bus/pci/drivers/pciback/allow_interrupt_control*

5.2.440 File testing/sysfs-driver-ppi

Has the following ABI:

- */sys/class/tpm/tpmX/ppi/*
- */sys/class/tpm/tpmX/ppi/version*
- */sys/class/tpm/tpmX/ppi/request*
- */sys/class/tpm/tpmX/ppi/response*
- */sys/class/tpm/tpmX/ppi/transition_action*
- */sys/class/tpm/tpmX/ppi/tcg_operations*
- */sys/class/tpm/tpmX/ppi/vs_operations*

5.2.441 File testing/sysfs-driver-qat

Has the following ABI:

- */sys/bus/pci/devices/<BDF>/qat/state*
- */sys/bus/pci/devices/<BDF>/qat/cfg_services*
- */sys/bus/pci/devices/<BDF>/qat/pm_idle_enabled*
- */sys/bus/pci/devices/<BDF>/qat/rp2srv*
- */sys/bus/pci/devices/<BDF>/qat/num_rps*

5.2.442 File testing/sysfs-driver-qat_ras

Has the following ABI:

- */sys/bus/pci/devices/<BDF>/qat_ras/errors_correctable*
- */sys/bus/pci/devices/<BDF>/qat_ras/errors_nonfatal*
- */sys/bus/pci/devices/<BDF>/qat_ras/errors_fatal*
- */sys/bus/pci/devices/<BDF>/qat_ras/reset_error_counters*

5.2.443 File testing/sysfs-driver-qat_rl

Has the following ABI:

- */sys/bus/pci/devices/<BDF>/qat_rl/sla_op*
- */sys/bus/pci/devices/<BDF>/qat_rl/rp*
- */sys/bus/pci/devices/<BDF>/qat_rl/id*
- */sys/bus/pci/devices/<BDF>/qat_rl/cir*
- */sys/bus/pci/devices/<BDF>/qat_rl/pir*
- */sys/bus/pci/devices/<BDF>/qat_rl/srv*
- */sys/bus/pci/devices/<BDF>/qat_rl/cap_rem*

5.2.444 File testing/sysfs-driver-samsung-laptop

Has the following ABI:

- */sys/devices/platform/samsung/performance_level*
- */sys/devices/platform/samsung/battery_life_extender*
- */sys/devices/platform/samsung/usb_charge*
- */sys/devices/platform/samsung/lid_handling*

5.2.445 File testing/sysfs-driver-st

Has the following ABI:

- */sys/bus/scsi/drivers/st/debug_flag*

5.2.446 File testing/sysfs-driver-tegra-fuse

Has the following ABI:

- */sys/devices/*/<our-device>/fuse*

5.2.447 File testing/sysfs-driver-toshiba_acpi

Has the following ABI:

- */sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/kbd_backlight_mode*
- */sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/kbd_backlight_timeout*
- */sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/position*
- */sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/touchpad*
- */sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/available_kbd_modes*
- */sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/kbd_type*
- */sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/usb_sleep_charge*
- */sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/sleep_functions_on_batte*
- */sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/usb_rapid_charge*
- */sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/usb_sleep_music*
- */sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/version*
- */sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/fan*
- */sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/kbd_function_keys*
- */sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/panel_power_on*
- */sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/usb_three*
- */sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS{1900,620{0,7,8}}:00/cooling_method*

5.2.448 File testing/sysfs-driver-toshiba_haps

Has the following ABI:

- */sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS620A:00/protection_level*
- */sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS620A:00/reset_protection*

5.2.449 File testing/sysfs-driver-typec-displayport

Has the following ABI:

- */sys/bus/typec/devices/.../displayport/configuration*
- */sys/bus/typec/devices/.../displayport/pin_assignment*
- */sys/bus/typec/devices/.../displayport/hpd*

5.2.450 File testing/sysfs-driver-uacce

Has the following ABI:

- */sys/class/uacce/<dev_name>/api*
- */sys/class/uacce/<dev_name>/flags*
- */sys/class/uacce/<dev_name>/available_instances*
- */sys/class/uacce/<dev_name>/isolate_strategy*
- */sys/class/uacce/<dev_name>/isolate*
- */sys/class/uacce/<dev_name>/algorithms*
- */sys/class/uacce/<dev_name>/region_mmio_size*
- */sys/class/uacce/<dev_name>/region_dus_size*

5.2.451 File testing/sysfs-driver-ucsi-ccg

Has the following ABI:

- */sys/bus/i2c/drivers/ucsi_ccg/.../do_flash*

5.2.452 File testing/sysfs-driver-ufs

Has the following ABI:

- */sys/bus/*/drivers/ufshcd/*/auto_hibern8*
- */sys/bus/platform/drivers/ufshcd/*/device_descriptor/device_type*
- */sys/bus/platform/devices/*.ufs/device_descriptor/device_type*
- */sys/bus/platform/drivers/ufshcd/*/device_descriptor/device_class*
- */sys/bus/platform/devices/*.ufs/device_descriptor/device_class*
- */sys/bus/platform/drivers/ufshcd/*/device_descriptor/device_sub_class*
- */sys/bus/platform/devices/*.ufs/device_descriptor/device_sub_class*
- */sys/bus/platform/drivers/ufshcd/*/device_descriptor/protocol*
- */sys/bus/platform/devices/*.ufs/device_descriptor/protocol*
- */sys/bus/platform/drivers/ufshcd/*/device_descriptor/number_of_luns*
- */sys/bus/platform/devices/*.ufs/device_descriptor/number_of_luns*

- /sys/bus/platform/drivers/ufshcd/*/device_descriptor/number_of_wluns
- /sys/bus/platform/devices/*.ufs/device_descriptor/number_of_wluns
- /sys/bus/platform/drivers/ufshcd/*/device_descriptor/boot_enable
- /sys/bus/platform/devices/*.ufs/device_descriptor/boot_enable
- /sys/bus/platform/drivers/ufshcd/*/device_descriptor/descriptor_access_enable
- /sys/bus/platform/devices/*.ufs/device_descriptor/descriptor_access_enable
- /sys/bus/platform/drivers/ufshcd/*/device_descriptor/initial_power_mode
- /sys/bus/platform/devices/*.ufs/device_descriptor/initial_power_mode
- /sys/bus/platform/drivers/ufshcd/*/device_descriptor/high_priority_lun
- /sys/bus/platform/devices/*.ufs/device_descriptor/high_priority_lun
- /sys/bus/platform/drivers/ufshcd/*/device_descriptor/secure_removal_type
- /sys/bus/platform/devices/*.ufs/device_descriptor/secure_removal_type
- /sys/bus/platform/drivers/ufshcd/*/device_descriptor/support_security_lun
- /sys/bus/platform/devices/*.ufs/device_descriptor/support_security_lun
- /sys/bus/platform/drivers/ufshcd/*/device_descriptor/bkops_termination_latency
- /sys/bus/platform/devices/*.ufs/device_descriptor/bkops_termination_latency
- /sys/bus/platform/drivers/ufshcd/*/device_descriptor/initial_active_icc_level
- /sys/bus/platform/devices/*.ufs/device_descriptor/initial_active_icc_level
- /sys/bus/platform/drivers/ufshcd/*/device_descriptor/specification_version
- /sys/bus/platform/devices/*.ufs/device_descriptor/specification_version
- /sys/bus/platform/drivers/ufshcd/*/device_descriptor/manufacturing_date
- /sys/bus/platform/devices/*.ufs/device_descriptor/manufacturing_date
- /sys/bus/platform/drivers/ufshcd/*/device_descriptor/manufacturer_id
- /sys/bus/platform/devices/*.ufs/device_descriptor/manufacturer_id
- /sys/bus/platform/drivers/ufshcd/*/device_descriptor/rtt_capability
- /sys/bus/platform/devices/*.ufs/device_descriptor/rtt_capability
- /sys/bus/platform/drivers/ufshcd/*/device_descriptor/rtc_update
- /sys/bus/platform/devices/*.ufs/device_descriptor/rtc_update
- /sys/bus/platform/drivers/ufshcd/*/device_descriptor/ufs_features
- /sys/bus/platform/devices/*.ufs/device_descriptor/ufs_features
- /sys/bus/platform/drivers/ufshcd/*/device_descriptor/ffu_timeout
- /sys/bus/platform/devices/*.ufs/device_descriptor/ffu_timeout
- /sys/bus/platform/drivers/ufshcd/*/device_descriptor/queue_depth
- /sys/bus/platform/devices/*.ufs/device_descriptor/queue_depth

- /sys/bus/platform/drivers/ufshcd/*/device_descriptor/device_version
- /sys/bus/platform/devices/*.ufs/device_descriptor/device_version
- /sys/bus/platform/drivers/ufshcd/*/device_descriptor/number_of_secure_wpa
- /sys/bus/platform/devices/*.ufs/device_descriptor/number_of_secure_wpa
- /sys/bus/platform/drivers/ufshcd/*/device_descriptor/psa_max_data_size
- /sys/bus/platform/devices/*.ufs/device_descriptor/psa_max_data_size
- /sys/bus/platform/drivers/ufshcd/*/device_descriptor/psa_state_timeout
- /sys/bus/platform/devices/*.ufs/device_descriptor/psa_state_timeout
- /sys/bus/platform/drivers/ufshcd/*/interconnect_descriptor/unipro_version
- /sys/bus/platform/devices/*.ufs/interconnect_descriptor/unipro_version
- /sys/bus/platform/drivers/ufshcd/*/interconnect_descriptor/mphy_version
- /sys/bus/platform/devices/*.ufs/interconnect_descriptor/mphy_version
- /sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/raw_device_capacity
- /sys/bus/platform/devices/*.ufs/geometry_descriptor/raw_device_capacity
- /sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/max_number_of_luns
- /sys/bus/platform/devices/*.ufs/geometry_descriptor/max_number_of_luns
- /sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/segment_size
- /sys/bus/platform/devices/*.ufs/geometry_descriptor/segment_size
- /sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/allocation_unit_size
- /sys/bus/platform/devices/*.ufs/geometry_descriptor/allocation_unit_size
- /sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/min_addressable_block_size
- /sys/bus/platform/devices/*.ufs/geometry_descriptor/min_addressable_block_size
- /sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/optimal_read_block_size
- /sys/bus/platform/devices/*.ufs/geometry_descriptor/optimal_read_block_size
- /sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/optimal_write_block_size
- /sys/bus/platform/devices/*.ufs/geometry_descriptor/optimal_write_block_size
- /sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/max_in_buffer_size
- /sys/bus/platform/devices/*.ufs/geometry_descriptor/max_in_buffer_size
- /sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/max_out_buffer_size
- /sys/bus/platform/devices/*.ufs/geometry_descriptor/max_out_buffer_size
- /sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/rpmb_rw_size
- /sys/bus/platform/devices/*.ufs/geometry_descriptor/rpmb_rw_size
- /sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/dyn_capacity_resource_policy
- /sys/bus/platform/devices/*.ufs/geometry_descriptor/dyn_capacity_resource_policy

- /sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/data_ordering
- /sys/bus/platform/devices/*.ufs/geometry_descriptor/data_ordering
- /sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/max_number_of_contexts
- /sys/bus/platform/devices/*.ufs/geometry_descriptor/max_number_of_contexts
- /sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/sys_data_tag_unit_size
- /sys/bus/platform/devices/*.ufs/geometry_descriptor/sys_data_tag_unit_size
- /sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/sys_data_tag_resource_size
- /sys/bus/platform/devices/*.ufs/geometry_descriptor/sys_data_tag_resource_size
- /sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/secure_removal_types
- /sys/bus/platform/devices/*.ufs/geometry_descriptor/secure_removal_types
- /sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/memory_types
- /sys/bus/platform/devices/*.ufs/geometry_descriptor/memory_types
- /sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/*_memory_max_alloc_units
- /sys/bus/platform/devices/*.ufs/geometry_descriptor/*_memory_max_alloc_units
- /sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/*_memory_capacity_adjustment_factor
- /sys/bus/platform/devices/*.ufs/geometry_descriptor/*_memory_capacity_adjustment_factor
- /sys/bus/platform/drivers/ufshcd/*/health_descriptor/eol_info
- /sys/bus/platform/devices/*.ufs/health_descriptor/eol_info
- /sys/bus/platform/drivers/ufshcd/*/health_descriptor/life_time_estimation_a
- /sys/bus/platform/devices/*.ufs/health_descriptor/life_time_estimation_a
- /sys/bus/platform/drivers/ufshcd/*/health_descriptor/life_time_estimation_b
- /sys/bus/platform/devices/*.ufs/health_descriptor/life_time_estimation_b
- /sys/bus/platform/drivers/ufshcd/*/power_descriptor/active_icc_levels_vcc*
- /sys/bus/platform/devices/*.ufs/power_descriptor/active_icc_levels_vcc*
- /sys/bus/platform/drivers/ufshcd/*/string_descriptors/manufacturer_name
- /sys/bus/platform/devices/*.ufs/string_descriptors/manufacturer_name
- /sys/bus/platform/drivers/ufshcd/*/string_descriptors/product_name
- /sys/bus/platform/devices/*.ufs/string_descriptors/product_name
- /sys/bus/platform/drivers/ufshcd/*/string_descriptors/oem_id
- /sys/bus/platform/devices/*.ufs/string_descriptors/oem_id
- /sys/bus/platform/drivers/ufshcd/*/string_descriptors/serial_number
- /sys/bus/platform/devices/*.ufs/string_descriptors/serial_number
- /sys/bus/platform/drivers/ufshcd/*/string_descriptors/product_revision
- /sys/bus/platform/devices/*.ufs/string_descriptors/product_revision

- /sys/class/scsi_device/*/device/unit_descriptor/boot_lun_id
- /sys/class/scsi_device/*/device/unit_descriptor/lun_write_protect
- /sys/class/scsi_device/*/device/unit_descriptor/lun_queue_depth
- /sys/class/scsi_device/*/device/unit_descriptor/psa_sensitive
- /sys/class/scsi_device/*/device/unit_descriptor/lun_memory_type
- /sys/class/scsi_device/*/device/unit_descriptor/data_reliability
- /sys/class/scsi_device/*/device/unit_descriptor/logical_block_size
- /sys/class/scsi_device/*/device/unit_descriptor/logical_block_count
- /sys/class/scsi_device/*/device/unit_descriptor/erase_block_size
- /sys/class/scsi_device/*/device/unit_descriptor/provisioning_type
- /sys/class/scsi_device/*/device/unit_descriptor/physical_memory_resourse_count
- /sys/class/scsi_device/*/device/unit_descriptor/context_capabilities
- /sys/class/scsi_device/*/device/unit_descriptor/large_unit_granularity
- /sys/bus/platform/drivers/ufshcd/*/flags/device_init
- /sys/bus/platform/devices/*.ufs/flags/device_init
- /sys/bus/platform/drivers/ufshcd/*/flags/permanent_wpe
- /sys/bus/platform/devices/*.ufs/flags/permanent_wpe
- /sys/bus/platform/drivers/ufshcd/*/flags/power_on_wpe
- /sys/bus/platform/devices/*.ufs/flags/power_on_wpe
- /sys/bus/platform/drivers/ufshcd/*/flags/bkops_enable
- /sys/bus/platform/devices/*.ufs/flags/bkops_enable
- /sys/bus/platform/drivers/ufshcd/*/flags/life_span_mode_enable
- /sys/bus/platform/devices/*.ufs/flags/life_span_mode_enable
- /sys/bus/platform/drivers/ufshcd/*/flags/phy_resource_removal
- /sys/bus/platform/devices/*.ufs/flags/phy_resource_removal
- /sys/bus/platform/drivers/ufshcd/*/flags/busy_rtc
- /sys/bus/platform/devices/*.ufs/flags/busy_rtc
- /sys/bus/platform/drivers/ufshcd/*/flags/disable_fw_update
- /sys/bus/platform/devices/*.ufs/flags/disable_fw_update
- /sys/bus/platform/drivers/ufshcd/*/attributes/boot_lun_enabled
- /sys/bus/platform/devices/*.ufs/attributes/boot_lun_enabled
- /sys/bus/platform/drivers/ufshcd/*/attributes/current_power_mode
- /sys/bus/platform/devices/*.ufs/attributes/current_power_mode
- /sys/bus/platform/drivers/ufshcd/*/attributes/active_icc_level

- /sys/bus/platform/devices/*.ufs/attributes/active_icc_level
- /sys/bus/platform/drivers/ufshcd/*/attributes/ooo_data_enabled
- /sys/bus/platform/devices/*.ufs/attributes/ooo_data_enabled
- /sys/bus/platform/drivers/ufshcd/*/attributes/bkops_status
- /sys/bus/platform/devices/*.ufs/attributes/bkops_status
- /sys/bus/platform/drivers/ufshcd/*/attributes/purge_status
- /sys/bus/platform/devices/*.ufs/attributes/purge_status
- /sys/bus/platform/drivers/ufshcd/*/attributes/max_data_in_size
- /sys/bus/platform/devices/*.ufs/attributes/max_data_in_size
- /sys/bus/platform/drivers/ufshcd/*/attributes/max_data_out_size
- /sys/bus/platform/devices/*.ufs/attributes/max_data_out_size
- /sys/bus/platform/drivers/ufshcd/*/attributes/reference_clock_frequency
- /sys/bus/platform/devices/*.ufs/attributes/reference_clock_frequency
- /sys/bus/platform/drivers/ufshcd/*/attributes/configuration_descriptor_lock
- /sys/bus/platform/devices/*.ufs/attributes/configuration_descriptor_lock
- /sys/bus/platform/drivers/ufshcd/*/attributes/max_number_of_rtt
- /sys/bus/platform/devices/*.ufs/attributes/max_number_of_rtt
- /sys/bus/platform/drivers/ufshcd/*/attributes/exception_event_control
- /sys/bus/platform/devices/*.ufs/attributes/exception_event_control
- /sys/bus/platform/drivers/ufshcd/*/attributes/exception_event_status
- /sys/bus/platform/devices/*.ufs/attributes/exception_event_status
- /sys/bus/platform/drivers/ufshcd/*/attributes/ffu_status
- /sys/bus/platform/devices/*.ufs/attributes/ffu_status
- /sys/bus/platform/drivers/ufshcd/*/attributes/psa_state
- /sys/bus/platform/devices/*.ufs/attributes/psa_state
- /sys/bus/platform/drivers/ufshcd/*/attributes/psa_data_size
- /sys/bus/platform/devices/*.ufs/attributes/psa_data_size
- /sys/class/scsi_device/*/device/dyn_cap_needed
- /sys/bus/platform/drivers/ufshcd/*/rpm_lvl
- /sys/bus/platform/devices/*.ufs/rpm_lvl
- /sys/bus/platform/drivers/ufshcd/*/rpm_target_dev_state
- /sys/bus/platform/devices/*.ufs/rpm_target_dev_state
- /sys/bus/platform/drivers/ufshcd/*/rpm_target_link_state
- /sys/bus/platform/devices/*.ufs/rpm_target_link_state

- */sys/bus/platform/drivers/ufshcd/*/spm_lvl*
- */sys/bus/platform/devices/*.ufs/spm_lvl*
- */sys/bus/platform/drivers/ufshcd/*/spm_target_dev_state*
- */sys/bus/platform/devices/*.ufs/spm_target_dev_state*
- */sys/bus/platform/drivers/ufshcd/*/spm_target_link_state*
- */sys/bus/platform/devices/*.ufs/spm_target_link_state*
- */sys/bus/platform/drivers/ufshcd/*/monitor/monitor_enable*
- */sys/bus/platform/devices/*.ufs/monitor/monitor_enable*
- */sys/bus/platform/drivers/ufshcd/*/monitor/monitor_chunk_size*
- */sys/bus/platform/devices/*.ufs/monitor/monitor_chunk_size*
- */sys/bus/platform/drivers/ufshcd/*/monitor/read_total_sectors*
- */sys/bus/platform/devices/*.ufs/monitor/read_total_sectors*
- */sys/bus/platform/drivers/ufshcd/*/monitor/read_total_busy*
- */sys/bus/platform/devices/*.ufs/monitor/read_total_busy*
- */sys/bus/platform/drivers/ufshcd/*/monitor/read_nr_requests*
- */sys/bus/platform/devices/*.ufs/monitor/read_nr_requests*
- */sys/bus/platform/drivers/ufshcd/*/monitor/read_req_latency_max*
- */sys/bus/platform/devices/*.ufs/monitor/read_req_latency_max*
- */sys/bus/platform/drivers/ufshcd/*/monitor/read_req_latency_min*
- */sys/bus/platform/devices/*.ufs/monitor/read_req_latency_min*
- */sys/bus/platform/drivers/ufshcd/*/monitor/read_req_latency_avg*
- */sys/bus/platform/devices/*.ufs/monitor/read_req_latency_avg*
- */sys/bus/platform/drivers/ufshcd/*/monitor/read_req_latency_sum*
- */sys/bus/platform/devices/*.ufs/monitor/read_req_latency_sum*
- */sys/bus/platform/drivers/ufshcd/*/monitor/write_total_sectors*
- */sys/bus/platform/devices/*.ufs/monitor/write_total_sectors*
- */sys/bus/platform/drivers/ufshcd/*/monitor/write_total_busy*
- */sys/bus/platform/devices/*.ufs/monitor/write_total_busy*
- */sys/bus/platform/drivers/ufshcd/*/monitor/write_nr_requests*
- */sys/bus/platform/devices/*.ufs/monitor/write_nr_requests*
- */sys/bus/platform/drivers/ufshcd/*/monitor/write_req_latency_max*
- */sys/bus/platform/devices/*.ufs/monitor/write_req_latency_max*
- */sys/bus/platform/drivers/ufshcd/*/monitor/write_req_latency_min*
- */sys/bus/platform/devices/*.ufs/monitor/write_req_latency_min*

- /sys/bus/platform/drivers/ufshcd/*/monitor/write_req_latency_avg
- /sys/bus/platform/devices/*.ufs/monitor/write_req_latency_avg
- /sys/bus/platform/drivers/ufshcd/*/monitor/write_req_latency_sum
- /sys/bus/platform/devices/*.ufs/monitor/write_req_latency_sum
- /sys/bus/platform/drivers/ufshcd/*/power_info/lane
- /sys/bus/platform/devices/*.ufs/power_info/lane
- /sys/bus/platform/drivers/ufshcd/*/power_info/mode
- /sys/bus/platform/devices/*.ufs/power_info/mode
- /sys/bus/platform/drivers/ufshcd/*/power_info/rate
- /sys/bus/platform/devices/*.ufs/power_info/rate
- /sys/bus/platform/drivers/ufshcd/*/power_info/gear
- /sys/bus/platform/devices/*.ufs/power_info/gear
- /sys/bus/platform/drivers/ufshcd/*/power_info/dev_pm
- /sys/bus/platform/devices/*.ufs/power_info/dev_pm
- /sys/bus/platform/drivers/ufshcd/*/power_info/link_state
- /sys/bus/platform/devices/*.ufs/power_info/link_state
- /sys/bus/platform/drivers/ufshcd/*/device_descriptor/wb_presv_us_en
- /sys/bus/platform/devices/*.ufs/device_descriptor/wb_presv_us_en
- /sys/bus/platform/drivers/ufshcd/*/device_descriptor/wb_shared_alloc_units
- /sys/bus/platform/devices/*.ufs/device_descriptor/wb_shared_alloc_units
- /sys/bus/platform/drivers/ufshcd/*/device_descriptor/wb_type
- /sys/bus/platform/devices/*.ufs/device_descriptor/wb_type
- /sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/wb_buff_cap_adj
- /sys/bus/platform/devices/*.ufs/geometry_descriptor/wb_buff_cap_adj
- /sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/wb_max_alloc_units
- /sys/bus/platform/devices/*.ufs/geometry_descriptor/wb_max_alloc_units
- /sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/wb_max_wb_luns
- /sys/bus/platform/devices/*.ufs/geometry_descriptor/wb_max_wb_luns
- /sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/wb_sup_red_type
- /sys/bus/platform/devices/*.ufs/geometry_descriptor/wb_sup_red_type
- /sys/bus/platform/drivers/ufshcd/*/geometry_descriptor/wb_sup_wb_type
- /sys/bus/platform/devices/*.ufs/geometry_descriptor/wb_sup_wb_type
- /sys/bus/platform/drivers/ufshcd/*/flags/wb_enable
- /sys/bus/platform/devices/*.ufs/flags/wb_enable

- */sys/bus/platform/drivers/ufshcd/*/flags/wb_flush_en*
- */sys/bus/platform/devices/*.ufs/flags/wb_flush_en*
- */sys/bus/platform/drivers/ufshcd/*/flags/wb_flush_during_h8*
- */sys/bus/platform/devices/*.ufs/flags/wb_flush_during_h8*
- */sys/bus/platform/drivers/ufshcd/*/attributes/wb_avail_buf*
- */sys/bus/platform/devices/*.ufs/attributes/wb_avail_buf*
- */sys/bus/platform/drivers/ufshcd/*/attributes/wb_cur_buf*
- */sys/bus/platform/devices/*.ufs/attributes/wb_cur_buf*
- */sys/bus/platform/drivers/ufshcd/*/attributes/wb_flush_status*
- */sys/bus/platform/devices/*.ufs/attributes/wb_flush_status*
- */sys/bus/platform/drivers/ufshcd/*/attributes/wb_life_time_est*
- */sys/bus/platform/devices/*.ufs/attributes/wb_life_time_est*
- */sys/class/scsi_device/*/device/unit_descriptor/wb_buf_alloc_units*
- */sys/bus/platform/drivers/ufshcd/*/wb_on*
- */sys/bus/platform/devices/*.ufs/wb_on*
- */sys/bus/platform/drivers/ufshcd/*/enable_wb_buf_flush*
- */sys/bus/platform/devices/*.ufs/enable_wb_buf_flush*
- */sys/bus/platform/drivers/ufshcd/*/wb_flush_threshold*
- */sys/bus/platform/devices/*.ufs/wb_flush_threshold*
- */sys/bus/platform/drivers/ufshcd/*/capabilities/*
- */sys/bus/platform/devices/*.ufs/capabilities/*
- */sys/bus/platform/drivers/ufshcd/*/capabilities/clock_scaling*
- */sys/bus/platform/devices/*.ufs/capabilities/clock_scaling*
- */sys/bus/platform/drivers/ufshcd/*/capabilities/write_booster*
- */sys/bus/platform/devices/*.ufs/capabilities/write_booster*
- */sys/bus/platform/drivers/ufshcd/*/rtc_update_ms*
- */sys/bus/platform/devices/*.ufs/rtc_update_ms*

5.2.453 File testing/sysfs-driver-w1_ds28e17

Has the following ABI:

- */sys/bus/w1/devices/19-<id>/speed*
- */sys/bus/w1/devices/19-<id>/stretch*

5.2.454 File testing/sysfs-driver-w1_therm

Has the following ABI:

- */sys/bus/w1/devices/.../alarms*
- */sys/bus/w1/devices/.../eeprom_cmd*
- */sys/bus/w1/devices/.../ext_power*
- */sys/bus/w1/devices/.../resolution*
- */sys/bus/w1/devices/.../temperature*
- */sys/bus/w1/devices/.../w1_slave*
- */sys/bus/w1/devices/w1_bus_masterXX/therm_bulk_read*
- */sys/bus/w1/devices/.../conv_time*
- */sys/bus/w1/devices/.../features*

5.2.455 File testing/sysfs-driver-wacom

Has the following ABI:

- */sys/bus/hid/devices/<bus>:<vid>:<pid>.<n>/speed*
- */sys/bus/hid/devices/<bus>:<vid>:<pid>.<n>/wacom_led/led*
- */sys/bus/hid/devices/<bus>:<vid>:<pid>.<n>/wacom_led/status0_luminance*
- */sys/bus/hid/devices/<bus>:<vid>:<pid>.<n>/wacom_led/status1_luminance*
- */sys/bus/hid/devices/<bus>:<vid>:<pid>.<n>/wacom_led/status_led0_select*
- */sys/bus/hid/devices/<bus>:<vid>:<pid>.<n>/wacom_led/status_led1_select*
- */sys/bus/hid/devices/<bus>:<vid>:<pid>.<n>/wacom_led/buttons_luminance*
- */sys/bus/hid/devices/<bus>:<vid>:<pid>.<n>/wacom_led/button<n>_rawimg*
- */sys/bus/hid/devices/<bus>:<vid>:<pid>.<n>/wacom_remote/unpair_remote*
- */sys/bus/hid/devices/<bus>:<vid>:<pid>.<n>/wacom_remote/<serial_number>/remote_mode*

5.2.456 File testing/sysfs-driver-xdata

Has the following ABI:

- */sys/class/misc/drivers/dw-xdata-pcie.<device>/write*
- */sys/class/misc/dw-xdata-pcie.<device>/read*

5.2.457 File testing/sysfs-driver-xen-blkback

Has the following ABI:

- */sys/module/xen_blkback/parameters/max_buffer_pages*
- */sys/module/xen_blkback/parameters/max_persistent_grants*
- */sys/module/xen_blkback/parameters/persistent_grant_unused_seconds*
- */sys/module/xen_blkback/parameters/buffer_squeeze_duration_ms*
- */sys/module/xen_blkback/parameters/feature_persistent*

5.2.458 File testing/sysfs-driver-xen-blkfront

Has the following ABI:

- */sys/module/xen_blkfront/parameters/max_indirect_segments*
- */sys/module/xen_blkfront/parameters/feature_persistent*

5.2.459 File testing/sysfs-driver-xilinx-tmr-manager

Has the following ABI:

- */sys/devices/platform/amba_pl/<dev>/errcnt*
- */sys/devices/platform/amba_pl/<dev>/dis_block_break*

5.2.460 File testing/sysfs-driver-zynqmp-fpga

Has the following ABI:

- */sys/bus/platform/drivers/zynqmp_fpga_manager/firmware:zynqmp-firmware:pcap/status*

5.2.461 File testing/sysfs-firmware-acpi

Has the following ABI:

- */sys/firmware/acpi/fpdt/*
- */sys/firmware/acpi/bgrt/*
- */sys/firmware/acpi/hotplug/*
- */sys/firmware/acpi/interrupts/*

5.2.462 File testing/sysfs-firmware-dmi-entries

Has the following ABI:

- */sys/firmware/dmi/entries/*

5.2.463 File testing/sysfs-firmware-dmi-tables

Has the following ABI:

- */sys/firmware/dmi/tables/*

5.2.464 File testing/sysfs-firmware-efi

Has the following ABI:

- */sys/firmware/efi/fw_vendor*
- */sys/firmware/efi/runtime*
- */sys/firmware/efi/config_table*
- */sys/firmware/efi/systab*
- */sys/firmware/efi/tables/rci2*

5.2.465 File testing/sysfs-firmware-efi-esrt

Has the following ABI:

- */sys/firmware/efi/esrt/*
- */sys/firmware/efi/esrt/fw_resource_count*
- */sys/firmware/efi/esrt/fw_resource_count_max*
- */sys/firmware/efi/esrt/fw_resource_version*
- */sys/firmware/efi/esrt/entries/entry<N>/*
- */sys/firmware/efi/esrt/entries/entry<N>/fw_type*
- */sys/firmware/efi/esrt/entries/entry<N>/fw_class*
- */sys/firmware/efi/esrt/entries/entry<N>/fw_version*
- */sys/firmware/efi/esrt/entries/entry<N>/lowest_supported_fw_version*
- */sys/firmware/efi/esrt/entries/entry<N>/capsule_flags*
- */sys/firmware/efi/esrt/entries/entry<N>/last_attempt_version*
- */sys/firmware/efi/esrt/entries/entry<N>/last_attempt_status*

5.2.466 File testing/sysfs-firmware-efi-runtime-map

Has the following ABI:

- */sys/firmware/efi/runtime-map/*

5.2.467 File testing/sysfs-firmware-gsmi

Has the following ABI:

- */sys/firmware/gsmi*

5.2.468 File testing/sysfs-firmware-initrd

Has the following ABI:

- */sys/firmware/initrd*

5.2.469 File testing/sysfs-firmware-lefi-boardinfo

Has the following ABI:

- */sys/firmware/lefi/boardinfo*

5.2.470 File testing/sysfs-firmware-log

Has the following ABI:

- */sys/firmware/log*

5.2.471 File testing/sysfs-firmware-memmap

Has the following ABI:

- */sys/firmware/memmap/*

5.2.472 File testing/sysfs-firmware-ofw

Has the following ABI:

- */sys/firmware/devicetree/**
- */sys/firmware/fdt*

5.2.473 File testing/sysfs-firmware-opal-powercap

Has the following ABI:

- */sys/firmware/opal/powercap*
- */sys/firmware/opal/powercap/system-powercap*

5.2.474 File testing/sysfs-firmware-opal-psr

Has the following ABI:

- */sys/firmware/opal/psr*
- */sys/firmware/opal/psr/cpu_to_gpu_X*

5.2.475 File testing/sysfs-firmware-opal-sensor-groups

Has the following ABI:

- */sys/firmware/opal/sensor_groups*
- */sys/firmware/opal/sensor_groups/<sensor_group_name>/clear*

5.2.476 File testing/sysfs-firmware-papr-energy-scale-info

Has the following ABI:

- */sys/firmware/papr/energy_scale_info*
- */sys/firmware/papr/energy_scale_info/<id>*
- */sys/firmware/papr/energy_scale_info/<id>/desc*
- */sys/firmware/papr/energy_scale_info/<id>/value*
- */sys/firmware/papr/energy_scale_info/<id>/value_desc*

5.2.477 File testing/sysfs-firmware-qemu_fw_cfg

Has the following ABI:

- */sys/firmware/qemu_fw_cfg/*

5.2.478 File testing/sysfs-firmware-sgi_uv

Has the following ABI:

- */sys/firmware/sgi_uv/*

5.2.479 File testing/sysfs-firmware-turris-mox-rwtm

Has the following ABI:

- */sys/firmware/turris-mox-rwtm/board_version*
- */sys/firmware/turris-mox-rwtm/mac_address**
- */sys/firmware/turris-mox-rwtm/pubkey*
- */sys/firmware/turris-mox-rwtm/ram_size*
- */sys/firmware/turris-mox-rwtm/serial_number*

5.2.480 File testing/sysfs-fs-erofs

Has the following ABI:

- */sys/fs/erofs/features/*
- */sys/fs/erofs/<disk>/sync_decompress*

5.2.481 File testing/sysfs-fs-ext4

Has the following ABI:

- */sys/fs/ext4/<disk>/mb_stats*
- */sys/fs/ext4/<disk>/mb_group_prealloc*
- */sys/fs/ext4/<disk>/mb_max_to_scan*
- */sys/fs/ext4/<disk>/mb_min_to_scan*
- */sys/fs/ext4/<disk>/mb_order2_req*
- */sys/fs/ext4/<disk>/mb_stream_req*
- */sys/fs/ext4/<disk>/inode_readahead_blks*
- */sys/fs/ext4/<disk>/delayed_allocation_blocks*
- */sys/fs/ext4/<disk>/lifetime_write_kbytes*
- */sys/fs/ext4/<disk>/session_write_kbytes*
- */sys/fs/ext4/<disk>/inode_goal*
- */sys/fs/ext4/<disk>/max_writeback_mb_bump*
- */sys/fs/ext4/<disk>/extent_max_zeroout_kb*
- */sys/fs/ext4/<disk>/journal_task*

5.2.482 File testing/sysfs-fs-f2fs

Has the following ABI:

- */sys/fs/f2fs/<disk>/gc_max_sleep_time*
- */sys/fs/f2fs/<disk>/gc_min_sleep_time*
- */sys/fs/f2fs/<disk>/gc_no_gc_sleep_time*
- */sys/fs/f2fs/<disk>/gc_idle*
- */sys/fs/f2fs/<disk>/reclaim_segments*
- */sys/fs/f2fs/<disk>/main_blkaddr*
- */sys/fs/f2fs/<disk>/ipu_policy*
- */sys/fs/f2fs/<disk>/min_ipu_util*
- */sys/fs/f2fs/<disk>/min_fsync_blocks*
- */sys/fs/f2fs/<disk>/min_seq_blocks*
- */sys/fs/f2fs/<disk>/min_hot_blocks*
- */sys/fs/f2fs/<disk>/min_ssr_sections*
- */sys/fs/f2fs/<disk>/max_small_discards*
- */sys/fs/f2fs/<disk>/max_ordered_discard*
- */sys/fs/f2fs/<disk>/max_discard_request*
- */sys/fs/f2fs/<disk>/min_discard_issue_time*
- */sys/fs/f2fs/<disk>/mid_discard_issue_time*
- */sys/fs/f2fs/<disk>/max_discard_issue_time*
- */sys/fs/f2fs/<disk>/discard_granularity*
- */sys/fs/f2fs/<disk>/umount_discard_timeout*
- */sys/fs/f2fs/<disk>/pending_discard*
- */sys/fs/f2fs/<disk>/max_victim_search*
- */sys/fs/f2fs/<disk>/migration_granularity*
- */sys/fs/f2fs/<disk>/dir_level*
- */sys/fs/f2fs/<disk>/ram_thresh*
- */sys/fs/f2fs/<disk>/cp_interval*
- */sys/fs/f2fs/<disk>/idle_interval*
- */sys/fs/f2fs/<disk>/discard_idle_interval*
- */sys/fs/f2fs/<disk>/gc_idle_interval*
- */sys/fs/f2fs/<disk>/iostat_enable*
- */sys/fs/f2fs/<disk>/ra_nid_pages*
- */sys/fs/f2fs/<disk>/dirty_nats_ratio*

- */sys/fs/f2fs/<disk>/lifetime_write_kbytes*
- */sys/fs/f2fs/<disk>/features*
- */sys/fs/f2fs/<disk>/feature_list/*
- */sys/fs/f2fs/features/*
- */sys/fs/f2fs/<disk>/inject_rate*
- */sys/fs/f2fs/<disk>/inject_type*
- */sys/fs/f2fs/<disk>/dirty_segments*
- */sys/fs/f2fs/<disk>/reserved_blocks*
- */sys/fs/f2fs/<disk>/current_reserved_blocks*
- */sys/fs/f2fs/<disk>/gc_urgent*
- */sys/fs/f2fs/<disk>/gc_urgent_sleep_time*
- */sys/fs/f2fs/<disk>/readdir_ra*
- */sys/fs/f2fs/<disk>/gc_pin_file_thresh*
- */sys/fs/f2fs/<disk>/extension_list*
- */sys/fs/f2fs/<disk>/unusable*
- */sys/fs/f2fs/<disk>/encoding*
- */sys/fs/f2fs/<disk>/free_segments*
- */sys/fs/f2fs/<disk>/cp_foreground_calls*
- */sys/fs/f2fs/<disk>/cp_background_calls*
- */sys/fs/f2fs/<disk>/gc_foreground_calls*
- */sys/fs/f2fs/<disk>/gc_background_calls*
- */sys/fs/f2fs/<disk>/moved_blocks_foreground*
- */sys/fs/f2fs/<disk>/moved_blocks_background*
- */sys/fs/f2fs/<disk>/avg_vblocks*
- */sys/fs/f2fs/<disk>/mounted_time_sec*
- */sys/fs/f2fs/<disk>/data_io_flag*
- */sys/fs/f2fs/<disk>/node_io_flag*
- */sys/fs/f2fs/<disk>/iostat_period_ms*
- */sys/fs/f2fs/<disk>/max_io_bytes*
- */sys/fs/f2fs/<disk>/stat/sb_status*
- */sys/fs/f2fs/<disk>/stat/cp_status*
- */sys/fs/f2fs/<disk>/stat/issued_discard*
- */sys/fs/f2fs/<disk>/stat/queued_discard*
- */sys/fs/f2fs/<disk>/stat/undiscard_blks*

- */sys/fs/f2fs/<disk>/ckpt_thread_ioprio*
- */sys/fs/f2fs/<disk>/ovp_segments*
- */sys/fs/f2fs/<disk>/compr_written_block*
- */sys/fs/f2fs/<disk>/compr_saved_block*
- */sys/fs/f2fs/<disk>/compr_new_inode*
- */sys/fs/f2fs/<disk>/atgc_candidate_ratio*
- */sys/fs/f2fs/<disk>/atgc_candidate_count*
- */sys/fs/f2fs/<disk>/atgc_age_weight*
- */sys/fs/f2fs/<disk>/atgc_age_threshold*
- */sys/fs/f2fs/<disk>/gc_reclaimed_segments*
- */sys/fs/f2fs/<disk>/gc_segment_mode*
- */sys/fs/f2fs/<disk>/seq_file_ra_mul*
- */sys/fs/f2fs/<disk>/max_fragment_chunk*
- */sys/fs/f2fs/<disk>/max_fragment_hole*
- */sys/fs/f2fs/<disk>/gc_remaining_trials*
- */sys/fs/f2fs/<disk>/max_roll_forward_node_blocks*
- */sys/fs/f2fs/<disk>/unusable_blocks_per_sec*
- */sys/fs/f2fs/<disk>/current_atomic_write*
- */sys/fs/f2fs/<disk>/peak_atomic_write*
- */sys/fs/f2fs/<disk>/committed_atomic_block*
- */sys/fs/f2fs/<disk>/revoked_atomic_block*
- */sys/fs/f2fs/<disk>/gc_mode*
- */sys/fs/f2fs/<disk>/discard_urgent_util*
- */sys/fs/f2fs/<disk>/hot_data_age_threshold*
- */sys/fs/f2fs/<disk>/warm_data_age_threshold*
- */sys/fs/f2fs/<disk>/fault_rate*
- */sys/fs/f2fs/<disk>/fault_type*
- */sys/fs/f2fs/<disk>/discard_io_aware_gran*
- */sys/fs/f2fs/<disk>/last_age_weight*
- */sys/fs/f2fs/<disk>/compress_watermark*
- */sys/fs/f2fs/<disk>/compress_percent*
- */sys/fs/f2fs/<disk>/discard_io_aware*

5.2.483 File testing/sysfs-fs-nilfs2

Has the following ABI:

- */sys/fs/nilfs2/features/revision*
- */sys/fs/nilfs2/features/README*
- */sys/fs/nilfs2/<device>/revision*
- */sys/fs/nilfs2/<device>/blocksize*
- */sys/fs/nilfs2/<device>/device_size*
- */sys/fs/nilfs2/<device>/free_blocks*
- */sys/fs/nilfs2/<device>/uuid*
- */sys/fs/nilfs2/<device>/volume_name*
- */sys/fs/nilfs2/<device>/README*
- */sys/fs/nilfs2/<device>/superblock/sb_write_time*
- */sys/fs/nilfs2/<device>/superblock/sb_write_time_secs*
- */sys/fs/nilfs2/<device>/superblock/sb_write_count*
- */sys/fs/nilfs2/<device>/superblock/sb_update_frequency*
- */sys/fs/nilfs2/<device>/superblock/README*
- */sys/fs/nilfs2/<device>/segctor/last_pseg_block*
- */sys/fs/nilfs2/<device>/segctor/last_seg_sequence*
- */sys/fs/nilfs2/<device>/segctor/last_seg_checkpoint*
- */sys/fs/nilfs2/<device>/segctor/current_seg_sequence*
- */sys/fs/nilfs2/<device>/segctor/current_last_full_seg*
- */sys/fs/nilfs2/<device>/segctor/next_full_seg*
- */sys/fs/nilfs2/<device>/segctor/next_pseg_offset*
- */sys/fs/nilfs2/<device>/segctor/next_checkpoint*
- */sys/fs/nilfs2/<device>/segctor/last_seg_write_time*
- */sys/fs/nilfs2/<device>/segctor/last_seg_write_time_secs*
- */sys/fs/nilfs2/<device>/segctor/last_nongc_write_time*
- */sys/fs/nilfs2/<device>/segctor/last_nongc_write_time_secs*
- */sys/fs/nilfs2/<device>/segctor/dirty_data_blocks_count*
- */sys/fs/nilfs2/<device>/segctor/README*
- */sys/fs/nilfs2/<device>/segments/segments_number*
- */sys/fs/nilfs2/<device>/segments/blocks_per_segment*
- */sys/fs/nilfs2/<device>/segments/clean_segments*
- */sys/fs/nilfs2/<device>/segments/dirty_segments*

- */sys/fs/nilfs2/<device>/segments/README*
- */sys/fs/nilfs2/<device>/checkpoints/checkpoints_number*
- */sys/fs/nilfs2/<device>/checkpoints/snapshots_number*
- */sys/fs/nilfs2/<device>/checkpoints/last_seg_checkpoint*
- */sys/fs/nilfs2/<device>/checkpoints/next_checkpoint*
- */sys/fs/nilfs2/<device>/checkpoints/README*
- */sys/fs/nilfs2/<device>/mounted_snapshots/README*
- */sys/fs/nilfs2/<device>/mounted_snapshots/<id>/inodes_count*
- */sys/fs/nilfs2/<device>/mounted_snapshots/<id>/blocks_count*
- */sys/fs/nilfs2/<device>/mounted_snapshots/<id>/README*

5.2.484 File testing/sysfs-fs-ubifs

Has the following ABI:

- */sys/fs/ubifsX_Y/error_magic*
- */sys/fs/ubifsX_Y/error_node*
- */sys/fs/ubifsX_Y/error_crc*

5.2.485 File testing/sysfs-fs-xfs

Has the following ABI:

- */sys/fs/xfs/<disk>/log/log_head_lsn*
- */sys/fs/xfs/<disk>/log/log_tail_lsn*
- */sys/fs/xfs/<disk>/log/reserve_grant_head*
- */sys/fs/xfs/<disk>/log/write_grant_head*

5.2.486 File testing/sysfs-hypervisor-xen

Has the following ABI:

- */sys/hypervisor/guest_type*
- */sys/hypervisor/pmu/pmu_mode*
- */sys/hypervisor/pmu/pmu_features*
- */sys/hypervisor/properties/buildid*

5.2.487 File testing/sysfs-ibft

Has the following ABI:

- */sys/firmware/ibft/initiator*
- */sys/firmware/ibft/targetX*
- */sys/firmware/ibft/ethernetX*
- */sys/firmware/ibft/acpi_header*

5.2.488 File testing/sysfs-kernel-address_bits

Has the following ABI:

- */sys/kernel/address_bit*

5.2.489 File testing/sysfs-kernel-boot_params

Has the following ABI:

- */sys/kernel/boot_params*

5.2.490 File testing/sysfs-kernel-btf

Has the following ABI:

- */sys/kernel/btf*
- */sys/kernel/btf/vmlinux*
- */sys/kernel/btf/<module-name>*

5.2.491 File testing/sysfs-kernel-cpu_byteorder

Has the following ABI:

- */sys/kernel/cpu_byteorder*

5.2.492 File testing/sysfs-kernel-dmabuf-buffers

Has the following ABI:

- */sys/kernel/dmabuf/buffers*
- */sys/kernel/dmabuf/buffers/<inode_number>/exporter_name*
- */sys/kernel/dmabuf/buffers/<inode_number>/size*

5.2.493 File testing/sysfs-kernel-fadump

Has the following ABI:

- */sys/kernel/fadump/**
- */sys/kernel/fadump/enabled*
- */sys/kernel/fadump/registered*
- */sys/kernel/fadump/release_mem*
- */sys/kernel/fadump/mem_reserved*

5.2.494 File testing/sysfs-kernel-fscaps

Has the following ABI:

- */sys/kernel/fscaps*

5.2.495 File testing/sysfs-kernel-iommu_groups

Has the following ABI:

- */sys/kernel/iommu_groups/*
- */sys/kernel/iommu_groups/reserved_regions*
- */sys/kernel/iommu_groups/<grp_id>/type*

5.2.496 File testing/sysfs-kernel-irq

Has the following ABI:

- */sys/kernel/irq*
- */sys/kernel/irq/<irq>/actions*
- */sys/kernel/irq/<irq>/chip_name*
- */sys/kernel/irq/<irq>/hwirq*
- */sys/kernel/irq/<irq>/name*
- */sys/kernel/irq/<irq>/per_cpu_count*
- */sys/kernel/irq/<irq>/type*
- */sys/kernel/irq/<irq>/wakeup*

5.2.497 File testing/sysfs-kernel-livepatch

Has the following ABI:

- */sys/kernel/livepatch*
- */sys/kernel/livepatch/<patch>*
- */sys/kernel/livepatch/<patch>/enabled*
- */sys/kernel/livepatch/<patch>/transition*
- */sys/kernel/livepatch/<patch>/force*
- */sys/kernel/livepatch/<patch>/<object>*
- */sys/kernel/livepatch/<patch>/<object>/patched*
- */sys/kernel/livepatch/<patch>/<object>/<function,sympos>*

5.2.498 File testing/sysfs-kernel-mm

Has the following ABI:

- */sys/kernel/mm*

5.2.499 File testing/sysfs-kernel-mm-cma

Has the following ABI:

- */sys/kernel/mm/cma/*
- */sys/kernel/mm/cma/<cma-heap-name>/alloc_pages_success*
- */sys/kernel/mm/cma/<cma-heap-name>/alloc_pages_fail*

5.2.500 File testing/sysfs-kernel-mm-damon

Has the following ABI:

- */sys/kernel/mm/damon/*
- */sys/kernel/mm/damon/admin/*
- */sys/kernel/mm/damon/admin/kdamonds/nr_kdamonds*
- */sys/kernel/mm/damon/admin/kdamonds/<K>/state*
- */sys/kernel/mm/damon/admin/kdamonds/<K>/pid*
- */sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/nr_contexts*
- */sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/avail_operations*
- */sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/operations*
- */sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/monitoring_attrs/intervals/sample_*
- */sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/monitoring_attrs/intervals/aggr_us*
- */sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/monitoring_attrs/intervals/update_*

- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/monitoring_attrs/nr_regions/min
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/monitoring_attrs/nr_regions/max
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/targets/nr_targets
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/targets/<T>/pid_target
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/targets/<T>/regions/nr_regions
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/targets/<T>/regions/<R>/start
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/targets/<T>/regions/<R>/end
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/nr_schemes
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/action
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/apply_interval_us
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/access_pattern/sz/mi
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/access_pattern/sz/ma
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/access_pattern/nr_ac
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/access_pattern/nr_ac
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/access_pattern/age/m
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/access_pattern/age/m
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/quotas/ms
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/quotas/bytes
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/quotas/reset_interva
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/quotas/goals/nr_goal
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/quotas/goals/<G>/ta
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/quotas/goals/<G>/cu
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/quotas/weights/sz_pe
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/quotas/weights/nr_ac
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/quotas/weights/age_
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/watermarks/metric
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/watermarks/interval
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/watermarks/high
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/watermarks/mid
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/watermarks/low
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/filters/nr_filters
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/filters/<F>/type
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/filters/<F>/memcg_p
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/filters/<F>/addr_sta

- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/filters/<F>/addr_end
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/filters/<F>/target_id
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/filters/<F>/matching
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/stats/nr_tried
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/stats/sz_tried
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/stats/nr_applied
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/stats/sz_applied
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/stats/qt_exceeds
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/tried_regions/total_b
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/tried_regions/<R>/s
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/tried_regions/<R>/e
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/tried_regions/<R>/n
- /sys/kernel/mm/damon/admin/kdamonds/<K>/contexts/<C>/schemes/<S>/tried_regions/<R>/a

5.2.501 File testing/sysfs-kernel-mm-hugepages

Has the following ABI:

- /sys/kernel/mm/hugepages/

5.2.502 File testing/sysfs-kernel-mm-ksm

Has the following ABI:

- /sys/kernel/mm/ksm
- /sys/kernel/mm/ksm/full_scans
- /sys/kernel/mm/ksm/pages_shared
- /sys/kernel/mm/ksm/pages_sharing
- /sys/kernel/mm/ksm/pages_to_scan
- /sys/kernel/mm/ksm/pages_unshared
- /sys/kernel/mm/ksm/pages_volatile
- /sys/kernel/mm/ksm/run
- /sys/kernel/mm/ksm/sleep_millisecs
- /sys/kernel/mm/ksm/merge_across_nodes
- /sys/kernel/mm/ksm/general_profit

5.2.503 File testing/sysfs-kernel-mm-memory-tiers

Has the following ABI:

- */sys/devices/virtual/memory_tiering/*
- */sys/devices/virtual/memory_tiering/memory_tierN/*

5.2.504 File testing/sysfs-kernel-mm-numa

Has the following ABI:

- */sys/kernel/mm/numa/*
- */sys/kernel/mm/numa/demotion_enabled*

5.2.505 File testing/sysfs-kernel-mm-swap

Has the following ABI:

- */sys/kernel/mm/swap/*
- */sys/kernel/mm/swap/vma_ra_enabled*

5.2.506 File testing/sysfs-kernel-oops_count

Has the following ABI:

- */sys/kernel/oops_count*

5.2.507 File testing/sysfs-kernel-reboot

Has the following ABI:

- */sys/kernel/reboot*
- */sys/kernel/reboot/mode*
- */sys/kernel/reboot/type*
- */sys/kernel/reboot/cpu*
- */sys/kernel/reboot/force*

5.2.508 File testing/sysfs-kernel-slab

Has the following ABI:

- */sys/kernel/slab*
- */sys/kernel/slab/<cache>/aliases*
- */sys/kernel/slab/<cache>/align*
- */sys/kernel/slab/<cache>/alloc_calls*

- */sys/kernel/slab/<cache>/alloc_fastpath*
- */sys/kernel/slab/<cache>/alloc_from_partial*
- */sys/kernel/slab/<cache>/alloc_refill*
- */sys/kernel/slab/<cache>/alloc_slab*
- */sys/kernel/slab/<cache>/alloc_slowpath*
- */sys/kernel/slab/<cache>/cache_dma*
- */sys/kernel/slab/<cache>/cpu_slabs*
- */sys/kernel/slab/<cache>/cpuslab_flush*
- */sys/kernel/slab/<cache>/ctor*
- */sys/kernel/slab/<cache>/deactivate_empty*
- */sys/kernel/slab/<cache>/deactivate_full*
- */sys/kernel/slab/<cache>/deactivate_remote_frees*
- */sys/kernel/slab/<cache>/deactivate_to_head*
- */sys/kernel/slab/<cache>/deactivate_to_tail*
- */sys/kernel/slab/<cache>/destroy_by_rcu*
- */sys/kernel/slab/<cache>/free_add_partial*
- */sys/kernel/slab/<cache>/free_calls*
- */sys/kernel/slab/<cache>/free_fastpath*
- */sys/kernel/slab/<cache>/free_frozen*
- */sys/kernel/slab/<cache>/free_remove_partial*
- */sys/kernel/slab/<cache>/free_slab*
- */sys/kernel/slab/<cache>/free_slowpath*
- */sys/kernel/slab/<cache>/hwcache_align*
- */sys/kernel/slab/<cache>/min_partial*
- */sys/kernel/slab/<cache>/object_size*
- */sys/kernel/slab/<cache>/objects*
- */sys/kernel/slab/<cache>/objects_partial*
- */sys/kernel/slab/<cache>/objs_per_slab*
- */sys/kernel/slab/<cache>/order*
- */sys/kernel/slab/<cache>/order_fallback*
- */sys/kernel/slab/<cache>/partial*
- */sys/kernel/slab/<cache>/poison*
- */sys/kernel/slab/<cache>/reclaim_account*
- */sys/kernel/slab/<cache>/red_zone*

- */sys/kernel/slab/<cache>/remote_node_defrag_ratio*
- */sys/kernel/slab/<cache>/sanity_checks*
- */sys/kernel/slab/<cache>/shrink*
- */sys/kernel/slab/<cache>/slab_size*
- */sys/kernel/slab/<cache>/slabs*
- */sys/kernel/slab/<cache>/store_user*
- */sys/kernel/slab/<cache>/total_objects*
- */sys/kernel/slab/<cache>/trace*
- */sys/kernel/slab/<cache>/validate*
- */sys/kernel/slab/<cache>/usersize*
- */sys/kernel/slab/<cache>/slabs_cpu_partial*
- */sys/kernel/slab/<cache>/cpu_partial*

5.2.509 File testing/sysfs-kernel-vmcoreinfo

Has the following ABI:

- */sys/kernel/vmcoreinfo*

5.2.510 File testing/sysfs-kernel-warn_count

Has the following ABI:

- */sys/kernel/warn_count*

5.2.511 File testing/sysfs-mce

Has the following ABI:

- */sys/devices/system/machinecheck/machinecheckX/*
- */sys/devices/system/machinecheck/machinecheckX/bank<Y>*
- */sys/devices/system/machinecheck/machinecheckX/check_interval*
- */sys/devices/system/machinecheck/machinecheckX/trigger*
- */sys/devices/system/machinecheck/machinecheckX/monarch_timeout*
- */sys/devices/system/machinecheck/machinecheckX/ignore_ce*
- */sys/devices/system/machinecheck/machinecheckX/dont_log_ce*
- */sys/devices/system/machinecheck/machinecheckX/cmci_disabled*

5.2.512 File testing/sysfs-memory-page-offline

Has the following ABI:

- */sys/devices/system/memory/soft_offline_page*
- */sys/devices/system/memory/hard_offline_page*

5.2.513 File testing/sysfs-module

Has the following ABI:

- */sys/module/pch_phub/drivers/.../pch_mac*
- */sys/module/pch_phub/drivers/.../pch_firmware*
- */sys/module/ehci_hcd/drivers/.../uframe_periodic_max*
- */sys/module/*/{coresize,initsize}*
- */sys/module/*/initstate*
- */sys/module/*/taint*
- */sys/module/grant_table/parameters/free_per_iteration*

5.2.514 File testing/sysfs-nvmem-cells

Has the following ABI:

- */sys/bus/nvmem/devices/.../cells/<cell-name>*

5.2.515 File testing/sysfs-ocfs2

Has the following ABI:

- */sys/fs/ocfs2/*
- */sys/fs/ocfs2/max_locking_protocol*
- */sys/fs/ocfs2/loaded_cluster_plugins*
- */sys/fs/ocfs2/active_cluster_plugin*
- */sys/fs/ocfs2/cluster_stack*

5.2.516 File testing/sysfs-platform-asus-laptop

Has the following ABI:

- */sys/devices/platform/asus_laptop/display*
- */sys/devices/platform/asus_laptop/gps*
- */sys/devices/platform/asus_laptop/ledd*
- */sys/devices/platform/asus_laptop/bluetooth*
- */sys/devices/platform/asus_laptop/wlan*

- */sys/devices/platform/asus_laptop/wimax*
- */sys/devices/platform/asus_laptop/wwan*

5.2.517 File testing/sysfs-platform-asus-wmi

Has the following ABI:

- */sys/devices/platform/<platform>/cpufv*
- */sys/devices/platform/<platform>/camera*
- */sys/devices/platform/<platform>/cardr*
- */sys/devices/platform/<platform>/touchpad*
- */sys/devices/platform/<platform>/lid_resume*
- */sys/devices/platform/<platform>/fan_boost_mode*
- */sys/devices/platform/<platform>/throttle_thermal_policy*
- */sys/devices/platform/<platform>/gpu_mux_mode*
- */sys/devices/platform/<platform>/dgpu_disable*
- */sys/devices/platform/<platform>/egpu_enable*
- */sys/devices/platform/<platform>/panel_od*
- */sys/devices/platform/<platform>/charge_mode*
- */sys/devices/platform/<platform>/egpu_connected*
- */sys/devices/platform/<platform>/mini_led_mode*
- */sys/devices/platform/<platform>/ppt_pl1_spl*
- */sys/devices/platform/<platform>/ppt_pl2_sppt*
- */sys/devices/platform/<platform>/ppt_fppt*
- */sys/devices/platform/<platform>/ppt_apu_sppt*
- */sys/devices/platform/<platform>/ppt_platform_sppt*
- */sys/devices/platform/<platform>/nv_dynamic_boost*
- */sys/devices/platform/<platform>/nv_temp_target*

5.2.518 File testing/sysfs-platform-at91

Has the following ABI:

- */sys/devices/platform/at91_can/net/<iface>/mb0_id*

5.2.519 File testing/sysfs-platform-brcmstb-gisb-arb

Has the following ABI:

- */sys/devices/../../gisb_arb_timeout*

5.2.520 File testing/sysfs-platform-brcmstb-memc

Has the following ABI:

- */sys/bus/platform/devices/*/srpd*
- */sys/bus/platform/devices/*/frequency*

5.2.521 File testing/sysfs-platform-chipidea-usb-otg

Has the following ABI:

- */sys/bus/platform/devices/ci_hdrc.0/inputs/a_bus_req*
- */sys/bus/platform/devices/ci_hdrc.0/inputs/a_bus_drop*
- */sys/bus/platform/devices/ci_hdrc.0/inputs/b_bus_req*
- */sys/bus/platform/devices/ci_hdrc.0/inputs/a_clr_err*

5.2.522 File testing/sysfs-platform-chipidea-usb2

Has the following ABI:

- */sys/bus/platform/devices/ci_hdrc.0/role*

5.2.523 File testing/sysfs-platform-dell-laptop

Has the following ABI:

- */sys/class/leds/dell::kbd_backlight/als_enabled*
- */sys/class/leds/dell::kbd_backlight/als_setting*
- */sys/class/leds/dell::kbd_backlight/start_triggers*
- */sys/class/leds/dell::kbd_backlight/stop_timeout*

5.2.524 File testing/sysfs-platform-dell-privacy-wmi

Has the following ABI:

- */sys/bus/wmi/devices/6932965F-1671-4CEB-B988-D3AB0A901919/dell_privacy_supported_type*
- */sys/bus/wmi/devices/6932965F-1671-4CEB-B988-D3AB0A901919/dell_privacy_current_state*

5.2.525 File testing/sysfs-platform-dell-smbios

Has the following ABI:

- */sys/devices/platform/<platform>/tokens/**

5.2.526 File testing/sysfs-platform-dell-wmi-ddv

Has the following ABI:

- */sys/class/power_supply/<battery_name>/eppid*

5.2.527 File testing/sysfs-platform-dfl-fme

Has the following ABI:

- */sys/bus/platform/devices/dfl-fme.0/ports_num*
- */sys/bus/platform/devices/dfl-fme.0/bitstream_id*
- */sys/bus/platform/devices/dfl-fme.0/bitstream_metadata*
- */sys/bus/platform/devices/dfl-fme.0/cache_size*
- */sys/bus/platform/devices/dfl-fme.0/fabric_version*
- */sys/bus/platform/devices/dfl-fme.0/socket_id*
- */sys/bus/platform/devices/dfl-fme.0/errors/pcie0_errors*
- */sys/bus/platform/devices/dfl-fme.0/errors/pcie1_errors*
- */sys/bus/platform/devices/dfl-fme.0/errors/nonfatal_errors*
- */sys/bus/platform/devices/dfl-fme.0/errors/catfatal_errors*
- */sys/bus/platform/devices/dfl-fme.0/errors/inject_errors*
- */sys/bus/platform/devices/dfl-fme.0/errors/fme_errors*
- */sys/bus/platform/devices/dfl-fme.0/errors/first_error*
- */sys/bus/platform/devices/dfl-fme.0/errors/next_error*
- */sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/name*
- */sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/temp1_input*
- */sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/temp1_max*
- */sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/temp1_crit*
- */sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/temp1_emergency*
- */sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/temp1_max_alarm*
- */sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/temp1_crit_alarm*
- */sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/temp1_max_policy*
- */sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/power1_input*
- */sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/power1_max*

- */sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/power1_crit*
- */sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/power1_max_alarm*
- */sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/power1_crit_alarm*
- */sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/power1_xeon_limit*
- */sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/power1_fpga_limit*
- */sys/bus/platform/devices/dfl-fme.0/hwmon/hwmonX/power1_ltr*

5.2.528 File testing/sysfs-platform-dfl-port

Has the following ABI:

- */sys/bus/platform/devices/dfl-port.0/id*
- */sys/bus/platform/devices/dfl-port.0/afu_id*
- */sys/bus/platform/devices/dfl-port.0/power_state*
- */sys/bus/platform/devices/dfl-port.0/ap1_event*
- */sys/bus/platform/devices/dfl-port.0/ap2_event*
- */sys/bus/platform/devices/dfl-port.0/ltr*
- */sys/bus/platform/devices/dfl-port.0/userclk_freqcmd*
- */sys/bus/platform/devices/dfl-port.0/userclk_freqsts*
- */sys/bus/platform/devices/dfl-port.0/userclk_freqcntrcmd*
- */sys/bus/platform/devices/dfl-port.0/userclk_freqcntrsts*
- */sys/bus/platform/devices/dfl-port.0/errors/errors*
- */sys/bus/platform/devices/dfl-port.0/errors/first_error*
- */sys/bus/platform/devices/dfl-port.0/errors/first_malformed_req*

5.2.529 File testing/sysfs-platform-dptf

Has the following ABI:

- */sys/bus/platform/devices/INT3407:00/dptf_power/charger_type*
- */sys/bus/platform/devices/INT3407:00/dptf_power/adapter_rating_mw*
- */sys/bus/platform/devices/INT3407:00/dptf_power/max_platform_power_mw*
- */sys/bus/platform/devices/INT3407:00/dptf_power/platform_power_source*
- */sys/bus/platform/devices/INT3407:00/dptf_power/battery_steady_power*
- */sys/bus/platform/devices/INT3407:00/dptf_power/rest_of_platform_power_mw*
- */sys/bus/platform/devices/INT3407:00/dptf_power/prochot_confirm*
- */sys/bus/platform/devices/INT3532:00/dptf_battery/max_platform_power_mw*
- */sys/bus/platform/devices/INT3532:00/dptf_battery/max_steady_state_power_mw*

- */sys/bus/platform/devices/INT3532:00/dptf_battery/high_freq_impedance_mohm*
- */sys/bus/platform/devices/INT3532:00/dptf_battery/no_load_voltage_mv*
- */sys/bus/platform/devices/INT3532:00/dptf_battery/current_discharge_capbility_ma*
- */sys/bus/platform/devices/INTC1045:00/pch_fivr_switch_frequency/freq_mhz_low_clock*
- */sys/bus/platform/devices/INTC1045:00/pch_fivr_switch_frequency/freq_mhz_high_clock*
- */sys/bus/platform/devices/INTC1045:00/pch_fivr_switch_frequency/fivr_switching_freq_mhz*
- */sys/bus/platform/devices/INTC1045:00/pch_fivr_switch_frequency/fivr_switching_fault_status*
- */sys/bus/platform/devices/INTC1045:00/pch_fivr_switch_frequency/ssc_clock_info*

5.2.530 File testing/sysfs-platform-eeepc-laptop

Has the following ABI:

- */sys/devices/platform/eeepc/disp*
- */sys/devices/platform/eeepc/camera*
- */sys/devices/platform/eeepc/cardr*
- */sys/devices/platform/eeepc/cpufv*
- */sys/devices/platform/eeepc/available_cpufv*

5.2.531 File testing/sysfs-platform-hidma

Has the following ABI:

- */sys/devices/platform/hidma-*/chid*

5.2.532 File testing/sysfs-platform-hidma-mgmt

Has the following ABI:

- */sys/devices/platform/hidma-mgmt*/chanops/chan*/priority*
- */sys/devices/platform/hidma-mgmt*/chanops/chan*/weight*
- */sys/devices/platform/hidma-mgmt*/chreset_timeout_cycles*
- */sys/devices/platform/hidma-mgmt*/dma_channels*
- */sys/devices/platform/hidma-mgmt*/hw_version_major*
- */sys/devices/platform/hidma-mgmt*/hw_version_minor*
- */sys/devices/platform/hidma-mgmt*/max_rd_xactions*
- */sys/devices/platform/hidma-mgmt*/max_read_request*
- */sys/devices/platform/hidma-mgmt*/max_wr_xactions*
- */sys/devices/platform/hidma-mgmt*/max_write_request*

5.2.533 File testing/sysfs-platform-i2c-demux-pinctrl

Has the following ABI:

- */sys/devices/platform/<i2c-demux-name>/available_masters*
- */sys/devices/platform/<i2c-demux-name>/current_master*

5.2.534 File testing/sysfs-platform-ideapad-laptop

Has the following ABI:

- */sys/bus/platform/devices/VPC2004:*/camera_power*
- */sys/bus/platform/devices/VPC2004:*/fan_mode*
- */sys/bus/platform/devices/VPC2004:*/touchpad*
- */sys/bus/platform/devices/VPC2004:*/conservation_mode*
- */sys/bus/platform/devices/VPC2004:*/fn_lock*
- */sys/bus/platform/devices/VPC2004:*/usb_charging*

5.2.535 File testing/sysfs-platform-intel-ifs

Device instance to test mapping intel_ifs_0 -> Scan Test intel_ifs_1 -> Array BIST test

Has the following ABI:

- */sys/devices/virtual/misc/intel_ifs_<N>/run_test*
- */sys/devices/virtual/misc/intel_ifs_<N>/status*
- */sys/devices/virtual/misc/intel_ifs_<N>/details*
- */sys/devices/virtual/misc/intel_ifs_<N>/image_version*
- */sys/devices/virtual/misc/intel_ifs_<N>/current_batch*

5.2.536 File testing/sysfs-platform-intel-pmc

Has the following ABI:

- */sys/devices/platform/<platform>/etr3*

5.2.537 File testing/sysfs-platform-intel-wmi-sbl-fw-update

Has the following ABI:

- */sys/bus/wmi/devices/44FADEB1-B204-40F2-8581-394BBDC1B651/firmware_update_request*

5.2.538 File testing/sysfs-platform-intel-wmi-thunderbolt

Has the following ABI:

- */sys/devices/platform/<platform>/force_power*

5.2.539 File testing/sysfs-platform-kim

Has the following ABI:

- */sys/devices/platform/kim/dev_name*
- */sys/devices/platform/kim/baud_rate*
- */sys/devices/platform/kim/flow_cntrl*
- */sys/devices/platform/kim/install*

5.2.540 File testing/sysfs-platform-lg-laptop

Has the following ABI:

- */sys/devices/platform/lg-laptop/reader_mode*
- */sys/devices/platform/lg-laptop/fn_lock*
- */sys/devices/platform/lg-laptop/battery_care_limit*
- */sys/devices/platform/lg-laptop/fan_mode*
- */sys/devices/platform/lg-laptop/usb_charge*

5.2.541 File testing/sysfs-platform-mellanox-bootctl

Has the following ABI:

- */sys/bus/platform/devices/MLNXBF04:00/lifecycle_state*
- */sys/bus/platform/devices/MLNXBF04:00/post_reset_wdog*
- */sys/bus/platform/devices/MLNXBF04:00/reset_action*
- */sys/bus/platform/devices/MLNXBF04:00/second_reset_action*
- */sys/bus/platform/devices/MLNXBF04:00/secure_boot_fuse_state*
- */sys/bus/platform/devices/MLNXBF04:00/bootfifo*
- */sys/bus/platform/devices/MLNXBF04:00/rsh_log*
- */sys/bus/platform/devices/MLNXBF04:00/oob_mac*
- */sys/bus/platform/devices/MLNXBF04:00/opn*
- */sys/bus/platform/devices/MLNXBF04:00/sku*
- */sys/bus/platform/devices/MLNXBF04:00/modl*
- */sys/bus/platform/devices/MLNXBF04:00/sn*
- */sys/bus/platform/devices/MLNXBF04:00/uuid*

- */sys/bus/platform/devices/MLNXBF04:00/rev*
- */sys/bus/platform/devices/MLNXBF04:00/mfg_lock*

5.2.542 File testing/sysfs-platform-msi-laptop

Has the following ABI:

- */sys/devices/platform/msi-laptop-pf/lcd_level*
- */sys/devices/platform/msi-laptop-pf/auto_brightness*
- */sys/devices/platform/msi-laptop-pf/wlan*
- */sys/devices/platform/msi-laptop-pf/bluetooth*
- */sys/devices/platform/msi-laptop-pf/touchpad*
- */sys/devices/platform/msi-laptop-pf/turbo_mode*
- */sys/devices/platform/msi-laptop-pf/eco_mode*
- */sys/devices/platform/msi-laptop-pf/turbo_cooldown*
- */sys/devices/platform/msi-laptop-pf/auto_fan*

5.2.543 File testing/sysfs-platform-phy-rcar-gen3-usb2

Has the following ABI:

- */sys/devices/platform/<phy-name>/role*

5.2.544 File testing/sysfs-platform-power-on-reason

Has the following ABI:

- */sys/devices/platform/.../power_on_reason*

5.2.545 File testing/sysfs-platform-renesas_usb3

Has the following ABI:

- */sys/devices/platform/<renesas_usb3's name>/role*

5.2.546 File testing/sysfs-platform-silicom

Has the following ABI:

- */sys/devices/platform/silicom-platform/uc_version*
- */sys/devices/platform/silicom-platform/power_cycle*
- */sys/devices/platform/silicom-platform/efuse_status*

5.2.547 File testing/sysfs-platform-sst-atom

Has the following ABI:

- */sys/devices/platform/8086<x>:00/firmware_version*

5.2.548 File testing/sysfs-platform-tahvo-usb

Has the following ABI:

- */sys/bus/platform/devices/tahvo-usb/otg_mode*
- */sys/bus/platform/devices/tahvo-usb/vbus*

5.2.549 File testing/sysfs-platform-ts5500

Has the following ABI:

- */sys/devices/platform/ts5500/adc*
- */sys/devices/platform/ts5500/ereset*
- */sys/devices/platform/ts5500/id*
- */sys/devices/platform/ts5500/jumpers*
- */sys/devices/platform/ts5500/name*
- */sys/devices/platform/ts5500/rs485*
- */sys/devices/platform/ts5500/sram*

5.2.550 File testing/sysfs-platform-twl4030-usb

Has the following ABI:

- */sys/bus/platform/devices/*twl4030-usb/vbus*

5.2.551 File testing/sysfs-platform-usbip-vudc

Has the following ABI:

- */sys/devices/platform/usbip-vudc.%d/dev_desc*
- */sys/devices/platform/usbip-vudc.%d/usbip_status*
- */sys/devices/platform/usbip-vudc.%d/usbip_sockfd*

5.2.552 File testing/sysfs-platform-wilco-ec

Has the following ABI:

- */sys/bus/platform/devices/GOOG000C\:00/boot_on_ac*
- */sys/bus/platform/devices/GOOG000C\:00/build_date*
- */sys/bus/platform/devices/GOOG000C\:00/build_revision*
- */sys/bus/platform/devices/GOOG000C\:00/model_number*
- */sys/bus/platform/devices/GOOG000C\:00/usb_charge*
- */sys/bus/platform/devices/GOOG000C\:00/version*

5.2.553 File testing/sysfs-platform_profile

Has the following ABI:

- */sys/firmware/acpi/platform_profile_choices*
- */sys/firmware/acpi/platform_profile*

5.2.554 File testing/sysfs-power

Has the following ABI:

- */sys/power/*
- */sys/power/state*
- */sys/power/mem_sleep*
- */sys/power/disk*
- */sys/power/image_size*
- */sys/power/pm_trace*
- */sys/power/pm_async*
- */sys/power/wakeup_count*
- */sys/power/reserved_size*
- */sys/power/autosleep*
- */sys/power/wake_lock*
- */sys/power/wake_unlock*
- */sys/power/pm_print_times*
- */sys/power/pm_wakeup_irq*
- */sys/power/pm_debug_messages*
- */sys/power/resume_offset*
- */sys/power/suspend_stats*
- */sys/power/suspend_stats/success*

- */sys/power/suspend_stats/fail*
- */sys/power/suspend_stats/failed_freeze*
- */sys/power/suspend_stats/failed_prepare*
- */sys/power/suspend_stats/failed_resume*
- */sys/power/suspend_stats/failed_resume_early*
- */sys/power/suspend_stats/failed_resume_noirq*
- */sys/power/suspend_stats/failed_suspend*
- */sys/power/suspend_stats/failed_suspend_late*
- */sys/power/suspend_stats/failed_suspend_noirq*
- */sys/power/suspend_stats/last_failed_dev*
- */sys/power/suspend_stats/last_failed_errno*
- */sys/power/suspend_stats/last_failed_step*
- */sys/power/suspend_stats/last_hw_sleep*
- */sys/power/suspend_stats/total_hw_sleep*
- */sys/power/suspend_stats/max_hw_sleep*
- */sys/power/sync_on_suspend*

5.2.555 File testing/sysfs-pps

Has the following ABI:

- */sys/class/pps/*
- */sys/class/pps/ppsX/*
- */sys/class/pps/ppsX/assert*
- */sys/class/pps/ppsX/clear*
- */sys/class/pps/ppsX/mode*
- */sys/class/pps/ppsX/echo*
- */sys/class/pps/ppsX/name*
- */sys/class/pps/ppsX/path*

5.2.556 File testing/sysfs-profiling

Has the following ABI:

- */sys/kernel/profiling*

5.2.557 File testing/sysfs-ptp

Has the following ABI:

- */sys/class/ptp/*
- */sys/class/ptp/ptp<N>/*
- */sys/class/ptp/ptp<N>/clock_name*
- */sys/class/ptp/ptp<N>/max_adjustment*
- */sys/class/ptp/ptp<N>/max_vclocks*
- */sys/class/ptp/ptp<N>/n_alarms*
- */sys/class/ptp/ptp<N>/n_external_timestamps*
- */sys/class/ptp/ptp<N>/n_periodic_outputs*
- */sys/class/ptp/ptp<N>/n_pins*
- */sys/class/ptp/ptp<N>/n_vclocks*
- */sys/class/ptp/ptp<N>/pins*
- */sys/class/ptp/ptp<N>/pps_available*
- */sys/class/ptp/ptp<N>/extts_enable*
- */sys/class/ptp/ptp<N>/fifo*
- */sys/class/ptp/ptp<N>/period*
- */sys/class/ptp/ptp<N>/pps_enable*

5.2.558 File testing/sysfs-secvar

Has the following ABI:

- */sys/firmware/secvar*
- */sys/firmware/secvar/vars*
- */sys/firmware/secvar/format*
- */sys/firmware/secvar/vars/<variable name>*
- */sys/firmware/secvar/vars/<variable_name>/size*
- */sys/firmware/secvar/vars/<variable_name>/data*
- */sys/firmware/secvar/vars/<variable_name>/update*
- */sys/firmware/secvar/config*
- */sys/firmware/secvar/config/version*
- */sys/firmware/secvar/config/max_object_size*
- */sys/firmware/secvar/config/total_size*
- */sys/firmware/secvar/config/used_space*
- */sys/firmware/secvar/config/supported_policies*

- */sys/firmware/secvar/config/signed_update_algorithms*

5.2.559 File testing/sysfs-timecard

Has the following ABI:

- */sys/class/timecard/*
- */sys/class/timecard/ocpN/*
- */sys/class/timecard/ocpN/available_clock_sources*
- */sys/class/timecard/ocpN/available_sma_inputs*
- */sys/class/timecard/ocpN/available_sma_outputs*
- */sys/class/timecard/ocpN/clock_source*
- */sys/class/timecard/ocpN/clock_status_drift*
- */sys/class/timecard/ocpN/clock_status_offset*
- */sys/class/timecard/ocpN/freqX*
- */sys/class/timecard/ocpN/freqX/frequency*
- */sys/class/timecard/ocpN/freqX/seconds*
- */sys/class/timecard/ocpN/genX*
- */sys/class/timecard/ocpN/genX/duty*
- */sys/class/timecard/ocpN/genX/period*
- */sys/class/timecard/ocpN/genX/phase*
- */sys/class/timecard/ocpN/genX/polarity*
- */sys/class/timecard/ocpN/genX/running*
- */sys/class/timecard/ocpN/genX/start*
- */sys/class/timecard/ocpN/genX/signal*
- */sys/class/timecard/ocpN/gnss_sync*
- */sys/class/timecard/ocpN/i2c*
- */sys/class/timecard/ocpN/irig_b_mode*
- */sys/class/timecard/ocpN/pps*
- */sys/class/timecard/ocpN/ptp*
- */sys/class/timecard/ocpN/serialnum*
- */sys/class/timecard/ocpN/sma1*
- */sys/class/timecard/ocpN/sma2*
- */sys/class/timecard/ocpN/sma3*
- */sys/class/timecard/ocpN/sma4*
- */sys/class/timecard/ocpN/tod_correction*

- */sys/class/timecard/ocpN/ts_window_adjust*
- */sys/class/timecard/ocpN/ttyGNSS*
- */sys/class/timecard/ocpN/ttyGNSS2*
- */sys/class/timecard/ocpN/ttyMAC*
- */sys/class/timecard/ocpN/ttyNMEA*
- */sys/class/timecard/ocpN/utc_tai_offset*

5.2.560 File testing/sysfs-tty

Has the following ABI:

- */sys/class/tty/console/active*
- */sys/class/tty/tty<x>/active*
- */sys/class/tty/ttyS<x>/uartclk*
- */sys/class/tty/ttyS<x>/type*
- */sys/class/tty/ttyS<x>/line*
- */sys/class/tty/ttyS<x>/port*
- */sys/class/tty/ttyS<x>/irq*
- */sys/class/tty/ttyS<x>/flags*
- */sys/class/tty/ttyS<x>/xmit_fifo_size*
- */sys/class/tty/ttyS<x>/close_delay*
- */sys/class/tty/ttyS<x>/closing_wait*
- */sys/class/tty/ttyS<x>/custom_divisor*
- */sys/class/tty/ttyS<x>/io_type*
- */sys/class/tty/ttyS<x>/iomem_base*
- */sys/class/tty/ttyS<x>/iomem_reg_shift*
- */sys/class/tty/ttyS<x>/rx_trig_bytes*
- */sys/class/tty/ttyS<x>/console*

5.2.561 File testing/sysfs-uevent

Has the following ABI:

- */sys/.../uevent*

5.2.562 File testing/usb-charger-uevent

Has the following ABI:

- *Raise a uevent when a USB charger is inserted or removed*

5.2.563 File testing/usb-uevent

Has the following ABI:

- *Raise a uevent when a USB Host Controller has died*

5.3 ABI obsolete symbols

Documents interfaces that are still remaining in the kernel, but are marked to be removed at some later point in time.

The description of the interface will document the reason why it is obsolete and when it can be expected to be removed.

5.3.1 Symbols under /proc/i8k

/proc/i8k

Defined on file *procfs-i8k*

Legacy interface for getting/setting sensor information like fan speed, temperature, serial number, hotkey status etc on Dell Laptops. Since the driver is now using the standard hwmon sysfs interface, the procfs interface is deprecated.

Users: https://github.com/vitorafsr/i8kutils

5.3.2 Symbols under /sys

/sys/.../iio:deviceX/scan_elements/in_accel_type
/sys/.../iio:deviceX/scan_elements/in_anglvel_type
/sys/.../iio:deviceX/scan_elements/in_magn_type
/sys/.../iio:deviceX/scan_elements/in_incli_type
/sys/.../iio:deviceX/scan_elements/in_voltageY_type
/sys/.../iio:deviceX/scan_elements/in_voltage_type
/sys/.../iio:deviceX/scan_elements/in_voltageY_supply_type
/sys/.../iio:deviceX/scan_elements/in_voltageY_i_type
/sys/.../iio:deviceX/scan_elements/in_voltageY_q_type
/sys/.../iio:deviceX/scan_elements/in_voltage_i_type
/sys/.../iio:deviceX/scan_elements/in_voltage_q_type
/sys/.../iio:deviceX/scan_elements/in_timestamp_type
/sys/.../iio:deviceX/scan_elements/in_pressureY_type
/sys/.../iio:deviceX/scan_elements/in_pressure_type
/sys/.../iio:deviceX/scan_elements/in_rot_quaternion_type
/sys/.../iio:deviceX/scan_elements/in_proximity_type

Defined on file *sysfs-bus-iio*

Description of the scan element data storage within the buffer and hence the form in which it is read from user-space. Form is [be|le]:[s|u]bits/storagebits[>>shift]. be or le specifies big or little endian. s or u specifies if signed (2's complement) or unsigned. bits is the number of bits of data and storagebits is the space (after padding) that it occupies in the buffer. shift if specified, is the shift that needs to be applied prior to masking out unused bits. Some devices put their data in the middle of the transferred elements with additional information on both sides. Note that some devices will have additional information in the unused bits so to get a clean value, the bits value must be used to mask the buffer output value appropriately. The storagebits value also specifies the data alignment. So s48/64>>2 will be a signed 48 bit integer stored in a 64 bit location aligned to a 64 bit boundary. To obtain the clean value, shift right 2 and apply a mask to zero the top 16 bits of the result. For other storage combinations this attribute will be extended appropriately.

Since kernel 5.11 the scan_elements attributes are merged into the bufferY directory, to be configurable per buffer.

/sys/.../iio:deviceX/scan_elements/in_accel_x_en
/sys/.../iio:deviceX/scan_elements/in_accel_y_en
/sys/.../iio:deviceX/scan_elements/in_accel_z_en
/sys/.../iio:deviceX/scan_elements/in_anglvel_x_en
/sys/.../iio:deviceX/scan_elements/in_anglvel_y_en
/sys/.../iio:deviceX/scan_elements/in_anglvel_z_en
/sys/.../iio:deviceX/scan_elements/in_magn_x_en
/sys/.../iio:deviceX/scan_elements/in_magn_y_en
/sys/.../iio:deviceX/scan_elements/in_magn_z_en
/sys/.../iio:deviceX/scan_elements/in_rot_from_north_magnetic_en
/sys/.../iio:deviceX/scan_elements/in_rot_from_north_true_en
/sys/.../iio:deviceX/scan_elements/in_rot_from_north_magnetic_tilt_comp_en
/sys/.../iio:deviceX/scan_elements/in_rot_from_north_true_tilt_comp_en
/sys/.../iio:deviceX/scan_elements/in_timestamp_en
/sys/.../iio:deviceX/scan_elements/in_voltageY_supply_en
/sys/.../iio:deviceX/scan_elements/in_voltageY_en
/sys/.../iio:deviceX/scan_elements/in_voltageY-voltageZ_en
/sys/.../iio:deviceX/scan_elements/in_voltageY_i_en
/sys/.../iio:deviceX/scan_elements/in_voltageY_q_en
/sys/.../iio:deviceX/scan_elements/in_voltage_i_en
/sys/.../iio:deviceX/scan_elements/in_voltage_q_en
/sys/.../iio:deviceX/scan_elements/in_incli_x_en
/sys/.../iio:deviceX/scan_elements/in_incli_y_en
/sys/.../iio:deviceX/scan_elements/in_pressureY_en
/sys/.../iio:deviceX/scan_elements/in_pressure_en
/sys/.../iio:deviceX/scan_elements/in_rot_quaternion_en
/sys/.../iio:deviceX/scan_elements/in_proximity_en

Defined on file *sysfs-bus-iio*

Scan element control for triggered data capture.

Since kernel 5.11 the scan_elements attributes are merged into the bufferY directory, to be configurable per buffer.

/sys/.../iio:deviceX/scan_elements/in_voltageY_index
/sys/.../iio:deviceX/scan_elements/in_voltageY_supply_index
/sys/.../iio:deviceX/scan_elements/in_voltageY_i_index
/sys/.../iio:deviceX/scan_elements/in_voltageY_q_index
/sys/.../iio:deviceX/scan_elements/in_voltage_i_index
/sys/.../iio:deviceX/scan_elements/in_voltage_q_index
/sys/.../iio:deviceX/scan_elements/in_accel_x_index
/sys/.../iio:deviceX/scan_elements/in_accel_y_index
/sys/.../iio:deviceX/scan_elements/in_accel_z_index
/sys/.../iio:deviceX/scan_elements/in_anglvel_x_index
/sys/.../iio:deviceX/scan_elements/in_anglvel_y_index
/sys/.../iio:deviceX/scan_elements/in_anglvel_z_index
/sys/.../iio:deviceX/scan_elements/in_magn_x_index
/sys/.../iio:deviceX/scan_elements/in_magn_y_index
/sys/.../iio:deviceX/scan_elements/in_magn_z_index
/sys/.../iio:deviceX/scan_elements/in_rot_from_north_magnetic_index
/sys/.../iio:deviceX/scan_elements/in_rot_from_north_true_index
/sys/.../iio:deviceX/scan_elements/in_rot_from_north_magnetic_tilt_comp_index
/sys/.../iio:deviceX/scan_elements/in_rot_from_north_true_tilt_comp_index
/sys/.../iio:deviceX/scan_elements/in_incli_x_index
/sys/.../iio:deviceX/scan_elements/in_incli_y_index
/sys/.../iio:deviceX/scan_elements/in_timestamp_index
/sys/.../iio:deviceX/scan_elements/in_pressureY_index
/sys/.../iio:deviceX/scan_elements/in_pressure_index
/sys/.../iio:deviceX/scan_elements/in_rot_quaternion_index
/sys/.../iio:deviceX/scan_elements/in_proximity_index

Defined on file *sysfs-bus-iio*

A single positive integer specifying the position of this scan element in the buffer. Note these are not dependent on what is enabled and may not be contiguous. Thus for user-space to establish the full layout these must be used in conjunction with all _en attributes to establish which channels are present, and the relevant _type attributes to establish the data storage format.

Since kernel 5.11 the scan_elements attributes are merged into the bufferY directory, to be configurable per buffer.

5.3.3 Symbols under /sys/bus

/sys/bus/iio/devices/iio:deviceX/buffer/data_available

Defined on file *sysfs-bus-iio*

A read-only value indicating the bytes of data available in the buffer. In the case of an output buffer, this indicates the amount of empty space available to write data to. In the case of an input buffer, this indicates the amount of data available for reading.

Since Kernel 5.11, multiple buffers are supported. so, it is better to use, instead:

/sys/bus/iio/devices/iio:deviceX/bufferY/data_available

/sys/bus/iio/devices/iio:deviceX/buffer/enable

Defined on file *sysfs-bus-iio*

Actually start the buffer capture up. Will start trigger if first device and appropriate.

Since Kernel 5.11, multiple buffers are supported. so, it is better to use, instead:

/sys/bus/iio/devices/iio:deviceX/bufferY/enable

/sys/bus/iio/devices/iio:deviceX/buffer/length

Defined on file *sysfs-bus-iio*

Number of scans contained by the buffer.

Since Kernel 5.11, multiple buffers are supported. so, it is better to use, instead:

/sys/bus/iio/devices/iio:deviceX/bufferY/length

/sys/bus/iio/devices/iio:deviceX/buffer/watermark

Defined on file *sysfs-bus-iio*

A single positive integer specifying the maximum number of scan elements to wait for.

Poll will block until the watermark is reached.

Blocking read will wait until the minimum between the requested read amount or the low water mark is available.

Non-blocking read will retrieve the available samples from the buffer even if there are less samples then watermark level. This allows the application to block on poll with a timeout and read the available samples after the timeout expires and thus have a maximum delay guarantee.

Since Kernel 5.11, multiple buffers are supported. so, it is better to use, instead:

/sys/bus/iio/devices/iio:deviceX/bufferY/watermark

/sys/bus/iio/devices/iio:deviceX/scan_elements

Defined on file *sysfs-bus-iio*

Directory containing interfaces for elements that will be captured for a single triggered sample set in the buffer.

Since kernel 5.11 the scan_elements attributes are merged into the bufferY directory, to be configurable per buffer.

/sys/bus/platform/devices/INT34D2:00/northpeak

Defined on file *sysfs-driver-intel_pmc_bxt*

This interface allows userspace to enable and disable Northpeak through the PMC/SCU.

Format: %u.

/sys/bus/platform/devices/INT34D2:00/simplecmd

Defined on file *sysfs-driver-intel_pmc_bxt*

This interface allows userspace to send an arbitrary IPC command to the PMC/SCU.

Format: %d %d where first number is command and second number is subcommand.

/sys/bus/usb/devices/.../power/level

Defined on file *sysfs-bus-usb*

Each USB device directory will contain a file named power/level. This file holds a power-level setting for the device, either "on" or "auto".

"on" means that the device is not allowed to autosuspend, although normal suspends for system sleep will still be honored. "auto" means the device will autosuspend and autoresume in the usual manner, according to the capabilities of its driver.

During normal use, devices should be left in the "auto" level. The "on" level is meant for administrative uses. If you want to suspend a device immediately but leave it free to wake up in response to I/O requests, you should write "0" to power/autosuspend.

Device not capable of proper suspend and resume should be left in the "on" level. Although the USB spec requires devices to support suspend/resume, many of them do not. In fact so many don't that by default, the USB core initializes all non-hub devices in the "on" level. Some drivers may change this setting when they are bound.

This file is deprecated and will be removed after 2010. Use the power/control file instead; it does exactly the same thing.

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/arvo/roccatarvo<minor>/actual_profile

Defined on file *sysfs-driver-hid-roccat-arvo*

The integer value of this attribute ranges from 1-5. When read, this attribute returns the number of the actual profile which is also the profile that's active on device startup. When written this attribute activates the selected profile immediately.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/arvo/roccatarvo<minor>/button

Defined on file *sysfs-driver-hid-roccat-arvo*

The keyboard can store short macros with consist of 1 button with several modifier keys internally. When written, this file lets one set the sequence for a specific button for a specific profile. Button and profile numbers are included in written data. The data has to be 24 bytes long. This file is writeonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/arvo/roccatarvo<minor>/info

Defined on file *sysfs-driver-hid-roccat-arvo*

When read, this file returns some info about the device like the installed firmware version. The size of the data is 8 bytes in size. This file is readonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/arvo/roccatarvo<minor>/key_mask

Defined on file *sysfs-driver-hid-roccat-arvo*

The keyboard lets the user deactivate 5 certain keys like the windows and application keys, to protect the user from the outcome of accidentally pressing them. The integer value of this attribute has bits 0-4 set depending on the state of the corresponding key. When read, this file returns the current state of the buttons. When written, the given buttons are activated/deactivated immediately.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/arvo/roccatarvo<minor>/mode_key

Defined on file *sysfs-driver-hid-roccat-arvo*

The keyboard has a condensed layout without num-lock key. Instead it uses a mode-key which activates a gaming mode where the assignment of the number block changes. The integer value of this attribute ranges from 0 (OFF) to 1 (ON). When read, this file returns the actual state of the key. When written, the key is activated/deactivated immediately.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/actual_profile

Defined on file *sysfs-driver-hid-roccat-isku*

The integer value of this attribute ranges from 0-4. When read, this attribute returns the number of the actual profile. This value is persistent, so its equivalent to the profile that's active when the device is powered on next time. When written, this file sets the number of the startup profile and the device activates this profile immediately.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/control

Defined on file *sysfs-driver-hid-roccat-isku*

When written, this file lets one select which data from which profile will be read next. The data has to be 3 bytes long. This file is writeonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/info

Defined on file *sysfs-driver-hid-roccat-isku*

When read, this file returns general data like firmware version. The data is 6 bytes long. This file is readonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/key_mask

Defined on file *sysfs-driver-hid-roccat-isku*

When written, this file lets one deactivate certain keys like windows and application keys, to prevent accidental presses. Profile number for which this settings occur is included in written data. The data has to be 6 bytes long. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/keys_capslock

Defined on file *sysfs-driver-hid-roccat-isku*

When written, this file lets one set the function of the capslock key for a specific profile. Profile number is included in written data. The data has to be 6 bytes long. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/keys_easyzone

Defined on file *sysfs-driver-hid-roccat-isku*

When written, this file lets one set the function of the easyzone keys for a specific profile. Profile number is included in written data. The data has to be 65 bytes long. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/keys_function

Defined on file *sysfs-driver-hid-roccat-isku*

When written, this file lets one set the function of the function keys for a specific profile. Profile number is included in written data. The data has to be 41 bytes long. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/keys_macro

Defined on file *sysfs-driver-hid-roccat-isku*

When written, this file lets one set the function of the macro keys for a specific profile. Profile number is included in written data. The data has to be 35 bytes long. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/keys_media

Defined on file *sysfs-driver-hid-roccat-isku*

When written, this file lets one set the function of the media keys for a specific profile. Profile number is included in written data. The data has to be 29 bytes long. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/keys_thumbster

Defined on file *sysfs-driver-hid-roccat-isku*

When written, this file lets one set the function of the thumbster keys for a specific profile. Profile number is included in written data. The data has to be 23 bytes long. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/last_set

Defined on file *sysfs-driver-hid-roccat-isku*

When written, this file lets one set the time in secs since epoch in which the last configuration took place. The data has to be 20 bytes long.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/light

Defined on file *sysfs-driver-hid-roccat-isku*

When written, this file lets one set the backlight intensity for a specific profile. Profile number is included in written data. The data has to be 10 bytes long for Isku, IskuFX needs 16 bytes of data. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/macro

Defined on file *sysfs-driver-hid-roccat-isku*

When written, this file lets one store macros with max 500 keystrokes for a specific button for a specific profile. Button and profile numbers are included in written data. The data has to be 2083 bytes long. Before reading this file, control has to be written to select which profile and key to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/reset

Defined on file *sysfs-driver-hid-roccat-isku*

When written, this file lets one reset the device. The data has to be 3 bytes long. This file is writeonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/talk

Defined on file *sysfs-driver-hid-roccat-isku*

When written, this file lets one trigger easyshift functionality from the host. The data has to be 16 bytes long. This file is writeonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/talkfx

Defined on file *sysfs-driver-hid-roccat-isku*

When written, this file lets one trigger temporary color schemes from the host. The data has to be 16 bytes long. This file is writeonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/actual_profile

Defined on file *sysfs-driver-hid-roccat-koneplus*

The integer value of this attribute ranges from 0-4. When read, this attribute returns the number of the actual profile. This value is persistent, so its equivalent to the profile that's active when the mouse is powered on next time. When written, this file sets the number of the startup profile and the mouse activates this profile immediately.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/firmware_versi

Defined on file *sysfs-driver-hid-roccat-koneplus*

When read, this file returns the raw integer version number of the firmware reported by the mouse. Using the integer value eases further usage in other programs. To receive the real version number the decimal point has to be shifted 2 positions to the left. E.g. a returned value of 121 means 1.21 This file is readonly. Please read binary attribute info which contains firmware version.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/info

Defined on file *sysfs-driver-hid-roccat-koneplus*

When read, this file returns general data like firmware version. When written, the device can be reset. The data is 8 bytes long.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/macro

Defined on file *sysfs-driver-hid-roccat-koneplus*

The mouse can store a macro with max 500 key/button strokes internally. When written, this file lets one set the sequence for a specific button for a specific profile. Button and profile numbers are included in written data. The data has to be 2082 bytes long. This file is writeonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/profile[1-5]_buttons

Defined on file *sysfs-driver-hid-roccat-koneplus*

The mouse can store 5 profiles which can be switched by the press of a button. A profile is split in settings and buttons. profile_buttons holds information about button layout. When read, these files return the respective profile buttons. The returned data is 77 bytes in size. This file is readonly. Write control to select profile and read profile_buttons instead.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/profile[1-5]_settings

Defined on file *sysfs-driver-hid-roccat-koneplus*

The mouse can store 5 profiles which can be switched by the press of a button. A profile is split in settings and buttons. profile_settings holds information like resolution, sensitivity and light effects. When read, these files return the respective profile settings. The returned data is 43 bytes in size. This file is readonly. Write control to select profile and read profile_settings instead.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/profile_buttons

Defined on file *sysfs-driver-hid-roccat-koneplus*

The mouse can store 5 profiles which can be switched by the press of a button. A profile is split in settings and buttons. profile_buttons holds information about button layout. When written, this file lets one write the respective profile buttons back to the mouse. The data has to be 77 bytes long. The mouse will reject invalid data. Which profile to write is determined by the profile number contained in the data. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/profile_settings

Defined on file *sysfs-driver-hid-roccat-koneplus*

The mouse can store 5 profiles which can be switched by the press of a button. A profile is split in settings and buttons. profile_settings holds information like resolution, sensitivity and light effects. When written, this file lets one write the respective profile settings back to the mouse. The data has to be 43 bytes long. The mouse will reject invalid data. Which profile to write is determined by the profile number contained in the data. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/sensor

Defined on file *sysfs-driver-hid-roccat-koneplus*

The mouse has a tracking- and a distance-control-unit. These can be activated/deactivated and the lift-off distance can be set. The data has to be 6 bytes long. This file is writeonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/startup_profile

Defined on file *sysfs-driver-hid-roccat-koneplus*

The integer value of this attribute ranges from 0-4. When read, this attribute returns the number of the actual profile. This value is persistent, so its equivalent to the profile that's active when the mouse is powered on next time. When written, this file sets the number of the startup profile and the mouse activates this profile immediately. Please use actual_profile, it does the same thing.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/talk

Defined on file *sysfs-driver-hid-roccat-koneplus*

Used to active some easy* functions of the mouse from outside. The data has to be 16 bytes long. This file is writeonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/tcu

Defined on file *sysfs-driver-hid-roccat-koneplus*

When written a calibration process for the tracking control unit can be initiated/cancelled. Also lets one read/write sensor registers. The data has to be 4 bytes long.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/tcu_image

Defined on file *sysfs-driver-hid-roccat-koneplus*

When read the mouse returns a 30x30 pixel image of the sampled underground. This works only in the course of a calibration process initiated with tcu. The returned data is 1028 bytes in size. This file is readonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/konepure/roccatkonepure<minor>/actual_profile

Defined on file *sysfs-driver-hid-roccat-konepure*

The mouse can store 5 profiles which can be switched by the press of a button. actual_profile holds number of actual profile. This value is persistent, so its value determines the profile that's active when the mouse is powered on next time. When written, the mouse activates the set profile immediately. The data has to be 3 bytes long. The mouse will reject invalid data.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/konepure/roccatkonepure<minor>/control

Defined on file *sysfs-driver-hid-roccat-konepure*

When written, this file lets one select which data from which profile will be read next. The data has to be 3 bytes long. This file is writeonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/konepure/roccatkonepure<minor>/info

Defined on file *sysfs-driver-hid-roccat-konepure*

When read, this file returns general data like firmware version. When written, the device can be reset. The data is 6 bytes long.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/konepure/roccatkonepure<minor>/macro

Defined on file *sysfs-driver-hid-roccat-konepure*

The mouse can store a macro with max 500 key/button strokes internally. When written, this file lets one set the sequence for a specific button for a specific profile. Button and profile numbers are included in written data. The data has to be 2082 bytes long. This file is writeonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/konepure/roccatkonepure<minor>/profile_button

Defined on file *sysfs-driver-hid-roccat-konepure*

The mouse can store 5 profiles which can be switched by the press of a button. A profile is split in settings and buttons. profile_buttons holds information about button layout. When written, this file lets one write the respective profile buttons back to the mouse. The data has to be 59 bytes long. The mouse will reject invalid data. Which profile to write is determined by the profile number contained in the data. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/konepure/roccatkonepure<minor>/profile_setting

Defined on file *sysfs-driver-hid-roccat-konepure*

The mouse can store 5 profiles which can be switched by the press of a button. A profile is split in settings and buttons. profile_settings holds information like resolution, sensitivity and light

effects. When written, this file lets one write the respective profile settings back to the mouse. The data has to be 31 bytes long. The mouse will reject invalid data. Which profile to write is determined by the profile number contained in the data. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/konepure/roccatkonepure<minor>/sensor

Defined on file *sysfs-driver-hid-roccat-konepure*

The mouse has a tracking- and a distance-control-unit. These can be activated/deactivated and the lift-off distance can be set. The data has to be 6 bytes long. This file is writeonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/konepure/roccatkonepure<minor>/talk

Defined on file *sysfs-driver-hid-roccat-konepure*

Used to active some easy* functions of the mouse from outside. The data has to be 16 bytes long. This file is writeonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/konepure/roccatkonepure<minor>/tcu

Defined on file *sysfs-driver-hid-roccat-konepure*

When written a calibration process for the tracking control unit can be initiated/cancelled. Also lets one read/write sensor registers. The data has to be 4 bytes long.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/konepure/roccatkonepure<minor>/tcu_image

Defined on file *sysfs-driver-hid-roccat-konepure*

When read the mouse returns a 30x30 pixel image of the sampled underground. This works only in the course of a calibration process initiated with tcu. The returned data is 1028 bytes in size. This file is readonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kovaplus/roccatkovaplus<minor>/actual_cpi

Defined on file *sysfs-driver-hid-roccat-kovaplus*

The integer value of this attribute ranges from 1-4. When read, this attribute returns the number of the active cpi level. This file is readonly. Has never been used. If bookkeeping is done, it's done in userland tools.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kovaplus/roccatkovaplus<minor>/actual_profile

Defined on file *sysfs-driver-hid-roccat-kovaplus*

The integer value of this attribute ranges from 0-4. When read, this attribute returns the number of the active profile. When written, the mouse activates this profile immediately. The profile that's active when powered down is the same that's active when the mouse is powered on.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kovaplus/roccatkovaplus<minor>/actual_sensitivi

Defined on file *sysfs-driver-hid-roccat-kovaplus*

The integer value of this attribute ranges from 1-10. When read, this attribute returns the number of the actual sensitivity in x direction. This file is readonly. Has never been used. If bookkeeping is done, it's done in userland tools.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kovaplus/roccatkovaplus<minor>/actual_sensitivi

Defined on file *sysfs-driver-hid-roccat-kovaplus*

The integer value of this attribute ranges from 1-10. When read, this attribute returns the number of the actual sensitivity in y direction. This file is readonly. Has never been used. If bookkeeping is done, it's done in userland tools.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kovaplus/roccatkovaplus<minor>/firmware_versio

Defined on file *sysfs-driver-hid-roccat-kovaplus*

When read, this file returns the raw integer version number of the firmware reported by the mouse. Using the integer value eases further usage in other programs. To receive the real version number the decimal point has to be shifted 2 positions to the left. E.g. a returned value of 121 means 1.21 This file is readonly. Obsoleted by binary sysfs attribute ”info”.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kovaplus/roccatkovaplus<minor>/info

Defined on file *sysfs-driver-hid-roccat-kovaplus*

When read, this file returns general data like firmware version. When written, the device can be reset. The data is 6 bytes long.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kovaplus/roccatkovaplus<minor>/profile[1-5]_buttons

Defined on file *sysfs-driver-hid-roccat-kovaplus*

The mouse can store 5 profiles which can be switched by the press of a button. A profile is split in settings and buttons. profile_buttons holds information about button layout. When read, these files return the respective profile buttons. The returned data is 23 bytes in size. This file is readonly. Write control to select profile and read profile_buttons instead.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kovaplus/roccatkovaplus<minor>/profile[1-5]_settings

Defined on file *sysfs-driver-hid-roccat-kovaplus*

The mouse can store 5 profiles which can be switched by the press of a button. A profile is split in settings and buttons. profile_settings holds information like resolution, sensitivity and light effects. When read, these files return the respective profile settings. The returned data is 16 bytes in size. This file is readonly. Write control to select profile and read profile_settings instead.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kovaplus/roccatkovaplus<minor>/profile_buttons

Defined on file *sysfs-driver-hid-roccat-kovaplus*

The mouse can store 5 profiles which can be switched by the press of a button. A profile is split in settings and buttons. profile_buttons holds information about button layout. When written, this file lets one write the respective profile buttons back to the mouse. The data has to be 23 bytes long. The mouse will reject invalid data. Which profile to write is determined by the profile number contained in the data. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kovaplus/roccatkovaplus<minor>/profile_settings

Defined on file *sysfs-driver-hid-roccat-kovaplus*

The mouse can store 5 profiles which can be switched by the press of a button. A profile is split in settings and buttons. profile_settings holds information like resolution, sensitivity and light effects. When written, this file lets one write the respective profile settings back to the mouse. The data has to be 16 bytes long. The mouse will reject invalid data. Which profile to write is determined by the profile number contained in the data. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/pyra/roccatpyra<minor>/actual_cpi

Defined on file *sysfs-driver-hid-roccat-pyra*

It is possible to switch the cpi setting of the mouse with the press of a button. When read, this file returns the raw number of the actual cpi setting reported by the mouse. This number has to be further processed to receive the real dpi value:

VALUE	DPI
1	400
2	800
4	1600

This file is readonly. Has never been used. If bookkeeping is done, it's done in userland tools.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/pyra/roccatpyra<minor>/actual_profile

Defined on file *sysfs-driver-hid-roccat-pyra*

When read, this file returns the number of the actual profile in range 0-4. This file is readonly. Please use binary attribute ”settings” which provides this information.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/pyra/roccatpyra<minor>/firmware_version

Defined on file *sysfs-driver-hid-roccat-pyra*

When read, this file returns the raw integer version number of the firmware reported by the mouse. Using the integer value eases further usage in other programs. To receive the real version number the decimal point has to be shifted 2 positions to the left. E.g. a returned value of 138 means 1.38 This file is readonly. Please use binary attribute ”info” which provides this information.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/pyra/roccatpyra<minor>/info

Defined on file *sysfs-driver-hid-roccat-pyra*

When read, this file returns general data like firmware version. When written, the device can be reset. The data is 6 bytes long.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/pyra/roccatpyra<minor>/profile[1-5]_buttons

Defined on file *sysfs-driver-hid-roccat-pyra*

The mouse can store 5 profiles which can be switched by the press of a button. A profile is split in settings and buttons. profile_buttons holds information about button layout. When read, these files return the respective profile buttons. The returned data is 19 bytes in size. This file is readonly. Write control to select profile and read profile_buttons instead.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/pyra/roccatpyra<minor>/profile[1-5]_settings

Defined on file *sysfs-driver-hid-roccat-pyra*

The mouse can store 5 profiles which can be switched by the press of a button. A profile is split in settings and buttons. profile_settings holds information like resolution, sensitivity and light effects. When read, these files return the respective profile settings. The returned data is 13 bytes in size. This file is readonly. Write control to select profile and read profile_settings instead.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/pyra/roccatpyra<minor>/profile_buttons

Defined on file *sysfs-driver-hid-roccat-pyra*

The mouse can store 5 profiles which can be switched by the press of a button. A profile is split in settings and buttons. profile_buttons holds information about button layout. When written, this file lets one write the respective profile buttons back to the mouse. The data has to be 19 bytes long. The mouse will reject invalid data. Which profile to write is determined by the profile number contained in the data. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/pyra/roccatpyra<minor>/profile_settings

Defined on file *sysfs-driver-hid-roccat-pyra*

The mouse can store 5 profiles which can be switched by the press of a button. A profile is split in settings and buttons. profile_settings holds information like resolution, sensitivity and light effects. When written, this file lets one write the respective profile settings back to the mouse. The data has to be 13 bytes long. The mouse will reject invalid data. Which profile to write is determined by the profile number contained in the data. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/pyra/roccatpyra<minor>/settings

Defined on file *sysfs-driver-hid-roccat-pyra*

When read, this file returns the settings stored in the mouse. The size of the data is 3 bytes and holds information on the startup_profile. When written, this file lets write settings back to the mouse. The data has to be 3 bytes long. The mouse will reject invalid data.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/pyra/roccatpyra<minor>/startup_profile

Defined on file *sysfs-driver-hid-roccat-pyra*

The integer value of this attribute ranges from 0-4. When read, this attribute returns the number of the profile that's active when the mouse is powered on. This file is readonly. Please use binary attribute ”settings” which provides this information.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/control

Defined on file *sysfs-driver-hid-roccat-ryos*

When written, this file lets one select which data from which profile will be read next. The data has to be 3 bytes long. This file is writeonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/custom_lights

Defined on file *sysfs-driver-hid-roccat-ryos*

When written, this file lets one set the actual per-key lighting. This attribute is only valid for the pro variant. The data has to be 20 bytes long. This file is writeonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/info

Defined on file *sysfs-driver-hid-roccat-ryos*

When read, this file returns general data like firmware version. The data is 8 bytes long. This file is readonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/key_mask

Defined on file *sysfs-driver-hid-roccat-ryos*

When written, this file lets one deactivate certain keys like windows and application keys, to prevent accidental presses. Profile index for which this settings occur is included in written data. The data has to be 6 bytes long. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/keys_easyzone

Defined on file *sysfs-driver-hid-roccat-ryos*

When written, this file lets one set the function of the easyzone keys for a specific profile. Profile index is included in written data. The data has to be 294 bytes long. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/keys_extra

Defined on file *sysfs-driver-hid-roccat-ryos*

When written, this file lets one set the function of the capslock and function keys for a specific profile. Profile index is included in written data. The data has to be 8 bytes long. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/keys_function

Defined on file *sysfs-driver-hid-roccat-ryos*

When written, this file lets one set the function of the function keys for a specific profile. Profile index is included in written data. The data has to be 95 bytes long. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/keys_macro

Defined on file *sysfs-driver-hid-roccat-ryos*

When written, this file lets one set the function of the macro keys for a specific profile. Profile index is included in written data. The data has to be 35 bytes long. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/keys_primary

Defined on file *sysfs-driver-hid-roccat-ryos*

When written, this file lets one set the default of all keys for a specific profile. Profile index is included in written data. The data has to be 125 bytes long. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/keys_thumbster

Defined on file *sysfs-driver-hid-roccat-ryos*

When written, this file lets one set the function of the thumbster keys for a specific profile. Profile index is included in written data. The data has to be 23 bytes long. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/light

Defined on file *sysfs-driver-hid-roccat-ryos*

When written, this file lets one set the backlight intensity for a specific profile. Profile index is included in written data. This attribute is only valid for the glow and pro variant. The data has to be 16 bytes long. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/light_control

Defined on file *sysfs-driver-hid-roccat-ryos*

When written, this file lets one switch between stored and custom light settings. This attribute is only valid for the pro variant. The data has to be 8 bytes long. This file is writeonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/light_macro

Defined on file *sysfs-driver-hid-roccat-ryos*

When written, this file lets one set a light macro that is looped whenever the device gets in dimness mode. This attribute is only valid for the pro variant. The data has to be 2002 bytes long. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/macro

Defined on file *sysfs-driver-hid-roccat-ryos*

When written, this file lets one store macros with max 480 keystrokes for a specific button for a specific profile. Button and profile indexes are included in written data. The data has to be 2002 bytes long. Before reading this file, control has to be written to select which profile and key to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/profile

Defined on file *sysfs-driver-hid-roccat-ryos*

The mouse can store 5 profiles which can be switched by the press of a button. profile holds index of actual profile. This value is persistent, so its value determines the profile that's active when the device is powered on next time. When written, the device activates the set profile immediately. The data has to be 3 bytes long. The device will reject invalid data.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/reset

Defined on file *sysfs-driver-hid-roccat-ryos*

When written, this file lets one reset the device. The data has to be 3 bytes long. This file is writeonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/stored_lights

Defined on file *sysfs-driver-hid-roccat-ryos*

When written, this file lets one set per-key lighting for different layers. This attribute is only valid for the pro variant. The data has to be 1382 bytes long. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/talk

Defined on file *sysfs-driver-hid-roccat-ryos*

When written, this file lets one trigger easyshift functionality from the host. The data has to be 16 bytes long. This file is writeonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/savu/roccatsavu<minor>/buttons

Defined on file *sysfs-driver-hid-roccat-savu*

The mouse can store 5 profiles which can be switched by the press of a button. A profile is split into general settings and button settings. The buttons variable holds information about button layout. When written, this file lets one write the respective profile buttons to the mouse. The data has to be 47 bytes long. The mouse will reject invalid data. Which profile to write is determined by the profile number contained in the data. Before reading this file, control has to be written to select which profile to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/savu/roccatsavu<minor>/control

Defined on file *sysfs-driver-hid-roccat-savu*

When written, this file lets one select which data from which profile will be read next. The data has to be 3 bytes long. This file is writeonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/savu/roccatsavu<minor>/general

Defined on file *sysfs-driver-hid-roccat-savu*

The mouse can store 5 profiles which can be switched by the press of a button. A profile is split into general settings and button settings. A profile holds information like resolution, sensitivity and light effects. When written, this file lets one write the respective profile settings back to the mouse. The data has to be 43 bytes long. The mouse will reject invalid data. Which profile to write is determined by the profile number contained in the data. This file is writeonly.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/savu/roccatsavu<minor>/info

Defined on file *sysfs-driver-hid-roccat-savu*

When read, this file returns general data like firmware version. When written, the device can be reset. The data is 8 bytes long.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/savu/roccatsavu<minor>/macro

Defined on file *sysfs-driver-hid-roccat-savu*

When written, this file lets one store macros with max 500 keystrokes for a specific button for a specific profile. Button and profile numbers are included in written data. The data has to be 2083 bytes long. Before reading this file, control has to be written to select which profile and key to read.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/savu/roccatsavu<minor>/profile

Defined on file *sysfs-driver-hid-roccat-savu*

The mouse can store 5 profiles which can be switched by the press of a button. profile holds number of actual profile. This value is persistent, so its value determines the profile that's active when the mouse is powered on next time. When written, the mouse activates the set profile immediately. The data has to be 3 bytes long. The mouse will reject invalid data.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/savu/roccatsavu<minor>/sensor

Defined on file *sysfs-driver-hid-roccat-savu*

The mouse has a Avago ADNS-3090 sensor. This file allows reading and writing of the mouse sensors registers. The data has to be 4 bytes long.

Users: http://roccat.sourceforge.net

/sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/control

Defined on file *sysfs-driver-hid-roccat-lua*

When written, cpi, button and light settings can be configured. When read, actual cpi setting and sensor data are returned. The data has to be 8 bytes long.

Users: http://roccat.sourceforge.net

5.3.4 Symbols under /sys/class

/sys/class/gpio/

Defined on file *sysfs-gpio*

As a Kconfig option, individual GPIO signals may be accessed from userspace. GPIOs are only made available to userspace by an explicit ”export” operation. If a given GPIO is not claimed

for use by kernel code, it may be exported by userspace (and unexported later). Kernel code may export it for complete or partial access.

GPIOs are identified as they are inside the kernel, using integers in the range 0..INT_MAX. See Documentation/admin-guide/gpio for more information.

```
/sys/class/gpio
    /export ... asks the kernel to export a GPIO to userspace
    /unexport ... to return a GPIO to the kernel
    /gpioN ... for each exported GPIO #N OR
    /<LINE-NAME> ... for a properly named GPIO line
        /value ... always readable, writes fail for input GPIOs
        /direction ... r/w as: in, out (default low); write: high, low
        /edge ... r/w as: none, falling, rising, both
    /gpiochipN ... for each gpiochip; #N is its first GPIO
        /base ... (r/o) same as N
        /label ... (r/o) descriptive, not necessarily unique
        /ngpio ... (r/o) number of GPIOs; numbered N to N + (ngpio - 1)
```

This ABI is deprecated and will be removed after 2020. It is replaced with the GPIO character device.

/sys/class/typec/<port|partner|cable>/<dev>/mode<index>/

Defined on file *sysfs-class-typec*

Every supported mode will have its own directory. The name of a mode will be "mode<index>" (for example mode1), where <index> is the actual index to the mode VDO returned by Discover Modes USB power delivery command.

/sys/class/typec/<port|partner|cable>/<dev>/mode<index>/active

Defined on file *sysfs-class-typec*

Shows if the mode is active or not. The attribute can be used for entering/exiting the mode with partners and cable plugs, and with the port alternate modes it can be used for disabling support for specific alternate modes. Entering/exiting modes is supported as synchronous operation so write(2) to the attribute does not return until the enter/exit mode operation has finished. The attribute is notified when the mode is entered/exited so poll(2) on the attribute wakes up. Entering/exiting a mode will also generate uevent KOBJ_CHANGE.

Valid values: yes, no

/sys/class/typec/<port|partner|cable>/<dev>/mode<index>/description

Defined on file *sysfs-class-typec*

Shows description of the mode. The description is optional for the drivers, just like with the Billboard Devices.

/sys/class/typec/<port|partner|cable>/<dev>/mode<index>/vdo

Defined on file *sysfs-class-typec*

Shows the VDO in hexadecimal returned by Discover Modes command for this mode.

/sys/class/typec/<port|partner|cable>/<dev>/svid

Defined on file *sysfs-class-typec*

The SVID (Standard or Vendor ID) assigned by USB-IF for this alternate mode.

5.3.5 Symbols under /sys/devices

/sys/devices/system/cpu/cpuidle/current_governor_ro

Defined on file *sysfs-cpuidle*

current_governor_ro shows current using cpuidle governor, but read only. with the update that cpuidle governor can be changed at runtime in default, both current_governor and current_governor_ro co-exist under /sys/devices/system/cpu/cpuidle/ file, it's duplicate so make current_governor_ro obsolete.

5.3.6 Symbols under /sys/firmware

/sys/firmware/acpi/hotplug/force_remove

Defined on file *sysfs-firmware-acpi*

Since the force_remove is inherently broken and dangerous to use for some hotplugable resources like memory (because ignoring the offline failure might lead to memory corruption and crashes) enabling this knob is not safe and thus unsupported.

5.3.7 Symbols under /sys/kernel

/sys/kernel/fadump_enabled

Defined on file *sysfs-kernel-fadump_enabled*

read only Primarily used to identify whether the FADump is enabled in the kernel or not. User: Kdump service

/sys/kernel/fadump_registered

Defined on file *sysfs-kernel-fadump_registered*

read/write Helps to control the dump collect feature from userspace. Setting 1 to this file enables the system to collect the dump and 0 to disable it. User: Kdump service

/sys/kernel/fadump_release_mem

Defined on file *sysfs-kernel-fadump_release_mem*

write only This is a special sysfs file and only available when the system is booted to capture the vmcore using FADump. It is used to release the memory reserved by FADump to save the crash dump.

5.3.8 Symbols under /sys/o2cb

/sys/o2cb

Defined on file *o2cb*

Ocfs2-tools looks at 'interface-revision' for versioning information. Each logmask/ file controls a set of debug prints and can be written into with the strings ”allow”, ”deny”, or ”off”. Reading the file returns the current state. Was renamed to /sys/fs/u2cb/

Users: ocfs2-tools. It's sufficient to mail proposed changes to ocfs2-devel@lists.linux.dev.

5.3.9 File obsolete/o2cb

Has the following ABI:

- */sys/o2cb*

5.3.10 File obsolete/procfs-i8k

Has the following ABI:

- */proc/i8k*

5.3.11 File obsolete/sysfs-bus-iio

Has the following ABI:

- */sys/bus/iio/devices/iio:deviceX/buffer/length*
- */sys/bus/iio/devices/iio:deviceX/buffer/enable*
- */sys/bus/iio/devices/iio:deviceX/scan_elements*
- */sys/.../iio:deviceX/scan_elements/in_accel_x_en*
- */sys/.../iio:deviceX/scan_elements/in_accel_y_en*
- */sys/.../iio:deviceX/scan_elements/in_accel_z_en*
- */sys/.../iio:deviceX/scan_elements/in_anglvel_x_en*
- */sys/.../iio:deviceX/scan_elements/in_anglvel_y_en*
- */sys/.../iio:deviceX/scan_elements/in_anglvel_z_en*
- */sys/.../iio:deviceX/scan_elements/in_magn_x_en*
- */sys/.../iio:deviceX/scan_elements/in_magn_y_en*
- */sys/.../iio:deviceX/scan_elements/in_magn_z_en*

- */sys/.../iio:deviceX/scan_elements/in_rot_from_north_magnetic_en*
- */sys/.../iio:deviceX/scan_elements/in_rot_from_north_true_en*
- */sys/.../iio:deviceX/scan_elements/in_rot_from_north_magnetic_tilt_comp_en*
- */sys/.../iio:deviceX/scan_elements/in_rot_from_north_true_tilt_comp_en*
- */sys/.../iio:deviceX/scan_elements/in_timestamp_en*
- */sys/.../iio:deviceX/scan_elements/in_voltageY_supply_en*
- */sys/.../iio:deviceX/scan_elements/in_voltageY_en*
- */sys/.../iio:deviceX/scan_elements/in_voltageY-voltageZ_en*
- */sys/.../iio:deviceX/scan_elements/in_voltageY_i_en*
- */sys/.../iio:deviceX/scan_elements/in_voltageY_q_en*
- */sys/.../iio:deviceX/scan_elements/in_voltage_i_en*
- */sys/.../iio:deviceX/scan_elements/in_voltage_q_en*
- */sys/.../iio:deviceX/scan_elements/in_incli_x_en*
- */sys/.../iio:deviceX/scan_elements/in_incli_y_en*
- */sys/.../iio:deviceX/scan_elements/in_pressureY_en*
- */sys/.../iio:deviceX/scan_elements/in_pressure_en*
- */sys/.../iio:deviceX/scan_elements/in_rot_quaternion_en*
- */sys/.../iio:deviceX/scan_elements/in_proximity_en*
- */sys/.../iio:deviceX/scan_elements/in_accel_type*
- */sys/.../iio:deviceX/scan_elements/in_anglvel_type*
- */sys/.../iio:deviceX/scan_elements/in_magn_type*
- */sys/.../iio:deviceX/scan_elements/in_incli_type*
- */sys/.../iio:deviceX/scan_elements/in_voltageY_type*
- */sys/.../iio:deviceX/scan_elements/in_voltage_type*
- */sys/.../iio:deviceX/scan_elements/in_voltageY_supply_type*
- */sys/.../iio:deviceX/scan_elements/in_voltageY_i_type*
- */sys/.../iio:deviceX/scan_elements/in_voltageY_q_type*
- */sys/.../iio:deviceX/scan_elements/in_voltage_i_type*
- */sys/.../iio:deviceX/scan_elements/in_voltage_q_type*
- */sys/.../iio:deviceX/scan_elements/in_timestamp_type*
- */sys/.../iio:deviceX/scan_elements/in_pressureY_type*
- */sys/.../iio:deviceX/scan_elements/in_pressure_type*
- */sys/.../iio:deviceX/scan_elements/in_rot_quaternion_type*
- */sys/.../iio:deviceX/scan_elements/in_proximity_type*

- */sys/.../iio:deviceX/scan_elements/in_voltageY_index*
- */sys/.../iio:deviceX/scan_elements/in_voltageY_supply_index*
- */sys/.../iio:deviceX/scan_elements/in_voltageY_i_index*
- */sys/.../iio:deviceX/scan_elements/in_voltageY_q_index*
- */sys/.../iio:deviceX/scan_elements/in_voltage_i_index*
- */sys/.../iio:deviceX/scan_elements/in_voltage_q_index*
- */sys/.../iio:deviceX/scan_elements/in_accel_x_index*
- */sys/.../iio:deviceX/scan_elements/in_accel_y_index*
- */sys/.../iio:deviceX/scan_elements/in_accel_z_index*
- */sys/.../iio:deviceX/scan_elements/in_anglvel_x_index*
- */sys/.../iio:deviceX/scan_elements/in_anglvel_y_index*
- */sys/.../iio:deviceX/scan_elements/in_anglvel_z_index*
- */sys/.../iio:deviceX/scan_elements/in_magn_x_index*
- */sys/.../iio:deviceX/scan_elements/in_magn_y_index*
- */sys/.../iio:deviceX/scan_elements/in_magn_z_index*
- */sys/.../iio:deviceX/scan_elements/in_rot_from_north_magnetic_index*
- */sys/.../iio:deviceX/scan_elements/in_rot_from_north_true_index*
- */sys/.../iio:deviceX/scan_elements/in_rot_from_north_magnetic_tilt_comp_index*
- */sys/.../iio:deviceX/scan_elements/in_rot_from_north_true_tilt_comp_index*
- */sys/.../iio:deviceX/scan_elements/in_incli_x_index*
- */sys/.../iio:deviceX/scan_elements/in_incli_y_index*
- */sys/.../iio:deviceX/scan_elements/in_timestamp_index*
- */sys/.../iio:deviceX/scan_elements/in_pressureY_index*
- */sys/.../iio:deviceX/scan_elements/in_pressure_index*
- */sys/.../iio:deviceX/scan_elements/in_rot_quaternion_index*
- */sys/.../iio:deviceX/scan_elements/in_proximity_index*
- */sys/bus/iio/devices/iio:deviceX/buffer/watermark*
- */sys/bus/iio/devices/iio:deviceX/buffer/data_available*

5.3.12 File obsolete/sysfs-bus-usb

Has the following ABI:

- */sys/bus/usb/devices/.../power/level*

5.3.13 File obsolete/sysfs-class-typec

These files are deprecated and will be removed. The same files are available under /sys/bus/typec (see *testing/sysfs-bus-typec*).

Has the following ABI:

- */sys/class/typec/<port|partner|cable>/<dev>/svid*
- */sys/class/typec/<port|partner|cable>/<dev>/mode<index>/*
- */sys/class/typec/<port|partner|cable>/<dev>/mode<index>/description*
- */sys/class/typec/<port|partner|cable>/<dev>/mode<index>/vdo*
- */sys/class/typec/<port|partner|cable>/<dev>/mode<index>/active*

5.3.14 File obsolete/sysfs-cpuidle

Has the following ABI:

- */sys/devices/system/cpu/cpuidle/current_governor_ro*

5.3.15 File obsolete/sysfs-driver-hid-roccat-arvo

Has the following ABI:

- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/arvo/roccatarvo<minor>/actual_profile*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/arvo/roccatarvo<minor>/button*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/arvo/roccatarvo<minor>/info*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/arvo/roccatarvo<minor>/key_mask*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/arvo/roccatarvo<minor>/mode_key*

5.3.16 File obsolete/sysfs-driver-hid-roccat-isku

Has the following ABI:

- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/actual_profile*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/info*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/key_mask*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/keys_capslock*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/keys_easyzone*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/keys_function*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/keys_macro*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/keys_media*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/keys_thumbster*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/last_set*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/light*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/macro*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/reset*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/control*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/talk*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/isku/roccatisku<minor>/talkfx*

5.3.17 File obsolete/sysfs-driver-hid-roccat-koneplus

Has the following ABI:

- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/actual_profile*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/startup_profile*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/firmware_version*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/info*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/macro*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/profile_buttons*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/profile[1-5]_buttons*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/profile_settings*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/profile[1-5]_settings*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/sensor*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/talk*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/tcu*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/koneplus/roccatkoneplus<minor>/tcu_image*

5.3.18 File obsolete/sysfs-driver-hid-roccat-konepure

Has the following ABI:

- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/konepure/roccatkonepure<minor>/actual_profile*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/konepure/roccatkonepure<minor>/control*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/konepure/roccatkonepure<minor>/info*

- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/konepure/roccatkonepure<minor>/macro*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/konepure/roccatkonepure<minor>/profile_buttons*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/konepure/roccatkonepure<minor>/profile_settings*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/konepure/roccatkonepure<minor>/sensor*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/konepure/roccatkonepure<minor>/talk*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/konepure/roccatkonepure<minor>/tcu*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/konepure/roccatkonepure<minor>/tcu_image*

5.3.19 File obsolete/sysfs-driver-hid-roccat-kovaplus

Has the following ABI:

- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kovaplus/roccatkovaplus<minor>/actual_cpi*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kovaplus/roccatkovaplus<minor>/actual_profile*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kovaplus/roccatkovaplus<minor>/actual_sensitivity_x*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kovaplus/roccatkovaplus<minor>/actual_sensitivity_y*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kovaplus/roccatkovaplus<minor>/firmware_version*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kovaplus/roccatkovaplus<minor>/info*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kovaplus/roccatkovaplus<minor>/profile_buttons*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kovaplus/roccatkovaplus<minor>/profile[1-5]_buttons*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kovaplus/roccatkovaplus<minor>/profile_settings*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/kovaplus/roccatkovaplus<minor>/profile[1-5]_settings*

5.3.20 File obsolete/sysfs-driver-hid-roccat-lua

Has the following ABI:

- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/control*

5.3.21 File obsolete/sysfs-driver-hid-roccat-pyra

Has the following ABI:

- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/pyra/roccatpyra<minor>/actual_cpi*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/pyra/roccatpyra<minor>/actual_profile*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/pyra/roccatpyra<minor>/firmware_version*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/pyra/roccatpyra<minor>/info*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/pyra/roccatpyra<minor>/profile_buttons*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/pyra/roccatpyra<minor>/profile[1-5]_buttons*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/pyra/roccatpyra<minor>/profile_settings*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/pyra/roccatpyra<minor>/profile[1-5]_settings*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/pyra/roccatpyra<minor>/settings*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/pyra/roccatpyra<minor>/startup_profile*

5.3.22 File obsolete/sysfs-driver-hid-roccat-ryos

Has the following ABI:

- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/control*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/profile*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/keys_primary*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/keys_function*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/keys_macro*

- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/keys_thumbster*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/keys_extra*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/keys_easyzone*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/key_mask*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/light*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/macro*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/info*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/reset*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/talk*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/light_control*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/stored_lights*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/custom_lights*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/ryos/roccatryos<minor>/light_macro*

5.3.23 File obsolete/sysfs-driver-hid-roccat-savu

Has the following ABI:

- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/savu/roccatsavu<minor>/buttons*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/savu/roccatsavu<minor>/control*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/savu/roccatsavu<minor>/general*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/savu/roccatsavu<minor>/info*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/savu/roccatsavu<minor>/macro*
- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/savu/roccatsavu<minor>/profile*

- */sys/bus/usb/devices/<busnum>-<devnum>:<config num>.<interface num>/<hid-bus>:<vendor-id>:<product-id>.<num>/savu/roccatsavu<minor>/sensor*

5.3.24 File obsolete/sysfs-driver-intel_pmc_bxt

These files allow sending arbitrary IPC commands to the PMC/SCU which may be dangerous. These will be removed eventually and should not be used in any new applications.

Has the following ABI:

- */sys/bus/platform/devices/INT34D2:00/simplecmd*
- */sys/bus/platform/devices/INT34D2:00/northpeak*

5.3.25 File obsolete/sysfs-firmware-acpi

Has the following ABI:

- */sys/firmware/acpi/hotplug/force_remove*

5.3.26 File obsolete/sysfs-gpio

Has the following ABI:

- */sys/class/gpio/*

5.3.27 File obsolete/sysfs-kernel-fadump_enabled

This ABI is renamed and moved to a new location /sys/kernel/fadump/enabled.

Has the following ABI:

- */sys/kernel/fadump_enabled*

5.3.28 File obsolete/sysfs-kernel-fadump_registered

This ABI is renamed and moved to a new location /sys/kernel/fadump/registered.

Has the following ABI:

- */sys/kernel/fadump_registered*

5.3.29 File obsolete/sysfs-kernel-fadump_release_mem

This ABI is renamed and moved to a new location /sys/kernel/fadump/release_mem.

Has the following ABI:

- */sys/kernel/fadump_release_mem*

5.4 ABI removed symbols

5.4.1 Symbols under /sys/bus

/sys/bus/nd/devices/regionX/nfit/ecc_unit_size

Defined on file *sysfs-bus-nfit*

(RO) Size of a write request to a DIMM that will not incur a read-modify-write cycle at the memory controller.

When the nfit driver initializes it runs an ARS (Address Range Scrub) operation across every pmem range. Part of that process involves determining the ARS capabilities of a given address range. One of the capabilities that is reported is the 'Clear Uncorrectable Error Range Length Unit Size' (see: ACPI 6.2 section 9.20.7.4 Function Index 1 - Query ARS Capabilities). This property indicates the boundary at which the NVDIMM may need to perform read-modify-write cycles to maintain ECC (Error Correcting Code) blocks.

5.4.2 Symbols under /sys/class

/sys/class/rfkill/rfkill[0-9]+/claim

Defined on file *sysfs-class-rfkill*

This file was deprecated because there no longer was a way to claim just control over a single rfkill instance. This file was scheduled to be removed in 2012, and was removed in 2016. Values: 0: Kernel handles events

5.4.3 Symbols under /sys/devices

/sys/devices/system/machinecheck/machinecheckX/tolerant

Defined on file *sysfs-mce*

Unused and obsolete after the advent of recoverable machine checks (see last sentence below) and those are present since 2010 (Nehalem).

Original description:

The entries appear for each CPU, but they are truly shared between all CPUs.

Tolerance level. When a machine check exception occurs for a non corrected machine check the kernel can take different actions.

Since machine check exceptions can happen any time it is sometimes risky for the kernel to kill a process because it defies normal kernel locking rules. The tolerance level configures how hard the kernel tries to recover even at some risk of deadlock. Higher tolerant values trade potentially better uptime with the risk of a crash or even corruption (for tolerant >= 3).

0	always panic on uncorrected errors, log corrected errors
1	panic or SIGBUS on uncorrected errors, log corrected errors
2	SIGBUS or log uncorrected errors, log corrected errors
3	never panic or SIGBUS, log all errors (for testing only)

Default: 1

Note this only makes a difference if the CPU allows recovery from a machine check exception. Current x86 CPUs generally do not.

5.4.4 Symbols under /sys/fs

/sys/fs/selinux/checkreqprot

Defined on file *sysfs-selinux-checkreqprot*

REMOVAL UPDATE: The SELinux checkreqprot functionality was removed in March 2023, the original deprecation notice is shown below.

The selinuxfs "checkreqprot" node allows SELinux to be configured to check the protection requested by userspace for mmap/mprotect calls instead of the actual protection applied by the kernel. This was a compatibility mechanism for legacy userspace and for the READ_IMPLIES_EXEC personality flag. However, if set to 1, it weakens security by allowing mappings to be made executable without authorization by policy. The default value of checkreqprot at boot was changed starting in Linux v4.4 to 0 (i.e. check the actual protection), and Android and Linux distributions have been explicitly writing a "0" to */sys/fs/selinux/checkreqprot* during initialization for some time. Support for setting checkreqprot to 1 will be removed no sooner than June 2021, at which point the kernel will always cease using checkreqprot internally and will always check the actual protections being applied upon mmap/mprotect calls. The checkreqprot selinuxfs node will remain for backward compatibility but will discard writes of the "0" value and will reject writes of the "1" value when this mechanism is removed.

/sys/fs/selinux/disable

Defined on file *sysfs-selinux-disable*

REMOVAL UPDATE: The SELinux runtime disable functionality was removed in March 2023, the original deprecation notice is shown below.

The selinuxfs "disable" node allows SELinux to be disabled at runtime prior to a policy being loaded into the kernel. If disabled via this mechanism, SELinux will remain disabled until the system is rebooted.

The preferred method of disabling SELinux is via the "selinux=0" boot parameter, but the selinuxfs "disable" node was created to make it easier for systems with primitive bootloaders that did not allow for easy modification of the kernel command line. Unfortunately, allowing for SELinux to be disabled at runtime makes it difficult to secure the kernel's LSM hooks using the "__ro_after_init" feature.

Thankfully, the need for the SELinux runtime disable appears to be gone, the default Kconfig configuration disables this selinuxfs node, and only one of the major distributions, Fedora, supports disabling SELinux at runtime. Fedora is in the process of removing the selinuxfs "disable"

node and once that is complete we will start the slow process of removing this code from the kernel.

More information on */sys/fs/selinux/disable* can be found under the CONFIG_SECURITY_SELINUX_DISABLE Kconfig option.

5.4.5 Symbols under /sys/kernel

/sys/kernel/fadump_release_opalcore

Defined on file *sysfs-kernel-fadump_release_opalcore*

write only The sysfs file is available when the system is booted to collect the dump on OPAL based machine. It used to release the memory used to collect the opalcore.

/sys/kernel/uids/<uid>/cpu_shares

Defined on file *sysfs-kernel-uids*

The */sys/kernel/uids/<uid>/cpu_shares* tunable is used to set the cpu bandwidth a user is allowed. This is a proportional value. What that means is that if there are two users logged in, each with an equal number of shares, then they will get equal CPU bandwidth. Another example would be, if User A has shares = 1024 and user B has shares = 2048, User B will get twice the CPU bandwidth user A will. For more details refer /scheduler/sched-design-CFS

5.4.6 Symbols under /sys/o2cb

/sys/o2cb symlink

Defined on file *o2cb*

This is a symlink: /sys/o2cb to /sys/fs/o2cb. The symlink is removed when new versions of ocfs2-tools which know to look in /sys/fs/o2cb are sufficiently prevalent. Don't code new software to look here, it should try /sys/fs/o2cb instead.

Users: ocfs2-tools. It's sufficient to mail proposed changes to ocfs2-devel@lists.linux.dev.

5.4.7 devfs

devfs

Defined on file *devfs*

devfs has been unmaintained for a number of years, has unfixable races, contains a naming policy within the kernel that is against the LSB, and can be replaced by using udev.

The files fs/devfs/*, include/linux/devfs_fs*.h were removed, along with the assorted devfs function calls throughout the kernel tree.

5.4.8 Symbols under dv1394

dv1394 (a.k.a. "OHCI-DV I/O support" for FireWire)

Defined on file *dv1394*

/dev/dv1394/* were character device files, one for each FireWire controller and for NTSC and PAL respectively, from which DV data could be received by read() or transmitted by write(). A few ioctl()s allowed limited control. This special-purpose interface has been superseded by libraw1394 + libiec61883 which are functionally equivalent, support HDV, and transparently work on top of the newer firewire kernel drivers.

Users:

ffmpeg/libavformat (if configured for DV1394)

5.4.9 ip_queue

ip_queue

Defined on file *ip_queue*

ip_queue has been replaced by nfnetlink_queue which provides more advanced queueing mechanism to user-space. The ip_queue module was already announced to become obsolete years ago.

5.4.10 Symbols under raw1394

raw1394 (a.k.a. "Raw IEEE1394 I/O support" for FireWire)

Defined on file *raw1394*

/dev/raw1394 was a character device file that allowed low-level access to FireWire buses. Its major drawbacks were its inability to implement sensible device security policies, and its low level of abstraction that required userspace clients to duplicate much of the kernel's ieee1394 core functionality.

Replaced by /dev/fw*, i.e. the <linux/firewire-cdev.h> ABI of firewire-core.

Users:

libraw1394 (works with firewire-cdev too, transparent to library ABI users)

5.4.11 tcp_dma_copybreak sysctl

tcp_dma_copybreak sysctl

Defined on file *net_dma*

Formerly the lower limit, in bytes, of the size of socket reads that will be offloaded to a DMA copy engine. Removed due to coherency issues of the cpu potentially touching the buffers while dma is in flight.

5.4.12 video1394 (a.k.a. "OHCI-1394 Video support" for FireWire)

video1394 (a.k.a. "OHCI-1394 Video support" for FireWire)

Defined on file *video1394*

/dev/video1394/* were character device files, one for each FireWire controller, which were used for isochronous I/O. It was added as an alternative to raw1394's isochronous I/O functionality which had performance issues in its first generation. Any video1394 user had to use raw1394 + libraw1394 too because video1394 did not provide asynchronous I/O for device discovery and configuration.

Replaced by /dev/fw*, i.e. the <linux/firewire-cdev.h> ABI of firewire-core.

Users:

libdc1394 (works with firewire-cdev too, transparent to library ABI users)

5.4.13 File removed/devfs

Has the following ABI:

- *devfs*

5.4.14 File removed/dv1394

Has the following ABI:

- *dv1394 (a.k.a. "OHCI-DV I/O support" for FireWire)*

5.4.15 File removed/ip_queue

Has the following ABI:

- *ip_queue*

5.4.16 File removed/net_dma

Has the following ABI:

- *tcp_dma_copybreak sysctl*

5.4.17 File removed/o2cb

Has the following ABI:

- */sys/o2cb symlink*

5.4.18 File removed/raw1394

Has the following ABI:

- *raw1394 (a.k.a. "Raw IEEE1394 I/O support" for FireWire)*

5.4.19 File removed/sysfs-bus-nfit

Has the following ABI:

- */sys/bus/nd/devices/regionX/nfit/ecc_unit_size*

5.4.20 File removed/sysfs-class-rfkill

rfkill - radio frequency (RF) connector kill switch support

For details to this subsystem look at /driver-api/rfkill.

Has the following ABI:

- */sys/class/rfkill/rfkill[0-9]+/claim*

5.4.21 File removed/sysfs-kernel-fadump_release_opalcore

This ABI is moved to /sys/firmware/opal/mpipl/release_core.

Has the following ABI:

- */sys/kernel/fadump_release_opalcore*

5.4.22 File removed/sysfs-kernel-uids

Has the following ABI:

- */sys/kernel/uids/<uid>/cpu_shares*

5.4.23 File removed/sysfs-mce

Has the following ABI:

- */sys/devices/system/machinecheck/machinecheckX/tolerant*

5.4.24 File removed/sysfs-selinux-checkreqprot

Has the following ABI:

- */sys/fs/selinux/checkreqprot*

5.4.25 File removed/sysfs-selinux-disable

Has the following ABI:

- */sys/fs/selinux/disable*

5.4.26 File removed/video1394

Has the following ABI:

- *video1394 (a.k.a. "OHCI-1394 Video support" for FireWire)*

CHAPTER

SIX

FEATURE STATUS ON ALL ARCHITECTURES

6.1 Subsystem: core

Feature	Kconfig / Description	Status per architecture
cBPF-JIT	`HAVE_CBPF_JIT` arch supports cBPF JIT optimizations	• **ok**: mips, powerpc sparc • **TODO**: alpha, arc, arm arm64, csky, hexagon loongarch, m68k, microblaze nios2, openrisc, parisc riscv, s390, sh, um, x86 xtensa
eBPF-JIT	`HAVE_EBPF_JIT` arch supports eBPF JIT optimizations	• **ok**: arm, arm64 loongarch, mips, powerpc riscv, s390, sparc, x86 • **TODO**: alpha, arc, csky hexagon, m68k, microblaze nios2, openrisc, parisc, sh um, xtensa
generic-idle-thread	`GENERIC_SMP_IDLE_THREAD` arch makes use of the generic SMP idle thread facility	• **ok**: alpha, arc, arm arm64, csky, hexagon loongarch, mips, openrisc parisc, powerpc, riscv s390, sh, sparc, x86 xtensa • **TODO**: m68k, microblaze nios2, um
jump-labels	`HAVE_ARCH_JUMP_LABEL` arch supports live patched, high efficiency branches	• **ok**: arc, arm, arm64 csky, loongarch, mips parisc, powerpc, riscv s390, sparc, x86, xtensa • **TODO**: alpha, hexagon m68k, microblaze, nios2 openrisc, sh, um
thread-info-in-task	`THREAD_INFO_IN_TASK` arch makes use of the core kernel facility to embed thread_info in task_struct	• **ok**: arm, arm64, parisc powerpc, riscv, s390, x86 • **TODO**

6.2 Subsystem: debug

Feature	Kconfig / Description	Status per architecture
debug-vm-pgtable	`ARCH_HAS_DEBUG_VM_PGTABLE` arch supports pgtable tests for semantics compliance	• **ok**: arc, arm64, parisc powerpc, riscv, s390, x86 xtensa • **TODO**: alpha, arm, csky hexagon, loongarch, m68k microblaze, mips, nios2 openrisc, sh, sparc, um
gcov-profile-all	`ARCH_HAS_GCOV_PROFILE_ALL` arch supports whole-kernel GCOV code coverage profiling	• **ok**: arm, arm64, csky microblaze, mips, powerpc riscv, s390, sh, um, x86 xtensa • **TODO**: alpha, arc hexagon, loongarch, m68k nios2, openrisc, parisc sparc
KASAN	`HAVE_ARCH_KASAN` arch supports the KASAN runtime memory checker	• **ok**: arm, arm64 loongarch, powerpc, riscv s390, um, x86, xtensa • **TODO**: alpha, arc, csky hexagon, m68k, microblaze mips, nios2, openrisc parisc, sh, sparc
kcov	`ARCH_HAS_KCOV` arch supports kcov for coverage-guided fuzzing	• **ok**: arm, arm64 loongarch, mips, powerpc riscv, s390, um, x86 xtensa • **TODO**: alpha, arc, csky hexagon, m68k, microblaze nios2, openrisc, parisc, sh sparc
kgdb	`HAVE_ARCH_KGDB` arch supports the kGDB kernel debugger	• **ok**: arc, arm, arm64 hexagon, loongarch microblaze, mips, nios2 parisc, powerpc, riscv, sh sparc, x86 • **TODO**: alpha, csky, m68k openrisc, s390, um, xtensa
kmemleak	`HAVE_DEBUG_KMEMLEAK` arch supports the kernel memory leak detector	• **ok**: arc, arm, arm64 csky, loongarch, microblaze mips, powerpc, riscv, s390 sh, sparc, um, x86, xtensa

6.3 Subsystem: io

Feature	Kconfig / Description	Status per architecture
dma-contiguous	`HAVE_DMA_CONTIGUOUS` arch supports the DMA CMA (continuous memory allocator)	• **ok**: arm, arm64, csky loongarch, microblaze, mips riscv, s390, x86, xtensa • **TODO**: alpha, arc hexagon, m68k, nios2 openrisc, parisc, powerpc sh, sparc, um

6.4 Subsystem: locking

Feature	Kconfig / Description	Status per architecture
cmpxchg-local	`HAVE_CMPXCHG_LOCAL` arch supports the this_cpu_cmpxchg() API	• **ok**: arm64, s390, x86 • **TODO**: alpha, arc, arm csky, hexagon, loongarch m68k, microblaze, mips nios2, openrisc, parisc powerpc, riscv, sh, sparc um, xtensa
lockdep	`LOCKDEP_SUPPORT` arch supports the runtime locking correctness debug facility	• **ok**: arc, arm, arm64 csky, hexagon, loongarch microblaze, mips, openrisc parisc, powerpc, riscv s390, sh, sparc, um, x86 xtensa • **TODO**: alpha, m68k nios2
queued-rwlocks	`ARCH_USE_QUEUED_RWLOCKS` arch supports queued rwlocks	• **ok**: arm64, csky loongarch, mips, openrisc powerpc, riscv, sparc, x86 xtensa • **TODO**: alpha, arc, arm hexagon, m68k, microblaze nios2, parisc, s390, sh, um
queued-spinlocks	`ARCH_USE_QUEUED_SPINLOCKS` arch supports queued spinlocks	• **ok**: arm64, csky loongarch, mips, openrisc powerpc, sparc, x86, xtensa • **TODO**: alpha, arc, arm hexagon, m68k, microblaze nios2, parisc, riscv, s390 sh, um

6.5 Subsystem: perf

Feature	Kconfig / Description	Status per architecture
kprobes-event	`HAVE_REGS_AND_STACK_ACCESS_API` arch supports kprobes with perf events	• **ok**: arc, arm, arm64 csky, hexagon, loongarch mips, parisc, powerpc riscv, s390, sh, sparc, x86 • **TODO**: alpha, m68k microblaze, nios2, openrisc um, xtensa
perf-regs	`HAVE_PERF_REGS` arch supports perf events register access	• **ok**: arm, arm64, csky loongarch, mips, powerpc riscv, s390, x86 • **TODO**: alpha, arc hexagon, m68k, microblaze nios2, openrisc, parisc, sh sparc, um, xtensa
perf-stackdump	`HAVE_PERF_USER_STACK_DUMP` arch supports perf events stack dumps	• **ok**: arm, arm64, csky loongarch, mips, powerpc riscv, s390, x86 • **TODO**: alpha, arc hexagon, m68k, microblaze nios2, openrisc, parisc, sh sparc, um, xtensa

6.6 Subsystem: sched

Feature	Kconfig / Description	Status per architecture
membarrier-sync-core	`ARCH_HAS_MEMBARRIER_SYNC_CORE` arch supports core serializing membarrier	• **ok**: arm, arm64, powerpc s390, x86 • **TODO**: alpha, arc, csky hexagon, loongarch, m68k microblaze, mips, nios2 openrisc, parisc, riscv, sh sparc, um, xtensa
numa-balancing	`ARCH_SUPPORTS_NUMA_BALANCING` arch supports NUMA balancing	• **ok**: arm64, loongarch powerpc, riscv, s390, x86 • **TODO**: alpha, mips sparc • **Not compatible**: arc arm, csky, hexagon, m68k microblaze, nios2, openrisc parisc, sh, um, xtensa

6.7 Subsystem: seccomp

Feature	Kconfig / Description	Status per architecture
seccomp-filter	`HAVE_ARCH_SECCOMP_FILTER` arch supports seccomp filters	• **ok**: arm, arm64, csky loongarch, m68k, mips parisc, powerpc, riscv s390, sh, um, x86, xtensa • **TODO**: alpha, arc hexagon, microblaze, nios2 openrisc, sparc

6.8 Subsystem: time

Feature	Kconfig / Description	Status per architecture
arch-tick-broadcast	`ARCH_HAS_TICK_BROADCAST` arch provides tick_broadcast()	• **ok**: arm, arm64 loongarch, mips, powerpc riscv, sh • **TODO**: alpha, arc, csky hexagon, m68k, microblaze nios2, openrisc, parisc s390, sparc, um, x86 xtensa
clockevents	`!LEGACY_TIMER_TICK` arch support generic clock events	• **ok**: alpha, arc, arm64 csky, hexagon, loongarch microblaze, mips, nios2 openrisc, powerpc, riscv s390, sh, sparc, um, x86 xtensa • **TODO**: arm, m68k, parisc
irq-time-acct	`HAVE_IRQ_TIME_ACCOUNTING` arch supports precise IRQ time accounting	• **ok**: arm, arm64 loongarch, mips, powerpc riscv, x86, xtensa • **TODO**: arc, csky hexagon, m68k, microblaze nios2, openrisc, sh, um • **Not compatible**: alpha parisc, s390, sparc
user-context-tracking	`HAVE_CONTEXT_TRACKING_USER` arch supports user context tracking for NO_HZ_FULL	• **ok**: arm, arm64, csky loongarch, mips, powerpc riscv, sparc, x86, xtensa • **TODO**: alpha, arc hexagon, m68k, microblaze nios2, openrisc, parisc s390, sh, um
virt-cpuacct	`HAVE_VIRT_CPU_ACCOUNTING` arch supports precise virtual CPU time accounting	• **ok**: alpha, arm, arm64 csky, loongarch, mips parisc, powerpc, s390 sparc, x86, xtensa • **TODO**: arc, hexagon m68k, microblaze, nios2 openrisc, riscv, sh, um

6.9 Subsystem: vm

Feature	Kconfig / Description	Status per architecture
batch-unmap-tlb-flush	`ARCH_WANT_BATCHED_UNMAP_TLB_FLUSH` arch supports deferral of TLB flush until multiple pages are unmapped	• **ok**: arm64, riscv, x86 • **TODO**: alpha, arc, arm csky, hexagon, loongarch mips, parisc, powerpc, s390 sh, sparc, xtensa • **Not compatible**: m68k microblaze, nios2, openrisc um
ELF-ASLR	`ARCH_WANT_DEFAULT_TOPDOWN_MMAP_LAYOUT` arch randomizes the stack, heap and binary images of ELF binaries	• **ok**: arm, arm64, csky loongarch, mips, parisc powerpc, riscv, s390, x86 • **TODO**: alpha, arc hexagon, m68k, microblaze nios2, openrisc, sh, sparc um, xtensa
huge-vmap	`HAVE_ARCH_HUGE_VMAP` arch supports the arch_vmap_pud_supported() and arch_vmap_pmd_supported() VM APIs	• **ok**: arm64, powerpc riscv, x86 • **TODO**: alpha, arc, arm csky, hexagon, loongarch m68k, microblaze, mips nios2, openrisc, parisc s390, sh, sparc, um, xtensa
ioremap_prot	`HAVE_IOREMAP_PROT` arch has ioremap_prot()	• **ok**: arc, arm64 loongarch, mips, powerpc s390, sh, x86 • **TODO**: alpha, arm, csky hexagon, m68k, microblaze nios2, openrisc, parisc riscv, sparc, um, xtensa
PG_uncached	`ARCH_USES_PG_UNCACHED` arch supports the PG_uncached page flag	• **ok**: x86 • **TODO**: alpha, arc, arm arm64, csky, hexagon loongarch, m68k, microblaze mips, nios2, openrisc parisc, powerpc, riscv s390, sh, sparc, um, xtensa
pte_special	`ARCH_HAS_PTE_SPECIAL` arch supports the pte_special()/pte_mkspecial() VM APIs	• **ok**: arc, arm, arm64 loongarch, mips, parisc powerpc, riscv, s390, sh

This section describes CPU vulnerabilities and their mitigations.

CHAPTER

SEVEN

HARDWARE VULNERABILITIES

This section describes CPU vulnerabilities and provides an overview of the possible mitigations along with guidance for selecting mitigations if they are configurable at compile, boot or run time.

7.1 Spectre Side Channels

Spectre is a class of side channel attacks that exploit branch prediction and speculative execution on modern CPUs to read memory, possibly bypassing access controls. Speculative execution side channel exploits do not modify memory but attempt to infer privileged data in the memory.

This document covers Spectre variant 1 and Spectre variant 2.

7.1.1 Affected processors

Speculative execution side channel methods affect a wide range of modern high performance processors, since most modern high speed processors use branch prediction and speculative execution.

The following CPUs are vulnerable:

- Intel Core, Atom, Pentium, and Xeon processors
- AMD Phenom, EPYC, and Zen processors
- IBM POWER and zSeries processors
- Higher end ARM processors
- Apple CPUs
- Higher end MIPS CPUs
- Likely most other high performance CPUs. Contact your CPU vendor for details.

Whether a processor is affected or not can be read out from the Spectre vulnerability files in sysfs. See *Spectre system information*.

7.1.2 Related CVEs

The following CVE entries describe Spectre variants:

CVE-2017-5753	Bounds check bypass	Spectre variant 1
CVE-2017-5715	Branch target injection	Spectre variant 2
CVE-2019-1125	Spectre v1 swapgs	Spectre variant 1 (swapgs)

7.1.3 Problem

CPUs use speculative operations to improve performance. That may leave traces of memory accesses or computations in the processor's caches, buffers, and branch predictors. Malicious software may be able to influence the speculative execution paths, and then use the side effects of the speculative execution in the CPUs' caches and buffers to infer privileged data touched during the speculative execution.

Spectre variant 1 attacks take advantage of speculative execution of conditional branches, while Spectre variant 2 attacks use speculative execution of indirect branches to leak privileged memory. See *[1] [5] [6] [7] [10] [11]*.

7.1.4 Spectre variant 1 (Bounds Check Bypass)

The bounds check bypass attack *[2]* takes advantage of speculative execution that bypasses conditional branch instructions used for memory access bounds check (e.g. checking if the index of an array results in memory access within a valid range). This results in memory accesses to invalid memory (with out-of-bound index) that are done speculatively before validation checks resolve. Such speculative memory accesses can leave side effects, creating side channels which leak information to the attacker.

There are some extensions of Spectre variant 1 attacks for reading data over the network, see *[12]*. However such attacks are difficult, low bandwidth, fragile, and are considered low risk.

Note that, despite ”Bounds Check Bypass” name, Spectre variant 1 is not only about user-controlled array bounds checks. It can affect any conditional checks. The kernel entry code interrupt, exception, and NMI handlers all have conditional swapgs checks. Those may be problematic in the context of Spectre v1, as kernel code can speculatively run with a user GS.

7.1.5 Spectre variant 2 (Branch Target Injection)

The branch target injection attack takes advantage of speculative execution of indirect branches *[3]*. The indirect branch predictors inside the processor used to guess the target of indirect branches can be influenced by an attacker, causing gadget code to be speculatively executed, thus exposing sensitive data touched by the victim. The side effects left in the CPU's caches during speculative execution can be measured to infer data values.

In Spectre variant 2 attacks, the attacker can steer speculative indirect branches in the victim to gadget code by poisoning the branch target buffer of a CPU used for predicting indirect branch addresses. Such poisoning could be done by indirect branching into existing code, with the address offset of the indirect branch under the attacker's control. Since the branch prediction on impacted hardware does not fully disambiguate branch address and uses the

offset for prediction, this could cause privileged code's indirect branch to jump to a gadget code with the same offset.

The most useful gadgets take an attacker-controlled input parameter (such as a register value) so that the memory read can be controlled. Gadgets without input parameters might be possible, but the attacker would have very little control over what memory can be read, reducing the risk of the attack revealing useful data.

One other variant 2 attack vector is for the attacker to poison the return stack buffer (RSB) *[13]* to cause speculative subroutine return instruction execution to go to a gadget. An attacker's imbalanced subroutine call instructions might ”poison” entries in the return stack buffer which are later consumed by a victim's subroutine return instructions. This attack can be mitigated by flushing the return stack buffer on context switch, or virtual machine (VM) exit.

On systems with simultaneous multi-threading (SMT), attacks are possible from the sibling thread, as level 1 cache and branch target buffer (BTB) may be shared between hardware threads in a CPU core. A malicious program running on the sibling thread may influence its peer's BTB to steer its indirect branch speculations to gadget code, and measure the speculative execution's side effects left in level 1 cache to infer the victim's data.

Yet another variant 2 attack vector is for the attacker to poison the Branch History Buffer (BHB) to speculatively steer an indirect branch to a specific Branch Target Buffer (BTB) entry, even if the entry isn't associated with the source address of the indirect branch. Specifically, the BHB might be shared across privilege levels even in the presence of Enhanced IBRS.

Currently the only known real-world BHB attack vector is via unprivileged eBPF. Therefore, it's highly recommended to not enable unprivileged eBPF, especially when eIBRS is used (without retpolines). For a full mitigation against BHB attacks, it's recommended to use retpolines (or eIBRS combined with retpolines).

7.1.6 Attack scenarios

The following list of attack scenarios have been anticipated, but may not cover all possible attack vectors.

1. A user process attacking the kernel

Spectre variant 1

> The attacker passes a parameter to the kernel via a register or via a known address in memory during a syscall. Such parameter may be used later by the kernel as an index to an array or to derive a pointer for a Spectre variant 1 attack. The index or pointer is invalid, but bound checks are bypassed in the code branch taken for speculative execution. This could cause privileged memory to be accessed and leaked.
>
> For kernel code that has been identified where data pointers could potentially be influenced for Spectre attacks, new ”nospec” accessor macros are used to prevent speculative loading of data.

Spectre variant 1 (swapgs)

An attacker can train the branch predictor to speculatively skip the swapgs path for an interrupt or exception. If they initialize the GS register to a user-space value, if the swapgs is speculatively skipped, subsequent GS-related percpu accesses in the speculation window will be done with the attacker-controlled GS value. This could cause privileged memory to be accessed and leaked.

For example:

```
if (coming from user space)
    swapgs
mov %gs:<percpu_offset>, %reg
mov (%reg), %reg1
```

When coming from user space, the CPU can speculatively skip the swapgs, and then do a speculative percpu load using the user GS value. So the user can speculatively force a read of any kernel value. If a gadget exists which uses the percpu value as an address in another load/store, then the contents of the kernel value may become visible via an L1 side channel attack.

A similar attack exists when coming from kernel space. The CPU can speculatively do the swapgs, causing the user GS to get used for the rest of the speculative window.

Spectre variant 2

A spectre variant 2 attacker can *poison* the branch target buffer (BTB) before issuing syscall to launch an attack. After entering the kernel, the kernel could use the poisoned branch target buffer on indirect jump and jump to gadget code in speculative execution.

If an attacker tries to control the memory addresses leaked during speculative execution, he would also need to pass a parameter to the gadget, either through a register or a known address in memory. After the gadget has executed, he can measure the side effect.

The kernel can protect itself against consuming poisoned branch target buffer entries by using return trampolines (also known as ”retpoline”) *[3] [9]* for all indirect branches. Return trampolines trap speculative execution paths to prevent jumping to gadget code during speculative execution. x86 CPUs with Enhanced Indirect Branch Restricted Speculation (Enhanced IBRS) available in hardware should use the feature to mitigate Spectre variant 2 instead of retpoline. Enhanced IBRS is more efficient than retpoline.

There may be gadget code in firmware which could be exploited with Spectre variant 2 attack by a rogue user process. To mitigate such attacks on x86, Indirect Branch Restricted Speculation (IBRS) feature is turned on before the kernel invokes any firmware code.

2. A user process attacking another user process

A malicious user process can try to attack another user process, either via a context switch on the same hardware thread, or from the sibling hyperthread sharing a physical processor core on simultaneous multi-threading (SMT) system.

Spectre variant 1 attacks generally require passing parameters between the processes, which needs a data passing relationship, such as remote procedure calls (RPC). Those parameters are used in gadget code to derive invalid data pointers accessing privileged memory in the attacked process.

Spectre variant 2 attacks can be launched from a rogue process by *poisoning* the branch target buffer. This can influence the indirect branch targets for a victim process that either runs later on the same hardware thread, or running concurrently on a sibling hardware thread sharing the same physical core.

A user process can protect itself against Spectre variant 2 attacks by using the prctl() syscall to disable indirect branch speculation for itself. An administrator can also cordon off an unsafe process from polluting the branch target buffer by disabling the process's indirect branch speculation. This comes with a performance cost from not using indirect branch speculation and clearing the branch target buffer. When SMT is enabled on x86, for a process that has indirect branch speculation disabled, Single Threaded Indirect Branch Predictors (STIBP) *[4]* are turned on to prevent the sibling thread from controlling branch target buffer. In addition, the Indirect Branch Prediction Barrier (IBPB) is issued to clear the branch target buffer when context switching to and from such process.

On x86, the return stack buffer is stuffed on context switch. This prevents the branch target buffer from being used for branch prediction when the return stack buffer underflows while switching to a deeper call stack. Any poisoned entries in the return stack buffer left by the previous process will also be cleared.

User programs should use address space randomization to make attacks more difficult (Set /proc/sys/kernel/randomize_va_space = 1 or 2).

3. A virtualized guest attacking the host

The attack mechanism is similar to how user processes attack the kernel. The kernel is entered via hyper-calls or other virtualization exit paths.

For Spectre variant 1 attacks, rogue guests can pass parameters (e.g. in registers) via hyper-calls to derive invalid pointers to speculate into privileged memory after entering the kernel. For places where such kernel code has been identified, nospec accessor macros are used to stop speculative memory access.

For Spectre variant 2 attacks, rogue guests can *poison* the branch target buffer or return stack buffer, causing the kernel to jump to gadget code in the speculative execution paths.

To mitigate variant 2, the host kernel can use return trampolines for indirect branches to bypass the poisoned branch target buffer, and flushing the return stack buffer on VM exit. This prevents rogue guests from affecting indirect branching in the host kernel.

To protect host processes from rogue guests, host processes can have indirect branch speculation disabled via prctl(). The branch target buffer is cleared before context switching to such processes.

4. A virtualized guest attacking other guest

A rogue guest may attack another guest to get data accessible by the other guest.

Spectre variant 1 attacks are possible if parameters can be passed between guests. This may be done via mechanisms such as shared memory or message passing. Such parameters could be used to derive data pointers to privileged data in guest. The privileged data could be accessed by gadget code in the victim's speculation paths.

Spectre variant 2 attacks can be launched from a rogue guest by *poisoning* the branch target buffer or the return stack buffer. Such poisoned entries could be used to influence speculation execution paths in the victim guest.

Linux kernel mitigates attacks to other guests running in the same CPU hardware thread by flushing the return stack buffer on VM exit, and clearing the branch target buffer before switching to a new guest.

If SMT is used, Spectre variant 2 attacks from an untrusted guest in the sibling hyperthread can be mitigated by the administrator, by turning off the unsafe guest's indirect branch speculation via prctl(). A guest can also protect itself by turning on microcode based mitigations (such as IBPB or STIBP on x86) within the guest.

7.1.7 Spectre system information

The Linux kernel provides a sysfs interface to enumerate the current mitigation status of the system for Spectre: whether the system is vulnerable, and which mitigations are active.

The sysfs file showing Spectre variant 1 mitigation status is:

/sys/devices/system/cpu/vulnerabilities/spectre_v1

The possible values in this file are:

'Not affected'	The processor is not vulnerable.
'Vulnerable: __user pointer sanitization and usercopy barriers only; no swapgs barriers'	The swapgs protections are disabled; otherwise it has protection in the kernel on a case by case base with explicit pointer sanitation and usercopy LFENCE barriers.
'Mitigation: usercopy/swapgs barriers and __user pointer sanitization'	Protection in the kernel on a case by case base with explicit pointer sanitation, usercopy LFENCE barriers, and swapgs LFENCE barriers.

However, the protections are put in place on a case by case basis, and there is no guarantee that all possible attack vectors for Spectre variant 1 are covered.

The spectre_v2 kernel file reports if the kernel has been compiled with retpoline mitigation or if the CPU has hardware mitigation, and if the CPU has support for additional process-specific mitigation.

This file also reports CPU features enabled by microcode to mitigate attack between user processes:

1. Indirect Branch Prediction Barrier (IBPB) to add additional isolation between processes of different users.
2. Single Thread Indirect Branch Predictors (STIBP) to add additional isolation between CPU threads running on the same core.

These CPU features may impact performance when used and can be enabled per process on a case-by-case base.

The sysfs file showing Spectre variant 2 mitigation status is:

/sys/devices/system/cpu/vulnerabilities/spectre_v2

The possible values in this file are:

- Kernel status:

'Not affected'	The processor is not vulnerable
'Mitigation: None'	Vulnerable, no mitigation
'Mitigation: Retpolines'	Use Retpoline thunks
'Mitigation: LFENCE'	Use LFENCE instructions
'Mitigation: Enhanced IBRS'	Hardware-focused mitigation
'Mitigation: Enhanced IBRS + Retpolines'	Hardware-focused + Retpolines
'Mitigation: Enhanced IBRS + LFENCE'	Hardware-focused + LFENCE

- Firmware status: Show if Indirect Branch Restricted Speculation (IBRS) is used to protect against Spectre variant 2 attacks when calling firmware (x86 only).

'IBRS_FW'	Protection against user program attacks when calling firmware

- Indirect branch prediction barrier (IBPB) status for protection between processes of different users. This feature can be controlled through prctl() per process, or through kernel command line options. This is an x86 only feature. For more details see below.

'IBPB: disabled'	IBPB unused
'IBPB: always-on'	Use IBPB on all tasks
'IBPB: conditional'	Use IBPB on SECCOMP or indirect branch restricted tasks

- Single threaded indirect branch prediction (STIBP) status for protection between different hyper threads. This feature can be controlled through prctl per process, or through kernel command line options. This is x86 only feature. For more details see below.

'STIBP: disabled'	STIBP unused
'STIBP: forced'	Use STIBP on all tasks
'STIBP: conditional'	Use STIBP on SECCOMP or indirect branch restricted tasks

- Return stack buffer (RSB) protection status:

'RSB filling'	Protection of RSB on context switch enabled

- EIBRS Post-barrier Return Stack Buffer (PBRSB) protection status:

'PBRSB-eIBRS: SW sequence'	CPU is affected and protection of RSB on VMEXIT enabled
'PBRSB-eIBRS: Vulnerable'	CPU is vulnerable
'PBRSB-eIBRS: Not affected'	CPU is not affected by PBRSB

Full mitigation might require a microcode update from the CPU vendor. When the necessary microcode is not available, the kernel will report vulnerability.

7.1.8 Turning on mitigation for Spectre variant 1 and Spectre variant 2

1. Kernel mitigation

Spectre variant 1

For the Spectre variant 1, vulnerable kernel code (as determined by code audit or scanning tools) is annotated on a case by case basis to use nospec accessor macros for bounds clipping *[2]* to avoid any usable disclosure gadgets. However, it may not cover all attack vectors for Spectre variant 1.

Copy-from-user code has an LFENCE barrier to prevent the access_ok() check from being mis-speculated. The barrier is done by the barrier_nospec() macro.

For the swapgs variant of Spectre variant 1, LFENCE barriers are added to interrupt, exception and NMI entry where needed. These barriers are done by the FENCE_SWAPGS_KERNEL_ENTRY and FENCE_SWAPGS_USER_ENTRY macros.

Spectre variant 2

For Spectre variant 2 mitigation, the compiler turns indirect calls or jumps in the kernel into equivalent return trampolines (retpolines) *[3] [9]* to go to the target addresses. Speculative execution paths under retpolines are trapped in an infinite loop to prevent any speculative execution jumping to a gadget.

To turn on retpoline mitigation on a vulnerable CPU, the kernel needs to be compiled with a gcc compiler that supports the -mindirect-branch=thunk-extern -mindirect-branch-register options. If the kernel is compiled with a Clang compiler, the compiler needs to support -mretpoline-external-thunk option. The kernel config CONFIG_RETPOLINE needs to be turned on, and the CPU needs to run with the latest updated microcode.

On Intel Skylake-era systems the mitigation covers most, but not all, cases. See *[3]* for more details.

On CPUs with hardware mitigation for Spectre variant 2 (e.g. IBRS or enhanced IBRS on x86), retpoline is automatically disabled at run time.

Systems which support enhanced IBRS (eIBRS) enable IBRS protection once at boot, by setting the IBRS bit, and they're automatically protected against Spectre v2 variant attacks.

On Intel's enhanced IBRS systems, this includes cross-thread branch target injections on SMT systems (STIBP). In other words, Intel eIBRS enables STIBP, too.

AMD Automatic IBRS does not protect userspace, and Legacy IBRS systems clear the IBRS bit on exit to userspace, therefore both explicitly enable STIBP.

The retpoline mitigation is turned on by default on vulnerable CPUs. It can be forced on or off by the administrator via the kernel command line and sysfs control files. See *Mitigation control on the kernel command line*.

On x86, indirect branch restricted speculation is turned on by default before invoking any firmware code to prevent Spectre variant 2 exploits using the firmware.

Using kernel address space randomization (CONFIG_RANDOMIZE_BASE=y and CONFIG_SLAB_FREELIST_RANDOM=y in the kernel configuration) makes attacks on the kernel generally more difficult.

2. User program mitigation

User programs can mitigate Spectre variant 1 using LFENCE or "bounds clipping". For more details see *[2]*.

For Spectre variant 2 mitigation, individual user programs can be compiled with return trampolines for indirect branches. This protects them from consuming poisoned entries in the branch target buffer left by malicious software.

On legacy IBRS systems, at return to userspace, implicit STIBP is disabled because the kernel clears the IBRS bit. In this case, the userspace programs can disable indirect branch speculation via prctl() (See Documentation/userspace-api/spec_ctrl.rst). On x86, this will turn on STIBP to guard against attacks from the sibling thread when the user program is running, and use IBPB to flush the branch target buffer when switching to/from the program.

Restricting indirect branch speculation on a user program will also prevent the program from launching a variant 2 attack on x86. Administrators can change that behavior via the kernel command line and sysfs control files. See *Mitigation control on the kernel command line*.

Programs that disable their indirect branch speculation will have more overhead and run slower.

User programs should use address space randomization (/proc/sys/kernel/randomize_va_space = 1 or 2) to make attacks more difficult.

3. VM mitigation

Within the kernel, Spectre variant 1 attacks from rogue guests are mitigated on a case by case basis in VM exit paths. Vulnerable code uses nospec accessor macros for "bounds clipping", to avoid any usable disclosure gadgets. However, this may not cover all variant 1 attack vectors.

For Spectre variant 2 attacks from rogue guests to the kernel, the Linux kernel uses retpoline or Enhanced IBRS to prevent consumption of poisoned entries in branch target buffer left by rogue guests. It also flushes the return stack buffer on every VM

exit to prevent a return stack buffer underflow so poisoned branch target buffer could be used, or attacker guests leaving poisoned entries in the return stack buffer.

To mitigate guest-to-guest attacks in the same CPU hardware thread, the branch target buffer is sanitized by flushing before switching to a new guest on a CPU.

The above mitigations are turned on by default on vulnerable CPUs.

To mitigate guest-to-guest attacks from sibling thread when SMT is in use, an untrusted guest running in the sibling thread can have its indirect branch speculation disabled by administrator via prctl().

The kernel also allows guests to use any microcode based mitigation they choose to use (such as IBPB or STIBP on x86) to protect themselves.

7.1.9 Mitigation control on the kernel command line

Spectre variant 2 mitigation can be disabled or force enabled at the kernel command line.

nospectre_v1

[X86,PPC] Disable mitigations for Spectre Variant 1 (bounds check bypass). With this option data leaks are possible in the system.

nospectre_v2

[X86] Disable all mitigations for the Spectre variant 2 (indirect branch prediction) vulnerability. System may allow data leaks with this option, which is equivalent to spectre_v2=off.

spectre_v2=

[X86] Control mitigation of Spectre variant 2 (indirect branch speculation) vulnerability. The default operation protects the kernel from user space attacks.

on
unconditionally enable, implies spectre_v2_user=on

off
unconditionally disable, implies spectre_v2_user=off

auto
kernel detects whether your CPU model is vulnerable

Selecting 'on' will, and 'auto' may, choose a mitigation method at run time according to the CPU, the available microcode, the setting of the CONFIG_RETPOLINE configuration option, and the compiler with which the kernel was built.

Selecting 'on' will also enable the mitigation against user space to user space task attacks.

Selecting 'off' will disable both the kernel and the user space protections.

Specific mitigations can also be selected manually:

retpoline auto pick between generic,lfence retpoline,generic Retpolines retpoline,lfence LFENCE; indirect branch retpoline,amd alias for retpoline,lfence eibrs Enhanced/Auto IBRS eibrs,retpoline Enhanced/Auto IBRS

+ Retpolines eibrs,lfence Enhanced/Auto IBRS + LFENCE ibrs use IBRS to protect kernel

Not specifying this option is equivalent to spectre_v2=auto.

In general the kernel by default selects reasonable mitigations for the current CPU. To disable Spectre variant 2 mitigations, boot with spectre_v2=off. Spectre variant 1 mitigations cannot be disabled.

For spectre_v2_user see *The kernel's command-line parameters*

7.1.10 Mitigation selection guide

1. Trusted userspace

If all userspace applications are from trusted sources and do not execute externally supplied untrusted code, then the mitigations can be disabled.

2. Protect sensitive programs

For security-sensitive programs that have secrets (e.g. crypto keys), protection against Spectre variant 2 can be put in place by disabling indirect branch speculation when the program is running (See Documentation/userspace-api/spec_ctrl.rst).

3. Sandbox untrusted programs

Untrusted programs that could be a source of attacks can be cordoned off by disabling their indirect branch speculation when they are run (See Documentation/userspace-api/spec_ctrl.rst). This prevents untrusted programs from polluting the branch target buffer. This behavior can be changed via the kernel command line and sysfs control files. See *Mitigation control on the kernel command line*.

3. High security mode

All Spectre variant 2 mitigations can be forced on at boot time for all programs (See the ”on” option in *Mitigation control on the kernel command line*). This will add overhead as indirect branch speculations for all programs will be restricted.

On x86, branch target buffer will be flushed with IBPB when switching to a new program. STIBP is left on all the time to protect programs against variant 2 attacks originating from programs running on sibling threads.

Alternatively, STIBP can be used only when running programs whose indirect branch speculation is explicitly disabled, while IBPB is still used all the time when switching to a new program to clear the branch target buffer (See ”ibpb” option in *Mitigation control on the kernel command line*). This ”ibpb” option has less performance cost than the ”on” option, which leaves STIBP on all the time.

7.1.11 References on Spectre

Intel white papers:

[1] Intel analysis of speculative execution side channels.

[2] Bounds check bypass.

[3] Deep dive: Retpoline: A branch target injection mitigation.

[4] Deep Dive: Single Thread Indirect Branch Predictors.

AMD white papers:

[5] AMD64 technology indirect branch control extension.

[6] Software techniques for managing speculation on AMD processors.

ARM white papers:

[7] Cache speculation side-channels.

[8] Cache speculation issues update.

Google white paper:

[9] Retpoline: a software construct for preventing branch-target-injection.

MIPS white paper:

[10] MIPS: response on speculative execution and side channel vulnerabilities.

Academic papers:

[11] Spectre Attacks: Exploiting Speculative Execution.

[12] NetSpectre: Read Arbitrary Memory over Network.

[13] Spectre Returns! Speculation Attacks using the Return Stack Buffer.

7.2 L1TF - L1 Terminal Fault

L1 Terminal Fault is a hardware vulnerability which allows unprivileged speculative access to data which is available in the Level 1 Data Cache when the page table entry controlling the virtual address, which is used for the access, has the Present bit cleared or other reserved bits set.

7.2.1 Affected processors

This vulnerability affects a wide range of Intel processors. The vulnerability is not present on:

- Processors from AMD, Centaur and other non Intel vendors
- Older processor models, where the CPU family is < 6
- A range of Intel ATOM processors (Cedarview, Cloverview, Lincroft, Penwell, Pineview, Silvermont, Airmont, Merrifield)
- The Intel XEON PHI family

- Intel processors which have the ARCH_CAP_RDCL_NO bit set in the IA32_ARCH_CAPABILITIES MSR. If the bit is set the CPU is not affected by the Meltdown vulnerability either. These CPUs should become available by end of 2018.

Whether a processor is affected or not can be read out from the L1TF vulnerability file in sysfs. See *L1TF system information*.

7.2.2 Related CVEs

The following CVE entries are related to the L1TF vulnerability:

CVE-2018-3615	L1 Terminal Fault	SGX related aspects
CVE-2018-3620	L1 Terminal Fault	OS, SMM related aspects
CVE-2018-3646	L1 Terminal Fault	Virtualization related aspects

7.2.3 Problem

If an instruction accesses a virtual address for which the relevant page table entry (PTE) has the Present bit cleared or other reserved bits set, then speculative execution ignores the invalid PTE and loads the referenced data if it is present in the Level 1 Data Cache, as if the page referenced by the address bits in the PTE was still present and accessible.

While this is a purely speculative mechanism and the instruction will raise a page fault when it is retired eventually, the pure act of loading the data and making it available to other speculative instructions opens up the opportunity for side channel attacks to unprivileged malicious code, similar to the Meltdown attack.

While Meltdown breaks the user space to kernel space protection, L1TF allows to attack any physical memory address in the system and the attack works across all protection domains. It allows an attack of SGX and also works from inside virtual machines because the speculation bypasses the extended page table (EPT) protection mechanism.

7.2.4 Attack scenarios

1. Malicious user space

Operating Systems store arbitrary information in the address bits of a PTE which is marked non present. This allows a malicious user space application to attack the physical memory to which these PTEs resolve. In some cases user-space can maliciously influence the information encoded in the address bits of the PTE, thus making attacks more deterministic and more practical.

The Linux kernel contains a mitigation for this attack vector, PTE inversion, which is permanently enabled and has no performance impact. The kernel ensures that the address bits of PTEs, which are not marked present, never point to cacheable physical memory space.

A system with an up to date kernel is protected against attacks from malicious user space applications.

2. Malicious guest in a virtual machine

The fact that L1TF breaks all domain protections allows malicious guest OSes, which can control the PTEs directly, and malicious guest user space applications, which run on an unprotected guest kernel lacking the PTE inversion mitigation for L1TF, to attack physical host memory.

A special aspect of L1TF in the context of virtualization is symmetric multi threading (SMT). The Intel implementation of SMT is called HyperThreading. The fact that Hyperthreads on the affected processors share the L1 Data Cache (L1D) is important for this. As the flaw allows only to attack data which is present in L1D, a malicious guest running on one Hyperthread can attack the data which is brought into the L1D by the context which runs on the sibling Hyperthread of the same physical core. This context can be host OS, host user space or a different guest.

If the processor does not support Extended Page Tables, the attack is only possible, when the hypervisor does not sanitize the content of the effective (shadow) page tables.

While solutions exist to mitigate these attack vectors fully, these mitigations are not enabled by default in the Linux kernel because they can affect performance significantly. The kernel provides several mechanisms which can be utilized to address the problem depending on the deployment scenario. The mitigations, their protection scope and impact are described in the next sections.

The default mitigations and the rationale for choosing them are explained at the end of this document. See *Default mitigations*.

7.2.5 L1TF system information

The Linux kernel provides a sysfs interface to enumerate the current L1TF status of the system: whether the system is vulnerable, and which mitigations are active. The relevant sysfs file is:

/sys/devices/system/cpu/vulnerabilities/l1tf

The possible values in this file are:

'Not affected'	The processor is not vulnerable
'Mitigation: PTE Inversion'	The host protection is active

If KVM/VMX is enabled and the processor is vulnerable then the following information is appended to the 'Mitigation: PTE Inversion' part:

- SMT status:

'VMX: SMT vulnerable'	SMT is enabled
'VMX: SMT disabled'	SMT is disabled

- L1D Flush mode:

'L1D vulnerable'	L1D flushing is disabled
'L1D conditional cache flushes'	L1D flush is conditionally enabled
'L1D cache flushes'	L1D flush is unconditionally enabled

The resulting grade of protection is discussed in the following sections.

7.2.6 Host mitigation mechanism

The kernel is unconditionally protected against L1TF attacks from malicious user space running on the host.

7.2.7 Guest mitigation mechanisms

1. L1D flush on VMENTER

To make sure that a guest cannot attack data which is present in the L1D the hypervisor flushes the L1D before entering the guest.

Flushing the L1D evicts not only the data which should not be accessed by a potentially malicious guest, it also flushes the guest data. Flushing the L1D has a performance impact as the processor has to bring the flushed guest data back into the L1D. Depending on the frequency of VMEXIT/VMENTER and the type of computations in the guest performance degradation in the range of 1% to 50% has been observed. For scenarios where guest VMEXIT/VMENTER are rare the performance impact is minimal. Virtio and mechanisms like posted interrupts are designed to confine the VMEXITs to a bare minimum, but specific configurations and application scenarios might still suffer from a high VMEXIT rate.

The kernel provides two L1D flush modes:

- conditional ('cond')
- unconditional ('always')

The conditional mode avoids L1D flushing after VMEXITs which execute only audited code paths before the corresponding VMENTER. These code paths have been verified that they cannot expose secrets or other interesting data to an attacker, but they can leak information about the address space layout of the hypervisor.

Unconditional mode flushes L1D on all VMENTER invocations and provides maximum protection. It has a higher overhead than the conditional mode. The overhead cannot be quantified correctly as it depends on the workload scenario and the resulting number of VMEXITs.

The general recommendation is to enable L1D flush on VMENTER. The kernel defaults to conditional mode on affected processors.

Note, that L1D flush does not prevent the SMT problem because the sibling thread will also bring back its data into the L1D which makes it attackable again.

L1D flush can be controlled by the administrator via the kernel command line and sysfs control files. See *Mitigation control on the kernel command line* and *Mitigation control for KVM - module parameter*.

2. Guest VCPU confinement to dedicated physical cores

To address the SMT problem, it is possible to make a guest or a group of guests affine to one or more physical cores. The proper mechanism for that is to utilize exclusive cpusets to ensure that no other guest or host tasks can run on these cores.

If only a single guest or related guests run on sibling SMT threads on the same physical core then they can only attack their own memory and restricted parts of the host memory.

Host memory is attackable, when one of the sibling SMT threads runs in host OS (hypervisor) context and the other in guest context. The amount of valuable information from the host OS context depends on the context which the host OS executes, i.e. interrupts, soft interrupts and kernel threads. The amount of valuable data from these contexts cannot be declared as non-interesting for an attacker without deep inspection of the code.

Note, that assigning guests to a fixed set of physical cores affects the ability of the scheduler to do load balancing and might have negative effects on CPU utilization depending on the hosting scenario. Disabling SMT might be a viable alternative for particular scenarios.

For further information about confining guests to a single or to a group of cores consult the cpusets documentation:

https://www.kernel.org/doc/Documentation/admin-guide/cgroup-v1/cpusets.rst

3. Interrupt affinity

Interrupts can be made affine to logical CPUs. This is not universally true because there are types of interrupts which are truly per CPU interrupts, e.g. the local timer interrupt. Aside of that multi queue devices affine their interrupts to single CPUs or groups of CPUs per queue without allowing the administrator to control the affinities.

Moving the interrupts, which can be affinity controlled, away from CPUs which run untrusted guests, reduces the attack vector space.

Whether the interrupts with are affine to CPUs, which run untrusted guests, provide interesting data for an attacker depends on the system configuration and the scenarios which run on the system. While for some of the interrupts it can be assumed that they won't expose interesting information beyond exposing hints about the host OS memory layout, there is no way to make general assumptions.

Interrupt affinity can be controlled by the administrator via the /proc/irq/$NR/smp_affinity[_list] files. Limited documentation is available at:

https://www.kernel.org/doc/Documentation/core-api/irq/irq-affinity.rst

4. SMT control

To prevent the SMT issues of L1TF it might be necessary to disable SMT completely. Disabling SMT can have a significant performance impact, but the impact depends on the hosting scenario and the type of workloads. The impact of disabling SMT needs also to be weighted against the impact of other mitigation solutions like confining guests to dedicated cores.

The kernel provides a sysfs interface to retrieve the status of SMT and to control it. It also provides a kernel command line interface to control SMT.

The kernel command line interface consists of the following options:

nosmt	Affects the bring up of the secondary CPUs during boot. The kernel tries to bring all present CPUs online during the boot process. "nosmt" makes sure that from each physical core only one - the so called primary (hyper) thread is activated. Due to a design flaw of Intel processors related to Machine Check Exceptions the non primary siblings have to be brought up at least partially and are then shut down again. "nosmt" can be undone via the sysfs interface.
nosmt=force	Has the same effect as "nosmt" but it does not allow to undo the SMT disable via the sysfs interface.

The sysfs interface provides two files:

- /sys/devices/system/cpu/smt/control
- /sys/devices/system/cpu/smt/active

/sys/devices/system/cpu/smt/control:

This file allows to read out the SMT control state and provides the ability to disable or (re)enable SMT. The possible states are:

on	SMT is supported by the CPU and enabled. All logical CPUs can be onlined and offlined without restrictions.
off	SMT is supported by the CPU and disabled. Only the so called primary SMT threads can be onlined and offlined without restrictions. An attempt to online a non-primary sibling is rejected
forceoff	Same as 'off' but the state cannot be controlled. Attempts to write to the control file are rejected.
notsupported	The processor does not support SMT. It's therefore not affected by the SMT implications of L1TF. Attempts to write to the control file are rejected.

The possible states which can be written into this file to control SMT state are:

- on

- off
- forceoff

/sys/devices/system/cpu/smt/active:

This file reports whether SMT is enabled and active, i.e. if on any physical core two or more sibling threads are online.

SMT control is also possible at boot time via the l1tf kernel command line parameter in combination with L1D flush control. See *Mitigation control on the kernel command line*.

5. Disabling EPT

Disabling EPT for virtual machines provides full mitigation for L1TF even with SMT enabled, because the effective page tables for guests are managed and sanitized by the hypervisor. Though disabling EPT has a significant performance impact especially when the Meltdown mitigation KPTI is enabled.

EPT can be disabled in the hypervisor via the 'kvm-intel.ept' parameter.

There is ongoing research and development for new mitigation mechanisms to address the performance impact of disabling SMT or EPT.

7.2.8 Mitigation control on the kernel command line

The kernel command line allows to control the L1TF mitigations at boot time with the option ”l1tf=”. The valid arguments for this option are:

full	Provides all available mitigations for the L1TF vulnerability. Disables SMT and enables all mitigations in the hypervisors, i.e. unconditional L1D flushing SMT control and L1D flush control via the sysfs interface is still possible after boot. Hypervisors will issue a warning when the first VM is started in a potentially insecure configuration, i.e. SMT enabled or L1D flush disabled.
full,force	Same as 'full', but disables SMT and L1D flush runtime control. Implies the 'nosmt=force' command line option. (i.e. sysfs control of SMT is disabled.)
flush	Leaves SMT enabled and enables the default hypervisor mitigation, i.e. conditional L1D flushing SMT control and L1D flush control via the sysfs interface is still possible after boot. Hypervisors will issue a warning when the first VM is started in a potentially insecure configuration, i.e. SMT enabled or L1D flush disabled.
flush,nosmt	Disables SMT and enables the default hypervisor mitigation, i.e. conditional L1D flushing. SMT control and L1D flush control via the sysfs interface is still possible after boot. Hypervisors will issue a warning when the first VM is started in a potentially insecure configuration, i.e. SMT enabled or L1D flush disabled.
flush,nowarn	Same as 'flush', but hypervisors will not warn when a VM is started in a potentially insecure configuration.
off	Disables hypervisor mitigations and doesn't emit any warnings. It also drops the swap size and available RAM limit restrictions on both hypervisor and bare metal.

The default is 'flush'. For details about L1D flushing see *1. L1D flush on VMENTER*.

7.2.9 Mitigation control for KVM - module parameter

The KVM hypervisor mitigation mechanism, flushing the L1D cache when entering a guest, can be controlled with a module parameter.

The option/parameter is "kvm-intel.vmentry_l1d_flush=". It takes the following arguments:

always	L1D cache flush on every VMENTER.
cond	Flush L1D on VMENTER only when the code between VMEXIT and VMENTER can leak host memory which is considered interesting for an attacker. This still can leak host memory which allows e.g. to determine the hosts address space layout.
never	Disables the mitigation

The parameter can be provided on the kernel command line, as a module parameter when loading the modules and at runtime modified via the sysfs file:

/sys/module/kvm_intel/parameters/vmentry_l1d_flush

The default is 'cond'. If 'l1tf=full,force' is given on the kernel command line, then 'always' is enforced and the kvm-intel.vmentry_l1d_flush module parameter is ignored and writes to the sysfs file are rejected.

7.2.10 Mitigation selection guide

1. No virtualization in use

The system is protected by the kernel unconditionally and no further action is required.

2. Virtualization with trusted guests

If the guest comes from a trusted source and the guest OS kernel is guaranteed to have the L1TF mitigations in place the system is fully protected against L1TF and no further action is required.

To avoid the overhead of the default L1D flushing on VMENTER the administrator can disable the flushing via the kernel command line and sysfs control files. See *Mitigation control on the kernel command line* and *Mitigation control for KVM - module parameter*.

3. Virtualization with untrusted guests

3.1. SMT not supported or disabled

If SMT is not supported by the processor or disabled in the BIOS or by the kernel, it's only required to enforce L1D flushing on VMENTER.

Conditional L1D flushing is the default behaviour and can be tuned. See *Mitigation control on the kernel command line* and *Mitigation control for KVM - module parameter*.

3.2. EPT not supported or disabled

If EPT is not supported by the processor or disabled in the hypervisor, the system is fully protected. SMT can stay enabled and L1D flushing on VMENTER is not required.

EPT can be disabled in the hypervisor via the 'kvm-intel.ept' parameter.

3.3. SMT and EPT supported and active

If SMT and EPT are supported and active then various degrees of mitigations can be employed:

- L1D flushing on VMENTER:

  L1D flushing on VMENTER is the minimal protection requirement, but it is only potent in combination with other mitigation methods.

  Conditional L1D flushing is the default behaviour and can be tuned. See *Mitigation control on the kernel command line* and *Mitigation control for KVM - module parameter*.

- Guest confinement:

  Confinement of guests to a single or a group of physical cores which are not running any other processes, can reduce the attack surface significantly, but interrupts, soft interrupts and kernel threads can still expose valuable data to a potential attacker. See *2. Guest VCPU confinement to dedicated physical cores*.

- Interrupt isolation:

  Isolating the guest CPUs from interrupts can reduce the attack surface further, but still allows a malicious guest to explore a limited amount of host physical memory. This can at least be used to gain knowledge about the host address space layout. The interrupts which have a fixed affinity to the CPUs which run the untrusted guests can depending on the scenario still trigger soft interrupts and schedule kernel threads which might expose valuable information. See *3. Interrupt affinity*.

The above three mitigation methods combined can provide protection to a certain degree, but the risk of the remaining attack surface has to be carefully analyzed. For full protection the following methods are available:

- Disabling SMT:

  Disabling SMT and enforcing the L1D flushing provides the maximum amount of protection. This mitigation is not depending on any of the above mitigation methods.

  SMT control and L1D flushing can be tuned by the command line parameters 'nosmt', 'l1tf', 'kvm-intel.vmentry_l1d_flush' and at run time with the matching sysfs control files. See *4. SMT control*, *Mitigation control on the kernel command line* and *Mitigation control for KVM - module parameter*.

- Disabling EPT:

  Disabling EPT provides the maximum amount of protection as well. It is not depending on any of the above mitigation methods. SMT can stay enabled and L1D flushing is not required, but the performance impact is significant.

  EPT can be disabled in the hypervisor via the 'kvm-intel.ept' parameter.

3.4. Nested virtual machines

When nested virtualization is in use, three operating systems are involved: the bare metal hypervisor, the nested hypervisor and the nested virtual machine. VMENTER operations from the nested hypervisor into the nested guest will always be processed by the bare metal hypervisor. If KVM is the bare metal hypervisor it will:

- Flush the L1D cache on every switch from the nested hypervisor to the nested virtual machine, so that the nested hypervisor's secrets are not exposed to the nested virtual machine;
- Flush the L1D cache on every switch from the nested virtual machine to the nested hypervisor; this is a complex operation, and flushing the L1D cache avoids that the bare metal hypervisor's secrets are exposed to the nested virtual machine;
- Instruct the nested hypervisor to not perform any L1D cache flush. This is an optimization to avoid double L1D flushing.

7.2.11 Default mitigations

The kernel default mitigations for vulnerable processors are:

- PTE inversion to protect against malicious user space. This is done unconditionally and cannot be controlled. The swap storage is limited to ~16TB.
- L1D conditional flushing on VMENTER when EPT is enabled for a guest.

The kernel does not by default enforce the disabling of SMT, which leaves SMT systems vulnerable when running untrusted guests with EPT enabled.

The rationale for this choice is:

- Force disabling SMT can break existing setups, especially with unattended updates.
- If regular users run untrusted guests on their machine, then L1TF is just an add on to other malware which might be embedded in an untrusted guest, e.g. spambots or attacks on the local network.

  There is no technical way to prevent a user from running untrusted code on their machines blindly.
- It's technically extremely unlikely and from today's knowledge even impossible that L1TF can be exploited via the most popular attack mechanisms like JavaScript because these mechanisms have no way to control PTEs. If this would be possible and not other mitigation would be possible, then the default might be different.
- The administrators of cloud and hosting setups have to carefully analyze the risk for their scenarios and make the appropriate mitigation choices, which might even vary across their deployed machines and also result in other changes of their overall setup. There is no way for the kernel to provide a sensible default for this kind of scenarios.

7.3 MDS - Microarchitectural Data Sampling

Microarchitectural Data Sampling is a hardware vulnerability which allows unprivileged speculative access to data which is available in various CPU internal buffers.

7.3.1 Affected processors

This vulnerability affects a wide range of Intel processors. The vulnerability is not present on:

- Processors from AMD, Centaur and other non Intel vendors
- Older processor models, where the CPU family is < 6
- Some Atoms (Bonnell, Saltwell, Goldmont, GoldmontPlus)
- Intel processors which have the ARCH_CAP_MDS_NO bit set in the IA32_ARCH_CAPABILITIES MSR.

Whether a processor is affected or not can be read out from the MDS vulnerability file in sysfs. See *MDS system information*.

Not all processors are affected by all variants of MDS, but the mitigation is identical for all of them so the kernel treats them as a single vulnerability.

7.3.2 Related CVEs

The following CVE entries are related to the MDS vulnerability:

CVE-2018-12126	MSBDS	Microarchitectural Store Buffer Data Sampling
CVE-2018-12130	MFBDS	Microarchitectural Fill Buffer Data Sampling
CVE-2018-12127	MLPDS	Microarchitectural Load Port Data Sampling
CVE-2019-11091	MDSUM	Microarchitectural Data Sampling Uncacheable Memory

7.3.3 Problem

When performing store, load, L1 refill operations, processors write data into temporary microarchitectural structures (buffers). The data in the buffer can be forwarded to load operations as an optimization.

Under certain conditions, usually a fault/assist caused by a load operation, data unrelated to the load memory address can be speculatively forwarded from the buffers. Because the load operation causes a fault or assist and its result will be discarded, the forwarded data will not cause incorrect program execution or state changes. But a malicious operation may be able to forward this speculative data to a disclosure gadget which allows in turn to infer the value via a cache side channel attack.

Because the buffers are potentially shared between Hyper-Threads cross Hyper-Thread attacks are possible.

Deeper technical information is available in the MDS specific x86 architecture section: Documentation/arch/x86/mds.rst.

7.3.4 Attack scenarios

Attacks against the MDS vulnerabilities can be mounted from malicious non- privileged user space applications running on hosts or guest. Malicious guest OSes can obviously mount attacks as well.

Contrary to other speculation based vulnerabilities the MDS vulnerability does not allow the attacker to control the memory target address. As a consequence the attacks are purely sampling based, but as demonstrated with the TLBleed attack samples can be postprocessed successfully.

Web-Browsers

> It's unclear whether attacks through Web-Browsers are possible at all. The exploitation through Java-Script is considered very unlikely, but other widely used web technologies like Webassembly could possibly be abused.

7.3.5 MDS system information

The Linux kernel provides a sysfs interface to enumerate the current MDS status of the system: whether the system is vulnerable, and which mitigations are active. The relevant sysfs file is:

/sys/devices/system/cpu/vulnerabilities/mds

The possible values in this file are:

'Not affected'	The processor is not vulnerable
'Vulnerable'	The processor is vulnerable, but no mitigation enabled
'Vulnerable: Clear CPU buffers attempted, no microcode'	The processor is vulnerable but microcode is not updated. The mitigation is enabled on a best effort basis. If the processor is vulnerable but the availability of the microcode based mitigation mechanism is not advertised via CPUID, the kernel selects a best effort mitigation mode. This mode invokes the mitigation instructions without a guarantee that they clear the CPU buffers. This is done to address virtualization scenarios where the host has the microcode update applied, but the hypervisor is not yet updated to expose the CPUID to the guest. If the host has updated microcode the protection takes effect; otherwise a few CPU cycles are wasted pointlessly.
'Mitigation: Clear CPU buffers'	The processor is vulnerable and the CPU buffer clearing mitigation is enabled.

If the processor is vulnerable then the following information is appended to the above information:

'SMT vulnerable'	SMT is enabled
'SMT mitigated'	SMT is enabled and mitigated
'SMT disabled'	SMT is disabled
'SMT Host state unknown'	Kernel runs in a VM, Host SMT state unknown

7.3.6 Mitigation mechanism

The kernel detects the affected CPUs and the presence of the microcode which is required.

If a CPU is affected and the microcode is available, then the kernel enables the mitigation by default. The mitigation can be controlled at boot time via a kernel command line option. See *Mitigation control on the kernel command line*.

CPU buffer clearing

The mitigation for MDS clears the affected CPU buffers on return to user space and when entering a guest.

If SMT is enabled it also clears the buffers on idle entry when the CPU is only affected by MSBDS and not any other MDS variant, because the other variants cannot be protected against cross Hyper-Thread attacks.

For CPUs which are only affected by MSBDS the user space, guest and idle transition mitigations are sufficient and SMT is not affected.

Virtualization mitigation

The protection for host to guest transition depends on the L1TF vulnerability of the CPU:

- CPU is affected by L1TF:

  If the L1D flush mitigation is enabled and up to date microcode is available, the L1D flush mitigation is automatically protecting the guest transition.

  If the L1D flush mitigation is disabled then the MDS mitigation is invoked explicit when the host MDS mitigation is enabled.

  For details on L1TF and virtualization see: *Documentation/admin-guide/hw-vuln//l1tf.rst*.

- CPU is not affected by L1TF:

  CPU buffers are flushed before entering the guest when the host MDS mitigation is enabled.

The resulting MDS protection matrix for the host to guest transition:

L1TF	MDS	VMX-L1FLUSH	Host MDS	MDS-State
Don't care	No	Don't care	N/A	Not affected
Yes	Yes	Disabled	Off	Vulnerable
Yes	Yes	Disabled	Full	Mitigated
Yes	Yes	Enabled	Don't care	Mitigated
No	Yes	N/A	Off	Vulnerable
No	Yes	N/A	Full	Mitigated

This only covers the host to guest transition, i.e. prevents leakage from host to guest, but does not protect the guest internally. Guests need to have their own protections.

XEON PHI specific considerations

The XEON PHI processor family is affected by MSBDS which can be exploited cross Hyper-Threads when entering idle states. Some XEON PHI variants allow to use MWAIT in user space (Ring 3) which opens an potential attack vector for malicious user space. The exposure can be disabled on the kernel command line with the 'ring3mwait=disable' command line option.

XEON PHI is not affected by the other MDS variants and MSBDS is mitigated before the CPU enters a idle state. As XEON PHI is not affected by L1TF either disabling SMT is not required for full protection.

SMT control

All MDS variants except MSBDS can be attacked cross Hyper-Threads. That means on CPUs which are affected by MFBDS or MLPDS it is necessary to disable SMT for full protection. These are most of the affected CPUs; the exception is XEON PHI, see *XEON PHI specific considerations*.

Disabling SMT can have a significant performance impact, but the impact depends on the type of workloads.

See the relevant chapter in the L1TF mitigation documentation for details: *Documentation/admin-guide/hw-vuln/l1tf.rst*.

7.3.7 Mitigation control on the kernel command line

The kernel command line allows to control the MDS mitigations at boot time with the option ”mds=”. The valid arguments for this option are:

full	If the CPU is vulnerable, enable all available mitigations for the MDS vulnerability, CPU buffer clearing on exit to userspace and when entering a VM. Idle transitions are protected as well if SMT is enabled. It does not automatically disable SMT.
full,nosmt	The same as mds=full, with SMT disabled on vulnerable CPUs. This is the complete mitigation.
off	Disables MDS mitigations completely.

Not specifying this option is equivalent to "mds=full". For processors that are affected by both TAA (TSX Asynchronous Abort) and MDS, specifying just "mds=off" without an accompanying "tsx_async_abort=off" will have no effect as the same mitigation is used for both vulnerabilities.

7.3.8 Mitigation selection guide

1. Trusted userspace

If all userspace applications are from a trusted source and do not execute untrusted code which is supplied externally, then the mitigation can be disabled.

2. Virtualization with trusted guests

The same considerations as above versus trusted user space apply.

3. Virtualization with untrusted guests

The protection depends on the state of the L1TF mitigations. See *Virtualization mitigation*.

If the MDS mitigation is enabled and SMT is disabled, guest to host and guest to guest attacks are prevented.

7.3.9 Default mitigations

The kernel default mitigations for vulnerable processors are:

- Enable CPU buffer clearing

The kernel does not by default enforce the disabling of SMT, which leaves SMT systems vulnerable when running untrusted code. The same rationale as for L1TF applies. See *Documentation/admin-guide/hw-vuln//l1tf.rst*.

7.4 TAA - TSX Asynchronous Abort

TAA is a hardware vulnerability that allows unprivileged speculative access to data which is available in various CPU internal buffers by using asynchronous aborts within an Intel TSX transactional region.

7.4.1 Affected processors

This vulnerability only affects Intel processors that support Intel Transactional Synchronization Extensions (TSX) when the TAA_NO bit (bit 8) is 0 in the IA32_ARCH_CAPABILITIES MSR. On processors where the MDS_NO bit (bit 5) is 0 in the IA32_ARCH_CAPABILITIES MSR, the existing MDS mitigations also mitigate against TAA.

Whether a processor is affected or not can be read out from the TAA vulnerability file in sysfs. See *TAA system information*.

7.4.2 Related CVEs

The following CVE entry is related to this TAA issue:

CVE-2019-11135	TAA	TSX Asynchronous Abort (TAA) condition on some microprocessors utilizing speculative execution may allow an authenticated user to potentially enable information disclosure via a side channel with local access.

7.4.3 Problem

When performing store, load or L1 refill operations, processors write data into temporary microarchitectural structures (buffers). The data in those buffers can be forwarded to load operations as an optimization.

Intel TSX is an extension to the x86 instruction set architecture that adds hardware transactional memory support to improve performance of multi-threaded software. TSX lets the processor expose and exploit concurrency hidden in an application due to dynamically avoiding unnecessary synchronization.

TSX supports atomic memory transactions that are either committed (success) or aborted. During an abort, operations that happened within the transactional region are rolled back. An asynchronous abort takes place, among other options, when a different thread accesses a cache line that is also used within the transactional region when that access might lead to a data race.

Immediately after an uncompleted asynchronous abort, certain speculatively executed loads may read data from those internal buffers and pass it to dependent operations. This can be then used to infer the value via a cache side channel attack.

Because the buffers are potentially shared between Hyper-Threads cross Hyper-Thread attacks are possible.

The victim of a malicious actor does not need to make use of TSX. Only the attacker needs to begin a TSX transaction and raise an asynchronous abort which in turn potentially leaks data stored in the buffers.

More detailed technical information is available in the TAA specific x86 architecture section: Documentation/arch/x86/tsx_async_abort.rst.

7.4.4 Attack scenarios

Attacks against the TAA vulnerability can be implemented from unprivileged applications running on hosts or guests.

As for MDS, the attacker has no control over the memory addresses that can be leaked. Only the victim is responsible for bringing data to the CPU. As a result, the malicious actor has to sample as much data as possible and then postprocess it to try to infer any useful information from it.

A potential attacker only has read access to the data. Also, there is no direct privilege escalation by using this technique.

7.4.5 TAA system information

The Linux kernel provides a sysfs interface to enumerate the current TAA status of mitigated systems. The relevant sysfs file is:

/sys/devices/system/cpu/vulnerabilities/tsx_async_abort

The possible values in this file are:

'Vulnerable'	The CPU is affected by this vulnerability and the microcode and kernel mitigation are not applied.
'Vulnerable: Clear CPU buffers attempted, no microcode'	The processor is vulnerable but microcode is not updated. The mitigation is enabled on a best effort basis. If the processor is vulnerable but the availability of the microcode based mitigation mechanism is not advertised via CPUID, the kernel selects a best effort mitigation mode. This mode invokes the mitigation instructions without a guarantee that they clear the CPU buffers. This is done to address virtualization scenarios where the host has the microcode update applied, but the hypervisor is not yet updated to expose the CPUID to the guest. If the host has updated microcode the protection takes effect; otherwise a few CPU cycles are wasted pointlessly.
'Mitigation: Clear CPU buffers'	The microcode has been updated to clear the buffers. TSX is still enabled.
'Mitigation: TSX disabled'	TSX is disabled.
'Not affected'	The CPU is not affected by this issue.

7.4.6 Mitigation mechanism

The kernel detects the affected CPUs and the presence of the microcode which is required. If a CPU is affected and the microcode is available, then the kernel enables the mitigation by default.

The mitigation can be controlled at boot time via a kernel command line option. See *Mitigation control on the kernel command line*.

Virtualization mitigation

Affected systems where the host has TAA microcode and TAA is mitigated by having disabled TSX previously, are not vulnerable regardless of the status of the VMs.

In all other cases, if the host either does not have the TAA microcode or the kernel is not mitigated, the system might be vulnerable.

7.4.7 Mitigation control on the kernel command line

The kernel command line allows to control the TAA mitigations at boot time with the option "tsx_async_abort=". The valid arguments for this option are:

off	This option disables the TAA mitigation on affected platforms. If the system has TSX enabled (see next parameter) and the CPU is affected, the system is vulnerable.
full	TAA mitigation is enabled. If TSX is enabled, on an affected system it will clear CPU buffers on ring transitions. On systems which are MDS-affected and deploy MDS mitigation, TAA is also mitigated. Specifying this option on those systems will have no effect.
full,nosmt	The same as tsx_async_abort=full, with SMT disabled on vulnerable CPUs that have TSX enabled. This is the complete mitigation. When TSX is disabled, SMT is not disabled because CPU is not vulnerable to cross-thread TAA attacks.

Not specifying this option is equivalent to "tsx_async_abort=full". For processors that are affected by both TAA and MDS, specifying just "tsx_async_abort=off" without an accompanying "mds=off" will have no effect as the same mitigation is used for both vulnerabilities.

The kernel command line also allows to control the TSX feature using the parameter "tsx=" on CPUs which support TSX control. MSR_IA32_TSX_CTRL is used to control the TSX feature and the enumeration of the TSX feature bits (RTM and HLE) in CPUID.

The valid options are:

off	Disables TSX on the system. Note that this option takes effect only on newer CPUs which are not vulnerable to MDS, i.e., have MSR_IA32_ARCH_CAPABILITIES.MDS_NO=1 and which get the new IA32_TSX_CTRL MSR through a microcode update. This new MSR allows for the reliable deactivation of the TSX functionality.
on	Enables TSX. Although there are mitigations for all known security vulnerabilities, TSX has been known to be an accelerator for several previous speculation-related CVEs, and so there may be unknown security risks associated with leaving it enabled.
auto	Disables TSX if X86_BUG_TAA is present, otherwise enables TSX on the system.

Not specifying this option is equivalent to "tsx=off".

The following combinations of the "tsx_async_abort" and "tsx" are possible. For affected platforms tsx=auto is equivalent to tsx=off and the result will be:

tsx=on	tsx_async_abort=full	The system will use VERW to clear CPU buffers. Cross-thread attacks are still possible on SMT machines.
tsx=on	tsx_async_abort=full,nosmt	As above, cross-thread attacks on SMT mitigated.
tsx=on	tsx_async_abort=off	The system is vulnerable.
tsx=off	tsx_async_abort=full	TSX might be disabled if microcode provides a TSX control MSR. If so, system is not vulnerable.
tsx=off	tsx_async_abort=full,nosmt	Ditto
tsx=off	tsx_async_abort=off	ditto

For unaffected platforms "tsx=on" and "tsx_async_abort=full" does not clear CPU buffers. For platforms without TSX control (MSR_IA32_ARCH_CAPABILITIES.MDS_NO=0) "tsx" command line argument has no effect.

For the affected platforms below table indicates the mitigation status for the combinations of CPUID bit MD_CLEAR and IA32_ARCH_CAPABILITIES MSR bits MDS_NO and TSX_CTRL_MSR.

MDS_NO	MD_CLEAR	TSX_CTRL_MSR	Status
0	0	0	Vulnerable (needs microcode)
0	1	0	MDS and TAA mitigated via VERW
1	1	0	MDS fixed, TAA vulnerable if TSX enabled because MD_CLEAR has no meaning and VERW is not guaranteed to clear buffers
1	X	1	MDS fixed, TAA can be mitigated by VERW or TSX_CTRL_MSR

7.4.8 Mitigation selection guide

1. Trusted userspace and guests

If all user space applications are from a trusted source and do not execute untrusted code which is supplied externally, then the mitigation can be disabled. The same applies to virtualized environments with trusted guests.

2. Untrusted userspace and guests

If there are untrusted applications or guests on the system, enabling TSX might allow a malicious actor to leak data from the host or from other processes running on the same physical core.

If the microcode is available and the TSX is disabled on the host, attacks are prevented in a virtualized environment as well, even if the VMs do not explicitly enable the mitigation.

7.4.9 Default mitigations

The kernel's default action for vulnerable processors is:

- Deploy TSX disable mitigation (tsx_async_abort=full tsx=off).

7.5 iTLB multihit

iTLB multihit is an erratum where some processors may incur a machine check error, possibly resulting in an unrecoverable CPU lockup, when an instruction fetch hits multiple entries in the instruction TLB. This can occur when the page size is changed along with either the physical address or cache type. A malicious guest running on a virtualized system can exploit this erratum to perform a denial of service attack.

7.5.1 Affected processors

Variations of this erratum are present on most Intel Core and Xeon processor models. The erratum is not present on:

- non-Intel processors
- Some Atoms (Airmont, Bonnell, Goldmont, GoldmontPlus, Saltwell, Silvermont)
- Intel processors that have the PSCHANGE_MC_NO bit set in the IA32_ARCH_CAPABILITIES MSR.

7.5.2 Related CVEs

The following CVE entry is related to this issue:

CVE-2018-12207	Machine Check Error Avoidance on Page Size Change

7.5.3 Problem

Privileged software, including OS and virtual machine managers (VMM), are in charge of memory management. A key component in memory management is the control of the page tables. Modern processors use virtual memory, a technique that creates the illusion of a very large memory for processors. This virtual space is split into pages of a given size. Page tables translate virtual addresses to physical addresses.

To reduce latency when performing a virtual to physical address translation, processors include a structure, called TLB, that caches recent translations. There are separate TLBs for instruction (iTLB) and data (dTLB).

Under this errata, instructions are fetched from a linear address translated using a 4 KB translation cached in the iTLB. Privileged software modifies the paging structure so that the same linear address using large page size (2 MB, 4 MB, 1 GB) with a different physical address or memory type. After the page structure modification but before the software invalidates any iTLB entries for the linear address, a code fetch that happens on the same linear address may cause a machine-check error which can result in a system hang or shutdown.

7.5.4 Attack scenarios

Attacks against the iTLB multihit erratum can be mounted from malicious guests in a virtualized system.

7.5.5 iTLB multihit system information

The Linux kernel provides a sysfs interface to enumerate the current iTLB multihit status of the system:whether the system is vulnerable and which mitigations are active. The relevant sysfs file is:

/sys/devices/system/cpu/vulnerabilities/itlb_multihit

The possible values in this file are:

Not affected	The processor is not vulnerable.
KVM: Mitigation: Split huge pages	Software changes mitigate this issue.
KVM: Mitigation: VMX unsupported	KVM is not vulnerable because Virtual Machine Extensions (VMX) is not supported.
KVM: Mitigation: VMX disabled	KVM is not vulnerable because Virtual Machine Extensions (VMX) is disabled.
KVM: Vulnerable	The processor is vulnerable, but no mitigation enabled

7.5.6 Enumeration of the erratum

A new bit has been allocated in the IA32_ARCH_CAPABILITIES (PSCHANGE_MC_NO) msr and will be set on CPU's which are mitigated against this issue.

IA32_ARCH_CAPABILITIES MSR	Not present	Possibly vulnerable,check model
IA32_ARCH_CAPABILITIES[PSCHANGE_MC_NO]	0	Likely vulnerable,check model
IA32_ARCH_CAPABILITIES[PSCHANGE_MC_NO]	1	Not vulnerable

7.5.7 Mitigation mechanism

This erratum can be mitigated by restricting the use of large page sizes to non-executable pages. This forces all iTLB entries to be 4K, and removes the possibility of multiple hits.

In order to mitigate the vulnerability, KVM initially marks all huge pages as non-executable. If the guest attempts to execute in one of those pages, the page is broken down into 4K pages, which are then marked executable.

If EPT is disabled or not available on the host, KVM is in control of TLB flushes and the problematic situation cannot happen. However, the shadow EPT paging mechanism used by nested virtualization is vulnerable, because the nested guest can trigger multiple iTLB hits by modifying its own (non-nested) page tables. For simplicity, KVM will make large pages non-executable in all shadow paging modes.

7.5.8 Mitigation control on the kernel command line and KVM - module parameter

The KVM hypervisor mitigation mechanism for marking huge pages as non-executable can be controlled with a module parameter ”nx_huge_pages=”. The kernel command line allows to control the iTLB multihit mitigations at boot time with the option ”kvm.nx_huge_pages=”.

The valid arguments for these options are:

force	Mitigation is enabled. In this case, the mitigation implements non-executable huge pages in Linux kernel KVM module. All huge pages in the EPT are marked as non-executable. If a guest attempts to execute in one of those pages, the page is broken down into 4K pages, which are then marked executable.
off	Mitigation is disabled.
auto	Enable mitigation only if the platform is affected and the kernel was not booted with the ”mitigations=off” command line parameter. This is the default option.

7.5.9 Mitigation selection guide

1. No virtualization in use

The system is protected by the kernel unconditionally and no further action is required.

2. Virtualization with trusted guests

If the guest comes from a trusted source, you may assume that the guest will not attempt to maliciously exploit these errata and no further action is required.

3. Virtualization with untrusted guests

If the guest comes from an untrusted source, the guest host kernel will need to apply iTLB multihit mitigation via the kernel command line or kvm module parameter.

7.6 SRBDS - Special Register Buffer Data Sampling

SRBDS is a hardware vulnerability that allows MDS *MDS - Microarchitectural Data Sampling* techniques to infer values returned from special register accesses. Special register accesses are accesses to off core registers. According to Intel's evaluation, the special register reads that have a security expectation of privacy are RDRAND, RDSEED and SGX EGETKEY.

When RDRAND, RDSEED and EGETKEY instructions are used, the data is moved to the core through the special register mechanism that is susceptible to MDS attacks.

7.6.1 Affected processors

Core models (desktop, mobile, Xeon-E3) that implement RDRAND and/or RDSEED may be affected.

A processor is affected by SRBDS if its Family_Model and stepping is in the following list, with the exception of the listed processors exporting MDS_NO while Intel TSX is available yet not enabled. The latter class of processors are only affected when Intel TSX is enabled by software using TSX_CTRL_MSR otherwise they are not affected.

common name	Family_Model	Stepping
IvyBridge	06_3AH	All
Haswell	06_3CH	All
Haswell_L	06_45H	All
Haswell_G	06_46H	All
Broadwell_G	06_47H	All
Broadwell	06_3DH	All
Skylake_L	06_4EH	All
Skylake	06_5EH	All
Kabylake_L	06_8EH	<= 0xC
Kabylake	06_9EH	<= 0xD

7.6.2 Related CVEs

The following CVE entry is related to this SRBDS issue:

CVE-2020-0543	SRBDS	Special Register Buffer Data Sampling

7.6.3 Attack scenarios

An unprivileged user can extract values returned from RDRAND and RDSEED executed on another core or sibling thread using MDS techniques.

7.6.4 Mitigation mechanism

Intel will release microcode updates that modify the RDRAND, RDSEED, and EGETKEY instructions to overwrite secret special register data in the shared staging buffer before the secret data can be accessed by another logical processor.

During execution of the RDRAND, RDSEED, or EGETKEY instructions, off-core accesses from other logical processors will be delayed until the special register read is complete and the secret data in the shared staging buffer is overwritten.

This has three effects on performance:

1. RDRAND, RDSEED, or EGETKEY instructions have higher latency.
2. Executing RDRAND at the same time on multiple logical processors will be serialized, resulting in an overall reduction in the maximum RDRAND bandwidth.
3. Executing RDRAND, RDSEED or EGETKEY will delay memory accesses from other logical processors that miss their core caches, with an impact similar to legacy locked cache-line-split accesses.

The microcode updates provide an opt-out mechanism (RNGDS_MITG_DIS) to disable the mitigation for RDRAND and RDSEED instructions executed outside of Intel Software Guard Extensions (Intel SGX) enclaves. On logical processors that disable the mitigation using this opt-out mechanism, RDRAND and RDSEED do not take longer to execute and do not impact performance of sibling logical processors memory accesses. The opt-out mechanism does not affect Intel SGX enclaves (including execution of RDRAND or RDSEED inside an enclave, as well as EGETKEY execution).

7.6.5 IA32_MCU_OPT_CTRL MSR Definition

Along with the mitigation for this issue, Intel added a new thread-scope IA32_MCU_OPT_CTRL MSR, (address 0x123). The presence of this MSR and RNGDS_MITG_DIS (bit 0) is enumerated by CPUID.(EAX=07H,ECX=0).EDX[SRBDS_CTRL = 9]==1. This MSR is introduced through the microcode update.

Setting IA32_MCU_OPT_CTRL[0] (RNGDS_MITG_DIS) to 1 for a logical processor disables the mitigation for RDRAND and RDSEED executed outside of an Intel SGX enclave on that logical processor. Opting out of the mitigation for a particular logical processor does not affect the RDRAND and RDSEED mitigations for other logical processors.

Note that inside of an Intel SGX enclave, the mitigation is applied regardless of the value of RNGDS_MITG_DS.

7.6.6 Mitigation control on the kernel command line

The kernel command line allows control over the SRBDS mitigation at boot time with the option "srbds=". The option for this is:

off	This option disables SRBDS mitigation for RDRAND and RDSEED on affected platforms.

7.6.7 SRBDS System Information

The Linux kernel provides vulnerability status information through sysfs. For SRBDS this can be accessed by the following sysfs file: /sys/devices/system/cpu/vulnerabilities/srbds

The possible values contained in this file are:

Not affected	Processor not vulnerable
Vulnerable	Processor vulnerable and mitigation disabled
Vulnerable: No microcode	Processor vulnerable and microcode is missing mitigation
Mitigation: Microcode	Processor is vulnerable and mitigation is in effect.
Mitigation: TSX disabled	Processor is only vulnerable when TSX is enabled while this system was booted with TSX disabled.
Unknown: Dependent on	
hypervisor status	Running on virtual guest processor that is affected but with no way to know if host processor is mitigated or vulnerable.

7.6.8 SRBDS Default mitigation

This new microcode serializes processor access during execution of RDRAND, RDSEED ensures that the shared buffer is overwritten before it is released for reuse. Use the "srbds=off" kernel command line to disable the mitigation for RDRAND and RDSEED.

7.7 Core Scheduling

Core scheduling support allows userspace to define groups of tasks that can share a core. These groups can be specified either for security usecases (one group of tasks don't trust another), or for performance usecases (some workloads may benefit from running on the same core as they don't need the same hardware resources of the shared core, or may prefer different cores if they do share hardware resource needs). This document only describes the security usecase.

7.7.1 Security usecase

A cross-HT attack involves the attacker and victim running on different Hyper Threads of the same core. MDS and L1TF are examples of such attacks. The only full mitigation of cross-HT attacks is to disable Hyper Threading (HT). Core scheduling is a scheduler feature that can mitigate some (not all) cross-HT attacks. It allows HT to be turned on safely by ensuring that only tasks in a user-designated trusted group can share a core. This increase in core sharing can also improve performance, however it is not guaranteed that performance will always improve, though that is seen to be the case with a number of real world workloads. In theory, core scheduling aims to perform at least as good as when Hyper Threading is disabled. In practice, this is mostly the case though not always: as synchronizing scheduling decisions across 2 or more CPUs in a core involves additional overhead - especially when the system is lightly loaded. When `total_threads <= N_CPUS/2`, the extra overhead may cause core scheduling to perform more poorly compared to SMT-disabled, where N_CPUS is the total number of CPUs. Please measure the performance of your workloads always.

7.7.2 Usage

Core scheduling support is enabled via the `CONFIG_SCHED_CORE` config option. Using this feature, userspace defines groups of tasks that can be co-scheduled on the same core. The core scheduler uses this information to make sure that tasks that are not in the same group never run simultaneously on a core, while doing its best to satisfy the system's scheduling requirements.

Core scheduling can be enabled via the `PR_SCHED_CORE` prctl interface. This interface provides support for the creation of core scheduling groups, as well as admission and removal of tasks from created groups:

```
#include <sys/prctl.h>

int prctl(int option, unsigned long arg2, unsigned long arg3,
        unsigned long arg4, unsigned long arg5);
```

option:
`PR_SCHED_CORE`

arg2:
Command for operation, must be one off:

- `PR_SCHED_CORE_GET` -- get core_sched cookie of `pid`.
- `PR_SCHED_CORE_CREATE` -- create a new unique cookie for `pid`.
- `PR_SCHED_CORE_SHARE_TO` -- push core_sched cookie to `pid`.
- `PR_SCHED_CORE_SHARE_FROM` -- pull core_sched cookie from `pid`.

arg3:
`pid` of the task for which the operation applies.

arg4:
`pid_type` for which the operation applies. It is one of `PR_SCHED_CORE_SCOPE_`-prefixed macro constants. For example, if arg4 is `PR_SCHED_CORE_SCOPE_THREAD_GROUP`, then the operation of this command will be performed for all tasks in the task group of `pid`.

arg5:
userspace pointer to an unsigned long for storing the cookie returned by `PR_SCHED_CORE_GET` command. Should be 0 for all other commands.

In order for a process to push a cookie to, or pull a cookie from a process, it is required to have the ptrace access mode: *PTRACE_MODE_READ_REALCREDS* to the process.

Building hierarchies of tasks

The simplest way to build hierarchies of threads/processes which share a cookie and thus a core is to rely on the fact that the core-sched cookie is inherited across forks/clones and execs, thus setting a cookie for the 'initial' script/executable/daemon will place every spawned child in the same core-sched group.

Cookie Transferral

Transferring a cookie between the current and other tasks is possible using PR_SCHED_CORE_SHARE_FROM and PR_SCHED_CORE_SHARE_TO to inherit a cookie from a specified task or a share a cookie with a task. In combination this allows a simple helper program to pull a cookie from a task in an existing core scheduling group and share it with already running tasks.

7.7.3 Design/Implementation

Each task that is tagged is assigned a cookie internally in the kernel. As mentioned in *Usage*, tasks with the same cookie value are assumed to trust each other and share a core.

The basic idea is that, every schedule event tries to select tasks for all the siblings of a core such that all the selected tasks running on a core are trusted (same cookie) at any point in time. Kernel threads are assumed trusted. The idle task is considered special, as it trusts everything and everything trusts it.

During a schedule() event on any sibling of a core, the highest priority task on the sibling's core is picked and assigned to the sibling calling schedule(), if the sibling has the task enqueued. For rest of the siblings in the core, highest priority task with the same cookie is selected if there is one runnable in their individual run queues. If a task with same cookie is not available, the idle task is selected. Idle task is globally trusted.

Once a task has been selected for all the siblings in the core, an IPI is sent to siblings for whom a new task was selected. Siblings on receiving the IPI will switch to the new task immediately. If an idle task is selected for a sibling, then the sibling is considered to be in a *forced idle* state. I.e., it may have tasks on its on runqueue to run, however it will still have to run idle. More on this in the next section.

Forced-idling of hyperthreads

The scheduler tries its best to find tasks that trust each other such that all tasks selected to be scheduled are of the highest priority in a core. However, it is possible that some runqueues had tasks that were incompatible with the highest priority ones in the core. Favoring security over fairness, one or more siblings could be forced to select a lower priority task if the highest priority task is not trusted with respect to the core wide highest priority task. If a sibling does not have a trusted task to run, it will be forced idle by the scheduler (idle thread is scheduled to run).

When the highest priority task is selected to run, a reschedule-IPI is sent to the sibling to force it into idle. This results in 4 cases which need to be considered depending on whether a VM or a regular usermode process was running on either HT:

```
       HT1 (attack)            HT2 (victim)
A      idle -> user space      user space -> idle
B      idle -> user space      guest -> idle
C      idle -> guest           user space -> idle
D      idle -> guest           guest -> idle
```

Note that for better performance, we do not wait for the destination CPU (victim) to enter idle mode. This is because the sending of the IPI would bring the destination CPU immediately into kernel mode from user space, or VMEXIT in the case of guests. At best, this would only leak some scheduler metadata which may not be worth protecting. It is also possible that the IPI is received too late on some architectures, but this has not been observed in the case of x86.

Trust model

Core scheduling maintains trust relationships amongst groups of tasks by assigning them a tag that is the same cookie value. When a system with core scheduling boots, all tasks are considered to trust each other. This is because the core scheduler does not have information about trust relationships until userspace uses the above mentioned interfaces, to communicate them. In other words, all tasks have a default cookie value of 0. and are considered system-wide trusted. The forced-idling of siblings running cookie-0 tasks is also avoided.

Once userspace uses the above mentioned interfaces to group sets of tasks, tasks within such groups are considered to trust each other, but do not trust those outside. Tasks outside the group also don't trust tasks within.

7.7.4 Limitations of core-scheduling

Core scheduling tries to guarantee that only trusted tasks run concurrently on a core. But there could be small window of time during which untrusted tasks run concurrently or kernel could be running concurrently with a task not trusted by kernel.

IPI processing delays

Core scheduling selects only trusted tasks to run together. IPI is used to notify the siblings to switch to the new task. But there could be hardware delays in receiving of the IPI on some arch (on x86, this has not been observed). This may cause an attacker task to start running on a CPU before its siblings receive the IPI. Even though cache is flushed on entry to user mode, victim tasks on siblings may populate data in the cache and micro architectural buffers after the attacker starts to run and this is a possibility for data leak.

7.7.5 Open cross-HT issues that core scheduling does not solve

1. For MDS

Core scheduling cannot protect against MDS attacks between the siblings running in user mode and the others running in kernel mode. Even though all siblings run tasks which trust each other, when the kernel is executing code on behalf of a task, it cannot trust the code running in the sibling. Such attacks are possible for any combination of sibling CPU modes (host or guest mode).

2. For L1TF

Core scheduling cannot protect against an L1TF guest attacker exploiting a guest or host victim. This is because the guest attacker can craft invalid PTEs which are not inverted due to a vulnerable guest kernel. The only solution is to disable EPT (Extended Page Tables).

For both MDS and L1TF, if the guest vCPU is configured to not trust each other (by tagging separately), then the guest to guest attacks would go away. Or it could be a system admin policy which considers guest to guest attacks as a guest problem.

Another approach to resolve these would be to make every untrusted task on the system to not trust every other untrusted task. While this could reduce parallelism of the untrusted tasks, it would still solve the above issues while allowing system processes (trusted tasks) to share a core.

3. Protecting the kernel (IRQ, syscall, VMEXIT)

Unfortunately, core scheduling does not protect kernel contexts running on sibling hyperthreads from one another. Prototypes of mitigations have been posted to LKML to solve this, but it is debatable whether such windows are practically exploitable, and whether the performance overhead of the prototypes are worth it (not to mention, the added code complexity).

7.7.6 Other Use cases

The main use case for Core scheduling is mitigating the cross-HT vulnerabilities with SMT enabled. There are other use cases where this feature could be used:

- Isolating tasks that needs a whole core: Examples include realtime tasks, tasks that uses SIMD instructions etc.
- Gang scheduling: Requirements for a group of tasks that needs to be scheduled together could also be realized using core scheduling. One example is vCPUs of a VM.

7.8 L1D Flushing

With an increasing number of vulnerabilities being reported around data leaks from the Level 1 Data cache (L1D) the kernel provides an opt-in mechanism to flush the L1D cache on context switch.

This mechanism can be used to address e.g. CVE-2020-0550. For applications the mechanism keeps them safe from vulnerabilities, related to leaks (snooping of) from the L1D cache.

7.8.1 Related CVEs

The following CVEs can be addressed by this mechanism

CVE-2020-0550	Improper Data Forwarding	OS related aspects

7.8.2 Usage Guidelines

Please see document: Documentation/userspace-api/spec_ctrl.rst for details.

NOTE: The feature is disabled by default, applications need to specifically opt into the feature to enable it.

7.8.3 Mitigation

When PR_SET_L1D_FLUSH is enabled for a task a flush of the L1D cache is performed when the task is scheduled out and the incoming task belongs to a different process and therefore to a different address space.

If the underlying CPU supports L1D flushing in hardware, the hardware mechanism is used, software fallback for the mitigation, is not supported.

7.8.4 Mitigation control on the kernel command line

The kernel command line allows to control the L1D flush mitigations at boot time with the option "l1d_flush=". The valid arguments for this option are:

on	Enables the prctl interface, applications trying to use the prctl() will fail with an error if l1d_flush is not enabled

By default the mechanism is disabled.

7.8.5 Limitations

The mechanism does not mitigate L1D data leaks between tasks belonging to different processes which are concurrently executing on sibling threads of a physical CPU core when SMT is enabled on the system.

This can be addressed by controlled placement of processes on physical CPU cores or by disabling SMT. See the relevant chapter in the L1TF mitigation document: *Documentation/admin-guide/hw-vuln/l1tf.rst*.

NOTE : The opt-in of a task for L1D flushing works only when the task's affinity is limited to cores running in non-SMT mode. If a task which requested L1D flushing is scheduled on a SMT-enabled core the kernel sends a SIGBUS to the task.

7.9 Processor MMIO Stale Data Vulnerabilities

Processor MMIO Stale Data Vulnerabilities are a class of memory-mapped I/O (MMIO) vulnerabilities that can expose data. The sequences of operations for exposing data range from simple to very complex. Because most of the vulnerabilities require the attacker to have access to MMIO, many environments are not affected. System environments using virtualization where MMIO access is provided to untrusted guests may need mitigation. These vulnerabilities are not transient execution attacks. However, these vulnerabilities may propagate stale data into core fill buffers where the data can subsequently be inferred by an unmitigated transient execution attack. Mitigation for these vulnerabilities includes a combination of microcode update and software changes, depending on the platform and usage model. Some of these mitigations are similar to those used to mitigate Microarchitectural Data Sampling (MDS) or those used to mitigate Special Register Buffer Data Sampling (SRBDS).

7.9.1 Data Propagators

Propagators are operations that result in stale data being copied or moved from one microarchitectural buffer or register to another. Processor MMIO Stale Data Vulnerabilities are operations that may result in stale data being directly read into an architectural, software-visible state or sampled from a buffer or register.

Fill Buffer Stale Data Propagator (FBSDP)

Stale data may propagate from fill buffers (FB) into the non-coherent portion of the uncore on some non-coherent writes. Fill buffer propagation by itself does not make stale data architecturally visible. Stale data must be propagated to a location where it is subject to reading or sampling.

Sideband Stale Data Propagator (SSDP)

The sideband stale data propagator (SSDP) is limited to the client (including Intel Xeon server E3) uncore implementation. The sideband response buffer is shared by all client cores. For non-coherent reads that go to sideband destinations, the uncore logic returns 64 bytes of data to the core, including both requested data and unrequested stale data, from a transaction buffer and the sideband response buffer. As a result, stale data from the sideband response and transaction buffers may now reside in a core fill buffer.

Primary Stale Data Propagator (PSDP)

The primary stale data propagator (PSDP) is limited to the client (including Intel Xeon server E3) uncore implementation. Similar to the sideband response buffer, the primary response buffer is shared by all client cores. For some processors, MMIO primary reads will return 64 bytes of data to the core fill buffer including both requested data and unrequested stale data. This is similar to the sideband stale data propagator.

7.9.2 Vulnerabilities

Device Register Partial Write (DRPW) (CVE-2022-21166)

Some endpoint MMIO registers incorrectly handle writes that are smaller than the register size. Instead of aborting the write or only copying the correct subset of bytes (for example, 2 bytes for a 2-byte write), more bytes than specified by the write transaction may be written to the register. On processors affected by FBSDP, this may expose stale data from the fill buffers of the core that created the write transaction.

Shared Buffers Data Sampling (SBDS) (CVE-2022-21125)

After propagators may have moved data around the uncore and copied stale data into client core fill buffers, processors affected by MFBDS can leak data from the fill buffer. It is limited to the client (including Intel Xeon server E3) uncore implementation.

Shared Buffers Data Read (SBDR) (CVE-2022-21123)

It is similar to Shared Buffer Data Sampling (SBDS) except that the data is directly read into the architectural software-visible state. It is limited to the client (including Intel Xeon server E3) uncore implementation.

7.9.3 Affected Processors

Not all the CPUs are affected by all the variants. For instance, most processors for the server market (excluding Intel Xeon E3 processors) are impacted by only Device Register Partial Write (DRPW).

Below is the list of affected Intel processors[1]:

Common name	Family_Model	Steppings
HASWELL_X	06_3FH	2,4
SKYLAKE_L	06_4EH	3
BROADWELL_X	06_4FH	All
SKYLAKE_X	06_55H	3,4,6,7,11
BROADWELL_D	06_56H	3,4,5
SKYLAKE	06_5EH	3
ICELAKE_X	06_6AH	4,5,6
ICELAKE_D	06_6CH	1
ICELAKE_L	06_7EH	5
ATOM_TREMONT_D	06_86H	All
LAKEFIELD	06_8AH	1
KABYLAKE_L	06_8EH	9 to 12
ATOM_TREMONT	06_96H	1
ATOM_TREMONT_L	06_9CH	0
KABYLAKE	06_9EH	9 to 13
COMETLAKE	06_A5H	2,3,5
COMETLAKE_L	06_A6H	0,1
ROCKETLAKE	06_A7H	1

If a CPU is in the affected processor list, but not affected by a variant, it is indicated by new bits in MSR IA32_ARCH_CAPABILITIES. As described in a later section, mitigation largely remains the same for all the variants, i.e. to clear the CPU fill buffers via VERW instruction.

7.9.4 New bits in MSRs

Newer processors and microcode update on existing affected processors added new bits to IA32_ARCH_CAPABILITIES MSR. These bits can be used to enumerate specific variants of Processor MMIO Stale Data vulnerabilities and mitigation capability.

[1] Affected Processors https://www.intel.com/content/www/us/en/developer/topic-technology/software-security-guidance/processors-affected-consolidated-product-cpu-model.html

MSR IA32_ARCH_CAPABILITIES

Bit 13 - SBDR_SSDP_NO - When set, processor is not affected by either the
Shared Buffers Data Read (SBDR) vulnerability or the sideband stale data propagator (SSDP).

Bit 14 - FBSDP_NO - When set, processor is not affected by the Fill Buffer
Stale Data Propagator (FBSDP).

Bit 15 - PSDP_NO - When set, processor is not affected by Primary Stale Data
Propagator (PSDP).

Bit 17 - FB_CLEAR - When set, VERW instruction will overwrite CPU fill buffer
values as part of MD_CLEAR operations. Processors that do not enumerate MDS_NO (meaning they are affected by MDS) but that do enumerate support for both L1D_FLUSH and MD_CLEAR implicitly enumerate FB_CLEAR as part of their MD_CLEAR support.

Bit 18 - FB_CLEAR_CTRL - Processor supports read and write to MSR
IA32_MCU_OPT_CTRL[FB_CLEAR_DIS]. On such processors, the FB_CLEAR_DIS bit can be set to cause the VERW instruction to not perform the FB_CLEAR action. Not all processors that support FB_CLEAR will support FB_CLEAR_CTRL.

MSR IA32_MCU_OPT_CTRL

Bit 3 - FB_CLEAR_DIS - When set, VERW instruction does not perform the FB_CLEAR action. This may be useful to reduce the performance impact of FB_CLEAR in cases where system software deems it warranted (for example, when performance is more critical, or the untrusted software has no MMIO access). Note that FB_CLEAR_DIS has no impact on enumeration (for example, it does not change FB_CLEAR or MD_CLEAR enumeration) and it may not be supported on all processors that enumerate FB_CLEAR.

7.9.5 Mitigation

Like MDS, all variants of Processor MMIO Stale Data vulnerabilities have the same mitigation strategy to force the CPU to clear the affected buffers before an attacker can extract the secrets.

This is achieved by using the otherwise unused and obsolete VERW instruction in combination with a microcode update. The microcode clears the affected CPU buffers when the VERW instruction is executed.

Kernel reuses the MDS function to invoke the buffer clearing:

```
mds_clear_cpu_buffers()
```

On MDS affected CPUs, the kernel already invokes CPU buffer clear on kernel/userspace, hypervisor/guest and C-state (idle) transitions. No additional mitigation is needed on such CPUs.

For CPUs not affected by MDS or TAA, mitigation is needed only for the attacker with MMIO capability. Therefore, VERW is not required for kernel/userspace. For virtualization case, VERW is only needed at VMENTER for a guest with MMIO capability.

Mitigation points

Return to user space

Same mitigation as MDS when affected by MDS/TAA, otherwise no mitigation needed.

C-State transition

Control register writes by CPU during C-state transition can propagate data from fill buffer to uncore buffers. Execute VERW before C-state transition to clear CPU fill buffers.

Guest entry point

Same mitigation as MDS when processor is also affected by MDS/TAA, otherwise execute VERW at VMENTER only for MMIO capable guests. On CPUs not affected by MDS/TAA, guest without MMIO access cannot extract secrets using Processor MMIO Stale Data vulnerabilities, so there is no need to execute VERW for such guests.

Mitigation control on the kernel command line

The kernel command line allows to control the Processor MMIO Stale Data mitigations at boot time with the option "mmio_stale_data=". The valid arguments for this option are:

full	If the CPU is vulnerable, enable mitigation; CPU buffer clearing on exit to userspace and when entering a VM. Idle transitions are protected as well. It does not automatically disable SMT.
full,nosmt	Same as full, with SMT disabled on vulnerable CPUs. This is the complete mitigation.
off	Disables mitigation completely.

If the CPU is affected and mmio_stale_data=off is not supplied on the kernel command line, then the kernel selects the appropriate mitigation.

Mitigation status information

The Linux kernel provides a sysfs interface to enumerate the current vulnerability status of the system: whether the system is vulnerable, and which mitigations are active. The relevant sysfs file is:

/sys/devices/system/cpu/vulnerabilities/mmio_stale_data

The possible values in this file are:

'Not affected'	The processor is not vulnerable
'Vulnerable'	The processor is vulnerable, but no mitigation enabled
'Vulnerable: Clear CPU buffers attempted, no microcode'	The processor is vulnerable but microcode is not updated. The mitigation is enabled on a best effort basis. If the processor is vulnerable but the availability of the microcode based mitigation mechanism is not advertised via CPUID, the kernel selects a best effort mitigation mode. This mode invokes the mitigation instructions without a guarantee that they clear the CPU buffers. This is done to address virtualization scenarios where the host has the microcode update applied, but the hypervisor is not yet updated to expose the CPUID to the guest. If the host has updated microcode the protection takes effect; otherwise a few CPU cycles are wasted pointlessly.
'Mitigation: Clear CPU buffers'	The processor is vulnerable and the CPU buffer clearing mitigation is enabled.
'Unknown: No mitigations'	The processor vulnerability status is unknown because it is out of Servicing period. Mitigation is not attempted.

Definitions:

Servicing period: The process of providing functional and security updates to Intel processors or platforms, utilizing the Intel Platform Update (IPU) process or other similar mechanisms.

End of Servicing Updates (ESU): ESU is the date at which Intel will no longer provide Servicing, such as through IPU or other similar update processes. ESU dates will typically be aligned to end of quarter.

If the processor is vulnerable then the following information is appended to the above information:

'SMT vulnerable'	SMT is enabled
'SMT disabled'	SMT is disabled
'SMT Host state unknown'	Kernel runs in a VM, Host SMT state unknown

References

7.10 Cross-Thread Return Address Predictions

Certain AMD and Hygon processors are subject to a cross-thread return address predictions vulnerability. When running in SMT mode and one sibling thread transitions out of C0 state, the other sibling thread could use return target predictions from the sibling thread that transitioned out of C0.

The Spectre v2 mitigations protect the Linux kernel, as it fills the return address prediction entries with safe targets when context switching to the idle thread. However, KVM does allow a VMM to prevent exiting guest mode when transitioning out of C0. This could result in a guest-controlled return target being consumed by the sibling thread.

7.10.1 Affected processors

The following CPUs are vulnerable:

- AMD Family 17h processors
- Hygon Family 18h processors

7.10.2 Related CVEs

The following CVE entry is related to this issue:

CVE-2022-27672	Cross-Thread Return Address Predictions

7.10.3 Problem

Affected SMT-capable processors support 1T and 2T modes of execution when SMT is enabled. In 2T mode, both threads in a core are executing code. For the processor core to enter 1T mode, it is required that one of the threads requests to transition out of the C0 state. This can be communicated with the HLT instruction or with an MWAIT instruction that requests non-C0. When the thread re-enters the C0 state, the processor transitions back to 2T mode, assuming the other thread is also still in C0 state.

In affected processors, the return address predictor (RAP) is partitioned depending on the SMT mode. For instance, in 2T mode each thread uses a private 16-entry RAP, but in 1T mode, the active thread uses a 32-entry RAP. Upon transition between 1T/2T mode, the RAP contents are not modified but the RAP pointers (which control the next return target to use for predictions) may change. This behavior may result in return targets from one SMT thread being used by RET predictions in the sibling thread following a 1T/2T switch. In particular, a RET instruction executed immediately after a transition to 1T may use a return target from the thread that just became idle. In theory, this could lead to information disclosure if the return targets used do not come from trustworthy code.

7.10.4 Attack scenarios

An attack can be mounted on affected processors by performing a series of CALL instructions with targeted return locations and then transitioning out of C0 state.

7.10.5 Mitigation mechanism

Before entering idle state, the kernel context switches to the idle thread. The context switch fills the RAP entries (referred to as the RSB in Linux) with safe targets by performing a sequence of CALL instructions.

Prevent a guest VM from directly putting the processor into an idle state by intercepting HLT and MWAIT instructions.

Both mitigations are required to fully address this issue.

7.10.6 Mitigation control on the kernel command line

Use existing Spectre v2 mitigations that will fill the RSB on context switch.

7.10.7 Mitigation control for KVM - module parameter

By default, the KVM hypervisor mitigates this issue by intercepting guest attempts to transition out of C0. A VMM can use the KVM_CAP_X86_DISABLE_EXITS capability to override those interceptions, but since this is not common, the mitigation that covers this path is not enabled by default.

The mitigation for the KVM_CAP_X86_DISABLE_EXITS capability can be turned on using the boolean module parameter mitigate_smt_rsb, e.g. `kvm.mitigate_smt_rsb=1`.

7.11 Speculative Return Stack Overflow (SRSO)

This is a mitigation for the speculative return stack overflow (SRSO) vulnerability found on AMD processors. The mechanism is by now the well known scenario of poisoning CPU functional units - the Branch Target Buffer (BTB) and Return Address Predictor (RAP) in this case - and then tricking the elevated privilege domain (the kernel) into leaking sensitive data.

AMD CPUs predict RET instructions using a Return Address Predictor (aka Return Address Stack/Return Stack Buffer). In some cases, a non-architectural CALL instruction (i.e., an instruction predicted to be a CALL but is not actually a CALL) can create an entry in the RAP which may be used to predict the target of a subsequent RET instruction.

The specific circumstances that lead to this varies by microarchitecture but the concern is that an attacker can mis-train the CPU BTB to predict non-architectural CALL instructions in kernel space and use this to control the speculative target of a subsequent kernel RET, potentially leading to information disclosure via a speculative side-channel.

The issue is tracked under CVE-2023-20569.

7.11.1 Affected processors

AMD Zen, generations 1-4. That is, all families 0x17 and 0x19. Older processors have not been investigated.

7.11.2 System information and options

First of all, it is required that the latest microcode be loaded for mitigations to be effective.

The sysfs file showing SRSO mitigation status is:

/sys/devices/system/cpu/vulnerabilities/spec_rstack_overflow

The possible values in this file are:

- 'Not affected':

  The processor is not vulnerable

- 'Vulnerable':

  The processor is vulnerable and no mitigations have been applied.

  - 'Vulnerable: No microcode':

    The processor is vulnerable, no microcode extending IBPB functionality to address the vulnerability has been applied.

  - 'Vulnerable: Safe RET, no microcode':

    The "Safe RET" mitigation (see below) has been applied to protect the kernel, but the IBPB-extending microcode has not been applied. User space tasks may still be vulnerable.

  - 'Vulnerable: Microcode, no safe RET':

    Extended IBPB functionality microcode patch has been applied. It does not address User->Kernel and Guest->Host transitions protection but it does address User->User and VM->VM attack vectors.

    Note that User->User mitigation is controlled by how the IBPB aspect in the Spectre v2 mitigation is selected:

    - conditional IBPB:

      where each process can select whether it needs an IBPB issued around it PR_SPEC_DISABLE/_ENABLE etc, see *Spectre Side Channels*

    - strict:

      i.e., always on - by supplying spectre_v2_user=on on the kernel command line

    (spec_rstack_overflow=microcode)

  - 'Mitigation: Safe RET':

    Combined microcode/software mitigation. It complements the extended IBPB microcode patch functionality by addressing User->Kernel and Guest->Host transitions protection.

    Selected by default or by spec_rstack_overflow=safe-ret

- 'Mitigation: IBPB':

  Similar protection as "safe RET" above but employs an IBPB barrier on privilege domain crossings (User->Kernel, Guest->Host).

  (spec_rstack_overflow=ibpb)

- 'Mitigation: IBPB on VMEXIT':

  Mitigation addressing the cloud provider scenario - the Guest->Host transitions only.

  (spec_rstack_overflow=ibpb-vmexit)

In order to exploit vulnerability, an attacker needs to:

- gain local access on the machine
- break kASLR
- find gadgets in the running kernel in order to use them in the exploit
- potentially create and pin an additional workload on the sibling thread, depending on the microarchitecture (not necessary on fam 0x19)
- run the exploit

Considering the performance implications of each mitigation type, the default one is 'Mitigation: safe RET' which should take care of most attack vectors, including the local User->Kernel one.

As always, the user is advised to keep her/his system up-to-date by applying software updates regularly.

The default setting will be reevaluated when needed and especially when new attack vectors appear.

As one can surmise, 'Mitigation: safe RET' does come at the cost of some performance depending on the workload. If one trusts her/his userspace and does not want to suffer the performance impact, one can always disable the mitigation with spec_rstack_overflow=off.

Similarly, 'Mitigation: IBPB' is another full mitigation type employing an indrect branch prediction barrier after having applied the required microcode patch for one's system. This mitigation comes also at a performance cost.

7.11.3 Mitigation: Safe RET

The mitigation works by ensuring all RET instructions speculate to a controlled location, similar to how speculation is controlled in the retpoline sequence. To accomplish this, the __x86_return_thunk forces the CPU to mispredict every function return using a 'safe return' sequence.

To ensure the safety of this mitigation, the kernel must ensure that the safe return sequence is itself free from attacker interference. In Zen3 and Zen4, this is accomplished by creating a BTB alias between the untraining function srso_alias_untrain_ret() and the safe return function srso_alias_safe_ret() which results in evicting a potentially poisoned BTB entry and using that safe one for all function returns.

In older Zen1 and Zen2, this is accomplished using a reinterpretation technique similar to Retbleed one: srso_untrain_ret() and srso_safe_ret().

7.12 GDS - Gather Data Sampling

Gather Data Sampling is a hardware vulnerability which allows unprivileged speculative access to data which was previously stored in vector registers.

7.12.1 Problem

When a gather instruction performs loads from memory, different data elements are merged into the destination vector register. However, when a gather instruction that is transiently executed encounters a fault, stale data from architectural or internal vector registers may get transiently forwarded to the destination vector register instead. This will allow a malicious attacker to infer stale data using typical side channel techniques like cache timing attacks. GDS is a purely sampling-based attack.

The attacker uses gather instructions to infer the stale vector register data. The victim does not need to do anything special other than use the vector registers. The victim does not need to use gather instructions to be vulnerable.

Because the buffers are shared between Hyper-Threads cross Hyper-Thread attacks are possible.

7.12.2 Attack scenarios

Without mitigation, GDS can infer stale data across virtually all permission boundaries:

> Non-enclaves can infer SGX enclave data Userspace can infer kernel data Guests can infer data from hosts Guest can infer guest from other guests Users can infer data from other users

Because of this, it is important to ensure that the mitigation stays enabled in lower-privilege contexts like guests and when running outside SGX enclaves.

The hardware enforces the mitigation for SGX. Likewise, VMMs should ensure that guests are not allowed to disable the GDS mitigation. If a host erred and allowed this, a guest could theoretically disable GDS mitigation, mount an attack, and re-enable it.

7.12.3 Mitigation mechanism

This issue is mitigated in microcode. The microcode defines the following new bits:

IA32_ARCH_CAPABILITIES[GDS_CTRL]	RO	Enumerates GDS vulnerability and mitigation support.
IA32_ARCH_CAPABILITIES[GDS_NO]	RO	Processor is not vulnerable.
IA32_MCU_OPT_CTRL[GDS_MITG_DIS]	RW	Disables the mitigation 0 by default.
IA32_MCU_OPT_CTRL[GDS_MITG_LOCK]	RW	GDS_MITG_DIS=0. Writes to GDS_MITG_DIS are ignored Can't be cleared once set.

GDS can also be mitigated on systems that don't have updated microcode by disabling AVX. This can be done by setting gather_data_sampling="force" or "clearcpuid=avx" on the kernel command-line.

If used, these options will disable AVX use by turning off XSAVE YMM support. However, the processor will still enumerate AVX support. Userspace that does not follow proper AVX enumeration to check both AVX *and* XSAVE YMM support will break.

7.12.4 Mitigation control on the kernel command line

The mitigation can be disabled by setting "gather_data_sampling=off" or "mitigations=off" on the kernel command line. Not specifying either will default to the mitigation being enabled. Specifying "gather_data_sampling=force" will use the microcode mitigation when available or disable AVX on affected systems where the microcode hasn't been updated to include the mitigation.

7.12.5 GDS System Information

The kernel provides vulnerability status information through sysfs. For GDS this can be accessed by the following sysfs file:

/sys/devices/system/cpu/vulnerabilities/gather_data_sampling

The possible values contained in this file are:

Not affected	Processor not vulnerable.
Vulnerable	Processor vulnerable and mitigation disabled.
Vulnerable: No microcode	Processor vulnerable and microcode is missing mitigation.
Mitigation: AVX disabled,	
no microcode	Processor is vulnerable and microcode is missing mitigation. AVX disabled as mitigation.
Mitigation: Microcode	Processor is vulnerable and mitigation is in effect.
Mitigation: Microcode (locked)	Processor is vulnerable and mitigation is in effect and cannot be disabled.
Unknown: Dependent on	
hypervisor status	Running on a virtual guest processor that is affected but with no way to know if host processor is mitigated or vulnerable.

7.12.6 GDS Default mitigation

The updated microcode will enable the mitigation by default. The kernel's default action is to leave the mitigation enabled.

Here is a set of documents aimed at users who are trying to track down problems and bugs in particular.

CHAPTER

EIGHT

REPORTING ISSUES

8.1 The short guide (aka TL;DR)

Are you facing a regression with vanilla kernels from the same stable or longterm series? One still supported? Then search the LKML and the Linux stable mailing list archives for matching reports to join. If you don't find any, install the latest release from that series. If it still shows the issue, report it to the stable mailing list (stable@vger.kernel.org) and CC the regressions list (regressions@lists.linux.dev); ideally also CC the maintainer and the mailing list for the subsystem in question.

In all other cases try your best guess which kernel part might be causing the issue. Check the MAINTAINERS file for how its developers expect to be told about problems, which most of the time will be by email with a mailing list in CC. Check the destination's archives for matching reports; search the LKML and the web, too. If you don't find any to join, install the latest mainline kernel. If the issue is present there, send a report.

The issue was fixed there, but you would like to see it resolved in a still supported stable or longterm series as well? Then install its latest release. If it shows the problem, search for the change that fixed it in mainline and check if backporting is in the works or was discarded; if it's neither, ask those who handled the change for it.

General remarks: When installing and testing a kernel as outlined above, ensure it's vanilla (IOW: not patched and not using add-on modules). Also make sure it's built and running in a healthy environment and not already tainted before the issue occurs.

If you are facing multiple issues with the Linux kernel at once, report each separately. While writing your report, include all information relevant to the issue, like the kernel and the distro used. In case of a regression, CC the regressions mailing list (regressions@lists.linux.dev) to your report. Also try to pin-point the culprit with a bisection; if you succeed, include its commit-id and CC everyone in the sign-off-by chain.

Once the report is out, answer any questions that come up and help where you can. That includes keeping the ball rolling by occasionally retesting with newer releases and sending a status update afterwards.

8.2 Step-by-step guide how to report issues to the kernel maintainers

The above TL;DR outlines roughly how to report issues to the Linux kernel developers. It might be all that's needed for people already familiar with reporting issues to Free/Libre & Open Source Software (FLOSS) projects. For everyone else there is this section. It is more detailed and uses a step-by-step approach. It still tries to be brief for readability and leaves out a lot of details; those are described below the step-by-step guide in a reference section, which explains each of the steps in more detail.

Note: this section covers a few more aspects than the TL;DR and does things in a slightly different order. That's in your interest, to make sure you notice early if an issue that looks like a Linux kernel problem is actually caused by something else. These steps thus help to ensure the time you invest in this process won't feel wasted in the end:

- Are you facing an issue with a Linux kernel a hardware or software vendor provided? Then in almost all cases you are better off to stop reading this document and reporting the issue to your vendor instead, unless you are willing to install the latest Linux version yourself. Be aware the latter will often be needed anyway to hunt down and fix issues.
- Perform a rough search for existing reports with your favorite internet search engine; additionally, check the archives of the Linux Kernel Mailing List (LKML). If you find matching reports, join the discussion instead of sending a new one.
- See if the issue you are dealing with qualifies as regression, security issue, or a really severe problem: those are 'issues of high priority' that need special handling in some steps that are about to follow.
- Make sure it's not the kernel's surroundings that are causing the issue you face.
- Create a fresh backup and put system repair and restore tools at hand.
- Ensure your system does not enhance its kernels by building additional kernel modules on-the-fly, which solutions like DKMS might be doing locally without your knowledge.
- Check if your kernel was 'tainted' when the issue occurred, as the event that made the kernel set this flag might be causing the issue you face.
- Write down coarsely how to reproduce the issue. If you deal with multiple issues at once, create separate notes for each of them and make sure they work independently on a freshly booted system. That's needed, as each issue needs to get reported to the kernel developers separately, unless they are strongly entangled.
- If you are facing a regression within a stable or longterm version line (say something broke when updating from 5.10.4 to 5.10.5), scroll down to 'Dealing with regressions within a stable and longterm kernel line'.
- Locate the driver or kernel subsystem that seems to be causing the issue. Find out how and where its developers expect reports. Note: most of the time this won't be bugzilla.kernel.org, as issues typically need to be sent by mail to a maintainer and a public mailing list.
- Search the archives of the bug tracker or mailing list in question thoroughly for reports that might match your issue. If you find anything, join the discussion instead of sending a new report.

After these preparations you'll now enter the main part:

- Unless you are already running the latest 'mainline' Linux kernel, better go and install it for the reporting process. Testing and reporting with the latest 'stable' Linux can be an acceptable alternative in some situations; during the merge window that actually might be even the best approach, but in that development phase it can be an even better idea to suspend your efforts for a few days anyway. Whatever version you choose, ideally use a 'vanilla' build. Ignoring these advices will dramatically increase the risk your report will be rejected or ignored.
- Ensure the kernel you just installed does not 'taint' itself when running.
- Reproduce the issue with the kernel you just installed. If it doesn't show up there, scroll down to the instructions for issues only happening with stable and longterm kernels.
- Optimize your notes: try to find and write the most straightforward way to reproduce your issue. Make sure the end result has all the important details, and at the same time is easy to read and understand for others that hear about it for the first time. And if you learned something in this process, consider searching again for existing reports about the issue.
- If your failure involves a 'panic', 'Oops', 'warning', or 'BUG', consider decoding the kernel log to find the line of code that triggered the error.
- If your problem is a regression, try to narrow down when the issue was introduced as much as possible.
- Start to compile the report by writing a detailed description about the issue. Always mention a few things: the latest kernel version you installed for reproducing, the Linux Distribution used, and your notes on how to reproduce the issue. Ideally, make the kernel's build configuration (.config) and the output from `dmesg` available somewhere on the net and link to it. Include or upload all other information that might be relevant, like the output/screenshot of an Oops or the output from `lspci`. Once you wrote this main part, insert a normal length paragraph on top of it outlining the issue and the impact quickly. On top of this add one sentence that briefly describes the problem and gets people to read on. Now give the thing a descriptive title or subject that yet again is shorter. Then you're ready to send or file the report like the MAINTAINERS file told you, unless you are dealing with one of those 'issues of high priority': they need special care which is explained in 'Special handling for high priority issues' below.
- Wait for reactions and keep the thing rolling until you can accept the outcome in one way or the other. Thus react publicly and in a timely manner to any inquiries. Test proposed fixes. Do proactive testing: retest with at least every first release candidate (RC) of a new mainline version and report your results. Send friendly reminders if things stall. And try to help yourself, if you don't get any help or if it's unsatisfying.

8.2.1 Reporting regressions within a stable and longterm kernel line

This subsection is for you, if you followed above process and got sent here at the point about regression within a stable or longterm kernel version line. You face one of those if something breaks when updating from 5.10.4 to 5.10.5 (a switch from 5.9.15 to 5.10.5 does not qualify). The developers want to fix such regressions as quickly as possible, hence there is a streamlined process to report them:

- Check if the kernel developers still maintain the Linux kernel version line you care about: go to the front page of kernel.org and make sure it mentions the latest release of the particular version line without an '[EOL]' tag.

- Check the archives of the Linux stable mailing list for existing reports.
- Install the latest release from the particular version line as a vanilla kernel. Ensure this kernel is not tainted and still shows the problem, as the issue might have already been fixed there. If you first noticed the problem with a vendor kernel, check a vanilla build of the last version known to work performs fine as well.
- Send a short problem report to the Linux stable mailing list (stable@vger.kernel.org) and CC the Linux regressions mailing list (regressions@lists.linux.dev); if you suspect the cause in a particular subsystem, CC its maintainer and its mailing list. Roughly describe the issue and ideally explain how to reproduce it. Mention the first version that shows the problem and the last version that's working fine. Then wait for further instructions.

The reference section below explains each of these steps in more detail.

8.2.2 Reporting issues only occurring in older kernel version lines

This subsection is for you, if you tried the latest mainline kernel as outlined above, but failed to reproduce your issue there; at the same time you want to see the issue fixed in a still supported stable or longterm series or vendor kernels regularly rebased on those. If that the case, follow these steps:

- Prepare yourself for the possibility that going through the next few steps might not get the issue solved in older releases: the fix might be too big or risky to get backported there.
- Perform the first three steps in the section ”Dealing with regressions within a stable and longterm kernel line” above.
- Search the Linux kernel version control system for the change that fixed the issue in mainline, as its commit message might tell you if the fix is scheduled for backporting already. If you don't find anything that way, search the appropriate mailing lists for posts that discuss such an issue or peer-review possible fixes; then check the discussions if the fix was deemed unsuitable for backporting. If backporting was not considered at all, join the newest discussion, asking if it's in the cards.
- One of the former steps should lead to a solution. If that doesn't work out, ask the maintainers for the subsystem that seems to be causing the issue for advice; CC the mailing list for the particular subsystem as well as the stable mailing list.

The reference section below explains each of these steps in more detail.

8.3 Reference section: Reporting issues to the kernel maintainers

The detailed guides above outline all the major steps in brief fashion, which should be enough for most people. But sometimes there are situations where even experienced users might wonder how to actually do one of those steps. That's what this section is for, as it will provide a lot more details on each of the above steps. Consider this as reference documentation: it's possible to read it from top to bottom. But it's mainly meant to skim over and a place to look up details how to actually perform those steps.

A few words of general advice before digging into the details:

- The Linux kernel developers are well aware this process is complicated and demands more than other FLOSS projects. We'd love to make it simpler. But that would require work in various places as well as some infrastructure, which would need constant maintenance; nobody has stepped up to do that work, so that's just how things are for now.
- A warranty or support contract with some vendor doesn't entitle you to request fixes from developers in the upstream Linux kernel community: such contracts are completely outside the scope of the Linux kernel, its development community, and this document. That's why you can't demand anything such a contract guarantees in this context, not even if the developer handling the issue works for the vendor in question. If you want to claim your rights, use the vendor's support channel instead. When doing so, you might want to mention you'd like to see the issue fixed in the upstream Linux kernel; motivate them by saying it's the only way to ensure the fix in the end will get incorporated in all Linux distributions.
- If you never reported an issue to a FLOSS project before you should consider reading How to Report Bugs Effectively, How To Ask Questions The Smart Way, and How to ask good questions.

With that off the table, find below the details on how to properly report issues to the Linux kernel developers.

8.3.1 Make sure you're using the upstream Linux kernel

> *Are you facing an issue with a Linux kernel a hardware or software vendor provided? Then in almost all cases you are better off to stop reading this document and reporting the issue to your vendor instead, unless you are willing to install the latest Linux version yourself. Be aware the latter will often be needed anyway to hunt down and fix issues.*

Like most programmers, Linux kernel developers don't like to spend time dealing with reports for issues that don't even happen with their current code. It's just a waste everybody's time, especially yours. Unfortunately such situations easily happen when it comes to the kernel and often leads to frustration on both sides. That's because almost all Linux-based kernels preinstalled on devices (Computers, Laptops, Smartphones, Routers, ...) and most shipped by Linux distributors are quite distant from the official Linux kernel as distributed by kernel.org: these kernels from these vendors are often ancient from the point of Linux development or heavily modified, often both.

Most of these vendor kernels are quite unsuitable for reporting issues to the Linux kernel developers: an issue you face with one of them might have been fixed by the Linux kernel developers months or years ago already; additionally, the modifications and enhancements by the vendor might be causing the issue you face, even if they look small or totally unrelated. That's why you should report issues with these kernels to the vendor. Its developers should look into the report and, in case it turns out to be an upstream issue, fix it directly upstream or forward the report there. In practice that often does not work out or might not what you want. You thus might want to consider circumventing the vendor by installing the very latest Linux kernel core yourself. If that's an option for you move ahead in this process, as a later step in this guide will explain how to do that once it rules out other potential causes for your issue.

Note, the previous paragraph is starting with the word 'most', as sometimes developers in fact are willing to handle reports about issues occurring with vendor kernels. If they do in the end highly depends on the developers and the issue in question. Your chances are quite good if the distributor applied only small modifications to a kernel based on a recent Linux version; that for example often holds true for the mainline kernels shipped by Debian GNU/Linux Sid or Fedora

Rawhide. Some developers will also accept reports about issues with kernels from distributions shipping the latest stable kernel, as long as its only slightly modified; that for example is often the case for Arch Linux, regular Fedora releases, and openSUSE Tumbleweed. But keep in mind, you better want to use a mainline Linux and avoid using a stable kernel for this process, as outlined in the section 'Install a fresh kernel for testing' in more detail.

Obviously you are free to ignore all this advice and report problems with an old or heavily modified vendor kernel to the upstream Linux developers. But note, those often get rejected or ignored, so consider yourself warned. But it's still better than not reporting the issue at all: sometimes such reports directly or indirectly will help to get the issue fixed over time.

8.3.2 Search for existing reports, first run

> *Perform a rough search for existing reports with your favorite internet search engine; additionally, check the archives of the Linux Kernel Mailing List (LKML). If you find matching reports, join the discussion instead of sending a new one.*

Reporting an issue that someone else already brought forward is often a waste of time for everyone involved, especially you as the reporter. So it's in your own interest to thoroughly check if somebody reported the issue already. At this step of the process it's okay to just perform a rough search: a later step will tell you to perform a more detailed search once you know where your issue needs to be reported to. Nevertheless, do not hurry with this step of the reporting process, it can save you time and trouble.

Simply search the internet with your favorite search engine first. Afterwards, search the Linux Kernel Mailing List (LKML) archives.

If you get flooded with results consider telling your search engine to limit search timeframe to the past month or year. And wherever you search, make sure to use good search terms; vary them a few times, too. While doing so try to look at the issue from the perspective of someone else: that will help you to come up with other words to use as search terms. Also make sure not to use too many search terms at once. Remember to search with and without information like the name of the kernel driver or the name of the affected hardware component. But its exact brand name (say 'ASUS Red Devil Radeon RX 5700 XT Gaming OC') often is not much helpful, as it is too specific. Instead try search terms like the model line (Radeon 5700 or Radeon 5000) and the code name of the main chip ('Navi' or 'Navi10') with and without its manufacturer ('AMD').

In case you find an existing report about your issue, join the discussion, as you might be able to provide valuable additional information. That can be important even when a fix is prepared or in its final stages already, as developers might look for people that can provide additional information or test a proposed fix. Jump to the section 'Duties after the report went out' for details on how to get properly involved.

Note, searching bugzilla.kernel.org might also be a good idea, as that might provide valuable insights or turn up matching reports. If you find the latter, just keep in mind: most subsystems expect reports in different places, as described below in the section ”Check where you need to report your issue”. The developers that should take care of the issue thus might not even be aware of the bugzilla ticket. Hence, check the ticket if the issue already got reported as outlined in this document and if not consider doing so.

8.3.3 Issue of high priority?

See if the issue you are dealing with qualifies as regression, security issue, or a really severe problem: those are 'issues of high priority' that need special handling in some steps that are about to follow.

Linus Torvalds and the leading Linux kernel developers want to see some issues fixed as soon as possible, hence there are 'issues of high priority' that get handled slightly differently in the reporting process. Three type of cases qualify: regressions, security issues, and really severe problems.

You deal with a regression if some application or practical use case running fine with one Linux kernel works worse or not at all with a newer version compiled using a similar configuration. The document *Reporting regressions* explains this in more detail. It also provides a good deal of other information about regressions you might want to be aware of; it for example explains how to add your issue to the list of tracked regressions, to ensure it won't fall through the cracks.

What qualifies as security issue is left to your judgment. Consider reading Documentation/process/security-bugs.rst before proceeding, as it provides additional details how to best handle security issues.

An issue is a 'really severe problem' when something totally unacceptably bad happens. That's for example the case when a Linux kernel corrupts the data it's handling or damages hardware it's running on. You're also dealing with a severe issue when the kernel suddenly stops working with an error message ('kernel panic') or without any farewell note at all. Note: do not confuse a 'panic' (a fatal error where the kernel stop itself) with a 'Oops' (a recoverable error), as the kernel remains running after the latter.

8.3.4 Ensure a healthy environment

Make sure it's not the kernel's surroundings that are causing the issue you face.

Problems that look a lot like a kernel issue are sometimes caused by build or runtime environment. It's hard to rule out that problem completely, but you should minimize it:

- Use proven tools when building your kernel, as bugs in the compiler or the binutils can cause the resulting kernel to misbehave.
- Ensure your computer components run within their design specifications; that's especially important for the main processor, the main memory, and the motherboard. Therefore, stop undervolting or overclocking when facing a potential kernel issue.
- Try to make sure it's not faulty hardware that is causing your issue. Bad main memory for example can result in a multitude of issues that will manifest itself in problems looking like kernel issues.
- If you're dealing with a filesystem issue, you might want to check the file system in question with `fsck`, as it might be damaged in a way that leads to unexpected kernel behavior.
- When dealing with a regression, make sure it's not something else that changed in parallel to updating the kernel. The problem for example might be caused by other software that was updated at the same time. It can also happen that a hardware component coincidentally just broke when you rebooted into a new kernel for the first time. Updating the systems BIOS or changing something in the BIOS Setup can also lead to problems that on look a lot like a kernel regression.

8.3.5 Prepare for emergencies

Create a fresh backup and put system repair and restore tools at hand.

Reminder, you are dealing with computers, which sometimes do unexpected things, especially if you fiddle with crucial parts like the kernel of its operating system. That's what you are about to do in this process. Thus, make sure to create a fresh backup; also ensure you have all tools at hand to repair or reinstall the operating system as well as everything you need to restore the backup.

8.3.6 Make sure your kernel doesn't get enhanced

Ensure your system does not enhance its kernels by building additional kernel modules on-the-fly, which solutions like DKMS might be doing locally without your knowledge.

The risk your issue report gets ignored or rejected dramatically increases if your kernel gets enhanced in any way. That's why you should remove or disable mechanisms like akmods and DKMS: those build add-on kernel modules automatically, for example when you install a new Linux kernel or boot it for the first time. Also remove any modules they might have installed. Then reboot before proceeding.

Note, you might not be aware that your system is using one of these solutions: they often get set up silently when you install Nvidia's proprietary graphics driver, VirtualBox, or other software that requires a some support from a module not part of the Linux kernel. That why your might need to uninstall the packages with such software to get rid of any 3rd party kernel module.

8.3.7 Check 'taint' flag

Check if your kernel was 'tainted' when the issue occurred, as the event that made the kernel set this flag might be causing the issue you face.

The kernel marks itself with a 'taint' flag when something happens that might lead to follow-up errors that look totally unrelated. The issue you face might be such an error if your kernel is tainted. That's why it's in your interest to rule this out early before investing more time into this process. This is the only reason why this step is here, as this process later will tell you to install the latest mainline kernel; you will need to check the taint flag again then, as that's when it matters because it's the kernel the report will focus on.

On a running system is easy to check if the kernel tainted itself: if `cat /proc/sys/kernel/tainted` returns '0' then the kernel is not tainted and everything is fine. Checking that file is impossible in some situations; that's why the kernel also mentions the taint status when it reports an internal problem (a 'kernel bug'), a recoverable error (a 'kernel Oops') or a non-recoverable error before halting operation (a 'kernel panic'). Look near the top of the error messages printed when one of these occurs and search for a line starting with 'CPU:'. It should end with 'Not tainted' if the kernel was not tainted when it noticed the problem; it was tainted if you see 'Tainted:' followed by a few spaces and some letters.

If your kernel is tainted, study *Tainted kernels* to find out why. Try to eliminate the reason. Often it's caused by one these three things:

1. A recoverable error (a 'kernel Oops') occurred and the kernel tainted itself, as the kernel knows it might misbehave in strange ways after that point. In that case check your kernel

or system log and look for a section that starts with this:

```
Oops: 0000 [#1] SMP
```

That's the first Oops since boot-up, as the '#1' between the brackets shows. Every Oops and any other problem that happens after that point might be a follow-up problem to that first Oops, even if both look totally unrelated. Rule this out by getting rid of the cause for the first Oops and reproducing the issue afterwards. Sometimes simply restarting will be enough, sometimes a change to the configuration followed by a reboot can eliminate the Oops. But don't invest too much time into this at this point of the process, as the cause for the Oops might already be fixed in the newer Linux kernel version you are going to install later in this process.

2. Your system uses a software that installs its own kernel modules, for example Nvidia's proprietary graphics driver or VirtualBox. The kernel taints itself when it loads such module from external sources (even if they are Open Source): they sometimes cause errors in unrelated kernel areas and thus might be causing the issue you face. You therefore have to prevent those modules from loading when you want to report an issue to the Linux kernel developers. Most of the time the easiest way to do that is: temporarily uninstall such software including any modules they might have installed. Afterwards reboot.

3. The kernel also taints itself when it's loading a module that resides in the staging tree of the Linux kernel source. That's a special area for code (mostly drivers) that does not yet fulfill the normal Linux kernel quality standards. When you report an issue with such a module it's obviously okay if the kernel is tainted; just make sure the module in question is the only reason for the taint. If the issue happens in an unrelated area reboot and temporarily block the module from being loaded by specifying `foo.blacklist=1` as kernel parameter (replace 'foo' with the name of the module in question).

8.3.8 Document how to reproduce issue

Write down coarsely how to reproduce the issue. If you deal with multiple issues at once, create separate notes for each of them and make sure they work independently on a freshly booted system. That's needed, as each issue needs to get reported to the kernel developers separately, unless they are strongly entangled.

If you deal with multiple issues at once, you'll have to report each of them separately, as they might be handled by different developers. Describing various issues in one report also makes it quite difficult for others to tear it apart. Hence, only combine issues in one report if they are very strongly entangled.

Additionally, during the reporting process you will have to test if the issue happens with other kernel versions. Therefore, it will make your work easier if you know exactly how to reproduce an issue quickly on a freshly booted system.

Note: it's often fruitless to report issues that only happened once, as they might be caused by a bit flip due to cosmic radiation. That's why you should try to rule that out by reproducing the issue before going further. Feel free to ignore this advice if you are experienced enough to tell a one-time error due to faulty hardware apart from a kernel issue that rarely happens and thus is hard to reproduce.

8.3.9 Regression in stable or longterm kernel?

> *If you are facing a regression within a stable or longterm version line (say something broke when updating from 5.10.4 to 5.10.5), scroll down to 'Dealing with regressions within a stable and longterm kernel line'.*

Regression within a stable and longterm kernel version line are something the Linux developers want to fix badly, as such issues are even more unwanted than regression in the main development branch, as they can quickly affect a lot of people. The developers thus want to learn about such issues as quickly as possible, hence there is a streamlined process to report them. Note, regressions with newer kernel version line (say something broke when switching from 5.9.15 to 5.10.5) do not qualify.

8.3.10 Check where you need to report your issue

> *Locate the driver or kernel subsystem that seems to be causing the issue. Find out how and where its developers expect reports. Note: most of the time this won't be bugzilla.kernel.org, as issues typically need to be sent by mail to a maintainer and a public mailing list.*

It's crucial to send your report to the right people, as the Linux kernel is a big project and most of its developers are only familiar with a small subset of it. Quite a few programmers for example only care for just one driver, for example one for a WiFi chip; its developer likely will only have small or no knowledge about the internals of remote or unrelated ”subsystems”, like the TCP stack, the PCIe/PCI subsystem, memory management or file systems.

Problem is: the Linux kernel lacks a central bug tracker where you can simply file your issue and make it reach the developers that need to know about it. That's why you have to find the right place and way to report issues yourself. You can do that with the help of a script (see below), but it mainly targets kernel developers and experts. For everybody else the MAINTAINERS file is the better place.

How to read the MAINTAINERS file

To illustrate how to use the MAINTAINERS file, lets assume the WiFi in your Laptop suddenly misbehaves after updating the kernel. In that case it's likely an issue in the WiFi driver. Obviously it could also be some code it builds upon, but unless you suspect something like that stick to the driver. If it's really something else, the driver's developers will get the right people involved.

Sadly, there is no way to check which code is driving a particular hardware component that is both universal and easy.

In case of a problem with the WiFi driver you for example might want to look at the output of `lspci -k`, as it lists devices on the PCI/PCIe bus and the kernel module driving it:

```
[user@something ~]$ lspci -k
[...]
3a:00.0 Network controller: Qualcomm Atheros QCA6174 802.11ac Wireless Network␣
↪Adapter (rev 32)
  Subsystem: Bigfoot Networks, Inc. Device 1535
  Kernel driver in use: ath10k_pci
```

```
  Kernel modules: ath10k_pci
[...]
```

But this approach won't work if your WiFi chip is connected over USB or some other internal bus. In those cases you might want to check your WiFi manager or the output of `ip link`. Look for the name of the problematic network interface, which might be something like 'wlp58s0'. This name can be used like this to find the module driving it:

```
[user@something ~]$ realpath --relative-to=/sys/module/ /sys/class/net/wlp58s0/
↪device/driver/module
ath10k_pci
```

In case tricks like these don't bring you any further, try to search the internet on how to narrow down the driver or subsystem in question. And if you are unsure which it is: just try your best guess, somebody will help you if you guessed poorly.

Once you know the driver or subsystem, you want to search for it in the MAINTAINERS file. In the case of 'ath10k_pci' you won't find anything, as the name is too specific. Sometimes you will need to search on the net for help; but before doing so, try a somewhat shorted or modified name when searching the MAINTAINERS file, as then you might find something like this:

```
QUALCOMM ATHEROS ATH10K WIRELESS DRIVER
Mail:           A. Some Human <shuman@example.com>
Mailing list:   ath10k@lists.infradead.org
Status:         Supported
Web-page:       https://wireless.wiki.kernel.org/en/users/Drivers/ath10k
SCM:            git git://git.kernel.org/pub/scm/linux/kernel/git/kvalo/ath.git
Files:          drivers/net/wireless/ath/ath10k/
```

Note: the line description will be abbreviations, if you read the plain MAINTAINERS file found in the root of the Linux source tree. 'Mail:' for example will be 'M:', 'Mailing list:' will be 'L', and 'Status:' will be 'S:'. A section near the top of the file explains these and other abbreviations.

First look at the line 'Status'. Ideally it should be 'Supported' or 'Maintained'. If it states 'Obsolete' then you are using some outdated approach that was replaced by a newer solution you need to switch to. Sometimes the code only has someone who provides 'Odd Fixes' when feeling motivated. And with 'Orphan' you are totally out of luck, as nobody takes care of the code anymore. That only leaves these options: arrange yourself to live with the issue, fix it yourself, or find a programmer somewhere willing to fix it.

After checking the status, look for a line starting with 'bugs:': it will tell you where to find a subsystem specific bug tracker to file your issue. The example above does not have such a line. That is the case for most sections, as Linux kernel development is completely driven by mail. Very few subsystems use a bug tracker, and only some of those rely on bugzilla.kernel.org.

In this and many other cases you thus have to look for lines starting with 'Mail:' instead. Those mention the name and the email addresses for the maintainers of the particular code. Also look for a line starting with 'Mailing list:', which tells you the public mailing list where the code is developed. Your report later needs to go by mail to those addresses. Additionally, for all issue reports sent by email, make sure to add the Linux Kernel Mailing List (LKML) <linux-kernel@vger.kernel.org> to CC. Don't omit either of the mailing lists when sending your issue report by mail later! Maintainers are busy people and might leave some work for other developers on the subsystem specific list; and LKML is important to have one place where all

issue reports can be found.

Finding the maintainers with the help of a script

For people that have the Linux sources at hand there is a second option to find the proper place to report: the script 'scripts/get_maintainer.pl' which tries to find all people to contact. It queries the MAINTAINERS file and needs to be called with a path to the source code in question. For drivers compiled as module if often can be found with a command like this:

```
$ modinfo ath10k_pci | grep filename | sed 's!/lib/modules/.*/kernel/!!; s!
↪filename:!!; s!\.ko\(\|\.xz\)!!'
drivers/net/wireless/ath/ath10k/ath10k_pci.ko
```

Pass parts of this to the script:

```
$ ./scripts/get_maintainer.pl -f drivers/net/wireless/ath/ath10k*
Some Human <shuman@example.com> (supporter:QUALCOMM ATHEROS ATH10K WIRELESS␣
↪DRIVER)
Another S. Human <asomehuman@example.com> (maintainer:NETWORKING DRIVERS)
ath10k@lists.infradead.org (open list:QUALCOMM ATHEROS ATH10K WIRELESS DRIVER)
linux-wireless@vger.kernel.org (open list:NETWORKING DRIVERS (WIRELESS))
netdev@vger.kernel.org (open list:NETWORKING DRIVERS)
linux-kernel@vger.kernel.org (open list)
```

Don't sent your report to all of them. Send it to the maintainers, which the script calls ”supporter:”; additionally CC the most specific mailing list for the code as well as the Linux Kernel Mailing List (LKML). In this case you thus would need to send the report to 'Some Human <shuman@example.com>' with 'ath10k@lists.infradead.org' and 'linux-kernel@vger.kernel.org' in CC.

Note: in case you cloned the Linux sources with git you might want to call `get_maintainer.pl` a second time with `--git`. The script then will look at the commit history to find which people recently worked on the code in question, as they might be able to help. But use these results with care, as it can easily send you in a wrong direction. That for example happens quickly in areas rarely changed (like old or unmaintained drivers): sometimes such code is modified during tree-wide cleanups by developers that do not care about the particular driver at all.

8.3.11 Search for existing reports, second run

> *Search the archives of the bug tracker or mailing list in question thoroughly for reports that might match your issue. If you find anything, join the discussion instead of sending a new report.*

As mentioned earlier already: reporting an issue that someone else already brought forward is often a waste of time for everyone involved, especially you as the reporter. That's why you should search for existing report again, now that you know where they need to be reported to. If it's mailing list, you will often find its archives on lore.kernel.org.

But some list are hosted in different places. That for example is the case for the ath10k WiFi driver used as example in the previous step. But you'll often find the archives for these lists easily on the net. Searching for 'archive ath10k@lists.infradead.org' for example will lead you to the Info page for the ath10k mailing list, which at the top links to its list archives. Sadly

this and quite a few other lists miss a way to search the archives. In those cases use a regular internet search engine and add something like 'site:lists.infradead.org/pipermail/ath10k/' to your search terms, which limits the results to the archives at that URL.

It's also wise to check the internet, LKML and maybe bugzilla.kernel.org again at this point. If your report needs to be filed in a bug tracker, you may want to check the mailing list archives for the subsystem as well, as someone might have reported it only there.

For details how to search and what to do if you find matching reports see ”Search for existing reports, first run” above.

Do not hurry with this step of the reporting process: spending 30 to 60 minutes or even more time can save you and others quite a lot of time and trouble.

8.3.12 Install a fresh kernel for testing

> *Unless you are already running the latest 'mainline' Linux kernel, better go and install it for the reporting process. Testing and reporting with the latest 'stable' Linux can be an acceptable alternative in some situations; during the merge window that actually might be even the best approach, but in that development phase it can be an even better idea to suspend your efforts for a few days anyway. Whatever version you choose, ideally use a 'vanilla' built. Ignoring these advices will dramatically increase the risk your report will be rejected or ignored.*

As mentioned in the detailed explanation for the first step already: Like most programmers, Linux kernel developers don't like to spend time dealing with reports for issues that don't even happen with the current code. It's just a waste everybody's time, especially yours. That's why it's in everybody's interest that you confirm the issue still exists with the latest upstream code before reporting it. You are free to ignore this advice, but as outlined earlier: doing so dramatically increases the risk that your issue report might get rejected or simply ignored.

In the scope of the kernel ”latest upstream” normally means:

- Install a mainline kernel; the latest stable kernel can be an option, but most of the time is better avoided. Longterm kernels (sometimes called 'LTS kernels') are unsuitable at this point of the process. The next subsection explains all of this in more detail.
- The over next subsection describes way to obtain and install such a kernel. It also outlines that using a pre-compiled kernel are fine, but better are vanilla, which means: it was built using Linux sources taken straight from kernel.org and not modified or enhanced in any way.

Choosing the right version for testing

Head over to kernel.org to find out which version you want to use for testing. Ignore the big yellow button that says 'Latest release' and look a little lower at the table. At its top you'll see a line starting with mainline, which most of the time will point to a pre-release with a version number like '5.8-rc2'. If that's the case, you'll want to use this mainline kernel for testing, as that where all fixes have to be applied first. Do not let that 'rc' scare you, these 'development kernels' are pretty reliable — and you made a backup, as you were instructed above, didn't you?

In about two out of every nine to ten weeks, mainline might point you to a proper release with a version number like '5.7'. If that happens, consider suspending the reporting process until the first pre-release of the next version (5.8-rc1) shows up on kernel.org. That's because the Linux

development cycle then is in its two-week long 'merge window'. The bulk of the changes and all intrusive ones get merged for the next release during this time. It's a bit more risky to use mainline during this period. Kernel developers are also often quite busy then and might have no spare time to deal with issue reports. It's also quite possible that one of the many changes applied during the merge window fixes the issue you face; that's why you soon would have to retest with a newer kernel version anyway, as outlined below in the section 'Duties after the report went out'.

That's why it might make sense to wait till the merge window is over. But don't to that if you're dealing with something that shouldn't wait. In that case consider obtaining the latest mainline kernel via git (see below) or use the latest stable version offered on kernel.org. Using that is also acceptable in case mainline for some reason does currently not work for you. An in general: using it for reproducing the issue is also better than not reporting it issue at all.

Better avoid using the latest stable kernel outside merge windows, as all fixes must be applied to mainline first. That's why checking the latest mainline kernel is so important: any issue you want to see fixed in older version lines needs to be fixed in mainline first before it can get backported, which can take a few days or weeks. Another reason: the fix you hope for might be too hard or risky for backporting; reporting the issue again hence is unlikely to change anything.

These aspects are also why longterm kernels (sometimes called ”LTS kernels”) are unsuitable for this part of the reporting process: they are to distant from the current code. Hence go and test mainline first and follow the process further: if the issue doesn't occur with mainline it will guide you how to get it fixed in older version lines, if that's in the cards for the fix in question.

How to obtain a fresh Linux kernel

Using a pre-compiled kernel: This is often the quickest, easiest, and safest way for testing — especially is you are unfamiliar with the Linux kernel. The problem: most of those shipped by distributors or add-on repositories are build from modified Linux sources. They are thus not vanilla and therefore often unsuitable for testing and issue reporting: the changes might cause the issue you face or influence it somehow.

But you are in luck if you are using a popular Linux distribution: for quite a few of them you'll find repositories on the net that contain packages with the latest mainline or stable Linux built as vanilla kernel. It's totally okay to use these, just make sure from the repository's description they are vanilla or at least close to it. Additionally ensure the packages contain the latest versions as offered on kernel.org. The packages are likely unsuitable if they are older than a week, as new mainline and stable kernels typically get released at least once a week.

Please note that you might need to build your own kernel manually later: that's sometimes needed for debugging or testing fixes, as described later in this document. Also be aware that pre-compiled kernels might lack debug symbols that are needed to decode messages the kernel prints when a panic, Oops, warning, or BUG occurs; if you plan to decode those, you might be better off compiling a kernel yourself (see the end of this subsection and the section titled 'Decode failure messages' for details).

Using git: Developers and experienced Linux users familiar with git are often best served by obtaining the latest Linux kernel sources straight from the official development repository on kernel.org. Those are likely a bit ahead of the latest mainline pre-release. Don't worry about it: they are as reliable as a proper pre-release, unless the kernel's development cycle is currently in the middle of a merge window. But even then they are quite reliable.

Conventional: People unfamiliar with git are often best served by downloading the sources as tarball from kernel.org.

How to actually build a kernel is not described here, as many websites explain the necessary steps already. If you are new to it, consider following one of those how-to's that suggest to use `make localmodconfig`, as that tries to pick up the configuration of your current kernel and then tries to adjust it somewhat for your system. That does not make the resulting kernel any better, but quicker to compile.

Note: If you are dealing with a panic, Oops, warning, or BUG from the kernel, please try to enable CONFIG_KALLSYMS when configuring your kernel. Additionally, enable CONFIG_DEBUG_KERNEL and CONFIG_DEBUG_INFO, too; the latter is the relevant one of those two, but can only be reached if you enable the former. Be aware CONFIG_DEBUG_INFO increases the storage space required to build a kernel by quite a bit. But that's worth it, as these options will allow you later to pinpoint the exact line of code that triggers your issue. The section 'Decode failure messages' below explains this in more detail.

But keep in mind: Always keep a record of the issue encountered in case it is hard to reproduce. Sending an undecoded report is better than not reporting the issue at all.

8.3.13 Check 'taint' flag

> *Ensure the kernel you just installed does not 'taint' itself when running.*

As outlined above in more detail already: the kernel sets a 'taint' flag when something happens that can lead to follow-up errors that look totally unrelated. That's why you need to check if the kernel you just installed does not set this flag. And if it does, you in almost all the cases needs to eliminate the reason for it before you reporting issues that occur with it. See the section above for details how to do that.

8.3.14 Reproduce issue with the fresh kernel

> *Reproduce the issue with the kernel you just installed. If it doesn't show up there, scroll down to the instructions for issues only happening with stable and longterm kernels.*

Check if the issue occurs with the fresh Linux kernel version you just installed. If it was fixed there already, consider sticking with this version line and abandoning your plan to report the issue. But keep in mind that other users might still be plagued by it, as long as it's not fixed in either stable and longterm version from kernel.org (and thus vendor kernels derived from those). If you prefer to use one of those or just want to help their users, head over to the section ”Details about reporting issues only occurring in older kernel version lines” below.

8.3.15 Optimize description to reproduce issue

> *Optimize your notes: try to find and write the most straightforward way to reproduce your issue. Make sure the end result has all the important details, and at the same time is easy to read and understand for others that hear about it for the first time. And if you learned something in this process, consider searching again for existing reports about the issue.*

An unnecessarily complex report will make it hard for others to understand your report. Thus try to find a reproducer that's straight forward to describe and thus easy to understand in written form. Include all important details, but at the same time try to keep it as short as possible.

In this in the previous steps you likely have learned a thing or two about the issue you face. Use this knowledge and search again for existing reports instead you can join.

8.3.16 Decode failure messages

> *If your failure involves a 'panic', 'Oops', 'warning', or 'BUG', consider decoding the kernel log to find the line of code that triggered the error.*

When the kernel detects an internal problem, it will log some information about the executed code. This makes it possible to pinpoint the exact line in the source code that triggered the issue and shows how it was called. But that only works if you enabled CONFIG_DEBUG_INFO and CONFIG_KALLSYMS when configuring your kernel. If you did so, consider to decode the information from the kernel's log. That will make it a lot easier to understand what lead to the 'panic', 'Oops', 'warning', or 'BUG', which increases the chances that someone can provide a fix.

Decoding can be done with a script you find in the Linux source tree. If you are running a kernel you compiled yourself earlier, call it like this:

```
[user@something ~]$ sudo dmesg | ./linux-5.10.5/scripts/decode_stacktrace.sh ./
↪linux-5.10.5/vmlinux
```

If you are running a packaged vanilla kernel, you will likely have to install the corresponding packages with debug symbols. Then call the script (which you might need to get from the Linux sources if your distro does not package it) like this:

```
[user@something ~]$ sudo dmesg | ./linux-5.10.5/scripts/decode_stacktrace.sh \
 /usr/lib/debug/lib/modules/5.10.10-4.1.x86_64/vmlinux /usr/src/kernels/5.10.
↪10-4.1.x86_64/
```

The script will work on log lines like the following, which show the address of the code the kernel was executing when the error occurred:

```
[   68.387301] RIP: 0010:test_module_init+0x5/0xffa [test_module]
```

Once decoded, these lines will look like this:

```
[   68.387301] RIP: 0010:test_module_init (/home/username/linux-5.10.5/test-
↪module/test-module.c:16) test_module
```

In this case the executed code was built from the file '~/linux-5.10.5/test-module/test-module.c' and the error occurred by the instructions found in line '16'.

The script will similarly decode the addresses mentioned in the section starting with 'Call trace', which show the path to the function where the problem occurred. Additionally, the script will show the assembler output for the code section the kernel was executing.

Note, if you can't get this to work, simply skip this step and mention the reason for it in the report. If you're lucky, it might not be needed. And if it is, someone might help you to get things going. Also be aware this is just one of several ways to decode kernel stack traces. Sometimes different steps will be required to retrieve the relevant details. Don't worry about that, if that's needed in your case, developers will tell you what to do.

8.3.17 Special care for regressions

> *If your problem is a regression, try to narrow down when the issue was introduced as much as possible.*

Linux lead developer Linus Torvalds insists that the Linux kernel never worsens, that's why he deems regressions as unacceptable and wants to see them fixed quickly. That's why changes that introduced a regression are often promptly reverted if the issue they cause can't get solved quickly any other way. Reporting a regression is thus a bit like playing a kind of trump card to get something quickly fixed. But for that to happen the change that's causing the regression needs to be known. Normally it's up to the reporter to track down the culprit, as maintainers often won't have the time or setup at hand to reproduce it themselves.

To find the change there is a process called 'bisection' which the document *Bisecting a bug* describes in detail. That process will often require you to build about ten to twenty kernel images, trying to reproduce the issue with each of them before building the next. Yes, that takes some time, but don't worry, it works a lot quicker than most people assume. Thanks to a 'binary search' this will lead you to the one commit in the source code management system that's causing the regression. Once you find it, search the net for the subject of the change, its commit id and the shortened commit id (the first 12 characters of the commit id). This will lead you to existing reports about it, if there are any.

Note, a bisection needs a bit of know-how, which not everyone has, and quite a bit of effort, which not everyone is willing to invest. Nevertheless, it's highly recommended performing a bisection yourself. If you really can't or don't want to go down that route at least find out which mainline kernel introduced the regression. If something for example breaks when switching from 5.5.15 to 5.8.4, then try at least all the mainline releases in that area (5.6, 5.7 and 5.8) to check when it first showed up. Unless you're trying to find a regression in a stable or longterm kernel, avoid testing versions which number has three sections (5.6.12, 5.7.8), as that makes the outcome hard to interpret, which might render your testing useless. Once you found the major version which introduced the regression, feel free to move on in the reporting process. But keep in mind: it depends on the issue at hand if the developers will be able to help without knowing the culprit. Sometimes they might recognize from the report want went wrong and can fix it; other times they will be unable to help unless you perform a bisection.

When dealing with regressions make sure the issue you face is really caused by the kernel and not by something else, as outlined above already.

In the whole process keep in mind: an issue only qualifies as regression if the older and the newer kernel got built with a similar configuration. This can be achieved by using `make olddefconfig`, as explained in more detail by *Reporting regressions*; that document also provides a good deal of other information about regressions you might want to be aware of.

8.3.18 Write and send the report

> *Start to compile the report by writing a detailed description about the issue. Always mention a few things: the latest kernel version you installed for reproducing, the Linux Distribution used, and your notes on how to reproduce the issue. Ideally, make the kernel's build configuration (.config) and the output from ``dmesg`` available somewhere on the net and link to it. Include or upload all other information that might be relevant, like the output/screenshot of an Oops or the output from ``lspci``. Once you wrote this main part, insert a normal length paragraph on top of it outlining the issue and the impact quickly. On top of this add one sentence that briefly describes the problem and gets people to read on. Now give the thing a descriptive title or subject that yet again is shorter. Then you're ready to send or file the report like the MAINTAINERS file told you, unless you are dealing with one of those 'issues of high priority': they need special care which is explained in 'Special handling for high priority issues' below.*

Now that you have prepared everything it's time to write your report. How to do that is partly explained by the three documents linked to in the preface above. That's why this text will only mention a few of the essentials as well as things specific to the Linux kernel.

There is one thing that fits both categories: the most crucial parts of your report are the title/subject, the first sentence, and the first paragraph. Developers often get quite a lot of mail. They thus often just take a few seconds to skim a mail before deciding to move on or look closer. Thus: the better the top section of your report, the higher are the chances that someone will look into it and help you. And that is why you should ignore them for now and write the detailed report first. ;-)

Things each report should mention

Describe in detail how your issue happens with the fresh vanilla kernel you installed. Try to include the step-by-step instructions you wrote and optimized earlier that outline how you and ideally others can reproduce the issue; in those rare cases where that's impossible try to describe what you did to trigger it.

Also include all the relevant information others might need to understand the issue and its environment. What's actually needed depends a lot on the issue, but there are some things you should include always:

- the output from `cat /proc/version`, which contains the Linux kernel version number and the compiler it was built with.
- the Linux distribution the machine is running (`hostnamectl | grep "Operating System"`)
- the architecture of the CPU and the operating system (`uname -mi`)
- if you are dealing with a regression and performed a bisection, mention the subject and the commit-id of the change that is causing it.

In a lot of cases it's also wise to make two more things available to those that read your report:

- the configuration used for building your Linux kernel (the '.config' file)
- the kernel's messages that you get from `dmesg` written to a file. Make sure that it starts with a line like 'Linux version 5.8-1 (foobar@example.com) (gcc (GCC) 10.2.1, GNU ld

version 2.34) #1 SMP Mon Aug 3 14:54:37 UTC 2020' If it's missing, then important messages from the first boot phase already got discarded. In this case instead consider using `journalctl -b 0 -k`; alternatively you can also reboot, reproduce the issue and call `dmesg` right afterwards.

These two files are big, that's why it's a bad idea to put them directly into your report. If you are filing the issue in a bug tracker then attach them to the ticket. If you report the issue by mail do not attach them, as that makes the mail too large; instead do one of these things:

- Upload the files somewhere public (your website, a public file paste service, a ticket created just for this purpose on bugzilla.kernel.org, ...) and include a link to them in your report. Ideally use something where the files stay available for years, as they could be useful to someone many years from now; this for example can happen if five or ten years from now a developer works on some code that was changed just to fix your issue.
- Put the files aside and mention you will send them later in individual replies to your own mail. Just remember to actually do that once the report went out. ;-)

Things that might be wise to provide

Depending on the issue you might need to add more background data. Here are a few suggestions what often is good to provide:

- If you are dealing with a 'warning', an 'OOPS' or a 'panic' from the kernel, include it. If you can't copy'n'paste it, try to capture a netconsole trace or at least take a picture of the screen.
- If the issue might be related to your computer hardware, mention what kind of system you use. If you for example have problems with your graphics card, mention its manufacturer, the card's model, and what chip is uses. If it's a laptop mention its name, but try to make sure it's meaningful. 'Dell XPS 13' for example is not, because it might be the one from 2012; that one looks not that different from the one sold today, but apart from that the two have nothing in common. Hence, in such cases add the exact model number, which for example are '9380' or '7390' for XPS 13 models introduced during 2019. Names like 'Lenovo Thinkpad T590' are also somewhat ambiguous: there are variants of this laptop with and without a dedicated graphics chip, so try to find the exact model name or specify the main components.
- Mention the relevant software in use. If you have problems with loading modules, you want to mention the versions of kmod, systemd, and udev in use. If one of the DRM drivers misbehaves, you want to state the versions of libdrm and Mesa; also specify your Wayland compositor or the X-Server and its driver. If you have a filesystem issue, mention the version of corresponding filesystem utilities (e2fsprogs, btrfs-progs, xfsprogs, ...).
- Gather additional information from the kernel that might be of interest. The output from `lspci -nn` will for example help others to identify what hardware you use. If you have a problem with hardware you even might want to make the output from `sudo lspci -vvv` available, as that provides insights how the components were configured. For some issues it might be good to include the contents of files like `/proc/cpuinfo`, `/proc/ioports`, `/proc/iomem`, `/proc/modules`, or `/proc/scsi/scsi`. Some subsystem also offer tools to collect relevant information. One such tool is `alsa-info.sh` which the audio/sound subsystem developers provide.

Those examples should give your some ideas of what data might be wise to attach, but you have to think yourself what will be helpful for others to know. Don't worry too much about forget-

ting something, as developers will ask for additional details they need. But making everything important available from the start increases the chance someone will take a closer look.

The important part: the head of your report

Now that you have the detailed part of the report prepared let's get to the most important section: the first few sentences. Thus go to the top, add something like 'The detailed description:' before the part you just wrote and insert two newlines at the top. Now write one normal length paragraph that describes the issue roughly. Leave out all boring details and focus on the crucial parts readers need to know to understand what this is all about; if you think this bug affects a lot of users, mention this to get people interested.

Once you did that insert two more lines at the top and write a one sentence summary that explains quickly what the report is about. After that you have to get even more abstract and write an even shorter subject/title for the report.

Now that you have written this part take some time to optimize it, as it is the most important parts of your report: a lot of people will only read this before they decide if reading the rest is time well spent.

Now send or file the report like the MAINTAINERS file told you, unless it's one of those 'issues of high priority' outlined earlier: in that case please read the next subsection first before sending the report on its way.

Special handling for high priority issues

Reports for high priority issues need special handling.

Severe issues: make sure the subject or ticket title as well as the first paragraph makes the severeness obvious.

Regressions: make the report's subject start with '[REGRESSION]'.

In case you performed a successful bisection, use the title of the change that introduced the regression as the second part of your subject. Make the report also mention the commit id of the culprit. In case of an unsuccessful bisection, make your report mention the latest tested version that's working fine (say 5.7) and the oldest where the issue occurs (say 5.8-rc1).

When sending the report by mail, CC the Linux regressions mailing list (regressions@lists.linux.dev). In case the report needs to be filed to some web tracker, proceed to do so. Once filed, forward the report by mail to the regressions list; CC the maintainer and the mailing list for the subsystem in question. Make sure to inline the forwarded report, hence do not attach it. Also add a short note at the top where you mention the URL to the ticket.

When mailing or forwarding the report, in case of a successful bisection add the author of the culprit to the recipients; also CC everyone in the signed-off-by chain, which you find at the end of its commit message.

Security issues: for these issues your will have to evaluate if a short-term risk to other users would arise if details were publicly disclosed. If that's not the case simply proceed with reporting the issue as described. For issues that bear such a risk you will need to adjust the reporting process slightly:

- If the MAINTAINERS file instructed you to report the issue by mail, do not CC any public mailing lists.

- If you were supposed to file the issue in a bug tracker make sure to mark the ticket as 'private' or 'security issue'. If the bug tracker does not offer a way to keep reports private, forget about it and send your report as a private mail to the maintainers instead.

In both cases make sure to also mail your report to the addresses the MAINTAINERS file lists in the section 'security contact'. Ideally directly CC them when sending the report by mail. If you filed it in a bug tracker, forward the report's text to these addresses; but on top of it put a small note where you mention that you filed it with a link to the ticket.

See Documentation/process/security-bugs.rst for more information.

8.3.19 Duties after the report went out

> *Wait for reactions and keep the thing rolling until you can accept the outcome in one way or the other. Thus react publicly and in a timely manner to any inquiries. Test proposed fixes. Do proactive testing: retest with at least every first release candidate (RC) of a new mainline version and report your results. Send friendly reminders if things stall. And try to help yourself, if you don't get any help or if it's unsatisfying.*

If your report was good and you are really lucky then one of the developers might immediately spot what's causing the issue; they then might write a patch to fix it, test it, and send it straight for integration in mainline while tagging it for later backport to stable and longterm kernels that need it. Then all you need to do is reply with a 'Thank you very much' and switch to a version with the fix once it gets released.

But this ideal scenario rarely happens. That's why the job is only starting once you got the report out. What you'll have to do depends on the situations, but often it will be the things listed below. But before digging into the details, here are a few important things you need to keep in mind for this part of the process.

General advice for further interactions

Always reply in public: When you filed the issue in a bug tracker, always reply there and do not contact any of the developers privately about it. For mailed reports always use the 'Reply-all' function when replying to any mails you receive. That includes mails with any additional data you might want to add to your report: go to your mail applications 'Sent' folder and use 'reply-all' on your mail with the report. This approach will make sure the public mailing list(s) and everyone else that gets involved over time stays in the loop; it also keeps the mail thread intact, which among others is really important for mailing lists to group all related mails together.

There are just two situations where a comment in a bug tracker or a 'Reply-all' is unsuitable:

- Someone tells you to send something privately.
- You were told to send something, but noticed it contains sensitive information that needs to be kept private. In that case it's okay to send it in private to the developer that asked for it. But note in the ticket or a mail that you did that, so everyone else knows you honored the request.

Do research before asking for clarifications or help: In this part of the process someone might tell you to do something that requires a skill you might not have mastered yet. For example, you might be asked to use some test tools you never have heard of yet; or you might be asked to apply a patch to the Linux kernel sources to test if it helps. In some cases it will be fine sending a reply asking for instructions how to do that. But before going that route try to

find the answer own your own by searching the internet; alternatively consider asking in other places for advice. For example ask a friend or post about it to a chatroom or forum you normally hang out.

Be patient: If you are really lucky you might get a reply to your report within a few hours. But most of the time it will take longer, as maintainers are scattered around the globe and thus might be in a different time zone – one where they already enjoy their night away from keyboard.

In general, kernel developers will take one to five business days to respond to reports. Sometimes it will take longer, as they might be busy with the merge windows, other work, visiting developer conferences, or simply enjoying a long summer holiday.

The 'issues of high priority' (see above for an explanation) are an exception here: maintainers should address them as soon as possible; that's why you should wait a week at maximum (or just two days if it's something urgent) before sending a friendly reminder.

Sometimes the maintainer might not be responding in a timely manner; other times there might be disagreements, for example if an issue qualifies as regression or not. In such cases raise your concerns on the mailing list and ask others for public or private replies how to move on. If that fails, it might be appropriate to get a higher authority involved. In case of a WiFi driver that would be the wireless maintainers; if there are no higher level maintainers or all else fails, it might be one of those rare situations where it's okay to get Linus Torvalds involved.

Proactive testing: Every time the first pre-release (the 'rc1') of a new mainline kernel version gets released, go and check if the issue is fixed there or if anything of importance changed. Mention the outcome in the ticket or in a mail you sent as reply to your report (make sure it has all those in the CC that up to that point participated in the discussion). This will show your commitment and that you are willing to help. It also tells developers if the issue persists and makes sure they do not forget about it. A few other occasional retests (for example with rc3, rc5 and the final) are also a good idea, but only report your results if something relevant changed or if you are writing something anyway.

With all these general things off the table let's get into the details of how to help to get issues resolved once they were reported.

Inquires and testing request

Here are your duties in case you got replies to your report:

Check who you deal with: Most of the time it will be the maintainer or a developer of the particular code area that will respond to your report. But as issues are normally reported in public it could be anyone that's replying — including people that want to help, but in the end might guide you totally off track with their questions or requests. That rarely happens, but it's one of many reasons why it's wise to quickly run an internet search to see who you're interacting with. By doing this you also get aware if your report was heard by the right people, as a reminder to the maintainer (see below) might be in order later if discussion fades out without leading to a satisfying solution for the issue.

Inquiries for data: Often you will be asked to test something or provide additional details. Try to provide the requested information soon, as you have the attention of someone that might help and risk losing it the longer you wait; that outcome is even likely if you do not provide the information within a few business days.

Requests for testing: When you are asked to test a diagnostic patch or a possible fix, try to test it in timely manner, too. But do it properly and make sure to not rush it: mixing things up can happen easily and can lead to a lot of confusion for everyone involved. A common mistake for example is thinking a proposed patch with a fix was applied, but in fact wasn't. Things like that happen even to experienced testers occasionally, but they most of the time will notice when the kernel with the fix behaves just as one without it.

What to do when nothing of substance happens

Some reports will not get any reaction from the responsible Linux kernel developers; or a discussion around the issue evolved, but faded out with nothing of substance coming out of it.

In these cases wait two (better: three) weeks before sending a friendly reminder: maybe the maintainer was just away from keyboard for a while when your report arrived or had something more important to take care of. When writing the reminder, kindly ask if anything else from your side is needed to get the ball running somehow. If the report got out by mail, do that in the first lines of a mail that is a reply to your initial mail (see above) which includes a full quote of the original report below: that's on of those few situations where such a 'TOFU' (Text Over, Fullquote Under) is the right approach, as then all the recipients will have the details at hand immediately in the proper order.

After the reminder wait three more weeks for replies. If you still don't get a proper reaction, you first should reconsider your approach. Did you maybe try to reach out to the wrong people? Was the report maybe offensive or so confusing that people decided to completely stay away from it? The best way to rule out such factors: show the report to one or two people familiar with FLOSS issue reporting and ask for their opinion. Also ask them for their advice how to move forward. That might mean: prepare a better report and make those people review it before you send it out. Such an approach is totally fine; just mention that this is the second and improved report on the issue and include a link to the first report.

If the report was proper you can send a second reminder; in it ask for advice why the report did not get any replies. A good moment for this second reminder mail is shortly after the first pre-release (the 'rc1') of a new Linux kernel version got published, as you should retest and provide a status update at that point anyway (see above).

If the second reminder again results in no reaction within a week, try to contact a higher-level maintainer asking for advice: even busy maintainers by then should at least have sent some kind of acknowledgment.

Remember to prepare yourself for a disappointment: maintainers ideally should react somehow to every issue report, but they are only obliged to fix those 'issues of high priority' outlined earlier. So don't be too devastating if you get a reply along the lines of 'thanks for the report, I have more important issues to deal with currently and won't have time to look into this for the foreseeable future'.

It's also possible that after some discussion in the bug tracker or on a list nothing happens anymore and reminders don't help to motivate anyone to work out a fix. Such situations can be devastating, but is within the cards when it comes to Linux kernel development. This and several other reasons for not getting help are explained in 'Why some issues won't get any reaction or remain unfixed after being reported' near the end of this document.

Don't get devastated if you don't find any help or if the issue in the end does not get solved: the Linux kernel is FLOSS and thus you can still help yourself. You for example could try to find others that are affected and team up with them to get the issue resolved. Such a team could

prepare a fresh report together that mentions how many you are and why this is something that in your option should get fixed. Maybe together you can also narrow down the root cause or the change that introduced a regression, which often makes developing a fix easier. And with a bit of luck there might be someone in the team that knows a bit about programming and might be able to write a fix.

8.3.20 Reference for "Reporting regressions within a stable and longterm kernel line"

This subsection provides details for the steps you need to perform if you face a regression within a stable and longterm kernel line.

Make sure the particular version line still gets support

> *Check if the kernel developers still maintain the Linux kernel version line you care about: go to the front page of kernel.org and make sure it mentions the latest release of the particular version line without an '[EOL]' tag.*

Most kernel version lines only get supported for about three months, as maintaining them longer is quite a lot of work. Hence, only one per year is chosen and gets supported for at least two years (often six). That's why you need to check if the kernel developers still support the version line you care for.

Note, if kernel.org lists two stable version lines on the front page, you should consider switching to the newer one and forget about the older one: support for it is likely to be abandoned soon. Then it will get a "end-of-life" (EOL) stamp. Version lines that reached that point still get mentioned on the kernel.org front page for a week or two, but are unsuitable for testing and reporting.

Search stable mailing list

> *Check the archives of the Linux stable mailing list for existing reports.*

Maybe the issue you face is already known and was fixed or is about to. Hence, search the archives of the Linux stable mailing list for reports about an issue like yours. If you find any matches, consider joining the discussion, unless the fix is already finished and scheduled to get applied soon.

Reproduce issue with the newest release

> *Install the latest release from the particular version line as a vanilla kernel. Ensure this kernel is not tainted and still shows the problem, as the issue might have already been fixed there. If you first noticed the problem with a vendor kernel, check a vanilla build of the last version known to work performs fine as well.*

Before investing any more time in this process you want to check if the issue was already fixed in the latest release of version line you're interested in. This kernel needs to be vanilla and shouldn't be tainted before the issue happens, as detailed outlined already above in the section "Install a fresh kernel for testing".

Did you first notice the regression with a vendor kernel? Then changes the vendor applied might be interfering. You need to rule that out by performing a recheck. Say something broke when you updated from 5.10.4-vendor.42 to 5.10.5-vendor.43. Then after testing the latest 5.10 release as outlined in the previous paragraph check if a vanilla build of Linux 5.10.4 works fine as well. If things are broken there, the issue does not qualify as upstream regression and you need switch back to the main step-by-step guide to report the issue.

Report the regression

> *Send a short problem report to the Linux stable mailing list (stable@vger.kernel.org) and CC the Linux regressions mailing list (regressions@lists.linux.dev); if you suspect the cause in a particular subsystem, CC its maintainer and its mailing list. Roughly describe the issue and ideally explain how to reproduce it. Mention the first version that shows the problem and the last version that's working fine. Then wait for further instructions.*

When reporting a regression that happens within a stable or longterm kernel line (say when updating from 5.10.4 to 5.10.5) a brief report is enough for the start to get the issue reported quickly. Hence a rough description to the stable and regressions mailing list is all it takes; but in case you suspect the cause in a particular subsystem, CC its maintainers and its mailing list as well, because that will speed things up.

And note, it helps developers a great deal if you can specify the exact version that introduced the problem. Hence if possible within a reasonable time frame, try to find that version using vanilla kernels. Lets assume something broke when your distributor released a update from Linux kernel 5.10.5 to 5.10.8. Then as instructed above go and check the latest kernel from that version line, say 5.10.9. If it shows the problem, try a vanilla 5.10.5 to ensure that no patches the distributor applied interfere. If the issue doesn't manifest itself there, try 5.10.7 and then (depending on the outcome) 5.10.8 or 5.10.6 to find the first version where things broke. Mention it in the report and state that 5.10.9 is still broken.

What the previous paragraph outlines is basically a rough manual 'bisection'. Once your report is out your might get asked to do a proper one, as it allows to pinpoint the exact change that causes the issue (which then can easily get reverted to fix the issue quickly). Hence consider to do a proper bisection right away if time permits. See the section 'Special care for regressions' and the document *Bisecting a bug* for details how to perform one. In case of a successful bisection add the author of the culprit to the recipients; also CC everyone in the signed-off-by chain, which you find at the end of its commit message.

8.3.21 Reference for ”Reporting issues only occurring in older kernel version lines”

This section provides details for the steps you need to take if you could not reproduce your issue with a mainline kernel, but want to see it fixed in older version lines (aka stable and longterm kernels).

Some fixes are too complex

> *Prepare yourself for the possibility that going through the next few steps might not get the issue solved in older releases: the fix might be too big or risky to get backported there.*

Even small and seemingly obvious code-changes sometimes introduce new and totally unexpected problems. The maintainers of the stable and longterm kernels are very aware of that and thus only apply changes to these kernels that are within rules outlined in Documentation/process/stable-kernel-rules.rst.

Complex or risky changes for example do not qualify and thus only get applied to mainline. Other fixes are easy to get backported to the newest stable and longterm kernels, but too risky to integrate into older ones. So be aware the fix you are hoping for might be one of those that won't be backported to the version line your care about. In that case you'll have no other choice then to live with the issue or switch to a newer Linux version, unless you want to patch the fix into your kernels yourself.

Common preparations

> *Perform the first three steps in the section "Reporting issues only occurring in older kernel version lines" above.*

You need to carry out a few steps already described in another section of this guide. Those steps will let you:

- Check if the kernel developers still maintain the Linux kernel version line you care about.
- Search the Linux stable mailing list for exiting reports.
- Check with the latest release.

Check code history and search for existing discussions

> *Search the Linux kernel version control system for the change that fixed the issue in mainline, as its commit message might tell you if the fix is scheduled for backporting already. If you don't find anything that way, search the appropriate mailing lists for posts that discuss such an issue or peer-review possible fixes; then check the discussions if the fix was deemed unsuitable for backporting. If backporting was not considered at all, join the newest discussion, asking if it's in the cards.*

In a lot of cases the issue you deal with will have happened with mainline, but got fixed there. The commit that fixed it would need to get backported as well to get the issue solved. That's why you want to search for it or any discussions abound it.

- First try to find the fix in the Git repository that holds the Linux kernel sources. You can do this with the web interfaces on kernel.org or its mirror on GitHub; if you have a local clone you alternatively can search on the command line with `git log --grep=<pattern>`.

  If you find the fix, look if the commit message near the end contains a 'stable tag' that looks like this:

  > Cc: <stable@vger.kernel.org> # 5.4+

If that's case the developer marked the fix safe for backporting to version line 5.4 and later. Most of the time it's getting applied there within two weeks, but sometimes it takes a bit longer.

- If the commit doesn't tell you anything or if you can't find the fix, look again for discussions about the issue. Search the net with your favorite internet search engine as well as the archives for the Linux kernel developers mailing list. Also read the section *Locate kernel area that causes the issue* above and follow the instructions to find the subsystem in question: its bug tracker or mailing list archive might have the answer you are looking for.
- If you see a proposed fix, search for it in the version control system as outlined above, as the commit might tell you if a backport can be expected.
  - Check the discussions for any indicators the fix might be too risky to get backported to the version line you care about. If that's the case you have to live with the issue or switch to the kernel version line where the fix got applied.
  - If the fix doesn't contain a stable tag and backporting was not discussed, join the discussion: mention the version where you face the issue and that you would like to see it fixed, if suitable.

Ask for advice

One of the former steps should lead to a solution. If that doesn't work out, ask the maintainers for the subsystem that seems to be causing the issue for advice; CC the mailing list for the particular subsystem as well as the stable mailing list.

If the previous three steps didn't get you closer to a solution there is only one option left: ask for advice. Do that in a mail you sent to the maintainers for the subsystem where the issue seems to have its roots; CC the mailing list for the subsystem as well as the stable mailing list (stable@vger.kernel.org).

8.4 Why some issues won't get any reaction or remain unfixed after being reported

When reporting a problem to the Linux developers, be aware only 'issues of high priority' (regressions, security issues, severe problems) are definitely going to get resolved. The maintainers or if all else fails Linus Torvalds himself will make sure of that. They and the other kernel developers will fix a lot of other issues as well. But be aware that sometimes they can't or won't help; and sometimes there isn't even anyone to send a report to.

This is best explained with kernel developers that contribute to the Linux kernel in their spare time. Quite a few of the drivers in the kernel were written by such programmers, often because they simply wanted to make their hardware usable on their favorite operating system.

These programmers most of the time will happily fix problems other people report. But nobody can force them to do, as they are contributing voluntarily.

Then there are situations where such developers really want to fix an issue, but can't: sometimes they lack hardware programming documentation to do so. This often happens when the publicly available docs are superficial or the driver was written with the help of reverse engineering.

Sooner or later spare time developers will also stop caring for the driver. Maybe their test hardware broke, got replaced by something more fancy, or is so old that it's something you don't find much outside of computer museums anymore. Sometimes developer stops caring for their code and Linux at all, as something different in their life became way more important. In some cases nobody is willing to take over the job as maintainer – and nobody can be forced to, as contributing to the Linux kernel is done on a voluntary basis. Abandoned drivers nevertheless remain in the kernel: they are still useful for people and removing would be a regression.

The situation is not that different with developers that are paid for their work on the Linux kernel. Those contribute most changes these days. But their employers sooner or later also stop caring for their code or make its programmer focus on other things. Hardware vendors for example earn their money mainly by selling new hardware; quite a few of them hence are not investing much time and energy in maintaining a Linux kernel driver for something they stopped selling years ago. Enterprise Linux distributors often care for a longer time period, but in new versions often leave support for old and rare hardware aside to limit the scope. Often spare time contributors take over once a company orphans some code, but as mentioned above: sooner or later they will leave the code behind, too.

Priorities are another reason why some issues are not fixed, as maintainers quite often are forced to set those, as time to work on Linux is limited. That's true for spare time or the time employers grant their developers to spend on maintenance work on the upstream kernel. Sometimes maintainers also get overwhelmed with reports, even if a driver is working nearly perfectly. To not get completely stuck, the programmer thus might have no other choice than to prioritize issue reports and reject some of them.

But don't worry too much about all of this, a lot of drivers have active maintainers who are quite interested in fixing as many issues as possible.

8.5 Closing words

Compared with other Free/Libre & Open Source Software it's hard to report issues to the Linux kernel developers: the length and complexity of this document and the implications between the lines illustrate that. But that's how it is for now. The main author of this text hopes documenting the state of the art will lay some groundwork to improve the situation over time.

CHAPTER

NINE

REPORTING REGRESSIONS

”*We don't cause regressions*” is the first rule of Linux kernel development; Linux founder and lead developer Linus Torvalds established it himself and ensures it's obeyed.

This document describes what the rule means for users and how the Linux kernel's development model ensures to address all reported regressions; aspects relevant for kernel developers are left to Documentation/process/handling-regressions.rst.

9.1 The important bits (aka ”TL;DR”)

1. It's a regression if something running fine with one Linux kernel works worse or not at all with a newer version. Note, the newer kernel has to be compiled using a similar configuration; the detailed explanations below describes this and other fine print in more detail.
2. Report your issue as outlined in *Reporting issues*, it already covers all aspects important for regressions and repeated below for convenience. Two of them are important: start your report's subject with ”[REGRESSION]” and CC or forward it to the regression mailing list (regressions@lists.linux.dev).
3. Optional, but recommended: when sending or forwarding your report, make the Linux kernel regression tracking bot ”regzbot” track the issue by specifying when the regression started like this:

```
#regzbot introduced v5.13..v5.14-rc1
```

9.2 All the details on Linux kernel regressions relevant for users

9.2.1 The important basics

What is a ”regression” and what is the ”no regressions rule”?

It's a regression if some application or practical use case running fine with one Linux kernel works worse or not at all with a newer version compiled using a similar configuration. The ”no regressions rule” forbids this to take place; if it happens by accident, developers that caused it are expected to quickly fix the issue.

It thus is a regression when a WiFi driver from Linux 5.13 works fine, but with 5.14 doesn't work at all, works significantly slower, or misbehaves somehow. It's also a regression if a perfectly working application suddenly shows erratic behavior with a newer kernel version; such issues

can be caused by changes in procfs, sysfs, or one of the many other interfaces Linux provides to userland software. But keep in mind, as mentioned earlier: 5.14 in this example needs to be built from a configuration similar to the one from 5.13. This can be achieved using `make olddefconfig`, as explained in more detail below.

Note the "practical use case" in the first sentence of this section: developers despite the "no regressions" rule are free to change any aspect of the kernel and even APIs or ABIs to userland, as long as no existing application or use case breaks.

Also be aware the "no regressions" rule covers only interfaces the kernel provides to the userland. It thus does not apply to kernel-internal interfaces like the module API, which some externally developed drivers use to hook into the kernel.

How do I report a regression?

Just report the issue as outlined in *Reporting issues*, it already describes the important points. The following aspects outlined there are especially relevant for regressions:

- When checking for existing reports to join, also search the archives of the Linux regressions mailing list and regzbot's web-interface.
- Start your report's subject with "[REGRESSION]".
- In your report, clearly mention the last kernel version that worked fine and the first broken one. Ideally try to find the exact change causing the regression using a bisection, as explained below in more detail.
- Remember to let the Linux regressions mailing list (regressions@lists.linux.dev) know about your report:
  - If you report the regression by mail, CC the regressions list.
  - If you report your regression to some bug tracker, forward the submitted report by mail to the regressions list while CCing the maintainer and the mailing list for the subsystem in question.

  If it's a regression within a stable or longterm series (e.g. v5.15.3..v5.15.5), remember to CC the Linux stable mailing list (stable@vger.kernel.org).

  In case you performed a successful bisection, add everyone to the CC the culprit's commit message mentions in lines starting with "Signed-off-by:".

When CCing for forwarding your report to the list, consider directly telling the aforementioned Linux kernel regression tracking bot about your report. To do that, include a paragraph like this in your mail:

```
#regzbot introduced: v5.13..v5.14-rc1
```

Regzbot will then consider your mail a report for a regression introduced in the specified version range. In above case Linux v5.13 still worked fine and Linux v5.14-rc1 was the first version where you encountered the issue. If you performed a bisection to find the commit that caused the regression, specify the culprit's commit-id instead:

```
#regzbot introduced: 1f2e3d4c5d
```

Placing such a "regzbot command" is in your interest, as it will ensure the report won't fall through the cracks unnoticed. If you omit this, the Linux kernel's regressions tracker will take

care of telling regzbot about your regression, as long as you send a copy to the regressions mailing lists. But the regression tracker is just one human which sometimes has to rest or occasionally might even enjoy some time away from computers (as crazy as that might sound). Relying on this person thus will result in an unnecessary delay before the regressions becomes mentioned on the list of tracked and unresolved Linux kernel regressions and the weekly regression reports sent by regzbot. Such delays can result in Linus Torvalds being unaware of important regressions when deciding between ”continue development or call this finished and release the final?”.

Are really all regressions fixed?

Nearly all of them are, as long as the change causing the regression (the ”culprit commit”) is reliably identified. Some regressions can be fixed without this, but often it's required.

Who needs to find the root cause of a regression?

Developers of the affected code area should try to locate the culprit on their own. But for them that's often impossible to do with reasonable effort, as quite a lot of issues only occur in a particular environment outside the developer's reach -- for example, a specific hardware platform, firmware, Linux distro, system's configuration, or application. That's why in the end it's often up to the reporter to locate the culprit commit; sometimes users might even need to run additional tests afterwards to pinpoint the exact root cause. Developers should offer advice and reasonably help where they can, to make this process relatively easy and achievable for typical users.

How can I find the culprit?

Perform a bisection, as roughly outlined in *Reporting issues* and described in more detail by *Bisecting a bug*. It might sound like a lot of work, but in many cases finds the culprit relatively quickly. If it's hard or time-consuming to reliably reproduce the issue, consider teaming up with other affected users to narrow down the search range together.

Who can I ask for advice when it comes to regressions?

Send a mail to the regressions mailing list (regressions@lists.linux.dev) while CCing the Linux kernel's regression tracker (regressions@leemhuis.info); if the issue might better be dealt with in private, feel free to omit the list.

9.2.2 Additional details about regressions

What is the goal of the ”no regressions rule”?

Users should feel safe when updating kernel versions and not have to worry something might break. This is in the interest of the kernel developers to make updating attractive: they don't want users to stay on stable or longterm Linux series that are either abandoned or more than one and a half years old. That's in everybody's interest, as those series might have known bugs, security issues, or other problematic aspects already fixed in later versions. Additionally, the kernel developers want to make it simple and appealing for users to test the latest pre-release

or regular release. That's also in everybody's interest, as it's a lot easier to track down and fix problems, if they are reported shortly after being introduced.

Is the ”no regressions” rule really adhered in practice?

It's taken really seriously, as can be seen by many mailing list posts from Linux creator and lead developer Linus Torvalds, some of which are quoted in Documentation/process/handling-regressions.rst.

Exceptions to this rule are extremely rare; in the past developers almost always turned out to be wrong when they assumed a particular situation was warranting an exception.

Who ensures the ”no regressions” is actually followed?

The subsystem maintainers should take care of that, which are watched and supported by the tree maintainers -- e.g. Linus Torvalds for mainline and Greg Kroah-Hartman et al. for various stable/longterm series.

All of them are helped by people trying to ensure no regression report falls through the cracks. One of them is Thorsten Leemhuis, who's currently acting as the Linux kernel's ”regressions tracker”; to facilitate this work he relies on regzbot, the Linux kernel regression tracking bot. That's why you want to bring your report on the radar of these people by CCing or forwarding each report to the regressions mailing list, ideally with a ”regzbot command” in your mail to get it tracked immediately.

How quickly are regressions normally fixed?

Developers should fix any reported regression as quickly as possible, to provide affected users with a solution in a timely manner and prevent more users from running into the issue; nevertheless developers need to take enough time and care to ensure regression fixes do not cause additional damage.

The answer thus depends on various factors like the impact of a regression, its age, or the Linux series in which it occurs. In the end though, most regressions should be fixed within two weeks.

Is it a regression, if the issue can be avoided by updating some software?

Almost always: yes. If a developer tells you otherwise, ask the regression tracker for advice as outlined above.

Is it a regression, if a newer kernel works slower or consumes more energy?

Yes, but the difference has to be significant. A five percent slow-down in a micro-benchmark thus is unlikely to qualify as regression, unless it also influences the results of a broad benchmark by more than one percent. If in doubt, ask for advice.

Is it a regression, if an external kernel module breaks when updating Linux?

No, as the ”no regression” rule is about interfaces and services the Linux kernel provides to the userland. It thus does not cover building or running externally developed kernel modules, as they run in kernel-space and hook into the kernel using internal interfaces occasionally changed.

How are regressions handled that are caused by security fixes?

In extremely rare situations security issues can't be fixed without causing regressions; those fixes are given way, as they are the lesser evil in the end. Luckily this middling almost always can be avoided, as key developers for the affected area and often Linus Torvalds himself try very hard to fix security issues without causing regressions.

If you nevertheless face such a case, check the mailing list archives if people tried their best to avoid the regression. If not, report it; if in doubt, ask for advice as outlined above.

What happens if fixing a regression is impossible without causing another?

Sadly these things happen, but luckily not very often; if they occur, expert developers of the affected code area should look into the issue to find a fix that avoids regressions or at least their impact. If you run into such a situation, do what was outlined already for regressions caused by security fixes: check earlier discussions if people already tried their best and ask for advice if in doubt.

A quick note while at it: these situations could be avoided, if people would regularly give mainline pre-releases (say v5.15-rc1 or -rc3) from each development cycle a test run. This is best explained by imagining a change integrated between Linux v5.14 and v5.15-rc1 which causes a regression, but at the same time is a hard requirement for some other improvement applied for 5.15-rc1. All these changes often can simply be reverted and the regression thus solved, if someone finds and reports it before 5.15 is released. A few days or weeks later this solution can become impossible, as some software might have started to rely on aspects introduced by one of the follow-up changes: reverting all changes would then cause a regression for users of said software and thus is out of the question.

Is it a regression, if some feature I relied on was removed months ago?

It is, but often it's hard to fix such regressions due to the aspects outlined in the previous section. It hence needs to be dealt with on a case-by-case basis. This is another reason why it's in everybody's interest to regularly test mainline pre-releases.

Does the ”no regression” rule apply if I seem to be the only affected person?

It does, but only for practical usage: the Linux developers want to be free to remove support for hardware only to be found in attics and museums anymore.

Note, sometimes regressions can't be avoided to make progress -- and the latter is needed to prevent Linux from stagnation. Hence, if only very few users seem to be affected by a regression, it for the greater good might be in their and everyone else's interest to lettings things pass. Especially if there is an easy way to circumvent the regression somehow, for example by updating some software or using a kernel parameter created just for this purpose.

Does the regression rule apply for code in the staging tree as well?

Not according to the help text for the configuration option covering all staging code, which since its early days states:

```
Please note that these drivers are under heavy development, may or
may not work, and may contain userspace interfaces that most likely
will be changed in the near future.
```

The staging developers nevertheless often adhere to the ”no regressions” rule, but sometimes bend it to make progress. That's for example why some users had to deal with (often negligible) regressions when a WiFi driver from the staging tree was replaced by a totally different one written from scratch.

Why do later versions have to be ”compiled with a similar configuration”?

Because the Linux kernel developers sometimes integrate changes known to cause regressions, but make them optional and disable them in the kernel's default configuration. This trick allows progress, as the ”no regressions” rule otherwise would lead to stagnation.

Consider for example a new security feature blocking access to some kernel interfaces often abused by malware, which at the same time are required to run a few rarely used applications. The outlined approach makes both camps happy: people using these applications can leave the new security feature off, while everyone else can enable it without running into trouble.

How to create a configuration similar to the one of an older kernel?

Start your machine with a known-good kernel and configure the newer Linux version with `make olddefconfig`. This makes the kernel's build scripts pick up the configuration file (the ”.config” file) from the running kernel as base for the new one you are about to compile; afterwards they set all new configuration options to their default value, which should disable new features that might cause regressions.

Can I report a regression I found with pre-compiled vanilla kernels?

You need to ensure the newer kernel was compiled with a similar configuration file as the older one (see above), as those that built them might have enabled some known-to-be incompatible feature for the newer kernel. If in doubt, report the matter to the kernel's provider and ask for advice.

9.2.3 More about regression tracking with ”regzbot”

What is regression tracking and why should I care about it?

Rules like ”no regressions” need someone to ensure they are followed, otherwise they are broken either accidentally or on purpose. History has shown this to be true for Linux kernel development as well. That's why Thorsten Leemhuis, the Linux Kernel's regression tracker, and some people try to ensure all regression are fixed by keeping an eye on them until they are resolved. Neither of them are paid for this, that's why the work is done on a best effort basis.

Why and how are Linux kernel regressions tracked using a bot?

Tracking regressions completely manually has proven to be quite hard due to the distributed and loosely structured nature of Linux kernel development process. That's why the Linux kernel's regression tracker developed regzbot to facilitate the work, with the long term goal to automate regression tracking as much as possible for everyone involved.

Regzbot works by watching for replies to reports of tracked regressions. Additionally, it's looking out for posted or committed patches referencing such reports with ”Link:” tags; replies to such patch postings are tracked as well. Combined this data provides good insights into the current state of the fixing process.

How to see which regressions regzbot tracks currently?

Check out regzbot's web-interface.

What kind of issues are supposed to be tracked by regzbot?

The bot is meant to track regressions, hence please don't involve regzbot for regular issues. But it's okay for the Linux kernel's regression tracker if you involve regzbot to track severe issues, like reports about hangs, corrupted data, or internal errors (Panic, Oops, BUG(), warning, ...).

How to change aspects of a tracked regression?

By using a 'regzbot command' in a direct or indirect reply to the mail with the report. The easiest way to do that: find the report in your ”Sent” folder or the mailing list archive and reply to it using your mailer's ”Reply-all” function. In that mail, use one of the following commands in a stand-alone paragraph (IOW: use blank lines to separate one or multiple of these commands from the rest of the mail's text).

- Update when the regression started to happen, for example after performing a bisection:

```
#regzbot introduced: 1f2e3d4c5d
```

- Set or update the title:

```
#regzbot title: foo
```

- Monitor a discussion or bugzilla.kernel.org ticket where additions aspects of the issue or a fix are discussed::

```
#regzbot monitor: https://lore.kernel.org/r/30th.anniversary.repost@klaava.
↪Helsinki.FI/
#regzbot monitor: https://bugzilla.kernel.org/show_bug.cgi?id=123456789
```

- Point to a place with further details of interest, like a mailing list post or a ticket in a bug tracker that are slightly related, but about a different topic:

```
#regzbot link: https://bugzilla.kernel.org/show_bug.cgi?id=123456789
```

- Mark a regression as invalid:

```
#regzbot invalid: wasn't a regression, problem has always existed
```

Regzbot supports a few other commands primarily used by developers or people tracking regressions. They and more details about the aforementioned regzbot commands can be found in the getting started guide and the reference documentation for regzbot.

CHAPTER

TEN

HOW TO QUICKLY BUILD A TRIMMED LINUX KERNEL

This guide explains how to swiftly build Linux kernels that are ideal for testing purposes, but perfectly fine for day-to-day use, too.

10.1 The essence of the process (aka 'TL;DR')

[If you are new to compiling Linux, ignore this TLDR and head over to the next section below: it contains a step-by-step guide, which is more detailed, but still brief and easy to follow; that guide and its accompanying reference section also mention alternatives, pitfalls, and additional aspects, all of which might be relevant for you.]

If your system uses techniques like Secure Boot, prepare it to permit starting self-compiled Linux kernels; install compilers and everything else needed for building Linux; make sure to have 12 Gigabyte free space in your home directory. Now run the following commands to download fresh Linux mainline sources, which you then use to configure, build and install your own kernel:

```
git clone --depth 1 -b master \
  https://git.kernel.org/pub/scm/linux/kernel/git/stable/linux.git ~/linux/
cd ~/linux/
# Hint: if you want to apply patches, do it at this point. See below for␣
↪details.
# Hint: it's recommended to tag your build at this point. See below for␣
↪details.
yes "" | make localmodconfig
# Hint: at this point you might want to adjust the build configuration; you'll
#   have to, if you are running Debian. See below for details.
make -j $(nproc --all)
# Note: on many commodity distributions the next command suffices, but on Arch
#   Linux, its derivatives, and some others it does not. See below for details.
command -v installkernel && sudo make modules_install install
reboot
```

If you later want to build a newer mainline snapshot, use these commands:

```
cd ~/linux/
git fetch --depth 1 origin
# Note: the next command will discard any changes you did to the code:
git checkout --force --detach origin/master
# Reminder: if you want to (re)apply patches, do it at this point.
```

```
# Reminder: you might want to add or modify a build tag at this point.
make olddefconfig
make -j $(nproc --all)
# Reminder: the next command on some distributions does not suffice.
command -v installkernel && sudo make modules_install install
reboot
```

10.2 Step-by-step guide

Compiling your own Linux kernel is easy in principle. There are various ways to do it. Which of them actually work and is the best depends on the circumstances.

This guide describes a way perfectly suited for those who want to quickly install Linux from sources without being bothered by complicated details; the goal is to cover everything typically needed on mainstream Linux distributions running on commodity PC or server hardware.

The described approach is great for testing purposes, for example to try a proposed fix or to check if a problem was already fixed in the latest codebase. Nonetheless, kernels built this way are also totally fine for day-to-day use while at the same time being easy to keep up to date.

The following steps describe the important aspects of the process; a comprehensive reference section later explains each of them in more detail. It sometimes also describes alternative approaches, pitfalls, as well as errors that might occur at a particular point -- and how to then get things rolling again.

- Create a fresh backup and put system repair and restore tools at hand, just to be prepared for the unlikely case of something going sideways.

  [*details*]

- On platforms with 'Secure Boot' or similar techniques, prepare everything to ensure the system will permit your self-compiled kernel to boot later. The quickest and easiest way to achieve this on commodity x86 systems is to disable such techniques in the BIOS setup utility; alternatively, remove their restrictions through a process initiated by `mokutil --disable-validation`.

  [*details*]

- Install all software required to build a Linux kernel. Often you will need: 'bc', 'binutils' ('ld' et al.), 'bison', 'flex', 'gcc', 'git', 'openssl', 'pahole', 'perl', and the development headers for 'libelf' and 'openssl'. The reference section shows how to quickly install those on various popular Linux distributions.

  [*details*]

- Ensure to have enough free space for building and installing Linux. For the latter 150 Megabyte in /lib/ and 100 in /boot/ are a safe bet. For storing sources and build artifacts 12 Gigabyte in your home directory should typically suffice. If you have less available, be sure to check the reference section for the step that explains adjusting your kernels build configuration: it mentions a trick that reduce the amount of required space in /home/ to around 4 Gigabyte.

  [*details*]

- Retrieve the sources of the Linux version you intend to build; then change into the directory holding them, as all further commands in this guide are meant to be executed from there.

  [Note: the following paragraphs describe how to retrieve the sources by partially cloning the Linux stable git repository. This is called a shallow clone. The reference section explains two alternatives: packaged archives and a full git clone ; prefer the latter, if downloading a lot of data does not bother you, as that will avoid some peculiar characteristics of shallow clones the reference section explains .]

  First, execute the following command to retrieve a fresh mainline codebase:

  ```
  git clone --no-checkout --depth 1 -b master \
    https://git.kernel.org/pub/scm/linux/kernel/git/stable/linux.git ~/linux/
  cd ~/linux/
  ```

  If you want to access recent mainline releases and pre-releases, deepen you clone's history to the oldest mainline version you are interested in:

  ```
  git fetch --shallow-exclude=v6.0 origin
  ```

  In case you want to access a stable/longterm release (say v6.1.5), simply add the branch holding that series; afterwards fetch the history at least up to the mainline version that started the series (v6.1):

  ```
  git remote set-branches --add origin linux-6.1.y
  git fetch --shallow-exclude=v6.0 origin
  ```

  Now checkout the code you are interested in. If you just performed the initial clone, you will be able to check out a fresh mainline codebase, which is ideal for checking whether developers already fixed an issue:

  ```
  git checkout --detach origin/master
  ```

  If you deepened your clone, you instead of `origin/master` can specify the version you deepened to (`v6.0` above); later releases like `v6.1` and pre-release like `v6.2-rc1` will work, too. Stable or longterm versions like `v6.1.5` work just the same, if you added the appropriate stable/longterm branch as described.

  [*details*]

- In case you want to apply a kernel patch, do so now. Often a command like this will do the trick:

  ```
  patch -p1 < ../proposed-fix.patch
  ```

  If the `-p1` is actually needed, depends on how the patch was created; in case it does not apply thus try without it.

  If you cloned the sources with git and anything goes sideways, run `git reset --hard` to undo any changes to the sources.

  [*details*]

  - If you patched your kernel or have one of the same version installed already, better add a unique tag to the one you are about to build:

```
echo "-proposed_fix" > localversion
```

Running `uname -r` under your kernel later will then print something like '6.1-rc4-proposed_fix'.

[*details*]

- Create the build configuration for your kernel based on an existing configuration.

  If you already prepared such a '.config' file yourself, copy it to ~/linux/ and run `make olddefconfig`.

  Use the same command, if your distribution or somebody else already tailored your running kernel to your or your hardware's needs: the make target 'olddefconfig' will then try to use that kernel's .config as base.

  Using this make target is fine for everybody else, too -- but you often can save a lot of time by using this command instead:

```
yes "" | make localmodconfig
```

  This will try to pick your distribution's kernel as base, but then disable modules for any features apparently superfluous for your setup. This will reduce the compile time enormously, especially if you are running an universal kernel from a commodity Linux distribution.

  There is a catch: 'localmodconfig' is likely to disable kernel features you did not use since you booted your Linux -- like drivers for currently disconnected peripherals or a virtualization software not haven't used yet. You can reduce or nearly eliminate that risk with tricks the reference section outlines; but when building a kernel just for quick testing purposes it is often negligible if such features are missing. But you should keep that aspect in mind when using a kernel built with this make target, as it might be the reason why something you only use occasionally stopped working.

  [*details*]

- Check if you might want to or have to adjust some kernel configuration options:
- Evaluate how you want to handle debug symbols. Enable them, if you later might need to decode a stack trace found for example in a 'panic', 'Oops', 'warning', or 'BUG'; on the other hand disable them, if you are short on storage space or prefer a smaller kernel binary. See the reference section for details on how to do either. If neither applies, it will likely be fine to simply not bother with this. [*details*]
- Are you running Debian? Then to avoid known problems by performing additional adjustments explained in the reference section. [*details*].
- If you want to influence the other aspects of the configuration, do so now by using make targets like 'menuconfig' or 'xconfig'. [*details*].

- Build the image and the modules of your kernel:

```
make -j $(nproc --all)
```

  If you want your kernel packaged up as deb, rpm, or tar file, see the reference section for alternatives.

[*details*]

- Now install your kernel:

```
command -v installkernel && sudo make modules_install install
```

Often all left for you to do afterwards is a `reboot`, as many commodity Linux distributions will then create an initramfs (also known as initrd) and an entry for your kernel in your bootloader's configuration; but on some distributions you have to take care of these two steps manually for reasons the reference section explains.

On a few distributions like Arch Linux and its derivatives the above command does nothing at all; in that case you have to manually install your kernel, as outlined in the reference section.

If you are running a immutable Linux distribution, check its documentation and the web to find out how to install your own kernel there.

[*details*]

- To later build another kernel you need similar steps, but sometimes slightly different commands.

First, switch back into the sources tree:

```
cd ~/linux/
```

In case you want to build a version from a stable/longterm series you have not used yet (say 6.2.y), tell git to track it:

```
git remote set-branches --add origin linux-6.2.y
```

Now fetch the latest upstream changes; you again need to specify the earliest version you care about, as git otherwise might retrieve the entire commit history:

```
git fetch --shallow-exclude=v6.0 origin
```

Now switch to the version you are interested in -- but be aware the command used here will discard any modifications you performed, as they would conflict with the sources you want to checkout:

```
git checkout --force --detach origin/master
```

At this point you might want to patch the sources again or set/modify a build tag, as explained earlier. Afterwards adjust the build configuration to the new codebase using olddefconfig, which will now adjust the configuration file you prepared earlier using localmodconfig (~/linux/.config) for your next kernel:

```
# reminder: if you want to apply patches, do it at this point
# reminder: you might want to update your build tag at this point
make olddefconfig
```

Now build your kernel:

```
make -j $(nproc --all)
```

Afterwards install the kernel as outlined above:

```
command -v installkernel && sudo make modules_install install
```

[*details*]

- Your kernel is easy to remove later, as its parts are only stored in two places and clearly identifiable by the kernel's release name. Just ensure to not delete the kernel you are running, as that might render your system unbootable.

  Start by deleting the directory holding your kernel's modules, which is named after its release name -- '6.0.1-foobar' in the following example:

  ```
  sudo rm -rf /lib/modules/6.0.1-foobar
  ```

  Now try the following command, which on some distributions will delete all other kernel files installed while also removing the kernel's entry from the bootloader configuration:

  ```
  command -v kernel-install && sudo kernel-install -v remove 6.0.1-foobar
  ```

  If that command does not output anything or fails, see the reference section; do the same if any files named '*6.0.1-foobar*' remain in /boot/.

  [*details*]

Did you run into trouble following any of the above steps that is not cleared up by the reference section below? Or do you have ideas how to improve the text? Then please take a moment of your time and let the maintainer of this document know by email (Thorsten Leemhuis <linux@leemhuis.info>), ideally while CCing the Linux docs mailing list (linux-doc@vger.kernel.org). Such feedback is vital to improve this document further, which is in everybody's interest, as it will enable more people to master the task described here.

10.3 Reference section for the step-by-step guide

This section holds additional information for each of the steps in the above guide.

10.3.1 Prepare for emergencies

Create a fresh backup and put system repair and restore tools at hand [...]

Remember, you are dealing with computers, which sometimes do unexpected things -- especially if you fiddle with crucial parts like the kernel of an operating system. That's what you are about to do in this process. Hence, better prepare for something going sideways, even if that should not happen.

[*back to step-by-step guide*]

10.3.2 Dealing with techniques like Secure Boot

> *On platforms with 'Secure Boot' or similar techniques, prepare everything to ensure the system will permit your self-compiled kernel to boot later.* [...]

Many modern systems allow only certain operating systems to start; they thus by default will reject booting self-compiled kernels.

You ideally deal with this by making your platform trust your self-built kernels with the help of a certificate and signing. How to do that is not described here, as it requires various steps that would take the text too far away from its purpose; '*Kernel module signing facility*' and various web sides already explain this in more detail.

Temporarily disabling solutions like Secure Boot is another way to make your own Linux boot. On commodity x86 systems it is possible to do this in the BIOS Setup utility; the steps to do so are not described here, as they greatly vary between machines.

On mainstream x86 Linux distributions there is a third and universal option: disable all Secure Boot restrictions for your Linux environment. You can initiate this process by running `mokutil --disable-validation`; this will tell you to create a one-time password, which is safe to write down. Now restart; right after your BIOS performed all self-tests the bootloader Shim will show a blue box with a message 'Press any key to perform MOK management'. Hit some key before the countdown exposes. This will open a menu and choose 'Change Secure Boot state' there. Shim's 'MokManager' will now ask you to enter three randomly chosen characters from the one-time password specified earlier. Once you provided them, confirm that you really want to disable the validation. Afterwards, permit MokManager to reboot the machine.

[*back to step-by-step guide*]

10.3.3 Install build requirements

> *Install all software required to build a Linux kernel.* [...]

The kernel is pretty stand-alone, but besides tools like the compiler you will sometimes need a few libraries to build one. How to install everything needed depends on your Linux distribution and the configuration of the kernel you are about to build.

Here are a few examples what you typically need on some mainstream distributions:

- Debian, Ubuntu, and derivatives:

```
sudo apt install bc binutils bison dwarves flex gcc git make openssl \
  pahole perl-base libssl-dev libelf-dev
```

- Fedora and derivatives:

```
sudo dnf install binutils /usr/include/{libelf.h,openssl/pkcs7.h} \
  /usr/bin/{bc,bison,flex,gcc,git,openssl,make,perl,pahole}
```

- openSUSE and derivatives:

```
sudo zypper install bc binutils bison dwarves flex gcc git make perl-base \
  openssl openssl-devel libelf-dev
```

In case you wonder why these lists include openssl and its development headers: they are needed for the Secure Boot support, which many distributions enable in their kernel configuration for x86 machines.

Sometimes you will need tools for compression formats like bzip2, gzip, lz4, lzma, lzo, xz, or zstd as well.

You might need additional libraries and their development headers in case you perform tasks not covered in this guide. For example, zlib will be needed when building kernel tools from the tools/ directory; adjusting the build configuration with make targets like 'menuconfig' or 'xconfig' will require development headers for ncurses or Qt5.

[*back to step-by-step guide*]

10.3.4 Space requirements

> *Ensure to have enough free space for building and installing Linux.* [...]

The numbers mentioned are rough estimates with a big extra charge to be on the safe side, so often you will need less.

If you have space constraints, remember to read the reference section when you reach the *section about configuration adjustments*', as ensuring debug symbols are disabled will reduce the consumed disk space by quite a few gigabytes.

[*back to step-by-step guide*]

10.3.5 Download the sources

> *Retrieve the sources of the Linux version you intend to build.* [...]

The step-by-step guide outlines how to retrieve Linux' sources using a shallow git clone. There is *more to tell about this method* and two alternate ways worth describing: *packaged archives* and *a full git clone*. And the aspects '*wouldn't it be wiser to use a proper pre-release than the latest mainline code*' and '*how to get an even fresher mainline codebase*' need elaboration, too.

Note, to keep things simple the commands used in this guide store the build artifacts in the source tree. If you prefer to separate them, simply add something like `O=~/linux-builddir/` to all make calls; also adjust the path in all commands that add files or modify any generated (like your '.config').

[*back to step-by-step guide*]

Noteworthy characteristics of shallow clones

The step-by-step guide uses a shallow clone, as it is the best solution for most of this document's target audience. There are a few aspects of this approach worth mentioning:

- This document in most places uses `git fetch` with `--shallow-exclude=` to specify the earliest version you care about (or to be precise: its git tag). You alternatively can use the parameter `--shallow-since=` to specify an absolute (say `'2023-07-15'`) or relative (`'12 months'`) date to define the depth of the history you want to download. As a second alternative, you can also specify a certain depth explicitly with a parameter like `--depth=1`, unless you add branches for stable/longterm kernels.

- When running `git fetch`, remember to always specify the oldest version, the time you care about, or an explicit depth as shown in the step-by-step guide. Otherwise you will risk downloading nearly the entire git history, which will consume quite a bit of time and bandwidth while also stressing the servers.

  Note, you do not have to use the same version or date all the time. But when you change it over time, git will deepen or flatten the history to the specified point. That allows you to retrieve versions you initially thought you did not need -- or it will discard the sources of older versions, for example in case you want to free up some disk space. The latter will happen automatically when using `--shallow-since=` or `--depth=`.

- Be warned, when deepening your clone you might encounter an error like 'fatal: error in object: unshallow cafecaca0c0dacafecaca0c0dacafecaca0c0da'. In that case run `git repack -d` and try again``

- In case you want to revert changes from a certain version (say Linux 6.3) or perform a bisection (v6.2..v6.3), better tell `git fetch` to retrieve objects up to three versions earlier (e.g. 6.0): `git describe` will then be able to describe most commits just like it would in a full git clone.

[*back to step-by-step guide*] [*back to section intro*]

Downloading the sources using a packages archive

People new to compiling Linux often assume downloading an archive via the front-page of https://kernel.org is the best approach to retrieve Linux' sources. It actually can be, if you are certain to build just one particular kernel version without changing any code. Thing is: you might be sure this will be the case, but in practice it often will turn out to be a wrong assumption.

That's because when reporting or debugging an issue developers will often ask to give another version a try. They also might suggest temporarily undoing a commit with `git revert` or might provide various patches to try. Sometimes reporters will also be asked to use `git bisect` to find the change causing a problem. These things rely on git or are a lot easier and quicker to handle with it.

A shallow clone also does not add any significant overhead. For example, when you use `git clone --depth=1` to create a shallow clone of the latest mainline codebase git will only retrieve a little more data than downloading the latest mainline pre-release (aka 'rc') via the front-page of kernel.org would.

A shallow clone therefore is often the better choice. If you nevertheless want to use a packaged source archive, download one via kernel.org; afterwards extract its content to some directory and change to the subdirectory created during extraction. The rest of the step-by-step guide will work just fine, apart from things that rely on git -- but this mainly concerns the section on successive builds of other versions.

[*back to step-by-step guide*] [*back to section intro*]

Downloading the sources using a full git clone

If downloading and storing a lot of data (~4,4 Gigabyte as of early 2023) is nothing that bothers you, instead of a shallow clone perform a full git clone instead. You then will avoid the specialties mentioned above and will have all versions and individual commits at hand at any time:

```
curl -L \
  https://git.kernel.org/pub/scm/linux/kernel/git/stable/linux.git/clone.
↪bundle \
  -o linux-stable.git.bundle
git clone linux-stable.git.bundle ~/linux/
rm linux-stable.git.bundle
cd ~/linux/
git remote set-url origin \
  https://git.kernel.org/pub/scm/linux/kernel/git/stable/linux.git
git fetch origin
git checkout --detach origin/master
```

[*back to step-by-step guide*] [*back to section intro*]

Proper pre-releases (RCs) vs. latest mainline

When cloning the sources using git and checking out origin/master, you often will retrieve a codebase that is somewhere between the latest and the next release or pre-release. This almost always is the code you want when giving mainline a shot: pre-releases like v6.1-rc5 are in no way special, as they do not get any significant extra testing before being published.

There is one exception: you might want to stick to the latest mainline release (say v6.1) before its successor's first pre-release (v6.2-rc1) is out. That is because compiler errors and other problems are more likely to occur during this time, as mainline then is in its 'merge window': a usually two week long phase, in which the bulk of the changes for the next release is merged.

[*back to step-by-step guide*] [*back to section intro*]

Avoiding the mainline lag

The explanations for both the shallow clone and the full clone both retrieve the code from the Linux stable git repository. That makes things simpler for this document's audience, as it allows easy access to both mainline and stable/longterm releases. This approach has just one downside:

Changes merged into the mainline repository are only synced to the master branch of the Linux stable repository every few hours. This lag most of the time is not something to worry about; but in case you really need the latest code, just add the mainline repo as additional remote and checkout the code from there:

```
git remote add mainline \
  https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git
git fetch mainline
git checkout --detach mainline/master
```

When doing this with a shallow clone, remember to call `git fetch` with one of the parameters described earlier to limit the depth.

[*back to step-by-step guide*] [*back to section intro*]

10.3.6 Patch the sources (optional)

> *In case you want to apply a kernel patch, do so now.* [...]

This is the point where you might want to patch your kernel -- for example when a developer proposed a fix and asked you to check if it helps. The step-by-step guide already explains everything crucial here.

[*back to step-by-step guide*]

10.3.7 Tagging this kernel build (optional, often wise)

> *If you patched your kernel or already have that kernel version installed, better tag your kernel by extending its release name:* [...]

Tagging your kernel will help avoid confusion later, especially when you patched your kernel. Adding an individual tag will also ensure the kernel's image and its modules are installed in parallel to any existing kernels.

There are various ways to add such a tag. The step-by-step guide realizes one by creating a 'localversion' file in your build directory from which the kernel build scripts will automatically pick up the tag. You can later change that file to use a different tag in subsequent builds or simply remove that file to dump the tag.

[*back to step-by-step guide*]

10.3.8 Define the build configuration for your kernel

> *Create the build configuration for your kernel based on an existing configuration.* [...]

There are various aspects for this steps that require a more careful explanation:

Pitfalls when using another configuration file as base

Make targets like localmodconfig and olddefconfig share a few common snares you want to be aware of:

- These targets will reuse a kernel build configuration in your build directory (e.g. '~/linux/.config'), if one exists. In case you want to start from scratch you thus need to delete it.
- The make targets try to find the configuration for your running kernel automatically, but might choose poorly. A line like '# using defaults found in /boot/config-6.0.7-250.fc36.x86_64' or 'using config: '/boot/config-6.0.7-250.fc36.x86_64' tells you which file they picked. If that is not the intended one, simply store it as '~/linux/.config' before using these make targets.

- Unexpected things might happen if you try to use a config file prepared for one kernel (say v6.0) on an older generation (say v5.15). In that case you might want to use a configuration as base which your distribution utilized when they used that or an slightly older kernel version.

Influencing the configuration

The make target olddefconfig and the `yes "" |` used when utilizing localmodconfig will set any undefined build options to their default value. This among others will disable many kernel features that were introduced after your base kernel was released.

If you want to set these configurations options manually, use `oldconfig` instead of `olddefconfig` or omit the `yes "" |` when utilizing localmodconfig. Then for each undefined configuration option you will be asked how to proceed. In case you are unsure what to answer, simply hit 'enter' to apply the default value.

Big pitfall when using localmodconfig

As explained briefly in the step-by-step guide already: with localmodconfig it can easily happen that your self-built kernel will lack modules for tasks you did not perform before utilizing this make target. That's because those tasks require kernel modules that are normally autoloaded when you perform that task for the first time; if you didn't perform that task at least once before using localmodonfig, the latter will thus assume these modules are superfluous and disable them.

You can try to avoid this by performing typical tasks that often will autoload additional kernel modules: start a VM, establish VPN connections, loop-mount a CD/DVD ISO, mount network shares (CIFS, NFS, ...), and connect all external devices (2FA keys, headsets, webcams, ...) as well as storage devices with file systems you otherwise do not utilize (btrfs, ext4, FAT, NTFS, XFS, ...). But it is hard to think of everything that might be needed -- even kernel developers often forget one thing or another at this point.

Do not let that risk bother you, especially when compiling a kernel only for testing purposes: everything typically crucial will be there. And if you forget something important you can turn on a missing feature later and quickly run the commands to compile and install a better kernel.

But if you plan to build and use self-built kernels regularly, you might want to reduce the risk by recording which modules your system loads over the course of a few weeks. You can automate this with modprobed-db. Afterwards use `LSMOD=<path>` to point localmodconfig to the list of modules modprobed-db noticed being used:

```
yes "" | make LSMOD="${HOME}"/.config/modprobed.db localmodconfig
```

Remote building with localmodconfig

If you want to use localmodconfig to build a kernel for another machine, run `lsmod > lsmod_foo-machine` on it and transfer that file to your build host. Now point the build scripts to the file like this: `yes "" | make LSMOD=~/lsmod_foo-machine localmodconfig`. Note, in this case you likely want to copy a base kernel configuration from the other machine over as well and place it as .config in your build directory.

[*back to step-by-step guide*]

10.3.9 Adjust build configuration

Check if you might want to or have to adjust some kernel configuration options:

Depending on your needs you at this point might want or have to adjust some kernel configuration options.

Debug symbols

Evaluate how you want to handle debug symbols. [...]

Most users do not need to care about this, it's often fine to leave everything as it is; but you should take a closer look at this, if you might need to decode a stack trace or want to reduce space consumption.

Having debug symbols available can be important when your kernel throws a 'panic', 'Oops', 'warning', or 'BUG' later when running, as then you will be able to find the exact place where the problem occurred in the code. But collecting and embedding the needed debug information takes time and consumes quite a bit of space: in late 2022 the build artifacts for a typical x86 kernel configured with localmodconfig consumed around 5 Gigabyte of space with debug symbols, but less than 1 when they were disabled. The resulting kernel image and the modules are bigger as well, which increases load times.

Hence, if you want a small kernel and are unlikely to decode a stack trace later, you might want to disable debug symbols to avoid above downsides:

```
./scripts/config --file .config -d DEBUG_INFO \
  -d DEBUG_INFO_DWARF_TOOLCHAIN_DEFAULT -d DEBUG_INFO_DWARF4 \
  -d DEBUG_INFO_DWARF5 -e CONFIG_DEBUG_INFO_NONE
make olddefconfig
```

You on the other hand definitely want to enable them, if there is a decent chance that you need to decode a stack trace later (as explained by 'Decode failure messages' in *Tainted kernels* in more detail):

```
./scripts/config --file .config -d DEBUG_INFO_NONE -e DEBUG_KERNEL
  -e DEBUG_INFO -e DEBUG_INFO_DWARF_TOOLCHAIN_DEFAULT -e KALLSYMS -e KALLSYMS_
↪ALL
make olddefconfig
```

Note, many mainstream distributions enable debug symbols in their kernel configurations -- make targets like localmodconfig and olddefconfig thus will often pick that setting up.

[*back to step-by-step guide*]

Distro specific adjustments

Are you running [...]

The following sections help you to avoid build problems that are known to occur when following this guide on a few commodity distributions.

Debian:

- Remove a stale reference to a certificate file that would cause your build to fail:

```
./scripts/config --file .config --set-str SYSTEM_TRUSTED_KEYS ''
```

  Alternatively, download the needed certificate and make that configuration option point to it, as the Debian handbook explains in more detail -- or generate your own, as explained in *Kernel module signing facility*.

[*back to step-by-step guide*]

Individual adjustments

If you want to influence the other aspects of the configuration, do so now [...]

You at this point can use a command like `make menuconfig` to enable or disable certain features using a text-based user interface; to use a graphical configuration utilize, use the make target `xconfig` or `gconfig` instead. All of them require development libraries from toolkits they are based on (ncurses, Qt5, Gtk2); an error message will tell you if something required is missing.

[*back to step-by-step guide*]

10.3.10 Build your kernel

Build the image and the modules of your kernel [...]

A lot can go wrong at this stage, but the instructions below will help you help yourself. Another subsection explains how to directly package your kernel up as deb, rpm or tar file.

Dealing with build errors

When a build error occurs, it might be caused by some aspect of your machine's setup that often can be fixed quickly; other times though the problem lies in the code and can only be fixed by a developer. A close examination of the failure messages coupled with some research on the internet will often tell you which of the two it is. To perform such a investigation, restart the build process like this:

```
make V=1
```

The `V=1` activates verbose output, which might be needed to see the actual error. To make it easier to spot, this command also omits the `-j $(nproc --all)` used earlier to utilize every CPU core in the system for the job -- but this parallelism also results in some clutter when failures occur.

After a few seconds the build process should run into the error again. Now try to find the most crucial line describing the problem. Then search the internet for the most important and non-generic section of that line (say 4 to 8 words); avoid or remove anything that looks remotely system-specific, like your username or local path names like `/home/username/linux/`. First try your regular internet search engine with that string, afterwards search Linux kernel mailing lists via lore.kernel.org/all/.

This most of the time will find something that will explain what is wrong; quite often one of the hits will provide a solution for your problem, too. If you do not find anything that matches your problem, try again from a different angle by modifying your search terms or using another line from the error messages.

In the end, most trouble you are to run into has likely been encountered and reported by others already. That includes issues where the cause is not your system, but lies the code. If you run into one of those, you might thus find a solution (e.g. a patch) or workaround for your problem, too.

Package your kernel up

The step-by-step guide uses the default make targets (e.g. 'bzImage' and 'modules' on x86) to build the image and the modules of your kernel, which later steps of the guide then install. You instead can also directly build everything and directly package it up by using one of the following targets:

- `make -j $(nproc --all) bindeb-pkg` to generate a deb package
- `make -j $(nproc --all) binrpm-pkg` to generate a rpm package
- `make -j $(nproc --all) tarbz2-pkg` to generate a bz2 compressed tarball

This is just a selection of available make targets for this purpose, see `make help` for others. You can also use these targets after running `make -j $(nproc --all)`, as they will pick up everything already built.

If you employ the targets to generate deb or rpm packages, ignore the step-by-step guide's instructions on installing and removing your kernel; instead install and remove the packages using the package utility for the format (e.g. dpkg and rpm) or a package management utility build on top of them (apt, aptitude, dnf/yum, zypper, ...). Be aware that the packages generated using these two make targets are designed to work on various distributions utilizing those formats, they thus will sometimes behave differently than your distribution's kernel packages.

[*back to step-by-step guide*]

10.3.11 Install your kernel

> *Now install your kernel* [...]

What you need to do after executing the command in the step-by-step guide depends on the existence and the implementation of an `installkernel` executable. Many commodity Linux distributions ship such a kernel installer in `/sbin/` that does everything needed, hence there is nothing left for you except rebooting. But some distributions contain an installkernel that does only part of the job -- and a few lack it completely and leave all the work to you.

If `installkernel` is found, the kernel's build system will delegate the actual installation of your kernel's image and related files to this executable. On almost all Linux distributions it will

store the image as '/boot/vmlinuz- <your kernel's release name>' and put a 'System.map-<your kernel's release name>' alongside it. Your kernel will thus be installed in parallel to any existing ones, unless you already have one with exactly the same release name.

Installkernel on many distributions will afterwards generate an 'initramfs' (often also called 'initrd'), which commodity distributions rely on for booting; hence be sure to keep the order of the two make targets used in the step-by-step guide, as things will go sideways if you install your kernel's image before its modules. Often installkernel will then add your kernel to the bootloader configuration, too. You have to take care of one or both of these tasks yourself, if your distributions installkernel doesn't handle them.

A few distributions like Arch Linux and its derivatives totally lack an installkernel executable. On those just install the modules using the kernel's build system and then install the image and the System.map file manually:

```
sudo make modules_install
sudo install -m 0600 $(make -s image_name) /boot/vmlinuz-$(make -s␣
↪kernelrelease)
sudo install -m 0600 System.map /boot/System.map-$(make -s kernelrelease)
```

If your distribution boots with the help of an initramfs, now generate one for your kernel using the tools your distribution provides for this process. Afterwards add your kernel to your bootloader configuration and reboot.

[*back to step-by-step guide*]

10.3.12 Another round later

> *To later build another kernel you need similar, but sometimes slightly different commands* [...]

The process to build later kernels is similar, but at some points slightly different. You for example do not want to use 'localmodconfig' for succeeding kernel builds, as you already created a trimmed down configuration you want to use from now on. Hence instead just use `oldconfig` or `olddefconfig` to adjust your build configurations to the needs of the kernel version you are about to build.

If you created a shallow-clone with git, remember what the *section that explained the setup described in more detail*: you need to use a slightly different `git fetch` command and when switching to another series need to add an additional remote branch.

[*back to step-by-step guide*]

10.3.13 Uninstall the kernel later

> *All parts of your installed kernel are identifiable by its release name and thus easy to remove later.* [...]

Do not worry installing your kernel manually and thus bypassing your distribution's packaging system will totally mess up your machine: all parts of your kernel are easy to remove later, as files are stored in two places only and normally identifiable by the kernel's release name.

One of the two places is a directory in /lib/modules/, which holds the modules for each installed kernel. This directory is named after the kernel's release name; hence, to remove all modules

for one of your kernels, simply remove its modules directory in /lib/modules/.

The other place is /boot/, where typically one to five files will be placed during installation of a kernel. All of them usually contain the release name in their file name, but how many files and their name depends somewhat on your distribution's installkernel executable (*see above*) and its initramfs generator. On some distributions the `kernel-install` command mentioned in the step-by-step guide will remove all of these files for you -- and the entry for your kernel in the bootloader configuration at the same time, too. On others you have to take care of these steps yourself. The following command should interactively remove the two main files of a kernel with the release name '6.0.1-foobar':

```
rm -i /boot/{System.map,vmlinuz}-6.0.1-foobar
```

Now remove the belonging initramfs, which often will be called something like `/boot/initramfs-6.0.1-foobar.img` or `/boot/initrd.img-6.0.1-foobar`. Afterwards check for other files in /boot/ that have '6.0.1-foobar' in their name and delete them as well. Now remove the kernel from your bootloader's configuration.

Note, be very careful with wildcards like '*' when deleting files or directories for kernels manually: you might accidentally remove files of a 6.0.11 kernel when all you want is to remove 6.0 or 6.0.1.

[*back to step-by-step guide*]

10.4 FAQ

10.4.1 Why does this 'how-to' not work on my system?

As initially stated, this guide is 'designed to cover everything typically needed [to build a kernel] on mainstream Linux distributions running on commodity PC or server hardware'. The outlined approach despite this should work on many other setups as well. But trying to cover every possible use-case in one guide would defeat its purpose, as without such a focus you would need dozens or hundreds of constructs along the lines of 'in case you are having <insert machine or distro>, you at this point have to do <this and that> <instead|additionally>'. Each of which would make the text longer, more complicated, and harder to follow.

That being said: this of course is a balancing act. Hence, if you think an additional use-case is worth describing, suggest it to the maintainers of this document, as *described above*.

CHAPTER

ELEVEN

BUG HUNTING

Kernel bug reports often come with a stack dump like the one below:

```
------------[ cut here ]------------
WARNING: CPU: 1 PID: 28102 at kernel/module.c:1108 module_put+0x57/0x70
Modules linked in: dvb_usb_gp8psk(-) dvb_usb dvb_core nvidia_drm(PO) nvidia_
 ↪modeset(PO) snd_hda_codec_hdmi snd_hda_intel snd_hda_codec snd_hwdep snd_hda_
 ↪core snd_pcm snd_timer snd soundcore nvidia(PO) [last unloaded: rc_core]
CPU: 1 PID: 28102 Comm: rmmod Tainted: P        WC O 4.8.4-build.1 #1
Hardware name: MSI MS-7309/MS-7309, BIOS V1.12 02/23/2009
 00000000 c12ba080 00000000 00000000 c103ed6a c1616014 00000001 00006dc6
 c1615862 00000454 c109e8a7 c109e8a7 00000009 ffffffff 00000000 f13f6a10
 f5f5a600 c103ee33 00000009 00000000 00000000 c109e8a7 f80ca4d0 c109f617
Call Trace:
 [<c12ba080>] ? dump_stack+0x44/0x64
 [<c103ed6a>] ? __warn+0xfa/0x120
 [<c109e8a7>] ? module_put+0x57/0x70
 [<c109e8a7>] ? module_put+0x57/0x70
 [<c103ee33>] ? warn_slowpath_null+0x23/0x30
 [<c109e8a7>] ? module_put+0x57/0x70
 [<f80ca4d0>] ? gp8psk_fe_set_frontend+0x460/0x460 [dvb_usb_gp8psk]
 [<c109f617>] ? symbol_put_addr+0x27/0x50
 [<f80bc9ca>] ? dvb_usb_adapter_frontend_exit+0x3a/0x70 [dvb_usb]
 [<f80bb3bf>] ? dvb_usb_exit+0x2f/0xd0 [dvb_usb]
 [<c13d03bc>] ? usb_disable_endpoint+0x7c/0xb0
 [<f80bb48a>] ? dvb_usb_device_exit+0x2a/0x50 [dvb_usb]
 [<c13d2882>] ? usb_unbind_interface+0x62/0x250
 [<c136b514>] ? __pm_runtime_idle+0x44/0x70
 [<c13620d8>] ? __device_release_driver+0x78/0x120
 [<c1362907>] ? driver_detach+0x87/0x90
 [<c1361c48>] ? bus_remove_driver+0x38/0x90
 [<c13d1c18>] ? usb_deregister+0x58/0xb0
 [<c109fbb0>] ? SyS_delete_module+0x130/0x1f0
 [<c1055654>] ? task_work_run+0x64/0x80
 [<c1000fa5>] ? exit_to_usermode_loop+0x85/0x90
 [<c10013f0>] ? do_fast_syscall_32+0x80/0x130
 [<c1549f43>] ? sysenter_past_esp+0x40/0x6a
---[ end trace 6ebc60ef3981792f ]---
```

Such stack traces provide enough information to identify the line inside the Kernel's source code where the bug happened. Depending on the severity of the issue, it may also contain the

word **Oops**, as on this one:

```
BUG: unable to handle kernel NULL pointer dereference at   (null)
IP: [<c06969d4>] iret_exc+0x7d0/0xa59
*pdpt = 000000002258a001 *pde = 0000000000000000
Oops: 0002 [#1] PREEMPT SMP
...
```

Despite being an **Oops** or some other sort of stack trace, the offended line is usually required to identify and handle the bug. Along this chapter, we'll refer to ”Oops” for all kinds of stack traces that need to be analyzed.

If the kernel is compiled with `CONFIG_DEBUG_INFO`, you can enhance the quality of the stack trace by using file:*scripts/decode_stacktrace.sh*.

11.1 Modules linked in

Modules that are tainted or are being loaded or unloaded are marked with ”(...)”, where the taint flags are described in file:Tainted kernels, ”being loaded” is annotated with ”+”, and ”being unloaded” is annotated with ”-”.

11.2 Where is the Oops message is located?

Normally the Oops text is read from the kernel buffers by klogd and handed to `syslogd` which writes it to a syslog file, typically `/var/log/messages` (depends on `/etc/syslog.conf`). On systems with systemd, it may also be stored by the `journald` daemon, and accessed by running `journalctl` command.

Sometimes `klogd` dies, in which case you can run `dmesg > file` to read the data from the kernel buffers and save it. Or you can `cat /proc/kmsg > file`, however you have to break in to stop the transfer, since `kmsg` is a ”never ending file”.

If the machine has crashed so badly that you cannot enter commands or the disk is not available then you have three options:

(1) Hand copy the text from the screen and type it in after the machine has restarted. Messy but it is the only option if you have not planned for a crash. Alternatively, you can take a picture of the screen with a digital camera - not nice, but better than nothing. If the messages scroll off the top of the console, you may find that booting with a higher resolution (e.g., `vga=791`) will allow you to read more of the text. (Caveat: This needs `vesafb`, so won't help for 'early' oopses.)

(2) Boot with a serial console (see *Documentation/admin-guide/serial-console.rst*), run a null modem to a second machine and capture the output there using your favourite communication program. Minicom works well.

(3) Use Kdump (see *Documentation for Kdump - The kexec-based Crash Dumping Solution*), extract the kernel ring buffer from old memory with using dmesg gdbmacro in Documentation/admin-guide/kdump/gdbmacros.txt.

11.3 Finding the bug's location

Reporting a bug works best if you point the location of the bug at the Kernel source file. There are two methods for doing that. Usually, using `gdb` is easier, but the Kernel should be pre-compiled with debug info.

11.3.1 gdb

The GNU debugger (`gdb`) is the best way to figure out the exact file and line number of the OOPS from the `vmlinux` file.

The usage of gdb works best on a kernel compiled with `CONFIG_DEBUG_INFO`. This can be set by running:

```
$ ./scripts/config -d COMPILE_TEST -e DEBUG_KERNEL -e DEBUG_INFO
```

On a kernel compiled with `CONFIG_DEBUG_INFO`, you can simply copy the EIP value from the OOPS:

```
EIP:    0060:[<c021e50e>]    Not tainted VLI
```

And use GDB to translate that to human-readable form:

```
$ gdb vmlinux
(gdb) l *0xc021e50e
```

If you don't have `CONFIG_DEBUG_INFO` enabled, you use the function offset from the OOPS:

```
EIP is at vt_ioctl+0xda8/0x1482
```

And recompile the kernel with `CONFIG_DEBUG_INFO` enabled:

```
$ ./scripts/config -d COMPILE_TEST -e DEBUG_KERNEL -e DEBUG_INFO
$ make vmlinux
$ gdb vmlinux
(gdb) l *vt_ioctl+0xda8
0x1888 is in vt_ioctl (drivers/tty/vt/vt_ioctl.c:293).
288   {
289           struct vc_data *vc = NULL;
290           int ret = 0;
291
292           console_lock();
293           if (VT_BUSY(vc_num))
294                   ret = -EBUSY;
295           else if (vc_num)
296                   vc = vc_deallocate(vc_num);
297           console_unlock();
```

or, if you want to be more verbose:

```
(gdb) p vt_ioctl
$1 = {int (struct tty_struct *, unsigned int, unsigned long)} 0xae0 <vt_ioctl>
(gdb) l *0xae0+0xda8
```

You could, instead, use the object file:

```
$ make drivers/tty/
$ gdb drivers/tty/vt/vt_ioctl.o
(gdb) l *vt_ioctl+0xda8
```

If you have a call trace, such as:

```
Call Trace:
 [<ffffffff8802c8e9>] :jbd:log_wait_commit+0xa3/0xf5
 [<ffffffff810482d9>] autoremove_wake_function+0x0/0x2e
 [<ffffffff8802770b>] :jbd:journal_stop+0x1be/0x1ee
 ...
```

this shows the problem likely is in the :jbd: module. You can load that module in gdb and list the relevant code:

```
$ gdb fs/jbd/jbd.ko
(gdb) l *log_wait_commit+0xa3
```

Note: You can also do the same for any function call at the stack trace, like this one:

```
[<f80bc9ca>] ? dvb_usb_adapter_frontend_exit+0x3a/0x70 [dvb_usb]
```

The position where the above call happened can be seen with:

```
$ gdb drivers/media/usb/dvb-usb/dvb-usb.o
(gdb) l *dvb_usb_adapter_frontend_exit+0x3a
```

11.3.2 objdump

To debug a kernel, use objdump and look for the hex offset from the crash output to find the valid line of code/assembler. Without debug symbols, you will see the assembler code for the routine shown, but if your kernel has debug symbols the C code will also be available. (Debug symbols can be enabled in the kernel hacking menu of the menu configuration.) For example:

```
$ objdump -r -S -l --disassemble net/dccp/ipv4.o
```

Note: You need to be at the top level of the kernel tree for this to pick up your C files.

If you don't have access to the source code you can still debug some crash dumps using the following method (example crash dump output as shown by Dave Miller):

```
EIP is at  +0x14/0x4c0
 ...
Code: 44 24 04 e8 6f 05 00 00 e9 e8 fe ff ff 8d 76 00 8d bc 27 00 00
00 00 55 57  56 53 81 ec bc 00 00 00 8b ac 24 d0 00 00 00 8b 5d 08
<8b> 83 3c 01 00 00 89 44  24 14 8b 45 28 85 c0 89 44 24 18 0f 85

Put the bytes into a "foo.s" file like this:

       .text
       .globl foo
foo:
       .byte  .... /* bytes from Code: part of OOPS dump */

Compile it with "gcc -c -o foo.o foo.s" then look at the output of
"objdump --disassemble foo.o".

Output:

ip_queue_xmit:
    push       %ebp
    push       %edi
    push       %esi
    push       %ebx
    sub        $0xbc, %esp
    mov        0xd0(%esp), %ebp        ! %ebp = arg0 (skb)
    mov        0x8(%ebp), %ebx         ! %ebx = skb->sk
    mov        0x13c(%ebx), %eax       ! %eax = inet_sk(sk)->opt
```

file:*scripts/decodecode* can be used to automate most of this, depending on what CPU architecture is being debugged.

11.4 Reporting the bug

Once you find where the bug happened, by inspecting its location, you could either try to fix it yourself or report it upstream.

In order to report it upstream, you should identify the mailing list used for the development of the affected code. This can be done by using the `get_maintainer.pl` script.

For example, if you find a bug at the gspca's sonixj.c file, you can get its maintainers with:

```
$ ./scripts/get_maintainer.pl -f drivers/media/usb/gspca/sonixj.c
Hans Verkuil <hverkuil@xs4all.nl> (odd fixer:GSPCA USB WEBCAM DRIVER,commit_
↪signer:1/1=100%)
Mauro Carvalho Chehab <mchehab@kernel.org> (maintainer:MEDIA INPUT␣
↪INFRASTRUCTURE (V4L/DVB),commit_signer:1/1=100%)
Tejun Heo <tj@kernel.org> (commit_signer:1/1=100%)
Bhaktipriya Shridhar <bhaktipriya96@gmail.com> (commit_signer:1/1=100%,
↪authored:1/1=100%,added_lines:4/4=100%,removed_lines:9/9=100%)
linux-media@vger.kernel.org (open list:GSPCA USB WEBCAM DRIVER)
```

```
linux-kernel@vger.kernel.org (open list)
```

Please notice that it will point to:

- The last developers that touched the source code (if this is done inside a git tree). On the above example, Tejun and Bhaktipriya (in this specific case, none really involved on the development of this file);
- The driver maintainer (Hans Verkuil);
- The subsystem maintainer (Mauro Carvalho Chehab);
- The driver and/or subsystem mailing list (linux-media@vger.kernel.org);
- the Linux Kernel mailing list (linux-kernel@vger.kernel.org).

Usually, the fastest way to have your bug fixed is to report it to mailing list used for the development of the code (linux-media ML) copying the driver maintainer (Hans).

If you are totally stumped as to whom to send the report, and `get_maintainer.pl` didn't provide you anything useful, send it to linux-kernel@vger.kernel.org.

Thanks for your help in making Linux as stable as humanly possible.

11.5 Fixing the bug

If you know programming, you could help us by not only reporting the bug, but also providing us with a solution. After all, open source is about sharing what you do and don't you want to be recognised for your genius?

If you decide to take this way, once you have worked out a fix please submit it upstream.

Please do read Documentation/process/submitting-patches.rst though to help your code get accepted.

11.6 Notes on Oops tracing with `klogd`

In order to help Linus and the other kernel developers there has been substantial support incorporated into `klogd` for processing protection faults. In order to have full support for address resolution at least version 1.3-pl3 of the `sysklogd` package should be used.

When a protection fault occurs the `klogd` daemon automatically translates important addresses in the kernel log messages to their symbolic equivalents. This translated kernel message is then forwarded through whatever reporting mechanism `klogd` is using. The protection fault message can be simply cut out of the message files and forwarded to the kernel developers.

Two types of address resolution are performed by `klogd`. The first is static translation and the second is dynamic translation. Static translation uses the System.map file. In order to do static translation the `klogd` daemon must be able to find a system map file at daemon initialization time. See the klogd man page for information on how `klogd` searches for map files.

Dynamic address translation is important when kernel loadable modules are being used. Since memory for kernel modules is allocated from the kernel's dynamic memory pools there are no fixed locations for either the start of the module or for functions and symbols in the module.

The kernel supports system calls which allow a program to determine which modules are loaded and their location in memory. Using these system calls the klogd daemon builds a symbol table which can be used to debug a protection fault which occurs in a loadable kernel module.

At the very minimum klogd will provide the name of the module which generated the protection fault. There may be additional symbolic information available if the developer of the loadable module chose to export symbol information from the module.

Since the kernel module environment can be dynamic there must be a mechanism for notifying the `klogd` daemon when a change in module environment occurs. There are command line options available which allow klogd to signal the currently executing daemon that symbol information should be refreshed. See the `klogd` manual page for more information.

A patch is included with the sysklogd distribution which modifies the `modules-2.0.0` package to automatically signal klogd whenever a module is loaded or unloaded. Applying this patch provides essentially seamless support for debugging protection faults which occur with kernel loadable modules.

The following is an example of a protection fault in a loadable module processed by `klogd`:

```
Aug 29 09:51:01 blizard kernel: Unable to handle kernel paging request at␣
↪virtual address f15e97cc
Aug 29 09:51:01 blizard kernel: current->tss.cr3 = 0062d000, %cr3 = 0062d000
Aug 29 09:51:01 blizard kernel: *pde = 00000000
Aug 29 09:51:01 blizard kernel: Oops: 0002
Aug 29 09:51:01 blizard kernel: CPU:    0
Aug 29 09:51:01 blizard kernel: EIP:    0010:[oops:_oops+16/3868]
Aug 29 09:51:01 blizard kernel: EFLAGS: 00010212
Aug 29 09:51:01 blizard kernel: eax: 315e97cc   ebx: 003a6f80   ecx: 001be77b ␣
↪ edx: 00237c0c
Aug 29 09:51:01 blizard kernel: esi: 00000000   edi: bffffdb3   ebp: 00589f90 ␣
↪ esp: 00589f8c
Aug 29 09:51:01 blizard kernel: ds: 0018   es: 0018   fs: 002b   gs: 002b  ␣
↪ss: 0018
Aug 29 09:51:01 blizard kernel: Process oops_test (pid: 3374, process nr: 21,␣
↪stackpage=00589000)
Aug 29 09:51:01 blizard kernel: Stack: 315e97cc 00589f98 0100b0b4 bffffed4␣
↪0012e38e 00240c64 003a6f80 00000001
Aug 29 09:51:01 blizard kernel:        00000000 00237810 bfffff00 0010a7fa␣
↪00000003 00000001 00000000 bfffff00
Aug 29 09:51:01 blizard kernel:        bffffdb3 bffffed4 ffffffda 0000002b␣
↪0007002b 0000002b 0000002b 00000036
Aug 29 09:51:01 blizard kernel: Call Trace: [oops:_oops_ioctl+48/80] [_sys_
↪ioctl+254/272] [_system_call+82/128]
Aug 29 09:51:01 blizard kernel: Code: c7 00 05 00 00 00 eb 08 90 90 90 90 90␣
↪90 90 90 89 ec 5d c3
```

```
Dr. G.W. Wettstein           Oncology Research Div. Computing Facility
Roger Maris Cancer Center    INTERNET: greg@wind.rmcc.com
820 4th St. N.
Fargo, ND  58122
Phone: 701-234-7556
```

CHAPTER

TWELVE

BISECTING A BUG

Last updated: 28 October 2016

12.1 Introduction

Always try the latest kernel from kernel.org and build from source. If you are not confident in doing that please report the bug to your distribution vendor instead of to a kernel developer.

Finding bugs is not always easy. Have a go though. If you can't find it don't give up. Report as much as you have found to the relevant maintainer. See MAINTAINERS for who that is for the subsystem you have worked on.

Before you submit a bug report read '*Reporting issues*'.

12.2 Devices not appearing

Often this is caused by udev/systemd. Check that first before blaming it on the kernel.

12.3 Finding patch that caused a bug

Using the provided tools with `git` makes finding bugs easy provided the bug is reproducible.

Steps to do it:

- build the Kernel from its git source
- start bisect with[1]:

```
$ git bisect start
```

- mark the broken changeset with:

```
$ git bisect bad [commit]
```

- mark a changeset where the code is known to work with:

```
$ git bisect good [commit]
```

[1] You can, optionally, provide both good and bad arguments at git start with `git bisect start [BAD] [GOOD]`

- rebuild the Kernel and test
- interact with git bisect by using either:

```
$ git bisect good
```

  or:

```
$ git bisect bad
```

  depending if the bug happened on the changeset you're testing
- After some interactions, git bisect will give you the changeset that likely caused the bug.
- For example, if you know that the current version is bad, and version 4.8 is good, you could do:

```
$ git bisect start
$ git bisect bad                 # Current version is bad
$ git bisect good v4.8
```

For further references, please read:

- The man page for `git-bisect`
- Fighting regressions with git bisect
- Fully automated bisecting with "git bisect run"
- Using Git bisect to figure out when brokenness was introduced

CHAPTER

THIRTEEN

TAINTED KERNELS

The kernel will mark itself as 'tainted' when something occurs that might be relevant later when investigating problems. Don't worry too much about this, most of the time it's not a problem to run a tainted kernel; the information is mainly of interest once someone wants to investigate some problem, as its real cause might be the event that got the kernel tainted. That's why bug reports from tainted kernels will often be ignored by developers, hence try to reproduce problems with an untainted kernel.

Note the kernel will remain tainted even after you undo what caused the taint (i.e. unload a proprietary kernel module), to indicate the kernel remains not trustworthy. That's also why the kernel will print the tainted state when it notices an internal problem (a 'kernel bug'), a recoverable error ('kernel oops') or a non-recoverable error ('kernel panic') and writes debug information about this to the logs `dmesg` outputs. It's also possible to check the tainted state at runtime through a file in `/proc/`.

13.1 Tainted flag in bugs, oops or panics messages

You find the tainted state near the top in a line starting with 'CPU:'; if or why the kernel was tainted is shown after the Process ID ('PID:') and a shortened name of the command ('Comm:') that triggered the event:

```
BUG: unable to handle kernel NULL pointer dereference at 0000000000000000
Oops: 0002 [#1] SMP PTI
CPU: 0 PID: 4424 Comm: insmod Tainted: P        W  O      4.20.0-0.rc6.fc30 #1
Hardware name: Red Hat KVM, BIOS 0.5.1 01/01/2011
RIP: 0010:my_oops_init+0x13/0x1000 [kpanic]
[...]
```

You'll find a 'Not tainted: ' there if the kernel was not tainted at the time of the event; if it was, then it will print 'Tainted: ' and characters either letters or blanks. In above example it looks like this:

```
Tainted: P        W  O
```

The meaning of those characters is explained in the table below. In this case the kernel got tainted earlier because a proprietary Module (`P`) was loaded, a warning occurred (`W`), and an externally-built module was loaded (`O`). To decode other letters use the table below.

13.2 Decoding tainted state at runtime

At runtime, you can query the tainted state by reading `cat /proc/sys/kernel/tainted`. If that returns `0`, the kernel is not tainted; any other number indicates the reasons why it is. The easiest way to decode that number is the script `tools/debugging/kernel-chktaint`, which your distribution might ship as part of a package called `linux-tools` or `kernel-tools`; if it doesn't you can download the script from git.kernel.org and execute it with `sh kernel-chktaint`, which would print something like this on the machine that had the statements in the logs that were quoted earlier:

```
Kernel is Tainted for following reasons:
 * Proprietary module was loaded (#0)
 * Kernel issued warning (#9)
 * Externally-built ('out-of-tree') module was loaded  (#12)
See Documentation/admin-guide/tainted-kernels.rst in the Linux kernel or
 https://www.kernel.org/doc/html/latest/admin-guide/tainted-kernels.html for
 a more details explanation of the various taint flags.
Raw taint value as int/string: 4609/'P       W  O     '
```

You can try to decode the number yourself. That's easy if there was only one reason that got your kernel tainted, as in this case you can find the number with the table below. If there were multiple reasons you need to decode the number, as it is a bitfield, where each bit indicates the absence or presence of a particular type of taint. It's best to leave that to the aforementioned script, but if you need something quick you can use this shell command to check which bits are set:

```
$ for i in $(seq 18); do echo $(($i-1)) $(($(cat /proc/sys/kernel/tainted)>>(
↪$i-1)&1));done
```

13.3 Table for decoding tainted state

Bit	Log	Number	Reason that got the kernel tainted
0	G/P	1	proprietary module was loaded
1	_/F	2	module was force loaded
2	_/S	4	kernel running on an out of specification system
3	_/R	8	module was force unloaded
4	_/M	16	processor reported a Machine Check Exception (MCE)
5	_/B	32	bad page referenced or some unexpected page flags
6	_/U	64	taint requested by userspace application
7	_/D	128	kernel died recently, i.e. there was an OOPS or BUG
8	_/A	256	ACPI table overridden by user
9	_/W	512	kernel issued warning
10	_/C	1024	staging driver was loaded
11	_/I	2048	workaround for bug in platform firmware applied
12	_/O	4096	externally-built (”out-of-tree”) module was loaded
13	_/E	8192	unsigned module was loaded
14	_/L	16384	soft lockup occurred
15	_/K	32768	kernel has been live patched
16	_/X	65536	auxiliary taint, defined for and used by distros
17	_/T	131072	kernel was built with the struct randomization plugin
18	_/N	262144	an in-kernel test has been run

Note: The character _ is representing a blank in this table to make reading easier.

13.4 More detailed explanation for tainting

0) `G` if all modules loaded have a GPL or compatible license, `P` if any proprietary module has been loaded. Modules without a MODULE_LICENSE or with a MODULE_LICENSE that is not recognised by insmod as GPL compatible are assumed to be proprietary.

1) `F` if any module was force loaded by `insmod -f`, `' '` if all modules were loaded normally.

2) `S` if the kernel is running on a processor or system that is out of specification: hardware has been put into an unsupported configuration, therefore proper execution cannot be guaranteed. Kernel will be tainted if, for example:

   - on x86: PAE is forced through forcepae on intel CPUs (such as Pentium M) which do not report PAE but may have a functional implementation, an SMP kernel is running on non officially capable SMP Athlon CPUs, MSRs are being poked at from userspace.
   - on arm: kernel running on certain CPUs (such as Keystone 2) without having certain kernel features enabled.
   - on arm64: there are mismatched hardware features between CPUs, the bootloader has booted CPUs in different modes.
   - certain drivers are being used on non supported architectures (such as scsi/snic on something else than x86_64, scsi/ips on non x86/x86_64/itanium, have broken firmware settings for the irqchip/irq-gic on arm64 ...).

- x86/x86_64: Microcode late loading is dangerous and will result in tainting the kernel. It requires that all CPUs rendezvous to make sure the update happens when the system is as quiescent as possible. However, a higher priority MCE/SMI/NMI can move control flow away from that rendezvous and interrupt the update, which can be detrimental to the machine.

3) `R` if a module was force unloaded by `rmmod -f`, `' '` if all modules were unloaded normally.
4) `M` if any processor has reported a Machine Check Exception, `' '` if no Machine Check Exceptions have occurred.
5) `B` If a page-release function has found a bad page reference or some unexpected page flags. This indicates a hardware problem or a kernel bug; there should be other information in the log indicating why this tainting occurred.
6) `U` if a user or user application specifically requested that the Tainted flag be set, `' '` otherwise.
7) `D` if the kernel has died recently, i.e. there was an OOPS or BUG.
8) `A` if an ACPI table has been overridden.
9) `W` if a warning has previously been issued by the kernel. (Though some warnings may set more specific taint flags.)
10) `C` if a staging driver has been loaded.
11) `I` if the kernel is working around a severe bug in the platform firmware (BIOS or similar).
12) `O` if an externally-built (”out-of-tree”) module has been loaded.
13) `E` if an unsigned module has been loaded in a kernel supporting module signature.
14) `L` if a soft lockup has previously occurred on the system.
15) `K` if the kernel has been live patched.
16) `X` Auxiliary taint, defined for and used by Linux distributors.
17) `T` Kernel was build with the randstruct plugin, which can intentionally produce extremely unusual kernel structure layouts (even performance pathological ones), which is important to know when debugging. Set at build time.

CHAPTER

FOURTEEN

RAMOOPS OOPS/PANIC LOGGER

Sergiu Iordache <sergiu@chromium.org>

Updated: 10 Feb 2021

14.1 Introduction

Ramoops is an oops/panic logger that writes its logs to RAM before the system crashes. It works by logging oopses and panics in a circular buffer. Ramoops needs a system with persistent RAM so that the content of that area can survive after a restart.

14.2 Ramoops concepts

Ramoops uses a predefined memory area to store the dump. The start and size and type of the memory area are set using three variables:

- `mem_address` for the start
- `mem_size` for the size. The memory size will be rounded down to a power of two.
- `mem_type` to specify if the memory type (default is pgprot_writecombine).

Typically the default value of `mem_type=0` should be used as that sets the pstore mapping to pgprot_writecombine. Setting `mem_type=1` attempts to use `pgprot_noncached`, which only works on some platforms. This is because pstore depends on atomic operations. At least on ARM, pgprot_noncached causes the memory to be mapped strongly ordered, and atomic operations on strongly ordered memory are implementation defined, and won't work on many ARMs such as omaps. Setting `mem_type=2` attempts to treat the memory region as normal memory, which enables full cache on it. This can improve the performance.

The memory area is divided into `record_size` chunks (also rounded down to power of two) and each kmesg dump writes a `record_size` chunk of information.

Limiting which kinds of kmsg dumps are stored can be controlled via the `max_reason` value, as defined in include/linux/kmsg_dump.h's `enum kmsg_dump_reason`. For example, to store both Oopses and Panics, `max_reason` should be set to 2 (KMSG_DUMP_OOPS), to store only Panics `max_reason` should be set to 1 (KMSG_DUMP_PANIC). Setting this to 0 (KMSG_DUMP_UNDEF), means the reason filtering will be controlled by the `printk.always_kmsg_dump` boot param: if unset, it'll be KMSG_DUMP_OOPS, otherwise KMSG_DUMP_MAX.

The module uses a counter to record multiple dumps but the counter gets reset on restart (i.e. new dumps after the restart will overwrite old ones).

Ramoops also supports software ECC protection of persistent memory regions. This might be useful when a hardware reset was used to bring the machine back to life (i.e. a watchdog triggered). In such cases, RAM may be somewhat corrupt, but usually it is restorable.

14.3 Setting the parameters

Setting the ramoops parameters can be done in several different manners:

A. Use the module parameters (which have the names of the variables described as before). For quick debugging, you can also reserve parts of memory during boot and then use the reserved memory for ramoops. For example, assuming a machine with > 128 MB of memory, the following kernel command line will tell the kernel to use only the first 128 MB of memory, and place ECC-protected ramoops region at 128 MB boundary:

```
mem=128M ramoops.mem_address=0x8000000 ramoops.ecc=1
```

B. Use Device Tree bindings, as described in `Documentation/devicetree/bindings/reserved-memory/ramoops.yaml`. For example:

```
reserved-memory {
        #address-cells = <2>;
        #size-cells = <2>;
        ranges;

        ramoops@8f000000 {
                compatible = "ramoops";
                reg = <0 0x8f000000 0 0x100000>;
                record-size = <0x4000>;
                console-size = <0x4000>;
        };
};
```

C. Use a platform device and set the platform data. The parameters can then be set through that platform data. An example of doing that is:

```
#include <linux/pstore_ram.h>
[...]

static struct ramoops_platform_data ramoops_data = {
      .mem_size               = <...>,
      .mem_address            = <...>,
      .mem_type               = <...>,
      .record_size            = <...>,
      .max_reason             = <...>,
      .ecc                    = <...>,
};
```

```
static struct platform_device ramoops_dev = {
      .name = "ramoops",
      .dev = {
              .platform_data = &ramoops_data,
      },
};

[... inside a function ...]
int ret;

ret = platform_device_register(&ramoops_dev);
if (ret) {
      printk(KERN_ERR "unable to register platform device\n");
      return ret;
}
```

You can specify either RAM memory or peripheral devices' memory. However, when specifying RAM, be sure to reserve the memory by issuing memblock_reserve() very early in the architecture code, e.g.:

```
#include <linux/memblock.h>

memblock_reserve(ramoops_data.mem_address, ramoops_data.mem_size);
```

14.4 Dump format

The data dump begins with a header, currently defined as ==== followed by a timestamp and a new line. The dump then continues with the actual data.

14.5 Reading the data

The dump data can be read from the pstore filesystem. The format for these files is `dmesg-ramoops-N`, where N is the record number in memory. To delete a stored record from RAM, simply unlink the respective pstore file.

14.6 Persistent function tracing

Persistent function tracing might be useful for debugging software or hardware related hangs. The functions call chain log is stored in a `ftrace-ramoops` file. Here is an example of usage:

```
# mount -t debugfs debugfs /sys/kernel/debug/
# echo 1 > /sys/kernel/debug/pstore/record_ftrace
# reboot -f
[...]
# mount -t pstore pstore /mnt/
# tail /mnt/ftrace-ramoops
```

```
0 ffffffff8101ea64  ffffffff8101bcda  native_apic_mem_read <- disconnect_bsp_
↪APIC+0x6a/0xc0
0 ffffffff8101ea44  ffffffff8101bcf6  native_apic_mem_write <- disconnect_bsp_
↪APIC+0x86/0xc0
0 ffffffff81020084  ffffffff8101a4b5  hpet_disable <- native_machine_
↪shutdown+0x75/0x90
0 ffffffff81005f94  ffffffff8101a4bb  iommu_shutdown_noop <- native_machine_
↪shutdown+0x7b/0x90
0 ffffffff8101a6a1  ffffffff8101a437  native_machine_emergency_restart <-␣
↪native_machine_restart+0x37/0x40
0 ffffffff811f9876  ffffffff8101a73a  acpi_reboot <- native_machine_emergency_
↪restart+0xaa/0x1e0
0 ffffffff8101a514  ffffffff8101a772  mach_reboot_fixups <- native_machine_
↪emergency_restart+0xe2/0x1e0
0 ffffffff811d9c54  ffffffff8101a7a0  __const_udelay <- native_machine_
↪emergency_restart+0x110/0x1e0
0 ffffffff811d9c34  ffffffff811d9c80  __delay <- __const_udelay+0x30/0x40
0 ffffffff811d9d14  ffffffff811d9c3f  delay_tsc <- __delay+0xf/0x20
```

CHAPTER

FIFTEEN

DYNAMIC DEBUG

15.1 Introduction

Dynamic debug allows you to dynamically enable/disable kernel debug-print code to obtain additional kernel information.

If `/proc/dynamic_debug/control` exists, your kernel has dynamic debug. You'll need root access (sudo su) to use this.

Dynamic debug provides:

- a Catalog of all *prdbgs* in your kernel. `cat /proc/dynamic_debug/control` to see them.
- a Simple query/command language to alter *prdbgs* by selecting on any combination of 0 or 1 of:
  - source filename
  - function name
  - line number (including ranges of line numbers)
  - module name
  - format string
  - class name (as known/declared by each module)

15.2 Viewing Dynamic Debug Behaviour

You can view the currently configured behaviour in the *prdbg* catalog:

```
:#> head -n7 /proc/dynamic_debug/control
# filename:lineno [module]function flags format
init/main.c:1179 [main]initcall_blacklist =_ "blacklisting initcall %s\012
init/main.c:1218 [main]initcall_blacklisted =_ "initcall %s blacklisted\012"
init/main.c:1424 [main]run_init_process =_ "  with arguments:\012"
init/main.c:1426 [main]run_init_process =_ "    %s\012"
init/main.c:1427 [main]run_init_process =_ "  with environment:\012"
init/main.c:1429 [main]run_init_process =_ "    %s\012"
```

The 3rd space-delimited column shows the current flags, preceded by a = for easy use with grep/cut. `=p` shows enabled callsites.

15.3 Controlling dynamic debug Behaviour

The behaviour of *prdbg* sites are controlled by writing query/commands to the control file. Example:

```
# grease the interface
:#> alias ddcmd='echo $* > /proc/dynamic_debug/control'

:#> ddcmd '-p; module main func run* +p'
:#> grep =p /proc/dynamic_debug/control
init/main.c:1424 [main]run_init_process =p "  with arguments:\012"
init/main.c:1426 [main]run_init_process =p "    %s\012"
init/main.c:1427 [main]run_init_process =p "  with environment:\012"
init/main.c:1429 [main]run_init_process =p "    %s\012"
```

Error messages go to console/syslog:

```
:#> ddcmd mode foo +p
dyndbg: unknown keyword "mode"
dyndbg: query parse failed
bash: echo: write error: Invalid argument
```

If debugfs is also enabled and mounted, `dynamic_debug/control` is also under the mount-dir, typically `/sys/kernel/debug/`.

15.4 Command Language Reference

At the basic lexical level, a command is a sequence of words separated by spaces or tabs. So these are all equivalent:

```
:#> ddcmd file svcsock.c line 1603 +p
:#> ddcmd "file svcsock.c line 1603 +p"
:#> ddcmd '  file   svcsock.c     line  1603 +p  '
```

Command submissions are bounded by a write() system call. Multiple commands can be written together, separated by `;` or `\n`:

```
:#> ddcmd "func pnpacpi_get_resources +p; func pnp_assign_mem +p"
:#> ddcmd <<"EOC"
func pnpacpi_get_resources +p
func pnp_assign_mem +p
EOC
:#> cat query-batch-file > /proc/dynamic_debug/control
```

You can also use wildcards in each query term. The match rule supports * (matches zero or more characters) and ? (matches exactly one character). For example, you can match all usb drivers:

```
:#> ddcmd file "drivers/usb/*" +p     # "" to suppress shell expansion
```

Syntactically, a command is pairs of keyword values, followed by a flags change or setting:

```
command ::= match-spec* flags-spec
```

The match-spec's select *prdbgs* from the catalog, upon which to apply the flags-spec, all constraints are ANDed together. An absent keyword is the same as keyword "*".

A match specification is a keyword, which selects the attribute of the callsite to be compared, and a value to compare against. Possible keywords are::

```
match-spec ::= 'func' string |
               'file' string |
               'module' string |
               'format' string |
               'class' string |
               'line' line-range

line-range ::= lineno |
               '-'lineno |
               lineno'-' |
               lineno'-'lineno

lineno ::= unsigned-int
```

Note: `line-range` cannot contain space, e.g. "1-30" is valid range but "1 - 30" is not.

The meanings of each keyword are:

func
The given string is compared against the function name of each callsite. Example:

```
func svc_tcp_accept
func *recv*             # in rfcomm, bluetooth, ping, tcp
```

file
The given string is compared against either the src-root relative pathname, or the basename of the source file of each callsite. Examples:

```
file svcsock.c
file kernel/freezer.c   # ie column 1 of control file
file drivers/usb/*      # all callsites under it
file inode.c:start_*    # parse :tail as a func (above)
file inode.c:1-100      # parse :tail as a line-range (above)
```

module
The given string is compared against the module name of each callsite. The module name is the string as seen in `lsmod`, i.e. without the directory or the `.ko` suffix and with `-` changed to `_`. Examples:

```
module sunrpc
module nfsd
module drm*     # both drm, drm_kms_helper
```

format

The given string is searched for in the dynamic debug format string. Note that the string does not need to match the entire format, only some part. Whitespace and other special characters can be escaped using C octal character escape `\ooo` notation, e.g. the space character is `\040`. Alternatively, the string can be enclosed in double quote characters (`"`) or single quote characters (`'`). Examples:

```
format svcrdma:         // many of the NFS/RDMA server pr_debugs
format readahead        // some pr_debugs in the readahead cache
format nfsd:\040SETATTR // one way to match a format with whitespace
format "nfsd: SETATTR"  // a neater way to match a format with whitespace
format 'nfsd: SETATTR'  // yet another way to match a format with␣
↪whitespace
```

class

The given class_name is validated against each module, which may have declared a list of known class_names. If the class_name is found for a module, callsite & class matching and adjustment proceeds. Examples:

```
class DRM_UT_KMS        # a DRM.debug category
class JUNK              # silent non-match
// class TLD_*          # NOTICE: no wildcard in class names
```

line

The given line number or range of line numbers is compared against the line number of each `pr_debug()` callsite. A single line number matches the callsite line number exactly. A range of line numbers matches any callsite between the first and last line number inclusive. An empty first number means the first line in the file, an empty last line number means the last line number in the file. Examples:

```
line 1603           // exactly line 1603
line 1600-1605      // the six lines from line 1600 to line 1605
line -1605          // the 1605 lines from line 1 to line 1605
line 1600-          // all lines from line 1600 to the end of the file
```

The flags specification comprises a change operation followed by one or more flag characters. The change operation is one of the characters:

```
-    remove the given flags
+    add the given flags
=    set the flags to the given flags
```

The flags are:

```
p    enables the pr_debug() callsite.
_    enables no flags.

Decorator flags add to the message-prefix, in order:
t    Include thread ID, or <intr>
m    Include module name
f    Include the function name
s    Include the source file name
```

```
l    Include line number
```

For `print_hex_dump_debug()` and `print_hex_dump_bytes()`, only the `p` flag has meaning, other flags are ignored.

Note the regexp `^[-+=][fslmpt_]+$` matches a flags specification. To clear all flags at once, use `=_` or `-fslmpt`.

15.5 Debug messages during Boot Process

To activate debug messages for core code and built-in modules during the boot process, even before userspace and debugfs exists, use `dyndbg="QUERY"` or `module.dyndbg="QUERY"`. QUERY follows the syntax described above, but must not exceed 1023 characters. Your bootloader may impose lower limits.

These `dyndbg` params are processed just after the ddebug tables are processed, as part of the early_initcall. Thus you can enable debug messages in all code run after this early_initcall via this boot parameter.

On an x86 system for example ACPI enablement is a subsys_initcall and:

```
dyndbg="file ec.c +p"
```

will show early Embedded Controller transactions during ACPI setup if your machine (typically a laptop) has an Embedded Controller. PCI (or other devices) initialization also is a hot candidate for using this boot parameter for debugging purposes.

If `foo` module is not built-in, `foo.dyndbg` will still be processed at boot time, without effect, but will be reprocessed when module is loaded later. Bare `dyndbg=` is only processed at boot.

15.6 Debug Messages at Module Initialization Time

When `modprobe foo` is called, modprobe scans `/proc/cmdline` for `foo.params`, strips `foo.`, and passes them to the kernel along with params given in modprobe args or `/etc/modprobe.d/*.conf` files, in the following order:

1. parameters given via `/etc/modprobe.d/*.conf`:

   ```
   options foo dyndbg=+pt
   options foo dyndbg # defaults to +p
   ```

2. `foo.dyndbg` as given in boot args, `foo.` is stripped and passed:

   ```
   foo.dyndbg=" func bar +p; func buz +mp"
   ```

3. args to modprobe:

   ```
   modprobe foo dyndbg==pmf # override previous settings
   ```

These `dyndbg` queries are applied in order, with last having final say. This allows boot args to override or modify those from `/etc/modprobe.d` (sensible, since 1 is system wide, 2 is kernel or boot specific), and modprobe args to override both.

In the `foo.dyndbg="QUERY"` form, the query must exclude `module foo`. `foo` is extracted from the param-name, and applied to each query in `QUERY`, and only 1 match-spec of each type is allowed.

The `dyndbg` option is a "fake" module parameter, which means:

- modules do not need to define it explicitly
- every module gets it tacitly, whether they use pr_debug or not
- it doesn't appear in `/sys/module/$module/parameters/` To see it, grep the control file, or inspect `/proc/cmdline`.

For `CONFIG_DYNAMIC_DEBUG` kernels, any settings given at boot-time (or enabled by `-DDEBUG` flag during compilation) can be disabled later via the debugfs interface if the debug messages are no longer needed:

```
echo "module module_name -p" > /proc/dynamic_debug/control
```

15.7 Examples

```
// enable the message at line 1603 of file svcsock.c
:#> ddcmd 'file svcsock.c line 1603 +p'

// enable all the messages in file svcsock.c
:#> ddcmd 'file svcsock.c +p'

// enable all the messages in the NFS server module
:#> ddcmd 'module nfsd +p'

// enable all 12 messages in the function svc_process()
:#> ddcmd 'func svc_process +p'

// disable all 12 messages in the function svc_process()
:#> ddcmd 'func svc_process -p'

// enable messages for NFS calls READ, READLINK, READDIR and READDIR+.
:#> ddcmd 'format "nfsd: READ" +p'

// enable messages in files of which the paths include string "usb"
:#> ddcmd 'file *usb* +p'

// enable all messages
:#> ddcmd '+p'

// add module, function to all enabled messages
:#> ddcmd '+mf'

// boot-args example, with newlines and comments for readability
Kernel command line: ...
  // see what's going on in dyndbg=value processing
  dynamic_debug.verbose=3
```

```
  // enable pr_debugs in the btrfs module (can be builtin or loadable)
  btrfs.dyndbg="+p"
  // enable pr_debugs in all files under init/
  // and the function parse_one, #cmt is stripped
  dyndbg="file init/* +p #cmt ; func parse_one +p"
  // enable pr_debugs in 2 functions in a module loaded later
  pc87360.dyndbg="func pc87360_init_device +p; func pc87360_find +p"
```

15.8 Kernel Configuration

Dynamic Debug is enabled via kernel config items:

```
CONFIG_DYNAMIC_DEBUG=y        # build catalog, enables CORE
CONFIG_DYNAMIC_DEBUG_CORE=y   # enable mechanics only, skip catalog
```

If you do not want to enable dynamic debug globally (i.e. in some embedded system), you may set `CONFIG_DYNAMIC_DEBUG_CORE` as basic support of dynamic debug and add `ccflags := -DDYNAMIC_DEBUG_MODULE` into the Makefile of any modules which you'd like to dynamically debug later.

15.9 Kernel *prdbg* API

The following functions are cataloged and controllable when dynamic debug is enabled:

```
pr_debug()
dev_dbg()
print_hex_dump_debug()
print_hex_dump_bytes()
```

Otherwise, they are off by default; `ccflags += -DDEBUG` or `#define DEBUG` in a source file will enable them appropriately.

If `CONFIG_DYNAMIC_DEBUG` is not set, `print_hex_dump_debug()` is just a shortcut for `print_hex_dump(KERN_DEBUG)`.

For `print_hex_dump_debug()`/`print_hex_dump_bytes()`, format string is its `prefix_str` argument, if it is constant string; or `hexdump` in case `prefix_str` is built dynamically.

CHAPTER

SIXTEEN

EXPLAINING THE ”NO WORKING INIT FOUND.” BOOT HANG MESSAGE

Authors
Andreas Mohr <andi at lisas period de> Cristian Souza <cristianmsbr at gmail period com>

This document provides some high-level reasons for failure (listed roughly in order of execution) to load the init binary.

1) **Unable to mount root FS**: Set ”debug” kernel parameter (in bootloader config file or CONFIG_CMDLINE) to get more detailed kernel messages.

2) **init binary doesn't exist on rootfs**: Make sure you have the correct root FS type (and `root=` kernel parameter points to the correct partition), required drivers such as storage hardware (such as SCSI or USB!) and filesystem (ext3, jffs2, etc.) are builtin (alternatively as modules, to be pre-loaded by an initrd).

3) **Broken console device**: Possibly a conflict in `console= setup` --> initial console unavailable. E.g. some serial consoles are unreliable due to serial IRQ issues (e.g. missing interrupt-based configuration). Try using a different `console= device` or e.g. `netconsole=`.

4) **Binary exists but dependencies not available**: E.g. required library dependencies of the init binary such as `/lib/ld-linux.so.2` missing or broken. Use `readelf -d <INIT>|grep NEEDED` to find out which libraries are required.

5) **Binary cannot be loaded**: Make sure the binary's architecture matches your hardware. E.g. i386 vs. x86_64 mismatch, or trying to load x86 on ARM hardware. In case you tried loading a non-binary file here (shell script?), you should make sure that the script specifies an interpreter in its shebang header line (`#!/...`) that is fully working (including its library dependencies). And before tackling scripts, better first test a simple non-script binary such as `/bin/sh` and confirm its successful execution. To find out more, add code `to init/main.c` to display kernel_execve()s return values.

Please extend this explanation whenever you find new failure causes (after all loading the init binary is a CRITICAL and hard transition step which needs to be made as painless as possible), then submit a patch to LKML. Further TODOs:

- Implement the various `run_init_process()` invocations via a struct array which can then store the `kernel_execve()` result value and on failure log it all by iterating over **all** results (very important usability fix).
- Try to make the implementation itself more helpful in general, e.g. by providing additional error messages at affected places.

CHAPTER

SEVENTEEN

DOCUMENTATION FOR KDUMP - THE KEXEC-BASED CRASH DUMPING SOLUTION

This document includes overview, setup and installation, and analysis information.

17.1 Documentation for Kdump - The kexec-based Crash Dumping Solution

This document includes overview, setup, installation, and analysis information.

17.1.1 Overview

Kdump uses kexec to quickly boot to a dump-capture kernel whenever a dump of the system kernel's memory needs to be taken (for example, when the system panics). The system kernel's memory image is preserved across the reboot and is accessible to the dump-capture kernel.

You can use common commands, such as cp, scp or makedumpfile to copy the memory image to a dump file on the local disk, or across the network to a remote system.

Kdump and kexec are currently supported on the x86, x86_64, ppc64, s390x, arm and arm64 architectures.

When the system kernel boots, it reserves a small section of memory for the dump-capture kernel. This ensures that ongoing Direct Memory Access (DMA) from the system kernel does not corrupt the dump-capture kernel. The kexec -p command loads the dump-capture kernel into this reserved memory.

On x86 machines, the first 640 KB of physical memory is needed for boot, regardless of where the kernel loads. For simpler handling, the whole low 1M is reserved to avoid any later kernel or device driver writing data into this area. Like this, the low 1M can be reused as system RAM by kdump kernel without extra handling.

On PPC64 machines first 32KB of physical memory is needed for booting regardless of where the kernel is loaded and to support 64K page size kexec backs up the first 64KB memory.

For s390x, when kdump is triggered, the crashkernel region is exchanged with the region [0, crashkernel region size] and then the kdump kernel runs in [0, crashkernel region size]. Therefore no relocatable kernel is needed for s390x.

All of the necessary information about the system kernel's core image is encoded in the ELF format, and stored in a reserved area of memory before a crash. The physical address of the start of the ELF header is passed to the dump-capture kernel through the elfcorehdr= boot

parameter. Optionally the size of the ELF header can also be passed when using the elfcorehdr=[size[KMG]@]offset[KMG] syntax.

With the dump-capture kernel, you can access the memory image through /proc/vmcore. This exports the dump as an ELF-format file that you can write out using file copy commands such as cp or scp. You can also use makedumpfile utility to analyze and write out filtered contents with options, e.g with '-d 31' it will only write out kernel data. Further, you can use analysis tools such as the GNU Debugger (GDB) and the Crash tool to debug the dump file. This method ensures that the dump pages are correctly ordered.

17.1.2 Setup and Installation

Install kexec-tools

1) Login as the root user.
2) Download the kexec-tools user-space package from the following URL:

http://kernel.org/pub/linux/utils/kernel/kexec/kexec-tools.tar.gz

This is a symlink to the latest version.

The latest kexec-tools git tree is available at:

- git://git.kernel.org/pub/scm/utils/kernel/kexec/kexec-tools.git
- http://www.kernel.org/pub/scm/utils/kernel/kexec/kexec-tools.git

There is also a gitweb interface available at http://www.kernel.org/git/?p=utils/kernel/kexec/kexec-tools.git

More information about kexec-tools can be found at http://horms.net/projects/kexec/

3) Unpack the tarball with the tar command, as follows:

```
tar xvpzf kexec-tools.tar.gz
```

4) Change to the kexec-tools directory, as follows:

```
cd kexec-tools-VERSION
```

5) Configure the package, as follows:

```
./configure
```

6) Compile the package, as follows:

```
make
```

7) Install the package, as follows:

```
make install
```

Build the system and dump-capture kernels

There are two possible methods of using Kdump.

1) Build a separate custom dump-capture kernel for capturing the kernel core dump.
2) Or use the system kernel binary itself as dump-capture kernel and there is no need to build a separate dump-capture kernel. This is possible only with the architectures which support a relocatable kernel. As of today, i386, x86_64, ppc64, arm and arm64 architectures support relocatable kernel.

Building a relocatable kernel is advantageous from the point of view that one does not have to build a second kernel for capturing the dump. But at the same time one might want to build a custom dump capture kernel suitable to his needs.

Following are the configuration setting required for system and dump-capture kernels for enabling kdump support.

System kernel config options

1) Enable ”kexec system call” or ”kexec file based system call” in ”Processor type and features.”:

```
CONFIG_KEXEC=y or CONFIG_KEXEC_FILE=y
```

And both of them will select KEXEC_CORE:

```
CONFIG_KEXEC_CORE=y
```

Subsequently, CRASH_CORE is selected by KEXEC_CORE:

```
CONFIG_CRASH_CORE=y
```

2) Enable ”sysfs file system support” in ”Filesystem” -> ”Pseudo filesystems.” This is usually enabled by default:

```
CONFIG_SYSFS=y
```

Note that ”sysfs file system support” might not appear in the ”Pseudo filesystems” menu if ”Configure standard kernel features (expert users)” is not enabled in ”General Setup.” In this case, check the .config file itself to ensure that sysfs is turned on, as follows:

```
grep 'CONFIG_SYSFS' .config
```

3) Enable ”Compile the kernel with debug info” in ”Kernel hacking.”:

```
CONFIG_DEBUG_INFO=Y
```

This causes the kernel to be built with debug symbols. The dump analysis tools require a vmlinux with debug symbols in order to read and analyze a dump file.

Dump-capture kernel config options (Arch Independent)

1) Enable ”kernel crash dumps” support under ”Processor type and features”:

```
CONFIG_CRASH_DUMP=y
```

2) Enable ”/proc/vmcore support” under ”Filesystems” -> ”Pseudo filesystems”:

```
CONFIG_PROC_VMCORE=y
```

(CONFIG_PROC_VMCORE is set by default when CONFIG_CRASH_DUMP is selected.)

Dump-capture kernel config options (Arch Dependent, i386 and x86_64)

1) On i386, enable high memory support under ”Processor type and features”:

```
CONFIG_HIGHMEM64G=y
```

or:

```
CONFIG_HIGHMEM4G
```

2) With CONFIG_SMP=y, usually nr_cpus=1 need specified on the kernel command line when loading the dump-capture kernel because one CPU is enough for kdump kernel to dump vmcore on most of systems.

However, you can also specify nr_cpus=X to enable multiple processors in kdump kernel. In this case, ”disable_cpu_apicid=” is needed to tell kdump kernel which cpu is 1st kernel's BSP. Please refer to admin-guide/kernel-parameters.txt for more details.

With CONFIG_SMP=n, the above things are not related.

3) A relocatable kernel is suggested to be built by default. If not yet, enable ”Build a relocatable kernel” support under ”Processor type and features”:

```
CONFIG_RELOCATABLE=y
```

4) Use a suitable value for ”Physical address where the kernel is loaded” (under ”Processor type and features”). This only appears when ”kernel crash dumps” is enabled. A suitable value depends upon whether kernel is relocatable or not.

If you are using a relocatable kernel use CONFIG_PHYSICAL_START=0x100000 This will compile the kernel for physical address 1MB, but given the fact kernel is relocatable, it can be run from any physical address hence kexec boot loader will load it in memory region reserved for dump-capture kernel.

Otherwise it should be the start of memory region reserved for second kernel using boot parameter ”crashkernel=Y@X”. Here X is start of memory region reserved for dump-capture kernel. Generally X is 16MB (0x1000000). So you can set CONFIG_PHYSICAL_START=0x1000000

5) Make and install the kernel and its modules. DO NOT add this kernel to the boot loader configuration files.

Dump-capture kernel config options (Arch Dependent, ppc64)

1) Enable "Build a kdump crash kernel" support under "Kernel" options:

```
CONFIG_CRASH_DUMP=y
```

2) Enable "Build a relocatable kernel" support:

```
CONFIG_RELOCATABLE=y
```

Make and install the kernel and its modules.

Dump-capture kernel config options (Arch Dependent, arm)

- To use a relocatable kernel, Enable "AUTO_ZRELADDR" support under "Boot" options:

```
AUTO_ZRELADDR=y
```

Dump-capture kernel config options (Arch Dependent, arm64)

- Please note that kvm of the dump-capture kernel will not be enabled on non-VHE systems even if it is configured. This is because the CPU will not be reset to EL2 on panic.

17.1.3 crashkernel syntax

1) crashkernel=size@offset

Here 'size' specifies how much memory to reserve for the dump-capture kernel and 'offset' specifies the beginning of this reserved memory. For example, "crashkernel=64M@16M" tells the system kernel to reserve 64 MB of memory starting at physical address 0x01000000 (16MB) for the dump-capture kernel.

The crashkernel region can be automatically placed by the system kernel at run time. This is done by specifying the base address as 0, or omitting it all together:

```
crashkernel=256M@0
```

or:

```
crashkernel=256M
```

If the start address is specified, note that the start address of the kernel will be aligned to a value (which is Arch dependent), so if the start address is not then any space below the alignment point will be wasted.

2) range1:size1[,range2:size2,...][@offset]

While the "crashkernel=size[@offset]" syntax is sufficient for most configurations, sometimes it's handy to have the reserved memory dependent on the value of System RAM -- that's mostly for distributors that pre-setup the kernel command line to avoid a unbootable system after some memory has been removed from the machine.

The syntax is:

```
crashkernel=<range1>:<size1>[,<range2>:<size2>,...][@offset]
range=start-[end]
```

For example:

```
crashkernel=512M-2G:64M,2G-:128M
```

This would mean:

1) if the RAM is smaller than 512M, then don't reserve anything (this is the "rescue" case)
2) if the RAM size is between 512M and 2G (exclusive), then reserve 64M
3) if the RAM size is larger than 2G, then reserve 128M

3) crashkernel=size,high and crashkernel=size,low

If memory above 4G is preferred, crashkernel=size,high can be used to fulfill that. With it, physical memory is allowed to be allocated from top, so could be above 4G if system has more than 4G RAM installed. Otherwise, memory region will be allocated below 4G if available.

When crashkernel=X,high is passed, kernel could allocate physical memory region above 4G, low memory under 4G is needed in this case. There are three ways to get low memory:

1) Kernel will allocate at least 256M memory below 4G automatically if crashkernel=Y,low is not specified.
2) Let user specify low memory size instead.
3) Specified value 0 will disable low memory allocation:

```
crashkernel=0,low
```

Boot into System Kernel

1) Update the boot loader (such as grub, yaboot, or lilo) configuration files as necessary.
2) Boot the system kernel with the boot parameter "crashkernel=Y@X".

On x86 and x86_64, use "crashkernel=Y[@X]". Most of the time, the start address 'X' is not necessary, kernel will search a suitable area. Unless an explicit start address is expected.

On ppc64, use "crashkernel=128M@32M".

On s390x, typically use "crashkernel=xxM". The value of xx is dependent on the memory consumption of the kdump system. In general this is not dependent on the memory size of the production system.

On arm, the use of "crashkernel=Y@X" is no longer necessary; the kernel will automatically locate the crash kernel image within the first 512MB of RAM if X is not given.

On arm64, use "crashkernel=Y[@X]". Note that the start address of the kernel, X if explicitly specified, must be aligned to 2MiB (0x200000).

17.1.4 Load the Dump-capture Kernel

After booting to the system kernel, dump-capture kernel needs to be loaded.

Based on the architecture and type of image (relocatable or not), one can choose to load the uncompressed vmlinux or compressed bzImage/vmlinuz of dump-capture kernel. Following is the summary.

For i386 and x86_64:

- Use bzImage/vmlinuz if kernel is relocatable.
- Use vmlinux if kernel is not relocatable.

For ppc64:

- Use vmlinux

For s390x:

- Use image or bzImage

For arm:

- Use zImage

For arm64:

- Use vmlinux or Image

If you are using an uncompressed vmlinux image then use following command to load dump-capture kernel:

```
kexec -p <dump-capture-kernel-vmlinux-image> \
--initrd=<initrd-for-dump-capture-kernel> --args-linux \
--append="root=<root-dev> <arch-specific-options>"
```

If you are using a compressed bzImage/vmlinuz, then use following command to load dump-capture kernel:

```
kexec -p <dump-capture-kernel-bzImage> \
--initrd=<initrd-for-dump-capture-kernel> \
--append="root=<root-dev> <arch-specific-options>"
```

If you are using a compressed zImage, then use following command to load dump-capture kernel:

```
kexec --type zImage -p <dump-capture-kernel-bzImage> \
--initrd=<initrd-for-dump-capture-kernel> \
--dtb=<dtb-for-dump-capture-kernel> \
--append="root=<root-dev> <arch-specific-options>"
```

If you are using an uncompressed Image, then use following command to load dump-capture kernel:

```
kexec -p <dump-capture-kernel-Image> \
--initrd=<initrd-for-dump-capture-kernel> \
--append="root=<root-dev> <arch-specific-options>"
```

Following are the arch specific command line options to be used while loading dump-capture kernel.

For i386 and x86_64:

> ”1 irqpoll nr_cpus=1 reset_devices”

For ppc64:

> ”1 maxcpus=1 noirqdistrib reset_devices”

For s390x:

> ”1 nr_cpus=1 cgroup_disable=memory”

For arm:

> ”1 maxcpus=1 reset_devices”

For arm64:

> ”1 nr_cpus=1 reset_devices”

Notes on loading the dump-capture kernel:

- By default, the ELF headers are stored in ELF64 format to support systems with more than 4GB memory. On i386, kexec automatically checks if the physical RAM size exceeds the 4 GB limit and if not, uses ELF32. So, on non-PAE systems, ELF32 is always used.

  The --elf32-core-headers option can be used to force the generation of ELF32 headers. This is necessary because GDB currently cannot open vmcore files with ELF64 headers on 32-bit systems.
- The ”irqpoll” boot parameter reduces driver initialization failures due to shared interrupts in the dump-capture kernel.
- You must specify <root-dev> in the format corresponding to the root device name in the output of mount command.
- Boot parameter ”1” boots the dump-capture kernel into single-user mode without networking. If you want networking, use ”3”.
- We generally don't have to bring up a SMP kernel just to capture the dump. Hence generally it is useful either to build a UP dump-capture kernel or specify maxcpus=1 option while loading dump-capture kernel. Note, though maxcpus always works, you had better replace it with nr_cpus to save memory if supported by the current ARCH, such as x86.
- You should enable multi-cpu support in dump-capture kernel if you intend to use multi-thread programs with it, such as parallel dump feature of makedumpfile. Otherwise, the multi-thread program may have a great performance degradation. To enable multi-cpu support, you should bring up an SMP dump-capture kernel and specify maxcpus/nr_cpus, disable_cpu_apicid=[X] options while loading it.
- For s390x there are two kdump modes: If a ELF header is specified with the elfcorehdr= kernel parameter, it is used by the kdump kernel as it is done on all other architectures. If no elfcorehdr= kernel parameter is specified, the s390x kdump kernel dynamically creates the header. The second mode has the advantage that for CPU and memory hotplug, kdump has not to be reloaded with kexec_load().
- For s390x systems with many attached devices the ”cio_ignore” kernel parameter should be used for the kdump kernel in order to prevent allocation of kernel memory for devices

that are not relevant for kdump. The same applies to systems that use SCSI/FCP devices. In that case the "allow_lun_scan" zfcp module parameter should be set to zero before setting FCP devices online.

17.1.5 Kernel Panic

After successfully loading the dump-capture kernel as previously described, the system will reboot into the dump-capture kernel if a system crash is triggered. Trigger points are located in panic(), die(), die_nmi() and in the sysrq handler (ALT-SysRq-c).

The following conditions will execute a crash trigger point:

If a hard lockup is detected and "NMI watchdog" is configured, the system will boot into the dump-capture kernel (die_nmi()).

If die() is called, and it happens to be a thread with pid 0 or 1, or die() is called inside interrupt context or die() is called and panic_on_oops is set, the system will boot into the dump-capture kernel.

On powerpc systems when a soft-reset is generated, die() is called by all cpus and the system will boot into the dump-capture kernel.

For testing purposes, you can trigger a crash by using "ALT-SysRq-c", "echo c > /proc/sysrq-trigger" or write a module to force the panic.

17.1.6 Write Out the Dump File

After the dump-capture kernel is booted, write out the dump file with the following command:

```
cp /proc/vmcore <dump-file>
```

or use scp to write out the dump file between hosts on a network, e.g:

```
scp /proc/vmcore remote_username@remote_ip:<dump-file>
```

You can also use makedumpfile utility to write out the dump file with specified options to filter out unwanted contents, e.g:

```
makedumpfile -l --message-level 1 -d 31 /proc/vmcore <dump-file>
```

17.1.7 Analysis

Before analyzing the dump image, you should reboot into a stable kernel.

You can do limited analysis using GDB on the dump file copied out of /proc/vmcore. Use the debug vmlinux built with -g and run the following command:

```
gdb vmlinux <dump-file>
```

Stack trace for the task on processor 0, register display, and memory display work fine.

Note: GDB cannot analyze core files generated in ELF64 format for x86. On systems with a maximum of 4GB of memory, you can generate ELF32-format headers using the --elf32-core-headers kernel option on the dump kernel.

You can also use the Crash utility to analyze dump files in Kdump format. Crash is available at the following URL:

https://github.com/crash-utility/crash

Crash document can be found at:
https://crash-utility.github.io/

17.1.8 Trigger Kdump on WARN()

The kernel parameter, panic_on_warn, calls panic() in all WARN() paths. This will cause a kdump to occur at the panic() call. In cases where a user wants to specify this during runtime, /proc/sys/kernel/panic_on_warn can be set to 1 to achieve the same behaviour.

17.1.9 Trigger Kdump on add_taint()

The kernel parameter panic_on_taint facilitates a conditional call to panic() from within add_taint() whenever the value set in this bitmask matches with the bit flag being set by add_taint(). This will cause a kdump to occur at the add_taint()->panic() call.

17.1.10 Contact

- kexec@lists.infradead.org

17.1.11 GDB macros

```
#
# This file contains a few gdb macros (user defined commands) to extract
# useful information from kernel crashdump (kdump) like stack traces of
# all the processes or a particular process and trapinfo.
#
# These macros can be used by copying this file in .gdbinit (put in home
# directory or current directory) or by invoking gdb command with
# --command=<command-file-name> option
#
# Credits:
# Alexander Nyberg <alexn@telia.com>
# V Srivatsa <vatsa@in.ibm.com>
# Maneesh Soni <maneesh@in.ibm.com>
#

define bttnobp
        set $tasks_off=((size_t)&((struct task_struct *)0)->tasks)
        set $pid_off=((size_t)&((struct task_struct *)0)->thread_group.next)
        set $init_t=&init_task
        set $next_t=(((char *)($init_t->tasks).next) - $tasks_off)
```

```
        set var $stacksize = sizeof(union thread_union)
        while ($next_t != $init_t)
                set $next_t=(struct task_struct *)$next_t
                printf "\npid %d; comm %s:\n", $next_t.pid, $next_t.comm
                printf "===================\n"
                set var $stackp = $next_t.thread.sp
                set var $stack_top = ($stackp & ~($stacksize - 1)) + $stacksize

                while ($stackp < $stack_top)
                        if (*($stackp) > _stext && *($stackp) < _sinittext)
                                info symbol *($stackp)
                        end
                        set $stackp += 4
                end
                set $next_th=(((char *)$next_t->thread_group.next) - $pid_off)
                while ($next_th != $next_t)
                        set $next_th=(struct task_struct *)$next_th
                        printf "\npid %d; comm %s:\n", $next_t.pid, $next_t.
↪comm
                        printf "===================\n"
                        set var $stackp = $next_t.thread.sp
                        set var $stack_top = ($stackp & ~($stacksize - 1)) +␣
↪stacksize

                        while ($stackp < $stack_top)
                                if (*($stackp) > _stext && *($stackp) < _
↪sinittext)
                                        info symbol *($stackp)
                                end
                                set $stackp += 4
                        end
                        set $next_th=(((char *)$next_th->thread_group.next) -
↪$pid_off)
                end
                set $next_t=(char *)($next_t->tasks.next) - $tasks_off
        end
end
document bttnobp
        dump all thread stack traces on a kernel compiled with !CONFIG_FRAME_
↪POINTER
end

define btthreadstack
        set var $pid_task = $arg0

        printf "\npid %d; comm %s:\n", $pid_task.pid, $pid_task.comm
        printf "task struct: "
        print $pid_task
        printf "===================\n"
        set var $stackp = $pid_task.thread.sp
        set var $stacksize = sizeof(union thread_union)
```

```
        set var $stack_top = ($stackp & ~($stacksize - 1)) + $stacksize
        set var $stack_bot = ($stackp & ~($stacksize - 1))

        set $stackp = *((unsigned long *) $stackp)
        while (($stackp < $stack_top) && ($stackp > $stack_bot))
                set var $addr = *(((unsigned long *) $stackp) + 1)
                info symbol $addr
                set $stackp = *((unsigned long *) $stackp)
        end
end
document btthreadstack
         dump a thread stack using the given task structure pointer
end


define btt
        set $tasks_off=((size_t)&((struct task_struct *)0)->tasks)
        set $pid_off=((size_t)&((struct task_struct *)0)->thread_group.next)
        set $init_t=&init_task
        set $next_t=(((char *)($init_t->tasks).next) - $tasks_off)
        while ($next_t != $init_t)
                set $next_t=(struct task_struct *)$next_t
                btthreadstack $next_t

                set $next_th=(((char *)$next_t->thread_group.next) - $pid_off)
                while ($next_th != $next_t)
                        set $next_th=(struct task_struct *)$next_th
                        btthreadstack $next_th
                        set $next_th=(((char *)$next_th->thread_group.next) -
↪$pid_off)
                end
                set $next_t=(char *)($next_t->tasks.next) - $tasks_off
        end
end
document btt
        dump all thread stack traces on a kernel compiled with CONFIG_FRAME_
↪POINTER
end

define btpid
        set var $pid = $arg0
        set $tasks_off=((size_t)&((struct task_struct *)0)->tasks)
        set $pid_off=((size_t)&((struct task_struct *)0)->thread_group.next)
        set $init_t=&init_task
        set $next_t=(((char *)($init_t->tasks).next) - $tasks_off)
        set var $pid_task = 0

        while ($next_t != $init_t)
                set $next_t=(struct task_struct *)$next_t

                if ($next_t.pid == $pid)
```

```
                        set $pid_task = $next_t
                end

                set $next_th=(((char *)$next_t->thread_group.next) - $pid_off)
                while ($next_th != $next_t)
                        set $next_th=(struct task_struct *)$next_th
                        if ($next_th.pid == $pid)
                                set $pid_task = $next_th
                        end
                        set $next_th=(((char *)$next_th->thread_group.next) -
↪$pid_off)
                end
                set $next_t=(char *)($next_t->tasks.next) - $tasks_off
        end

        btthreadstack $pid_task
end
document btpid
        backtrace of pid
end


define trapinfo
        set var $pid = $arg0
        set $tasks_off=((size_t)&((struct task_struct *)0)->tasks)
        set $pid_off=((size_t)&((struct task_struct *)0)->thread_group.next)
        set $init_t=&init_task
        set $next_t=(((char *)($init_t->tasks).next) - $tasks_off)
        set var $pid_task = 0

        while ($next_t != $init_t)
                set $next_t=(struct task_struct *)$next_t

                if ($next_t.pid == $pid)
                        set $pid_task = $next_t
                end

                set $next_th=(((char *)$next_t->thread_group.next) - $pid_off)
                while ($next_th != $next_t)
                        set $next_th=(struct task_struct *)$next_th
                        if ($next_th.pid == $pid)
                                set $pid_task = $next_th
                        end
                        set $next_th=(((char *)$next_th->thread_group.next) -
↪$pid_off)
                end
                set $next_t=(char *)($next_t->tasks.next) - $tasks_off
        end

        printf "Trapno %ld, cr2 0x%lx, error_code %ld\n", $pid_task.thread.
↪trap_no, \
```

```
                                $pid_task.thread.cr2, $pid_task.thread.error_
↪code

end
document trapinfo
        Run info threads and lookup pid of thread #1
        'trapinfo <pid>' will tell you by which trap & possibly
        address the kernel panicked.
end

define dump_record
        set var $desc = $arg0
        set var $info = $arg1
        if ($argc > 2)
                set var $prev_flags = $arg2
        else
                set var $prev_flags = 0
        end

        set var $prefix = 1
        set var $newline = 1

        set var $begin = $desc->text_blk_lpos.begin % (1U << prb->text_data_
↪ring.size_bits)
        set var $next = $desc->text_blk_lpos.next % (1U << prb->text_data_ring.
↪size_bits)

        # handle data-less record
        if ($begin & 1)
                set var $text_len = 0
                set var $log = ""
        else
                # handle wrapping data block
                if ($begin > $next)
                        set var $begin = 0
                end

                # skip over descriptor id
                set var $begin = $begin + sizeof(long)

                # handle truncated message
                if ($next - $begin < $info->text_len)
                        set var $text_len = $next - $begin
                else
                        set var $text_len = $info->text_len
                end

                set var $log = &prb->text_data_ring.data[$begin]
        end

        # prev & LOG_CONT && !(info->flags & LOG_PREIX)
```

```
        if (($prev_flags & 8) && !($info->flags & 4))
                set var $prefix = 0
        end

        # info->flags & LOG_CONT
        if ($info->flags & 8)
                # (prev & LOG_CONT && !(prev & LOG_NEWLINE))
                if (($prev_flags & 8) && !($prev_flags & 2))
                        set var $prefix = 0
                end
                # (!(info->flags & LOG_NEWLINE))
                if (!($info->flags & 2))
                        set var $newline = 0
                end
        end

        if ($prefix)
                printf "[%5lu.%06lu] ", $info->ts_nsec / 1000000000, $info->ts_
↪nsec % 1000000000
        end
        if ($text_len)
                eval "printf \"%%%d.%ds\", $log", $text_len, $text_len
        end
        if ($newline)
                printf "\n"
        end

        # handle dictionary data

        set var $dict = &$info->dev_info.subsystem[0]
        set var $dict_len = sizeof($info->dev_info.subsystem)
        if ($dict[0] != '\0')
                printf " SUBSYSTEM="
                set var $idx = 0
                while ($idx < $dict_len)
                        set var $c = $dict[$idx]
                        if ($c == '\0')
                                loop_break
                        else
                                if ($c < ' ' || $c >= 127 || $c == '\\')
                                        printf "\\x%02x", $c
                                else
                                        printf "%c", $c
                                end
                        end
                        set var $idx = $idx + 1
                end
                printf "\n"
        end

        set var $dict = &$info->dev_info.device[0]
```

```
        set var $dict_len = sizeof($info->dev_info.device)
        if ($dict[0] != '\0')
                printf " DEVICE="
                set var $idx = 0
                while ($idx < $dict_len)
                        set var $c = $dict[$idx]
                        if ($c == '\0')
                                loop_break
                        else
                                if ($c < ' ' || $c >= 127 || $c == '\\')
                                        printf "\\x%02x", $c
                                else
                                        printf "%c", $c
                                end
                        end
                        set var $idx = $idx + 1
                end
                printf "\n"
        end
end
document dump_record
        Dump a single record. The first parameter is the descriptor,
        the second parameter is the info, the third parameter is
        optional and specifies the previous record's flags, used for
        properly formatting continued lines.
end

define dmesg
        # definitions from kernel/printk/printk_ringbuffer.h
        set var $desc_committed = 1
        set var $desc_finalized = 2
        set var $desc_sv_bits = sizeof(long) * 8
        set var $desc_flags_shift = $desc_sv_bits - 2
        set var $desc_flags_mask = 3 << $desc_flags_shift
        set var $id_mask = ~$desc_flags_mask

        set var $desc_count = 1U << prb->desc_ring.count_bits
        set var $prev_flags = 0

        set var $id = prb->desc_ring.tail_id.counter
        set var $end_id = prb->desc_ring.head_id.counter

        while (1)
                set var $desc = &prb->desc_ring.descs[$id % $desc_count]
                set var $info = &prb->desc_ring.infos[$id % $desc_count]

                # skip non-committed record
                set var $state = 3 & ($desc->state_var.counter >> $desc_flags_
↪shift)
                if ($state == $desc_committed || $state == $desc_finalized)
                        dump_record $desc $info $prev_flags
```

```
                        set var $prev_flags = $info->flags
                end

                if ($id == $end_id)
                        loop_break
                end
                set var $id = ($id + 1) & $id_mask
        end
end
document dmesg
        print the kernel ring buffer
end
```

17.2 VMCOREINFO

17.2.1 What is it?

VMCOREINFO is a special ELF note section. It contains various information from the kernel like structure size, page size, symbol values, field offsets, etc. These data are packed into an ELF note section and used by user-space tools like crash and makedumpfile to analyze a kernel's memory layout.

17.2.2 Common variables

init_uts_ns.name.release

The version of the Linux kernel. Used to find the corresponding source code from which the kernel has been built. For example, crash uses it to find the corresponding vmlinux in order to process vmcore.

PAGE_SIZE

The size of a page. It is the smallest unit of data used by the memory management facilities. It is usually 4096 bytes of size and a page is aligned on 4096 bytes. Used for computing page addresses.

init_uts_ns

The UTS namespace which is used to isolate two specific elements of the system that relate to the uname(2) system call. It is named after the data structure used to store information returned by the uname(2) system call.

User-space tools can get the kernel name, host name, kernel release number, kernel version, architecture name and OS type from it.

(uts_namespace, name)

Offset of the name's member. Crash Utility and Makedumpfile get the start address of the init_uts_ns.name from this.

node_online_map

An array node_states[N_ONLINE] which represents the set of online nodes in a system, one bit position per node number. Used to keep track of which nodes are in the system and online.

swapper_pg_dir

The global page directory pointer of the kernel. Used to translate virtual to physical addresses.

_stext

Defines the beginning of the text section. In general, _stext indicates the kernel start address. Used to convert a virtual address from the direct kernel map to a physical address.

vmap_area_list

Stores the virtual area list. makedumpfile gets the vmalloc start value from this variable and its value is necessary for vmalloc translation.

mem_map

Physical addresses are translated to struct pages by treating them as an index into the mem_map array. Right-shifting a physical address PAGE_SHIFT bits converts it into a page frame number which is an index into that mem_map array.

Used to map an address to the corresponding struct page.

contig_page_data

Makedumpfile gets the pglist_data structure from this symbol, which is used to describe the memory layout.

User-space tools use this to exclude free pages when dumping memory.

mem_section|(mem_section, NR_SECTION_ROOTS)|(mem_section, section_mem_map)

The address of the mem_section array, its length, structure size, and the section_mem_map offset.

It exists in the sparse memory mapping model, and it is also somewhat similar to the mem_map variable, both of them are used to translate an address.

MAX_PHYSMEM_BITS

Defines the maximum supported physical address space memory.

page

The size of a page structure. struct page is an important data structure and it is widely used to compute contiguous memory.

pglist_data

The size of a pglist_data structure. This value is used to check if the pglist_data structure is valid. It is also used for checking the memory type.

zone

The size of a zone structure. This value is used to check if the zone structure has been found. It is also used for excluding free pages.

free_area

The size of a free_area structure. It indicates whether the free_area structure is valid or not. Useful when excluding free pages.

list_head

The size of a list_head structure. Used when iterating lists in a post-mortem analysis session.

nodemask_t

The size of a nodemask_t type. Used to compute the number of online nodes.

(page, flags|_refcount|mapping|lru|_mapcount|private|compound_order|compound_head)

User-space tools compute their values based on the offset of these variables. The variables are used when excluding unnecessary pages.

(pglist_data, node_zones|nr_zones|node_mem_map|node_start_pfn|node_spanned_pages|no

On NUMA machines, each NUMA node has a pg_data_t to describe its memory layout. On UMA machines there is a single pglist_data which describes the whole memory.

These values are used to check the memory type and to compute the virtual address for memory map.

(zone, free_area|vm_stat|spanned_pages)

Each node is divided into a number of blocks called zones which represent ranges within memory. A zone is described by a structure zone.

User-space tools compute required values based on the offset of these variables.

(free_area, free_list)

Offset of the free_list's member. This value is used to compute the number of free pages.

Each zone has a free_area structure array called free_area[NR_PAGE_ORDERS]. The free_list represents a linked list of free page blocks.

(list_head, next|prev)

Offsets of the list_head's members. list_head is used to define a circular linked list. User-space tools need these in order to traverse lists.

(vmap_area, va_start|list)

Offsets of the vmap_area's members. They carry vmalloc-specific information. Makedumpfile gets the start address of the vmalloc region from this.

(zone.free_area, NR_PAGE_ORDERS)

Free areas descriptor. User-space tools use this value to iterate the free_area ranges. NR_PAGE_ORDERS is used by the zone buddy allocator.

prb

A pointer to the printk ringbuffer (struct printk_ringbuffer). This may be pointing to the static boot ringbuffer or the dynamically allocated ringbuffer, depending on when the core dump occurred. Used by user-space tools to read the active kernel log buffer.

printk_rb_static

A pointer to the static boot printk ringbuffer. If @prb has a different value, this is useful for viewing the initial boot messages, which may have been overwritten in the dynamically allocated ringbuffer.

clear_seq

The sequence number of the printk() record after the last clear command. It indicates the first record after the last SYSLOG_ACTION_CLEAR, like issued by 'dmesg -c'. Used by user-space tools to dump a subset of the dmesg log.

printk_ringbuffer

The size of a printk_ringbuffer structure. This structure contains all information required for accessing the various components of the kernel log buffer.

(printk_ringbuffer, desc_ring|text_data_ring|dict_data_ring|fail)

Offsets for the various components of the printk ringbuffer. Used by user-space tools to view the kernel log buffer without requiring the declaration of the structure.

prb_desc_ring

The size of the prb_desc_ring structure. This structure contains information about the set of record descriptors.

(prb_desc_ring, count_bits|descs|head_id|tail_id)

Offsets for the fields describing the set of record descriptors. Used by user-space tools to be able to traverse the descriptors without requiring the declaration of the structure.

prb_desc

The size of the prb_desc structure. This structure contains information about a single record descriptor.

(prb_desc, info|state_var|text_blk_lpos|dict_blk_lpos)

Offsets for the fields describing a record descriptors. Used by user-space tools to be able to read descriptors without requiring the declaration of the structure.

prb_data_blk_lpos

The size of the prb_data_blk_lpos structure. This structure contains information about where the text or dictionary data (data block) is located within the respective data ring.

(prb_data_blk_lpos, begin|next)

Offsets for the fields describing the location of a data block. Used by user-space tools to be able to locate data blocks without requiring the declaration of the structure.

printk_info

The size of the printk_info structure. This structure contains all the meta-data for a record.

(printk_info, seq|ts_nsec|text_len|dict_len|caller_id)

Offsets for the fields providing the meta-data for a record. Used by user-space tools to be able to read the information without requiring the declaration of the structure.

prb_data_ring

The size of the prb_data_ring structure. This structure contains information about a set of data blocks.

(prb_data_ring, size_bits|data|head_lpos|tail_lpos)

Offsets for the fields describing a set of data blocks. Used by user-space tools to be able to access the data blocks without requiring the declaration of the structure.

atomic_long_t

The size of the atomic_long_t structure. Used by user-space tools to be able to copy the full structure, regardless of its architecture-specific implementation.

(atomic_long_t, counter)

Offset for the long value of an atomic_long_t variable. Used by user-space tools to access the long value without requiring the architecture-specific declaration.

(free_area.free_list, MIGRATE_TYPES)

The number of migrate types for pages. The free_list is described by the array. Used by tools to compute the number of free pages.

NR_FREE_PAGES

On linux-2.6.21 or later, the number of free pages is in vm_stat[NR_FREE_PAGES]. Used to get the number of free pages.

PG_lru|PG_private|PG_swapcache|PG_swapbacked|PG_slab|PG_hwpoision|PG_head_mask|PG

Page attributes. These flags are used to filter various unnecessary for dumping pages.

PAGE_BUDDY_MAPCOUNT_VALUE(~PG_buddy)|PAGE_OFFLINE_MAPCOUNT_VALUE(~PG_offlin

More page attributes. These flags are used to filter various unnecessary for dumping pages.

17.2.3 x86_64

phys_base

Used to convert the virtual address of an exported kernel symbol to its corresponding physical address.

init_top_pgt

Used to walk through the whole page table and convert virtual addresses to physical addresses. The init_top_pgt is somewhat similar to swapper_pg_dir, but it is only used in x86_64.

pgtable_l5_enabled

User-space tools need to know whether the crash kernel was in 5-level paging mode.

node_data

This is a struct pglist_data array and stores all NUMA nodes information. Makedumpfile gets the pglist_data structure from it.

(node_data, MAX_NUMNODES)

The maximum number of nodes in system.

KERNELOFFSET

The kernel randomization offset. Used to compute the page offset. If KASLR is disabled, this value is zero.

KERNEL_IMAGE_SIZE

Currently unused by Makedumpfile. Used to compute the module virtual address by Crash.

sme_mask

AMD-specific with SME support: it indicates the secure memory encryption mask. Makedumpfile tools need to know whether the crash kernel was encrypted. If SME is enabled in the first kernel, the crash kernel's page table entries (pgd/pud/pmd/pte) contain the memory encryption mask. This is used to remove the SME mask and obtain the true physical address.

Currently, sme_mask stores the value of the C-bit position. If needed, additional SME-relevant info can be placed in that variable.

For example:

```
[ misc                ][ enc bit  ][ other misc SME info       ]
0000_0000_0000_0000_1000_0000_0000_0000_0000_0000_..._0000
63   59   55   51   47   43   39   35   31   27   ... 3
```

17.2.4 x86_32

X86_PAE

Denotes whether physical address extensions are enabled. It has the cost of a higher page table lookup overhead, and also consumes more page table space per process. Used to check whether PAE was enabled in the crash kernel when converting virtual addresses to physical addresses.

17.2.5 ARM64

VA_BITS

The maximum number of bits for virtual addresses. Used to compute the virtual memory ranges.

kimage_voffset

The offset between the kernel virtual and physical mappings. Used to translate virtual to physical addresses.

PHYS_OFFSET

Indicates the physical address of the start of memory. Similar to kimage_voffset, which is used to translate virtual to physical addresses.

KERNELOFFSET

The kernel randomization offset. Used to compute the page offset. If KASLR is disabled, this value is zero.

KERNELPACMASK

The mask to extract the Pointer Authentication Code from a kernel virtual address.

TCR_EL1.T1SZ

Indicates the size offset of the memory region addressed by TTBR1_EL1. The region size is 2^(64-T1SZ) bytes.

TTBR1_EL1 is the table base address register specified by ARMv8-A architecture which is used to lookup the page-tables for the Virtual addresses in the higher VA range (refer to ARMv8 ARM document for more details).

MODULES_VADDR|MODULES_END|VMALLOC_START|VMALLOC_END|VMEMMAP_START|VMEM

Used to get the correct ranges:

MODULES_VADDR ~ MODULES_END-1 : Kernel module space. VMALLOC_START ~ VMALLOC_END-1 : vmalloc() / ioremap() space. VMEMMAP_START ~ VMEMMAP_END-1 : vmemmap region, used for struct page array.

17.2.6 arm

ARM_LPAE

It indicates whether the crash kernel supports large physical address extensions. Used to translate virtual to physical addresses.

17.2.7 s390

lowcore_ptr

An array with a pointer to the lowcore of every CPU. Used to print the psw and all registers information.

high_memory

Used to get the vmalloc_start address from the high_memory symbol.

(lowcore_ptr, NR_CPUS)

The maximum number of CPUs.

17.2.8 powerpc

node_data|(node_data, MAX_NUMNODES)

See above.

contig_page_data

See above.

vmemmap_list

The vmemmap_list maintains the entire vmemmap physical mapping. Used to get vmemmap list count and populated vmemmap regions info. If the vmemmap address translation information is stored in the crash kernel, it is used to translate vmemmap kernel virtual addresses.

mmu_vmemmap_psize

The size of a page. Used to translate virtual to physical addresses.

mmu_psize_defs

Page size definitions, i.e. 4k, 64k, or 16M.

Used to make vtop translations.

vmemmap_backing|(vmemmap_backing, list)|(vmemmap_backing, phys)|(vmemmap_backing, virt_addr)

The vmemmap virtual address space management does not have a traditional page table to track which virtual struct pages are backed by a physical mapping. The virtual to physical mappings are tracked in a simple linked list format.

User-space tools need to know the offset of list, phys and virt_addr when computing the count of vmemmap regions.

mmu_psize_def|(mmu_psize_def, shift)

The size of a struct mmu_psize_def and the offset of mmu_psize_def's member.

Used in vtop translations.

17.2.9 sh

node_data|(node_data, MAX_NUMNODES)

See above.

X2TLB

Indicates whether the crashed kernel enabled SH extended mode.

17.2.10 RISCV64

VA_BITS

The maximum number of bits for virtual addresses. Used to compute the virtual memory ranges.

PAGE_OFFSET

Indicates the virtual kernel start address of the direct-mapped RAM region.

phys_ram_base

Indicates the start physical RAM address.

MODULES_VADDR|MODULES_END|VMALLOC_START|VMALLOC_END|VMEMMAP_START|VMEM

Used to get the correct ranges:

- MODULES_VADDR ~ MODULES_END : Kernel module space.
- VMALLOC_START ~ VMALLOC_END : vmalloc() / ioremap() space.
- VMEMMAP_START ~ VMEMMAP_END : vmemmap space, used for struct page array.
- KERNEL_LINK_ADDR : start address of Kernel link and BPF

va_kernel_pa_offset

Indicates the offset between the kernel virtual and physical mappings. Used to translate virtual to physical addresses.

CHAPTER

EIGHTEEN

PERFORMANCE MONITOR SUPPORT

18.1 HiSilicon SoC uncore Performance Monitoring Unit (PMU)

The HiSilicon SoC chip includes various independent system device PMUs such as L3 cache (L3C), Hydra Home Agent (HHA) and DDRC. These PMUs are independent and have hardware logic to gather statistics and performance information.

The HiSilicon SoC encapsulates multiple CPU and IO dies. Each CPU cluster (CCL) is made up of 4 cpu cores sharing one L3 cache; each CPU die is called Super CPU cluster (SCCL) and is made up of 6 CCLs. Each SCCL has two HHAs (0 - 1) and four DDRCs (0 - 3), respectively.

18.1.1 HiSilicon SoC uncore PMU driver

Each device PMU has separate registers for event counting, control and interrupt, and the PMU driver shall register perf PMU drivers like L3C, HHA and DDRC etc. The available events and configuration options shall be described in the sysfs, see:

/sys/devices/hisi_sccl{X}_<l3c{Y}/hha{Y}/ddrc{Y}>/, or /sys/bus/event_source/devices/hisi_sccl{X}_
The "perf list" command shall list the available events from sysfs.

Each L3C, HHA and DDRC is registered as a separate PMU with perf. The PMU name will appear in event listing as hisi_sccl<sccl-id>_module<index-id>. where "sccl-id" is the identifier of the SCCL and "index-id" is the index of module.

e.g. hisi_sccl3_l3c0/rd_hit_cpipe is READ_HIT_CPIPE event of L3C index #0 in SCCL ID #3.

e.g. hisi_sccl1_hha0/rx_operations is RX_OPERATIONS event of HHA index #0 in SCCL ID #1.

The driver also provides a "cpumask" sysfs attribute, which shows the CPU core ID used to count the uncore PMU event.

Example usage of perf:

```
$# perf list
hisi_sccl3_l3c0/rd_hit_cpipe/ [kernel PMU event]
------------------------------------------
hisi_sccl3_l3c0/wr_hit_cpipe/ [kernel PMU event]
------------------------------------------
hisi_sccl1_l3c0/rd_hit_cpipe/ [kernel PMU event]
------------------------------------------
hisi_sccl1_l3c0/wr_hit_cpipe/ [kernel PMU event]
------------------------------------------
```

```
$# perf stat -a -e hisi_sccl3_l3c0/rd_hit_cpipe/ sleep 5
$# perf stat -a -e hisi_sccl3_l3c0/config=0x02/ sleep 5
```

For HiSilicon uncore PMU v2 whose identifier is 0x30, the topology is the same as PMU v1, but some new functions are added to the hardware.

1. L3C PMU supports filtering by core/thread within the cluster which can be specified as a bitmap:

```
$# perf stat -a -e hisi_sccl3_l3c0/config=0x02,tt_core=0x3/ sleep 5
```

This will only count the operations from core/thread 0 and 1 in this cluster.

2. Tracetag allow the user to chose to count only read, write or atomic operations via the tt_req parameeter in perf. The default value counts all operations. tt_req is 3bits, 3'b100 represents read operations, 3'b101 represents write operations, 3'b110 represents atomic store operations and 3'b111 represents atomic non-store operations, other values are reserved:

```
$# perf stat -a -e hisi_sccl3_l3c0/config=0x02,tt_req=0x4/ sleep 5
```

This will only count the read operations in this cluster.

3. Datasrc allows the user to check where the data comes from. It is 5 bits. Some important codes are as follows:

- 5'b00001: comes from L3C in this die;
- 5'b01000: comes from L3C in the cross-die;
- 5'b01001: comes from L3C which is in another socket;
- 5'b01110: comes from the local DDR;
- 5'b01111: comes from the cross-die DDR;
- 5'b10000: comes from cross-socket DDR;

etc, it is mainly helpful to find that the data source is nearest from the CPU cores. If datasrc_cfg is used in the multi-chips, the datasrc_skt shall be configured in perf command:

```
$# perf stat -a -e hisi_sccl3_l3c0/config=0xb9,datasrc_cfg=0xE/,
hisi_sccl3_l3c0/config=0xb9,datasrc_cfg=0xF/ sleep 5
```

4. Some HiSilicon SoCs encapsulate multiple CPU and IO dies. Each CPU die contains several Compute Clusters (CCLs). The I/O dies are called Super I/O clusters (SICL) containing multiple I/O clusters (ICLs). Each CCL/ICL in the SoC has a unique ID. Each ID is 11bits, include a 6-bit SCCL-ID and 5-bit CCL/ICL-ID. For I/O die, the ICL-ID is followed by:

- 5'b00000: I/O_MGMT_ICL;
- 5'b00001: Network_ICL;
- 5'b00011: HAC_ICL;
- 5'b10000: PCIe_ICL;

5. uring_channel: UC PMU events 0x47~0x59 supports filtering by tx request uring channel. It is 2 bits. Some important codes are as follows:

- 2'b11: count the events which sent to the uring_ext (MATA) channel;
- 2'b01: is the same as 2'b11;
- 2'b10: count the events which sent to the uring (non-MATA) channel;
- 2'b00: default value, count the events which sent to the both uring and uring_ext channel;

Users could configure IDs to count data come from specific CCL/ICL, by setting srcid_cmd & srcid_msk, and data desitined for specific CCL/ICL by setting tgtid_cmd & tgtid_msk. A set bit in srcid_msk/tgtid_msk means the PMU will not check the bit when matching against the srcid_cmd/tgtid_cmd.

If all of these options are disabled, it can works by the default value that doesn't distinguish the filter condition and ID information and will return the total counter values in the PMU counters.

The current driver does not support sampling. So "perf record" is unsupported. Also attach to a task is unsupported as the events are all uncore.

Note: Please contact the maintainer for a complete list of events supported for the PMU devices in the SoC and its information if needed.

18.2 HiSilicon PCIe Performance Monitoring Unit (PMU)

On Hip09, HiSilicon PCIe Performance Monitoring Unit (PMU) could monitor bandwidth, latency, bus utilization and buffer occupancy data of PCIe.

Each PCIe Core has a PMU to monitor multi Root Ports of this PCIe Core and all Endpoints downstream these Root Ports.

18.2.1 HiSilicon PCIe PMU driver

The PCIe PMU driver registers a perf PMU with the name of its sicl-id and PCIe Core id.:

```
/sys/bus/event_source/hisi_pcie<sicl>_core<core>
```

PMU driver provides description of available events and filter options in sysfs, see /sys/bus/event_source/devices/hisi_pcie<sicl>_core<core>.

The "format" directory describes all formats of the config (events) and config1 (filter options) fields of the perf_event_attr structure. The "events" directory describes all documented events shown in perf list.

The "identifier" sysfs file allows users to identify the version of the PMU hardware device.

The "bus" sysfs file allows users to get the bus number of Root Ports monitored by PMU.

Example usage of perf:

```
$# perf list
hisi_pcie0_core0/rx_mwr_latency/ [kernel PMU event]
hisi_pcie0_core0/rx_mwr_cnt/ [kernel PMU event]
--------------------------------------------

$# perf stat -e hisi_pcie0_core0/rx_mwr_latency/
```

```
$# perf stat -e hisi_pcie0_core0/rx_mwr_cnt/
$# perf stat -g -e hisi_pcie0_core0/rx_mwr_latency/ -e hisi_pcie0_core0/rx_mwr_
↪cnt/
```

The current driver does not support sampling. So "perf record" is unsupported. Also attach to a task is unsupported for PCIe PMU.

Filter options

1. Target filter

   PMU could only monitor the performance of traffic downstream target Root Ports or downstream target Endpoint. PCIe PMU driver support "port" and "bdf" interfaces for users, and these two interfaces aren't supported at the same time.

   - port

     "port" filter can be used in all PCIe PMU events, target Root Port can be selected by configuring the 16-bits-bitmap "port". Multi ports can be selected for AP-layer-events, and only one port can be selected for TL/DL-layer-events.

     For example, if target Root Port is 0000:00:00.0 (x8 lanes), bit0 of bitmap should be set, port=0x1; if target Root Port is 0000:00:04.0 (x4 lanes), bit8 is set, port=0x100; if these two Root Ports are both monitored, port=0x101.

     Example usage of perf:

     ```
     $# perf stat -e hisi_pcie0_core0/rx_mwr_latency,port=0x1/ sleep 5
     ```

   - bdf

     "bdf" filter can only be used in bandwidth events, target Endpoint is selected by configuring BDF to "bdf". Counter only counts the bandwidth of message requested by target Endpoint.

     For example, "bdf=0x3900" means BDF of target Endpoint is 0000:39:00.0.

     Example usage of perf:

     ```
     $# perf stat -e hisi_pcie0_core0/rx_mrd_flux,bdf=0x3900/ sleep 5
     ```

2. Trigger filter

   Event statistics start when the first time TLP length is greater/smaller than trigger condition. You can set the trigger condition by writing "trig_len", and set the trigger mode by writing "trig_mode". This filter can only be used in bandwidth events.

   For example, "trig_len=4" means trigger condition is 2^4 DW, "trig_mode=0" means statistics start when TLP length > trigger condition, "trig_mode=1" means start when TLP length < condition.

   Example usage of perf:

   ```
   $# perf stat -e hisi_pcie0_core0/rx_mrd_flux,trig_len=0x4,trig_mode=1/␣
   ↪sleep 5
   ```

3. Threshold filter

   Counter counts when TLP length within the specified range. You can set the threshold by writing "thr_len", and set the threshold mode by writing "thr_mode". This filter can only be used in bandwidth events.

   For example, "thr_len=4" means threshold is 2^4 DW, "thr_mode=0" means counter counts when TLP length >= threshold, and "thr_mode=1" means counts when TLP length < threshold.

   Example usage of perf:

```
$# perf stat -e hisi_pcie0_core0/rx_mrd_flux,thr_len=0x4,thr_mode=1/ sleep␣
↪5
```

4. TLP Length filter

   When counting bandwidth, the data can be composed of certain parts of TLP packets. You can specify it through "len_mode":

   - 2'b00: Reserved (Do not use this since the behaviour is undefined)
   - 2'b01: Bandwidth of TLP payloads
   - 2'b10: Bandwidth of TLP headers
   - 2'b11: Bandwidth of both TLP payloads and headers

   For example, "len_mode=2" means only counting the bandwidth of TLP headers and "len_mode=3" means the final bandwidth data is composed of both TLP headers and payloads. Default value if not specified is 2'b11.

   Example usage of perf:

```
$# perf stat -e hisi_pcie0_core0/rx_mrd_flux,len_mode=0x1/ sleep 5
```

18.3 HNS3 Performance Monitoring Unit (PMU)

HNS3(HiSilicon network system 3) Performance Monitoring Unit (PMU) is an End Point device to collect performance statistics of HiSilicon SoC NIC. On Hip09, each SICL(Super I/O cluster) has one PMU device.

HNS3 PMU supports collection of performance statistics such as bandwidth, latency, packet rate and interrupt rate.

Each HNS3 PMU supports 8 hardware events.

18.3.1 HNS3 PMU driver

The HNS3 PMU driver registers a perf PMU with the name of its sicl id.:

```
/sys/devices/hns3_pmu_sicl_<sicl_id>
```

PMU driver provides description of available events, filter modes, format, identifier and cpumask in sysfs.

The "events" directory describes the event code of all supported events shown in perf list.

The "filtermode" directory describes the supported filter modes of each event.

The "format" directory describes all formats of the config (events) and config1 (filter options) fields of the perf_event_attr structure.

The "identifier" file shows version of PMU hardware device.

The "bdf_min" and "bdf_max" files show the supported bdf range of each pmu device.

The "hw_clk_freq" file shows the hardware clock frequency of each pmu device.

Example usage of checking event code and subevent code:

```
$# cat /sys/devices/hns3_pmu_sicl_0/events/dly_tx_normal_to_mac_time
config=0x00204
$# cat /sys/devices/hns3_pmu_sicl_0/events/dly_tx_normal_to_mac_packet_num
config=0x10204
```

Each performance statistic has a pair of events to get two values to calculate real performance data in userspace.

The bits 0~15 of config (here 0x0204) are the true hardware event code. If two events have same value of bits 0~15 of config, that means they are event pair. And the bit 16 of config indicates getting counter 0 or counter 1 of hardware event.

After getting two values of event pair in userspace, the formula of computation to calculate real performance data is::

```
counter 0 / counter 1
```

Example usage of checking supported filter mode:

```
$# cat /sys/devices/hns3_pmu_sicl_0/filtermode/bw_ssu_rpu_byte_num
filter mode supported: global/port/port-tc/func/func-queue/
```

Example usage of perf:

```
$# perf list
hns3_pmu_sicl_0/bw_ssu_rpu_byte_num/ [kernel PMU event]
hns3_pmu_sicl_0/bw_ssu_rpu_time/     [kernel PMU event]
------------------------------------------

$# perf stat -g -e hns3_pmu_sicl_0/bw_ssu_rpu_byte_num,global=1/ -e hns3_pmu_
↪sicl_0/bw_ssu_rpu_time,global=1/ -I 1000
or
$# perf stat -g -e hns3_pmu_sicl_0/config=0x00002,global=1/ -e hns3_pmu_sicl_0/
↪config=0x10002,global=1/ -I 1000
```

Filter modes

1. global mode PMU collect performance statistics for all HNS3 PCIe functions of IO DIE. Set the "global" filter option to 1 will enable this mode. Example usage of perf:

```
$# perf stat -a -e hns3_pmu_sicl_0/config=0x1020F,global=1/ -I 1000
```

2. port mode PMU collect performance statistic of one whole physical port. The port id is same as mac id. The "tc" filter option must be set to 0xF in this mode, here tc stands for traffic class.

Example usage of perf:

```
$# perf stat -a -e hns3_pmu_sicl_0/config=0x1020F,port=0,tc=0xF/ -I 1000
```

3. port-tc mode PMU collect performance statistic of one tc of physical port. The port id is same as mac id. The "tc" filter option must be set to 0 ~ 7 in this mode. Example usage of perf:

```
$# perf stat -a -e hns3_pmu_sicl_0/config=0x1020F,port=0,tc=0/ -I 1000
```

4. func mode PMU collect performance statistic of one PF/VF. The function id is BDF of PF/VF, its conversion formula:

```
func = (bus << 8) + (device << 3) + (function)
```

for example:
BDF func 35:00.0 0x3500 35:00.1 0x3501 35:01.0 0x3508

In this mode, the "queue" filter option must be set to 0xFFFF. Example usage of perf:

```
$# perf stat -a -e hns3_pmu_sicl_0/config=0x1020F,bdf=0x3500,queue=0xFFFF/ -I␣
↪1000
```

5. func-queue mode PMU collect performance statistic of one queue of PF/VF. The function id is BDF of PF/VF, the "queue" filter option must be set to the exact queue id of function. Example usage of perf:

```
$# perf stat -a -e hns3_pmu_sicl_0/config=0x1020F,bdf=0x3500,queue=0/ -I 1000
```

6. func-intr mode PMU collect performance statistic of one interrupt of PF/VF. The function id is BDF of PF/VF, the "intr" filter option must be set to the exact interrupt id of function. Example usage of perf:

```
$# perf stat -a -e hns3_pmu_sicl_0/config=0x00301,bdf=0x3500,intr=0/ -I 1000
```

18.4 Freescale i.MX8 DDR Performance Monitoring Unit (PMU)

There are no performance counters inside the DRAM controller, so performance signals are brought out to the edge of the controller where a set of 4 x 32 bit counters is implemented. This is controlled by the CSV modes programmed in counter control register which causes a large number of PERF signals to be generated.

Selection of the value for each counter is done via the config registers. There is one register for each counter. Counter 0 is special in that it always counts "time" and when expired causes a lock on itself and the other counters and an interrupt is raised. If any other counter overflows, it continues counting, and no interrupt is raised.

The "format" directory describes format of the config (event ID) and config1/2 (AXI filter setting) fields of the perf_event_attr structure, see /sys/bus/event_source/ devices/imx8_ddr0/format/. The "events" directory describes the events types hardware supported that can be used with perf tool, see /sys/bus/event_source/ devices/imx8_ddr0/events/. The "caps" directory describes filter features implemented in DDR PMU, see /sys/bus/events_source/devices/imx8_ddr0/caps/.

```
perf stat -a -e imx8_ddr0/cycles/ cmd
perf stat -a -e imx8_ddr0/read/,imx8_ddr0/write/ cmd
```

AXI filtering is only used by CSV modes 0x41 (axid-read) and 0x42 (axid-write) to count reading or writing matches filter setting. Filter setting is various from different DRAM controller implementations, which is distinguished by quirks in the driver. You also can dump info from userspace, "caps" directory show the type of AXI filter (filter, enhanced_filter and super_filter). Value 0 for un-supported, and value 1 for supported.

- With DDR_CAP_AXI_ID_FILTER quirk(filter: 1, enhanced_filter: 0, super_filter: 0). Filter is defined with two configuration parts: --AXI_ID defines AxID matching value. --AXI_MASKING defines which bits of AxID are meaningful for the matching.
  - 0: corresponding bit is masked.
  - 1: corresponding bit is not masked, i.e. used to do the matching.

  AXI_ID and AXI_MASKING are mapped on DPCR1 register in performance counter. When non-masked bits are matching corresponding AXI_ID bits then counter is incremented. Perf counter is incremented if:

  ```
  AxID && AXI_MASKING == AXI_ID && AXI_MASKING
  ```

  This filter doesn't support filter different AXI ID for axid-read and axid-write event at the same time as this filter is shared between counters.

  ```
  perf stat -a -e imx8_ddr0/axid-read,axi_mask=0xMMMM,axi_id=0xDDDD/ cmd
  perf stat -a -e imx8_ddr0/axid-write,axi_mask=0xMMMM,axi_id=0xDDDD/ cmd
  ```

  Note: axi_mask is inverted in userspace(i.e. set bits are bits to mask), and it will be reverted in driver automatically. so that the user can just specify axi_id to monitor a specific id, rather than having to specify axi_mask.

  ```
  perf stat -a -e imx8_ddr0/axid-read,axi_id=0x12/ cmd, which will monitor␣
  ↪ARID=0x12
  ```

- With DDR_CAP_AXI_ID_FILTER_ENHANCED quirk(filter: 1, enhanced_filter: 1, super_filter: 0). This is an extension to the DDR_CAP_AXI_ID_FILTER quirk which permits counting the number of bytes (as opposed to the number of bursts) from DDR read and write transactions concurrently with another set of data counters.
- With DDR_CAP_AXI_ID_PORT_CHANNEL_FILTER quirk(filter: 0, enhanced_filter: 0, super_filter: 1). There is a limitation in previous AXI filter, it cannot filter different IDs at the same time as the filter is shared between counters. This quirk is the extension of AXI ID filter. One improvement is that counter 1-3 has their own filter, means that it supports concurrently filter various IDs. Another improvement is that counter 1-3 supports AXI PORT and CHANNEL selection. Support selecting address channel or data channel.

  Filter is defined with 2 configuration registers per counter 1-3. --Counter N MASK COMP register - including AXI_ID and AXI_MASKING. --Counter N MUX CNTL register - including AXI CHANNEL and AXI PORT.

  - 0: address channel
  - 1: data channel

  PMU in DDR subsystem, only one single port0 exists, so axi_port is reserved which should be 0.

```
perf stat -a -e imx8_ddr0/axid-read,axi_mask=0xMMMM,axi_id=0xDDDD,axi_
↪channel=0xH/ cmd
perf stat -a -e imx8_ddr0/axid-write,axi_mask=0xMMMM,axi_id=0xDDDD,axi_
↪channel=0xH/ cmd
```

Note: axi_channel is inverted in userspace, and it will be reverted in driver automatically. So that users do not need specify axi_channel if want to monitor data channel from DDR transactions, since data channel is more meaningful.

18.5 Qualcomm Technologies Level-2 Cache Performance Monitoring Unit (PMU)

This driver supports the L2 cache clusters found in Qualcomm Technologies Centriq SoCs. There are multiple physical L2 cache clusters, each with their own PMU. Each cluster has one or more CPUs associated with it.

There is one logical L2 PMU exposed, which aggregates the results from the physical PMUs.

The driver provides a description of its available events and configuration options in sysfs, see /sys/devices/l2cache_0.

The "format" directory describes the format of the events.

Events can be envisioned as a 2-dimensional array. Each column represents a group of events. There are 8 groups. Only one entry from each group can be in use at a time. If multiple events from the same group are specified, the conflicting events cannot be counted at the same time.

Events are specified as 0xCCG, where CC is 2 hex digits specifying the code (array row) and G specifies the group (column) 0-7.

In addition there is a cycle counter event specified by the value 0xFE which is outside the above scheme.

The driver provides a ”cpumask” sysfs attribute which contains a mask consisting of one CPU per cluster which will be used to handle all the PMU events on that cluster.

Examples for use with perf:

```
perf stat -e l2cache_0/config=0x001/,l2cache_0/config=0x042/ -a sleep 1

perf stat -e l2cache_0/config=0xfe/ -C 2 sleep 1
```

The driver does not support sampling, therefore ”perf record” will not work. Per-task perf sessions are not supported.

18.6 Qualcomm Datacenter Technologies L3 Cache Performance Monitoring Unit (PMU)

This driver supports the L3 cache PMUs found in Qualcomm Datacenter Technologies Centriq SoCs. The L3 cache on these SOCs is composed of multiple slices, shared by all cores within a socket. Each slice is exposed as a separate uncore perf PMU with device name l3cache_<socket>_<instance>. User space is responsible for aggregating across slices.

The driver provides a description of its available events and configuration options in sysfs, see /sys/devices/l3cache*. Given that these are uncore PMUs the driver also exposes a ”cpumask” sysfs attribute which contains a mask consisting of one CPU per socket which will be used to handle all the PMU events on that socket.

The hardware implements 32bit event counters and has a flat 8bit event space exposed via the ”event” format attribute. In addition to the 32bit physical counters the driver supports virtual 64bit hardware counters by using hardware counter chaining. This feature is exposed via the ”lc” (long counter) format flag. E.g.:

```
perf stat -e l3cache_0_0/read-miss,lc/
```

Given that these are uncore PMUs the driver does not support sampling, therefore ”perf record” will not work. Per-task perf sessions are not supported.

18.7 ARM Cache Coherent Network

CCN-504 is a ring-bus interconnect consisting of 11 crosspoints (XPs), with each crosspoint supporting up to two device ports, so nodes (devices) 0 and 1 are connected to crosspoint 0, nodes 2 and 3 to crosspoint 1 etc.

18.7.1 PMU (perf) driver

The CCN driver registers a perf PMU driver, which provides description of available events and configuration options in sysfs, see /sys/bus/event_source/devices/ccn*.

The "format" directory describes format of the config, config1 and config2 fields of the perf_event_attr structure. The "events" directory provides configuration templates for all documented events, that can be used with perf tool. For example "xp_valid_flit" is an equivalent of "type=0x8,event=0x4". Other parameters must be explicitly specified.

For events originating from device, "node" defines its index.

Crosspoint PMU events require "xp" (index), "bus" (bus number) and "vc" (virtual channel ID).

Crosspoint watchpoint-based events (special "event" value 0xfe) require "xp" and "vc" as above plus "port" (device port index), "dir" (transmit/receive direction), comparator values ("cmp_l" and "cmp_h") and "mask", being index of the comparator mask.

Masks are defined separately from the event description (due to limited number of the config values) in the "cmp_mask" directory, with first 8 configurable by user and additional 4 hard-coded for the most frequent use cases.

Cycle counter is described by a "type" value 0xff and does not require any other settings.

The driver also provides a "cpumask" sysfs attribute, which contains a single CPU ID, of the processor which will be used to handle all the CCN PMU events. It is recommended that the user space tools request the events on this processor (if not, the perf_event->cpu value will be overwritten anyway). In case of this processor being offlined, the events are migrated to another one and the attribute is updated.

Example of perf tool use:

```
/ # perf list | grep ccn
  ccn/cycles/                                        [Kernel PMU event]
<...>
  ccn/xp_valid_flit,xp=?,port=?,vc=?,dir=?/          [Kernel PMU event]
<...>

/ # perf stat -a -e ccn/cycles/,ccn/xp_valid_flit,xp=1,port=0,vc=1,dir=1/ \
                                                                      sleep 1
```

The driver does not support sampling, therefore "perf record" will not work. Per-task (without "-a") perf sessions are not supported.

18.8 Arm Coherent Mesh Network PMU

CMN-600 is a configurable mesh interconnect consisting of a rectangular grid of crosspoints (XPs), with each crosspoint supporting up to two device ports to which various AMBA CHI agents are attached.

CMN implements a distributed PMU design as part of its debug and trace functionality. This consists of a local monitor (DTM) at every XP, which counts up to 4 event signals from the connected device nodes and/or the XP itself. Overflow from these local counters is accumulated in up to 8 global counters implemented by the main controller (DTC), which provides overall PMU control and interrupts for global counter overflow.

18.8.1 PMU events

The PMU driver registers a single PMU device for the whole interconnect, see /sys/bus/event_source/devices/arm_cmn_0. Multi-chip systems may link more than one CMN together via external CCIX links - in this situation, each mesh counts its own events entirely independently, and additional PMU devices will be named arm_cmn_{1..n}.

Most events are specified in a format based directly on the TRM definitions - ”type” selects the respective node type, and ”eventid” the event number. Some events require an additional occupancy ID, which is specified by ”occupid”.

- Since RN-D nodes do not have any distinct events from RN-I nodes, they are treated as the same type (0xa), and the common event templates are named ”rnid_*”.
- The cycle counter is treated as a synthetic event belonging to the DTC node (”type” == 0x3, ”eventid” is ignored).
- XP events also encode the port and channel in the ”eventid” field, to match the underlying pmu_event0_id encoding for the pmu_event_sel register. The event templates are named with prefixes to cover all permutations.

By default each event provides an aggregate count over all nodes of the given type. To target a specific node, ”bynodeid” must be set to 1 and ”nodeid” to the appropriate value derived from the CMN configuration (as defined in the ”Node ID Mapping” section of the TRM).

18.8.2 Watchpoints

The PMU can also count watchpoint events to monitor specific flit traffic. Watchpoints are treated as a synthetic event type, and like PMU events can be global or targeted with a particular XP's ”nodeid” value. Since the watchpoint direction is otherwise implicit in the underlying register selection, separate events are provided for flit uploads and downloads.

The flit match value and mask are passed in config1 and config2 (”val” and ”mask” respectively). ”wp_dev_sel”, ”wp_chn_sel”, ”wp_grp” and ”wp_exclusive” are specified per the TRM definitions for dtm_wp_config0. Where a watchpoint needs to match fields from both match groups on the REQ or SNP channel, it can be specified as two events - one for each group - with the same nonzero ”combine” value. The count for such a pair of combined events will be attributed to the primary match. Watchpoint events with a ”combine” value of 0 are considered independent and will count individually.

18.9 APM X-Gene SoC Performance Monitoring Unit (PMU)

X-Gene SoC PMU consists of various independent system device PMUs such as L3 cache(s), I/O bridge(s), memory controller bridge(s) and memory controller(s). These PMU devices are loosely architected to follow the same model as the PMU for ARM cores. The PMUs share the same top level interrupt and status CSR region.

18.9.1 PMU (perf) driver

The xgene-pmu driver registers several perf PMU drivers. Each of the perf driver provides description of its available events and configuration options in sysfs, see /sys/devices/<l3cX/iobX/mcbX/mcX>/.

The "format" directory describes format of the config (event ID), config1 (agent ID) fields of the perf_event_attr structure. The "events" directory provides configuration templates for all supported event types that can be used with perf tool. For example, "l3c0/bank-fifo-full/" is an equivalent of "l3c0/config=0x0b/".

Most of the SoC PMU has a specific list of agent ID used for monitoring performance of a specific datapath. For example, agents of a L3 cache can be a specific CPU or an I/O bridge. Each PMU has a set of 2 registers capable of masking the agents from which the request come from. If the bit with the bit number corresponding to the agent is set, the event is counted only if it is caused by a request from that agent. Each agent ID bit is inversely mapped to a corresponding bit in "config1" field. By default, the event will be counted for all agent requests (config1 = 0x0). For all the supported agents of each PMU, please refer to APM X-Gene User Manual.

Each perf driver also provides a "cpumask" sysfs attribute, which contains a single CPU ID of the processor which will be used to handle all the PMU events.

Example for perf tool use:

```
/ # perf list | grep -e l3c -e iob -e mcb -e mc
  l3c0/ackq-full/                                    [Kernel PMU event]
<...>
  mcb1/mcb-csw-stall/                                [Kernel PMU event]

/ # perf stat -a -e l3c0/read-miss/,mcb1/csw-write-request/ sleep 1

/ # perf stat -a -e l3c0/read-miss,config1=0xfffffffffffffffe/ sleep 1
```

The driver does not support sampling, therefore "perf record" will not work. Per-task (without "-a") perf sessions are not supported.

18.10 ARM DynamIQ Shared Unit (DSU) PMU

ARM DynamIQ Shared Unit integrates one or more cores with an L3 memory system, control logic and external interfaces to form a multicore cluster. The PMU allows counting the various events related to the L3 cache, Snoop Control Unit etc, using 32bit independent counters. It also provides a 64bit cycle counter.

The PMU can only be accessed via CPU system registers and are common to the cores connected to the same DSU. Like most of the other uncore PMUs, DSU PMU doesn't support process specific events and cannot be used in sampling mode.

The DSU provides a bitmap for a subset of implemented events via hardware registers. There is no way for the driver to determine if the other events are available or not. Hence the driver exposes only those events advertised by the DSU, in "events" directory under:

```
/sys/bus/event_sources/devices/arm_dsu_<N>/
```

The user should refer to the TRM of the product to figure out the supported events and use the raw event code for the unlisted events.

The driver also exposes the CPUs connected to the DSU instance in ”associated_cpus”.

e.g usage:

```
perf stat -a -e arm_dsu_0/cycles/
```

18.11 Cavium ThunderX2 SoC Performance Monitoring Unit (PMU UNCORE)

The ThunderX2 SoC PMU consists of independent, system-wide, per-socket PMUs such as the Level 3 Cache (L3C), DDR4 Memory Controller (DMC) and Cavium Coherent Processor Interconnect (CCPI2).

The DMC has 8 interleaved channels and the L3C has 16 interleaved tiles. Events are counted for the default channel (i.e. channel 0) and prorated to the total number of channels/tiles.

The DMC and L3C support up to 4 counters, while the CCPI2 supports up to 8 counters. Counters are independently programmable to different events and can be started and stopped individually. None of the counters support an overflow interrupt. DMC and L3C counters are 32-bit and read every 2 seconds. The CCPI2 counters are 64-bit and assumed not to overflow in normal operation.

PMU UNCORE (perf) driver:

The thunderx2_pmu driver registers per-socket perf PMUs for the DMC and L3C devices. Each PMU can be used to count up to 4 (DMC/L3C) or up to 8 (CCPI2) events simultaneously. The PMUs provide a description of their available events and configuration options under sysfs, see /sys/devices/uncore_<l3c_S/dmc_S/ccpi2_S/>; S is the socket id.

The driver does not support sampling, therefore ”perf record” will not work. Per-task perf sessions are also not supported.

Examples:

```
# perf stat -a -e uncore_dmc_0/cnt_cycles/ sleep 1

# perf stat -a -e \
uncore_dmc_0/cnt_cycles/,\
uncore_dmc_0/data_transfers/,\
uncore_dmc_0/read_txns/,\
uncore_dmc_0/write_txns/ sleep 1

# perf stat -a -e \
uncore_l3c_0/read_request/,\
uncore_l3c_0/read_hit/,\
uncore_l3c_0/inv_request/,\
uncore_l3c_0/inv_hit/ sleep 1
```

18.12 Alibaba's T-Head SoC Uncore Performance Monitoring Unit (PMU)

The Yitian 710, custom-built by Alibaba Group's chip development business, T-Head, implements uncore PMU for performance and functional debugging to facilitate system maintenance.

18.12.1 DDR Sub-System Driveway (DRW) PMU Driver

Yitian 710 employs eight DDR5/4 channels, four on each die. Each DDR5 channel is independent of others to service system memory requests. And one DDR5 channel is split into two independent sub-channels. The DDR Sub-System Driveway implements separate PMUs for each sub-channel to monitor various performance metrics.

The Driveway PMU devices are named as ali_drw_<sys_base_addr> with perf. For example, ali_drw_21000 and ali_drw_21080 are two PMU devices for two sub-channels of the same channel in die 0. And the PMU device of die 1 is prefixed with ali_drw_400XXXXX, e.g. ali_drw_40021000.

Each sub-channel has 36 PMU counters in total, which is classified into four groups:

- Group 0: PMU Cycle Counter. This group has one pair of counters pmu_cycle_cnt_low and pmu_cycle_cnt_high, that is used as the cycle count based on DDRC core clock.
- Group 1: PMU Bandwidth Counters. This group has 8 counters that are used to count the total access number of either the eight bank groups in a selected rank, or four ranks separately in the first 4 counters. The base transfer unit is 64B.
- Group 2: PMU Retry Counters. This group has 10 counters, that intend to count the total retry number of each type of uncorrectable error.
- Group 3: PMU Common Counters. This group has 16 counters, that are used to count the common events.

For now, the Driveway PMU driver only uses counters in group 0 and group 3.

The DDR Controller (DDRCTL) and DDR PHY combine to create a complete solution for connecting an SoC application bus to DDR memory devices. The DDRCTL receives transactions Host Interface (HIF) which is custom-defined by Synopsys. These transactions are queued internally and scheduled for access while satisfying the SDRAM protocol timing requirements, transaction priorities, and dependencies between the transactions. The DDRCTL in turn issues commands on the DDR PHY Interface (DFI) to the PHY module, which launches and captures data to and from the SDRAM. The driveway PMUs have hardware logic to gather statistics and performance logging signals on HIF, DFI, etc.

By counting the READ, WRITE and RMW commands sent to the DDRC through the HIF interface, we could calculate the bandwidth. Example usage of counting memory data bandwidth:

```
perf stat \
  -e ali_drw_21000/hif_wr/ \
  -e ali_drw_21000/hif_rd/ \
  -e ali_drw_21000/hif_rmw/ \
  -e ali_drw_21000/cycle/ \
  -e ali_drw_21080/hif_wr/ \
  -e ali_drw_21080/hif_rd/ \
```

```
  -e ali_drw_21080/hif_rmw/ \
  -e ali_drw_21080/cycle/ \
  -e ali_drw_23000/hif_wr/ \
  -e ali_drw_23000/hif_rd/ \
  -e ali_drw_23000/hif_rmw/ \
  -e ali_drw_23000/cycle/ \
  -e ali_drw_23080/hif_wr/ \
  -e ali_drw_23080/hif_rd/ \
  -e ali_drw_23080/hif_rmw/ \
  -e ali_drw_23080/cycle/ \
  -e ali_drw_25000/hif_wr/ \
  -e ali_drw_25000/hif_rd/ \
  -e ali_drw_25000/hif_rmw/ \
  -e ali_drw_25000/cycle/ \
  -e ali_drw_25080/hif_wr/ \
  -e ali_drw_25080/hif_rd/ \
  -e ali_drw_25080/hif_rmw/ \
  -e ali_drw_25080/cycle/ \
  -e ali_drw_27000/hif_wr/ \
  -e ali_drw_27000/hif_rd/ \
  -e ali_drw_27000/hif_rmw/ \
  -e ali_drw_27000/cycle/ \
  -e ali_drw_27080/hif_wr/ \
  -e ali_drw_27080/hif_rd/ \
  -e ali_drw_27080/hif_rmw/ \
  -e ali_drw_27080/cycle/ -- sleep 10
```

Example usage of counting all memory read/write bandwidth by metric:

```
perf stat -M ddr_read_bandwidth.all -- sleep 10
perf stat -M ddr_write_bandwidth.all -- sleep 10
```

The average DRAM bandwidth can be calculated as follows:

- Read Bandwidth = perf_hif_rd * DDRC_WIDTH * DDRC_Freq / DDRC_Cycle
- Write Bandwidth = (perf_hif_wr + perf_hif_rmw) * DDRC_WIDTH * DDRC_Freq / DDRC_Cycle

Here, DDRC_WIDTH = 64 bytes.

The current driver does not support sampling. So ”perf record” is unsupported. Also attach to a task is unsupported as the events are all uncore.

18.13 Synopsys DesignWare Cores (DWC) PCIe Performance Monitoring Unit (PMU)

18.13.1 DesignWare Cores (DWC) PCIe PMU

The PMU is a PCIe configuration space register block provided by each PCIe Root Port in a Vendor-Specific Extended Capability named RAS D.E.S (Debug, Error injection, and Statistics).

As the name indicates, the RAS DES capability supports system level debugging, AER error injection, and collection of statistics. To facilitate collection of statistics, Synopsys DesignWare Cores PCIe controller provides the following two features:

- one 64-bit counter for Time Based Analysis (RX/TX data throughput and time spent in each low-power LTSSM state) and
- one 32-bit counter for Event Counting (error and non-error events for a specified lane)

Note: There is no interrupt for counter overflow.

Time Based Analysis

Using this feature you can obtain information regarding RX/TX data throughput and time spent in each low-power LTSSM state by the controller. The PMU measures data in two categories:

- Group#0: Percentage of time the controller stays in LTSSM states.
- Group#1: Amount of data processed (Units of 16 bytes).

Lane Event counters

Using this feature you can obtain Error and Non-Error information in specific lane by the controller. The PMU event is selected by all of:

- Group i
- Event j within the Group i
- Lane k

Some of the events only exist for specific configurations.

18.13.2 DesignWare Cores (DWC) PCIe PMU Driver

This driver adds PMU devices for each PCIe Root Port named based on the BDF of the Root Port. For example,

```
30:03.0 PCI bridge: Device 1ded:8000 (rev 01)
```

the PMU device name for this Root Port is dwc_rootport_3018.

The DWC PCIe PMU driver registers a perf PMU driver, which provides description of available events and configuration options in sysfs, see /sys/bus/event_source/devices/dwc_rootport_{bdf}.

The "format" directory describes format of the config fields of the perf_event_attr structure. The "events" directory provides configuration templates for all documented events. For example, "Rx_PCIe_TLP_Data_Payload" is an equivalent of "eventid=0x22,type=0x1".

The "perf list" command shall list the available events from sysfs, e.g.:

```
$# perf list | grep dwc_rootport
<...>
dwc_rootport_3018/Rx_PCIe_TLP_Data_Payload/        [Kernel PMU event]
<...>
dwc_rootport_3018/rx_memory_read,lane=?/                [Kernel PMU event]
```

Time Based Analysis Event Usage

Example usage of counting PCIe RX TLP data payload (Units of bytes):

```
$# perf stat -a -e dwc_rootport_3018/Rx_PCIe_TLP_Data_Payload/
```

The average RX/TX bandwidth can be calculated using the following formula:

> PCIe RX Bandwidth = Rx_PCIe_TLP_Data_Payload / Measure_Time_Window PCIe TX Bandwidth = Tx_PCIe_TLP_Data_Payload / Measure_Time_Window

Lane Event Usage

Each lane has the same event set and to avoid generating a list of hundreds of events, the user need to specify the lane ID explicitly, e.g.:

```
$# perf stat -a -e dwc_rootport_3018/rx_memory_read,lane=4/
```

The driver does not support sampling, therefore "perf record" will not work. Per-task (without "-a") perf sessions are not supported.

18.14 NVIDIA Tegra SoC Uncore Performance Monitoring Unit (PMU)

The NVIDIA Tegra SoC includes various system PMUs to measure key performance metrics like memory bandwidth, latency, and utilization:

- Scalable Coherency Fabric (SCF)
- NVLink-C2C0
- NVLink-C2C1
- CNVLink
- PCIE

18.14.1 PMU Driver

The PMUs in this document are based on ARM CoreSight PMU Architecture as described in document: ARM IHI 0091. Since this is a standard architecture, the PMUs are managed by a common driver "arm-cs-arch-pmu". This driver describes the available events and configuration of each PMU in sysfs. Please see the sections below to get the sysfs path of each PMU. Like other uncore PMU drivers, the driver provides "cpumask" sysfs attribute to show the CPU id used to handle the PMU event. There is also "associated_cpus" sysfs attribute, which contains a list of CPUs associated with the PMU instance.

18.14.2 SCF PMU

The SCF PMU monitors system level cache events, CPU traffic, and strongly-ordered (SO) PCIE write traffic to local/remote memory. Please see *Traffic Coverage* for more info about the PMU traffic coverage.

The events and configuration options of this PMU device are described in sysfs, see /sys/bus/event_sources/devices/nvidia_scf_pmu_<socket-id>.

Example usage:

- Count event id 0x0 in socket 0:

```
perf stat -a -e nvidia_scf_pmu_0/event=0x0/
```

- Count event id 0x0 in socket 1:

```
perf stat -a -e nvidia_scf_pmu_1/event=0x0/
```

18.14.3 NVLink-C2C0 PMU

The NVLink-C2C0 PMU monitors incoming traffic from a GPU/CPU connected with NVLink-C2C (Chip-2-Chip) interconnect. The type of traffic captured by this PMU varies dependent on the chip configuration:

- NVIDIA Grace Hopper Superchip: Hopper GPU is connected with Grace SoC.

  In this config, the PMU captures GPU ATS translated or EGM traffic from the GPU.

- NVIDIA Grace CPU Superchip: two Grace CPU SoCs are connected.

  In this config, the PMU captures read and relaxed ordered (RO) writes from PCIE device of the remote SoC.

Please see *Traffic Coverage* for more info about the PMU traffic coverage.

The events and configuration options of this PMU device are described in sysfs, see /sys/bus/event_sources/devices/nvidia_nvlink_c2c0_pmu_<socket-id>.

Example usage:

- Count event id 0x0 from the GPU/CPU connected with socket 0:

```
perf stat -a -e nvidia_nvlink_c2c0_pmu_0/event=0x0/
```

- Count event id 0x0 from the GPU/CPU connected with socket 1:

```
perf stat -a -e nvidia_nvlink_c2c0_pmu_1/event=0x0/
```

- Count event id 0x0 from the GPU/CPU connected with socket 2:

```
perf stat -a -e nvidia_nvlink_c2c0_pmu_2/event=0x0/
```

- Count event id 0x0 from the GPU/CPU connected with socket 3:

```
perf stat -a -e nvidia_nvlink_c2c0_pmu_3/event=0x0/
```

18.14.4 NVLink-C2C1 PMU

The NVLink-C2C1 PMU monitors incoming traffic from a GPU connected with NVLink-C2C (Chip-2-Chip) interconnect. This PMU captures untranslated GPU traffic, in contrast with NvLink-C2C0 PMU that captures ATS translated traffic. Please see *Traffic Coverage* for more info about the PMU traffic coverage.

The events and configuration options of this PMU device are described in sysfs, see /sys/bus/event_sources/devices/nvidia_nvlink_c2c1_pmu_<socket-id>.

Example usage:

- Count event id 0x0 from the GPU connected with socket 0:

```
perf stat -a -e nvidia_nvlink_c2c1_pmu_0/event=0x0/
```

- Count event id 0x0 from the GPU connected with socket 1:

```
perf stat -a -e nvidia_nvlink_c2c1_pmu_1/event=0x0/
```

- Count event id 0x0 from the GPU connected with socket 2:

```
perf stat -a -e nvidia_nvlink_c2c1_pmu_2/event=0x0/
```

- Count event id 0x0 from the GPU connected with socket 3:

```
perf stat -a -e nvidia_nvlink_c2c1_pmu_3/event=0x0/
```

18.14.5 CNVLink PMU

The CNVLink PMU monitors traffic from GPU and PCIE device on remote sockets to local memory. For PCIE traffic, this PMU captures read and relaxed ordered (RO) write traffic. Please see *Traffic Coverage* for more info about the PMU traffic coverage.

The events and configuration options of this PMU device are described in sysfs, see /sys/bus/event_sources/devices/nvidia_cnvlink_pmu_<socket-id>.

Each SoC socket can be connected to one or more sockets via CNVLink. The user can use "rem_socket" bitmap parameter to select the remote socket(s) to monitor. Each bit represents the socket number, e.g. "rem_socket=0xE" corresponds to socket 1 to 3. /sys/bus/event_sources/devices/nvidia_cnvlink_pmu_<socket-id>/format/rem_socket shows the valid bits that can be set in the "rem_socket" parameter.

The PMU can not distinguish the remote traffic initiator, therefore it does not provide filter to select the traffic source to monitor. It reports combined traffic from remote GPU and PCIE devices.

Example usage:

- Count event id 0x0 for the traffic from remote socket 1, 2, and 3 to socket 0:

```
perf stat -a -e nvidia_cnvlink_pmu_0/event=0x0,rem_socket=0xE/
```

- Count event id 0x0 for the traffic from remote socket 0, 2, and 3 to socket 1:

```
perf stat -a -e nvidia_cnvlink_pmu_1/event=0x0,rem_socket=0xD/
```

- Count event id 0x0 for the traffic from remote socket 0, 1, and 3 to socket 2:

```
perf stat -a -e nvidia_cnvlink_pmu_2/event=0x0,rem_socket=0xB/
```

- Count event id 0x0 for the traffic from remote socket 0, 1, and 2 to socket 3:

```
perf stat -a -e nvidia_cnvlink_pmu_3/event=0x0,rem_socket=0x7/
```

18.14.6 PCIE PMU

The PCIE PMU monitors all read/write traffic from PCIE root ports to local/remote memory. Please see *Traffic Coverage* for more info about the PMU traffic coverage.

The events and configuration options of this PMU device are described in sysfs, see /sys/bus/event_sources/devices/nvidia_pcie_pmu_<socket-id>.

Each SoC socket can support multiple root ports. The user can use "root_port" bitmap parameter to select the port(s) to monitor, i.e. "root_port=0xF" corresponds to root port 0 to 3. /sys/bus/event_sources/devices/nvidia_pcie_pmu_<socket-id>/format/root_port shows the valid bits that can be set in the "root_port" parameter.

Example usage:

- Count event id 0x0 from root port 0 and 1 of socket 0:

```
perf stat -a -e nvidia_pcie_pmu_0/event=0x0,root_port=0x3/
```

- Count event id 0x0 from root port 0 and 1 of socket 1:

```
perf stat -a -e nvidia_pcie_pmu_1/event=0x0,root_port=0x3/
```

18.14.7 Traffic Coverage

The PMU traffic coverage may vary dependent on the chip configuration:

- **NVIDIA Grace Hopper Superchip**: Hopper GPU is connected with Grace SoC.

  Example configuration with two Grace SoCs:

```
**********************************
 ↪**********************************
* SOCKET-A                       *          * SOCKET-B
 ↪*
*                                *          *
 ↪*
*                     ::::::::   *          *  ::::::::
 ↪*
*                     : PCIE :   *          *  : PCIE :
 ↪*
*                     ::::::::   *          *  ::::::::
 ↪*
*                         |      *          *      |
 ↪*
*                         |      *          *      |
 ↪*
*  :::::::            ::::::::: *          *  :::::::::            :::::::
 ↪*
*  :     :            :       : *          *  :       :            :     :
 ↪*
*  : GPU :<--NVLink-->: Grace :<---CNVLink--->: Grace :<--NVLink-->: GPU :
 ↪*
*  :     :    C2C     :  SoC  : *          *  :  SoC  :    C2C     :     :
 ↪*
*  :::::::            ::::::::: *          *  :::::::::            :::::::
 ↪*
*     |                   |      *          *      |                   |
 ↪*
*     |                   |      *          *      |                   |
 ↪*
*  &&&&&&&&           &&&&&&&&  *          *   &&&&&&&&           &&&&&&&&
 ↪*
*  & GMEM &           & CMEM &  *          *   & CMEM &           & GMEM &
 ↪*
*  &&&&&&&&           &&&&&&&&  *          *   &&&&&&&&           &&&&&&&&
 ↪*
*                                *          *
 ↪*
**********************************
 ↪**********************************

GMEM = GPU Memory (e.g. HBM)
CMEM = CPU Memory (e.g. LPDDR5X)
```

Following table contains traffic coverage of Grace SoC PMU in socket-A:

```
+--------------+-------+-----------+-----------+-----+----------+----------
↪+
|              |                       Source                              ␣
↪|
+              +-------+-----------+-----------+-----+----------+----------
↪+
| Destination  |       |GPU ATS    |GPU Not-ATS|     | Socket-B | Socket-B␣
↪|
|              |PCI R/W|Translated,|Translated | CPU | CPU/PCIE1| GPU/
↪PCIE2|
|              |       |EGM        |           |     |          |         ␣
↪|
+==============+=======+===========+===========+=====+==========+==========+
| Local        | PCIE  |NVLink-C2C0|NVLink-C2C1| SCF | SCF PMU  | CNVLink ␣
↪|
| SYSRAM/CMEM  | PMU   |PMU        |PMU        | PMU |          | PMU     ␣
↪|
+--------------+-------+-----------+-----------+-----+----------+----------
↪+
| Local GMEM   | PCIE  |    N/A    |NVLink-C2C1| SCF | SCF PMU  | CNVLink ␣
↪|
|              | PMU   |           |PMU        | PMU |          | PMU     ␣
↪|
+--------------+-------+-----------+-----------+-----+----------+----------
↪+
| Remote       | PCIE  |NVLink-C2C0|NVLink-C2C1| SCF |          |         ␣
↪|
| SYSRAM/CMEM  | PMU   |PMU        |PMU        | PMU |   N/A    |   N/A   ␣
↪|
| over CNVLink |       |           |           |     |          |         ␣
↪|
+--------------+-------+-----------+-----------+-----+----------+----------
↪+
| Remote GMEM  | PCIE  |NVLink-C2C0|NVLink-C2C1| SCF |          |         ␣
↪|
| over CNVLink | PMU   |PMU        |PMU        | PMU |   N/A    |   N/A   ␣
↪|
+--------------+-------+-----------+-----------+-----+----------+----------
↪+

PCIE1 traffic represents strongly ordered (SO) writes.
PCIE2 traffic represents reads and relaxed ordered (RO) writes.
```

- **NVIDIA Grace CPU Superchip**: two Grace CPU SoCs are connected.

  Example configuration with two Grace SoCs:

```
*******************             *******************
* SOCKET-A        *             * SOCKET-B        *
*                 *             *                 *
*    ::::::::     *             *    ::::::::     *
*    : PCIE :     *             *    : PCIE :     *
*    ::::::::     *             *    ::::::::     *
*        |        *             *        |        *
*        |        *             *        |        *
*    :::::::::    *             *    :::::::::    *
*    :       :    *             *    :       :    *
*    : Grace :<--------NVLink------->: Grace :    *
*    :  SoC  :    *     C2C     *    :  SoC  :    *
*    :::::::::    *             *    :::::::::    *
*        |        *             *        |        *
*        |        *             *        |        *
*     &&&&&&&&    *             *     &&&&&&&&    *
*     & CMEM &    *             *     & CMEM &    *
*     &&&&&&&&    *             *     &&&&&&&&    *
*                 *             *                 *
*******************             *******************

GMEM = GPU Memory (e.g. HBM)
CMEM = CPU Memory (e.g. LPDDR5X)
```

Following table contains traffic coverage of Grace SoC PMU in socket-A:

```
+-----------------+-----------+---------+----------+-------------+
|                 |                      Source                  |
+                 +-----------+---------+----------+-------------+
| Destination     |           |         | Socket-B | Socket-B    |
|                 |  PCI R/W  |   CPU   | CPU/PCIE1| PCIE2       |
|                 |           |         |          |             |
+=================+===========+=========+==========+=============+
| Local           |  PCIE PMU | SCF PMU | SCF PMU  | NVLink-C2C0 |
| SYSRAM/CMEM     |           |         |          | PMU         |
+-----------------+-----------+---------+----------+-------------+
| Remote          |           |         |          |             |
| SYSRAM/CMEM     |  PCIE PMU | SCF PMU |   N/A    |     N/A     |
| over NVLink-C2C |           |         |          |             |
+-----------------+-----------+---------+----------+-------------+

PCIE1 traffic represents strongly ordered (SO) writes.
PCIE2 traffic represents reads and relaxed ordered (RO) writes.
```

18.15 Amlogic SoC DDR Bandwidth Performance Monitoring Unit (PMU)

The Amlogic Meson G12 SoC contains a bandwidth monitor inside DRAM controller. The monitor includes 4 channels. Each channel can count the request accessing DRAM. The channel can count up to 3 AXI port simultaneously. It can be helpful to show if the performance bottleneck is on DDR bandwidth.

Currently, this driver supports the following 5 perf events:

- meson_ddr_bw/total_rw_bytes/
- meson_ddr_bw/chan_1_rw_bytes/
- meson_ddr_bw/chan_2_rw_bytes/
- meson_ddr_bw/chan_3_rw_bytes/
- meson_ddr_bw/chan_4_rw_bytes/

meson_ddr_bw/chan_{1,2,3,4}_rw_bytes/ events are channel-specific events. Each channel support filtering, which can let the channel to monitor individual IP module in SoC.

Below are DDR access request event filter keywords:

- arm - from CPU
- vpu_read1 - from OSD + VPP read
- gpu - from 3D GPU
- pcie - from PCIe controller
- hdcp - from HDCP controller
- hevc_front - from HEVC codec front end
- usb3_0 - from USB3.0 controller
- hevc_back - from HEVC codec back end
- h265enc - from HEVC encoder
- vpu_read2 - from DI read
- vpu_write1 - from VDIN write
- vpu_write2 - from di write
- vdec - from legacy codec video decoder
- hcodec - from H264 encoder
- ge2d - from ge2d
- spicc1 - from SPI controller 1
- usb0 - from USB2.0 controller 0
- dma - from system DMA controller 1
- arb0 - from arb0
- sd_emmc_b - from SD eMMC b controller

- usb1 - from USB2.0 controller 1
- audio - from Audio module
- sd_emmc_c - from SD eMMC c controller
- spicc2 - from SPI controller 2
- ethernet - from Ethernet controller

Examples:

- Show the total DDR bandwidth per seconds:

```
perf stat -a -e meson_ddr_bw/total_rw_bytes/ -I 1000 sleep 10
```

- Show individual DDR bandwidth from CPU and GPU respectively, as well as sum of them:

```
perf stat -a -e meson_ddr_bw/chan_1_rw_bytes,arm=1/ -I 1000 sleep 10
perf stat -a -e meson_ddr_bw/chan_2_rw_bytes,gpu=1/ -I 1000 sleep 10
perf stat -a -e meson_ddr_bw/chan_3_rw_bytes,arm=1,gpu=1/ -I 1000 sleep 10
```

18.16 CXL Performance Monitoring Unit (CPMU)

The CXL rev 3.0 specification provides a definition of CXL Performance Monitoring Unit in section 13.2: Performance Monitoring.

CXL components (e.g. Root Port, Switch Upstream Port, End Point) may have any number of CPMU instances. CPMU capabilities are fully discoverable from the devices. The specification provides event definitions for all CXL protocol message types and a set of additional events for things commonly counted on CXL devices (e.g. DRAM events).

18.16.1 CPMU driver

The CPMU driver registers a perf PMU with the name pmu_mem<X>.<Y> on the CXL bus representing the Yth CPMU for memX.

/sys/bus/cxl/device/pmu_mem<X>.<Y>

The associated PMU is registered as

/sys/bus/event_sources/devices/cxl_pmu_mem<X>.<Y>

In common with other CXL bus devices, the id has no specific meaning and the relationship to specific CXL device should be established via the device parent of the device on the CXL bus.

PMU driver provides description of available events and filter options in sysfs.

The "format" directory describes all formats of the config (event vendor id, group id and mask) config1 (threshold, filter enables) and config2 (filter parameters) fields of the perf_event_attr structure. The "events" directory describes all documented events show in perf list.

The events shown in perf list are the most fine grained events with a single bit of the event mask set. More general events may be enable by setting multiple mask bits in config. For example, all Device to Host Read Requests may be captured on a single counter by setting the bits for all of

- d2h_req_rdcurr
- d2h_req_rdown
- d2h_req_rdshared
- d2h_req_rdany
- d2h_req_rdownnodata

Example of usage:

```
$#perf list
cxl_pmu_mem0.0/clock_ticks/                        [Kernel PMU event]
cxl_pmu_mem0.0/d2h_req_rdshared/                   [Kernel PMU event]
cxl_pmu_mem0.0/h2d_req_snpcur/                     [Kernel PMU event]
cxl_pmu_mem0.0/h2d_req_snpdata/                    [Kernel PMU event]
cxl_pmu_mem0.0/h2d_req_snpinv/                     [Kernel PMU event]
------------------------------------------------------------

$# perf stat -a -e cxl_pmu_mem0.0/clock_ticks/ -e cxl_pmu_mem0.0/d2h_req_
↪rdshared/
```

Vendor specific events may also be available and if so can be used via

$# perf stat -a -e cxl_pmu_mem0.0/vid=VID,gid=GID,mask=MASK/

The driver does not support sampling so "perf record" is unsupported. It only supports system-wide counting so attaching to a task is unsupported.

18.17 Ampere SoC Performance Monitoring Unit (PMU)

Ampere SoC PMU is a generic PMU IP that follows Arm CoreSight PMU architecture. Therefore, the driver is implemented as a submodule of arm_cspmu driver. At the first phase it's used for counting MCU events on AmpereOne.

18.17.1 MCU PMU events

The PMU driver supports setting filters for "rank", "bank", and "threshold". Note, that the filters are per PMU instance rather than per event.

Example for perf tool use:

```
/ # perf list ampere

  ampere_mcu_pmu_0/act_sent/                         [Kernel PMU event]
  <...>
  ampere_mcu_pmu_1/rd_sent/                          [Kernel PMU event]
  <...>

/ # perf stat -a -e ampere_mcu_pmu_0/act_sent,bank=5,rank=3,threshold=2/,
↪ampere_mcu_pmu_1/rd_sent/ \
      sleep 1
```

CHAPTER

NINETEEN

PSTORE BLOCK OOPS/PANIC LOGGER

19.1 Introduction

pstore block (pstore/blk) is an oops/panic logger that writes its logs to a block device and non-block device before the system crashes. You can get these log files by mounting pstore filesystem like:

```
mount -t pstore pstore /sys/fs/pstore
```

19.2 pstore block concepts

pstore/blk provides efficient configuration method for pstore/blk, which divides all configurations into two parts, configurations for user and configurations for driver.

Configurations for user determine how pstore/blk works, such as pmsg_size, kmsg_size and so on. All of them support both Kconfig and module parameters, but module parameters have priority over Kconfig.

Configurations for driver are all about block device and non-block device, such as total_size of block device and read/write operations.

19.3 Configurations for user

All of these configurations support both Kconfig and module parameters, but module parameters have priority over Kconfig.

Here is an example for module parameters:

```
pstore_blk.blkdev=/dev/mmcblk0p7 pstore_blk.kmsg_size=64 best_effort=y
```

The detail of each configurations may be of interest to you.

19.3.1 blkdev

The block device to use. Most of the time, it is a partition of block device. It's required for pstore/blk. It is also used for MTD device.

When pstore/blk is built as a module, ”blkdev” accepts the following variants:

1. /dev/<disk_name> represents the device number of disk
2. /dev/<disk_name><decimal> represents the device number of partition - device number of disk plus the partition number
3. /dev/<disk_name>p<decimal> - same as the above; this form is used when disk name of partitioned disk ends with a digit.

When pstore/blk is built into the kernel, ”blkdev” accepts the following variants:

1. <hex_major><hex_minor> device number in hexadecimal representation, with no leading 0x, for example b302.
2. PARTUUID=00112233-4455-6677-8899-AABBCCDDEEFF represents the unique id of a partition if the partition table provides it. The UUID may be either an EFI/GPT UUID, or refer to an MSDOS partition using the format SSSSSSSS-PP, where SSSSSSSS is a zero-filled hex representation of the 32-bit ”NT disk signature”, and PP is a zero-filled hex representation of the 1-based partition number.
3. PARTUUID=<UUID>/PARTNROFF=<int> to select a partition in relation to a partition with a known unique id.
4. <major>:<minor> major and minor number of the device separated by a colon.

It accepts the following variants for MTD device:

1. <device name> MTD device name. ”pstore” is recommended.
2. <device number> MTD device number.

19.3.2 kmsg_size

The chunk size in KB for oops/panic front-end. It **MUST** be a multiple of 4. It's optional if you do not care about the oops/panic log.

There are multiple chunks for oops/panic front-end depending on the remaining space except other pstore front-ends.

pstore/blk will log to oops/panic chunks one by one, and always overwrite the oldest chunk if there is no more free chunk.

19.3.3 pmsg_size

The chunk size in KB for pmsg front-end. It **MUST** be a multiple of 4. It's optional if you do not care about the pmsg log.

Unlike oops/panic front-end, there is only one chunk for pmsg front-end.

Pmsg is a user space accessible pstore object. Writes to */dev/pmsg0* are appended to the chunk. On reboot the contents are available in */sys/fs/pstore/pmsg-pstore-blk-0*.

19.3.4 console_size

The chunk size in KB for console front-end. It **MUST** be a multiple of 4. It's optional if you do not care about the console log.

Similar to pmsg front-end, there is only one chunk for console front-end.

All log of console will be appended to the chunk. On reboot the contents are available in */sys/fs/pstore/console-pstore-blk-0*.

19.3.5 ftrace_size

The chunk size in KB for ftrace front-end. It **MUST** be a multiple of 4. It's optional if you do not care about the ftrace log.

Similar to oops front-end, there are multiple chunks for ftrace front-end depending on the count of cpu processors. Each chunk size is equal to ftrace_size / processors_count.

All log of ftrace will be appended to the chunk. On reboot the contents are combined and available in */sys/fs/pstore/ftrace-pstore-blk-0*.

Persistent function tracing might be useful for debugging software or hardware related hangs. Here is an example of usage:

```
# mount -t pstore pstore /sys/fs/pstore
# mount -t debugfs debugfs /sys/kernel/debug/
# echo 1 > /sys/kernel/debug/pstore/record_ftrace
# reboot -f
[...]
# mount -t pstore pstore /sys/fs/pstore
# tail /sys/fs/pstore/ftrace-pstore-blk-0
CPU:0 ts:5914676 c0063828  c0063b94  call_cpuidle <- cpu_startup_entry+0x1b8/
↪0x1e0
CPU:0 ts:5914678 c039ecdc  c006385c  cpuidle_enter_state <- call_cpuidle+0x44/
↪0x48
CPU:0 ts:5914680 c039e9a0  c039ecf0  cpuidle_enter_freeze <- cpuidle_enter_
↪state+0x304/0x314
CPU:0 ts:5914681 c0063870  c039ea30  sched_idle_set_state <- cpuidle_enter_
↪state+0x44/0x314
CPU:1 ts:5916720 c0160f59  c015ee04  kernfs_unmap_bin_file <- __kernfs_
↪remove+0x140/0x204
CPU:1 ts:5916721 c05ca625  c015ee0c  __mutex_lock_slowpath <- __kernfs_
↪remove+0x148/0x204
```

```
CPU:1 ts:5916723 c05c813d  c05ca630  yield_to <- __mutex_lock_slowpath+0x314/
↪0x358
CPU:1 ts:5916724 c05ca2d1  c05ca638  __ww_mutex_lock <- __mutex_lock_
↪slowpath+0x31c/0x358
```

19.3.6 max_reason

Limiting which kinds of kmsg dumps are stored can be controlled via the `max_reason` value, as defined in include/linux/kmsg_dump.h's `enum kmsg_dump_reason`. For example, to store both Oopses and Panics, `max_reason` should be set to 2 (KMSG_DUMP_OOPS), to store only Panics `max_reason` should be set to 1 (KMSG_DUMP_PANIC). Setting this to 0 (KMSG_DUMP_UNDEF), means the reason filtering will be controlled by the `printk.always_kmsg_dump` boot param: if unset, it'll be KMSG_DUMP_OOPS, otherwise KMSG_DUMP_MAX.

19.4 Configurations for driver

A device driver uses `register_pstore_device` with `struct pstore_device_info` to register to pstore/blk.

int **register_pstore_device**(struct *pstore_device_info* *dev)

register non-block device to pstore/blk

Parameters

struct pstore_device_info *dev
non-block device information

Return

- 0 - OK
- Others - something error.

void **unregister_pstore_device**(struct *pstore_device_info* *dev)

unregister non-block device from pstore/blk

Parameters

struct pstore_device_info *dev
non-block device information

19.5 Compression and header

Block device is large enough for uncompressed oops data. Actually we do not recommend data compression because pstore/blk will insert some information into the first line of oops/panic data. For example:

```
Panic: Total 16 times
```

It means that it's OOPS|Panic for the 16th time since the first booting. Sometimes the number of occurrences of oops|panic since the first booting is important to judge whether the system is stable.

The following line is inserted by pstore filesystem. For example:

```
Oops#2 Part1
```

It means that it's OOPS for the 2nd time on the last boot.

19.6 Reading the data

The dump data can be read from the pstore filesystem. The format for these files is `dmesg-pstore-blk-[N]` for oops/panic front-end, `pmsg-pstore-blk-0` for pmsg front-end and so on. The timestamp of the dump file records the trigger time. To delete a stored record from block device, simply unlink the respective pstore file.

19.7 Attentions in panic read/write APIs

If on panic, the kernel is not going to run for much longer, the tasks will not be scheduled and most kernel resources will be out of service. It looks like a single-threaded program running on a single-core computer.

The following points require special attention for panic read/write APIs:

1. Can **NOT** allocate any memory. If you need memory, just allocate while the block driver is initializing rather than waiting until the panic.
2. Must be polled, **NOT** interrupt driven. No task schedule any more. The block driver should delay to ensure the write succeeds, but NOT sleep.
3. Can **NOT** take any lock. There is no other task, nor any shared resource; you are safe to break all locks.
4. Just use CPU to transfer. Do not use DMA to transfer unless you are sure that DMA will not keep lock.
5. Control registers directly. Please control registers directly rather than use Linux kernel resources. Do I/O map while initializing rather than wait until a panic occurs.
6. Reset your block device and controller if necessary. If you are not sure of the state of your block device and controller when a panic occurs, you are safe to stop and reset them.

pstore/blk supports psblk_blkdev_info(), which is defined in *linux/pstore_blk.h*, to get information of using block device, such as the device number, sector count and start sector of the whole disk.

19.8 pstore block internals

For developer reference, here are all the important structures and APIs:

struct **psz_buffer**

header of zone to flush to storage

Definition:

```
struct psz_buffer {
#define PSZ_SIG (0x43474244) ;
    uint32_t sig;
    atomic_t datalen;
    atomic_t start;
    uint8_t data[];
};
```

Members

sig

signature to indicate header (PSZ_SIG xor PSZONE-type value)

datalen

length of data in **data**

start

offset into **data** where the beginning of the stored bytes begin

data

zone data.

struct **psz_kmsg_header**

kmsg dump-specific header to flush to storage

Definition:

```
struct psz_kmsg_header {
#define PSTORE_KMSG_HEADER_MAGIC 0x4dfc3ae5 ;
    uint32_t magic;
    struct timespec64 time;
    bool compressed;
    uint32_t counter;
    enum kmsg_dump_reason reason;
    uint8_t data[];
};
```

Members

magic

magic num for kmsg dump header

time

kmsg dump trigger time

compressed

whether conpressed

counter
kmsg dump counter

reason
the kmsg dump reason (e.g. oops, panic, etc)

data
pointer to log data

Description

This is a sub-header for a kmsg dump, trailing after *psz_buffer*.

struct **pstore_zone**
single stored buffer

Definition:

```
struct pstore_zone {
    loff_t off;
    const char *name;
    enum pstore_type_id type;
    struct psz_buffer *buffer;
    struct psz_buffer *oldbuf;
    size_t buffer_size;
    bool should_recover;
    atomic_t dirty;
};
```

Members

off
zone offset of storage

name
front-end name for this zone

type
front-end type for this zone

buffer
pointer to data buffer managed by this zone

oldbuf
pointer to old data buffer

buffer_size
bytes in **buffer->data**

should_recover
whether this zone should recover from storage

dirty
whether the data in **buffer** dirty

Description

zone structure in memory.

struct **psz_context**

all about running state of pstore/zone

Definition:

```
struct psz_context {
    struct pstore_zone **kpszs;
    struct pstore_zone *ppsz;
    struct pstore_zone *cpsz;
    struct pstore_zone **fpszs;
    unsigned int kmsg_max_cnt;
    unsigned int kmsg_read_cnt;
    unsigned int kmsg_write_cnt;
    unsigned int pmsg_read_cnt;
    unsigned int console_read_cnt;
    unsigned int ftrace_max_cnt;
    unsigned int ftrace_read_cnt;
    unsigned int oops_counter;
    unsigned int panic_counter;
    atomic_t recovered;
    atomic_t on_panic;
    struct mutex pstore_zone_info_lock;
    struct pstore_zone_info *pstore_zone_info;
    struct pstore_info pstore;
};
```

Members

kpszs
kmsg dump storage zones

ppsz
pmsg storage zone

cpsz
console storage zone

fpszs
ftrace storage zones

kmsg_max_cnt
max count of **kpszs**

kmsg_read_cnt
counter of total read kmsg dumps

kmsg_write_cnt
counter of total kmsg dump writes

pmsg_read_cnt
counter of total read pmsg zone

console_read_cnt
counter of total read console zone

ftrace_max_cnt
max count of **fpszs**

ftrace_read_cnt
counter of max read ftrace zone

oops_counter
counter of oops dumps

panic_counter
counter of panic dumps

recovered
whether finished recovering data from storage

on_panic
whether panic is happening

pstore_zone_info_lock
lock to **pstore_zone_info**

pstore_zone_info
information from backend

pstore
structure for pstore

enum **psz_flush_mode**
flush mode for psz_zone_write()

Constants

FLUSH_NONE
do not flush to storage but update data on memory

FLUSH_PART
just flush part of data including meta data to storage

FLUSH_META
just flush meta data of zone to storage

FLUSH_ALL
flush all of zone

int **psz_recovery**(struct *psz_context* *cxt)
recover data from storage

Parameters

struct psz_context *cxt
the context of pstore/zone

Description

recovery means reading data back from storage after rebooting

Return

0 on success, others on failure.

struct **pstore_zone_info**
pstore/zone back-end driver structure

Definition:

```
struct pstore_zone_info {
    struct module *owner;
    const char *name;
    unsigned long total_size;
    unsigned long kmsg_size;
    int max_reason;
    unsigned long pmsg_size;
    unsigned long console_size;
    unsigned long ftrace_size;
    pstore_zone_read_op read;
    pstore_zone_write_op write;
    pstore_zone_erase_op erase;
    pstore_zone_write_op panic_write;
};
```

Members

owner
Module which is responsible for this back-end driver.

name
Name of the back-end driver.

total_size
The total size in bytes pstore/zone can use. It must be greater than 4096 and be multiple of 4096.

kmsg_size
The size of oops/panic zone. Zero means disabled, otherwise, it must be multiple of SECTOR_SIZE(512 Bytes).

max_reason
Maximum kmsg dump reason to store.

pmsg_size
The size of pmsg zone which is the same as **kmsg_size**.

console_size
The size of console zone which is the same as **kmsg_size**.

ftrace_size
The size of ftrace zone which is the same as **kmsg_size**.

read
The general read operation. Both of the function parameters **size** and **offset** are relative value to storage. On success, the number of bytes should be returned, others mean error.

write
The same as **read**, but the following error number: -EBUSY means try to write again later. -ENOMSG means to try next zone.

erase
The general erase operation for device with special removing job. Both of the function parameters **size** and **offset** are relative value to storage. Return 0 on success and others on failure.

panic_write
The write operation only used for panic case. It's optional if you do not care panic log. The parameters are relative value to storage. On success, the number of bytes should be returned, others excluding -ENOMSG mean error. -ENOMSG means to try next zone.

struct **pstore_device_info**
back-end pstore/blk driver structure.

Definition:

```
struct pstore_device_info {
    unsigned int flags;
    struct pstore_zone_info zone;
};
```

Members

flags
Refer to macro starting with PSTORE_FLAGS defined in linux/pstore.h. It means what front-ends this device support. Zero means all backends for compatible.

zone
The *struct pstore_zone_info* details.

struct **pstore_blk_config**
the pstore_blk backend configuration

Definition:

```
struct pstore_blk_config {
    char device[80];
    enum kmsg_dump_reason max_reason;
    unsigned long kmsg_size;
    unsigned long pmsg_size;
    unsigned long console_size;
    unsigned long ftrace_size;
};
```

Members

device
Name of the desired block device

max_reason
Maximum kmsg dump reason to store to block device

kmsg_size
Total size of for kmsg dumps

pmsg_size
Total size of the pmsg storage area

console_size
Total size of the console storage area

ftrace_size
Total size for ftrace logging data (for all CPUs)

int **pstore_blk_get_config**(struct *pstore_blk_config* *info)

get a copy of the pstore_blk backend configuration

Parameters

struct pstore_blk_config *info

The sturct pstore_blk_config to be filled in

Description

Failure returns negative error code, and success returns 0.

This is the beginning of a section with information of interest to application developers. Documents covering various aspects of the kernel ABI will be found here.

CHAPTER

TWENTY

RULES ON HOW TO ACCESS INFORMATION IN SYSFS

The kernel-exported sysfs exports internal kernel implementation details and depends on internal kernel structures and layout. It is agreed upon by the kernel developers that the Linux kernel does not provide a stable internal API. Therefore, there are aspects of the sysfs interface that may not be stable across kernel releases.

To minimize the risk of breaking users of sysfs, which are in most cases low-level userspace applications, with a new kernel release, the users of sysfs must follow some rules to use an as-abstract-as-possible way to access this filesystem. The current udev and HAL programs already implement this and users are encouraged to plug, if possible, into the abstractions these programs provide instead of accessing sysfs directly.

But if you really do want or need to access sysfs directly, please follow the following rules and then your programs should work with future versions of the sysfs interface.

- **Do not use libsysfs**
  It makes assumptions about sysfs which are not true. Its API does not offer any abstraction, it exposes all the kernel driver-core implementation details in its own API. Therefore it is not better than reading directories and opening the files yourself. Also, it is not actively maintained, in the sense of reflecting the current kernel development. The goal of providing a stable interface to sysfs has failed; it causes more problems than it solves. It violates many of the rules in this document.

- **sysfs is always at `/sys`**
  Parsing `/proc/mounts` is a waste of time. Other mount points are a system configuration bug you should not try to solve. For test cases, possibly support a `SYSFS_PATH` environment variable to overwrite the application's behavior, but never try to search for sysfs. Never try to mount it, if you are not an early boot script.

- **devices are only "devices"**
  There is no such thing like class-, bus-, physical devices, interfaces, and such that you can rely on in userspace. Everything is just simply a "device". Class-, bus-, physical, ... types are just kernel implementation details which should not be expected by applications that look for devices in sysfs.

  The properties of a device are:

  - devpath (`/devices/pci0000:00/0000:00:1d.1/usb2/2-2/2-2:1.0`)
    - identical to the DEVPATH value in the event sent from the kernel at device creation and removal
    - the unique key to the device at that point in time

    * the kernel's path to the device directory without the leading `/sys`, and always starting with a slash
    * all elements of a devpath must be real directories. Symlinks pointing to /sys/devices must always be resolved to their real target and the target path must be used to access the device. That way the devpath to the device matches the devpath of the kernel used at event time.
    * using or exposing symlink values as elements in a devpath string is a bug in the application
  - kernel name (`sda`, `tty`, `0000:00:1f.2`, ...)
    * a directory name, identical to the last element of the devpath
    * applications need to handle spaces and characters like `!` in the name
  - subsystem (`block`, `tty`, `pci`, ...)
    * simple string, never a path or a link
    * retrieved by reading the "subsystem"-link and using only the last element of the target path
  - driver (`tg3`, `ata_piix`, `uhci_hcd`)
    * a simple string, which may contain spaces, never a path or a link
    * it is retrieved by reading the "driver"-link and using only the last element of the target path
    * devices which do not have "driver"-link just do not have a driver; copying the driver value in a child device context is a bug in the application
  - attributes
    * the files in the device directory or files below subdirectories of the same device directory
    * accessing attributes reached by a symlink pointing to another device, like the "device"-link, is a bug in the application

  Everything else is just a kernel driver-core implementation detail that should not be assumed to be stable across kernel releases.

- **Properties of parent devices never belong into a child device.**
  Always look at the parent devices themselves for determining device context properties. If the device `eth0` or `sda` does not have a "driver"-link, then this device does not have a driver. Its value is empty. Never copy any property of the parent-device into a child-device. Parent device properties may change dynamically without any notice to the child device.

- **Hierarchy in a single device tree**
  There is only one valid place in sysfs where hierarchy can be examined and this is below: `/sys/devices.` It is planned that all device directories will end up in the tree below this directory.

- **Classification by subsystem**
  There are currently three places for classification of devices: `/sys/block, /sys/class` and `/sys/bus.` It is planned that these will not contain any device directories themselves, but only flat lists of symlinks pointing to the unified `/sys/devices` tree. All

three places have completely different rules on how to access device information. It is planned to merge all three classification directories into one place at `/sys/subsystem`, following the layout of the bus directories. All buses and classes, including the converted block subsystem, will show up there. The devices belonging to a subsystem will create a symlink in the "devices" directory at `/sys/subsystem/<name>/devices`,

If `/sys/subsystem` exists, `/sys/bus`, `/sys/class` and `/sys/block` can be ignored. If it does not exist, you always have to scan all three places, as the kernel is free to move a subsystem from one place to the other, as long as the devices are still reachable by the same subsystem name.

Assuming `/sys/class/<subsystem>` and `/sys/bus/<subsystem>`, or `/sys/block` and `/sys/class/block` are not interchangeable is a bug in the application.

- **Block**
  The converted block subsystem at `/sys/class/block` or `/sys/subsystem/block` will contain the links for disks and partitions at the same level, never in a hierarchy. Assuming the block subsystem to contain only disks and not partition devices in the same flat list is a bug in the application.

- **"device"-link and <subsystem>:<kernel name>-links**
  Never depend on the "device"-link. The "device"-link is a workaround for the old layout, where class devices are not created in `/sys/devices/` like the bus devices. If the link-resolving of a device directory does not end in `/sys/devices/`, you can use the "device"-link to find the parent devices in `/sys/devices/`, That is the single valid use of the "device"-link; it must never appear in any path as an element. Assuming the existence of the "device"-link for a device in `/sys/devices/` is a bug in the application. Accessing `/sys/class/net/eth0/device` is a bug in the application.

  Never depend on the class-specific links back to the `/sys/class` directory. These links are also a workaround for the design mistake that class devices are not created in `/sys/devices.` If a device directory does not contain directories for child devices, these links may be used to find the child devices in `/sys/class.` That is the single valid use of these links; they must never appear in any path as an element. Assuming the existence of these links for devices which are real child device directories in the `/sys/devices` tree is a bug in the application.

  It is planned to remove all these links when all class device directories live in `/sys/devices.`

- **Position of devices along device chain can change.**
  Never depend on a specific parent device position in the devpath, or the chain of parent devices. The kernel is free to insert devices into the chain. You must always request the parent device you are looking for by its subsystem value. You need to walk up the chain until you find the device that matches the expected subsystem. Depending on a specific position of a parent device or exposing relative paths using `../` to access the chain of parents is a bug in the application.

- **When reading and writing sysfs device attribute files, avoid dependency**
  on specific error codes wherever possible. This minimizes coupling to the error handling implementation within the kernel.

  In general, failures to read or write sysfs device attributes shall propagate errors wherever possible. Common errors include, but are not limited to:

  `-EIO`: The read or store operation is not supported, typically returned by the

sysfs system itself if the read or store pointer is `NULL`.

`-ENXIO`: The read or store operation failed

Error codes will not be changed without good reason, and should a change to error codes result in user-space breakage, it will be fixed, or the the offending change will be reverted.

Userspace applications can, however, expect the format and contents of the attribute files to remain consistent in the absence of a version attribute change in the context of a given attribute.

This is the beginning of a section with information of interest to application developers and system integrators doing analysis of the Linux kernel for safety critical applications. Documents supporting analysis of kernel interactions with applications, and key kernel subsystems expectations will be found here.

CHAPTER

TWENTYONE

DISCOVERING LINUX KERNEL SUBSYSTEMS USED BY A WORKLOAD

Authors

- Shuah Khan <skhan@linuxfoundation.org>
- Shefali Sharma <sshefali021@gmail.com>

maintained-by
Shuah Khan <skhan@linuxfoundation.org>

21.1 Key Points

- Understanding system resources necessary to build and run a workload is important.
- Linux tracing and strace can be used to discover the system resources in use by a workload. The completeness of the system usage information depends on the completeness of coverage of a workload.
- Performance and security of the operating system can be analyzed with the help of tools such as: perf, stress-ng, paxtest.
- Once we discover and understand the workload needs, we can focus on them to avoid regressions and use it to evaluate safety considerations.

21.2 Methodology

strace is a diagnostic, instructional, and debugging tool and can be used to discover the system resources in use by a workload. Once we discover and understand the workload needs, we can focus on them to avoid regressions and use it to evaluate safety considerations. We use strace tool to trace workloads.

This method of tracing using strace tells us the system calls invoked by the workload and doesn't include all the system calls that can be invoked by it. In addition, this tracing method tells us just the code paths within these system calls that are invoked. As an example, if a workload opens a file and reads from it successfully, then the success path is the one that is traced. Any error paths in that system call will not be traced. If there is a workload that provides full coverage of a workload then the method outlined here will trace and find all possible code paths. The completeness of the system usage information depends on the completeness of coverage of a workload.

The goal is tracing a workload on a system running a default kernel without requiring custom kernel installs.

21.3 How do we gather fine-grained system information?

strace tool can be used to trace system calls made by a process and signals it receives. System calls are the fundamental interface between an application and the operating system kernel. They enable a program to request services from the kernel. For instance, the open() system call in Linux is used to provide access to a file in the file system. strace enables us to track all the system calls made by an application. It lists all the system calls made by a process and their resulting output.

You can generate profiling data combining strace and perf record tools to record the events and information associated with a process. This provides insight into the process. ”perf annotate” tool generates the statistics of each instruction of the program. This document goes over the details of how to gather fine-grained information on a workload's usage of system resources.

We used strace to trace the perf, stress-ng, paxtest workloads to illustrate our methodology to discover resources used by a workload. This process can be applied to trace other workloads.

21.4 Getting the system ready for tracing

Before we can get started we will show you how to get your system ready. We assume that you have a Linux distribution running on a physical system or a virtual machine. Most distributions will include strace command. Let’s install other tools that aren’t usually included to build Linux kernel. Please note that the following works on Debian based distributions. You might have to find equivalent packages on other Linux distributions.

Install tools to build Linux kernel and tools in kernel repository. scripts/ver_linux is a good way to check if your system already has the necessary tools:

```
sudo apt-get build-essentials flex bison yacc
sudo apt install libelf-dev systemtap-sdt-dev libaudit-dev libslang2-dev␣
↪libperl-dev libdw-dev
```

cscope is a good tool to browse kernel sources. Let's install it now:

```
sudo apt-get install cscope
```

Install stress-ng and paxtest:

```
apt-get install stress-ng
apt-get install paxtest
```

21.5 Workload overview

As mentioned earlier, we used strace to trace perf bench, stress-ng and paxtest workloads to show how to analyze a workload and identify Linux subsystems used by these workloads. Let's start with an overview of these three workloads to get a better understanding of what they do and how to use them.

21.5.1 perf bench (all) workload

The perf bench command contains multiple multi-threaded microkernel benchmarks for executing different subsystems in the Linux kernel and system calls. This allows us to easily measure the impact of changes, which can help mitigate performance regressions. It also acts as a common benchmarking framework, enabling developers to easily create test cases, integrate transparently, and use performance-rich tooling subsystems.

21.5.2 Stress-ng netdev stressor workload

stress-ng is used for performing stress testing on the kernel. It allows you to exercise various physical subsystems of the computer, as well as interfaces of the OS kernel, using ”stressor-s”. They are available for CPU, CPU cache, devices, I/O, interrupts, file system, memory, network, operating system, pipelines, schedulers, and virtual machines. Please refer to the stress-ng man-page to find the description of all the available stressor-s. The netdev stressor starts specified number (N) of workers that exercise various netdevice ioctl commands across all the available network devices.

21.5.3 paxtest kiddie workload

paxtest is a program that tests buffer overflows in the kernel. It tests kernel enforcements over memory usage. Generally, execution in some memory segments makes buffer overflows possible. It runs a set of programs that attempt to subvert memory usage. It is used as a regression test suite for PaX, but might be useful to test other memory protection patches for the kernel. We used paxtest kiddie mode which looks for simple vulnerabilities.

21.6 What is strace and how do we use it?

As mentioned earlier, strace which is a useful diagnostic, instructional, and debugging tool and can be used to discover the system resources in use by a workload. It can be used:

- To see how a process interacts with the kernel.
- To see why a process is failing or hanging.
- For reverse engineering a process.
- To find the files on which a program depends.
- For analyzing the performance of an application.
- For troubleshooting various problems related to the operating system.

In addition, strace can generate run-time statistics on times, calls, and errors for each system call and report a summary when program exits, suppressing the regular output. This attempts to show system time (CPU time spent running in the kernel) independent of wall clock time. We plan to use these features to get information on workload system usage.

strace command supports basic, verbose, and stats modes. strace command when run in verbose mode gives more detailed information about the system calls invoked by a process.

Running strace -c generates a report of the percentage of time spent in each system call, the total time in seconds, the microseconds per call, the total number of calls, the count of each system call that has failed with an error and the type of system call made.

- Usage: strace <command we want to trace>
- Verbose mode usage: strace -v <command>
- Gather statistics: strace -c <command>

We used the "-c" option to gather fine-grained run-time statistics in use by three workloads we have chose for this analysis.

- perf
- stress-ng
- paxtest

21.7 What is cscope and how do we use it?

Now let's look at cscope, a command line tool for browsing C, C++ or Java code-bases. We can use it to find all the references to a symbol, global definitions, functions called by a function, functions calling a function, text strings, regular expression patterns, files including a file.

We can use cscope to find which system call belongs to which subsystem. This way we can find the kernel subsystems used by a process when it is executed.

Let's checkout the latest Linux repository and build cscope database:

```
git clone git://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git␣
↪linux
cd linux
cscope -R -p10  # builds cscope.out database before starting browse session
cscope -d -p10  # starts browse session on cscope.out database
```

Note: Run "cscope -R -p10" to build the database and c"scope -d -p10" to enter into the browsing session. cscope by default cscope.out database. To get out of this mode press ctrl+d. -p option is used to specify the number of file path components to display. -p10 is optimal for browsing kernel sources.

21.8 What is perf and how do we use it?

Perf is an analysis tool based on Linux 2.6+ systems, which abstracts the CPU hardware difference in performance measurement in Linux, and provides a simple command line interface. Perf is based on the perf_events interface exported by the kernel. It is very useful for profiling the system and finding performance bottlenecks in an application.

If you haven't already checked out the Linux mainline repository, you can do so and then build kernel and perf tool:

```
git clone git://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git␣
↪linux
cd linux
make -j3 all
cd tools/perf
make
```

Note: The perf command can be built without building the kernel in the repository and can be run on older kernels. However matching the kernel and perf revisions gives more accurate information on the subsystem usage.

We used ”perf stat” and ”perf bench” options. For a detailed information on the perf tool, run ”perf -h”.

21.8.1 perf stat

The perf stat command generates a report of various hardware and software events. It does so with the help of hardware counter registers found in modern CPUs that keep the count of these activities. ”perf stat cal” shows stats for cal command.

21.8.2 Perf bench

The perf bench command contains multiple multi-threaded microkernel benchmarks for executing different subsystems in the Linux kernel and system calls. This allows us to easily measure the impact of changes, which can help mitigate performance regressions. It also acts as a common benchmarking framework, enabling developers to easily create test cases, integrate transparently, and use performance-rich tooling.

”perf bench all” command runs the following benchmarks:

- sched/messaging
- sched/pipe
- syscall/basic
- mem/memcpy
- mem/memset

21.9 What is stress-ng and how do we use it?

As mentioned earlier, stress-ng is used for performing stress testing on the kernel. It allows you to exercise various physical subsystems of the computer, as well as interfaces of the OS kernel, using stressor-s. They are available for CPU, CPU cache, devices, I/O, interrupts, file system, memory, network, operating system, pipelines, schedulers, and virtual machines.

The netdev stressor starts N workers that exercise various netdevice ioctl commands across all the available network devices. The following ioctls are exercised:

- SIOCGIFCONF, SIOCGIFINDEX, SIOCGIFNAME, SIOCGIFFLAGS
- SIOCGIFADDR, SIOCGIFNETMASK, SIOCGIFMETRIC, SIOCGIFMTU
- SIOCGIFHWADDR, SIOCGIFMAP, SIOCGIFTXQLEN

The following command runs the stressor:

```
stress-ng --netdev 1 -t 60 --metrics command.
```

We can use the perf record command to record the events and information associated with a process. This command records the profiling data in the perf.data file in the same directory.

Using the following commands you can record the events associated with the netdev stressor, view the generated report perf.data and annotate the to view the statistics of each instruction of the program:

```
perf record stress-ng --netdev 1 -t 60 --metrics command.
perf report
perf annotate
```

21.10 What is paxtest and how do we use it?

paxtest is a program that tests buffer overflows in the kernel. It tests kernel enforcements over memory usage. Generally, execution in some memory segments makes buffer overflows possible. It runs a set of programs that attempt to subvert memory usage. It is used as a regression test suite for PaX, and will be useful to test other memory protection patches for the kernel.

paxtest provides kiddie and blackhat modes. The paxtest kiddie mode runs in normal mode, whereas the blackhat mode tries to get around the protection of the kernel testing for vulnerabilities. We focus on the kiddie mode here and combine ”paxtest kiddie” run with ”perf record” to collect CPU stack traces for the paxtest kiddie run to see which function is calling other functions in the performance profile. Then the ”dwarf” (DWARF's Call Frame Information) mode can be used to unwind the stack.

The following command can be used to view resulting report in call-graph format:

```
perf record --call-graph dwarf paxtest kiddie
perf report --stdio
```

21.11 Tracing workloads

Now that we understand the workloads, let's start tracing them.

21.11.1 Tracing perf bench all workload

Run the following command to trace perf bench all workload:

```
strace -c perf bench all
```

System Calls made by the workload

The below table shows the system calls invoked by the workload, number of times each system call is invoked, and the corresponding Linux subsystem.

System Call	# calls	Linux Subsystem	System Call (API)
getppid	10000001	Process Mgmt	sys_getpid()
clone	1077	Process Mgmt.	sys_clone()
prctl	23	Process Mgmt.	sys_prctl()
prlimit64	7	Process Mgmt.	sys_prlimit64()
getpid	10	Process Mgmt.	sys_getpid()
uname	3	Process Mgmt.	sys_uname()
sysinfo	1	Process Mgmt.	sys_sysinfo()
getuid	1	Process Mgmt.	sys_getuid()
getgid	1	Process Mgmt.	sys_getgid()
geteuid	1	Process Mgmt.	sys_geteuid()
getegid	1	Process Mgmt.	sys_getegid
close	49951	Filesystem	sys_close()
pipe	604	Filesystem	sys_pipe()
openat	48560	Filesystem	sys_opennat()
fstat	8338	Filesystem	sys_fstat()
stat	1573	Filesystem	sys_stat()
pread64	9646	Filesystem	sys_pread64()
getdents64	1873	Filesystem	sys_getdents64()
access	3	Filesystem	sys_access()
lstat	1880	Filesystem	sys_lstat()
lseek	6	Filesystem	sys_lseek()
ioctl	3	Filesystem	sys_ioctl()
dup2	1	Filesystem	sys_dup2()
execve	2	Filesystem	sys_execve()
fcntl	8779	Filesystem	sys_fcntl()
statfs	1	Filesystem	sys_statfs()
epoll_create	2	Filesystem	sys_epoll_create()
epoll_ctl	64	Filesystem	sys_epoll_ctl()
newfstatat	8318	Filesystem	sys_newfstatat()
eventfd2	192	Filesystem	sys_eventfd2()
mmap	243	Memory Mgmt.	sys_mmap()
mprotect	32	Memory Mgmt.	sys_mprotect()

continues on next page

Table 1 – continued from previous page

System Call	# calls	Linux Subsystem	System Call (API)
brk	21	Memory Mgmt.	sys_brk()
munmap	128	Memory Mgmt.	sys_munmap()
set_mempolicy	156	Memory Mgmt.	sys_set_mempolicy()
set_tid_address	1	Process Mgmt.	sys_set_tid_address()
set_robust_list	1	Futex	sys_set_robust_list()
futex	341	Futex	sys_futex()
sched_getaffinity	79	Scheduler	sys_sched_getaffinity()
sched_setaffinity	223	Scheduler	sys_sched_setaffinity()
socketpair	202	Network	sys_socketpair()
rt_sigprocmask	21	Signal	sys_rt_sigprocmask()
rt_sigaction	36	Signal	sys_rt_sigaction()
rt_sigreturn	2	Signal	sys_rt_sigreturn()
wait4	889	Time	sys_wait4()
clock_nanosleep	37	Time	sys_clock_nanosleep()
capget	4	Capability	sys_capget()

21.11.2 Tracing stress-ng netdev stressor workload

Run the following command to trace stress-ng netdev stressor workload:

```
strace -c  stress-ng --netdev 1 -t 60 --metrics
```

System Calls made by the workload

The below table shows the system calls invoked by the workload, number of times each system call is invoked, and the corresponding Linux subsystem.

System Call	# calls	Linux Subsystem	System Call (API)
openat	74	Filesystem	sys_openat()
close	75	Filesystem	sys_close()
read	58	Filesystem	sys_read()
fstat	20	Filesystem	sys_fstat()
flock	10	Filesystem	sys_flock()
write	7	Filesystem	sys_write()
getdents64	8	Filesystem	sys_getdents64()
pread64	8	Filesystem	sys_pread64()
lseek	1	Filesystem	sys_lseek()
access	2	Filesystem	sys_access()
getcwd	1	Filesystem	sys_getcwd()
execve	1	Filesystem	sys_execve()
mmap	61	Memory Mgmt.	sys_mmap()
munmap	3	Memory Mgmt.	sys_munmap()
mprotect	20	Memory Mgmt.	sys_mprotect()
mlock	2	Memory Mgmt.	sys_mlock()
brk	3	Memory Mgmt.	sys_brk()
rt_sigaction	21	Signal	sys_rt_sigaction()
rt_sigprocmask	1	Signal	sys_rt_sigprocmask()

continues on next page

Table 2 – continued from previous page

System Call	# calls	Linux Subsystem	System Call (API)
sigaltstack	1	Signal	sys_sigaltstack()
rt_sigreturn	1	Signal	sys_rt_sigreturn()
getpid	8	Process Mgmt.	sys_getpid()
prlimit64	5	Process Mgmt.	sys_prlimit64()
arch_prctl	2	Process Mgmt.	sys_arch_prctl()
sysinfo	2	Process Mgmt.	sys_sysinfo()
getuid	2	Process Mgmt.	sys_getuid()
uname	1	Process Mgmt.	sys_uname()
setpgid	1	Process Mgmt.	sys_setpgid()
getrusage	1	Process Mgmt.	sys_getrusage()
geteuid	1	Process Mgmt.	sys_geteuid()
getppid	1	Process Mgmt.	sys_getppid()
sendto	3	Network	sys_sendto()
connect	1	Network	sys_connect()
socket	1	Network	sys_socket()
clone	1	Process Mgmt.	sys_clone()
set_tid_address	1	Process Mgmt.	sys_set_tid_address()
wait4	2	Time	sys_wait4()
alarm	1	Time	sys_alarm()
set_robust_list	1	Futex	sys_set_robust_list()

21.11.3 Tracing paxtest kiddie workload

Run the following command to trace paxtest kiddie workload:

```
strace -c paxtest kiddie
```

System Calls made by the workload

The below table shows the system calls invoked by the workload, number of times each system call is invoked, and the corresponding Linux subsystem.

System Call	# calls	Linux Subsystem	System Call (API)
read	3	Filesystem	sys_read()
write	11	Filesystem	sys_write()
close	41	Filesystem	sys_close()
stat	24	Filesystem	sys_stat()
fstat	2	Filesystem	sys_fstat()
pread64	6	Filesystem	sys_pread64()
access	1	Filesystem	sys_access()
pipe	1	Filesystem	sys_pipe()
dup2	24	Filesystem	sys_dup2()
execve	1	Filesystem	sys_execve()
fcntl	26	Filesystem	sys_fcntl()
openat	14	Filesystem	sys_openat()
rt_sigaction	7	Signal	sys_rt_sigaction()
rt_sigreturn	38	Signal	sys_rt_sigreturn()
clone	38	Process Mgmt.	sys_clone()
wait4	44	Time	sys_wait4()
mmap	7	Memory Mgmt.	sys_mmap()
mprotect	3	Memory Mgmt.	sys_mprotect()
munmap	1	Memory Mgmt.	sys_munmap()
brk	3	Memory Mgmt.	sys_brk()
getpid	1	Process Mgmt.	sys_getpid()
getuid	1	Process Mgmt.	sys_getuid()
getgid	1	Process Mgmt.	sys_getgid()
geteuid	2	Process Mgmt.	sys_geteuid()
getegid	1	Process Mgmt.	sys_getegid()
getppid	1	Process Mgmt.	sys_getppid()
arch_prctl	2	Process Mgmt.	sys_arch_prctl()

21.12 Conclusion

This document is intended to be used as a guide on how to gather fine-grained information on the resources in use by workloads using strace.

21.13 References

- Discovery Linux Kernel Subsystems used by OpenAPS
- ELISA-White-Papers-Discovering Linux kernel subsystems used by a workload
- strace
- perf
- paxtest README
- stress-ng
- Monitoring and managing system status and performance

The rest of this manual consists of various unordered guides on how to configure specific aspects of kernel behavior to your liking.

CHAPTER

TWENTYTWO

ACPI SUPPORT

Here we document in detail how to interact with various mechanisms in the Linux ACPI support.

22.1 Upgrading ACPI tables via initrd

22.1.1 What is this about

If the ACPI_TABLE_UPGRADE compile option is true, it is possible to upgrade the ACPI execution environment that is defined by the ACPI tables via upgrading the ACPI tables provided by the BIOS with an instrumented, modified, more recent version one, or installing brand new ACPI tables.

When building initrd with kernel in a single image, option ACPI_TABLE_OVERRIDE_VIA_BUILTIN_INITRD should also be true for this feature to work.

For a full list of ACPI tables that can be upgraded/installed, take a look at the char **table_sigs[MAX_ACPI_SIGNATURE];* definition in drivers/acpi/tables.c.

All ACPI tables iasl (Intel's ACPI compiler and disassembler) knows should be overridable, except:

- ACPI_SIG_RSDP (has a signature of 6 bytes)
- ACPI_SIG_FACS (does not have an ordinary ACPI table header)

Both could get implemented as well.

22.1.2 What is this for

Complain to your platform/BIOS vendor if you find a bug which is so severe that a workaround is not accepted in the Linux kernel. And this facility allows you to upgrade the buggy tables before your platform/BIOS vendor releases an upgraded BIOS binary.

This facility can be used by platform/BIOS vendors to provide a Linux compatible environment without modifying the underlying platform firmware.

This facility also provides a powerful feature to easily debug and test ACPI BIOS table compatibility with the Linux kernel by modifying old platform provided ACPI tables or inserting new ACPI tables.

It can and should be enabled in any kernel because there is no functional change with not instrumented initrds.

22.1.3 How does it work

```
# Extract the machine's ACPI tables:
cd /tmp
acpidump >acpidump
acpixtract -a acpidump
# Disassemble, modify and recompile them:
iasl -d *.dat
# For example add this statement into a _PRT (PCI Routing Table) function
# of the DSDT:
Store("HELLO WORLD", debug)
# And increase the OEM Revision. For example, before modification:
DefinitionBlock ("DSDT.aml", "DSDT", 2, "INTEL ", "TEMPLATE", 0x00000000)
# After modification:
DefinitionBlock ("DSDT.aml", "DSDT", 2, "INTEL ", "TEMPLATE", 0x00000001)
iasl -sa dsdt.dsl
# Add the raw ACPI tables to an uncompressed cpio archive.
# They must be put into a /kernel/firmware/acpi directory inside the cpio
# archive. Note that if the table put here matches a platform table
# (similar Table Signature, and similar OEMID, and similar OEM Table ID)
# with a more recent OEM Revision, the platform table will be upgraded by
# this table. If the table put here doesn't match a platform table
# (dissimilar Table Signature, or dissimilar OEMID, or dissimilar OEM Table
# ID), this table will be appended.
mkdir -p kernel/firmware/acpi
cp dsdt.aml kernel/firmware/acpi
# A maximum of "NR_ACPI_INITRD_TABLES (64)" tables are currently allowed
# (see osl.c):
iasl -sa facp.dsl
iasl -sa ssdt1.dsl
cp facp.aml kernel/firmware/acpi
cp ssdt1.aml kernel/firmware/acpi
# The uncompressed cpio archive must be the first. Other, typically
# compressed cpio archives, must be concatenated on top of the uncompressed
# one. Following command creates the uncompressed cpio archive and
# concatenates the original initrd on top:
find kernel | cpio -H newc --create > /boot/instrumented_initrd
cat /boot/initrd >>/boot/instrumented_initrd
# reboot with increased acpi debug level, e.g. boot params:
acpi.debug_level=0x2 acpi.debug_layer=0xFFFFFFFF
# and check your syslog:
[    1.268089] ACPI: PCI Interrupt Routing Table [\_SB_.PCI0._PRT]
[    1.272091] [ACPI Debug]  String [0x0B] "HELLO WORLD"
```

iasl is able to disassemble and recompile quite a lot different, also static ACPI tables.

22.1.4 Where to retrieve userspace tools

iasl and acpixtract are part of Intel's ACPICA project: https://acpica.org/

and should be packaged by distributions (for example in the acpica package on SUSE).

acpidump can be found in Len Browns pmtools: ftp://kernel.org/pub/linux/kernel/people/lenb/acpi/utils/pmtools/acpidump

This tool is also part of the acpica package on SUSE. Alternatively, used ACPI tables can be retrieved via sysfs in latest kernels: /sys/firmware/acpi/tables

22.2 SSDT Overlays

In order to support ACPI open-ended hardware configurations (e.g. development boards) we need a way to augment the ACPI configuration provided by the firmware image. A common example is connecting sensors on I2C / SPI buses on development boards.

Although this can be accomplished by creating a kernel platform driver or recompiling the firmware image with updated ACPI tables, neither is practical: the former proliferates board specific kernel code while the latter requires access to firmware tools which are often not publicly available.

Because ACPI supports external references in AML code a more practical way to augment firmware ACPI configuration is by dynamically loading user defined SSDT tables that contain the board specific information.

For example, to enumerate a Bosch BMA222E accelerometer on the I2C bus of the Minnowboard MAX development board exposed via the LSE connector [1], the following ASL code can be used:

```
DefinitionBlock ("minnowmax.aml", "SSDT", 1, "Vendor", "Accel", 0x00000003)
{
    External (\_SB.I2C6, DeviceObj)

    Scope (\_SB.I2C6)
    {
        Device (STAC)
        {
            Name (_HID, "BMA222E")
            Name (RBUF, ResourceTemplate ()
            {
                I2cSerialBus (0x0018, ControllerInitiated, 0x00061A80,
                            AddressingMode7Bit, "\\_SB.I2C6", 0x00,
                            ResourceConsumer, ,)
                GpioInt (Edge, ActiveHigh, Exclusive, PullDown, 0x0000,
                        "\\_SB.GPO2", 0x00, ResourceConsumer, , )
                { // Pin list
                    0
                }
            })

            Method (_CRS, 0, Serialized)
```

```
            {
                Return (RBUF)
            }
        }
    }
}
```

which can then be compiled to AML binary format:

```
$ iasl minnowmax.asl

Intel ACPI Component Architecture
ASL Optimizing Compiler version 20140214-64 [Mar 29 2014]
Copyright (c) 2000 - 2014 Intel Corporation

ASL Input:     minnomax.asl - 30 lines, 614 bytes, 7 keywords
AML Output:    minnowmax.aml - 165 bytes, 6 named objects, 1 executable opcodes
```

[1] https://www.elinux.org/Minnowboard:MinnowMax#Low_Speed_Expansion_.28Top.29

The resulting AML code can then be loaded by the kernel using one of the methods below.

22.2.1 Loading ACPI SSDTs from initrd

This option allows loading of user defined SSDTs from initrd and it is useful when the system does not support EFI or when there is not enough EFI storage.

It works in a similar way with initrd based ACPI tables override/upgrade: SSDT AML code must be placed in the first, uncompressed, initrd under the "kernel/firmware/acpi" path. Multiple files can be used and this will translate in loading multiple tables. Only SSDT and OEM tables are allowed. See initrd_table_override.txt for more details.

Here is an example:

```
# Add the raw ACPI tables to an uncompressed cpio archive.
# They must be put into a /kernel/firmware/acpi directory inside the
# cpio archive.
# The uncompressed cpio archive must be the first.
# Other, typically compressed cpio archives, must be
# concatenated on top of the uncompressed one.
mkdir -p kernel/firmware/acpi
cp ssdt.aml kernel/firmware/acpi

# Create the uncompressed cpio archive and concatenate the original initrd
# on top:
find kernel | cpio -H newc --create > /boot/instrumented_initrd
cat /boot/initrd >>/boot/instrumented_initrd
```

22.2.2 Loading ACPI SSDTs from EFI variables

This is the preferred method, when EFI is supported on the platform, because it allows a persistent, OS independent way of storing the user defined SSDTs. There is also work underway to implement EFI support for loading user defined SSDTs and using this method will make it easier to convert to the EFI loading mechanism when that will arrive. To enable it, the CONFIG_EFI_CUSTOM_SSDT_OVERLAYS should be chosen to y.

In order to load SSDTs from an EFI variable the `"efivar_ssdt=..."` kernel command line parameter can be used (the name has a limitation of 16 characters). The argument for the option is the variable name to use. If there are multiple variables with the same name but with different vendor GUIDs, all of them will be loaded.

In order to store the AML code in an EFI variable the efivarfs filesystem can be used. It is enabled and mounted by default in /sys/firmware/efi/efivars in all recent distribution.

Creating a new file in /sys/firmware/efi/efivars will automatically create a new EFI variable. Updating a file in /sys/firmware/efi/efivars will update the EFI variable. Please note that the file name needs to be specially formatted as ”Name-GUID” and that the first 4 bytes in the file (little-endian format) represent the attributes of the EFI variable (see EFI_VARIABLE_MASK in include/linux/efi.h). Writing to the file must also be done with one write operation.

For example, you can use the following bash script to create/update an EFI variable with the content from a given file:

```
#!/bin/sh -e

while [ -n "$1" ]; do
        case "$1" in
        "-f") filename="$2"; shift;;
        "-g") guid="$2"; shift;;
        *) name="$1";;
        esac
        shift
done

usage()
{
        echo "Syntax: ${0##*/} -f filename [ -g guid ] name"
        exit 1
}

[ -n "$name" -a -f "$filename" ] || usage

EFIVARFS="/sys/firmware/efi/efivars"

[ -d "$EFIVARFS" ] || exit 2

if stat -tf $EFIVARFS | grep -q -v de5e81e4; then
        mount -t efivarfs none $EFIVARFS
fi

# try to pick up an existing GUID
```

```
[ -n "$guid" ] || guid=$(find "$EFIVARFS" -name "$name-*" | head -n1 | cut -f2-
↪ -d-)

# use a randomly generated GUID
[ -n "$guid" ] || guid="$(cat /proc/sys/kernel/random/uuid)"

# efivarfs expects all of the data in one write
tmp=$(mktemp)
/bin/echo -ne "\007\000\000\000" | cat - $filename > $tmp
dd if=$tmp of="$EFIVARFS/$name-$guid" bs=$(stat -c %s $tmp)
rm $tmp
```

22.2.3 Loading ACPI SSDTs from configfs

This option allows loading of user defined SSDTs from user space via the configfs interface. The CONFIG_ACPI_CONFIGFS option must be select and configfs must be mounted. In the following examples, we assume that configfs has been mounted in /sys/kernel/config.

New tables can be loading by creating new directories in /sys/kernel/config/acpi/table and writing the SSDT AML code in the aml attribute:

```
cd /sys/kernel/config/acpi/table
mkdir my_ssdt
cat ~/ssdt.aml > my_ssdt/aml
```

22.3 Collaborative Processor Performance Control (CPPC)

22.3.1 CPPC

CPPC defined in the ACPI spec describes a mechanism for the OS to manage the performance of a logical processor on a contiguous and abstract performance scale. CPPC exposes a set of registers to describe abstract performance scale, to request performance levels and to measure per-cpu delivered performance.

For more details on CPPC please refer to the ACPI specification at:

http://uefi.org/specifications

Some of the CPPC registers are exposed via sysfs under:

```
/sys/devices/system/cpu/cpuX/acpi_cppc/
```

for each cpu X:

```
$ ls -lR  /sys/devices/system/cpu/cpu0/acpi_cppc/
/sys/devices/system/cpu/cpu0/acpi_cppc/:
total 0
-r--r--r-- 1 root root 65536 Mar  5 19:38 feedback_ctrs
-r--r--r-- 1 root root 65536 Mar  5 19:38 highest_perf
-r--r--r-- 1 root root 65536 Mar  5 19:38 lowest_freq
```

```
-r--r--r-- 1 root root 65536 Mar  5 19:38 lowest_nonlinear_perf
-r--r--r-- 1 root root 65536 Mar  5 19:38 lowest_perf
-r--r--r-- 1 root root 65536 Mar  5 19:38 nominal_freq
-r--r--r-- 1 root root 65536 Mar  5 19:38 nominal_perf
-r--r--r-- 1 root root 65536 Mar  5 19:38 reference_perf
-r--r--r-- 1 root root 65536 Mar  5 19:38 wraparound_time
```

- highest_perf : Highest performance of this processor (abstract scale).
- nominal_perf : Highest sustained performance of this processor (abstract scale).
- lowest_nonlinear_perf : Lowest performance of this processor with nonlinear power savings (abstract scale).
- lowest_perf : Lowest performance of this processor (abstract scale).
- lowest_freq : CPU frequency corresponding to lowest_perf (in MHz).
- nominal_freq : CPU frequency corresponding to nominal_perf (in MHz). The above frequencies should only be used to report processor performance in frequency instead of abstract scale. These values should not be used for any functional decisions.
- feedback_ctrs : Includes both Reference and delivered performance counter. Reference counter ticks up proportional to processor's reference performance. Delivered counter ticks up proportional to processor's delivered performance.
- wraparound_time: Minimum time for the feedback counters to wraparound (seconds).
- reference_perf : Performance level at which reference performance counter accumulates (abstract scale).

22.3.2 Computing Average Delivered Performance

Below describes the steps to compute the average performance delivered by taking two different snapshots of feedback counters at time T1 and T2.

T1: Read feedback_ctrs as fbc_t1
Wait or run some workload

T2: Read feedback_ctrs as fbc_t2

```
delivered_counter_delta = fbc_t2[del] - fbc_t1[del]
reference_counter_delta = fbc_t2[ref] - fbc_t1[ref]

delivered_perf = (reference_perf x delivered_counter_delta) / reference_
↪counter_delta
```

22.4 ACPI Fan Performance States

When the optional _FPS object is present under an ACPI device representing a fan (for example, PNP0C0B or INT3404), the ACPI fan driver creates additional "state*" attributes in the sysfs directory of the ACPI device in question. These attributes list properties of fan performance states.

For more information on _FPS refer to the ACPI specification at:

http://uefi.org/specifications

For instance, the contents of the INT3404 ACPI device sysfs directory may look as follows:

```
$ ls -l /sys/bus/acpi/devices/INT3404:00/
total 0
...
-r--r--r-- 1 root root 4096 Dec 13 20:38 state0
-r--r--r-- 1 root root 4096 Dec 13 20:38 state1
-r--r--r-- 1 root root 4096 Dec 13 20:38 state10
-r--r--r-- 1 root root 4096 Dec 13 20:38 state11
-r--r--r-- 1 root root 4096 Dec 13 20:38 state2
-r--r--r-- 1 root root 4096 Dec 13 20:38 state3
-r--r--r-- 1 root root 4096 Dec 13 20:38 state4
-r--r--r-- 1 root root 4096 Dec 13 20:38 state5
-r--r--r-- 1 root root 4096 Dec 13 20:38 state6
-r--r--r-- 1 root root 4096 Dec 13 20:38 state7
-r--r--r-- 1 root root 4096 Dec 13 20:38 state8
-r--r--r-- 1 root root 4096 Dec 13 20:38 state9
-r--r--r-- 1 root root 4096 Dec 13 01:00 status
...
```

where each of the "state*" files represents one performance state of the fan and contains a colon-separated list of 5 integer numbers (fields) with the following interpretation:

```
control_percent:trip_point_index:speed_rpm:noise_level_mdb:power_mw
```

- `control_percent`: The percent value to be used to set the fan speed to a specific level using the _FSL object (0-100).
- `trip_point_index`: The active cooling trip point number that corresponds to this performance state (0-9).
- `speed_rpm`: Speed of the fan in rotations per minute.
- `noise_level_mdb`: Audible noise emitted by the fan in this state in millidecibels.
- `power_mw`: Power draw of the fan in this state in milliwatts.

For example:

```
$cat /sys/bus/acpi/devices/INT3404:00/state1
25:0:3200:12500:1250
```

When a given field is not populated or its value provided by the platform firmware is invalid, the "not-defined" string is shown instead of the value.

22.4.1 ACPI Fan Fine Grain Control

When _FIF object specifies support for fine grain control, then fan speed can be set from 0 to 100% with the recommended minimum ”step size” via _FSL object. User can adjust fan speed using thermal sysfs cooling device.

Here use can look at fan performance states for a reference speed (speed_rpm) and set it by changing cooling device cur_state. If the fine grain control is supported then user can also adjust to some other speeds which are not defined in the performance states.

The support of fine grain control is presented via sysfs attribute ”fine_grain_control”. If fine grain control is present, this attribute will show ”1” otherwise ”0”.

This sysfs attribute is presented in the same directory as performance states.

22.4.2 ACPI Fan Performance Feedback

The optional _FST object provides status information for the fan device. This includes field to provide current fan speed in revolutions per minute at which the fan is rotating.

This speed is presented in the sysfs using the attribute ”fan_speed_rpm”, in the same directory as performance states.

CHAPTER

TWENTYTHREE

ATA OVER ETHERNET (AOE)

23.1 Introduction

ATA over Ethernet is a network protocol that provides simple access to block storage on the LAN.

http://support.coraid.com/documents/AoEr11.txt

The EtherDrive (R) HOWTO for 2.6 and 3.x kernels is found at ...

http://support.coraid.com/support/linux/EtherDrive-2.6-HOWTO.html

It has many tips and hints! Please see, especially, recommended tunings for virtual memory:

http://support.coraid.com/support/linux/EtherDrive-2.6-HOWTO-5.html#ss5.19

The aoetools are userland programs that are designed to work with this driver. The aoetools are on sourceforge.

http://aoetools.sourceforge.net/

The scripts in this Documentation/admin-guide/aoe directory are intended to document the use of the driver and are not necessary if you install the aoetools.

23.2 Creating Device Nodes

Users of udev should find the block device nodes created automatically, but to create all the necessary device nodes, use the udev configuration rules provided in udev.txt (in this directory).

There is a udev-install.sh script that shows how to install these rules on your system.

There is also an autoload script that shows how to edit /etc/modprobe.d/aoe.conf to ensure that the aoe module is loaded when necessary. Preloading the aoe module is preferable to autoloading, however, because AoE discovery takes a few seconds. It can be confusing when an AoE device is not present the first time the a command is run but appears a second later.

23.3 Using Device Nodes

”cat /dev/etherd/err” blocks, waiting for error diagnostic output, like any retransmitted packets.

”echo eth2 eth4 > /dev/etherd/interfaces” tells the aoe driver to limit ATA over Ethernet traffic to eth2 and eth4. AoE traffic from untrusted networks should be ignored as a matter of security. See also the aoe_iflist driver option described below.

”echo > /dev/etherd/discover” tells the driver to find out what AoE devices are available.

In the future these character devices may disappear and be replaced by sysfs counterparts. Using the commands in aoetools insulates users from these implementation details.

The block devices are named like this:

```
e{shelf}.{slot}
e{shelf}.{slot}p{part}
```

... so that ”e0.2” is the third blade from the left (slot 2) in the first shelf (shelf address zero). That's the whole disk. The first partition on that disk would be ”e0.2p1”.

23.4 Using sysfs

Each aoe block device in /sys/block has the extra attributes of state, mac, and netif. The state attribute is ”up” when the device is ready for I/O and ”down” if detected but unusable. The ”down,closewait” state shows that the device is still open and cannot come up again until it has been closed.

The mac attribute is the ethernet address of the remote AoE device. The netif attribute is the network interface on the localhost through which we are communicating with the remote AoE device.

There is a script in this directory that formats this information in a convenient way. Users with aoetools should use the aoe-stat command:

```
root@makki root# sh Documentation/admin-guide/aoe/status.sh
   e10.0            eth3              up
   e10.1            eth3              up
   e10.2            eth3              up
   e10.3            eth3              up
   e10.4            eth3              up
   e10.5            eth3              up
   e10.6            eth3              up
   e10.7            eth3              up
   e10.8            eth3              up
   e10.9            eth3              up
    e4.0            eth1              up
    e4.1            eth1              up
    e4.2            eth1              up
    e4.3            eth1              up
```

```
    e4.4            eth1            up
    e4.5            eth1            up
    e4.6            eth1            up
    e4.7            eth1            up
    e4.8            eth1            up
    e4.9            eth1            up
```

Use /sys/module/aoe/parameters/aoe_iflist (or better, the driver option discussed below) instead of /dev/etherd/interfaces to limit AoE traffic to the network interfaces in the given whitespace-separated list. Unlike the old character device, the sysfs entry can be read from as well as written to.

It's helpful to trigger discovery after setting the list of allowed interfaces. The aoe-tools package provides an aoe-discover script for this purpose. You can also directly use the /dev/etherd/discover special file described above.

23.5 Driver Options

There is a boot option for the built-in aoe driver and a corresponding module parameter, aoe_iflist. Without this option, all network interfaces may be used for ATA over Ethernet. Here is a usage example for the module parameter:

```
modprobe aoe_iflist="eth1 eth3"
```

The aoe_deadsecs module parameter determines the maximum number of seconds that the driver will wait for an AoE device to provide a response to an AoE command. After aoe_deadsecs seconds have elapsed, the AoE device will be marked as ”down”. A value of zero is supported for testing purposes and makes the aoe driver keep trying AoE commands forever.

The aoe_maxout module parameter has a default of 128. This is the maximum number of unresponded packets that will be sent to an AoE target at one time.

The aoe_dyndevs module parameter defaults to 1, meaning that the driver will assign a block device minor number to a discovered AoE target based on the order of its discovery. With dynamic minor device numbers in use, a greater range of AoE shelf and slot addresses can be supported. Users with udev will never have to think about minor numbers. Using aoe_dyndevs=0 allows device nodes to be pre-created using a static minor-number scheme with the aoe-mkshelf script in the aoetools.

23.6 TODO

There is a potential for deadlock when allocating a struct sk_buff for data that needs to be written out to aoe storage. If the data is being written from a dirty page in order to free that page, and if there are no other pages available, then deadlock may occur when a free page is needed for the sk_buff allocation. This situation has not been observed, but it would be nice to eliminate any potential for deadlock under memory pressure.

Because ATA over Ethernet is not fragmented by the kernel's IP code, the destructor member of the struct sk_buff is available to the aoe driver. By using a mempool for allocating all but

the first few sk_buffs, and by registering a destructor, we should be able to efficiently allocate sk_buffs without introducing any potential for deadlock.

23.7 Example of udev rules

```
# These rules tell udev what device nodes to create for aoe support.
# They may be installed along the following lines.  Check the section
# 8 udev manpage to see whether your udev supports SUBSYSTEM, and
# whether it uses one or two equal signs for SUBSYSTEM and KERNEL.
#
#   ecashin@makki ~$ su
#   Password:
#   bash# find /etc -type f -name udev.conf
#   /etc/udev/udev.conf
#   bash# grep udev_rules= /etc/udev/udev.conf
#   udev_rules="/etc/udev/rules.d/"
#   bash# ls /etc/udev/rules.d/
#   10-wacom.rules  50-udev.rules
#   bash# cp /path/to/linux/Documentation/admin-guide/aoe/udev.txt \
#           /etc/udev/rules.d/60-aoe.rules
#

# aoe char devices
SUBSYSTEM=="aoe", KERNEL=="discover",   NAME="etherd/%k", GROUP="disk",␣
↪MODE="0220"
SUBSYSTEM=="aoe", KERNEL=="err",        NAME="etherd/%k", GROUP="disk",␣
↪MODE="0440"
SUBSYSTEM=="aoe", KERNEL=="interfaces", NAME="etherd/%k", GROUP="disk",␣
↪MODE="0220"
SUBSYSTEM=="aoe", KERNEL=="revalidate", NAME="etherd/%k", GROUP="disk",␣
↪MODE="0220"
SUBSYSTEM=="aoe", KERNEL=="flush",      NAME="etherd/%k", GROUP="disk",␣
↪MODE="0220"

# aoe block devices
KERNEL=="etherd*",       GROUP="disk"
```

23.8 Example of udev install rules script

```
# install the aoe-specific udev rules from udev.txt into
# the system's udev configuration
#

me="`basename $0`"

# find udev.conf, often /etc/udev/udev.conf
# (or environment can specify where to find udev.conf)
```

```
#
if test -z "$conf"; then
        if test -r /etc/udev/udev.conf; then
                conf=/etc/udev/udev.conf
        else
                conf="`find /etc -type f -name udev.conf 2> /dev/null`"
                if test -z "$conf" || test ! -r "$conf"; then
                        echo "$me Error: no udev.conf found" 1>&2
                        exit 1
                fi
        fi
fi

# find the directory where udev rules are stored, often
# /etc/udev/rules.d
#
rules_d="`sed -n '/^udev_rules=/{ s!udev_rules=!!; s!\"!!g; p; }'
 ↪$conf`"
if test -z "$rules_d" ; then
        rules_d=/etc/udev/rules.d
fi
if test ! -d "$rules_d"; then
        echo "$me Error: cannot find udev rules directory" 1>&2
        exit 1
fi
sh -xc "cp `dirname $0`/udev.txt $rules_d/60-aoe.rules"
```

23.9 Example script to get status

```
#! /bin/sh
# collate and present sysfs information about AoE storage
#
# A more complete version of this script is aoe-stat, in the
# aoetools.

set -e
format="%8s\t%8s\t%8s\n"
me=`basename $0`
sysd=${sysfs_dir:-/sys}

# printf "$format" device mac netif state

# Suse 9.1 Pro doesn't put /sys in /etc/mtab
#test -z "`mount | grep sysfs`" && {
test ! -d "$sysd/block" && {
        echo "$me Error: sysfs is not mounted" 1>&2
        exit 1
}
```

```
for d in `ls -d $sysd/block/etherd* 2>/dev/null | grep -v p` end; do
        # maybe ls comes up empty, so we use "end"
        test $d = end && continue

        dev=`echo "$d" | sed 's/.*!//'`
        printf "$format" \
                "$dev" \
                "`cat \"$d/netif\"`" \
                "`cat \"$d/state\"`"
done | sort
```

23.10 Example of AoE autoload script

```
#!/bin/sh
# set aoe to autoload by installing the
# aliases in /etc/modprobe.d/

f=/etc/modprobe.d/aoe.conf

if test ! -r $f || test ! -w $f; then
        echo "cannot configure $f for module autoloading" 1>&2
        exit 1
fi

grep major-152 $f >/dev/null
if [ $? = 1 ]; then
        echo alias block-major-152 aoe >> $f
        echo alias char-major-152 aoe >> $f
fi
```

CHAPTER

TWENTYFOUR

AUXILIARY DISPLAY SUPPORT

24.1 ks0108 LCD Controller Driver Documentation

License
GPLv2

Author & Maintainer
Miguel Ojeda <ojeda@kernel.org>

Date
2006-10-27

24.1.1 1. Driver Information

This driver supports the ks0108 LCD controller.

24.1.2 2. Device Information

Manufacturer
Samsung

Device Name
KS0108 LCD Controller

Device Code
ks0108

Webpage

-

Device Webpage

-

Type
LCD Controller (Liquid Crystal Display Controller)

Width
64

Height
64

Colors
2 (B/N)

Pages
8

Addresses
64 each page

Data size
1 byte each address

Memory size
8 * 64 * 1 = 512 bytes

24.1.3 3. Wiring

The driver supports data parallel port wiring.

If you aren't building LCD related hardware, you should check your LCD specific wiring information in the same folder.

For example, check *cfag12864b LCD Driver Documentation*

24.2 cfag12864b LCD Driver Documentation

License
GPLv2

Author & Maintainer
Miguel Ojeda <ojeda@kernel.org>

Date
2006-10-27

24.2.1 1. Driver Information

This driver supports a cfag12864b LCD.

24.2.2 2. Device Information

Manufacturer
Crystalfontz

Device Name
Crystalfontz 12864b LCD Series

Device Code
cfag12864b

Webpage
http://www.crystalfontz.com

Device Webpage
http://www.crystalfontz.com/products/12864b/

Type
LCD (Liquid Crystal Display)

Width
128

Height
64

Colors
2 (B/N)

Controller
ks0108

Controllers
2

Pages
8 each controller

Addresses
64 each page

Data size
1 byte each address

Memory size
2 * 8 * 64 * 1 = 1024 bytes = 1 Kbyte

24.2.3 3. Wiring

The cfag12864b LCD Series don't have official wiring.

The common wiring is done to the parallel port as shown:

```
Parallel Port                          cfag12864b

  Name Pin#                            Pin# Name

Strobe ( 1)-------------------------------(17) Enable
Data 0 ( 2)-------------------------------( 4) Data 0
Data 1 ( 3)-------------------------------( 5) Data 1
Data 2 ( 4)-------------------------------( 6) Data 2
Data 3 ( 5)-------------------------------( 7) Data 3
Data 4 ( 6)-------------------------------( 8) Data 4
Data 5 ( 7)-------------------------------( 9) Data 5
Data 6 ( 8)-------------------------------(10) Data 6
Data 7 ( 9)-------------------------------(11) Data 7
       (10)                       [+5v]---( 1) Vdd
       (11)                       [GND]---( 2) Ground
       (12)                       [+5v]---(14) Reset
       (13)                       [GND]---(15) Read / Write
```

```
  Line (14)------------------------------(13) Controller Select 1
       (15)
  Init (16)------------------------------(12) Controller Select 2
Select (17)------------------------------(16) Data / Instruction
Ground (18)---[GND]              [+5v]---(19) LED +
Ground (19)---[GND]
Ground (20)---[GND]              E    A             Values:
Ground (21)---[GND]       [GND]---[P1]---(18) Vee    - R = Resistor = 22 ohm
Ground (22)---[GND]                |                 - P1 = Preset = 10 Kohm
Ground (23)---[GND]       ----   S ------( 3) V0     - P2 = Preset = 1 Kohm
Ground (24)---[GND]       |  |
Ground (25)---[GND] [GND]---[P2]---[R]---(20) LED -
```

24.2.4 4. Userspace Programming

The cfag12864bfb describes a framebuffer device (/dev/fbX).

It has a size of 1024 bytes = 1 Kbyte. Each bit represents one pixel. If the bit is high, the pixel will turn on. If the pixel is low, the pixel will turn off.

You can use the framebuffer as a file: fopen, fwrite, fclose... Although the LCD won't get updated until the next refresh time arrives.

Also, you can mmap the framebuffer: open & mmap, munmap & close... which is the best option for most uses.

Check samples/auxdisplay/cfag12864b-example.c for a real working userspace complete program with usage examples.

CHAPTER

TWENTYFIVE

A BLOCK LAYER CACHE (BCACHE)

Say you've got a big slow raid 6, and an ssd or three. Wouldn't it be nice if you could use them as cache... Hence bcache.

The bcache wiki can be found at:
https://bcache.evilpiepirate.org

This is the git repository of bcache-tools:
https://git.kernel.org/pub/scm/linux/kernel/git/colyli/bcache-tools.git/

The latest bcache kernel code can be found from mainline Linux kernel:
https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/

It's designed around the performance characteristics of SSDs - it only allocates in erase block sized buckets, and it uses a hybrid btree/log to track cached extents (which can be anywhere from a single sector to the bucket size). It's designed to avoid random writes at all costs; it fills up an erase block sequentially, then issues a discard before reusing it.

Both writethrough and writeback caching are supported. Writeback defaults to off, but can be switched on and off arbitrarily at runtime. Bcache goes to great lengths to protect your data - it reliably handles unclean shutdown. (It doesn't even have a notion of a clean shutdown; bcache simply doesn't return writes as completed until they're on stable storage).

Writeback caching can use most of the cache for buffering writes - writing dirty data to the backing device is always done sequentially, scanning from the start to the end of the index.

Since random IO is what SSDs excel at, there generally won't be much benefit to caching large sequential IO. Bcache detects sequential IO and skips it; it also keeps a rolling average of the IO sizes per task, and as long as the average is above the cutoff it will skip all IO from that task - instead of caching the first 512k after every seek. Backups and large file copies should thus entirely bypass the cache.

In the event of a data IO error on the flash it will try to recover by reading from disk or invalidating cache entries. For unrecoverable errors (meta data or dirty data), caching is automatically disabled; if dirty data was present in the cache it first disables writeback caching and waits for all dirty data to be flushed.

Getting started: You'll need bcache util from the bcache-tools repository. Both the cache device and backing device must be formatted before use:

```
bcache make -B /dev/sdb
bcache make -C /dev/sdc
```

bcache make has the ability to format multiple devices at the same time - if you format your backing devices and cache device at the same time, you won't have to manually attach:

```
bcache make -B /dev/sda /dev/sdb -C /dev/sdc
```

If your bcache-tools is not updated to latest version and does not have the unified *bcache* utility, you may use the legacy *make-bcache* utility to format bcache device with same -B and -C parameters.

bcache-tools now ships udev rules, and bcache devices are known to the kernel immediately. Without udev, you can manually register devices like this:

```
echo /dev/sdb > /sys/fs/bcache/register
echo /dev/sdc > /sys/fs/bcache/register
```

Registering the backing device makes the bcache device show up in /dev; you can now format it and use it as normal. But the first time using a new bcache device, it'll be running in passthrough mode until you attach it to a cache. If you are thinking about using bcache later, it is recommended to setup all your slow devices as bcache backing devices without a cache, and you can choose to add a caching device later. See 'ATTACHING' section below.

The devices show up as:

```
/dev/bcache<N>
```

As well as (with udev):

```
/dev/bcache/by-uuid/<uuid>
/dev/bcache/by-label/<label>
```

To get started:

```
mkfs.ext4 /dev/bcache0
mount /dev/bcache0 /mnt
```

You can control bcache devices through sysfs at /sys/block/bcache<N>/bcache . You can also control them through /sys/fs//bcache/<cset-uuid>/ .

Cache devices are managed as sets; multiple caches per set isn't supported yet but will allow for mirroring of metadata and dirty data in the future. Your new cache set shows up as /sys/fs/bcache/<UUID>

25.1 Attaching

After your cache device and backing device are registered, the backing device must be attached to your cache set to enable caching. Attaching a backing device to a cache set is done thusly, with the UUID of the cache set in /sys/fs/bcache:

```
echo <CSET-UUID> > /sys/block/bcache0/bcache/attach
```

This only has to be done once. The next time you reboot, just reregister all your bcache devices. If a backing device has data in a cache somewhere, the /dev/bcache<N> device won't be created until the cache shows up - particularly important if you have writeback caching turned on.

If you're booting up and your cache device is gone and never coming back, you can force run the backing device:

```
echo 1 > /sys/block/sdb/bcache/running
```

(You need to use /sys/block/sdb (or whatever your backing device is called), not /sys/block/bcache0, because bcache0 doesn't exist yet. If you're using a partition, the bcache directory would be at /sys/block/sdb/sdb2/bcache)

The backing device will still use that cache set if it shows up in the future, but all the cached data will be invalidated. If there was dirty data in the cache, don't expect the filesystem to be recoverable - you will have massive filesystem corruption, though ext4's fsck does work miracles.

25.2 Error Handling

Bcache tries to transparently handle IO errors to/from the cache device without affecting normal operation; if it sees too many errors (the threshold is configurable, and defaults to 0) it shuts down the cache device and switches all the backing devices to passthrough mode.

- For reads from the cache, if they error we just retry the read from the backing device.
- For writethrough writes, if the write to the cache errors we just switch to invalidating the data at that lba in the cache (i.e. the same thing we do for a write that bypasses the cache)
- For writeback writes, we currently pass that error back up to the filesystem/userspace. This could be improved - we could retry it as a write that skips the cache so we don't have to error the write.
- When we detach, we first try to flush any dirty data (if we were running in writeback mode). It currently doesn't do anything intelligent if it fails to read some of the dirty data, though.

25.3 Howto/cookbook

A) Starting a bcache with a missing caching device

If registering the backing device doesn't help, it's already there, you just need to force it to run without the cache:

```
host:~# echo /dev/sdb1 > /sys/fs/bcache/register
[  119.844831] bcache: register_bcache() error opening /dev/sdb1: device␣
↪already registered
```

Next, you try to register your caching device if it's present. However if it's absent, or registration fails for some reason, you can still start your bcache without its cache, like so:

```
host:/sys/block/sdb/sdb1/bcache# echo 1 > running
```

Note that this may cause data loss if you were running in writeback mode.

B) Bcache does not find its cache:

```
host:/sys/block/md5/bcache# echo 0226553a-37cf-41d5-b3ce-8b1e944543a8 >␣
↪attach
[ 1933.455082] bcache: bch_cached_dev_attach() Couldn't find uuid for md5␣
↪in set
```

```
[ 1933.478179] bcache: __cached_dev_store() Can't attach 0226553a-37cf-
↪41d5-b3ce-8b1e944543a8
[ 1933.478179] : cache set not found
```

In this case, the caching device was simply not registered at boot or disappeared and came back, and needs to be (re-)registered:

```
host:/sys/block/md5/bcache# echo /dev/sdh2 > /sys/fs/bcache/register
```

C) Corrupt bcache crashes the kernel at device registration time:

This should never happen. If it does happen, then you have found a bug! Please report it to the bcache development list: linux-bcache@vger.kernel.org

Be sure to provide as much information that you can including kernel dmesg output if available so that we may assist.

D) Recovering data without bcache:

If bcache is not available in the kernel, a filesystem on the backing device is still available at an 8KiB offset. So either via a loopdev of the backing device created with --offset 8K, or any value defined by --data-offset when you originally formatted bcache with *bcache make*.

For example:

```
losetup -o 8192 /dev/loop0 /dev/your_bcache_backing_dev
```

This should present your unmodified backing device data in /dev/loop0

If your cache is in writethrough mode, then you can safely discard the cache device without losing data.

E) Wiping a cache device

```
host:~# wipefs -a /dev/sdh2
16 bytes were erased at offset 0x1018 (bcache)
they were: c6 85 73 f6 4e 1a 45 ca 82 65 f5 7f 48 ba 6d 81
```

After you boot back with bcache enabled, you recreate the cache and attach it:

```
host:~# bcache make -C /dev/sdh2
UUID:                   7be7e175-8f4c-4f99-94b2-9c904d227045
Set UUID:               5bc072a8-ab17-446d-9744-e247949913c1
version:                0
nbuckets:               106874
block_size:             1
bucket_size:            1024
nr_in_set:              1
nr_this_dev:            0
first_bucket:           1
[  650.511912] bcache: run_cache_set() invalidating existing data
[  650.549228] bcache: register_cache() registered cache device sdh2
```

start backing device with missing cache:

```
host:/sys/block/md5/bcache# echo 1 > running
```

attach new cache:

```
host:/sys/block/md5/bcache# echo 5bc072a8-ab17-446d-9744-e247949913c1 > attach
[  865.276616] bcache: bch_cached_dev_attach() Caching md5 as bcache0 on set␣
↪5bc072a8-ab17-446d-9744-e247949913c1
```

F) Remove or replace a caching device:

```
host:/sys/block/sda/sda7/bcache# echo 1 > detach
[  695.872542] bcache: cached_dev_detach_finish() Caching disabled for sda7

host:~# wipefs -a /dev/nvme0n1p4
wipefs: error: /dev/nvme0n1p4: probing initialization failed: Device or␣
↪resource busy
Ooops, it's disabled, but not unregistered, so it's still protected
```

We need to go and unregister it:

```
host:/sys/fs/bcache/b7ba27a1-2398-4649-8ae3-0959f57ba128# ls -l cache0
lrwxrwxrwx 1 root root 0 Feb 25 18:33 cache0 -> ../../../devices/pci0000:00/
↪0000:00:1d.0/0000:70:00.0/nvme/nvme0/nvme0n1/nvme0n1p4/bcache/
host:/sys/fs/bcache/b7ba27a1-2398-4649-8ae3-0959f57ba128# echo 1 > stop
kernel: [  917.041908] bcache: cache_set_free() Cache set b7ba27a1-2398-4649-
↪8ae3-0959f57ba128 unregistered
```

Now we can wipe it:

```
host:~# wipefs -a /dev/nvme0n1p4
/dev/nvme0n1p4: 16 bytes were erased at offset 0x00001018 (bcache): c6 85 73␣
↪f6 4e 1a 45 ca 82 65 f5 7f 48 ba 6d 81
```

G) dm-crypt and bcache

First setup bcache unencrypted and then install dmcrypt on top of /dev/bcache<N> This will work faster than if you dmcrypt both the backing and caching devices and then install bcache on top. [benchmarks?]

H) Stop/free a registered bcache to wipe and/or recreate it

Suppose that you need to free up all bcache references so that you can fdisk run and re-register a changed partition table, which won't work if there are any active backing or caching devices left on it:

1) Is it present in /dev/bcache* ? (there are times where it won't be)

   If so, it's easy:

   ```
   host:/sys/block/bcache0/bcache# echo 1 > stop
   ```

2) But if your backing device is gone, this won't work:

```
host:/sys/block/bcache0# cd bcache
bash: cd: bcache: No such file or directory
```

In this case, you may have to unregister the dmcrypt block device that references this bcache to free it up:

```
host:~# dmsetup remove oldds1
bcache: bcache_device_free() bcache0 stopped
bcache: cache_set_free() Cache set 5bc072a8-ab17-446d-9744-e247949913c1␣
↪unregistered
```

This causes the backing bcache to be removed from /sys/fs/bcache and then it can be reused. This would be true of any block device stacking where bcache is a lower device.

3) In other cases, you can also look in /sys/fs/bcache/:

```
host:/sys/fs/bcache# ls -l */{cache?,bdev?}
lrwxrwxrwx 1 root root 0 Mar  5 09:39 0226553a-37cf-41d5-b3ce-8b1e944543a8/
↪bdev1 -> ../../../devices/virtual/block/dm-1/bcache/
lrwxrwxrwx 1 root root 0 Mar  5 09:39 0226553a-37cf-41d5-b3ce-8b1e944543a8/
↪cache0 -> ../../../devices/virtual/block/dm-4/bcache/
lrwxrwxrwx 1 root root 0 Mar  5 09:39 5bc072a8-ab17-446d-9744-e247949913c1/
↪cache0 -> ../../../devices/pci0000:00/0000:00:01.0/0000:01:00.0/ata10/
↪host9/target9:0:0/9:0:0:0/block/sdl/sdl2/bcache/
```

The device names will show which UUID is relevant, cd in that directory and stop the cache:

```
host:/sys/fs/bcache/5bc072a8-ab17-446d-9744-e247949913c1# echo 1 > stop
```

This will free up bcache references and let you reuse the partition for other purposes.

25.4 Troubleshooting performance

Bcache has a bunch of config options and tunables. The defaults are intended to be reasonable for typical desktop and server workloads, but they're not what you want for getting the best possible numbers when benchmarking.

- Backing device alignment

  The default metadata size in bcache is 8k. If your backing device is RAID based, then be sure to align this by a multiple of your stride width using *bcache make --data-offset*. If you intend to expand your disk array in the future, then multiply a series of primes by your raid stripe size to get the disk multiples that you would like.

  For example: If you have a 64k stripe size, then the following offset would provide alignment for many common RAID5 data spindle counts:

  ```
  64k * 2*2*2*3*3*5*7 bytes = 161280k
  ```

  That space is wasted, but for only 157.5MB you can grow your RAID 5 volume to the following data-spindle counts without re-aligning:

```
3,4,5,6,7,8,9,10,12,14,15,18,20,21 ...
```

- Bad write performance

  If write performance is not what you expected, you probably wanted to be running in writeback mode, which isn't the default (not due to a lack of maturity, but simply because in writeback mode you'll lose data if something happens to your SSD):

```
# echo writeback > /sys/block/bcache0/bcache/cache_mode
```

- Bad performance, or traffic not going to the SSD that you'd expect

  By default, bcache doesn't cache everything. It tries to skip sequential IO - because you really want to be caching the random IO, and if you copy a 10 gigabyte file you probably don't want that pushing 10 gigabytes of randomly accessed data out of your cache.

  But if you want to benchmark reads from cache, and you start out with fio writing an 8 gigabyte test file - so you want to disable that:

```
# echo 0 > /sys/block/bcache0/bcache/sequential_cutoff
```

  To set it back to the default (4 mb), do:

```
# echo 4M > /sys/block/bcache0/bcache/sequential_cutoff
```

- Traffic's still going to the spindle/still getting cache misses

  In the real world, SSDs don't always keep up with disks - particularly with slower SSDs, many disks being cached by one SSD, or mostly sequential IO. So you want to avoid being bottlenecked by the SSD and having it slow everything down.

  To avoid that bcache tracks latency to the cache device, and gradually throttles traffic if the latency exceeds a threshold (it does this by cranking down the sequential bypass).

  You can disable this if you need to by setting the thresholds to 0:

```
# echo 0 > /sys/fs/bcache/<cache set>/congested_read_threshold_us
# echo 0 > /sys/fs/bcache/<cache set>/congested_write_threshold_us
```

  The default is 2000 us (2 milliseconds) for reads, and 20000 for writes.

- Still getting cache misses, of the same data

  One last issue that sometimes trips people up is actually an old bug, due to the way cache coherency is handled for cache misses. If a btree node is full, a cache miss won't be able to insert a key for the new data and the data won't be written to the cache.

  In practice this isn't an issue because as soon as a write comes along it'll cause the btree node to be split, and you need almost no write traffic for this to not show up enough to be noticeable (especially since bcache's btree nodes are huge and index large regions of the device). But when you're benchmarking, if you're trying to warm the cache by reading a bunch of data and there's no other traffic - that can be a problem.

  Solution: warm the cache by doing writes, or use the testing branch (there's a fix for the issue there).

25.5 Sysfs - backing device

Available at /sys/block/<bdev>/bcache, /sys/block/bcache*/bcache and (if attached) /sys/fs/bcache/<cset-uuid>/bdev*

attach
Echo the UUID of a cache set to this file to enable caching.

cache_mode
Can be one of either writethrough, writeback, writearound or none.

clear_stats
Writing to this file resets the running total stats (not the day/hour/5 minute decaying versions).

detach
Write to this file to detach from a cache set. If there is dirty data in the cache, it will be flushed first.

dirty_data
Amount of dirty data for this backing device in the cache. Continuously updated unlike the cache set's version, but may be slightly off.

label
Name of underlying device.

readahead
Size of readahead that should be performed. Defaults to 0. If set to e.g. 1M, it will round cache miss reads up to that size, but without overlapping existing cache entries.

running
1 if bcache is running (i.e. whether the /dev/bcache device exists, whether it's in passthrough mode or caching).

sequential_cutoff
A sequential IO will bypass the cache once it passes this threshold; the most recent 128 IOs are tracked so sequential IO can be detected even when it isn't all done at once.

sequential_merge
If non zero, bcache keeps a list of the last 128 requests submitted to compare against all new requests to determine which new requests are sequential continuations of previous requests for the purpose of determining sequential cutoff. This is necessary if the sequential cutoff value is greater than the maximum acceptable sequential size for any single request.

state
The backing device can be in one of four different states:

no cache: Has never been attached to a cache set.

clean: Part of a cache set, and there is no cached dirty data.

dirty: Part of a cache set, and there is cached dirty data.

inconsistent: The backing device was forcibly run by the user when there was dirty data cached but the cache set was unavailable; whatever data was on the backing device has likely been corrupted.

stop
Write to this file to shut down the bcache device and close the backing device.

writeback_delay
When dirty data is written to the cache and it previously did not contain any, waits some number of seconds before initiating writeback. Defaults to 30.

writeback_percent
If nonzero, bcache tries to keep around this percentage of the cache dirty by throttling background writeback and using a PD controller to smoothly adjust the rate.

writeback_rate
Rate in sectors per second - if writeback_percent is nonzero, background writeback is throttled to this rate. Continuously adjusted by bcache but may also be set by the user.

writeback_running
If off, writeback of dirty data will not take place at all. Dirty data will still be added to the cache until it is mostly full; only meant for benchmarking. Defaults to on.

25.5.1 Sysfs - backing device stats

There are directories with these numbers for a running total, as well as versions that decay over the past day, hour and 5 minutes; they're also aggregated in the cache set directory as well.

bypassed
Amount of IO (both reads and writes) that has bypassed the cache

cache_hits, cache_misses, cache_hit_ratio
Hits and misses are counted per individual IO as bcache sees them; a partial hit is counted as a miss.

cache_bypass_hits, cache_bypass_misses
Hits and misses for IO that is intended to skip the cache are still counted, but broken out here.

cache_miss_collisions
Counts instances where data was going to be inserted into the cache from a cache miss, but raced with a write and data was already present (usually 0 since the synchronization for cache misses was rewritten)

25.5.2 Sysfs - cache set

Available at /sys/fs/bcache/<cset-uuid>

average_key_size
Average data per key in the btree.

bdev<0..n>
Symlink to each of the attached backing devices.

block_size
Block size of the cache devices.

btree_cache_size
Amount of memory currently used by the btree cache

bucket_size
Size of buckets

cache<0..n>
Symlink to each of the cache devices comprising this cache set.

cache_available_percent
Percentage of cache device which doesn't contain dirty data, and could potentially be used for writeback. This doesn't mean this space isn't used for clean cached data; the unused statistic (in priority_stats) is typically much lower.

clear_stats
Clears the statistics associated with this cache

dirty_data
Amount of dirty data is in the cache (updated when garbage collection runs).

flash_vol_create
Echoing a size to this file (in human readable units, k/M/G) creates a thinly provisioned volume backed by the cache set.

io_error_halflife, io_error_limit
These determines how many errors we accept before disabling the cache. Each error is decayed by the half life (in # ios). If the decaying count reaches io_error_limit dirty data is written out and the cache is disabled.

journal_delay_ms
Journal writes will delay for up to this many milliseconds, unless a cache flush happens sooner. Defaults to 100.

root_usage_percent
Percentage of the root btree node in use. If this gets too high the node will split, increasing the tree depth.

stop
Write to this file to shut down the cache set - waits until all attached backing devices have been shut down.

tree_depth
Depth of the btree (A single node btree has depth 0).

unregister
Detaches all backing devices and closes the cache devices; if dirty data is present it will disable writeback caching and wait for it to be flushed.

25.5.3 Sysfs - cache set internal

This directory also exposes timings for a number of internal operations, with separate files for average duration, average frequency, last occurrence and max duration: garbage collection, btree read, btree node sorts and btree splits.

active_journal_entries
Number of journal entries that are newer than the index.

btree_nodes
Total nodes in the btree.

btree_used_percent
Average fraction of btree in use.

bset_tree_stats
Statistics about the auxiliary search trees

btree_cache_max_chain
Longest chain in the btree node cache's hash table

cache_read_races
Counts instances where while data was being read from the cache, the bucket was reused and invalidated - i.e. where the pointer was stale after the read completed. When this occurs the data is reread from the backing device.

trigger_gc
Writing to this file forces garbage collection to run.

25.5.4 Sysfs - Cache device

Available at /sys/block/<cdev>/bcache

block_size
Minimum granularity of writes - should match hardware sector size.

btree_written
Sum of all btree writes, in (kilo/mega/giga) bytes

bucket_size
Size of buckets

cache_replacement_policy
One of either lru, fifo or random.

discard
Boolean; if on a discard/TRIM will be issued to each bucket before it is reused. Defaults to off, since SATA TRIM is an unqueued command (and thus slow).

freelist_percent
Size of the freelist as a percentage of nbuckets. Can be written to to increase the number of buckets kept on the freelist, which lets you artificially reduce the size of the cache at runtime. Mostly for testing purposes (i.e. testing how different size caches affect your hit rate), but since buckets are discarded when they move on to the freelist will also make the SSD's garbage collection easier by effectively giving it more reserved space.

io_errors
Number of errors that have occurred, decayed by io_error_halflife.

metadata_written
Sum of all non data writes (btree writes and all other metadata).

nbuckets
Total buckets in this cache

priority_stats
Statistics about how recently data in the cache has been accessed. This can reveal your working set size. Unused is the percentage of the cache that doesn't contain any data. Metadata is bcache's metadata overhead. Average is the average priority of cache buckets. Next is a list of quantiles with the priority threshold of each.

written

Sum of all data that has been written to the cache; comparison with btree_written gives the amount of write inflation in bcache.

CHAPTER

TWENTYSIX

THE ANDROID BINDERFS FILESYSTEM

Android binderfs is a filesystem for the Android binder IPC mechanism. It allows to dynamically add and remove binder devices at runtime. Binder devices located in a new binderfs instance are independent of binder devices located in other binderfs instances. Mounting a new binderfs instance makes it possible to get a set of private binder devices.

26.1 Mounting binderfs

Android binderfs can be mounted with:

```
mkdir /dev/binderfs
mount -t binder binder /dev/binderfs
```

at which point a new instance of binderfs will show up at `/dev/binderfs`. In a fresh instance of binderfs no binder devices will be present. There will only be a `binder-control` device which serves as the request handler for binderfs. Mounting another binderfs instance at a different location will create a new and separate instance from all other binderfs mounts. This is identical to the behavior of e.g. `devpts` and `tmpfs`. The Android binderfs filesystem can be mounted in user namespaces.

26.2 Options

max
: binderfs instances can be mounted with a limit on the number of binder devices that can be allocated. The `max=<count>` mount option serves as a per-instance limit. If `max=<count>` is set then only `<count>` number of binder devices can be allocated in this binderfs instance.

stats
: Using `stats=global` enables global binder statistics. `stats=global` is only available for a binderfs instance mounted in the initial user namespace. An attempt to use the option to mount a binderfs instance in another user namespace will return a permission error.

26.3 Allocating binder Devices

To allocate a new binder device in a binderfs instance a request needs to be sent through the `binder-control` device node. A request is sent in the form of an ioctl().

What a program needs to do is to open the `binder-control` device node and send a `BINDER_CTL_ADD` request to the kernel. Users of binderfs need to tell the kernel which name the new binder device should get. By default a name can only contain up to `BINDERFS_MAX_NAME` chars including the terminating zero byte.

Once the request is made via an ioctl() passing a `struct binder_device` with the name to the kernel it will allocate a new binder device and return the major and minor number of the new device in the struct (This is necessary because binderfs allocates a major device number dynamically.). After the ioctl() returns there will be a new binder device located under /dev/binderfs with the chosen name.

26.4 Deleting binder Devices

Binderfs binder devices can be deleted via unlink(). This means that the rm() tool can be used to delete them. Note that the `binder-control` device cannot be deleted since this would make the binderfs instance unusable. The `binder-control` device will be deleted when the binderfs instance is unmounted and all references to it have been dropped.

26.5 Binder features

Assuming an instance of binderfs has been mounted at `/dev/binderfs`, the features supported by the binder driver can be located under `/dev/binderfs/features/`. The presence of individual files can be tested to determine whether a particular feature is supported by the driver.

Example:

```
cat /dev/binderfs/features/oneway_spam_detection
1
```

CHAPTER

TWENTYSEVEN

KERNEL SUPPORT FOR MISCELLANEOUS BINARY FORMATS (BINFMT_MISC)

This Kernel feature allows you to invoke almost (for restrictions see below) every program by simply typing its name in the shell. This includes for example compiled Java(TM), Python or Emacs programs.

To achieve this you must tell binfmt_misc which interpreter has to be invoked with which binary. Binfmt_misc recognises the binary-type by matching some bytes at the beginning of the file with a magic byte sequence (masking out specified bits) you have supplied. Binfmt_misc can also recognise a filename extension aka `.com` or `.exe`.

First you must mount binfmt_misc:

```
mount binfmt_misc -t binfmt_misc /proc/sys/fs/binfmt_misc
```

To actually register a new binary type, you have to set up a string looking like `:name:type:offset:magic:mask:interpreter:flags` (where you can choose the `:` upon your needs) and echo it to `/proc/sys/fs/binfmt_misc/register`.

Here is what the fields mean:

- **`name`**
  is an identifier string. A new /proc file will be created with this name below `/proc/sys/fs/binfmt_misc`; cannot contain slashes `/` for obvious reasons.
- **`type`**
  is the type of recognition. Give `M` for magic and `E` for extension.
- **`offset`**
  is the offset of the magic/mask in the file, counted in bytes. This defaults to 0 if you omit it (i.e. you write `:name:type::magic...`). Ignored when using filename extension matching.
- **`magic`**
  is the byte sequence binfmt_misc is matching for. The magic string may contain hex-encoded characters like `\x0a` or `\xA4`. Note that you must escape any NUL bytes; parsing halts at the first one. In a shell environment you might have to write `\\x0a` to prevent the shell from eating your `\`. If you chose filename extension matching, this is the extension to be recognised (without the `.`, the `\x0a` specials are not allowed). Extension matching is case sensitive, and slashes `/` are not allowed!
- **`mask`**
  is an (optional, defaults to all 0xff) mask. You can mask out some bits from matching by supplying a string like magic and as long as magic. The mask is anded with the

byte sequence of the file. Note that you must escape any NUL bytes; parsing halts at the first one. Ignored when using filename extension matching.

- **`interpreter`**
  is the program that should be invoked with the binary as first argument (specify the full path)
- **`flags`**
  is an optional field that controls several aspects of the invocation of the interpreter. It is a string of capital letters, each controls a certain aspect. The following flags are supported:

  `P` - preserve-argv[0]
  Legacy behavior of binfmt_misc is to overwrite the original argv[0] with the full path to the binary. When this flag is included, binfmt_misc will add an argument to the argument vector for this purpose, thus preserving the original `argv[0]`. e.g. If your interp is set to `/bin/foo` and you run `blah` (which is in `/usr/local/bin`), then the kernel will execute `/bin/foo` with `argv[]` set to `["/bin/foo", "/usr/local/bin/blah", "blah"]`. The interp has to be aware of this so it can execute `/usr/local/bin/blah` with `argv[]` set to `["blah"]`.

  `O` - open-binary
  Legacy behavior of binfmt_misc is to pass the full path of the binary to the interpreter as an argument. When this flag is included, binfmt_misc will open the file for reading and pass its descriptor as an argument, instead of the full path, thus allowing the interpreter to execute non-readable binaries. This feature should be used with care - the interpreter has to be trusted not to emit the contents of the non-readable binary.

  `C` - credentials
  Currently, the behavior of binfmt_misc is to calculate the credentials and security token of the new process according to the interpreter. When this flag is included, these attributes are calculated according to the binary. It also implies the `O` flag. This feature should be used with care as the interpreter will run with root permissions when a setuid binary owned by root is run with binfmt_misc.

  `F` - fix binary
  The usual behaviour of binfmt_misc is to spawn the binary lazily when the misc format file is invoked. However, this doesn't work very well in the face of mount namespaces and changeroots, so the `F` mode opens the binary as soon as the emulation is installed and uses the opened image to spawn the emulator, meaning it is always available once installed, regardless of how the environment changes.

There are some restrictions:

- the whole register string may not exceed 1920 characters
- the magic must reside in the first 128 bytes of the file, i.e. offset+size(magic) has to be less than 128
- the interpreter string may not exceed 127 characters

To use binfmt_misc you have to mount it first. You can mount it with `mount -t binfmt_misc`

`none /proc/sys/fs/binfmt_misc` command, or you can add a line `none /proc/sys/fs/binfmt_misc binfmt_misc defaults 0 0` to your `/etc/fstab` so it auto mounts on boot.

You may want to add the binary formats in one of your `/etc/rc` scripts during boot-up. Read the manual of your init program to figure out how to do this right.

Think about the order of adding entries! Later added entries are matched first!

A few examples (assumed you are in `/proc/sys/fs/binfmt_misc`):

- enable support for em86 (like binfmt_em86, for Alpha AXP only):

```
echo ':i386:M::\x7fELF\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x02\
↪x00\x03:\xff\xff\xff\xff\xff\xfe\xfe\xff\xff\xff\xff\xff\xff\xff\xff\xff\
↪xfb\xff\xff:/bin/em86:' > register
echo ':i486:M::\x7fELF\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x02\
↪x00\x06:\xff\xff\xff\xff\xff\xfe\xfe\xff\xff\xff\xff\xff\xff\xff\xff\xff\
↪xfb\xff\xff:/bin/em86:' > register
```

- enable support for packed DOS applications (pre-configured dosemu hdimages):

```
echo ':DEXE:M::\x0eDEX::/usr/bin/dosexec:' > register
```

- enable support for Windows executables using wine:

```
echo ':DOSWin:M::MZ::/usr/local/bin/wine:' > register
```

For java support see *Java(tm) Binary Kernel Support for Linux v1.03*

You can enable/disable binfmt_misc or one binary type by echoing 0 (to disable) or 1 (to enable) to `/proc/sys/fs/binfmt_misc/status` or `/proc/.../the_name`. Catting the file tells you the current status of `binfmt_misc/the_entry`.

You can remove one entry or all entries by echoing -1 to `/proc/.../the_name` or `/proc/sys/fs/binfmt_misc/status`.

27.1 Hints

If you want to pass special arguments to your interpreter, you can write a wrapper script for it. See *Documentation/admin-guide/java.rst* for an example.

Your interpreter should NOT look in the PATH for the filename; the kernel passes it the full filename (or the file descriptor) to use. Using `$PATH` can cause unexpected behaviour and can be a security hazard.

Richard Günther <rguenth@tat.physik.uni-tuebingen.de>

CHAPTER

TWENTYEIGHT

BLOCK DEVICES

28.1 Floppy Driver

28.1.1 FAQ list:

A FAQ list may be found in the fdutils package (see below), and also at <https://fdutils.linux.lu/faq.html>.

28.1.2 LILO configuration options (Thinkpad users, read this)

The floppy driver is configured using the 'floppy=' option in lilo. This option can be typed at the boot prompt, or entered in the lilo configuration file.

Example: If your kernel is called linux-2.6.9, type the following line at the lilo boot prompt (if you have a thinkpad):

```
linux-2.6.9 floppy=thinkpad
```

You may also enter the following line in /etc/lilo.conf, in the description of linux-2.6.9:

```
append = "floppy=thinkpad"
```

Several floppy related options may be given, example:

```
linux-2.6.9 floppy=daring floppy=two_fdc
append = "floppy=daring floppy=two_fdc"
```

If you give options both in the lilo config file and on the boot prompt, the option strings of both places are concatenated, the boot prompt options coming last. That's why there are also options to restore the default behavior.

28.1.3 Module configuration options

If you use the floppy driver as a module, use the following syntax:

```
modprobe floppy floppy="<options>"
```

Example:

```
modprobe floppy floppy="omnibook messages"
```

If you need certain options enabled every time you load the floppy driver, you can put:

```
options floppy floppy="omnibook messages"
```

in a configuration file in /etc/modprobe.d/.

The floppy driver related options are:

floppy=asus_pci
Sets the bit mask to allow only units 0 and 1. (default)

floppy=daring
Tells the floppy driver that you have a well behaved floppy controller. This allows more efficient and smoother operation, but may fail on certain controllers. This may speed up certain operations.

floppy=0,daring
Tells the floppy driver that your floppy controller should be used with caution.

floppy=one_fdc
Tells the floppy driver that you have only one floppy controller. (default)

floppy=two_fdc / floppy=<address>,two_fdc
Tells the floppy driver that you have two floppy controllers. The second floppy controller is assumed to be at <address>. This option is not needed if the second controller is at address 0x370, and if you use the 'cmos' option.

floppy=thinkpad
Tells the floppy driver that you have a Thinkpad. Thinkpads use an inverted convention for the disk change line.

floppy=0,thinkpad
Tells the floppy driver that you don't have a Thinkpad.

floppy=omnibook / floppy=nodma
Tells the floppy driver not to use Dma for data transfers. This is needed on HP Omnibooks, which don't have a workable DMA channel for the floppy driver. This option is also useful if you frequently get ”Unable to allocate DMA memory” messages. Indeed, dma memory needs to be continuous in physical memory, and is thus harder to find, whereas non-dma buffers may be allocated in virtual memory. However, I advise against this if you have an FDC without a FIFO (8272A or 82072). 82072A and later are OK. You also need at least a 486 to use nodma. If you use nodma mode, I suggest you also set the FIFO threshold to 10 or lower, in order to limit the number of data transfer interrupts.

If you have a FIFO-able FDC, the floppy driver automatically falls back on non DMA mode if no DMA-able memory can be found. If you want to avoid this,

explicitly ask for 'yesdma'.

floppy=yesdma
Tells the floppy driver that a workable DMA channel is available. (default)

floppy=nofifo
Disables the FIFO entirely. This is needed if you get "Bus master arbitration error" messages from your Ethernet card (or from other devices) while accessing the floppy.

floppy=usefifo
Enables the FIFO. (default)

floppy=<threshold>,fifo_depth
Sets the FIFO threshold. This is mostly relevant in DMA mode. If this is higher, the floppy driver tolerates more interrupt latency, but it triggers more interrupts (i.e. it imposes more load on the rest of the system). If this is lower, the interrupt latency should be lower too (faster processor). The benefit of a lower threshold is less interrupts.

To tune the fifo threshold, switch on over/underrun messages using 'floppy-control --messages'. Then access a floppy disk. If you get a huge amount of "Over/Underrun - retrying" messages, then the fifo threshold is too low. Try with a higher value, until you only get an occasional Over/Underrun. It is a good idea to compile the floppy driver as a module when doing this tuning. Indeed, it allows to try different fifo values without rebooting the machine for each test. Note that you need to do 'floppycontrol --messages' every time you re-insert the module.

Usually, tuning the fifo threshold should not be needed, as the default (0xa) is reasonable.

floppy=<drive>,<type>,cmos
Sets the CMOS type of <drive> to <type>. This is mandatory if you have more than two floppy drives (only two can be described in the physical CMOS), or if your BIOS uses non-standard CMOS types. The CMOS types are:

0	Use the value of the physical CMOS
1	5 1/4 DD
2	5 1/4 HD
3	3 1/2 DD
4	3 1/2 HD
5	3 1/2 ED
6	3 1/2 ED
16	unknown or not installed

(Note: there are two valid types for ED drives. This is because 5 was initially chosen to represent floppy *tapes*, and 6 for ED drives. AMI ignored this, and used 5 for ED drives. That's why the floppy driver handles both.)

floppy=unexpected_interrupts
Print a warning message when an unexpected interrupt is received. (default)

floppy=no_unexpected_interrupts / floppy=L40SX
Don't print a message when an unexpected interrupt is received. This is needed on IBM L40SX laptops in certain video modes. (There seems to be an interaction

between video and floppy. The unexpected interrupts affect only performance, and can be safely ignored.)

floppy=broken_dcl
Don't use the disk change line, but assume that the disk was changed whenever the device node is reopened. Needed on some boxes where the disk change line is broken or unsupported. This should be regarded as a stopgap measure, indeed it makes floppy operation less efficient due to unneeded cache flushings, and slightly more unreliable. Please verify your cable, connection and jumper settings if you have any DCL problems. However, some older drives, and also some laptops are known not to have a DCL.

floppy=debug
Print debugging messages.

floppy=messages
Print informational messages for some operations (disk change notifications, warnings about over and underruns, and about autodetection).

floppy=silent_dcl_clear
Uses a less noisy way to clear the disk change line (which doesn't involve seeks). Implied by 'daring' option.

floppy=<nr>,irq
Sets the floppy IRQ to <nr> instead of 6.

floppy=<nr>,dma
Sets the floppy DMA channel to <nr> instead of 2.

floppy=slow
Use PS/2 stepping rate:

```
PS/2 floppies have much slower step rates than regular floppies.
It's been recommended that take about 1/4 of the default speed
in some more extreme cases.
```

28.1.4 Supporting utilities and additional documentation:

Additional parameters of the floppy driver can be configured at runtime. Utilities which do this can be found in the fdutils package. This package also contains a new version of mtools which allows to access high capacity disks (up to 1992K on a high density 3 1/2 disk!). It also contains additional documentation about the floppy driver.

The latest version can be found at fdutils homepage:

https://fdutils.linux.lu

The fdutils releases can be found at:

https://fdutils.linux.lu/download.html

http://www.tux.org/pub/knaff/fdutils/

ftp://metalab.unc.edu/pub/Linux/utils/disk-management/

28.1.5 Reporting problems about the floppy driver

If you have a question or a bug report about the floppy driver, mail me at Alain.Knaff@poboxes.com . If you post to Usenet, preferably use comp.os.linux.hardware. As the volume in these groups is rather high, be sure to include the word "floppy" (or "FLOPPY") in the subject line. If the reported problem happens when mounting floppy disks, be sure to mention also the type of the filesystem in the subject line.

Be sure to read the FAQ before mailing/posting any bug reports!

Alain

28.1.6 Changelog

10-30-2004 :
Cleanup, updating, add reference to module configuration. James Nelson <james4765@gmail.com>

6-3-2000 :
Original Document

28.2 Network Block Device (TCP version)

28.2.1 1) Overview

What is it: With this compiled in the kernel (or as a module), Linux can use a remote server as one of its block devices. So every time the client computer wants to read, e.g., /dev/nb0, it sends a request over TCP to the server, which will reply with the data read. This can be used for stations with low disk space (or even diskless) to borrow disk space from another computer. Unlike NFS, it is possible to put any filesystem on it, etc.

For more information, or to download the nbd-client and nbd-server tools, go to https://github.com/NetworkBlockDevice/nbd.

The nbd kernel module need only be installed on the client system, as the nbd-server is completely in userspace. In fact, the nbd-server has been successfully ported to other operating systems, including Windows.

28.2.2 A) NBD parameters

max_part
Number of partitions per device (default: 0).

nbds_max
Number of block devices that should be initialized (default: 16).

28.3 Linux and parallel port IDE devices

PARIDE v1.03 (c) 1997-8 Grant Guenther <grant@torque.net> PATA_PARPORT (c) 2023 Ondrej Zary

28.3.1 1. Introduction

Owing to the simplicity and near universality of the parallel port interface to personal computers, many external devices such as portable hard-disk, CD-ROM, LS-120 and tape drives use the parallel port to connect to their host computer. While some devices (notably scanners) use ad-hoc methods to pass commands and data through the parallel port interface, most external devices are actually identical to an internal model, but with a parallel-port adapter chip added in. Some of the original parallel port adapters were little more than mechanisms for multiplexing a SCSI bus. (The Iomega PPA-3 adapter used in the ZIP drives is an example of this approach). Most current designs, however, take a different approach. The adapter chip reproduces a small ISA or IDE bus in the external device and the communication protocol provides operations for reading and writing device registers, as well as data block transfer functions. Sometimes, the device being addressed via the parallel cable is a standard SCSI controller like an NCR 5380. The ”ditto” family of external tape drives use the ISA replicator to interface a floppy disk controller, which is then connected to a floppy-tape mechanism. The vast majority of external parallel port devices, however, are now based on standard IDE type devices, which require no intermediate controller. If one were to open up a parallel port CD-ROM drive, for instance, one would find a standard ATAPI CD-ROM drive, a power supply, and a single adapter that interconnected a standard PC parallel port cable and a standard IDE cable. It is usually possible to exchange the CD-ROM device with any other device using the IDE interface.

The document describes the support in Linux for parallel port IDE devices. It does not cover parallel port SCSI devices, ”ditto” tape drives or scanners. Many different devices are supported by the parallel port IDE subsystem, including:

- MicroSolutions backpack CD-ROM
- MicroSolutions backpack PD/CD
- MicroSolutions backpack hard-drives
- MicroSolutions backpack 8000t tape drive
- SyQuest EZ-135, EZ-230 & SparQ drives
- Avatar Shark
- Imation Superdisk LS-120
- Maxell Superdisk LS-120
- FreeCom Power CD
- Hewlett-Packard 5GB and 8GB tape drives
- Hewlett-Packard 7100 and 7200 CD-RW drives

as well as most of the clone and no-name products on the market.

To support such a wide range of devices, pata_parport is actually structured in two parts. There is a base pata_parport module which provides an interface to kernel libata subsystem, registry and some common methods for accessing the parallel ports.

The second component is a set of low-level protocol drivers for each of the parallel port IDE adapter chips. Thanks to the interest and encouragement of Linux users from many parts of the world, support is available for almost all known adapter protocols:

aten	ATEN EH-100	(HK)
bpck	Microsolutions backpack	(US)
comm	DataStor (old-type) "commuter" adapter	(TW)
dstr	DataStor EP-2000	(TW)
epat	Shuttle EPAT	(UK)
epia	Shuttle EPIA	(UK)
fit2	FIT TD-2000	(US)
fit3	FIT TD-3000	(US)
friq	Freecom IQ cable	(DE)
frpw	Freecom Power	(DE)
kbic	KingByte KBIC-951A and KBIC-971A	(TW)
ktti	KT Technology PHd adapter	(SG)
on20	OnSpec 90c20	(US)
on26	OnSpec 90c26	(US)

28.3.2 2. Using pata_parport subsystem

While configuring the Linux kernel, you may choose either to build the pata_parport drivers into your kernel, or to build them as modules.

In either case, you will need to select "Parallel port IDE device support" and at least one of the parallel port communication protocols. If you do not know what kind of parallel port adapter is used in your drive, you could begin by checking the file names and any text files on your DOS installation floppy. Alternatively, you can look at the markings on the adapter chip itself. That's usually sufficient to identify the correct device.

You can actually select all the protocol modules, and allow the pata_parport subsystem to try them all for you.

For the "brand-name" products listed above, here are the protocol and high-level drivers that you would use:

Manufacturer	Model	Protocol
MicroSolutions	CD-ROM	bpck
MicroSolutions	PD drive	bpck
MicroSolutions	hard-drive	bpck
MicroSolutions	8000t tape	bpck
SyQuest	EZ, SparQ	epat
Imation	Superdisk	epat
Maxell	Superdisk	friq
Avatar	Shark	epat
FreeCom	CD-ROM	frpw
Hewlett-Packard	5GB Tape	epat
Hewlett-Packard	7200e (CD)	epat
Hewlett-Packard	7200e (CD-R)	epat

All parports and all protocol drivers are probed automatically unless probe=0 parameter is used. So just ”modprobe epat” is enough for a Imation SuperDisk drive to work.

Manual device creation:

```
# echo "port protocol mode unit delay" >/sys/bus/pata_parport/new_device
```

where:

port	parport name (or ”auto” for all parports)
protocol	protocol name (or ”auto” for all protocols)
mode	mode number (protocol-specific) or -1 for probe
unit	unit number (for backpack only, see below)
delay	I/O delay (see troubleshooting section below)

If you happen to be using a MicroSolutions backpack device, you will also need to know the unit ID number for each drive. This is usually the last two digits of the drive's serial number (but read MicroSolutions' documentation about this).

If you omit the parameters from the end, defaults will be used, e.g.:

Probe all parports with all protocols:

```
# echo auto >/sys/bus/pata_parport/new_device
```

Probe parport0 using protocol epat and mode 4 (EPP-16):

```
# echo "parport0 epat 4" >/sys/bus/pata_parport/new_device
```

Probe parport0 using all protocols:

```
# echo "parport0 auto" >/sys/bus/pata_parport/new_device
```

Probe all parports using protoocol epat:

```
# echo "auto epat" >/sys/bus/pata_parport/new_device
```

Deleting devices:

```
# echo pata_parport.0 >/sys/bus/pata_parport/delete_device
```

28.3.3 3. Troubleshooting

3.1 Use EPP mode if you can

The most common problems that people report with the pata_parport drivers concern the parallel port CMOS settings. At this time, none of the protocol modules support ECP mode, or any ECP combination modes. If you are able to do so, please set your parallel port into EPP mode using your CMOS setup procedure.

3.2 Check the port delay

Some parallel ports cannot reliably transfer data at full speed. To offset the errors, the protocol modules introduce a ”port delay” between each access to the i/o ports. Each protocol sets a default value for this delay. In most cases, the user can override the default and set it to 0 - resulting in somewhat higher transfer rates. In some rare cases (especially with older 486 systems) the default delays are not long enough. if you experience corrupt data transfers, or unexpected failures, you may wish to increase the port delay.

3.3 Some drives need a printer reset

There appear to be a number of ”noname” external drives on the market that do not always power up correctly. We have noticed this with some drives based on OnSpec and older Freecom adapters. In these rare cases, the adapter can often be reinitialised by issuing a ”printer reset” on the parallel port. As the reset operation is potentially disruptive in multiple device environments, the pata_parport drivers will not do it automatically. You can however, force a printer reset by doing:

```
insmod lp reset=1
rmmod lp
```

If you have one of these marginal cases, you should probably build your pata_parport drivers as modules, and arrange to do the printer reset before loading the pata_parport drivers.

28.4 Using the RAM disk block device with Linux

28.4.1 1) Overview

The RAM disk driver is a way to use main system memory as a block device. It is required for initrd, an initial filesystem used if you need to load modules in order to access the root filesystem (see *Using the initial RAM disk (initrd)*). It can also be used for a temporary filesystem for crypto work, since the contents are erased on reboot.

The RAM disk dynamically grows as more space is required. It does this by using RAM from the buffer cache. The driver marks the buffers it is using as dirty so that the VM subsystem does not try to reclaim them later.

The RAM disk supports up to 16 RAM disks by default, and can be reconfigured to support an unlimited number of RAM disks (at your own risk). Just change the configuration symbol BLK_DEV_RAM_COUNT in the Block drivers config menu and (re)build the kernel.

To use RAM disk support with your system, run './MAKEDEV ram' from the /dev directory. RAM disks are all major number 1, and start with minor number 0 for /dev/ram0, etc. If used, modern kernels use /dev/ram0 for an initrd.

The new RAM disk also has the ability to load compressed RAM disk images, allowing one to squeeze more programs onto an average installation or rescue floppy disk.

28.4.2 2) Parameters

2a) Kernel Command Line Parameters

ramdisk_size=N
Size of the ramdisk.

This parameter tells the RAM disk driver to set up RAM disks of N k size. The default is 4096 (4 MB).

2b) Module parameters

rd_nr
/dev/ramX devices created.

max_part
Maximum partition number.

rd_size
See ramdisk_size.

28.4.3 3) Using "rdev"

"rdev" is an obsolete, deprecated, antiquated utility that could be used to set the boot device in a Linux kernel image.

Instead of using rdev, just place the boot device information on the kernel command line and pass it to the kernel from the bootloader.

You can also pass arguments to the kernel by setting FDARGS in arch/x86/boot/Makefile and specify in initrd image by setting FDINITRD in arch/x86/boot/Makefile.

Some of the kernel command line boot options that may apply here are:

```
ramdisk_start=N
ramdisk_size=M
```

If you make a boot disk that has LILO, then for the above, you would use:

```
append = "ramdisk_start=N ramdisk_size=M"
```

28.4.4 4) An Example of Creating a Compressed RAM Disk

To create a RAM disk image, you will need a spare block device to construct it on. This can be the RAM disk device itself, or an unused disk partition (such as an unmounted swap partition). For this example, we will use the RAM disk device, "/dev/ram0".

Note: This technique should not be done on a machine with less than 8 MB of RAM. If using a spare disk partition instead of /dev/ram0, then this restriction does not apply.

a) Decide on the RAM disk size that you want. Say 2 MB for this example. Create it by writing to the RAM disk device. (This step is not currently required, but may be in the future.) It is wise to zero out the area (esp. for disks) so that maximal compression is achieved for the unused blocks of the image that you are about to create:

```
dd if=/dev/zero of=/dev/ram0 bs=1k count=2048
```

b) Make a filesystem on it. Say ext2fs for this example:

```
mke2fs -vm0 /dev/ram0 2048
```

c) Mount it, copy the files you want to it (eg: /etc/* /dev/* ...) and unmount it again.

d) Compress the contents of the RAM disk. The level of compression will be approximately 50% of the space used by the files. Unused space on the RAM disk will compress to almost nothing:

```
dd if=/dev/ram0 bs=1k count=2048 | gzip -v9 > /tmp/ram_image.gz
```

e) Put the kernel onto the floppy:

```
dd if=zImage of=/dev/fd0 bs=1k
```

f) Put the RAM disk image onto the floppy, after the kernel. Use an offset that is slightly larger than the kernel, so that you can put another (possibly larger) kernel onto the same floppy later without overlapping the RAM disk image. An offset of 400 kB for kernels about 350 kB in size would be reasonable. Make sure offset+size of ram_image.gz is not larger than the total space on your floppy (usually 1440 kB):

```
dd if=/tmp/ram_image.gz of=/dev/fd0 bs=1k seek=400
```

g) Make sure that you have already specified the boot information in FDARGS and FDINITRD or that you use a bootloader to pass kernel command line boot options to the kernel.

That is it. You now have your boot/root compressed RAM disk floppy. Some users may wish to combine steps (d) and (f) by using a pipe.

Paul Gortmaker 12/95

28.4.5 Changelog:

SEPT-2020 :

Removed usage of "rdev"

10-22-04 :
Updated to reflect changes in command line options, remove obsolete references, general cleanup. James Nelson (james4765@gmail.com)

12-95 :
Original Document

28.5 zram: Compressed RAM-based block devices

28.5.1 Introduction

The zram module creates RAM-based block devices named /dev/zram<id> (<id> = 0, 1, ...). Pages written to these disks are compressed and stored in memory itself. These disks allow very fast I/O and compression provides good amounts of memory savings. Some of the use cases include /tmp storage, use as swap disks, various caches under /var and maybe many more. :)

Statistics for individual zram devices are exported through sysfs nodes at /sys/block/zram<id>/

28.5.2 Usage

There are several ways to configure and manage zram device(-s):

a) using zram and zram_control sysfs attributes

b) using zramctl utility, provided by util-linux (util-linux@vger.kernel.org).

In this document we will describe only 'manual' zram configuration steps, IOW, zram and zram_control sysfs attributes.

In order to get a better idea about zramctl please consult util-linux documentation, zramctl man-page or *zramctl --help*. Please be informed that zram maintainers do not develop/maintain util-linux or zramctl, should you have any questions please contact util-linux@vger.kernel.org

Following shows a typical sequence of steps for using zram.

28.5.3 WARNING

For the sake of simplicity we skip error checking parts in most of the examples below. However, it is your sole responsibility to handle errors.

zram sysfs attributes always return negative values in case of errors. The list of possible return codes:

-EBUSY	an attempt to modify an attribute that cannot be changed once the device has been initialised. Please reset device first.
-ENOMEM	zram was not able to allocate enough memory to fulfil your needs.
-EINVAL	invalid input has been provided.

If you use 'echo', the returned value is set by the 'echo' utility, and, in general case, something like:

```
echo 3 > /sys/block/zram0/max_comp_streams
if [ $? -ne 0 ]; then
        handle_error
fi
```

should suffice.

28.5.4 1) Load Module

```
modprobe zram num_devices=4
```

This creates 4 devices: /dev/zram{0,1,2,3}

num_devices parameter is optional and tells zram how many devices should be pre-created. Default: 1.

28.5.5 2) Set max number of compression streams

Regardless of the value passed to this attribute, ZRAM will always allocate multiple compression streams - one per online CPU - thus allowing several concurrent compression operations. The number of allocated compression streams goes down when some of the CPUs become offline. There is no single-compression-stream mode anymore, unless you are running a UP system or have only 1 CPU online.

To find out how many streams are currently available:

```
cat /sys/block/zram0/max_comp_streams
```

28.5.6 3) Select compression algorithm

Using comp_algorithm device attribute one can see available and currently selected (shown in square brackets) compression algorithms, or change the selected compression algorithm (once the device is initialised there is no way to change compression algorithm).

Examples:

```
#show supported compression algorithms
cat /sys/block/zram0/comp_algorithm
lzo [lz4]

#select lzo compression algorithm
echo lzo > /sys/block/zram0/comp_algorithm
```

For the time being, the *comp_algorithm* content does not necessarily show every compression algorithm supported by the kernel. We keep this list primarily to simplify device configuration and one can configure a new device with a compression algorithm that is not listed in *comp_algorithm*. The thing is that, internally, ZRAM uses Crypto API and, if some of the algorithms were built as modules, it's impossible to list all of them using, for instance, /proc/crypto or any other method. This, however, has an advantage of permitting the usage of custom crypto compression modules (implementing S/W or H/W compression).

28.5.7 4) Set Disksize

Set disk size by writing the value to sysfs node 'disksize'. The value can be either in bytes or you can use mem suffixes. Examples:

```
# Initialize /dev/zram0 with 50MB disksize
echo $((50*1024*1024)) > /sys/block/zram0/disksize

# Using mem suffixes
echo 256K > /sys/block/zram0/disksize
echo 512M > /sys/block/zram0/disksize
echo 1G > /sys/block/zram0/disksize
```

Note: There is little point creating a zram of greater than twice the size of memory since we expect a 2:1 compression ratio. Note that zram uses about 0.1% of the size of the disk when not in use so a huge zram is wasteful.

28.5.8 5) Set memory limit: Optional

Set memory limit by writing the value to sysfs node 'mem_limit'. The value can be either in bytes or you can use mem suffixes. In addition, you could change the value in runtime. Examples:

```
# limit /dev/zram0 with 50MB memory
echo $((50*1024*1024)) > /sys/block/zram0/mem_limit

# Using mem suffixes
echo 256K > /sys/block/zram0/mem_limit
echo 512M > /sys/block/zram0/mem_limit
echo 1G > /sys/block/zram0/mem_limit

# To disable memory limit
echo 0 > /sys/block/zram0/mem_limit
```

28.5.9 6) Activate

```
mkswap /dev/zram0
swapon /dev/zram0

mkfs.ext4 /dev/zram1
mount /dev/zram1 /tmp
```

28.5.10 7) Add/remove zram devices

zram provides a control interface, which enables dynamic (on-demand) device addition and removal.

In order to add a new /dev/zramX device, perform a read operation on the hot_add attribute. This will return either the new device's device id (meaning that you can use /dev/zram<id>) or an error code.

Example:

```
cat /sys/class/zram-control/hot_add
1
```

To remove the existing /dev/zramX device (where X is a device id) execute:

```
echo X > /sys/class/zram-control/hot_remove
```

28.5.11 8) Stats

Per-device statistics are exported as various nodes under /sys/block/zram<id>/

A brief description of exported device attributes follows. For more details please read Documentation/ABI/testing/sysfs-block-zram.

Name	access	description
disksize	RW	show and set the device's disk size
initstate	RO	shows the initialization state of the device
reset	WO	trigger device reset
mem_used_max	WO	reset the *mem_used_max* counter (see later)
mem_limit	WO	specifies the maximum amount of memory ZRAM can use to store the compressed data
writeback_limit	WO	specifies the maximum amount of write IO zram can write out to backing device as 4KB unit
writeback_limit_enable	RW	show and set writeback_limit feature
max_comp_streams	RW	the number of possible concurrent compress operations
comp_algorithm	RW	show and change the compression algorithm
compact	WO	trigger memory compaction
debug_stat	RO	this file is used for zram debugging purposes
backing_dev	RW	set up backend storage for zram to write out
idle	WO	mark allocated slot as idle

User space is advised to use the following files to read the device statistics.

File /sys/block/zram<id>/stat

Represents block layer statistics. Read Documentation/block/stat.rst for details.

File /sys/block/zram<id>/io_stat

The stat file represents device's I/O statistics not accounted by block layer and, thus, not available in zram<id>/stat file. It consists of a single line of text and contains the following stats separated by whitespace:

failed_reads	The number of failed reads
failed_writes	The number of failed writes
invalid_io	The number of non-page-size-aligned I/O requests
notify_free	Depending on device usage scenario it may account a) the number of pages freed because of swap slot free notifications b) the number of pages freed because of REQ_OP_DISCARD requests sent by bio. The former ones are sent to a swap block device when a swap slot is freed, which implies that this disk is being used as a swap disk. The latter ones are sent by filesystem mounted with discard option, whenever some data blocks are getting discarded.

File /sys/block/zram<id>/mm_stat

The mm_stat file represents the device's mm statistics. It consists of a single line of text and contains the following stats separated by whitespace:

orig_data_size	uncompressed size of data stored in this disk. Unit: bytes
compr_data_size	compressed size of data stored in this disk
mem_used_total	the amount of memory allocated for this disk. This includes allocator fragmentation and metadata overhead, allocated for this disk. So, allocator space efficiency can be calculated using compr_data_size and this statistic. Unit: bytes
mem_limit	the maximum amount of memory ZRAM can use to store the compressed data
mem_used_max	the maximum amount of memory zram has consumed to store the data
same_pages	the number of same element filled pages written to this disk. No memory is allocated for such pages.
pages_compacted	the number of pages freed during compaction
huge_pages	the number of incompressible pages
huge_pages_since	the number of incompressible pages since zram set up

File /sys/block/zram<id>/bd_stat

The bd_stat file represents a device's backing device statistics. It consists of a single line of text and contains the following stats separated by whitespace:

bd_count	size of data written in backing device. Unit: 4K bytes
bd_reads	the number of reads from backing device Unit: 4K bytes
bd_writes	the number of writes to backing device Unit: 4K bytes

28.5.12 9) Deactivate

```
swapoff /dev/zram0
umount /dev/zram1
```

28.5.13 10) Reset

Write any positive value to 'reset' sysfs node:

```
echo 1 > /sys/block/zram0/reset
echo 1 > /sys/block/zram1/reset
```

This frees all the memory allocated for the given device and resets the disksize to zero. You must set the disksize again before reusing the device.

28.5.14 Optional Feature

writeback

With CONFIG_ZRAM_WRITEBACK, zram can write idle/incompressible page to backing storage rather than keeping it in memory. To use the feature, admin should set up backing device via:

```
echo /dev/sda5 > /sys/block/zramX/backing_dev
```

before disksize setting. It supports only partitions at this moment. If admin wants to use incompressible page writeback, they could do it via:

```
echo huge > /sys/block/zramX/writeback
```

To use idle page writeback, first, user need to declare zram pages as idle:

```
echo all > /sys/block/zramX/idle
```

From now on, any pages on zram are idle pages. The idle mark will be removed until someone requests access of the block. IOW, unless there is access request, those pages are still idle pages. Additionally, when CONFIG_ZRAM_TRACK_ENTRY_ACTIME is enabled pages can be marked as idle based on how long (in seconds) it's been since they were last accessed:

```
echo 86400 > /sys/block/zramX/idle
```

In this example all pages which haven't been accessed in more than 86400 seconds (one day) will be marked idle.

Admin can request writeback of those idle pages at right timing via:

```
echo idle > /sys/block/zramX/writeback
```

With the command, zram will writeback idle pages from memory to the storage.

Additionally, if a user choose to writeback only huge and idle pages this can be accomplished with:

```
echo huge_idle > /sys/block/zramX/writeback
```

If a user chooses to writeback only incompressible pages (pages that none of algorithms can compress) this can be accomplished with:

```
echo incompressible > /sys/block/zramX/writeback
```

If an admin wants to write a specific page in zram device to the backing device, they could write a page index into the interface:

```
echo "page_index=1251" > /sys/block/zramX/writeback
```

If there are lots of write IO with flash device, potentially, it has flash wearout problem so that admin needs to design write limitation to guarantee storage health for entire product life.

To overcome the concern, zram supports ”writeback_limit” feature. The ”writeback_limit_enable”'s default value is 0 so that it doesn't limit any writeback. IOW, if admin wants to apply writeback budget, they should enable writeback_limit_enable via:

```
$ echo 1 > /sys/block/zramX/writeback_limit_enable
```

Once writeback_limit_enable is set, zram doesn't allow any writeback until admin sets the budget via /sys/block/zramX/writeback_limit.

(If admin doesn't enable writeback_limit_enable, writeback_limit's value assigned via /sys/block/zramX/writeback_limit is meaningless.)

If admin wants to limit writeback as per-day 400M, they could do it like below:

```
$ MB_SHIFT=20
$ 4K_SHIFT=12
$ echo $((400<<MB_SHIFT>>4K_SHIFT)) > \
        /sys/block/zram0/writeback_limit.
$ echo 1 > /sys/block/zram0/writeback_limit_enable
```

If admins want to allow further write again once the budget is exhausted, they could do it like below:

```
$ echo $((400<<MB_SHIFT>>4K_SHIFT)) > \
        /sys/block/zram0/writeback_limit
```

If an admin wants to see the remaining writeback budget since last set:

```
$ cat /sys/block/zramX/writeback_limit
```

If an admin wants to disable writeback limit, they could do:

```
$ echo 0 > /sys/block/zramX/writeback_limit_enable
```

The writeback_limit count will reset whenever you reset zram (e.g., system reboot, echo 1 > /sys/block/zramX/reset) so keeping how many of writeback happened until you reset the zram to allocate extra writeback budget in next setting is user's job.

If admin wants to measure writeback count in a certain period, they could know it via /sys/block/zram0/bd_stat's 3rd column.

recompression

With CONFIG_ZRAM_MULTI_COMP, zram can recompress pages using alternative (secondary) compression algorithms. The basic idea is that alternative compression algorithm can provide better compression ratio at a price of (potentially) slower compression/decompression speeds. Alternative compression algorithm can, for example, be more successful compressing huge pages (those that default algorithm failed to compress). Another application is idle pages recompression - pages that are cold and sit in the memory can be recompressed using more effective algorithm and, hence, reduce zsmalloc memory usage.

With CONFIG_ZRAM_MULTI_COMP, zram supports up to 4 compression algorithms: one primary and up to 3 secondary ones. Primary zram compressor is explained in "3) Select compression algorithm", secondary algorithms are configured using recomp_algorithm device attribute.

Example::

```
#show supported recompression algorithms
cat /sys/block/zramX/recomp_algorithm
#1: lzo lzo-rle lz4 lz4hc [zstd]
#2: lzo lzo-rle lz4 [lz4hc] zstd
```

Alternative compression algorithms are sorted by priority. In the example above, zstd is used as the first alternative algorithm, which has priority of 1, while lz4hc is configured as a compression algorithm with priority 2. Alternative compression algorithm's priority is provided during algorithms configuration::

```
#select zstd recompression algorithm, priority 1
echo "algo=zstd priority=1" > /sys/block/zramX/recomp_algorithm

#select deflate recompression algorithm, priority 2
echo "algo=deflate priority=2" > /sys/block/zramX/recomp_algorithm
```

Another device attribute that CONFIG_ZRAM_MULTI_COMP enables is recompress, which controls recompression.

Examples::

```
#IDLE pages recompression is activated by `idle` mode
echo "type=idle" > /sys/block/zramX/recompress

#HUGE pages recompression is activated by `huge` mode
echo "type=huge" > /sys/block/zram0/recompress

#HUGE_IDLE pages recompression is activated by `huge_idle` mode
echo "type=huge_idle" > /sys/block/zramX/recompress
```

The number of idle pages can be significant, so user-space can pass a size threshold (in bytes) to the recompress knob: zram will recompress only pages of equal or greater size::

```
#recompress all pages larger than 3000 bytes
echo "threshold=3000" > /sys/block/zramX/recompress

#recompress idle pages larger than 2000 bytes
echo "type=idle threshold=2000" > /sys/block/zramX/recompress
```

Recompression of idle pages requires memory tracking.

During re-compression for every page, that matches re-compression criteria, ZRAM iterates the list of registered alternative compression algorithms in order of their priorities. ZRAM stops either when re-compression was successful (re-compressed object is smaller in size than the original one) and matches re-compression criteria (e.g. size threshold) or when there are no secondary algorithms left to try. If none of the secondary algorithms can successfully re-compressed the page such a page is marked as incompressible, so ZRAM will not attempt to re-compress it in the future.

This re-compression behaviour, when it iterates through the list of registered compression algorithms, increases our chances of finding the algorithm that successfully compresses a particular page. Sometimes, however, it is convenient (and sometimes even necessary) to limit recompression to only one particular algorithm so that it will not try any other algorithms. This can be achieved by providing a algo=NAME parameter::

```
#use zstd algorithm only (if registered)
echo "type=huge algo=zstd" > /sys/block/zramX/recompress
```

28.5.15 memory tracking

With CONFIG_ZRAM_MEMORY_TRACKING, user can know information of the zram block. It could be useful to catch cold or incompressible pages of the process with*pagemap.

If you enable the feature, you could see block state via /sys/kernel/debug/zram/zram0/block_state". The output is as follows:

```
300    75.033841 .wh...
301    63.806904 s.....
302    63.806919 ..hi..
303    62.801919 ....r.
304   146.781902 ..hi.n
```

First column
zram's block index.

Second column
access time since the system was booted

Third column
state of the block:

s:
same page

w:
written page to backing store

h:
huge page

i:
idle page

r:
recompressed page (secondary compression algorithm)

n:
none (including secondary) of algorithms could compress it

First line of above example says 300th block is accessed at 75.033841sec and the block's state is huge so it is written back to the backing storage. It's a debugging feature so anyone shouldn't rely on it to work properly.

Nitin Gupta ngupta@vflare.org

28.6 Distributed Replicated Block Device - DRBD

28.6.1 Description

DRBD is a shared-nothing, synchronously replicated block device. It is designed to serve as a building block for high availability clusters and in this context, is a "drop-in" replacement for shared storage. Simplistically, you could see it as a network RAID 1.

Please visit https://www.drbd.org to find out more.

kernel data structure for DRBD-9

This describes the in kernel data structure for DRBD-9. Starting with Linux v3.14 we are reorganizing DRBD to use this data structure.

Basic Data Structure

A node has a number of DRBD resources. Each such resource has a number of devices (aka volumes) and connections to other nodes ("peer nodes"). Each DRBD device is represented by a block device locally.

The DRBD objects are interconnected to form a matrix as depicted below; a drbd_peer_device object sits at each intersection between a drbd_device and a drbd_connection:

```
/--------------+---------------+.....+---------------\
|   resource   |    device     |     |    device     |
+--------------+---------------+.....+---------------+
|  connection  |  peer_device  |     |  peer_device  |
+--------------+---------------+.....+---------------+
:              :               :     :               :
:              :               :     :               :
+--------------+---------------+.....+---------------+
|  connection  |  peer_device  |     |  peer_device  |
\--------------+---------------+.....+---------------/
```

In this table, horizontally, devices can be accessed from resources by their volume number. Likewise, peer_devices can be accessed from connections by their volume number. Objects

in the vertical direction are connected by double linked lists. There are back pointers from peer_devices to their connections a devices, and from connections and devices to their resource.

All resources are in the drbd_resources double-linked list. In addition, all devices can be accessed by their minor device number via the drbd_devices idr.

The drbd_resource, drbd_connection, and drbd_device objects are reference counted. The peer_device objects only serve to establish the links between devices and connections; their lifetime is determined by the lifetime of the device and connection which they reference.

Data flows that Relate some functions, and write packets

Sub graphs of DRBD's state transitions

DRBD-8.3 data flow

Checksum based Resync, case not in sync

Checksum based Resync, case in sync

Online verify

DRBD 8 data flow
Regular mirrored write, 512-32K
al_begin_io()
drbd_make_request()
w_send_dblock()
Data
drbd_endio_write_pri()
receive_Data()
al_complete_io()
drbd_endio_write_sec()
e_end_block()
WriteAck
got_BlockAck()
Diskless read, 512-32K
drbd_make_request()
w_send_read_req()
DataRequest
receive_DataRequest()
drbd_endio_read_sec()
w_e_end_data_req()
DataReply
receive_DataReply()
Resync blocks, 4-32K
rs_begin_io()
w_make_resync_request()
RSDataRequest
receive_DataRequest()
rs_begin_io()
drbd_endio_read_sec()
w_e_end_rsdata_req()
RSDataReply
receive_RSDataReply()
rs_complete_io()
drbd_endio_write_sec()
e_end_resync_block()
WriteAck
rs_complete_io()
got_BlockAck()

StandAllone
ioctl_set_net()
checks in receive_param()
WFConnection
in connect() after accept
unable to bind()
WFReportParams
Unconnected
sync_handshake()
sync_handshake()
WFBitMapS
* on network error
WFBitMapT
receive_bitmap()
in receive_param()
receive_bitmap()
SyncSource
SyncTarget
PausedSyncS
Connected
PausedSyncT

Diskless
ioctl_set_disk()
ioctl_set_disk()
Consistent
ioctl_set_disk()
receive_param()
Outdated
start resync
start resync
receive_param()
sending notify to peer
Inconsistent
io completion error
io completion error
resync completed
ioctl_replicate
io completion error
UpToDate
io completion error
Failed

Secondary
recv state packet
recv state packet
Primary
connection lost
connected
connection lost
connected
Unknown

CHAPTER

TWENTYNINE

BOOT CONFIGURATION

Author
Masami Hiramatsu <mhiramat@kernel.org>

29.1 Overview

The boot configuration expands the current kernel command line to support additional key-value data when booting the kernel in an efficient way. This allows administrators to pass a structured-Key config file.

29.2 Config File Syntax

The boot config syntax is a simple structured key-value. Each key consists of dot-connected-words, and key and value are connected by =. The value has to be terminated by semi-colon (`;`) or newline (`\n`). For array value, array entries are separated by comma (`,`).

```
KEY[.WORD[...]] = VALUE[, VALUE2[...]][;]
```

Unlike the kernel command line syntax, spaces are OK around the comma and =.

Each key word must contain only alphabets, numbers, dash (`-`) or underscore (`_`). And each value only contains printable characters or spaces except for delimiters such as semi-colon (`;`), new-line (`\n`), comma (`,`), hash (`#`) and closing brace (`}`).

If you want to use those delimiters in a value, you can use either double- quotes (`"VALUE"`) or single-quotes (`'VALUE'`) to quote it. Note that you can not escape these quotes.

There can be a key which doesn't have value or has an empty value. Those keys are used for checking if the key exists or not (like a boolean).

29.2.1 Key-Value Syntax

The boot config file syntax allows user to merge partially same word keys by brace. For example:

```
foo.bar.baz = value1
foo.bar.qux.quux = value2
```

These can be written also in:

```
foo.bar {
   baz = value1
   qux.quux = value2
}
```

Or more shorter, written as following:

```
foo.bar { baz = value1; qux.quux = value2 }
```

In both styles, same key words are automatically merged when parsing it at boot time. So you can append similar trees or key-values.

29.2.2 Same-key Values

It is prohibited that two or more values or arrays share a same-key. For example,:

```
foo = bar, baz
foo = qux  # !ERROR! we can not re-define same key
```

If you want to update the value, you must use the override operator `:=` explicitly. For example:

```
foo = bar, baz
foo := qux
```

then, the `qux` is assigned to `foo` key. This is useful for overriding the default value by adding (partial) custom bootconfigs without parsing the default bootconfig.

If you want to append the value to existing key as an array member, you can use `+=` operator. For example:

```
foo = bar, baz
foo += qux
```

In this case, the key `foo` has `bar`, `baz` and `qux`.

Moreover, sub-keys and a value can coexist under a parent key. For example, following config is allowed.:

```
foo = value1
foo.bar = value2
foo := value3 # This will update foo's value.
```

Note, since there is no syntax to put a raw value directly under a structured key, you have to define it outside of the brace. For example:

```
foo {
    bar = value1
    bar {
        baz = value2
        qux = value3
    }
}
```

Also, the order of the value node under a key is fixed. If there are a value and subkeys, the value is always the first child node of the key. Thus if user specifies subkeys first, e.g.:

```
foo.bar = value1
foo = value2
```

In the program (and /proc/bootconfig), it will be shown as below:

```
foo = value2
foo.bar = value1
```

29.2.3 Comments

The config syntax accepts shell-script style comments. The comments starting with hash (”#”) until newline (”n”) will be ignored.

```
# comment line
foo = value # value is set to foo.
bar = 1, # 1st element
      2, # 2nd element
      3  # 3rd element
```

This is parsed as below:

```
foo = value
bar = 1, 2, 3
```

Note that you can not put a comment between value and delimiter(`,` or `;`). This means following config has a syntax error

```
key = 1 # comment
      ,2
```

29.3 /proc/bootconfig

/proc/bootconfig is a user-space interface of the boot config. Unlike /proc/cmdline, this file shows the key-value style list. Each key-value pair is shown in each line with following style:

```
KEY[.WORDS...] = "[VALUE]"[,"VALUE2"...]
```

29.4 Boot Kernel With a Boot Config

There are two options to boot the kernel with bootconfig: attaching the bootconfig to the initrd image or embedding it in the kernel itself.

29.4.1 Attaching a Boot Config to Initrd

Since the boot configuration file is loaded with initrd by default, it will be added to the end of the initrd (initramfs) image file with padding, size, checksum and 12-byte magic word as below.

[initrd][bootconfig][padding][size(le32)][checksum(le32)][#BOOTCONFIGn]

The size and checksum fields are unsigned 32bit little endian value.

When the boot configuration is added to the initrd image, the total file size is aligned to 4 bytes. To fill the gap, null characters (`\0`) will be added. Thus the `size` is the length of the bootconfig file + padding bytes.

The Linux kernel decodes the last part of the initrd image in memory to get the boot configuration data. Because of this "piggyback" method, there is no need to change or update the boot loader and the kernel image itself as long as the boot loader passes the correct initrd file size. If by any chance, the boot loader passes a longer size, the kernel fails to find the bootconfig data.

To do this operation, Linux kernel provides `bootconfig` command under tools/bootconfig, which allows admin to apply or delete the config file to/from initrd image. You can build it by the following command:

```
# make -C tools/bootconfig
```

To add your boot config file to initrd image, run bootconfig as below (Old data is removed automatically if exists):

```
# tools/bootconfig/bootconfig -a your-config /boot/initrd.img-X.Y.Z
```

To remove the config from the image, you can use -d option as below:

```
# tools/bootconfig/bootconfig -d /boot/initrd.img-X.Y.Z
```

Then add "bootconfig" on the normal kernel command line to tell the kernel to look for the bootconfig at the end of the initrd file. Alternatively, build your kernel with the `CONFIG_BOOT_CONFIG_FORCE` Kconfig option selected.

29.4.2 Embedding a Boot Config into Kernel

If you can not use initrd, you can also embed the bootconfig file in the kernel by Kconfig options. In this case, you need to recompile the kernel with the following configs:

```
CONFIG_BOOT_CONFIG_EMBED=y
CONFIG_BOOT_CONFIG_EMBED_FILE="/PATH/TO/BOOTCONFIG/FILE"
```

`CONFIG_BOOT_CONFIG_EMBED_FILE` requires an absolute path or a relative path to the bootconfig file from source tree or object tree. The kernel will embed it as the default bootconfig.

Just as when attaching the bootconfig to the initrd, you need `bootconfig` option on the kernel command line to enable the embedded bootconfig, or, alternatively, build your kernel with the `CONFIG_BOOT_CONFIG_FORCE` Kconfig option selected.

Note that even if you set this option, you can override the embedded bootconfig by another bootconfig which attached to the initrd.

29.5 Kernel parameters via Boot Config

In addition to the kernel command line, the boot config can be used for passing the kernel parameters. All the key-value pairs under `kernel` key will be passed to kernel cmdline directly. Moreover, the key-value pairs under `init` will be passed to init process via the cmdline. The parameters are concatenated with user-given kernel cmdline string as the following order, so that the command line parameter can override bootconfig parameters (this depends on how the subsystem handles parameters but in general, earlier parameter will be overwritten by later one.):

```
[bootconfig params][cmdline params] -- [bootconfig init params][cmdline init␣
↪params]
```

Here is an example of the bootconfig file for kernel/init parameters.:

```
kernel {
  root = 01234567-89ab-cdef-0123-456789abcd
}
init {
 splash
}
```

This will be copied into the kernel cmdline string as the following:

```
root="01234567-89ab-cdef-0123-456789abcd" -- splash
```

If user gives some other command line like,:

```
ro bootconfig -- quiet
```

The final kernel cmdline will be the following:

```
root="01234567-89ab-cdef-0123-456789abcd" ro bootconfig -- splash quiet
```

29.6 Config File Limitation

Currently the maximum config size size is 32KB and the total key-words (not key-value entries) must be under 1024 nodes. Note: this is not the number of entries but nodes, an entry must consume more than 2 nodes (a key-word and a value). So theoretically, it will be up to 512 key-value pairs. If keys contains 3 words in average, it can contain 256 key-value pairs. In most cases, the number of config items will be under 100 entries and smaller than 8KB, so it would be enough. If the node number exceeds 1024, parser returns an error even if the file size is smaller than 32KB. (Note that this maximum size is not including the padding null characters.) Anyway, since bootconfig command verifies it when appending a boot config to initrd image, user can notice it before boot.

29.7 Bootconfig APIs

User can query or loop on key-value pairs, also it is possible to find a root (prefix) key node and find key-values under that node.

If you have a key string, you can query the value directly with the key using xbc_find_value(). If you want to know what keys exist in the boot config, you can use xbc_for_each_key_value() to iterate key-value pairs. Note that you need to use xbc_array_for_each_value() for accessing each array's value, e.g.:

```
vnode = NULL;
xbc_find_value("key.word", &vnode);
if (vnode && xbc_node_is_array(vnode))
   xbc_array_for_each_value(vnode, value) {
     printk("%s ", value);
   }
```

If you want to focus on keys which have a prefix string, you can use xbc_find_node() to find a node by the prefix string, and iterate keys under the prefix node with xbc_node_for_each_key_value().

But the most typical usage is to get the named value under prefix or get the named array under prefix as below:

```
root = xbc_find_node("key.prefix");
value = xbc_node_find_value(root, "option", &vnode);
...
xbc_node_for_each_array_value(root, "array-option", value, anode) {
   ...
}
```

This accesses a value of ”key.prefix.option” and an array of ”key.prefix.array-option”.

Locking is not needed, since after initialization, the config becomes read-only. All data and keys must be copied if you need to modify it.

29.8 Functions and structures

uint32_t **xbc_calc_checksum**(void *data, uint32_t size)

Calculate checksum of bootconfig

Parameters

void *data
Bootconfig data.

uint32_t size
The size of the bootconfig data.

Description

Calculate the checksum value of the bootconfig data. The checksum will be used with the BOOTCONFIG_MAGIC and the size for embedding the bootconfig in the initrd image.

bool **xbc_node_is_value**(struct xbc_node *node)

Test the node is a value node

Parameters

struct xbc_node *node
An XBC node.

Description

Test the **node** is a value node and return true if a value node, false if not.

bool **xbc_node_is_key**(struct xbc_node *node)

Test the node is a key node

Parameters

struct xbc_node *node
An XBC node.

Description

Test the **node** is a key node and return true if a key node, false if not.

bool **xbc_node_is_array**(struct xbc_node *node)

Test the node is an arraied value node

Parameters

struct xbc_node *node
An XBC node.

Description

Test the **node** is an arraied value node.

bool **xbc_node_is_leaf**(struct xbc_node *node)

Test the node is a leaf key node

Parameters

struct xbc_node *node
An XBC node.

Description

Test the **node** is a leaf key node which is a key node and has a value node or no child. Returns true if it is a leaf node, or false if not. Note that the leaf node can have subkey nodes in addition to the value node.

const char ***xbc_find_value**(const char *key, struct xbc_node **vnode)

Find a value which matches the key

Parameters

const char *key
Search key

struct xbc_node **vnode
A container pointer of XBC value node.

Description

Search a value whose key matches **key** from whole of XBC tree and return the value if found. Found value node is stored in ***vnode**. Note that this can return 0-length string and store NULL in ***vnode** for key-only (non-value) entry.

struct xbc_node ***xbc_find_node**(const char *key)

Find a node which matches the key

Parameters

const char *key
Search key

Description

Search a (key) node whose key matches **key** from whole of XBC tree and return the node if found. If not found, returns NULL.

struct xbc_node ***xbc_node_get_subkey**(struct xbc_node *node)

Return the first subkey node if exists

Parameters

struct xbc_node *node
Parent node

Description

Return the first subkey node of the **node**. If the **node** has no child or only value node, this will return NULL.

xbc_array_for_each_value

xbc_array_for_each_value (anode, value)

Iterate value nodes on an array

Parameters

anode
An XBC arraied value node

value
A value

Description

Iterate array value nodes and values starts from **anode**. This is expected to be used with xbc_find_value() and *xbc_node_find_value()*, so that user can process each array entry node.

xbc_node_for_each_child

`xbc_node_for_each_child (parent, child)`

Iterate child nodes

Parameters

parent
An XBC node.

child
Iterated XBC node.

Description

Iterate child nodes of **parent**. Each child nodes are stored to **child**. The **child** can be mixture of a value node and subkey nodes.

xbc_node_for_each_subkey

`xbc_node_for_each_subkey (parent, child)`

Iterate child subkey nodes

Parameters

parent
An XBC node.

child
Iterated XBC node.

Description

Iterate subkey nodes of **parent**. Each child nodes are stored to **child**. The **child** is only the subkey node.

xbc_node_for_each_array_value

`xbc_node_for_each_array_value (node, key, anode, value)`

Iterate array entries of geven key

Parameters

node
An XBC node.

key
A key string searched under **node**

anode
Iterated XBC node of array entry.

value
Iterated value of array entry.

Description

Iterate array entries of given **key** under **node**. Each array entry node is stored to **anode** and **value**. If the **node** doesn't have **key** node, it does nothing. Note that even if the found key node has only one value (not array) this executes block once. However, if the found key node has no value (key-only node), this does nothing. So don't use this for testing the key-value pair existence.

xbc_node_for_each_key_value

`xbc_node_for_each_key_value (node, knode, value)`

Iterate key-value pairs under a node

Parameters

node
An XBC node.

knode
Iterated key node

value
Iterated value string

Description

Iterate key-value pairs under **node**. Each key node and value string are stored in **knode** and **value** respectively.

xbc_for_each_key_value

`xbc_for_each_key_value (knode, value)`

Iterate key-value pairs

Parameters

knode
Iterated key node

value
Iterated value string

Description

Iterate key-value pairs in whole XBC tree. Each key node and value string are stored in **knode** and **value** respectively.

int **xbc_node_compose_key**(struct xbc_node *node, char *buf, size_t size)
Compose full key string of the XBC node

Parameters

struct xbc_node *node
An XBC node.

char *buf
A buffer to store the key.

size_t size
The size of the **buf**.

Description

Compose the full-length key of the **node** into **buf**. Returns the total length of the key stored in **buf**. Or returns -EINVAL if **node** is NULL, and -ERANGE if the key depth is deeper than max depth.

int **xbc_get_info**(int *node_size, size_t *data_size)

Get the information of loaded boot config

Parameters

int *node_size
A pointer to store the number of nodes.

size_t *data_size
A pointer to store the size of bootconfig data.

Description

Get the number of used nodes in **node_size** if it is not NULL, and the size of bootconfig data in **data_size** if it is not NULL. Return 0 if the boot config is initialized, or return -ENODEV.

struct xbc_node ***xbc_root_node**(void)

Get the root node of extended boot config

Parameters

void
no arguments

Description

Return the address of root node of extended boot config. If the extended boot config is not initiized, return NULL.

int **xbc_node_index**(struct xbc_node *node)

Get the index of XBC node

Parameters

struct xbc_node *node
A target node of getting index.

Description

Return the index number of **node** in XBC node list.

struct xbc_node ***xbc_node_get_parent**(struct xbc_node *node)

Get the parent XBC node

Parameters

struct xbc_node *node
An XBC node.

Description

Return the parent node of **node**. If the node is top node of the tree, return NULL.

struct xbc_node ***xbc_node_get_child**(struct xbc_node *node)

Get the child XBC node

Parameters

struct xbc_node *node
An XBC node.

Description

Return the first child node of **node**. If the node has no child, return NULL.

struct xbc_node ***xbc_node_get_next**(struct xbc_node *node)
Get the next sibling XBC node

Parameters

struct xbc_node *node
An XBC node.

Description

Return the NEXT sibling node of **node**. If the node has no next sibling, return NULL. Note that even if this returns NULL, it doesn't mean **node** has no siblings. (You also has to check whether the parent's child node is **node** or not.)

const char ***xbc_node_get_data**(struct xbc_node *node)
Get the data of XBC node

Parameters

struct xbc_node *node
An XBC node.

Description

Return the data (which is always a null terminated string) of **node**. If the node has invalid data, warn and return NULL.

struct xbc_node ***xbc_node_find_subkey**(struct xbc_node *parent, const char *key)
Find a subkey node which matches given key

Parameters

struct xbc_node *parent
An XBC node.

const char *key
A key string.

Description

Search a key node under **parent** which matches **key**. The **key** can contain several words jointed with '.'. If **parent** is NULL, this searches the node from whole tree. Return NULL if no node is matched.

const char ***xbc_node_find_value**(struct xbc_node *parent, const char *key, struct xbc_node **vnode)
Find a value node which matches given key

Parameters

struct xbc_node *parent
An XBC node.

const char *key
A key string.

struct xbc_node **vnode
A container pointer of found XBC node.

Description

Search a value node under **parent** whose (parent) key node matches **key**, store it in ***vnode**, and returns the value string. The **key** can contain several words jointed with '.'. If **parent** is NULL, this searches the node from whole tree. Return the value string if a matched key found, return NULL if no node is matched. Note that this returns 0-length string and stores NULL in ***vnode** if the key has no value. And also it will return the value of the first entry if the value is an array.

int **xbc_node_compose_key_after**(struct xbc_node *root, struct xbc_node *node, char *buf, size_t size)

Compose partial key string of the XBC node

Parameters

struct xbc_node *root
Root XBC node

struct xbc_node *node
Target XBC node.

char *buf
A buffer to store the key.

size_t size
The size of the **buf**.

Description

Compose the partial key of the **node** into **buf**, which is starting right after **root** (**root** is not included.) If **root** is NULL, this returns full key words of **node**. Returns the total length of the key stored in **buf**. Returns -EINVAL if **node** is NULL or **root** is not the ancestor of **node** or **root** is **node**, or returns -ERANGE if the key depth is deeper than max depth. This is expected to be used with xbc_find_node() to list up all (child) keys under given key.

struct xbc_node ***xbc_node_find_next_leaf**(struct xbc_node *root, struct xbc_node *node)

Find the next leaf node under given node

Parameters

struct xbc_node *root
An XBC root node

struct xbc_node *node
An XBC node which starts from.

Description

Search the next leaf node (which means the terminal key node) of **node** under **root** node (including **root** node itself). Return the next node or NULL if next leaf node is not found.

const char ***xbc_node_find_next_key_value**(struct xbc_node *root, struct xbc_node **leaf)

Find the next key-value pair nodes

Parameters

struct xbc_node *root
An XBC root node

struct xbc_node **leaf
A container pointer of XBC node which starts from.

Description

Search the next leaf node (which means the terminal key node) of ***leaf** under **root** node. Returns the value and update ***leaf** if next leaf node is found, or NULL if no next leaf node is found. Note that this returns 0-length string if the key has no value, or the value of the first entry if the value is an array.

void **xbc_exit**(void)
Clean up all parsed bootconfig

Parameters

void
no arguments

Description

This clears all data structures of parsed bootconfig on memory. If you need to reuse *xbc_init()* with new boot config, you can use this.

int **xbc_init**(const char *data, size_t size, const char **emsg, int *epos)
Parse given XBC file and build XBC internal tree

Parameters

const char *data
The boot config text original data

size_t size
The size of **data**

const char **emsg
A pointer of const char * to store the error message

int *epos
A pointer of int to store the error position

Description

This parses the boot config text in **data**. **size** must be smaller than XBC_DATA_MAX. Return the number of stored nodes (>0) if succeeded, or -errno if there is any error. In error cases, **emsg** will be updated with an error message and **epos** will be updated with the error position which is the byte offset of **buf**. If the error is not a parser error, **epos** will be -1.

CHAPTER

THIRTY

LINUX BRAILLE CONSOLE

To get early boot messages on a braille device (before userspace screen readers can start), you first need to compile the support for the usual serial console (see *Documentation/admin-guide/serial-console.rst*), and for braille device (in *Device Drivers → Accessibility support → Console on braille device*).

Then you need to specify a `console=brl`, option on the kernel command line, the format is:

```
console=brl,serial_options...
```

where `serial_options...` are the same as described in *Documentation/admin-guide/serial-console.rst*.

So for instance you can use `console=brl,ttyS0` if the braille device is connected to the first serial port, and `console=brl,ttyS0,115200` to override the baud rate to 115200, etc.

By default, the braille device will just show the last kernel message (console mode). To review previous messages, press the Insert key to switch to the VT review mode. In review mode, the arrow keys permit to browse in the VT content, `PAGE-UP/PAGE-DOWN` keys go at the top/bottom of the screen, and the `HOME` key goes back to the cursor, hence providing very basic screen reviewing facility.

Sound feedback can be obtained by adding the `braille_console.sound=1` kernel parameter.

For simplicity, only one braille console can be enabled, other uses of `console=brl,...` will be discarded. Also note that it does not interfere with the console selection mechanism described in *Documentation/admin-guide/serial-console.rst*.

For now, only the VisioBraille device is supported.

Samuel Thibault <samuel.thibault@ens-lyon.org>

CHAPTER

THIRTYONE

BTMRVL DRIVER

All commands are used via debugfs interface.

31.1 Set/get driver configurations

Path: /debug/btmrvl/config/

gpiogap=[n], hscfgcmd

These commands are used to configure the host sleep parameters:: bit 8:0 -- Gap bit 16:8 -- GPIO

where GPIO is the pin number of GPIO used to wake up the host. It could be any valid GPIO pin# (e.g. 0-7) or 0xff (SDIO interface wakeup will be used instead).

where Gap is the gap in milli seconds between wakeup signal and wakeup event, or 0xff for special host sleep setting.

Usage:

```
# Use SDIO interface to wake up the host and set GAP to 0x80:
echo 0xff80 > /debug/btmrvl/config/gpiogap
echo 1 > /debug/btmrvl/config/hscfgcmd

# Use GPIO pin #3 to wake up the host and set GAP to 0xff:
echo 0x03ff >  /debug/btmrvl/config/gpiogap
echo 1 > /debug/btmrvl/config/hscfgcmd
```

psmode=[n], pscmd

These commands are used to enable/disable auto sleep mode

where the option is:

```
1       -- Enable auto sleep mode
0       -- Disable auto sleep mode
```

Usage:

```
# Enable auto sleep mode
echo 1 > /debug/btmrvl/config/psmode
echo 1 > /debug/btmrvl/config/pscmd

# Disable auto sleep mode
```

```
echo 0 > /debug/btmrvl/config/psmode
echo 1 > /debug/btmrvl/config/pscmd
```

hsmode=[n], hscmd
These commands are used to enable host sleep or wake up firmware

where the option is:

```
1       -- Enable host sleep
0       -- Wake up firmware
```

Usage:

```
# Enable host sleep
echo 1 > /debug/btmrvl/config/hsmode
echo 1 > /debug/btmrvl/config/hscmd

# Wake up firmware
echo 0 > /debug/btmrvl/config/hsmode
echo 1 > /debug/btmrvl/config/hscmd
```

31.2 Get driver status

Path: /debug/btmrvl/status/

Usage:

```
cat /debug/btmrvl/status/<args>
```

where the args are:

curpsmode
This command displays current auto sleep status.

psstate
This command display the power save state.

hsstate
This command display the host sleep state.

txdnldrdy
This command displays the value of Tx download ready flag.

31.3 Issuing a raw hci command

Use hcitool to issue raw hci command, refer to hcitool manual

Usage:

```
Hcitool cmd <ogf> <ocf> [Parameters]
```

Interface Control Command:

```
hcitool cmd 0x3f 0x5b 0xf5 0x01 0x00    --Enable All interface
hcitool cmd 0x3f 0x5b 0xf5 0x01 0x01    --Enable Wlan interface
hcitool cmd 0x3f 0x5b 0xf5 0x01 0x02    --Enable BT interface
hcitool cmd 0x3f 0x5b 0xf5 0x00 0x00    --Disable All interface
hcitool cmd 0x3f 0x5b 0xf5 0x00 0x01    --Disable Wlan interface
hcitool cmd 0x3f 0x5b 0xf5 0x00 0x02    --Disable BT interface
```

31.4 SD8688 firmware

Images:

- /lib/firmware/sd8688_helper.bin
- /lib/firmware/sd8688.bin

The images can be downloaded from:

git.infradead.org/users/dwmw2/linux-firmware.git/libertas/

CHAPTER

THIRTYTWO

CONTROL GROUPS VERSION 1

32.1 Control Groups

Written by Paul Menage <menage@google.com> based on *CPUSETS*

Original copyright statements from cpusets.txt:

Portions Copyright (C) 2004 BULL SA.

Portions Copyright (c) 2004-2006 Silicon Graphics, Inc.

Modified by Paul Jackson <pj@sgi.com>

Modified by Christoph Lameter <cl@linux.com>

32.1.1 1. Control Groups

1.1 What are cgroups ?

Control Groups provide a mechanism for aggregating/partitioning sets of tasks, and all their future children, into hierarchical groups with specialized behaviour.

Definitions:

A *cgroup* associates a set of tasks with a set of parameters for one or more subsystems.

A *subsystem* is a module that makes use of the task grouping facilities provided by cgroups to treat groups of tasks in particular ways. A subsystem is typically a "resource controller" that schedules a resource or applies per-cgroup limits, but it may be anything that wants to act on a group of processes, e.g. a virtualization subsystem.

A *hierarchy* is a set of cgroups arranged in a tree, such that every task in the system is in exactly one of the cgroups in the hierarchy, and a set of subsystems; each subsystem has system-specific state attached to each cgroup in the hierarchy. Each hierarchy has an instance of the cgroup virtual filesystem associated with it.

At any one time there may be multiple active hierarchies of task cgroups. Each hierarchy is a partition of all tasks in the system.

User-level code may create and destroy cgroups by name in an instance of the cgroup virtual file system, specify and query to which cgroup a task is assigned, and list the task PIDs assigned to a cgroup. Those creations and assignments only affect the hierarchy associated with that instance of the cgroup file system.

On their own, the only use for cgroups is for simple job tracking. The intention is that other subsystems hook into the generic cgroup support to provide new attributes for cgroups, such as accounting/limiting the resources which processes in a cgroup can access. For example, cpusets (see *CPUSETS*) allow you to associate a set of CPUs and a set of memory nodes with the tasks in each cgroup.

1.2 Why are cgroups needed ?

There are multiple efforts to provide process aggregations in the Linux kernel, mainly for resource-tracking purposes. Such efforts include cpusets, CKRM/ResGroups, UserBeanCounters, and virtual server namespaces. These all require the basic notion of a grouping/partitioning of processes, with newly forked processes ending up in the same group (cgroup) as their parent process.

The kernel cgroup patch provides the minimum essential kernel mechanisms required to efficiently implement such groups. It has minimal impact on the system fast paths, and provides hooks for specific subsystems such as cpusets to provide additional behaviour as desired.

Multiple hierarchy support is provided to allow for situations where the division of tasks into cgroups is distinctly different for different subsystems - having parallel hierarchies allows each hierarchy to be a natural division of tasks, without having to handle complex combinations of tasks that would be present if several unrelated subsystems needed to be forced into the same tree of cgroups.

At one extreme, each resource controller or subsystem could be in a separate hierarchy; at the other extreme, all subsystems would be attached to the same hierarchy.

As an example of a scenario (originally proposed by vatsa@in.ibm.com) that can benefit from multiple hierarchies, consider a large university server with various users - students, professors, system tasks etc. The resource planning for this server could be along the following lines:

```
CPU :          "Top cpuset"
                /       \
        CPUSet1         CPUSet2
           |               |
        (Professors)    (Students)

        In addition (system tasks) are attached to topcpuset (so
        that they can run anywhere) with a limit of 20%

Memory : Professors (50%), Students (30%), system (20%)

Disk : Professors (50%), Students (30%), system (20%)

Network : WWW browsing (20%), Network File System (60%), others (20%)
                         / \
        Professors (15%)  students (5%)
```

Browsers like Firefox/Lynx go into the WWW network class, while (k)nfsd goes into the NFS network class.

At the same time Firefox/Lynx will share an appropriate CPU/Memory class depending on who launched it (prof/student).

With the ability to classify tasks differently for different resources (by putting those resource subsystems in different hierarchies), the admin can easily set up a script which receives exec notifications and depending on who is launching the browser he can:

```
# echo browser_pid > /sys/fs/cgroup/<restype>/<userclass>/tasks
```

With only a single hierarchy, he now would potentially have to create a separate cgroup for every browser launched and associate it with appropriate network and other resource class. This may lead to proliferation of such cgroups.

Also let's say that the administrator would like to give enhanced network access temporarily to a student's browser (since it is night and the user wants to do online gaming :)) OR give one of the student's simulation apps enhanced CPU power.

With ability to write PIDs directly to resource classes, it's just a matter of:

```
# echo pid > /sys/fs/cgroup/network/<new_class>/tasks
(after some time)
# echo pid > /sys/fs/cgroup/network/<orig_class>/tasks
```

Without this ability, the administrator would have to split the cgroup into multiple separate ones and then associate the new cgroups with the new resource classes.

1.3 How are cgroups implemented ?

Control Groups extends the kernel as follows:

- Each task in the system has a reference-counted pointer to a css_set.
- A css_set contains a set of reference-counted pointers to cgroup_subsys_state objects, one for each cgroup subsystem registered in the system. There is no direct link from a task to the cgroup of which it's a member in each hierarchy, but this can be determined by following pointers through the cgroup_subsys_state objects. This is because accessing the subsystem state is something that's expected to happen frequently and in performance-critical code, whereas operations that require a task's actual cgroup assignments (in particular, moving between cgroups) are less common. A linked list runs through the cg_list field of each task_struct using the css_set, anchored at css_set->tasks.
- A cgroup hierarchy filesystem can be mounted for browsing and manipulation from user space.
- You can list all the tasks (by PID) attached to any cgroup.

The implementation of cgroups requires a few, simple hooks into the rest of the kernel, none in performance-critical paths:

- in init/main.c, to initialize the root cgroups and initial css_set at system boot.
- in fork and exit, to attach and detach a task from its css_set.

In addition, a new file system of type ”cgroup” may be mounted, to enable browsing and modifying the cgroups presently known to the kernel. When mounting a cgroup hierarchy, you may specify a comma-separated list of subsystems to mount as the filesystem mount options. By default, mounting the cgroup filesystem attempts to mount a hierarchy containing all registered subsystems.

If an active hierarchy with exactly the same set of subsystems already exists, it will be reused for the new mount. If no existing hierarchy matches, and any of the requested subsystems are in use in an existing hierarchy, the mount will fail with -EBUSY. Otherwise, a new hierarchy is activated, associated with the requested subsystems.

It's not currently possible to bind a new subsystem to an active cgroup hierarchy, or to unbind a subsystem from an active cgroup hierarchy. This may be possible in future, but is fraught with nasty error-recovery issues.

When a cgroup filesystem is unmounted, if there are any child cgroups created below the top-level cgroup, that hierarchy will remain active even though unmounted; if there are no child cgroups then the hierarchy will be deactivated.

No new system calls are added for cgroups - all support for querying and modifying cgroups is via this cgroup file system.

Each task under /proc has an added file named 'cgroup' displaying, for each active hierarchy, the subsystem names and the cgroup name as the path relative to the root of the cgroup file system.

Each cgroup is represented by a directory in the cgroup file system containing the following files describing that cgroup:

- tasks: list of tasks (by PID) attached to that cgroup. This list is not guaranteed to be sorted. Writing a thread ID into this file moves the thread into this cgroup.
- cgroup.procs: list of thread group IDs in the cgroup. This list is not guaranteed to be sorted or free of duplicate TGIDs, and userspace should sort/uniquify the list if this property is required. Writing a thread group ID into this file moves all threads in that group into this cgroup.
- notify_on_release flag: run the release agent on exit?
- release_agent: the path to use for release notifications (this file exists in the top cgroup only)

Other subsystems such as cpusets may add additional files in each cgroup dir.

New cgroups are created using the mkdir system call or shell command. The properties of a cgroup, such as its flags, are modified by writing to the appropriate file in that cgroups directory, as listed above.

The named hierarchical structure of nested cgroups allows partitioning a large system into nested, dynamically changeable, ”soft-partitions”.

The attachment of each task, automatically inherited at fork by any children of that task, to a cgroup allows organizing the work load on a system into related sets of tasks. A task may be re-attached to any other cgroup, if allowed by the permissions on the necessary cgroup file system directories.

When a task is moved from one cgroup to another, it gets a new css_set pointer - if there's an already existing css_set with the desired collection of cgroups then that group is reused, otherwise a new css_set is allocated. The appropriate existing css_set is located by looking into a hash table.

To allow access from a cgroup to the css_sets (and hence tasks) that comprise it, a set of cg_cgroup_link objects form a lattice; each cg_cgroup_link is linked into a list of cg_cgroup_links for a single cgroup on its cgrp_link_list field, and a list of cg_cgroup_links for a single css_set on its cg_link_list.

Thus the set of tasks in a cgroup can be listed by iterating over each css_set that references the cgroup, and sub-iterating over each css_set's task set.

The use of a Linux virtual file system (vfs) to represent the cgroup hierarchy provides for a familiar permission and name space for cgroups, with a minimum of additional kernel code.

1.4 What does notify_on_release do ?

If the notify_on_release flag is enabled (1) in a cgroup, then whenever the last task in the cgroup leaves (exits or attaches to some other cgroup) and the last child cgroup of that cgroup is removed, then the kernel runs the command specified by the contents of the "release_agent" file in that hierarchy's root directory, supplying the pathname (relative to the mount point of the cgroup file system) of the abandoned cgroup. This enables automatic removal of abandoned cgroups. The default value of notify_on_release in the root cgroup at system boot is disabled (0). The default value of other cgroups at creation is the current value of their parents' notify_on_release settings. The default value of a cgroup hierarchy's release_agent path is empty.

1.5 What does clone_children do ?

This flag only affects the cpuset controller. If the clone_children flag is enabled (1) in a cgroup, a new cpuset cgroup will copy its configuration from the parent during initialization.

1.6 How do I use cgroups ?

To start a new job that is to be contained within a cgroup, using the "cpuset" cgroup subsystem, the steps are something like:

```
1) mount -t tmpfs cgroup_root /sys/fs/cgroup
2) mkdir /sys/fs/cgroup/cpuset
3) mount -t cgroup -ocpuset cpuset /sys/fs/cgroup/cpuset
4) Create the new cgroup by doing mkdir's and write's (or echo's) in
   the /sys/fs/cgroup/cpuset virtual file system.
5) Start a task that will be the "founding father" of the new job.
6) Attach that task to the new cgroup by writing its PID to the
   /sys/fs/cgroup/cpuset tasks file for that cgroup.
7) fork, exec or clone the job tasks from this founding father task.
```

For example, the following sequence of commands will setup a cgroup named "Charlie", containing just CPUs 2 and 3, and Memory Node 1, and then start a subshell 'sh' in that cgroup:

```
mount -t tmpfs cgroup_root /sys/fs/cgroup
mkdir /sys/fs/cgroup/cpuset
mount -t cgroup cpuset -ocpuset /sys/fs/cgroup/cpuset
cd /sys/fs/cgroup/cpuset
mkdir Charlie
cd Charlie
/bin/echo 2-3 > cpuset.cpus
/bin/echo 1 > cpuset.mems
/bin/echo $$ > tasks
sh
```

```
# The subshell 'sh' is now running in cgroup Charlie
# The next line should display '/Charlie'
cat /proc/self/cgroup
```

32.1.2 2. Usage Examples and Syntax

2.1 Basic Usage

Creating, modifying, using cgroups can be done through the cgroup virtual filesystem.

To mount a cgroup hierarchy with all available subsystems, type:

```
# mount -t cgroup xxx /sys/fs/cgroup
```

The "xxx" is not interpreted by the cgroup code, but will appear in /proc/mounts so may be any useful identifying string that you like.

Note: Some subsystems do not work without some user input first. For instance, if cpusets are enabled the user will have to populate the cpus and mems files for each new cgroup created before that group can be used.

As explained in section *1.2 Why are cgroups needed?* you should create different hierarchies of cgroups for each single resource or group of resources you want to control. Therefore, you should mount a tmpfs on /sys/fs/cgroup and create directories for each cgroup resource or resource group:

```
# mount -t tmpfs cgroup_root /sys/fs/cgroup
# mkdir /sys/fs/cgroup/rg1
```

To mount a cgroup hierarchy with just the cpuset and memory subsystems, type:

```
# mount -t cgroup -o cpuset,memory hier1 /sys/fs/cgroup/rg1
```

While remounting cgroups is currently supported, it is not recommend to use it. Remounting allows changing bound subsystems and release_agent. Rebinding is hardly useful as it only works when the hierarchy is empty and release_agent itself should be replaced with conventional fsnotify. The support for remounting will be removed in the future.

To Specify a hierarchy's release_agent:

```
# mount -t cgroup -o cpuset,release_agent="/sbin/cpuset_release_agent" \
  xxx /sys/fs/cgroup/rg1
```

Note that specifying 'release_agent' more than once will return failure.

Note that changing the set of subsystems is currently only supported when the hierarchy consists of a single (root) cgroup. Supporting the ability to arbitrarily bind/unbind subsystems from an existing cgroup hierarchy is intended to be implemented in the future.

Then under /sys/fs/cgroup/rg1 you can find a tree that corresponds to the tree of the cgroups in the system. For instance, /sys/fs/cgroup/rg1 is the cgroup that holds the whole system.

If you want to change the value of release_agent:

```
# echo "/sbin/new_release_agent" > /sys/fs/cgroup/rg1/release_agent
```

It can also be changed via remount.

If you want to create a new cgroup under /sys/fs/cgroup/rg1:

```
# cd /sys/fs/cgroup/rg1
# mkdir my_cgroup
```

Now you want to do something with this cgroup:

cd my_cgroup

In this directory you can find several files:

```
# ls
cgroup.procs notify_on_release tasks
(plus whatever files added by the attached subsystems)
```

Now attach your shell to this cgroup:

```
# /bin/echo $$ > tasks
```

You can also create cgroups inside your cgroup by using mkdir in this directory:

```
# mkdir my_sub_cs
```

To remove a cgroup, just use rmdir:

```
# rmdir my_sub_cs
```

This will fail if the cgroup is in use (has cgroups inside, or has processes attached, or is held alive by other subsystem-specific reference).

2.2 Attaching processes

```
# /bin/echo PID > tasks
```

Note that it is PID, not PIDs. You can only attach ONE task at a time. If you have several tasks to attach, you have to do it one after another:

```
# /bin/echo PID1 > tasks
# /bin/echo PID2 > tasks
        ...
# /bin/echo PIDn > tasks
```

You can attach the current shell task by echoing 0:

```
# echo 0 > tasks
```

You can use the cgroup.procs file instead of the tasks file to move all threads in a threadgroup at once. Echoing the PID of any task in a threadgroup to cgroup.procs causes all tasks in that

threadgroup to be attached to the cgroup. Writing 0 to cgroup.procs moves all tasks in the writing task's threadgroup.

Note: Since every task is always a member of exactly one cgroup in each mounted hierarchy, to remove a task from its current cgroup you must move it into a new cgroup (possibly the root cgroup) by writing to the new cgroup's tasks file.

Note: Due to some restrictions enforced by some cgroup subsystems, moving a process to another cgroup can fail.

2.3 Mounting hierarchies by name

Passing the name=<x> option when mounting a cgroups hierarchy associates the given name with the hierarchy. This can be used when mounting a pre-existing hierarchy, in order to refer to it by name rather than by its set of active subsystems. Each hierarchy is either nameless, or has a unique name.

The name should match [w.-]+

When passing a name=<x> option for a new hierarchy, you need to specify subsystems manually; the legacy behaviour of mounting all subsystems when none are explicitly specified is not supported when you give a subsystem a name.

The name of the subsystem appears as part of the hierarchy description in /proc/mounts and /proc/<pid>/cgroups.

32.1.3 3. Kernel API

3.1 Overview

Each kernel subsystem that wants to hook into the generic cgroup system needs to create a cgroup_subsys object. This contains various methods, which are callbacks from the cgroup system, along with a subsystem ID which will be assigned by the cgroup system.

Other fields in the cgroup_subsys object include:

- subsys_id: a unique array index for the subsystem, indicating which entry in cgroup->subsys[] this subsystem should be managing.
- name: should be initialized to a unique subsystem name. Should be no longer than MAX_CGROUP_TYPE_NAMELEN.
- early_init: indicate if the subsystem needs early initialization at system boot.

Each cgroup object created by the system has an array of pointers, indexed by subsystem ID; this pointer is entirely managed by the subsystem; the generic cgroup code will never touch this pointer.

3.2 Synchronization

There is a global mutex, cgroup_mutex, used by the cgroup system. This should be taken by anything that wants to modify a cgroup. It may also be taken to prevent cgroups from being modified, but more specific locks may be more appropriate in that situation.

See kernel/cgroup.c for more details.

Subsystems can take/release the cgroup_mutex via the functions cgroup_lock()/cgroup_unlock().

Accessing a task's cgroup pointer may be done in the following ways: - while holding cgroup_mutex - while holding the task's alloc_lock (via task_lock()) - inside an rcu_read_lock() section via rcu_dereference()

3.3 Subsystem API

Each subsystem should:

- add an entry in linux/cgroup_subsys.h
- define a cgroup_subsys object called <name>_cgrp_subsys

Each subsystem may export the following methods. The only mandatory methods are css_alloc/free. Any others that are null are presumed to be successful no-ops.

`struct cgroup_subsys_state *css_alloc(struct cgroup *cgrp)` (cgroup_mutex held by caller)

Called to allocate a subsystem state object for a cgroup. The subsystem should allocate its subsystem state object for the passed cgroup, returning a pointer to the new object on success or a ERR_PTR() value. On success, the subsystem pointer should point to a structure of type cgroup_subsys_state (typically embedded in a larger subsystem-specific object), which will be initialized by the cgroup system. Note that this will be called at initialization to create the root subsystem state for this subsystem; this case can be identified by the passed cgroup object having a NULL parent (since it's the root of the hierarchy) and may be an appropriate place for initialization code.

`int css_online(struct cgroup *cgrp)` (cgroup_mutex held by caller)

Called after @cgrp successfully completed all allocations and made visible to cgroup_for_each_child/descendant_*() iterators. The subsystem may choose to fail creation by returning -errno. This callback can be used to implement reliable state sharing and propagation along the hierarchy. See the comment on cgroup_for_each_descendant_pre() for details.

`void css_offline(struct cgroup *cgrp);` (cgroup_mutex held by caller)

This is the counterpart of css_online() and called iff css_online() has succeeded on @cgrp. This signifies the beginning of the end of @cgrp. @cgrp is being removed and the subsystem should start dropping all references it's holding on @cgrp. When all references are dropped, cgroup removal will proceed to the next step - css_free(). After this callback, @cgrp should be considered dead to the subsystem.

`void css_free(struct cgroup *cgrp)` (cgroup_mutex held by caller)

The cgroup system is about to free @cgrp; the subsystem should free its subsystem state object. By the time this method is called, @cgrp is completely unused; @cgrp->parent is still valid.

(Note - can also be called for a newly-created cgroup if an error occurs after this subsystem's create() method has been called for the new cgroup).

`int can_attach(struct cgroup *cgrp, struct cgroup_taskset *tset)` (cgroup_mutex held by caller)

Called prior to moving one or more tasks into a cgroup; if the subsystem returns an error, this will abort the attach operation. @tset contains the tasks to be attached and is guaranteed to have at least one task in it.

If there are multiple tasks in the taskset, then:

- it's guaranteed that all are from the same thread group
- @tset contains all tasks from the thread group whether or not they're switching cgroups
- the first task is the leader

Each @tset entry also contains the task's old cgroup and tasks which aren't switching cgroup can be skipped easily using the cgroup_taskset_for_each() iterator. Note that this isn't called on a fork. If this method returns 0 (success) then this should remain valid while the caller holds cgroup_mutex and it is ensured that either attach() or cancel_attach() will be called in future.

`void css_reset(struct cgroup_subsys_state *css)` (cgroup_mutex held by caller)

An optional operation which should restore @css's configuration to the initial state. This is currently only used on the unified hierarchy when a subsystem is disabled on a cgroup through "cgroup.subtree_control" but should remain enabled because other subsystems depend on it. cgroup core makes such a css invisible by removing the associated interface files and invokes this callback so that the hidden subsystem can return to the initial neutral state. This prevents unexpected resource control from a hidden css and ensures that the configuration is in the initial state when it is made visible again later.

`void cancel_attach(struct cgroup *cgrp, struct cgroup_taskset *tset)` (cgroup_mutex held by caller)

Called when a task attach operation has failed after can_attach() has succeeded. A subsystem whose can_attach() has some side-effects should provide this function, so that the subsystem can implement a rollback. If not, not necessary. This will be called only about subsystems whose can_attach() operation have succeeded. The parameters are identical to can_attach().

`void attach(struct cgroup *cgrp, struct cgroup_taskset *tset)` (cgroup_mutex held by caller)

Called after the task has been attached to the cgroup, to allow any post-attachment activity that requires memory allocations or blocking. The parameters are identical to can_attach().

`void fork(struct task_struct *task)`

Called when a task is forked into a cgroup.

`void exit(struct task_struct *task)`

Called during task exit.

`void free(struct task_struct *task)`

Called when the task_struct is freed.

`void bind(struct cgroup *root)` (cgroup_mutex held by caller)

Called when a cgroup subsystem is rebound to a different hierarchy and root cgroup. Currently this will only involve movement between the default hierarchy (which never has sub-cgroups) and a hierarchy that is being created/destroyed (and hence has no sub-cgroups).

32.1.4 4. Extended attribute usage

cgroup filesystem supports certain types of extended attributes in its directories and files. The current supported types are:

- Trusted (XATTR_TRUSTED)
- Security (XATTR_SECURITY)

Both require CAP_SYS_ADMIN capability to set.

Like in tmpfs, the extended attributes in cgroup filesystem are stored using kernel memory and it's advised to keep the usage at minimum. This is the reason why user defined extended attributes are not supported, since any user can do it and there's no limit in the value size.

The current known users for this feature are SELinux to limit cgroup usage in containers and systemd for assorted meta data like main PID in a cgroup (systemd creates a cgroup per service).

32.1.5 5. Questions

```
Q: what's up with this '/bin/echo' ?
A: bash's builtin 'echo' command does not check calls to write() against
   errors. If you use it in the cgroup file system, you won't be
   able to tell whether a command succeeded or failed.

Q: When I attach processes, only the first of the line gets really attached !
A: We can only return one error code per call to write(). So you should also
   put only ONE PID.
```

32.2 Block IO Controller

32.2.1 Overview

cgroup subsys ”blkio” implements the block io controller. There seems to be a need of various kinds of IO control policies (like proportional BW, max BW) both at leaf nodes as well as at intermediate nodes in a storage hierarchy. Plan is to use the same cgroup based management interface for blkio controller and based on user options switch IO policies in the background.

One IO control policy is throttling policy which can be used to specify upper IO rate limits on devices. This policy is implemented in generic block layer and can be used on leaf nodes as well as higher level logical devices like device mapper.

32.2.2 HOWTO

Throttling/Upper Limit policy

Enable Block IO controller:

```
CONFIG_BLK_CGROUP=y
```

Enable throttling in block layer:

```
CONFIG_BLK_DEV_THROTTLING=y
```

Mount blkio controller (see cgroups.txt, Why are cgroups needed?):

```
mount -t cgroup -o blkio none /sys/fs/cgroup/blkio
```

Specify a bandwidth rate on particular device for root group. The format for policy is "<major>:<minor> <bytes_per_second>":

```
echo "8:16  1048576" > /sys/fs/cgroup/blkio/blkio.throttle.read_bps_device
```

This will put a limit of 1MB/second on reads happening for root group on device having major/minor number 8:16.

Run dd to read a file and see if rate is throttled to 1MB/s or not:

```
# dd iflag=direct if=/mnt/common/zerofile of=/dev/null bs=4K count=1024
1024+0 records in
1024+0 records out
4194304 bytes (4.2 MB) copied, 4.0001 s, 1.0 MB/s
```

Limits for writes can be put using blkio.throttle.write_bps_device file.

32.2.3 Hierarchical Cgroups

Throttling implements hierarchy support; however, throttling's hierarchy support is enabled iff "sane_behavior" is enabled from cgroup side, which currently is a development option and not publicly available.

If somebody created a hierarchy like as follows:

```
   root
   /  \
test1 test2
   |
test3
```

Throttling with "sane_behavior" will handle the hierarchy correctly. For throttling, all limits apply to the whole subtree while all statistics are local to the IOs directly generated by tasks in that cgroup.

Throttling without "sane_behavior" enabled from cgroup side will practically treat all groups at same level as if it looks like the following:

```
        pivot
     /  /   \  \
root  test1 test2  test3
```

32.2.4 Various user visible config options

CONFIG_BLK_CGROUP
Block IO controller.

CONFIG_BFQ_CGROUP_DEBUG
Debug help. Right now some additional stats file show up in cgroup if this option is enabled.

CONFIG_BLK_DEV_THROTTLING
Enable block device throttling support in block layer.

32.2.5 Details of cgroup files

Proportional weight policy files

blkio.bfq.weight
Specifies per cgroup weight. This is default weight of the group on all the devices until and unless overridden by per device rule (see *blkio.bfq.weight_device* below).

Currently allowed range of weights is from 1 to 1000. For more details, see Documentation/block/bfq-iosched.rst.

blkio.bfq.weight_device
Specifies per cgroup per device weights, overriding the default group weight. For more details, see Documentation/block/bfq-iosched.rst.

Following is the format:

```
# echo dev_maj:dev_minor weight > blkio.bfq.weight_device
```

Configure weight=300 on /dev/sdb (8:16) in this cgroup:

```
# echo 8:16 300 > blkio.bfq.weight_device
# cat blkio.bfq.weight_device
dev     weight
8:16    300
```

Configure weight=500 on /dev/sda (8:0) in this cgroup:

```
# echo 8:0 500 > blkio.bfq.weight_device
# cat blkio.bfq.weight_device
dev     weight
8:0     500
8:16    300
```

Remove specific weight for /dev/sda in this cgroup:

```
# echo 8:0 0 > blkio.bfq.weight_device
# cat blkio.bfq.weight_device
dev     weight
8:16    300
```

blkio.time
Disk time allocated to cgroup per device in milliseconds. First two fields specify the major and minor number of the device and third field specifies the disk time allocated to group in milliseconds.

blkio.sectors
Number of sectors transferred to/from disk by the group. First two fields specify the major and minor number of the device and third field specifies the number of sectors transferred by the group to/from the device.

blkio.io_service_bytes
Number of bytes transferred to/from the disk by the group. These are further divided by the type of operation - read or write, sync or async. First two fields specify the major and minor number of the device, third field specifies the operation type and the fourth field specifies the number of bytes.

blkio.io_serviced
Number of IOs (bio) issued to the disk by the group. These are further divided by the type of operation - read or write, sync or async. First two fields specify the major and minor number of the device, third field specifies the operation type and the fourth field specifies the number of IOs.

blkio.io_service_time
Total amount of time between request dispatch and request completion for the IOs done by this cgroup. This is in nanoseconds to make it meaningful for flash devices too. For devices with queue depth of 1, this time represents the actual service time. When queue_depth > 1, that is no longer true as requests may be served out of order. This may cause the service time for a given IO to include the service time of multiple IOs when served out of order which may result in total io_service_time > actual time elapsed. This time is further divided by the type of operation - read or write, sync or async. First two fields specify the major and minor number of the device, third field specifies the operation type and the fourth field specifies the io_service_time in ns.

blkio.io_wait_time
Total amount of time the IOs for this cgroup spent waiting in the scheduler queues for service. This can be greater than the total time elapsed since it is cumulative io_wait_time for all IOs. It is not a measure of total time the cgroup spent waiting but rather a measure of the wait_time for its individual IOs. For devices with queue_depth > 1 this metric does not include the time spent waiting for service once the IO is dispatched to the device but till it actually gets serviced (there might be a time lag here due to re-ordering of requests by the device). This is in nanoseconds to make it meaningful for flash devices too. This time is further divided by the type of operation - read or write, sync or async. First two fields specify the major and minor number of the device, third field specifies the operation type and the fourth field specifies the io_wait_time in ns.

blkio.io_merged
Total number of bios/requests merged into requests belonging to this cgroup.

This is further divided by the type of operation - read or write, sync or async.

blkio.io_queued
Total number of requests queued up at any given instant for this cgroup. This is further divided by the type of operation - read or write, sync or async.

blkio.avg_queue_size
Debugging aid only enabled if CONFIG_BFQ_CGROUP_DEBUG=y. The average queue size for this cgroup over the entire time of this cgroup's existence. Queue size samples are taken each time one of the queues of this cgroup gets a timeslice.

blkio.group_wait_time
Debugging aid only enabled if CONFIG_BFQ_CGROUP_DEBUG=y. This is the amount of time the cgroup had to wait since it became busy (i.e., went from 0 to 1 request queued) to get a timeslice for one of its queues. This is different from the io_wait_time which is the cumulative total of the amount of time spent by each IO in that cgroup waiting in the scheduler queue. This is in nanoseconds. If this is read when the cgroup is in a waiting (for timeslice) state, the stat will only report the group_wait_time accumulated till the last time it got a timeslice and will not include the current delta.

blkio.empty_time
Debugging aid only enabled if CONFIG_BFQ_CGROUP_DEBUG=y. This is the amount of time a cgroup spends without any pending requests when not being served, i.e., it does not include any time spent idling for one of the queues of the cgroup. This is in nanoseconds. If this is read when the cgroup is in an empty state, the stat will only report the empty_time accumulated till the last time it had a pending request and will not include the current delta.

blkio.idle_time
Debugging aid only enabled if CONFIG_BFQ_CGROUP_DEBUG=y. This is the amount of time spent by the IO scheduler idling for a given cgroup in anticipation of a better request than the existing ones from other queues/cgroups. This is in nanoseconds. If this is read when the cgroup is in an idling state, the stat will only report the idle_time accumulated till the last idle period and will not include the current delta.

blkio.dequeue
Debugging aid only enabled if CONFIG_BFQ_CGROUP_DEBUG=y. This gives the statistics about how many a times a group was dequeued from service tree of the device. First two fields specify the major and minor number of the device and third field specifies the number of times a group was dequeued from a particular device.

blkio.*_recursive
Recursive version of various stats. These files show the same information as their non-recursive counterparts but include stats from all the descendant cgroups.

Throttling/Upper limit policy files

blkio.throttle.read_bps_device
Specifies upper limit on READ rate from the device. IO rate is specified in bytes per second. Rules are per device. Following is the format:

```
echo "<major>:<minor>  <rate_bytes_per_second>" > /cgrp/blkio.
↪throttle.read_bps_device
```

blkio.throttle.write_bps_device
Specifies upper limit on WRITE rate to the device. IO rate is specified in bytes per second. Rules are per device. Following is the format:

```
echo "<major>:<minor>  <rate_bytes_per_second>" > /cgrp/blkio.
↪throttle.write_bps_device
```

blkio.throttle.read_iops_device
Specifies upper limit on READ rate from the device. IO rate is specified in IO per second. Rules are per device. Following is the format:

```
echo "<major>:<minor>  <rate_io_per_second>" > /cgrp/blkio.
↪throttle.read_iops_device
```

blkio.throttle.write_iops_device
Specifies upper limit on WRITE rate to the device. IO rate is specified in io per second. Rules are per device. Following is the format:

```
echo "<major>:<minor>  <rate_io_per_second>" > /cgrp/blkio.
↪throttle.write_iops_device
```

Note: If both BW and IOPS rules are specified for a device, then IO is subjected to both the constraints.

blkio.throttle.io_serviced
Number of IOs (bio) issued to the disk by the group. These are further divided by the type of operation - read or write, sync or async. First two fields specify the major and minor number of the device, third field specifies the operation type and the fourth field specifies the number of IOs.

blkio.throttle.io_service_bytes
Number of bytes transferred to/from the disk by the group. These are further divided by the type of operation - read or write, sync or async. First two fields specify the major and minor number of the device, third field specifies the operation type and the fourth field specifies the number of bytes.

Common files among various policies

blkio.reset_stats
Writing an int to this file will result in resetting all the stats for that cgroup.

32.3 CPU Accounting Controller

The CPU accounting controller is used to group tasks using cgroups and account the CPU usage of these groups of tasks.

The CPU accounting controller supports multi-hierarchy groups. An accounting group accumulates the CPU usage of all of its child groups and the tasks directly present in its group.

Accounting groups can be created by first mounting the cgroup filesystem:

```
# mount -t cgroup -ocpuacct none /sys/fs/cgroup
```

With the above step, the initial or the parent accounting group becomes visible at /sys/fs/cgroup. At bootup, this group includes all the tasks in the system. /sys/fs/cgroup/tasks lists the tasks in this cgroup. /sys/fs/cgroup/cpuacct.usage gives the CPU time (in nanoseconds) obtained by this group which is essentially the CPU time obtained by all the tasks in the system.

New accounting groups can be created under the parent group /sys/fs/cgroup:

```
# cd /sys/fs/cgroup
# mkdir g1
# echo $$ > g1/tasks
```

The above steps create a new group g1 and move the current shell process (bash) into it. CPU time consumed by this bash and its children can be obtained from g1/cpuacct.usage and the same is accumulated in /sys/fs/cgroup/cpuacct.usage also.

cpuacct.stat file lists a few statistics which further divide the CPU time obtained by the cgroup into user and system times. Currently the following statistics are supported:

user: Time spent by tasks of the cgroup in user mode. system: Time spent by tasks of the cgroup in kernel mode.

user and system are in USER_HZ unit.

cpuacct controller uses percpu_counter interface to collect user and system times. This has two side effects:

- It is theoretically possible to see wrong values for user and system times. This is because percpu_counter_read() on 32bit systems isn't safe against concurrent writes.
- It is possible to see slightly outdated values for user and system times due to the batch processing nature of percpu_counter.

32.4 CPUSETS

Copyright (C) 2004 BULL SA.

Written by Simon.Derr@bull.net

- Portions Copyright (c) 2004-2006 Silicon Graphics, Inc.
- Modified by Paul Jackson <pj@sgi.com>
- Modified by Christoph Lameter <cl@linux.com>
- Modified by Paul Menage <menage@google.com>
- Modified by Hidetoshi Seto <seto.hidetoshi@jp.fujitsu.com>

32.4.1 1. Cpusets

1.1 What are cpusets ?

Cpusets provide a mechanism for assigning a set of CPUs and Memory Nodes to a set of tasks. In this document ”Memory Node” refers to an on-line node that contains memory.

Cpusets constrain the CPU and Memory placement of tasks to only the resources within a task's current cpuset. They form a nested hierarchy visible in a virtual file system. These are the essential hooks, beyond what is already present, required to manage dynamic job placement on large systems.

Cpusets use the generic cgroup subsystem described in *Control Groups*.

Requests by a task, using the sched_setaffinity(2) system call to include CPUs in its CPU affinity mask, and using the mbind(2) and set_mempolicy(2) system calls to include Memory Nodes in its memory policy, are both filtered through that task's cpuset, filtering out any CPUs or Memory Nodes not in that cpuset. The scheduler will not schedule a task on a CPU that is not allowed in its cpus_allowed vector, and the kernel page allocator will not allocate a page on a node that is not allowed in the requesting task's mems_allowed vector.

User level code may create and destroy cpusets by name in the cgroup virtual file system, manage the attributes and permissions of these cpusets and which CPUs and Memory Nodes are assigned to each cpuset, specify and query to which cpuset a task is assigned, and list the task pids assigned to a cpuset.

1.2 Why are cpusets needed ?

The management of large computer systems, with many processors (CPUs), complex memory cache hierarchies and multiple Memory Nodes having non-uniform access times (NUMA) presents additional challenges for the efficient scheduling and memory placement of processes.

Frequently more modest sized systems can be operated with adequate efficiency just by letting the operating system automatically share the available CPU and Memory resources amongst the requesting tasks.

But larger systems, which benefit more from careful processor and memory placement to reduce memory access times and contention, and which typically represent a larger investment for the customer, can benefit from explicitly placing jobs on properly sized subsets of the system.

This can be especially valuable on:

- Web Servers running multiple instances of the same web application,
- Servers running different applications (for instance, a web server and a database), or
- NUMA systems running large HPC applications with demanding performance characteristics.

These subsets, or ”soft partitions” must be able to be dynamically adjusted, as the job mix changes, without impacting other concurrently executing jobs. The location of the running jobs pages may also be moved when the memory locations are changed.

The kernel cpuset patch provides the minimum essential kernel mechanisms required to efficiently implement such subsets. It leverages existing CPU and Memory Placement facilities in the Linux kernel to avoid any additional impact on the critical scheduler or memory allocator code.

1.3 How are cpusets implemented ?

Cpusets provide a Linux kernel mechanism to constrain which CPUs and Memory Nodes are used by a process or set of processes.

The Linux kernel already has a pair of mechanisms to specify on which CPUs a task may be scheduled (sched_setaffinity) and on which Memory Nodes it may obtain memory (mbind, set_mempolicy).

Cpusets extends these two mechanisms as follows:

- Cpusets are sets of allowed CPUs and Memory Nodes, known to the kernel.
- Each task in the system is attached to a cpuset, via a pointer in the task structure to a reference counted cgroup structure.
- Calls to sched_setaffinity are filtered to just those CPUs allowed in that task's cpuset.
- Calls to mbind and set_mempolicy are filtered to just those Memory Nodes allowed in that task's cpuset.
- The root cpuset contains all the systems CPUs and Memory Nodes.
- For any cpuset, one can define child cpusets containing a subset of the parents CPU and Memory Node resources.
- The hierarchy of cpusets can be mounted at /dev/cpuset, for browsing and manipulation from user space.
- A cpuset may be marked exclusive, which ensures that no other cpuset (except direct ancestors and descendants) may contain any overlapping CPUs or Memory Nodes.
- You can list all the tasks (by pid) attached to any cpuset.

The implementation of cpusets requires a few, simple hooks into the rest of the kernel, none in performance critical paths:

- in init/main.c, to initialize the root cpuset at system boot.
- in fork and exit, to attach and detach a task from its cpuset.
- in sched_setaffinity, to mask the requested CPUs by what's allowed in that task's cpuset.

- in sched.c migrate_live_tasks(), to keep migrating tasks within the CPUs allowed by their cpuset, if possible.
- in the mbind and set_mempolicy system calls, to mask the requested Memory Nodes by what's allowed in that task's cpuset.
- in page_alloc.c, to restrict memory to allowed nodes.
- in vmscan.c, to restrict page recovery to the current cpuset.

You should mount the ”cgroup” filesystem type in order to enable browsing and modifying the cpusets presently known to the kernel. No new system calls are added for cpusets - all support for querying and modifying cpusets is via this cpuset file system.

The /proc/<pid>/status file for each task has four added lines, displaying the task's cpus_allowed (on which CPUs it may be scheduled) and mems_allowed (on which Memory Nodes it may obtain memory), in the two formats seen in the following example:

```
Cpus_allowed:   ffffffff,ffffffff,ffffffff,ffffffff
Cpus_allowed_list:      0-127
Mems_allowed:   ffffffff,ffffffff
Mems_allowed_list:      0-63
```

Each cpuset is represented by a directory in the cgroup file system containing (on top of the standard cgroup files) the following files describing that cpuset:

- cpuset.cpus: list of CPUs in that cpuset
- cpuset.mems: list of Memory Nodes in that cpuset
- cpuset.memory_migrate flag: if set, move pages to cpusets nodes
- cpuset.cpu_exclusive flag: is cpu placement exclusive?
- cpuset.mem_exclusive flag: is memory placement exclusive?
- cpuset.mem_hardwall flag: is memory allocation hardwalled
- cpuset.memory_pressure: measure of how much paging pressure in cpuset
- cpuset.memory_spread_page flag: if set, spread page cache evenly on allowed nodes
- cpuset.memory_spread_slab flag: if set, spread slab cache evenly on allowed nodes
- cpuset.sched_load_balance flag: if set, load balance within CPUs on that cpuset
- cpuset.sched_relax_domain_level: the searching range when migrating tasks

In addition, only the root cpuset has the following file:

- cpuset.memory_pressure_enabled flag: compute memory_pressure?

New cpusets are created using the mkdir system call or shell command. The properties of a cpuset, such as its flags, allowed CPUs and Memory Nodes, and attached tasks, are modified by writing to the appropriate file in that cpusets directory, as listed above.

The named hierarchical structure of nested cpusets allows partitioning a large system into nested, dynamically changeable, ”soft-partitions”.

The attachment of each task, automatically inherited at fork by any children of that task, to a cpuset allows organizing the work load on a system into related sets of tasks such that each set is constrained to using the CPUs and Memory Nodes of a particular cpuset. A task may

be re-attached to any other cpuset, if allowed by the permissions on the necessary cpuset file system directories.

Such management of a system ”in the large” integrates smoothly with the detailed placement done on individual tasks and memory regions using the sched_setaffinity, mbind and set_mempolicy system calls.

The following rules apply to each cpuset:

- Its CPUs and Memory Nodes must be a subset of its parents.
- It can't be marked exclusive unless its parent is.
- If its cpu or memory is exclusive, they may not overlap any sibling.

These rules, and the natural hierarchy of cpusets, enable efficient enforcement of the exclusive guarantee, without having to scan all cpusets every time any of them change to ensure nothing overlaps a exclusive cpuset. Also, the use of a Linux virtual file system (vfs) to represent the cpuset hierarchy provides for a familiar permission and name space for cpusets, with a minimum of additional kernel code.

The cpus and mems files in the root (top_cpuset) cpuset are read-only. The cpus file automatically tracks the value of cpu_online_mask using a CPU hotplug notifier, and the mems file automatically tracks the value of node_states[N_MEMORY]--i.e., nodes with memory--using the cpuset_track_online_nodes() hook.

The cpuset.effective_cpus and cpuset.effective_mems files are normally read-only copies of cpuset.cpus and cpuset.mems files respectively. If the cpuset cgroup filesystem is mounted with the special ”cpuset_v2_mode” option, the behavior of these files will become similar to the corresponding files in cpuset v2. In other words, hotplug events will not change cpuset.cpus and cpuset.mems. Those events will only affect cpuset.effective_cpus and cpuset.effective_mems which show the actual cpus and memory nodes that are currently used by this cpuset. See *Control Group v2* for more information about cpuset v2 behavior.

1.4 What are exclusive cpusets ?

If a cpuset is cpu or mem exclusive, no other cpuset, other than a direct ancestor or descendant, may share any of the same CPUs or Memory Nodes.

A cpuset that is cpuset.mem_exclusive *or* cpuset.mem_hardwall is ”hardwalled”, i.e. it restricts kernel allocations for page, buffer and other data commonly shared by the kernel across multiple users. All cpusets, whether hardwalled or not, restrict allocations of memory for user space. This enables configuring a system so that several independent jobs can share common kernel data, such as file system pages, while isolating each job's user allocation in its own cpuset. To do this, construct a large mem_exclusive cpuset to hold all the jobs, and construct child, non-mem_exclusive cpusets for each individual job. Only a small amount of typical kernel memory, such as requests from interrupt handlers, is allowed to be taken outside even a mem_exclusive cpuset.

1.5 What is memory_pressure ?

The memory_pressure of a cpuset provides a simple per-cpuset metric of the rate that the tasks in a cpuset are attempting to free up in use memory on the nodes of the cpuset to satisfy additional memory requests.

This enables batch managers monitoring jobs running in dedicated cpusets to efficiently detect what level of memory pressure that job is causing.

This is useful both on tightly managed systems running a wide mix of submitted jobs, which may choose to terminate or re-prioritize jobs that are trying to use more memory than allowed on the nodes assigned to them, and with tightly coupled, long running, massively parallel scientific computing jobs that will dramatically fail to meet required performance goals if they start to use more memory than allowed to them.

This mechanism provides a very economical way for the batch manager to monitor a cpuset for signs of memory pressure. It's up to the batch manager or other user code to decide what to do about it and take action.

==>

> Unless this feature is enabled by writing "1" to the special file /dev/cpuset/memory_pressure_enabled, the hook in the rebalance code of __alloc_pages() for this metric reduces to simply noticing that the cpuset_memory_pressure_enabled flag is zero. So only systems that enable this feature will compute the metric.

Why a per-cpuset, running average:

> Because this meter is per-cpuset, rather than per-task or mm, the system load imposed by a batch scheduler monitoring this metric is sharply reduced on large systems, because a scan of the tasklist can be avoided on each set of queries.
>
> Because this meter is a running average, instead of an accumulating counter, a batch scheduler can detect memory pressure with a single read, instead of having to read and accumulate results for a period of time.
>
> Because this meter is per-cpuset rather than per-task or mm, the batch scheduler can obtain the key information, memory pressure in a cpuset, with a single read, rather than having to query and accumulate results over all the (dynamically changing) set of tasks in the cpuset.

A per-cpuset simple digital filter (requires a spinlock and 3 words of data per-cpuset) is kept, and updated by any task attached to that cpuset, if it enters the synchronous (direct) page reclaim code.

A per-cpuset file provides an integer number representing the recent (half-life of 10 seconds) rate of direct page reclaims caused by the tasks in the cpuset, in units of reclaims attempted per second, times 1000.

1.6 What is memory spread ?

There are two boolean flag files per cpuset that control where the kernel allocates pages for the file system buffers and related in kernel data structures. They are called 'cpuset.memory_spread_page' and 'cpuset.memory_spread_slab'.

If the per-cpuset boolean flag file 'cpuset.memory_spread_page' is set, then the kernel will spread the file system buffers (page cache) evenly over all the nodes that the faulting task is allowed to use, instead of preferring to put those pages on the node where the task is running.

If the per-cpuset boolean flag file 'cpuset.memory_spread_slab' is set, then the kernel will spread some file system related slab caches, such as for inodes and dentries evenly over all the nodes that the faulting task is allowed to use, instead of preferring to put those pages on the node where the task is running.

The setting of these flags does not affect anonymous data segment or stack segment pages of a task.

By default, both kinds of memory spreading are off, and memory pages are allocated on the node local to where the task is running, except perhaps as modified by the task's NUMA mempolicy or cpuset configuration, so long as sufficient free memory pages are available.

When new cpusets are created, they inherit the memory spread settings of their parent.

Setting memory spreading causes allocations for the affected page or slab caches to ignore the task's NUMA mempolicy and be spread instead. Tasks using mbind() or set_mempolicy() calls to set NUMA mempolicies will not notice any change in these calls as a result of their containing task's memory spread settings. If memory spreading is turned off, then the currently specified NUMA mempolicy once again applies to memory page allocations.

Both 'cpuset.memory_spread_page' and 'cpuset.memory_spread_slab' are boolean flag files. By default they contain ”0”, meaning that the feature is off for that cpuset. If a ”1” is written to that file, then that turns the named feature on.

The implementation is simple.

Setting the flag 'cpuset.memory_spread_page' turns on a per-process flag PFA_SPREAD_PAGE for each task that is in that cpuset or subsequently joins that cpuset. The page allocation calls for the page cache is modified to perform an inline check for this PFA_SPREAD_PAGE task flag, and if set, a call to a new routine cpuset_mem_spread_node() returns the node to prefer for the allocation.

Similarly, setting 'cpuset.memory_spread_slab' turns on the flag PFA_SPREAD_SLAB, and appropriately marked slab caches will allocate pages from the node returned by cpuset_mem_spread_node().

The cpuset_mem_spread_node() routine is also simple. It uses the value of a per-task rotor cpuset_mem_spread_rotor to select the next node in the current task's mems_allowed to prefer for the allocation.

This memory placement policy is also known (in other contexts) as round-robin or interleave.

This policy can provide substantial improvements for jobs that need to place thread local data on the corresponding node, but that need to access large file system data sets that need to be spread across the several nodes in the jobs cpuset in order to fit. Without this policy, especially for jobs that might have one thread reading in the data set, the memory allocation across the nodes in the jobs cpuset can become very uneven.

1.7 What is sched_load_balance ?

The kernel scheduler (kernel/sched/core.c) automatically load balances tasks. If one CPU is underutilized, kernel code running on that CPU will look for tasks on other more overloaded CPUs and move those tasks to itself, within the constraints of such placement mechanisms as cpusets and sched_setaffinity.

The algorithmic cost of load balancing and its impact on key shared kernel data structures such as the task list increases more than linearly with the number of CPUs being balanced. So the scheduler has support to partition the systems CPUs into a number of sched domains such that it only load balances within each sched domain. Each sched domain covers some subset of the CPUs in the system; no two sched domains overlap; some CPUs might not be in any sched domain and hence won't be load balanced.

Put simply, it costs less to balance between two smaller sched domains than one big one, but doing so means that overloads in one of the two domains won't be load balanced to the other one.

By default, there is one sched domain covering all CPUs, including those marked isolated using the kernel boot time ”isolcpus=” argument. However, the isolated CPUs will not participate in load balancing, and will not have tasks running on them unless explicitly assigned.

This default load balancing across all CPUs is not well suited for the following two situations:

1) On large systems, load balancing across many CPUs is expensive. If the system is managed using cpusets to place independent jobs on separate sets of CPUs, full load balancing is unnecessary.

2) Systems supporting realtime on some CPUs need to minimize system overhead on those CPUs, including avoiding task load balancing if that is not needed.

When the per-cpuset flag ”cpuset.sched_load_balance” is enabled (the default setting), it requests that all the CPUs in that cpusets allowed 'cpuset.cpus' be contained in a single sched domain, ensuring that load balancing can move a task (not otherwised pinned, as by sched_setaffinity) from any CPU in that cpuset to any other.

When the per-cpuset flag ”cpuset.sched_load_balance” is disabled, then the scheduler will avoid load balancing across the CPUs in that cpuset, --except-- in so far as is necessary because some overlapping cpuset has ”sched_load_balance” enabled.

So, for example, if the top cpuset has the flag ”cpuset.sched_load_balance” enabled, then the scheduler will have one sched domain covering all CPUs, and the setting of the ”cpuset.sched_load_balance” flag in any other cpusets won't matter, as we're already fully load balancing.

Therefore in the above two situations, the top cpuset flag ”cpuset.sched_load_balance” should be disabled, and only some of the smaller, child cpusets have this flag enabled.

When doing this, you don't usually want to leave any unpinned tasks in the top cpuset that might use non-trivial amounts of CPU, as such tasks may be artificially constrained to some subset of CPUs, depending on the particulars of this flag setting in descendant cpusets. Even if such a task could use spare CPU cycles in some other CPUs, the kernel scheduler might not consider the possibility of load balancing that task to that underused CPU.

Of course, tasks pinned to a particular CPU can be left in a cpuset that disables ”cpuset.sched_load_balance” as those tasks aren't going anywhere else anyway.

There is an impedance mismatch here, between cpusets and sched domains. Cpusets are hierarchical and nest. Sched domains are flat; they don't overlap and each CPU is in at most one sched domain.

It is necessary for sched domains to be flat because load balancing across partially overlapping sets of CPUs would risk unstable dynamics that would be beyond our understanding. So if each of two partially overlapping cpusets enables the flag 'cpuset.sched_load_balance', then we form a single sched domain that is a superset of both. We won't move a task to a CPU outside its cpuset, but the scheduler load balancing code might waste some compute cycles considering that possibility.

This mismatch is why there is not a simple one-to-one relation between which cpusets have the flag ”cpuset.sched_load_balance” enabled, and the sched domain configuration. If a cpuset enables the flag, it will get balancing across all its CPUs, but if it disables the flag, it will only be assured of no load balancing if no other overlapping cpuset enables the flag.

If two cpusets have partially overlapping 'cpuset.cpus' allowed, and only one of them has this flag enabled, then the other may find its tasks only partially load balanced, just on the overlapping CPUs. This is just the general case of the top_cpuset example given a few paragraphs above. In the general case, as in the top cpuset case, don't leave tasks that might use non-trivial amounts of CPU in such partially load balanced cpusets, as they may be artificially constrained to some subset of the CPUs allowed to them, for lack of load balancing to the other CPUs.

CPUs in ”cpuset.isolcpus” were excluded from load balancing by the isolcpus= kernel boot option, and will never be load balanced regardless of the value of ”cpuset.sched_load_balance” in any cpuset.

1.7.1 sched_load_balance implementation details.

The per-cpuset flag 'cpuset.sched_load_balance' defaults to enabled (contrary to most cpuset flags.) When enabled for a cpuset, the kernel will ensure that it can load balance across all the CPUs in that cpuset (makes sure that all the CPUs in the cpus_allowed of that cpuset are in the same sched domain.)

If two overlapping cpusets both have 'cpuset.sched_load_balance' enabled, then they will be (must be) both in the same sched domain.

If, as is the default, the top cpuset has 'cpuset.sched_load_balance' enabled, then by the above that means there is a single sched domain covering the whole system, regardless of any other cpuset settings.

The kernel commits to user space that it will avoid load balancing where it can. It will pick as fine a granularity partition of sched domains as it can while still providing load balancing for any set of CPUs allowed to a cpuset having 'cpuset.sched_load_balance' enabled.

The internal kernel cpuset to scheduler interface passes from the cpuset code to the scheduler code a partition of the load balanced CPUs in the system. This partition is a set of subsets (represented as an array of struct cpumask) of CPUs, pairwise disjoint, that cover all the CPUs that must be load balanced.

The cpuset code builds a new such partition and passes it to the scheduler sched domain setup code, to have the sched domains rebuilt as necessary, whenever:

- the 'cpuset.sched_load_balance' flag of a cpuset with non-empty CPUs changes,
- or CPUs come or go from a cpuset with this flag enabled,

- or 'cpuset.sched_relax_domain_level' value of a cpuset with non-empty CPUs and with this flag enabled changes,
- or a cpuset with non-empty CPUs and with this flag enabled is removed,
- or a cpu is offlined/onlined.

This partition exactly defines what sched domains the scheduler should setup - one sched domain for each element (struct cpumask) in the partition.

The scheduler remembers the currently active sched domain partitions. When the scheduler routine partition_sched_domains() is invoked from the cpuset code to update these sched domains, it compares the new partition requested with the current, and updates its sched domains, removing the old and adding the new, for each change.

1.8 What is sched_relax_domain_level ?

In sched domain, the scheduler migrates tasks in 2 ways; periodic load balance on tick, and at time of some schedule events.

When a task is woken up, scheduler try to move the task on idle CPU. For example, if a task A running on CPU X activates another task B on the same CPU X, and if CPU Y is X's sibling and performing idle, then scheduler migrate task B to CPU Y so that task B can start on CPU Y without waiting task A on CPU X.

And if a CPU run out of tasks in its runqueue, the CPU try to pull extra tasks from other busy CPUs to help them before it is going to be idle.

Of course it takes some searching cost to find movable tasks and/or idle CPUs, the scheduler might not search all CPUs in the domain every time. In fact, in some architectures, the searching ranges on events are limited in the same socket or node where the CPU locates, while the load balance on tick searches all.

For example, assume CPU Z is relatively far from CPU X. Even if CPU Z is idle while CPU X and the siblings are busy, scheduler can't migrate woken task B from X to Z since it is out of its searching range. As the result, task B on CPU X need to wait task A or wait load balance on the next tick. For some applications in special situation, waiting 1 tick may be too long.

The 'cpuset.sched_relax_domain_level' file allows you to request changing this searching range as you like. This file takes int value which indicates size of searching range in levels ideally as follows, otherwise initial value -1 that indicates the cpuset has no request.

-1	no request. use system default or follow request of others.
0	no search.
1	search siblings (hyperthreads in a core).
2	search cores in a package.
3	search cpus in a node [= system wide on non-NUMA system]
4	search nodes in a chunk of node [on NUMA system]
5	search system wide [on NUMA system]

The system default is architecture dependent. The system default can be changed using the relax_domain_level= boot parameter.

This file is per-cpuset and affect the sched domain where the cpuset belongs to. Therefore if the flag 'cpuset.sched_load_balance' of a cpuset is disabled, then 'cpuset.sched_relax_domain_level'

have no effect since there is no sched domain belonging the cpuset.

If multiple cpusets are overlapping and hence they form a single sched domain, the largest value among those is used. Be careful, if one requests 0 and others are -1 then 0 is used.

Note that modifying this file will have both good and bad effects, and whether it is acceptable or not depends on your situation. Don't modify this file if you are not sure.

If your situation is:

- The migration costs between each cpu can be assumed considerably small(for you) due to your special application's behavior or special hardware support for CPU cache etc.
- The searching cost doesn't have impact(for you) or you can make the searching cost enough small by managing cpuset to compact etc.
- The latency is required even it sacrifices cache hit rate etc. then increasing 'sched_relax_domain_level' would benefit you.

1.9 How do I use cpusets ?

In order to minimize the impact of cpusets on critical kernel code, such as the scheduler, and due to the fact that the kernel does not support one task updating the memory placement of another task directly, the impact on a task of changing its cpuset CPU or Memory Node placement, or of changing to which cpuset a task is attached, is subtle.

If a cpuset has its Memory Nodes modified, then for each task attached to that cpuset, the next time that the kernel attempts to allocate a page of memory for that task, the kernel will notice the change in the task's cpuset, and update its per-task memory placement to remain within the new cpusets memory placement. If the task was using mempolicy MPOL_BIND, and the nodes to which it was bound overlap with its new cpuset, then the task will continue to use whatever subset of MPOL_BIND nodes are still allowed in the new cpuset. If the task was using MPOL_BIND and now none of its MPOL_BIND nodes are allowed in the new cpuset, then the task will be essentially treated as if it was MPOL_BIND bound to the new cpuset (even though its NUMA placement, as queried by get_mempolicy(), doesn't change). If a task is moved from one cpuset to another, then the kernel will adjust the task's memory placement, as above, the next time that the kernel attempts to allocate a page of memory for that task.

If a cpuset has its 'cpuset.cpus' modified, then each task in that cpuset will have its allowed CPU placement changed immediately. Similarly, if a task's pid is written to another cpuset's 'tasks' file, then its allowed CPU placement is changed immediately. If such a task had been bound to some subset of its cpuset using the sched_setaffinity() call, the task will be allowed to run on any CPU allowed in its new cpuset, negating the effect of the prior sched_setaffinity() call.

In summary, the memory placement of a task whose cpuset is changed is updated by the kernel, on the next allocation of a page for that task, and the processor placement is updated immediately.

Normally, once a page is allocated (given a physical page of main memory) then that page stays on whatever node it was allocated, so long as it remains allocated, even if the cpusets memory placement policy 'cpuset.mems' subsequently changes. If the cpuset flag file 'cpuset.memory_migrate' is set true, then when tasks are attached to that cpuset, any pages that task had allocated to it on nodes in its previous cpuset are migrated to the task's new

cpuset. The relative placement of the page within the cpuset is preserved during these migration operations if possible. For example if the page was on the second valid node of the prior cpuset then the page will be placed on the second valid node of the new cpuset.

Also if 'cpuset.memory_migrate' is set true, then if that cpuset's 'cpuset.mems' file is modified, pages allocated to tasks in that cpuset, that were on nodes in the previous setting of 'cpuset.mems', will be moved to nodes in the new setting of 'mems.' Pages that were not in the task's prior cpuset, or in the cpuset's prior 'cpuset.mems' setting, will not be moved.

There is an exception to the above. If hotplug functionality is used to remove all the CPUs that are currently assigned to a cpuset, then all the tasks in that cpuset will be moved to the nearest ancestor with non-empty cpus. But the moving of some (or all) tasks might fail if cpuset is bound with another cgroup subsystem which has some restrictions on task attaching. In this failing case, those tasks will stay in the original cpuset, and the kernel will automatically update their cpus_allowed to allow all online CPUs. When memory hotplug functionality for removing Memory Nodes is available, a similar exception is expected to apply there as well. In general, the kernel prefers to violate cpuset placement, over starving a task that has had all its allowed CPUs or Memory Nodes taken offline.

There is a second exception to the above. GFP_ATOMIC requests are kernel internal allocations that must be satisfied, immediately. The kernel may drop some request, in rare cases even panic, if a GFP_ATOMIC alloc fails. If the request cannot be satisfied within the current task's cpuset, then we relax the cpuset, and look for memory anywhere we can find it. It's better to violate the cpuset than stress the kernel.

To start a new job that is to be contained within a cpuset, the steps are:

1) mkdir /sys/fs/cgroup/cpuset
2) mount -t cgroup -ocpuset cpuset /sys/fs/cgroup/cpuset
3) Create the new cpuset by doing mkdir's and write's (or echo's) in the /sys/fs/cgroup/cpuset virtual file system.
4) Start a task that will be the ”founding father” of the new job.
5) Attach that task to the new cpuset by writing its pid to the /sys/fs/cgroup/cpuset tasks file for that cpuset.
6) fork, exec or clone the job tasks from this founding father task.

For example, the following sequence of commands will setup a cpuset named ”Charlie”, containing just CPUs 2 and 3, and Memory Node 1, and then start a subshell 'sh' in that cpuset:

```
mount -t cgroup -ocpuset cpuset /sys/fs/cgroup/cpuset
cd /sys/fs/cgroup/cpuset
mkdir Charlie
cd Charlie
/bin/echo 2-3 > cpuset.cpus
/bin/echo 1 > cpuset.mems
/bin/echo $$ > tasks
sh
# The subshell 'sh' is now running in cpuset Charlie
# The next line should display '/Charlie'
cat /proc/self/cpuset
```

There are ways to query or modify cpusets:

- via the cpuset file system directly, using the various cd, mkdir, echo, cat, rmdir commands from the shell, or their equivalent from C.
- via the C library libcpuset.
- via the C library libcgroup. (https://github.com/libcgroup/libcgroup/)
- via the python application cset. (http://code.google.com/p/cpuset/)

The sched_setaffinity calls can also be done at the shell prompt using SGI's runon or Robert Love's taskset. The mbind and set_mempolicy calls can be done at the shell prompt using the numactl command (part of Andi Kleen's numa package).

32.4.2 2. Usage Examples and Syntax

2.1 Basic Usage

Creating, modifying, using the cpusets can be done through the cpuset virtual filesystem.

To mount it, type: # mount -t cgroup -o cpuset cpuset /sys/fs/cgroup/cpuset

Then under /sys/fs/cgroup/cpuset you can find a tree that corresponds to the tree of the cpusets in the system. For instance, /sys/fs/cgroup/cpuset is the cpuset that holds the whole system.

If you want to create a new cpuset under /sys/fs/cgroup/cpuset:

```
# cd /sys/fs/cgroup/cpuset
# mkdir my_cpuset
```

Now you want to do something with this cpuset:

```
# cd my_cpuset
```

In this directory you can find several files:

```
# ls
cgroup.clone_children  cpuset.memory_pressure
cgroup.event_control   cpuset.memory_spread_page
cgroup.procs           cpuset.memory_spread_slab
cpuset.cpu_exclusive   cpuset.mems
cpuset.cpus            cpuset.sched_load_balance
cpuset.mem_exclusive   cpuset.sched_relax_domain_level
cpuset.mem_hardwall    notify_on_release
cpuset.memory_migrate  tasks
```

Reading them will give you information about the state of this cpuset: the CPUs and Memory Nodes it can use, the processes that are using it, its properties. By writing to these files you can manipulate the cpuset.

Set some flags:

```
# /bin/echo 1 > cpuset.cpu_exclusive
```

Add some cpus:

```
# /bin/echo 0-7 > cpuset.cpus
```

Add some mems:

```
# /bin/echo 0-7 > cpuset.mems
```

Now attach your shell to this cpuset:

```
# /bin/echo $$ > tasks
```

You can also create cpusets inside your cpuset by using mkdir in this directory:

```
# mkdir my_sub_cs
```

To remove a cpuset, just use rmdir:

```
# rmdir my_sub_cs
```

This will fail if the cpuset is in use (has cpusets inside, or has processes attached).

Note that for legacy reasons, the "cpuset" filesystem exists as a wrapper around the cgroup filesystem.

The command:

```
mount -t cpuset X /sys/fs/cgroup/cpuset
```

is equivalent to:

```
mount -t cgroup -ocpuset,noprefix X /sys/fs/cgroup/cpuset
echo "/sbin/cpuset_release_agent" > /sys/fs/cgroup/cpuset/release_agent
```

2.2 Adding/removing cpus

This is the syntax to use when writing in the cpus or mems files in cpuset directories:

```
# /bin/echo 1-4 > cpuset.cpus         -> set cpus list to cpus 1,2,3,4
# /bin/echo 1,2,3,4 > cpuset.cpus     -> set cpus list to cpus 1,2,3,4
```

To add a CPU to a cpuset, write the new list of CPUs including the CPU to be added. To add 6 to the above cpuset:

```
# /bin/echo 1-4,6 > cpuset.cpus       -> set cpus list to cpus 1,2,3,4,6
```

Similarly to remove a CPU from a cpuset, write the new list of CPUs without the CPU to be removed.

To remove all the CPUs:

```
# /bin/echo "" > cpuset.cpus          -> clear cpus list
```

2.3 Setting flags

The syntax is very simple:

```
# /bin/echo 1 > cpuset.cpu_exclusive  -> set flag 'cpuset.cpu_exclusive'
# /bin/echo 0 > cpuset.cpu_exclusive  -> unset flag 'cpuset.cpu_exclusive'
```

2.4 Attaching processes

```
# /bin/echo PID > tasks
```

Note that it is PID, not PIDs. You can only attach ONE task at a time. If you have several tasks to attach, you have to do it one after another:

```
# /bin/echo PID1 > tasks
# /bin/echo PID2 > tasks
      ...
# /bin/echo PIDn > tasks
```

32.4.3 3. Questions

Q:
what's up with this '/bin/echo' ?

A:
bash's builtin 'echo' command does not check calls to write() against errors. If you use it in the cpuset file system, you won't be able to tell whether a command succeeded or failed.

Q:
When I attach processes, only the first of the line gets really attached !

A:
We can only return one error code per call to write(). So you should also put only ONE pid.

32.4.4 4. Contact

Web: http://www.bullopensource.org/cpuset

32.5 Device Whitelist Controller

32.5.1 1. Description

Implement a cgroup to track and enforce open and mknod restrictions on device files. A device cgroup associates a device access whitelist with each cgroup. A whitelist entry has 4 fields. 'type' is a (all), c (char), or b (block). 'all' means it applies to all types and all major and minor numbers. Major and minor are either an integer or * for all. Access is a composition of r (read), w (write), and m (mknod).

The root device cgroup starts with rwm to 'all'. A child device cgroup gets a copy of the parent. Administrators can then remove devices from the whitelist or add new entries. A child cgroup can never receive a device access which is denied by its parent.

32.5.2 2. User Interface

An entry is added using devices.allow, and removed using devices.deny. For instance:

```
echo 'c 1:3 mr' > /sys/fs/cgroup/1/devices.allow
```

allows cgroup 1 to read and mknod the device usually known as /dev/null. Doing:

```
echo a > /sys/fs/cgroup/1/devices.deny
```

will remove the default 'a *:* rwm' entry. Doing:

```
echo a > /sys/fs/cgroup/1/devices.allow
```

will add the 'a *:* rwm' entry to the whitelist.

32.5.3 3. Security

Any task can move itself between cgroups. This clearly won't suffice, but we can decide the best way to adequately restrict movement as people get some experience with this. We may just want to require CAP_SYS_ADMIN, which at least is a separate bit from CAP_MKNOD. We may want to just refuse moving to a cgroup which isn't a descendant of the current one. Or we may want to use CAP_MAC_ADMIN, since we really are trying to lock down root.

CAP_SYS_ADMIN is needed to modify the whitelist or move another task to a new cgroup. (Again we'll probably want to change that).

A cgroup may not be granted more permissions than the cgroup's parent has.

32.5.4 4. Hierarchy

device cgroups maintain hierarchy by making sure a cgroup never has more access permissions than its parent. Every time an entry is written to a cgroup's devices.deny file, all its children will have that entry removed from their whitelist and all the locally set whitelist entries will be re-evaluated. In case one of the locally set whitelist entries would provide more access than the cgroup's parent, it'll be removed from the whitelist.

Example:

```
  A
 / \
    B

group        behavior       exceptions
A            allow          "b 8:* rwm", "c 116:1 rw"
B            deny           "c 1:3 rwm", "c 116:2 rwm", "b 3:* rwm"
```

If a device is denied in group A:

```
# echo "c 116:* r" > A/devices.deny
```

it'll propagate down and after revalidating B's entries, the whitelist entry ”c 116:2 rwm” will be removed:

```
group        whitelist entries                        denied devices
A            all                                      "b 8:* rwm", "c 116:* rw"
B            "c 1:3 rwm", "b 3:* rwm"                 all the rest
```

In case parent's exceptions change and local exceptions are not allowed anymore, they'll be deleted.

Notice that new whitelist entries will not be propagated:

```
  A
 / \
    B

group        whitelist entries                        denied devices
A            "c 1:3 rwm", "c 1:5 r"                   all the rest
B            "c 1:3 rwm", "c 1:5 r"                   all the rest
```

when adding `c *:3 rwm`:

```
# echo "c *:3 rwm" >A/devices.allow
```

the result:

```
group        whitelist entries                        denied devices
A            "c *:3 rwm", "c 1:5 r"                   all the rest
B            "c 1:3 rwm", "c 1:5 r"                   all the rest
```

but now it'll be possible to add new entries to B:

```
# echo "c 2:3 rwm" >B/devices.allow
# echo "c 50:3 r" >B/devices.allow
```

or even:

```
# echo "c *:3 rwm" >B/devices.allow
```

Allowing or denying all by writing 'a' to devices.allow or devices.deny will not be possible once the device cgroups has children.

4.1 Hierarchy (internal implementation)

device cgroups is implemented internally using a behavior (ALLOW, DENY) and a list of exceptions. The internal state is controlled using the same user interface to preserve compatibility with the previous whitelist-only implementation. Removal or addition of exceptions that will reduce the access to devices will be propagated down the hierarchy. For every propagated exception, the effective rules will be re-evaluated based on current parent's access rules.

32.6 Cgroup Freezer

The cgroup freezer is useful to batch job management system which start and stop sets of tasks in order to schedule the resources of a machine according to the desires of a system administrator. This sort of program is often used on HPC clusters to schedule access to the cluster as a whole. The cgroup freezer uses cgroups to describe the set of tasks to be started/stopped by the batch job management system. It also provides a means to start and stop the tasks composing the job.

The cgroup freezer will also be useful for checkpointing running groups of tasks. The freezer allows the checkpoint code to obtain a consistent image of the tasks by attempting to force the tasks in a cgroup into a quiescent state. Once the tasks are quiescent another task can walk /proc or invoke a kernel interface to gather information about the quiesced tasks. Checkpointed tasks can be restarted later should a recoverable error occur. This also allows the checkpointed tasks to be migrated between nodes in a cluster by copying the gathered information to another node and restarting the tasks there.

Sequences of SIGSTOP and SIGCONT are not always sufficient for stopping and resuming tasks in userspace. Both of these signals are observable from within the tasks we wish to freeze. While SIGSTOP cannot be caught, blocked, or ignored it can be seen by waiting or ptracing parent tasks. SIGCONT is especially unsuitable since it can be caught by the task. Any programs designed to watch for SIGSTOP and SIGCONT could be broken by attempting to use SIGSTOP and SIGCONT to stop and resume tasks. We can demonstrate this problem using nested bash shells:

```
$ echo $$
16644
$ bash
$ echo $$
16690

From a second, unrelated bash shell:
$ kill -SIGSTOP 16690
$ kill -SIGCONT 16690

<at this point 16690 exits and causes 16644 to exit too>
```

This happens because bash can observe both signals and choose how it responds to them.

Another example of a program which catches and responds to these signals is gdb. In fact any program designed to use ptrace is likely to have a problem with this method of stopping and resuming tasks.

In contrast, the cgroup freezer uses the kernel freezer code to prevent the freeze/unfreeze cycle

from becoming visible to the tasks being frozen. This allows the bash example above and gdb to run as expected.

The cgroup freezer is hierarchical. Freezing a cgroup freezes all tasks belonging to the cgroup and all its descendant cgroups. Each cgroup has its own state (self-state) and the state inherited from the parent (parent-state). Iff both states are THAWED, the cgroup is THAWED.

The following cgroupfs files are created by cgroup freezer.

- freezer.state: Read-write.

  When read, returns the effective state of the cgroup - "THAWED", "FREEZING" or "FROZEN". This is the combined self and parent-states. If any is freezing, the cgroup is freezing (FREEZING or FROZEN).

  FREEZING cgroup transitions into FROZEN state when all tasks belonging to the cgroup and its descendants become frozen. Note that a cgroup reverts to FREEZING from FROZEN after a new task is added to the cgroup or one of its descendant cgroups until the new task is frozen.

  When written, sets the self-state of the cgroup. Two values are allowed - "FROZEN" and "THAWED". If FROZEN is written, the cgroup, if not already freezing, enters FREEZING state along with all its descendant cgroups.

  If THAWED is written, the self-state of the cgroup is changed to THAWED. Note that the effective state may not change to THAWED if the parent-state is still freezing. If a cgroup's effective state becomes THAWED, all its descendants which are freezing because of the cgroup also leave the freezing state.

- freezer.self_freezing: Read only.

  Shows the self-state. 0 if the self-state is THAWED; otherwise, 1. This value is 1 iff the last write to freezer.state was "FROZEN".

- freezer.parent_freezing: Read only.

  Shows the parent-state. 0 if none of the cgroup's ancestors is frozen; otherwise, 1.

The root cgroup is non-freezable and the above interface files don't exist.

- Examples of usage:

```
# mkdir /sys/fs/cgroup/freezer
# mount -t cgroup -ofreezer freezer /sys/fs/cgroup/freezer
# mkdir /sys/fs/cgroup/freezer/0
# echo $some_pid > /sys/fs/cgroup/freezer/0/tasks
```

to get status of the freezer subsystem:

```
# cat /sys/fs/cgroup/freezer/0/freezer.state
THAWED
```

to freeze all tasks in the container:

```
# echo FROZEN > /sys/fs/cgroup/freezer/0/freezer.state
# cat /sys/fs/cgroup/freezer/0/freezer.state
FREEZING
```

```
# cat /sys/fs/cgroup/freezer/0/freezer.state
FROZEN
```

to unfreeze all tasks in the container:

```
# echo THAWED > /sys/fs/cgroup/freezer/0/freezer.state
# cat /sys/fs/cgroup/freezer/0/freezer.state
THAWED
```

This is the basic mechanism which should do the right thing for user space task in a simple scenario.

32.7 HugeTLB Controller

HugeTLB controller can be created by first mounting the cgroup filesystem.

mount -t cgroup -o hugetlb none /sys/fs/cgroup

With the above step, the initial or the parent HugeTLB group becomes visible at /sys/fs/cgroup. At bootup, this group includes all the tasks in the system. /sys/fs/cgroup/tasks lists the tasks in this cgroup.

New groups can be created under the parent group /sys/fs/cgroup:

```
# cd /sys/fs/cgroup
# mkdir g1
# echo $$ > g1/tasks
```

The above steps create a new group g1 and move the current shell process (bash) into it.

Brief summary of control files:

```
hugetlb.<hugepagesize>.rsvd.limit_in_bytes            # set/show limit of
 ↪"hugepagesize" hugetlb reservations
hugetlb.<hugepagesize>.rsvd.max_usage_in_bytes        # show max "hugepagesize
 ↪" hugetlb reservations and no-reserve faults
hugetlb.<hugepagesize>.rsvd.usage_in_bytes            # show current␣
 ↪reservations and no-reserve faults for "hugepagesize" hugetlb
hugetlb.<hugepagesize>.rsvd.failcnt                   # show the number of␣
 ↪allocation failure due to HugeTLB reservation limit
hugetlb.<hugepagesize>.limit_in_bytes                 # set/show limit of
 ↪"hugepagesize" hugetlb faults
hugetlb.<hugepagesize>.max_usage_in_bytes             # show max "hugepagesize
 ↪" hugetlb  usage recorded
hugetlb.<hugepagesize>.usage_in_bytes                 # show current usage for
 ↪"hugepagesize" hugetlb
hugetlb.<hugepagesize>.failcnt                        # show the number of␣
 ↪allocation failure due to HugeTLB usage limit
hugetlb.<hugepagesize>.numa_stat                      # show the numa␣
 ↪information of the hugetlb memory charged to this cgroup
```

For a system supporting three hugepage sizes (64k, 32M and 1G), the control files include:

```
hugetlb.1GB.limit_in_bytes
hugetlb.1GB.max_usage_in_bytes
hugetlb.1GB.numa_stat
hugetlb.1GB.usage_in_bytes
hugetlb.1GB.failcnt
hugetlb.1GB.rsvd.limit_in_bytes
hugetlb.1GB.rsvd.max_usage_in_bytes
hugetlb.1GB.rsvd.usage_in_bytes
hugetlb.1GB.rsvd.failcnt
hugetlb.64KB.limit_in_bytes
hugetlb.64KB.max_usage_in_bytes
hugetlb.64KB.numa_stat
hugetlb.64KB.usage_in_bytes
hugetlb.64KB.failcnt
hugetlb.64KB.rsvd.limit_in_bytes
hugetlb.64KB.rsvd.max_usage_in_bytes
hugetlb.64KB.rsvd.usage_in_bytes
hugetlb.64KB.rsvd.failcnt
hugetlb.32MB.limit_in_bytes
hugetlb.32MB.max_usage_in_bytes
hugetlb.32MB.numa_stat
hugetlb.32MB.usage_in_bytes
hugetlb.32MB.failcnt
hugetlb.32MB.rsvd.limit_in_bytes
hugetlb.32MB.rsvd.max_usage_in_bytes
hugetlb.32MB.rsvd.usage_in_bytes
hugetlb.32MB.rsvd.failcnt
```

1. Page fault accounting

hugetlb.<hugepagesize>.limit_in_bytes hugetlb.<hugepagesize>.max_usage_in_bytes hugetlb.<hugepagesize>.usage_in_bytes hugetlb.<hugepagesize>.failcnt

The HugeTLB controller allows users to limit the HugeTLB usage (page fault) per control group and enforces the limit during page fault. Since HugeTLB doesn't support page reclaim, enforcing the limit at page fault time implies that, the application will get SIGBUS signal if it tries to fault in HugeTLB pages beyond its limit. Therefore the application needs to know exactly how many HugeTLB pages it uses before hand, and the sysadmin needs to make sure that there are enough available on the machine for all the users to avoid processes getting SIGBUS.

2. Reservation accounting

hugetlb.<hugepagesize>.rsvd.limit_in_bytes hugetlb.<hugepagesize>.rsvd.max_usage_in_bytes hugetlb.<hugepagesize>.rsvd.usage_in_bytes hugetlb.<hugepagesize>.rsvd.failcnt

The HugeTLB controller allows to limit the HugeTLB reservations per control group and enforces the controller limit at reservation time and at the fault of HugeTLB memory for which no reservation exists. Since reservation limits are enforced at reservation time (on mmap or shget), reservation limits never causes the application to get SIGBUS signal if the memory was reserved before hand. For MAP_NORESERVE allocations, the reservation limit behaves the same as the fault limit, enforcing memory usage at fault time and causing the application to receive a SIGBUS if it's crossing its limit.

Reservation limits are superior to page fault limits described above, since reservation limits

are enforced at reservation time (on mmap or shget), and never causes the application to get SIGBUS signal if the memory was reserved before hand. This allows for easier fallback to alternatives such as non-HugeTLB memory for example. In the case of page fault accounting, it's very hard to avoid processes getting SIGBUS since the sysadmin needs precisely know the HugeTLB usage of all the tasks in the system and make sure there is enough pages to satisfy all requests. Avoiding tasks getting SIGBUS on overcommited systems is practically impossible with page fault accounting.

3. Caveats with shared memory

For shared HugeTLB memory, both HugeTLB reservation and page faults are charged to the first task that causes the memory to be reserved or faulted, and all subsequent uses of this reserved or faulted memory is done without charging.

Shared HugeTLB memory is only uncharged when it is unreserved or deallocated. This is usually when the HugeTLB file is deleted, and not when the task that caused the reservation or fault has exited.

4. Caveats with HugeTLB cgroup offline.

When a HugeTLB cgroup goes offline with some reservations or faults still charged to it, the behavior is as follows:

- The fault charges are charged to the parent HugeTLB cgroup (reparented),
- the reservation charges remain on the offline HugeTLB cgroup.

This means that if a HugeTLB cgroup gets offlined while there is still HugeTLB reservations charged to it, that cgroup persists as a zombie until all HugeTLB reservations are uncharged. HugeTLB reservations behave in this manner to match the memory controller whose cgroups also persist as zombie until all charged memory is uncharged. Also, the tracking of HugeTLB reservations is a bit more complex compared to the tracking of HugeTLB faults, so it is significantly harder to reparent reservations at offline time.

32.8 Memory Resource Controller(Memcg) Implementation Memo

Last Updated: 2010/2

Base Kernel Version: based on 2.6.33-rc7-mm(candidate for 34).

Because VM is getting complex (one of reasons is memcg...), memcg's behavior is complex. This is a document for memcg's internal behavior. Please note that implementation details can be changed.

(*) Topics on API should be in *Memory Resource Controller*)

32.8.1 0. How to record usage ?

2 objects are used.

page_cgroup ....an object per page.

Allocated at boot or memory hotplug. Freed at memory hot removal.

swap_cgroup ... an entry per swp_entry.

Allocated at swapon(). Freed at swapoff().

The page_cgroup has USED bit and double count against a page_cgroup never occurs. swap_cgroup is used only when a charged page is swapped-out.

32.8.2 1. Charge

a page/swp_entry may be charged (usage += PAGE_SIZE) at

mem_cgroup_try_charge()

32.8.3 2. Uncharge

a page/swp_entry may be uncharged (usage -= PAGE_SIZE) by

mem_cgroup_uncharge()
Called when a page's refcount goes down to 0.

mem_cgroup_uncharge_swap()
Called when swp_entry's refcnt goes down to 0. A charge against swap disappears.

32.8.4 3. charge-commit-cancel

Memcg pages are charged in two steps:

- mem_cgroup_try_charge()
- mem_cgroup_commit_charge() or mem_cgroup_cancel_charge()

At try_charge(), there are no flags to say ”this page is charged”. at this point, usage += PAGE_SIZE.

At commit(), the page is associated with the memcg.

At cancel(), simply usage -= PAGE_SIZE.

Under below explanation, we assume CONFIG_SWAP=y.

32.8.5 4. Anonymous

Anonymous page is newly allocated at

- page fault into MAP_ANONYMOUS mapping.
- Copy-On-Write.

4.1 Swap-in. At swap-in, the page is taken from swap-cache. There are 2 cases.

(a) If the SwapCache is newly allocated and read, it has no charges.

(b) If the SwapCache has been mapped by processes, it has been charged already.

4.2 Swap-out. At swap-out, typical state transition is below.

(a) add to swap cache. (marked as SwapCache) swp_entry's refcnt += 1.

(b) fully unmapped. swp_entry's refcnt += # of ptes.

(c) write back to swap.

(d) delete from swap cache. (remove from SwapCache) swp_entry's refcnt -= 1.

Finally, at task exit, (e) zap_pte() is called and swp_entry's refcnt -=1 -> 0.

32.8.6 5. Page Cache

Page Cache is charged at - filemap_add_folio().

The logic is very clear. (About migration, see below)

Note:
__remove_from_page_cache() is called by remove_from_page_cache() and __remove_mapping().

32.8.7 6. Shmem(tmpfs) Page Cache

The best way to understand shmem's page state transition is to read mm/shmem.c.

But brief explanation of the behavior of memcg around shmem will be helpful to understand the logic.

Shmem's page (just leaf page, not direct/indirect block) can be on

- radix-tree of shmem's inode.
- SwapCache.
- Both on radix-tree and SwapCache. This happens at swap-in and swap-out,

It's charged when...

- A new page is added to shmem's radix-tree.
- A swp page is read. (move a charge from swap_cgroup to page_cgroup)

32.8.8 7. Page Migration

mem_cgroup_migrate()

32.8.9 8. LRU

Each memcg has its own vector of LRUs (inactive anon, active anon, inactive file, active file, unevictable) of pages from each node, each LRU handled under a single lru_lock for that memcg and node.

32.8.10 9. Typical Tests.

Tests for racy cases.

9.1 Small limit to memcg.

When you do test to do racy case, it's good test to set memcg's limit to be very small rather than GB. Many races found in the test under xKB or xxMB limits.

(Memory behavior under GB and Memory behavior under MB shows very different situation.)

9.2 Shmem

Historically, memcg's shmem handling was poor and we saw some amount of troubles here. This is because shmem is page-cache but can be SwapCache. Test with shmem/tmpfs is always good test.

9.3 Migration

For NUMA, migration is an another special case. To do easy test, cpuset is useful. Following is a sample script to do migration:

```
mount -t cgroup -o cpuset none /opt/cpuset

mkdir /opt/cpuset/01
echo 1 > /opt/cpuset/01/cpuset.cpus
echo 0 > /opt/cpuset/01/cpuset.mems
echo 1 > /opt/cpuset/01/cpuset.memory_migrate
mkdir /opt/cpuset/02
echo 1 > /opt/cpuset/02/cpuset.cpus
echo 1 > /opt/cpuset/02/cpuset.mems
echo 1 > /opt/cpuset/02/cpuset.memory_migrate
```

In above set, when you moves a task from 01 to 02, page migration to node 0 to node 1 will occur. Following is a script to migrate all under cpuset.:

```
--
move_task()
{
for pid in $1
do
        /bin/echo $pid >$2/tasks 2>/dev/null
        echo -n $pid
        echo -n " "
done
echo END
}

G1_TASK=`cat ${G1}/tasks`
G2_TASK=`cat ${G2}/tasks`
move_task "${G1_TASK}" ${G2} &
--
```

9.4 Memory hotplug

memory hotplug test is one of good test.

to offline memory, do following:

```
# echo offline > /sys/devices/system/memory/memoryXXX/state
```

(XXX is the place of memory)

This is an easy way to test page migration, too.

9.5 nested cgroups

Use tests like the following for testing nested cgroups:

```
mkdir /opt/cgroup/01/child_a
mkdir /opt/cgroup/01/child_b

set limit to 01.
add limit to 01/child_b
run jobs under child_a and child_b
```

create/delete following groups at random while jobs are running:

```
/opt/cgroup/01/child_a/child_aa
/opt/cgroup/01/child_b/child_bb
/opt/cgroup/01/child_c
```

running new jobs in new group is also good.

9.6 Mount with other subsystems

Mounting with other subsystems is a good test because there is a race and lock dependency with other cgroup subsystems.

example:

```
# mount -t cgroup none /cgroup -o cpuset,memory,cpu,devices
```

and do task move, mkdir, rmdir etc...under this.

9.7 swapoff

Besides management of swap is one of complicated parts of memcg, call path of swap-in at swapoff is not same as usual swap-in path.. It's worth to be tested explicitly.

For example, test like following is good:

(Shell-A):

```
# mount -t cgroup none /cgroup -o memory
# mkdir /cgroup/test
# echo 40M > /cgroup/test/memory.limit_in_bytes
# echo 0 > /cgroup/test/tasks
```

Run malloc(100M) program under this. You'll see 60M of swaps.

(Shell-B):

```
# move all tasks in /cgroup/test to /cgroup
# /sbin/swapoff -a
# rmdir /cgroup/test
# kill malloc task.
```

Of course, tmpfs v.s. swapoff test should be tested, too.

9.8 OOM-Killer

Out-of-memory caused by memcg's limit will kill tasks under the memcg. When hierarchy is used, a task under hierarchy will be killed by the kernel.

In this case, panic_on_oom shouldn't be invoked and tasks in other groups shouldn't be killed.

It's not difficult to cause OOM under memcg as following.

Case A) when you can swapoff:

```
#swapoff -a
#echo 50M > /memory.limit_in_bytes
```

run 51M of malloc

Case B) when you use mem+swap limitation:

```
#echo 50M > memory.limit_in_bytes
#echo 50M > memory.memsw.limit_in_bytes
```

run 51M of malloc

9.9 Move charges at task migration

Charges associated with a task can be moved along with task migration.

(Shell-A):

```
#mkdir /cgroup/A
#echo $$ >/cgroup/A/tasks
```

run some programs which uses some amount of memory in /cgroup/A.

(Shell-B):

```
#mkdir /cgroup/B
#echo 1 >/cgroup/B/memory.move_charge_at_immigrate
#echo "pid of the program running in group A" >/cgroup/B/tasks
```

You can see charges have been moved by reading `*.usage_in_bytes` or memory.stat of both A and B.

See 8.2 of *Memory Resource Controller* to see what value should be written to move_charge_at_immigrate.

9.10 Memory thresholds

Memory controller implements memory thresholds using cgroups notification API. You can use tools/cgroup/cgroup_event_listener.c to test it.

(Shell-A) Create cgroup and run event listener:

```
# mkdir /cgroup/A
# ./cgroup_event_listener /cgroup/A/memory.usage_in_bytes 5M
```

(Shell-B) Add task to cgroup and try to allocate and free memory:

```
# echo $$ >/cgroup/A/tasks
# a="$(dd if=/dev/zero bs=1M count=10)"
# a=
```

You will see message from cgroup_event_listener every time you cross the thresholds.

Use /cgroup/A/memory.memsw.usage_in_bytes to test memsw thresholds.

It's good idea to test root cgroup as well.

32.9 Memory Resource Controller

Caution: This document is hopelessly outdated and it asks for a complete rewrite. It still contains a useful information so we are keeping it here but make sure to check the current code if you need a deeper understanding.

Note: The Memory Resource Controller has generically been referred to as the memory controller in this document. Do not confuse memory controller used here with the memory controller that is used in hardware.

Hint: When we mention a cgroup (cgroupfs's directory) with memory controller, we call it "memory cgroup". When you see git-log and source code, you'll see patch's title and function names tend to use "memcg". In this document, we avoid using it.

32.9.1 Benefits and Purpose of the memory controller

The memory controller isolates the memory behaviour of a group of tasks from the rest of the system. The article on LWN[12] mentions some probable uses of the memory controller. The memory controller can be used to

a. Isolate an application or a group of applications Memory-hungry applications can be isolated and limited to a smaller amount of memory.

b. Create a cgroup with a limited amount of memory; this can be used as a good alternative to booting with mem=XXXX.

c. Virtualization solutions can control the amount of memory they want to assign to a virtual machine instance.

d. A CD/DVD burner could control the amount of memory used by the rest of the system to ensure that burning does not fail due to lack of available memory.

e. There are several other use cases; find one or use the controller just for fun (to learn and hack on the VM subsystem).

Current Status: linux-2.6.34-mmotm(development version of 2010/April)

Features:

- accounting anonymous pages, file caches, swap caches usage and limiting them.
- pages are linked to per-memcg LRU exclusively, and there is no global LRU.
- optionally, memory+swap usage can be accounted and limited.
- hierarchical accounting
- soft limit
- moving (recharging) account at moving a task is selectable.

[12] Corbet, Jonathan, Controlling memory use in cgroups, http://lwn.net/Articles/243795/

- usage threshold notifier
- memory pressure notifier
- oom-killer disable knob and oom-notifier
- Root cgroup has no limit controls.

Kernel memory support is a work in progress, and the current version provides basically functionality. (See *section 2.7*)

Brief summary of control files.

tasks	attach a task(thread) and show list of threads
cgroup.procs	show list of processes
cgroup.event_control	an interface for event_fd() This knob is not available on CONFIG_PREEMPT_RT systems.
memory.usage_in_bytes	show current usage for memory (See 5.5 for details)
memory.memsw.usage_in_bytes	show current usage for memory+Swap (See 5.5 for details)
memory.limit_in_bytes	set/show limit of memory usage
memory.memsw.limit_in_bytes	set/show limit of memory+Swap usage
memory.failcnt	show the number of memory usage hits limits
memory.memsw.failcnt	show the number of memory+Swap hits limits
memory.max_usage_in_bytes	show max memory usage recorded
memory.memsw.max_usage_in_bytes	show max memory+Swap usage recorded
memory.soft_limit_in_bytes	set/show soft limit of memory usage This knob is not available on CONFIG_PREEMPT_RT systems.
memory.stat	show various statistics
memory.use_hierarchy	set/show hierarchical account enabled This knob is deprecated and shouldn't be used.
memory.force_empty	trigger forced page reclaim
memory.pressure_level	set memory pressure notifications
memory.swappiness	set/show swappiness parameter of vmscan (See sysctl's vm.swappiness)
memory.move_charge_at_immigrate	set/show controls of moving charges This knob is deprecated and shouldn't be used.
memory.oom_control	set/show oom controls.
memory.numa_stat	show the number of memory usage per numa node
memory.kmem.limit_in_bytes	Deprecated knob to set and read the kernel memory hard limit. Kernel hard limit is not supported since 5.16. Writing any value to do file will not have any effect same as if nokmem kernel parameter was specified. Kernel memory is still charged and reported by memory.kmem.usage_in_bytes.
memory.kmem.usage_in_bytes	show current kernel memory allocation
memory.kmem.failcnt	show the number of kernel memory usage hits limits
memory.kmem.max_usage_in_bytes	show max kernel memory usage recorded
memory.kmem.tcp.limit_in_bytes	set/show hard limit for tcp buf memory
memory.kmem.tcp.usage_in_bytes	show current tcp buf memory allocation
memory.kmem.tcp.failcnt	show the number of tcp buf memory usage hits limits
memory.kmem.tcp.max_usage_in_bytes	show max tcp buf memory usage recorded

32.9.2 1. History

The memory controller has a long history. A request for comments for the memory controller was posted by Balbir Singh[1]. At the time the RFC was posted there were several implementations for memory control. The goal of the RFC was to build consensus and agreement for the minimal features required for memory control. The first RSS controller was posted by Balbir Singh[2] in Feb 2007. Pavel Emelianov[3][4][5] has since posted three versions of the RSS controller. At OLS, at the resource management BoF, everyone suggested that we handle both page cache and RSS together. Another request was raised to allow user space handling of OOM. The current memory controller is at version 6; it combines both mapped (RSS) and unmapped Page Cache Control[11].

32.9.3 2. Memory Control

Memory is a unique resource in the sense that it is present in a limited amount. If a task requires a lot of CPU processing, the task can spread its processing over a period of hours, days, months or years, but with memory, the same physical memory needs to be reused to accomplish the task.

The memory controller implementation has been divided into phases. These are:

1. Memory controller
2. mlock(2) controller
3. Kernel user memory accounting and slab control
4. user mappings length controller

The memory controller is the first controller developed.

2.1. Design

The core of the design is a counter called the page_counter. The page_counter tracks the current memory usage and limit of the group of processes associated with the controller. Each cgroup has a memory controller specific data structure (mem_cgroup) associated with it.

[1] Singh, Balbir. RFC: Memory Controller, http://lwn.net/Articles/206697/

[2] Singh, Balbir. Memory Controller (RSS Control), http://lwn.net/Articles/222762/

[3] Emelianov, Pavel. Resource controllers based on process cgroups https://lore.kernel.org/r/45ED7DEC.7010403@sw.ru

[4] Emelianov, Pavel. RSS controller based on process cgroups (v2) https://lore.kernel.org/r/461A3010.90403@sw.ru

[5] Emelianov, Pavel. RSS controller based on process cgroups (v3) https://lore.kernel.org/r/465D9739.8070209@openvz.org

[11] Singh, Balbir. Memory controller introduction (v6), https://lore.kernel.org/r/20070817084228.26003.12568.sendpatchset@balbir-laptop

2.2. Accounting

Listing 1: Figure 1: Hierarchy of Accounting

```
            +--------------------+
            |  mem_cgroup        |
            |  (page_counter)    |
            +--------------------+
             /            ^      \
            /             |       \
       +---------------+  |        +---------------+
       | mm_struct     |  |....    | mm_struct     |
       |               |  |        |               |
       +---------------+  |        +---------------+
                          |
                          + --------------+
                                          |
       +---------------+           +------+--------+
       | page          +---------->  page_cgroup|
       |               |           |               |
       +---------------+           +---------------+
```

Figure 1 shows the important aspects of the controller

1. Accounting happens per cgroup
2. Each mm_struct knows about which cgroup it belongs to
3. Each page has a pointer to the page_cgroup, which in turn knows the cgroup it belongs to

The accounting is done as follows: mem_cgroup_charge_common() is invoked to set up the necessary data structures and check if the cgroup that is being charged is over its limit. If it is, then reclaim is invoked on the cgroup. More details can be found in the reclaim section of this document. If everything goes well, a page meta-data-structure called page_cgroup is updated. page_cgroup has its own LRU on cgroup. (*) page_cgroup structure is allocated at boot/memory-hotplug time.

2.2.1 Accounting details

All mapped anon pages (RSS) and cache pages (Page Cache) are accounted. Some pages which are never reclaimable and will not be on the LRU are not accounted. We just account pages under usual VM management.

RSS pages are accounted at page_fault unless they've already been accounted for earlier. A file page will be accounted for as Page Cache when it's inserted into inode (xarray). While it's mapped into the page tables of processes, duplicate accounting is carefully avoided.

An RSS page is unaccounted when it's fully unmapped. A PageCache page is unaccounted when it's removed from xarray. Even if RSS pages are fully unmapped (by kswapd), they may exist as SwapCache in the system until they are really freed. Such SwapCaches are also accounted. A swapped-in page is accounted after adding into swapcache.

Note: The kernel does swapin-readahead and reads multiple swaps at once. Since page's memcg recorded into swap whatever memsw enabled, the page will be accounted after swapin.

At page migration, accounting information is kept.

Note: we just account pages-on-LRU because our purpose is to control amount of used pages; not-on-LRU pages tend to be out-of-control from VM view.

2.3 Shared Page Accounting

Shared pages are accounted on the basis of the first touch approach. The cgroup that first touches a page is accounted for the page. The principle behind this approach is that a cgroup that aggressively uses a shared page will eventually get charged for it (once it is uncharged from the cgroup that brought it in -- this will happen on memory pressure).

But see *section 8.2* when moving a task to another cgroup, its pages may be recharged to the new cgroup, if move_charge_at_immigrate has been chosen.

2.4 Swap Extension

Swap usage is always recorded for each of cgroup. Swap Extension allows you to read and limit it.

When CONFIG_SWAP is enabled, following files are added.

- memory.memsw.usage_in_bytes.
- memory.memsw.limit_in_bytes.

memsw means memory+swap. Usage of memory+swap is limited by memsw.limit_in_bytes.

Example: Assume a system with 4G of swap. A task which allocates 6G of memory (by mistake) under 2G memory limitation will use all swap. In this case, setting memsw.limit_in_bytes=3G will prevent bad use of swap. By using the memsw limit, you can avoid system OOM which can be caused by swap shortage.

2.4.1 why 'memory+swap' rather than swap

The global LRU(kswapd) can swap out arbitrary pages. Swap-out means to move account from memory to swap...there is no change in usage of memory+swap. In other words, when we want to limit the usage of swap without affecting global LRU, memory+swap limit is better than just limiting swap from an OS point of view.

2.4.2. What happens when a cgroup hits memory.memsw.limit_in_bytes

When a cgroup hits memory.memsw.limit_in_bytes, it's useless to do swap-out in this cgroup. Then, swap-out will not be done by cgroup routine and file caches are dropped. But as mentioned above, global LRU can do swapout memory from it for sanity of the system's memory management state. You can't forbid it by cgroup.

2.5 Reclaim

Each cgroup maintains a per cgroup LRU which has the same structure as global VM. When a cgroup goes over its limit, we first try to reclaim memory from the cgroup so as to make space for the new pages that the cgroup has touched. If the reclaim is unsuccessful, an OOM routine is invoked to select and kill the bulkiest task in the cgroup. (See *10. OOM Control* below.)

The reclaim algorithm has not been modified for cgroups, except that pages that are selected for reclaiming come from the per-cgroup LRU list.

Note: Reclaim does not work for the root cgroup, since we cannot set any limits on the root cgroup.

Note: When panic_on_oom is set to "2", the whole system will panic.

When oom event notifier is registered, event will be delivered. (See *oom_control* section)

2.6 Locking

Lock order is as follows:

```
Page lock (PG_locked bit of page->flags)
  mm->page_table_lock or split pte_lock
    folio_memcg_lock (memcg->move_lock)
      mapping->i_pages lock
        lruvec->lru_lock.
```

Per-node-per-memcgroup LRU (cgroup's private LRU) is guarded by lruvec->lru_lock; PG_lru bit of page->flags is cleared before isolating a page from its LRU under lruvec->lru_lock.

2.7 Kernel Memory Extension

With the Kernel memory extension, the Memory Controller is able to limit the amount of kernel memory used by the system. Kernel memory is fundamentally different than user memory, since it can't be swapped out, which makes it possible to DoS the system by consuming too much of this precious resource.

Kernel memory accounting is enabled for all memory cgroups by default. But it can be disabled system-wide by passing cgroup.memory=nokmem to the kernel at boot time. In this case, kernel memory will not be accounted at all.

Kernel memory limits are not imposed for the root cgroup. Usage for the root cgroup may or may not be accounted. The memory used is accumulated into memory.kmem.usage_in_bytes, or in a separate counter when it makes sense. (currently only for tcp).

The main "kmem" counter is fed into the main counter, so kmem charges will also be visible from the user counter.

Currently no soft limit is implemented for kernel memory. It is future work to trigger slab reclaim when those limits are reached.

2.7.1 Current Kernel Memory resources accounted

stack pages:
every process consumes some stack pages. By accounting into kernel memory, we prevent new processes from being created when the kernel memory usage is too high.

slab pages:
pages allocated by the SLAB or SLUB allocator are tracked. A copy of each kmem_cache is created every time the cache is touched by the first time from inside the memcg. The creation is done lazily, so some objects can still be skipped while the cache is being created. All objects in a slab page should belong to the same memcg. This only fails to hold when a task is migrated to a different memcg during the page allocation by the cache.

sockets memory pressure:
some sockets protocols have memory pressure thresholds. The Memory Controller allows them to be controlled individually per cgroup, instead of globally.

tcp memory pressure:
sockets memory pressure for the tcp protocol.

2.7.2 Common use cases

Because the "kmem" counter is fed to the main user counter, kernel memory can never be limited completely independently of user memory. Say "U" is the user limit, and "K" the kernel limit. There are three possible ways limits can be set:

U != 0, K = unlimited:
This is the standard memcg limitation mechanism already present before kmem accounting. Kernel memory is completely ignored.

U != 0, K < U:
Kernel memory is a subset of the user memory. This setup is useful in deployments where the total amount of memory per-cgroup is overcommitted. Overcommitting kernel memory limits is definitely not recommended, since the box can still run out of non-reclaimable memory. In this case, the admin could set up K so that the sum of all groups is never greater than the total memory, and freely set U at the cost of his QoS.

> **Warning:** In the current implementation, memory reclaim will NOT be triggered for a cgroup when it hits K while staying below U, which makes this setup impractical.

U != 0, K >= U:
Since kmem charges will also be fed to the user counter and reclaim will be triggered for the cgroup for both kinds of memory. This setup gives the admin a unified view of memory, and it is also useful for people who just want to track kernel memory usage.

32.9.4 3. User Interface

To use the user interface:

1. Enable CONFIG_CGROUPS and CONFIG_MEMCG options
2. Prepare the cgroups (see *Why are cgroups needed?* for the background information):

```
# mount -t tmpfs none /sys/fs/cgroup
# mkdir /sys/fs/cgroup/memory
# mount -t cgroup none /sys/fs/cgroup/memory -o memory
```

3. Make the new group and move bash into it:

```
# mkdir /sys/fs/cgroup/memory/0
# echo $$ > /sys/fs/cgroup/memory/0/tasks
```

4. Since now we're in the 0 cgroup, we can alter the memory limit:

```
# echo 4M > /sys/fs/cgroup/memory/0/memory.limit_in_bytes
```

The limit can now be queried:

```
# cat /sys/fs/cgroup/memory/0/memory.limit_in_bytes
4194304
```

Note: We can use a suffix (k, K, m, M, g or G) to indicate values in kilo, mega or gigabytes. (Here, Kilo, Mega, Giga are Kibibytes, Mebibytes, Gibibytes.)

Note: We can write "-1" to reset the `*.limit_in_bytes(unlimited)`.

Note: We cannot set limits on the root cgroup any more.

We can check the usage:

```
# cat /sys/fs/cgroup/memory/0/memory.usage_in_bytes
1216512
```

A successful write to this file does not guarantee a successful setting of this limit to the value written into the file. This can be due to a number of factors, such as rounding up to page boundaries or the total availability of memory on the system. The user is required to re-read this file after a write to guarantee the value committed by the kernel:

```
# echo 1 > memory.limit_in_bytes
# cat memory.limit_in_bytes
4096
```

The memory.failcnt field gives the number of times that the cgroup limit was exceeded.

The memory.stat file gives accounting information. Now, the number of caches, RSS and Active pages/Inactive pages are shown.

32.9.5 4. Testing

For testing features and implementation, see memcg_test.txt.

Performance test is also important. To see pure memory controller's overhead, testing on tmpfs will give you good numbers of small overheads. Example: do kernel make on tmpfs.

Page-fault scalability is also important. At measuring parallel page fault test, multi-process test may be better than multi-thread test because it has noise of shared objects/status.

But the above two are testing extreme situations. Trying usual test under memory controller is always helpful.

4.1 Troubleshooting

Sometimes a user might find that the application under a cgroup is terminated by the OOM killer. There are several causes for this:

1. The cgroup limit is too low (just too low to do anything useful)
2. The user is using anonymous memory and swap is turned off or too low

A sync followed by echo 1 > /proc/sys/vm/drop_caches will help get rid of some of the pages cached in the cgroup (page cache pages).

To know what happens, disabling OOM_Kill as per *"10. OOM Control"* (below) and seeing what happens will be helpful.

4.2 Task migration

When a task migrates from one cgroup to another, its charge is not carried forward by default. The pages allocated from the original cgroup still remain charged to it, the charge is dropped when the page is freed or reclaimed.

You can move charges of a task along with task migration. See *8. "Move charges at task migration"*

4.3 Removing a cgroup

A cgroup can be removed by rmdir, but as discussed in *sections 4.1* and *4.2*, a cgroup might have some charge associated with it, even though all tasks have migrated away from it. (because we charge against pages, not against tasks.)

We move the stats to parent, and no change on the charge except uncharging from the child.

Charges recorded in swap information is not updated at removal of cgroup. Recorded information is discarded and a cgroup which uses swap (swapcache) will be charged as a new owner of it.

32.9.6 5. Misc. interfaces

5.1 force_empty

memory.force_empty interface is provided to make cgroup's memory usage empty. When writing anything to this:

```
# echo 0 > memory.force_empty
```

the cgroup will be reclaimed and as many pages reclaimed as possible.

The typical use case for this interface is before calling rmdir(). Though rmdir() offlines memcg, but the memcg may still stay there due to charged file caches. Some out-of-use page caches may keep charged until memory pressure happens. If you want to avoid that, force_empty will be useful.

5.2 stat file

memory.stat file includes following statistics:

- per-memory cgroup local status

cache	# of bytes of page cache memory.
rss	# of bytes of anonymous and swap cache memory (includes transparent hugepages).
rss_huge	# of bytes of anonymous transparent hugepages.
mapped_file	# of bytes of mapped file (includes tmpfs/shmem)
pgpgin	# of charging events to the memory cgroup. The charging event happens each time a page is accounted as either mapped anon page(RSS) or cache page(Page Cache) to the cgroup.
pgpgout	# of uncharging events to the memory cgroup. The uncharging event happens each time a page is unaccounted from the cgroup.
swap	# of bytes of swap usage
swapcached	# of bytes of swap cached in memory
dirty	# of bytes that are waiting to get written back to the disk.
writeback	# of bytes of file/anon cache that are queued for syncing to disk.
inactive_anon	# of bytes of anonymous and swap cache memory on inactive LRU list.
active_anon	# of bytes of anonymous and swap cache memory on active LRU list.
inactive_file	# of bytes of file-backed memory and MADV_FREE anonymous memory (LazyFree pages) on inactive LRU list.
active_file	# of bytes of file-backed memory on active LRU list.
unevictable	# of bytes of memory that cannot be reclaimed (mlocked etc).

- status considering hierarchy (see memory.use_hierarchy settings):

hierarchical_memory_limit	# of bytes of memory limit with regard to hierarchy under which the memory cgroup is
hierarchical_memsw_limit	# of bytes of memory+swap limit with regard to hierarchy under which memory cgroup is.
total_<counter>	# hierarchical version of <counter>, which in addition to the cgroup's own value includes the sum of all hierarchical children's values of <counter>, i.e. total_cache

- additional vm parameters (depends on CONFIG_DEBUG_VM):

recent_rotated_anon	VM internal parameter. (see mm/vmscan.c)
recent_rotated_file	VM internal parameter. (see mm/vmscan.c)
recent_scanned_anon	VM internal parameter. (see mm/vmscan.c)
recent_scanned_file	VM internal parameter. (see mm/vmscan.c)

Hint: recent_rotated means recent frequency of LRU rotation. recent_scanned means recent # of scans to LRU. showing for better debug please see the code for meanings.

Note: Only anonymous and swap cache memory is listed as part of 'rss' stat. This should not be confused with the true 'resident set size' or the amount of physical memory used by the cgroup.

'rss + mapped_file" will give you resident set size of cgroup.

(Note: file and shmem may be shared among other cgroups. In that case, mapped_file is accounted only when the memory cgroup is owner of page cache.)

5.3 swappiness

Overrides /proc/sys/vm/swappiness for the particular group. The tunable in the root cgroup corresponds to the global swappiness setting.

Please note that unlike during the global reclaim, limit reclaim enforces that 0 swappiness really prevents from any swapping even if there is a swap storage available. This might lead to memcg OOM killer if there are no file pages to reclaim.

5.4 failcnt

A memory cgroup provides memory.failcnt and memory.memsw.failcnt files. This failcnt(== failure count) shows the number of times that a usage counter hit its limit. When a memory cgroup hits a limit, failcnt increases and memory under it will be reclaimed.

You can reset failcnt by writing 0 to failcnt file:

```
# echo 0 > .../memory.failcnt
```

5.5 usage_in_bytes

For efficiency, as other kernel components, memory cgroup uses some optimization to avoid unnecessary cacheline false sharing. usage_in_bytes is affected by the method and doesn't show 'exact' value of memory (and swap) usage, it's a fuzz value for efficient access. (Of course, when necessary, it's synchronized.) If you want to know more exact memory usage, you should use RSS+CACHE(+SWAP) value in memory.stat(see 5.2).

5.6 numa_stat

This is similar to numa_maps but operates on a per-memcg basis. This is useful for providing visibility into the numa locality information within an memcg since the pages are allowed to be allocated from any physical node. One of the use cases is evaluating application performance by combining this information with the application's CPU allocation.

Each memcg's numa_stat file includes ”total”, ”file”, ”anon” and ”unevictable” per-node page counts including ”hierarchical_<counter>” which sums up all hierarchical children's values in addition to the memcg's own value.

The output format of memory.numa_stat is:

```
total=<total pages> N0=<node 0 pages> N1=<node 1 pages> ...
file=<total file pages> N0=<node 0 pages> N1=<node 1 pages> ...
anon=<total anon pages> N0=<node 0 pages> N1=<node 1 pages> ...
unevictable=<total anon pages> N0=<node 0 pages> N1=<node 1 pages> ...
hierarchical_<counter>=<counter pages> N0=<node 0 pages> N1=<node 1 pages> ...
```

The ”total” count is sum of file + anon + unevictable.

32.9.7 6. Hierarchy support

The memory controller supports a deep hierarchy and hierarchical accounting. The hierarchy is created by creating the appropriate cgroups in the cgroup filesystem. Consider for example, the following cgroup filesystem hierarchy:

```
    root
  /  |   \
 /   |    \
a    b     c
           | \
           |  \
           d   e
```

In the diagram above, with hierarchical accounting enabled, all memory usage of e, is accounted to its ancestors up until the root (i.e, c and root). If one of the ancestors goes over its limit, the reclaim algorithm reclaims from the tasks in the ancestor and the children of the ancestor.

6.1 Hierarchical accounting and reclaim

Hierarchical accounting is enabled by default. Disabling the hierarchical accounting is deprecated. An attempt to do it will result in a failure and a warning printed to dmesg.

For compatibility reasons writing 1 to memory.use_hierarchy will always pass:

```
# echo 1 > memory.use_hierarchy
```

32.9.8 7. Soft limits

Soft limits allow for greater sharing of memory. The idea behind soft limits is to allow control groups to use as much of the memory as needed, provided

a. There is no memory contention

b. They do not exceed their hard limit

When the system detects memory contention or low memory, control groups are pushed back to their soft limits. If the soft limit of each control group is very high, they are pushed back as much as possible to make sure that one control group does not starve the others of memory.

Please note that soft limits is a best-effort feature; it comes with no guarantees, but it does its best to make sure that when memory is heavily contended for, memory is allocated based on the soft limit hints/setup. Currently soft limit based reclaim is set up such that it gets invoked from balance_pgdat (kswapd).

7.1 Interface

Soft limits can be setup by using the following commands (in this example we assume a soft limit of 256 MiB):

```
# echo 256M > memory.soft_limit_in_bytes
```

If we want to change this to 1G, we can at any time use:

```
# echo 1G > memory.soft_limit_in_bytes
```

Note: Soft limits take effect over a long period of time, since they involve reclaiming memory for balancing between memory cgroups

Note: It is recommended to set the soft limit always below the hard limit, otherwise the hard limit will take precedence.

32.9.9 8. Move charges at task migration (DEPRECATED!)

THIS IS DEPRECATED!

It's expensive and unreliable! It's better practice to launch workload tasks directly from inside their target cgroup. Use dedicated workload cgroups to allow fine-grained policy adjustments without having to move physical pages between control domains.

Users can move charges associated with a task along with task migration, that is, uncharge task's pages from the old cgroup and charge them to the new cgroup. This feature is not supported in !CONFIG_MMU environments because of lack of page tables.

8.1 Interface

This feature is disabled by default. It can be enabled (and disabled again) by writing to memory.move_charge_at_immigrate of the destination cgroup.

If you want to enable it:

```
# echo (some positive value) > memory.move_charge_at_immigrate
```

Note: Each bits of move_charge_at_immigrate has its own meaning about what type of charges should be moved. See *section 8.2* for details.

Note: Charges are moved only when you move mm->owner, in other words, a leader of a thread group.

Note: If we cannot find enough space for the task in the destination cgroup, we try to make space by reclaiming memory. Task migration may fail if we cannot make enough space.

Note: It can take several seconds if you move charges much.

And if you want disable it again:

```
# echo 0 > memory.move_charge_at_immigrate
```

8.2 Type of charges which can be moved

Each bit in move_charge_at_immigrate has its own meaning about what type of charges should be moved. But in any case, it must be noted that an account of a page or a swap can be moved only when it is charged to the task's current (old) memory cgroup.

bit	what type of charges would be moved ?
0	A charge of an anonymous page (or swap of it) used by the target task. You must enable Swap Extension (see 2.4) to enable move of swap charges.
1	A charge of file pages (normal file, tmpfs file (e.g. ipc shared memory) and swaps of tmpfs file) mmapped by the target task. Unlike the case of anonymous pages, file pages (and swaps) in the range mmapped by the task will be moved even if the task hasn't done page fault, i.e. they might not be the task's "RSS", but other task's "RSS" that maps the same file. And mapcount of the page is ignored (the page can be moved even if page_mapcount(page) > 1). You must enable Swap Extension (see 2.4) to enable move of swap charges.

8.3 TODO

- All of moving charge operations are done under cgroup_mutex. It's not good behavior to hold the mutex too long, so we may need some trick.

32.9.10 9. Memory thresholds

Memory cgroup implements memory thresholds using the cgroups notification API (see cgroups.txt). It allows to register multiple memory and memsw thresholds and gets notifications when it crosses.

To register a threshold, an application must:

- create an eventfd using eventfd(2);
- open memory.usage_in_bytes or memory.memsw.usage_in_bytes;
- write string like "<event_fd> <fd of memory.usage_in_bytes> <threshold>" to cgroup.event_control.

Application will be notified through eventfd when memory usage crosses threshold in any direction.

It's applicable for root and non-root cgroup.

32.9.11 10. OOM Control

memory.oom_control file is for OOM notification and other controls.

Memory cgroup implements OOM notifier using the cgroup notification API (See cgroups.txt). It allows to register multiple OOM notification delivery and gets notification when OOM happens.

To register a notifier, an application must:

- create an eventfd using eventfd(2)
- open memory.oom_control file
- write string like "<event_fd> <fd of memory.oom_control>" to cgroup.event_control

The application will be notified through eventfd when OOM happens. OOM notification doesn't work for the root cgroup.

You can disable the OOM-killer by writing "1" to memory.oom_control file, as:

#echo 1 > memory.oom_control

If OOM-killer is disabled, tasks under cgroup will hang/sleep in memory cgroup's OOM-waitqueue when they request accountable memory.

For running them, you have to relax the memory cgroup's OOM status by

- enlarge limit or reduce usage.

To reduce usage,

- kill some tasks.
- move some tasks to other group with account migration.
- remove some files (on tmpfs?)

Then, stopped tasks will work again.

At reading, current status of OOM is shown.

- oom_kill_disable 0 or 1 (if 1, oom-killer is disabled)
- under_oom 0 or 1 (if 1, the memory cgroup is under OOM, tasks may be stopped.)
- oom_kill integer counter The number of processes belonging to this cgroup killed by any kind of OOM killer.

32.9.12 11. Memory Pressure

The pressure level notifications can be used to monitor the memory allocation cost; based on the pressure, applications can implement different strategies of managing their memory resources. The pressure levels are defined as following:

The ”low” level means that the system is reclaiming memory for new allocations. Monitoring this reclaiming activity might be useful for maintaining cache level. Upon notification, the program (typically ”Activity Manager”) might analyze vmstat and act in advance (i.e. prematurely shutdown unimportant services).

The ”medium” level means that the system is experiencing medium memory pressure, the system might be making swap, paging out active file caches, etc. Upon this event applications may decide to further analyze vmstat/zoneinfo/memcg or internal memory usage statistics and free any resources that can be easily reconstructed or re-read from a disk.

The ”critical” level means that the system is actively thrashing, it is about to out of memory (OOM) or even the in-kernel OOM killer is on its way to trigger. Applications should do whatever they can to help the system. It might be too late to consult with vmstat or any other statistics, so it's advisable to take an immediate action.

By default, events are propagated upward until the event is handled, i.e. the events are not pass-through. For example, you have three cgroups: A->B->C. Now you set up an event listener on cgroups A, B and C, and suppose group C experiences some pressure. In this situation, only group C will receive the notification, i.e. groups A and B will not receive it. This is done to avoid excessive ”broadcasting” of messages, which disturbs the system and which is especially bad if we are low on memory or thrashing. Group B, will receive notification only if there are no event listeners for group C.

There are three optional modes that specify different propagation behavior:

- ”default”: this is the default behavior specified above. This mode is the same as omitting the optional mode parameter, preserved by backwards compatibility.
- ”hierarchy”: events always propagate up to the root, similar to the default behavior, except that propagation continues regardless of whether there are event listeners at each level, with the ”hierarchy” mode. In the above example, groups A, B, and C will receive notification of memory pressure.
- ”local”: events are pass-through, i.e. they only receive notifications when memory pressure is experienced in the memcg for which the notification is registered. In the above example, group C will receive notification if registered for ”local” notification and the group experiences memory pressure. However, group B will never receive notification, regardless if there is an event listener for group C or not, if group B is registered for local notification.

The level and event notification mode (”hierarchy” or ”local”, if necessary) are specified by a comma-delimited string, i.e. ”low,hierarchy” specifies hierarchical, pass-through, notification for all ancestor memcgs. Notification that is the default, non pass-through behavior, does not specify a mode. ”medium,local” specifies pass-through notification for the medium level.

The file memory.pressure_level is only used to setup an eventfd. To register a notification, an application must:

- create an eventfd using eventfd(2);
- open memory.pressure_level;
- write string as ”<event_fd> <fd of memory.pressure_level> <level[,mode]>” to cgroup.event_control.

Application will be notified through eventfd when memory pressure is at the specific level (or higher). Read/write operations to memory.pressure_level are no implemented.

Test:

Here is a small script example that makes a new cgroup, sets up a memory limit, sets up a notification in the cgroup and then makes child cgroup experience a critical pressure:

```
# cd /sys/fs/cgroup/memory/
# mkdir foo
# cd foo
# cgroup_event_listener memory.pressure_level low,hierarchy &
# echo 8000000 > memory.limit_in_bytes
# echo 8000000 > memory.memsw.limit_in_bytes
# echo $$ > tasks
# dd if=/dev/zero | read x
```

(Expect a bunch of notifications, and eventually, the oom-killer will trigger.)

32.9.13 12. TODO

1. Make per-cgroup scanner reclaim not-shared pages first
2. Teach controller to account for shared-pages
3. Start reclamation in the background when the limit is not yet hit but the usage is getting closer

32.9.14 Summary

Overall, the memory controller has been a stable controller and has been commented and discussed quite extensively in the community.

32.9.15 References

6. Menage, Paul. Control Groups v10, http://lwn.net/Articles/236032/
7. Vaidyanathan, Srinivasan, Control Groups: Pagecache accounting and control subsystem (v3), http://lwn.net/Articles/235534/
8. Singh, Balbir. RSS controller v2 test results (lmbench), https://lore.kernel.org/r/464C95D4.7070806@linux.vnet.ibm.com
9. Singh, Balbir. RSS controller v2 AIM9 results https://lore.kernel.org/r/464D267A.50107@linux.vnet.ibm.com
10. Singh, Balbir. Memory controller v6 test results, https://lore.kernel.org/r/20070819094658.654.84837.sendpatchset@balbir-laptop

32.10 Misc controller

Please refer "Misc" documentation in *Control Group v2*

32.11 Network classifier cgroup

The Network classifier cgroup provides an interface to tag network packets with a class identifier (classid).

The Traffic Controller (tc) can be used to assign different priorities to packets from different cgroups. Also, Netfilter (iptables) can use this tag to perform actions on such packets.

Creating a net_cls cgroups instance creates a net_cls.classid file. This net_cls.classid value is initialized to 0.

You can write hexadecimal values to net_cls.classid; the format for these values is 0xAAAABBBB; AAAA is the major handle number and BBBB is the minor handle number. Reading net_cls.classid yields a decimal result.

Example:

```
mkdir /sys/fs/cgroup/net_cls
mount -t cgroup -onet_cls net_cls /sys/fs/cgroup/net_cls
mkdir /sys/fs/cgroup/net_cls/0
echo 0x100001 >  /sys/fs/cgroup/net_cls/0/net_cls.classid
```

- setting a 10:1 handle:

```
cat /sys/fs/cgroup/net_cls/0/net_cls.classid
1048577
```

- configuring tc:

```
tc qdisc add dev eth0 root handle 10: htb
tc class add dev eth0 parent 10: classid 10:1 htb rate 40mbit
```

- creating traffic class 10:1:

```
tc filter add dev eth0 parent 10: protocol ip prio 10 handle 1: cgroup
```

configuring iptables, basic example:

```
iptables -A OUTPUT -m cgroup ! --cgroup 0x100001 -j DROP
```

32.12 Network priority cgroup

The Network priority cgroup provides an interface to allow an administrator to dynamically set the priority of network traffic generated by various applications

Nominally, an application would set the priority of its traffic via the SO_PRIORITY socket option. This however, is not always possible because:

1) The application may not have been coded to set this value
2) The priority of application traffic is often a site-specific administrative decision rather than an application defined one.

This cgroup allows an administrator to assign a process to a group which defines the priority of egress traffic on a given interface. Network priority groups can be created by first mounting the cgroup filesystem:

```
# mount -t cgroup -onet_prio none /sys/fs/cgroup/net_prio
```

With the above step, the initial group acting as the parent accounting group becomes visible at '/sys/fs/cgroup/net_prio'. This group includes all tasks in the system. '/sys/fs/cgroup/net_prio/tasks' lists the tasks in this cgroup.

Each net_prio cgroup contains two files that are subsystem specific

net_prio.prioidx
This file is read-only, and is simply informative. It contains a unique integer value that the kernel uses as an internal representation of this cgroup.

net_prio.ifpriomap
This file contains a map of the priorities assigned to traffic originating from processes in

this group and egressing the system on various interfaces. It contains a list of tuples in the form <ifname priority>. Contents of this file can be modified by echoing a string into the file using the same tuple format. For example:

```
echo "eth0 5" > /sys/fs/cgroups/net_prio/iscsi/net_prio.ifpriomap
```

This command would force any traffic originating from processes belonging to the iscsi net_prio cgroup and egressing on interface eth0 to have the priority of said traffic set to the value 5. The parent accounting group also has a writeable 'net_prio.ifpriomap' file that can be used to set a system default priority.

Priorities are set immediately prior to queueing a frame to the device queueing discipline (qdisc) so priorities will be assigned prior to the hardware queue selection being made.

One usage for the net_prio cgroup is with mqprio qdisc allowing application traffic to be steered to hardware/driver based traffic classes. These mappings can then be managed by administrators or other networking protocols such as DCBX.

A new net_prio cgroup inherits the parent's configuration.

32.13 Process Number Controller

32.13.1 Abstract

The process number controller is used to allow a cgroup hierarchy to stop any new tasks from being fork()'d or clone()'d after a certain limit is reached.

Since it is trivial to hit the task limit without hitting any kmemcg limits in place, PIDs are a fundamental resource. As such, PID exhaustion must be preventable in the scope of a cgroup hierarchy by allowing resource limiting of the number of tasks in a cgroup.

32.13.2 Usage

In order to use the *pids* controller, set the maximum number of tasks in pids.max (this is not available in the root cgroup for obvious reasons). The number of processes currently in the cgroup is given by pids.current.

Organisational operations are not blocked by cgroup policies, so it is possible to have pids.current > pids.max. This can be done by either setting the limit to be smaller than pids.current, or attaching enough processes to the cgroup such that pids.current > pids.max. However, it is not possible to violate a cgroup policy through fork() or clone(). fork() and clone() will return -EAGAIN if the creation of a new process would cause a cgroup policy to be violated.

To set a cgroup to have no limit, set pids.max to ”max”. This is the default for all new cgroups (N.B. that PID limits are hierarchical, so the most stringent limit in the hierarchy is followed).

pids.current tracks all child cgroup hierarchies, so parent/pids.current is a superset of parent/child/pids.current.

The pids.events file contains event counters:

- max: Number of times fork failed because limit was hit.

32.13.3 Example

First, we mount the pids controller:

```
# mkdir -p /sys/fs/cgroup/pids
# mount -t cgroup -o pids none /sys/fs/cgroup/pids
```

Then we create a hierarchy, set limits and attach processes to it:

```
# mkdir -p /sys/fs/cgroup/pids/parent/child
# echo 2 > /sys/fs/cgroup/pids/parent/pids.max
# echo $$ > /sys/fs/cgroup/pids/parent/cgroup.procs
# cat /sys/fs/cgroup/pids/parent/pids.current
2
#
```

It should be noted that attempts to overcome the set limit (2 in this case) will fail:

```
# cat /sys/fs/cgroup/pids/parent/pids.current
2
# ( /bin/echo "Here's some processes for you." | cat )
sh: fork: Resource temporary unavailable
#
```

Even if we migrate to a child cgroup (which doesn't have a set limit), we will not be able to overcome the most stringent limit in the hierarchy (in this case, parent's):

```
# echo $$ > /sys/fs/cgroup/pids/parent/child/cgroup.procs
# cat /sys/fs/cgroup/pids/parent/pids.current
2
# cat /sys/fs/cgroup/pids/parent/child/pids.current
2
# cat /sys/fs/cgroup/pids/parent/child/pids.max
max
# ( /bin/echo "Here's some processes for you." | cat )
sh: fork: Resource temporary unavailable
#
```

We can set a limit that is smaller than pids.current, which will stop any new processes from being forked at all (note that the shell itself counts towards pids.current):

```
# echo 1 > /sys/fs/cgroup/pids/parent/pids.max
# /bin/echo "We can't even spawn a single process now."
sh: fork: Resource temporary unavailable
# echo 0 > /sys/fs/cgroup/pids/parent/pids.max
# /bin/echo "We can't even spawn a single process now."
sh: fork: Resource temporary unavailable
#
```

32.14 RDMA Controller

32.14.1 1. Overview

1-1. What is RDMA controller?

RDMA controller allows user to limit RDMA/IB specific resources that a given set of processes can use. These processes are grouped using RDMA controller.

RDMA controller defines two resources which can be limited for processes of a cgroup.

1-2. Why RDMA controller needed?

Currently user space applications can easily take away all the rdma verb specific resources such as AH, CQ, QP, MR etc. Due to which other applications in other cgroup or kernel space ULPs may not even get chance to allocate any rdma resources. This can lead to service unavailability.

Therefore RDMA controller is needed through which resource consumption of processes can be limited. Through this controller different rdma resources can be accounted.

1-3. How is RDMA controller implemented?

RDMA cgroup allows limit configuration of resources. Rdma cgroup maintains resource accounting per cgroup, per device using resource pool structure. Each such resource pool is limited up to 64 resources in given resource pool by rdma cgroup, which can be extended later if required.

This resource pool object is linked to the cgroup css. Typically there are 0 to 4 resource pool instances per cgroup, per device in most use cases. But nothing limits to have it more. At present hundreds of RDMA devices per single cgroup may not be handled optimally, however there is no known use case or requirement for such configuration either.

Since RDMA resources can be allocated from any process and can be freed by any of the child processes which shares the address space, rdma resources are always owned by the creator cgroup css. This allows process migration from one to other cgroup without major complexity of transferring resource ownership; because such ownership is not really present due to shared nature of rdma resources. Linking resources around css also ensures that cgroups can be deleted after processes migrated. This allow progress migration as well with active resources, even though that is not a primary use case.

Whenever RDMA resource charging occurs, owner rdma cgroup is returned to the caller. Same rdma cgroup should be passed while uncharging the resource. This also allows process migrated with active RDMA resource to charge to new owner cgroup for new resource. It also allows to uncharge resource of a process from previously charged cgroup which is migrated to new cgroup, even though that is not a primary use case.

Resource pool object is created in following situations. (a) User sets the limit and no previous resource pool exist for the device of interest for the cgroup. (b) No resource limits were configured, but IB/RDMA stack tries to charge the resource. So that it correctly uncharge them when applications are running without limits and later on when limits are enforced during uncharging, otherwise usage count will drop to negative.

Resource pool is destroyed if all the resource limits are set to max and it is the last resource getting deallocated.

User should set all the limit to max value if it intents to remove/unconfigure the resource pool for a particular device.

IB stack honors limits enforced by the rdma controller. When application query about maximum resource limits of IB device, it returns minimum of what is configured by user for a given cgroup and what is supported by IB device.

Following resources can be accounted by rdma controller.

hca_handle	Maximum number of HCA Handles
hca_object	Maximum number of HCA Objects

32.14.2 2. Usage Examples

(a) Configure resource limit:

```
echo mlx4_0 hca_handle=2 hca_object=2000 > /sys/fs/cgroup/rdma/1/rdma.max
echo ocrdma1 hca_handle=3 > /sys/fs/cgroup/rdma/2/rdma.max
```

(b) Query resource limit:

```
cat /sys/fs/cgroup/rdma/2/rdma.max
#Output:
mlx4_0 hca_handle=2 hca_object=2000
ocrdma1 hca_handle=3 hca_object=max
```

(c) Query current usage:

```
cat /sys/fs/cgroup/rdma/2/rdma.current
#Output:
mlx4_0 hca_handle=1 hca_object=20
ocrdma1 hca_handle=1 hca_object=23
```

(d) Delete resource limit:

```
echo mlx4_0 hca_handle=max hca_object=max > /sys/fs/cgroup/rdma/1/rdma.max
```

CHAPTER

THIRTYTHREE

CONTROL GROUP V2

Date
October, 2015

Author
Tejun Heo <tj@kernel.org>

This is the authoritative documentation on the design, interface and conventions of cgroup v2. It describes all userland-visible aspects of cgroup including core and specific controller behaviors. All future changes must be reflected in this document. Documentation for v1 is available under *Documentation/admin-guide/cgroup-v1/index.rst*.

33.1 Introduction

33.1.1 Terminology

”cgroup” stands for ”control group” and is never capitalized. The singular form is used to designate the whole feature and also as a qualifier as in ”cgroup controllers”. When explicitly referring to multiple individual control groups, the plural form ”cgroups” is used.

33.1.2 What is cgroup?

cgroup is a mechanism to organize processes hierarchically and distribute system resources along the hierarchy in a controlled and configurable manner.

cgroup is largely composed of two parts - the core and controllers. cgroup core is primarily responsible for hierarchically organizing processes. A cgroup controller is usually responsible for distributing a specific type of system resource along the hierarchy although there are utility controllers which serve purposes other than resource distribution.

cgroups form a tree structure and every process in the system belongs to one and only one cgroup. All threads of a process belong to the same cgroup. On creation, all processes are put in the cgroup that the parent process belongs to at the time. A process can be migrated to another cgroup. Migration of a process doesn't affect already existing descendant processes.

Following certain structural constraints, controllers may be enabled or disabled selectively on a cgroup. All controller behaviors are hierarchical - if a controller is enabled on a cgroup, it affects all processes which belong to the cgroups consisting the inclusive sub-hierarchy of the cgroup. When a controller is enabled on a nested cgroup, it always restricts the resource distribution further. The restrictions set closer to the root in the hierarchy can not be overridden from further away.

33.2 Basic Operations

33.2.1 Mounting

Unlike v1, cgroup v2 has only single hierarchy. The cgroup v2 hierarchy can be mounted with the following mount command:

```
# mount -t cgroup2 none $MOUNT_POINT
```

cgroup2 filesystem has the magic number 0x63677270 (”cgrp”). All controllers which support v2 and are not bound to a v1 hierarchy are automatically bound to the v2 hierarchy and show up at the root. Controllers which are not in active use in the v2 hierarchy can be bound to other hierarchies. This allows mixing v2 hierarchy with the legacy v1 multiple hierarchies in a fully backward compatible way.

A controller can be moved across hierarchies only after the controller is no longer referenced in its current hierarchy. Because per-cgroup controller states are destroyed asynchronously and controllers may have lingering references, a controller may not show up immediately on the v2 hierarchy after the final umount of the previous hierarchy. Similarly, a controller should be fully disabled to be moved out of the unified hierarchy and it may take some time for the disabled controller to become available for other hierarchies; furthermore, due to inter-controller dependencies, other controllers may need to be disabled too.

While useful for development and manual configurations, moving controllers dynamically between the v2 and other hierarchies is strongly discouraged for production use. It is recommended to decide the hierarchies and controller associations before starting using the controllers after system boot.

During transition to v2, system management software might still automount the v1 cgroup filesystem and so hijack all controllers during boot, before manual intervention is possible. To make testing and experimenting easier, the kernel parameter cgroup_no_v1= allows disabling controllers in v1 and make them always available in v2.

cgroup v2 currently supports the following mount options.

nsdelegate
Consider cgroup namespaces as delegation boundaries. This option is system wide and can only be set on mount or modified through remount from the init namespace. The mount option is ignored on non-init namespace mounts. Please refer to the Delegation section for details.

favordynmods
Reduce the latencies of dynamic cgroup modifications such as task migrations and controller on/offs at the cost of making hot path operations such as forks and exits more expensive. The static usage pattern of creating a cgroup, enabling controllers, and then seeding it with CLONE_INTO_CGROUP is not affected by this option.

memory_localevents
Only populate memory.events with data for the current cgroup, and not any subtrees. This is legacy behaviour, the default behaviour without this option is to include subtree counts. This option is system wide and can only be set on mount or modified through remount from the init namespace. The mount option is ignored on non-init namespace mounts.

memory_recursiveprot
Recursively apply memory.min and memory.low protection to entire subtrees, without requiring explicit downward propagation into leaf cgroups. This allows protecting entire subtrees from one another, while retaining free competition within those subtrees. This should have been the default behavior but is a mount-option to avoid regressing setups relying on the original semantics (e.g. specifying bogusly high 'bypass' protection values at higher tree levels).

memory_hugetlb_accounting
Count HugeTLB memory usage towards the cgroup's overall memory usage for the memory controller (for the purpose of statistics reporting and memory protetion). This is a new behavior that could regress existing setups, so it must be explicitly opted in with this mount option.

A few caveats to keep in mind:

- There is no HugeTLB pool management involved in the memory controller. The pre-allocated pool does not belong to anyone. Specifically, when a new HugeTLB folio is allocated to the pool, it is not accounted for from the perspective of the memory controller. It is only charged to a cgroup when it is actually used (for e.g at page fault time). Host memory overcommit management has to consider this when configuring hard limits. In general, HugeTLB pool management should be done via other mechanisms (such as the HugeTLB controller).
- Failure to charge a HugeTLB folio to the memory controller results in SIGBUS. This could happen even if the HugeTLB pool still has pages available (but the cgroup limit is hit and reclaim attempt fails).
- Charging HugeTLB memory towards the memory controller affects memory protection and reclaim dynamics. Any userspace tuning (of low, min limits for e.g) needs to take this into account.
- HugeTLB pages utilized while this option is not selected will not be tracked by the memory controller (even if cgroup v2 is remounted later on).

33.2.2 Organizing Processes and Threads

Processes

Initially, only the root cgroup exists to which all processes belong. A child cgroup can be created by creating a sub-directory:

```
# mkdir $CGROUP_NAME
```

A given cgroup may have multiple child cgroups forming a tree structure. Each cgroup has a read-writable interface file ”cgroup.procs”. When read, it lists the PIDs of all processes which belong to the cgroup one-per-line. The PIDs are not ordered and the same PID may show up more than once if the process got moved to another cgroup and then back or the PID got recycled while reading.

A process can be migrated into a cgroup by writing its PID to the target cgroup's ”cgroup.procs” file. Only one process can be migrated on a single write(2) call. If a process is composed of multiple threads, writing the PID of any thread migrates all threads of the process.

When a process forks a child process, the new process is born into the cgroup that the forking process belongs to at the time of the operation. After exit, a process stays associated with the cgroup that it belonged to at the time of exit until it's reaped; however, a zombie process does not appear in ”cgroup.procs” and thus can't be moved to another cgroup.

A cgroup which doesn't have any children or live processes can be destroyed by removing the directory. Note that a cgroup which doesn't have any children and is associated only with zombie processes is considered empty and can be removed:

```
# rmdir $CGROUP_NAME
```

”/proc/$PID/cgroup” lists a process's cgroup membership. If legacy cgroup is in use in the system, this file may contain multiple lines, one for each hierarchy. The entry for cgroup v2 is always in the format ”0::$PATH”:

```
# cat /proc/842/cgroup
...
0::/test-cgroup/test-cgroup-nested
```

If the process becomes a zombie and the cgroup it was associated with is removed subsequently, ” (deleted)” is appended to the path:

```
# cat /proc/842/cgroup
...
0::/test-cgroup/test-cgroup-nested (deleted)
```

Threads

cgroup v2 supports thread granularity for a subset of controllers to support use cases requiring hierarchical resource distribution across the threads of a group of processes. By default, all threads of a process belong to the same cgroup, which also serves as the resource domain to host resource consumptions which are not specific to a process or thread. The thread mode allows threads to be spread across a subtree while still maintaining the common resource domain for them.

Controllers which support thread mode are called threaded controllers. The ones which don't are called domain controllers.

Marking a cgroup threaded makes it join the resource domain of its parent as a threaded cgroup. The parent may be another threaded cgroup whose resource domain is further up in the hierarchy. The root of a threaded subtree, that is, the nearest ancestor which is not threaded, is called threaded domain or thread root interchangeably and serves as the resource domain for the entire subtree.

Inside a threaded subtree, threads of a process can be put in different cgroups and are not subject to the no internal process constraint - threaded controllers can be enabled on non-leaf cgroups whether they have threads in them or not.

As the threaded domain cgroup hosts all the domain resource consumptions of the subtree, it is considered to have internal resource consumptions whether there are processes in it or not and can't have populated child cgroups which aren't threaded. Because the root cgroup is not subject to no internal process constraint, it can serve both as a threaded domain and a parent to domain cgroups.

The current operation mode or type of the cgroup is shown in the "cgroup.type" file which indicates whether the cgroup is a normal domain, a domain which is serving as the domain of a threaded subtree, or a threaded cgroup.

On creation, a cgroup is always a domain cgroup and can be made threaded by writing "threaded" to the "cgroup.type" file. The operation is single direction:

```
# echo threaded > cgroup.type
```

Once threaded, the cgroup can't be made a domain again. To enable the thread mode, the following conditions must be met.

- As the cgroup will join the parent's resource domain. The parent must either be a valid (threaded) domain or a threaded cgroup.
- When the parent is an unthreaded domain, it must not have any domain controllers enabled or populated domain children. The root is exempt from this requirement.

Topology-wise, a cgroup can be in an invalid state. Please consider the following topology:

```
A (threaded domain) - B (threaded) - C (domain, just created)
```

C is created as a domain but isn't connected to a parent which can host child domains. C can't be used until it is turned into a threaded cgroup. "cgroup.type" file will report "domain (invalid)" in these cases. Operations which fail due to invalid topology use EOPNOTSUPP as the errno.

A domain cgroup is turned into a threaded domain when one of its child cgroup becomes threaded or threaded controllers are enabled in the "cgroup.subtree_control" file while there are processes in the cgroup. A threaded domain reverts to a normal domain when the conditions clear.

When read, "cgroup.threads" contains the list of the thread IDs of all threads in the cgroup. Except that the operations are per-thread instead of per-process, "cgroup.threads" has the same format and behaves the same way as "cgroup.procs". While "cgroup.threads" can be written to in any cgroup, as it can only move threads inside the same threaded domain, its operations are confined inside each threaded subtree.

The threaded domain cgroup serves as the resource domain for the whole subtree, and, while the threads can be scattered across the subtree, all the processes are considered to be in the threaded domain cgroup. "cgroup.procs" in a threaded domain cgroup contains the PIDs of all processes in the subtree and is not readable in the subtree proper. However, "cgroup.procs" can be written to from anywhere in the subtree to migrate all threads of the matching process to the cgroup.

Only threaded controllers can be enabled in a threaded subtree. When a threaded controller is enabled inside a threaded subtree, it only accounts for and controls resource consumptions associated with the threads in the cgroup and its descendants. All consumptions which aren't tied to a specific thread belong to the threaded domain cgroup.

Because a threaded subtree is exempt from no internal process constraint, a threaded controller must be able to handle competition between threads in a non-leaf cgroup and its child cgroups. Each threaded controller defines how such competitions are handled.

Currently, the following controllers are threaded and can be enabled in a threaded cgroup:

```
- cpu
- cpuset
```

```
- perf_event
- pids
```

33.2.3 [Un]populated Notification

Each non-root cgroup has a ”cgroup.events” file which contains ”populated” field indicating whether the cgroup's sub-hierarchy has live processes in it. Its value is 0 if there is no live process in the cgroup and its descendants; otherwise, 1. poll and [id]notify events are triggered when the value changes. This can be used, for example, to start a clean-up operation after all processes of a given sub-hierarchy have exited. The populated state updates and notifications are recursive. Consider the following sub-hierarchy where the numbers in the parentheses represent the numbers of processes in each cgroup:

```
A(4) - B(0) - C(1)
            \ D(0)
```

A, B and C's ”populated” fields would be 1 while D's 0. After the one process in C exits, B and C's ”populated” fields would flip to ”0” and file modified events will be generated on the ”cgroup.events” files of both cgroups.

33.2.4 Controlling Controllers

Enabling and Disabling

Each cgroup has a ”cgroup.controllers” file which lists all controllers available for the cgroup to enable:

```
# cat cgroup.controllers
cpu io memory
```

No controller is enabled by default. Controllers can be enabled and disabled by writing to the ”cgroup.subtree_control” file:

```
# echo "+cpu +memory -io" > cgroup.subtree_control
```

Only controllers which are listed in ”cgroup.controllers” can be enabled. When multiple operations are specified as above, either they all succeed or fail. If multiple operations on the same controller are specified, the last one is effective.

Enabling a controller in a cgroup indicates that the distribution of the target resource across its immediate children will be controlled. Consider the following sub-hierarchy. The enabled controllers are listed in parentheses:

```
A(cpu,memory) - B(memory) - C()
                          \ D()
```

As A has ”cpu” and ”memory” enabled, A will control the distribution of CPU cycles and memory to its children, in this case, B. As B has ”memory” enabled but not ”CPU”, C and D will compete freely on CPU cycles but their division of memory available to B will be controlled.

As a controller regulates the distribution of the target resource to the cgroup's children, enabling it creates the controller's interface files in the child cgroups. In the above example, enabling ”cpu” on B would create the ”cpu.” prefixed controller interface files in C and D. Likewise, disabling ”memory” from B would remove the ”memory.” prefixed controller interface files from C and D. This means that the controller interface files - anything which doesn't start with ”cgroup.” are owned by the parent rather than the cgroup itself.

Top-down Constraint

Resources are distributed top-down and a cgroup can further distribute a resource only if the resource has been distributed to it from the parent. This means that all non-root ”cgroup.subtree_control” files can only contain controllers which are enabled in the parent's ”cgroup.subtree_control” file. A controller can be enabled only if the parent has the controller enabled and a controller can't be disabled if one or more children have it enabled.

No Internal Process Constraint

Non-root cgroups can distribute domain resources to their children only when they don't have any processes of their own. In other words, only domain cgroups which don't contain any processes can have domain controllers enabled in their ”cgroup.subtree_control” files.

This guarantees that, when a domain controller is looking at the part of the hierarchy which has it enabled, processes are always only on the leaves. This rules out situations where child cgroups compete against internal processes of the parent.

The root cgroup is exempt from this restriction. Root contains processes and anonymous resource consumption which can't be associated with any other cgroups and requires special treatment from most controllers. How resource consumption in the root cgroup is governed is up to each controller (for more information on this topic please refer to the Non-normative information section in the Controllers chapter).

Note that the restriction doesn't get in the way if there is no enabled controller in the cgroup's ”cgroup.subtree_control”. This is important as otherwise it wouldn't be possible to create children of a populated cgroup. To control resource distribution of a cgroup, the cgroup must create children and transfer all its processes to the children before enabling controllers in its ”cgroup.subtree_control” file.

33.2.5 Delegation

Model of Delegation

A cgroup can be delegated in two ways. First, to a less privileged user by granting write access of the directory and its ”cgroup.procs”, ”cgroup.threads” and ”cgroup.subtree_control” files to the user. Second, if the ”nsdelegate” mount option is set, automatically to a cgroup namespace on namespace creation.

Because the resource control interface files in a given directory control the distribution of the parent's resources, the delegatee shouldn't be allowed to write to them. For the first method, this is achieved by not granting access to these files. For the second, the kernel rejects writes to all files other than ”cgroup.procs” and ”cgroup.subtree_control” on a namespace root from inside the namespace.

The end results are equivalent for both delegation types. Once delegated, the user can build sub-hierarchy under the directory, organize processes inside it as it sees fit and further distribute the resources it received from the parent. The limits and other settings of all resource controllers are hierarchical and regardless of what happens in the delegated sub-hierarchy, nothing can escape the resource restrictions imposed by the parent.

Currently, cgroup doesn't impose any restrictions on the number of cgroups in or nesting depth of a delegated sub-hierarchy; however, this may be limited explicitly in the future.

Delegation Containment

A delegated sub-hierarchy is contained in the sense that processes can't be moved into or out of the sub-hierarchy by the delegatee.

For delegations to a less privileged user, this is achieved by requiring the following conditions for a process with a non-root euid to migrate a target process into a cgroup by writing its PID to the ”cgroup.procs” file.

- The writer must have write access to the ”cgroup.procs” file.
- The writer must have write access to the ”cgroup.procs” file of the common ancestor of the source and destination cgroups.

The above two constraints ensure that while a delegatee may migrate processes around freely in the delegated sub-hierarchy it can't pull in from or push out to outside the sub-hierarchy.

For an example, let's assume cgroups C0 and C1 have been delegated to user U0 who created C00, C01 under C0 and C10 under C1 as follows and all processes under C0 and C1 belong to U0:

```
~~~~~~~~~~~~~ - C0 - C00
~ cgroup    ~      \ C01
~ hierarchy ~
~~~~~~~~~~~~~ - C1 - C10
```

Let's also say U0 wants to write the PID of a process which is currently in C10 into ”C00/cgroup.procs”. U0 has write access to the file; however, the common ancestor of the source cgroup C10 and the destination cgroup C00 is above the points of delegation and U0 would not have write access to its ”cgroup.procs” files and thus the write will be denied with -EACCES.

For delegations to namespaces, containment is achieved by requiring that both the source and destination cgroups are reachable from the namespace of the process which is attempting the migration. If either is not reachable, the migration is rejected with -ENOENT.

33.2.6 Guidelines

Organize Once and Control

Migrating a process across cgroups is a relatively expensive operation and stateful resources such as memory are not moved together with the process. This is an explicit design decision as there often exist inherent trade-offs between migration and various hot paths in terms of synchronization cost.

As such, migrating processes across cgroups frequently as a means to apply different resource restrictions is discouraged. A workload should be assigned to a cgroup according to the system's logical and resource structure once on start-up. Dynamic adjustments to resource distribution can be made by changing controller configuration through the interface files.

Avoid Name Collisions

Interface files for a cgroup and its children cgroups occupy the same directory and it is possible to create children cgroups which collide with interface files.

All cgroup core interface files are prefixed with ”cgroup.” and each controller's interface files are prefixed with the controller name and a dot. A controller's name is composed of lower case alphabets and '_'s but never begins with an '_' so it can be used as the prefix character for collision avoidance. Also, interface file names won't start or end with terms which are often used in categorizing workloads such as job, service, slice, unit or workload.

cgroup doesn't do anything to prevent name collisions and it's the user's responsibility to avoid them.

33.3 Resource Distribution Models

cgroup controllers implement several resource distribution schemes depending on the resource type and expected use cases. This section describes major schemes in use along with their expected behaviors.

33.3.1 Weights

A parent's resource is distributed by adding up the weights of all active children and giving each the fraction matching the ratio of its weight against the sum. As only children which can make use of the resource at the moment participate in the distribution, this is work-conserving. Due to the dynamic nature, this model is usually used for stateless resources.

All weights are in the range [1, 10000] with the default at 100. This allows symmetric multiplicative biases in both directions at fine enough granularity while staying in the intuitive range.

As long as the weight is in range, all configuration combinations are valid and there is no reason to reject configuration changes or process migrations.

”cpu.weight” proportionally distributes CPU cycles to active children and is an example of this type.

33.3.2 Limits

A child can only consume up to the configured amount of the resource. Limits can be over-committed - the sum of the limits of children can exceed the amount of resource available to the parent.

Limits are in the range [0, max] and defaults to ”max”, which is noop.

As limits can be over-committed, all configuration combinations are valid and there is no reason to reject configuration changes or process migrations.

”io.max” limits the maximum BPS and/or IOPS that a cgroup can consume on an IO device and is an example of this type.

33.3.3 Protections

A cgroup is protected up to the configured amount of the resource as long as the usages of all its ancestors are under their protected levels. Protections can be hard guarantees or best effort soft boundaries. Protections can also be over-committed in which case only up to the amount available to the parent is protected among children.

Protections are in the range [0, max] and defaults to 0, which is noop.

As protections can be over-committed, all configuration combinations are valid and there is no reason to reject configuration changes or process migrations.

”memory.low” implements best-effort memory protection and is an example of this type.

33.3.4 Allocations

A cgroup is exclusively allocated a certain amount of a finite resource. Allocations can't be over-committed - the sum of the allocations of children can not exceed the amount of resource available to the parent.

Allocations are in the range [0, max] and defaults to 0, which is no resource.

As allocations can't be over-committed, some configuration combinations are invalid and should be rejected. Also, if the resource is mandatory for execution of processes, process migrations may be rejected.

”cpu.rt.max” hard-allocates realtime slices and is an example of this type.

33.4 Interface Files

33.4.1 Format

All interface files should be in one of the following formats whenever possible:

```
New-line separated values
(when only one value can be written at once)

      VAL0\n
      VAL1\n
```

```
      ...

Space separated values
(when read-only or multiple values can be written at once)

      VAL0 VAL1 ...\n

Flat keyed

      KEY0 VAL0\n
      KEY1 VAL1\n
      ...

Nested keyed

      KEY0 SUB_KEY0=VAL00 SUB_KEY1=VAL01...
      KEY1 SUB_KEY0=VAL10 SUB_KEY1=VAL11...
      ...
```

For a writable file, the format for writing should generally match reading; however, controllers may allow omitting later fields or implement restricted shortcuts for most common use cases.

For both flat and nested keyed files, only the values for a single key can be written at a time. For nested keyed files, the sub key pairs may be specified in any order and not all pairs have to be specified.

33.4.2 Conventions

- Settings for a single feature should be contained in a single file.
- The root cgroup should be exempt from resource control and thus shouldn't have resource control interface files.
- The default time unit is microseconds. If a different unit is ever used, an explicit unit suffix must be present.
- A parts-per quantity should use a percentage decimal with at least two digit fractional part - e.g. 13.40.
- If a controller implements weight based resource distribution, its interface file should be named "weight" and have the range [1, 10000] with 100 as the default. The values are chosen to allow enough and symmetric bias in both directions while keeping it intuitive (the default is 100%).
- If a controller implements an absolute resource guarantee and/or limit, the interface files should be named "min" and "max" respectively. If a controller implements best effort resource guarantee and/or limit, the interface files should be named "low" and "high" respectively.

  In the above four control files, the special token "max" should be used to represent upward infinity for both reading and writing.
- If a setting has a configurable default value and keyed specific overrides, the default entry should be keyed with "default" and appear as the first entry in the file.

The default value can be updated by writing either ”default $VAL” or ”$VAL”.

When writing to update a specific override, ”default” can be used as the value to indicate removal of the override. Override entries with ”default” as the value must not appear when read.

For example, a setting which is keyed by major:minor device numbers with integer values may look like the following:

```
# cat cgroup-example-interface-file
default 150
8:0 300
```

The default value can be updated by:

```
# echo 125 > cgroup-example-interface-file
```

or:

```
# echo "default 125" > cgroup-example-interface-file
```

An override can be set by:

```
# echo "8:16 170" > cgroup-example-interface-file
```

and cleared by:

```
# echo "8:0 default" > cgroup-example-interface-file
# cat cgroup-example-interface-file
default 125
8:16 170
```

- For events which are not very high frequency, an interface file ”events” should be created which lists event key value pairs. Whenever a notifiable event happens, file modified event should be generated on the file.

33.4.3 Core Interface Files

All cgroup core files are prefixed with ”cgroup.”

cgroup.type

A read-write single value file which exists on non-root cgroups.

When read, it indicates the current type of the cgroup, which can be one of the following values.

- ”domain” : A normal valid domain cgroup.
- ”domain threaded” : A threaded domain cgroup which is serving as the root of a threaded subtree.
- ”domain invalid” : A cgroup which is in an invalid state. It can't be populated or have controllers enabled. It may be allowed to become a threaded cgroup.
- ”threaded” : A threaded cgroup which is a member of a threaded subtree.

A cgroup can be turned into a threaded cgroup by writing "threaded" to this file.

cgroup.procs
A read-write new-line separated values file which exists on all cgroups.

When read, it lists the PIDs of all processes which belong to the cgroup one-per-line. The PIDs are not ordered and the same PID may show up more than once if the process got moved to another cgroup and then back or the PID got recycled while reading.

A PID can be written to migrate the process associated with the PID to the cgroup. The writer should match all of the following conditions.

- It must have write access to the "cgroup.procs" file.
- It must have write access to the "cgroup.procs" file of the common ancestor of the source and destination cgroups.

When delegating a sub-hierarchy, write access to this file should be granted along with the containing directory.

In a threaded cgroup, reading this file fails with EOPNOTSUPP as all the processes belong to the thread root. Writing is supported and moves every thread of the process to the cgroup.

cgroup.threads
A read-write new-line separated values file which exists on all cgroups.

When read, it lists the TIDs of all threads which belong to the cgroup one-per-line. The TIDs are not ordered and the same TID may show up more than once if the thread got moved to another cgroup and then back or the TID got recycled while reading.

A TID can be written to migrate the thread associated with the TID to the cgroup. The writer should match all of the following conditions.

- It must have write access to the "cgroup.threads" file.
- The cgroup that the thread is currently in must be in the same resource domain as the destination cgroup.
- It must have write access to the "cgroup.procs" file of the common ancestor of the source and destination cgroups.

When delegating a sub-hierarchy, write access to this file should be granted along with the containing directory.

cgroup.controllers
A read-only space separated values file which exists on all cgroups.

It shows space separated list of all controllers available to the cgroup. The controllers are not ordered.

cgroup.subtree_control
A read-write space separated values file which exists on all cgroups. Starts out empty.

When read, it shows space separated list of the controllers which are enabled to control resource distribution from the cgroup to its children.

Space separated list of controllers prefixed with '+' or '-' can be written to enable or disable controllers. A controller name prefixed with '+' enables the controller and '-' disables. If a controller appears more than once on the list, the last one is effective. When multiple enable and disable operations are specified, either all succeed or all fail.

cgroup.events
A read-only flat-keyed file which exists on non-root cgroups. The following entries are defined. Unless specified otherwise, a value change in this file generates a file modified event.

populated
1 if the cgroup or its descendants contains any live processes; otherwise, 0.

frozen
1 if the cgroup is frozen; otherwise, 0.

cgroup.max.descendants
A read-write single value files. The default is ”max”.

Maximum allowed number of descent cgroups. If the actual number of descendants is equal or larger, an attempt to create a new cgroup in the hierarchy will fail.

cgroup.max.depth
A read-write single value files. The default is ”max”.

Maximum allowed descent depth below the current cgroup. If the actual descent depth is equal or larger, an attempt to create a new child cgroup will fail.

cgroup.stat
A read-only flat-keyed file with the following entries:

nr_descendants
Total number of visible descendant cgroups.

nr_dying_descendants
Total number of dying descendant cgroups. A cgroup becomes dying after being deleted by a user. The cgroup will remain in dying state for some time undefined time (which can depend on system load) before being completely destroyed.

A process can't enter a dying cgroup under any circumstances, a dying cgroup can't revive.

A dying cgroup can consume system resources not exceeding limits, which were active at the moment of cgroup deletion.

cgroup.freeze
A read-write single value file which exists on non-root cgroups. Allowed values are ”0” and ”1”. The default is ”0”.

Writing ”1” to the file causes freezing of the cgroup and all descendant cgroups. This means that all belonging processes will be stopped and will not run until the cgroup will be explicitly unfrozen. Freezing of the cgroup may take some time; when this action is completed, the ”frozen” value in the cgroup.events control file will be updated to ”1” and the corresponding notification will be issued.

A cgroup can be frozen either by its own settings, or by settings of any ancestor cgroups. If any of ancestor cgroups is frozen, the cgroup will remain frozen.

Processes in the frozen cgroup can be killed by a fatal signal. They also can enter and leave a frozen cgroup: either by an explicit move by a user, or if freezing of the cgroup races with fork(). If a process is moved to a frozen cgroup, it stops. If a process is moved out of a frozen cgroup, it becomes running.

Frozen status of a cgroup doesn't affect any cgroup tree operations: it's possible to delete a frozen (and empty) cgroup, as well as create new sub-cgroups.

cgroup.kill
A write-only single value file which exists in non-root cgroups. The only allowed value is "1".

Writing "1" to the file causes the cgroup and all descendant cgroups to be killed. This means that all processes located in the affected cgroup tree will be killed via SIGKILL.

Killing a cgroup tree will deal with concurrent forks appropriately and is protected against migrations.

In a threaded cgroup, writing this file fails with EOPNOTSUPP as killing cgroups is a process directed operation, i.e. it affects the whole thread-group.

cgroup.pressure
A read-write single value file that allowed values are "0" and "1". The default is "1".

Writing "0" to the file will disable the cgroup PSI accounting. Writing "1" to the file will re-enable the cgroup PSI accounting.

This control attribute is not hierarchical, so disable or enable PSI accounting in a cgroup does not affect PSI accounting in descendants and doesn't need pass enablement via ancestors from root.

The reason this control attribute exists is that PSI accounts stalls for each cgroup separately and aggregates it at each level of the hierarchy. This may cause non-negligible overhead for some workloads when under deep level of the hierarchy, in which case this control attribute can be used to disable PSI accounting in the non-leaf cgroups.

irq.pressure
A read-write nested-keyed file.

Shows pressure stall information for IRQ/SOFTIRQ. See Documentation/accounting/psi.rst for details.

33.5 Controllers

33.5.1 CPU

The ”cpu” controllers regulates distribution of CPU cycles. This controller implements weight and absolute bandwidth limit models for normal scheduling policy and absolute bandwidth allocation model for realtime scheduling policy.

In all the above models, cycles distribution is defined only on a temporal base and it does not account for the frequency at which tasks are executed. The (optional) utilization clamping support allows to hint the schedutil cpufreq governor about the minimum desired frequency which should always be provided by a CPU, as well as the maximum desired frequency, which should not be exceeded by a CPU.

WARNING: cgroup2 doesn't yet support control of realtime processes and the cpu controller can only be enabled when all RT processes are in the root cgroup. Be aware that system management software may already have placed RT processes into nonroot cgroups during the system boot process, and these processes may need to be moved to the root cgroup before the cpu controller can be enabled.

CPU Interface Files

All time durations are in microseconds.

cpu.stat
A read-only flat-keyed file. This file exists whether the controller is enabled or not.

It always reports the following three stats:

- usage_usec
- user_usec
- system_usec

and the following five when the controller is enabled:

- nr_periods
- nr_throttled
- throttled_usec
- nr_bursts
- burst_usec

cpu.weight
A read-write single value file which exists on non-root cgroups. The default is ”100”.

For non idle groups (cpu.idle = 0), the weight is in the range [1, 10000].

If the cgroup has been configured to be SCHED_IDLE (cpu.idle = 1), then the weight will show as a 0.

cpu.weight.nice

A read-write single value file which exists on non-root cgroups. The default is "0".

The nice value is in the range [-20, 19].

This interface file is an alternative interface for "cpu.weight" and allows reading and setting weight using the same values used by nice(2). Because the range is smaller and granularity is coarser for the nice values, the read value is the closest approximation of the current weight.

cpu.max

A read-write two value file which exists on non-root cgroups. The default is "max 100000".

The maximum bandwidth limit. It's in the following format:

```
$MAX $PERIOD
```

which indicates that the group may consume up to $MAX in each $PERIOD duration. "max" for $MAX indicates no limit. If only one number is written, $MAX is updated.

cpu.max.burst

A read-write single value file which exists on non-root cgroups. The default is "0".

The burst in the range [0, $MAX].

cpu.pressure

A read-write nested-keyed file.

Shows pressure stall information for CPU. See Documentation/accounting/psi.rst for details.

cpu.uclamp.min

A read-write single value file which exists on non-root cgroups. The default is "0", i.e. no utilization boosting.

The requested minimum utilization (protection) as a percentage rational number, e.g. 12.34 for 12.34%.

This interface allows reading and setting minimum utilization clamp values similar to the sched_setattr(2). This minimum utilization value is used to clamp the task specific minimum utilization clamp.

The requested minimum utilization (protection) is always capped by the current value for the maximum utilization (limit), i.e. *cpu.uclamp.max*.

cpu.uclamp.max

A read-write single value file which exists on non-root cgroups. The default is "max". i.e. no utilization capping

The requested maximum utilization (limit) as a percentage rational number, e.g. 98.76 for 98.76%.

This interface allows reading and setting maximum utilization clamp values similar to the sched_setattr(2). This maximum utilization value is used to clamp the task specific maximum utilization clamp.

cpu.idle
A read-write single value file which exists on non-root cgroups. The default is 0.

This is the cgroup analog of the per-task SCHED_IDLE sched policy. Setting this value to a 1 will make the scheduling policy of the cgroup SCHED_IDLE. The threads inside the cgroup will retain their own relative priorities, but the cgroup itself will be treated as very low priority relative to its peers.

33.5.2 Memory

The "memory" controller regulates distribution of memory. Memory is stateful and implements both limit and protection models. Due to the intertwining between memory usage and reclaim pressure and the stateful nature of memory, the distribution model is relatively complex.

While not completely water-tight, all major memory usages by a given cgroup are tracked so that the total memory consumption can be accounted and controlled to a reasonable extent. Currently, the following types of memory usages are tracked.

- Userland memory - page cache and anonymous memory.
- Kernel data structures such as dentries and inodes.
- TCP socket buffers.

The above list may expand in the future for better coverage.

Memory Interface Files

All memory amounts are in bytes. If a value which is not aligned to PAGE_SIZE is written, the value may be rounded up to the closest PAGE_SIZE multiple when read back.

memory.current
A read-only single value file which exists on non-root cgroups.

The total amount of memory currently being used by the cgroup and its descendants.

memory.min
A read-write single value file which exists on non-root cgroups. The default is "0".

Hard memory protection. If the memory usage of a cgroup is within its effective min boundary, the cgroup's memory won't be reclaimed under any conditions. If there is no unprotected reclaimable memory available, OOM killer is invoked. Above the effective min boundary (or effective low boundary if it is higher), pages are reclaimed proportionally to the overage, reducing reclaim pressure for smaller overages.

Effective min boundary is limited by memory.min values of all ancestor cgroups. If there is memory.min overcommitment (child cgroup or cgroups are requiring more protected memory than parent will allow), then each child cgroup will get the part of parent's protection proportional to its actual memory usage below memory.min.

Putting more memory than generally available under this protection is discouraged and may lead to constant OOMs.

If a memory cgroup is not populated with processes, its memory.min is ignored.

memory.low

A read-write single value file which exists on non-root cgroups. The default is "0".

Best-effort memory protection. If the memory usage of a cgroup is within its effective low boundary, the cgroup's memory won't be reclaimed unless there is no reclaimable memory available in unprotected cgroups. Above the effective low boundary (or effective min boundary if it is higher), pages are reclaimed proportionally to the overage, reducing reclaim pressure for smaller overages.

Effective low boundary is limited by memory.low values of all ancestor cgroups. If there is memory.low overcommitment (child cgroup or cgroups are requiring more protected memory than parent will allow), then each child cgroup will get the part of parent's protection proportional to its actual memory usage below memory.low.

Putting more memory than generally available under this protection is discouraged.

memory.high

A read-write single value file which exists on non-root cgroups. The default is "max".

Memory usage throttle limit. If a cgroup's usage goes over the high boundary, the processes of the cgroup are throttled and put under heavy reclaim pressure.

Going over the high limit never invokes the OOM killer and under extreme conditions the limit may be breached. The high limit should be used in scenarios where an external process monitors the limited cgroup to alleviate heavy reclaim pressure.

memory.max

A read-write single value file which exists on non-root cgroups. The default is "max".

Memory usage hard limit. This is the main mechanism to limit memory usage of a cgroup. If a cgroup's memory usage reaches this limit and can't be reduced, the OOM killer is invoked in the cgroup. Under certain circumstances, the usage may go over the limit temporarily.

In default configuration regular 0-order allocations always succeed unless OOM killer chooses current task as a victim.

Some kinds of allocations don't invoke the OOM killer. Caller could retry them differently, return into userspace as -ENOMEM or silently ignore in cases like disk readahead.

memory.reclaim

A write-only nested-keyed file which exists for all cgroups.

This is a simple interface to trigger memory reclaim in the target cgroup.

This file accepts a single key, the number of bytes to reclaim. No nested keys are currently supported.

Example:

```
echo "1G" > memory.reclaim
```

The interface can be later extended with nested keys to configure the reclaim behavior. For example, specify the type of memory to reclaim from (anon, file, ..).

Please note that the kernel can over or under reclaim from the target cgroup. If less bytes are reclaimed than the specified amount, -EAGAIN is returned.

Please note that the proactive reclaim (triggered by this interface) is not meant to indicate memory pressure on the memory cgroup. Therefore socket memory balancing triggered by the memory reclaim normally is not exercised in this case. This means that the networking layer will not adapt based on reclaim induced by memory.reclaim.

memory.peak
A read-only single value file which exists on non-root cgroups.

The max memory usage recorded for the cgroup and its descendants since the creation of the cgroup.

memory.oom.group
A read-write single value file which exists on non-root cgroups. The default value is "0".

Determines whether the cgroup should be treated as an indivisible workload by the OOM killer. If set, all tasks belonging to the cgroup or to its descendants (if the memory cgroup is not a leaf cgroup) are killed together or not at all. This can be used to avoid partial kills to guarantee workload integrity.

Tasks with the OOM protection (oom_score_adj set to -1000) are treated as an exception and are never killed.

If the OOM killer is invoked in a cgroup, it's not going to kill any tasks outside of this cgroup, regardless memory.oom.group values of ancestor cgroups.

memory.events
A read-only flat-keyed file which exists on non-root cgroups. The following entries are defined. Unless specified otherwise, a value change in this file generates a file modified event.

Note that all fields in this file are hierarchical and the file modified event can be generated due to an event down the hierarchy. For the local events at the cgroup level see memory.events.local.

low
The number of times the cgroup is reclaimed due to high memory pressure even though its usage is under the low boundary. This usually indicates that the low boundary is over-committed.

high
The number of times processes of the cgroup are throttled and routed to perform direct memory reclaim because the high memory boundary was exceeded. For a cgroup whose memory usage is capped by the high limit rather than global memory pressure, this event's occurrences are expected.

max
The number of times the cgroup's memory usage was about to go over the max boundary. If direct reclaim fails to bring it down, the cgroup goes to OOM state.

oom
The number of time the cgroup's memory usage was reached the limit and allocation was about to fail.

This event is not raised if the OOM killer is not considered as an option, e.g. for failed high-order allocations or if caller asked to not retry attempts.

oom_kill
The number of processes belonging to this cgroup killed by any kind of OOM killer.

oom_group_kill
The number of times a group OOM has occurred.

memory.events.local
Similar to memory.events but the fields in the file are local to the cgroup i.e. not hierarchical. The file modified event generated on this file reflects only the local events.

memory.stat
A read-only flat-keyed file which exists on non-root cgroups.

This breaks down the cgroup's memory footprint into different types of memory, type-specific details, and other information on the state and past events of the memory management system.

All memory amounts are in bytes.

The entries are ordered to be human readable, and new entries can show up in the middle. Don't rely on items remaining in a fixed position; use the keys to look up specific values!

If the entry has no per-node counter (or not show in the memory.numa_stat). We use 'npn' (non-per-node) as the tag to indicate that it will not show in the memory.numa_stat.

anon
Amount of memory used in anonymous mappings such as brk(), sbrk), and mmap(MAP_ANONYMOUS)

file
Amount of memory used to cache filesystem data, including tmpfs and shared memory.

kernel (npn)
Amount of total kernel memory, including (kernel_stack, pagetables, percpu, vmalloc, slab) in addition to other kernel memory use cases.

kernel_stack
Amount of memory allocated to kernel stacks.

pagetables
Amount of memory allocated for page tables.

sec_pagetables
Amount of memory allocated for secondary page tables, this currently includes KVM mmu allocations on x86 and arm64.

percpu (npn)
Amount of memory used for storing per-cpu kernel data structures.

sock (npn)
Amount of memory used in network transmission buffers

vmalloc (npn)
Amount of memory used for vmap backed memory.

shmem
Amount of cached filesystem data that is swap-backed, such as tmpfs, shm segments, shared anonymous mmap()s

zswap
Amount of memory consumed by the zswap compression backend.

zswapped
Amount of application memory swapped out to zswap.

file_mapped
Amount of cached filesystem data mapped with mmap()

file_dirty
Amount of cached filesystem data that was modified but not yet written back to disk

file_writeback
Amount of cached filesystem data that was modified and is currently being written back to disk

swapcached
Amount of swap cached in memory. The swapcache is accounted against both memory and swap usage.

anon_thp
Amount of memory used in anonymous mappings backed by transparent hugepages

file_thp
Amount of cached filesystem data backed by transparent hugepages

shmem_thp
Amount of shm, tmpfs, shared anonymous mmap()s backed by transparent hugepages

inactive_anon, active_anon, inactive_file, active_file, unevictable
Amount of memory, swap-backed and filesystem-backed, on the internal memory management lists used by the page reclaim algorithm.

As these represent internal list state (eg. shmem pages are on anon memory management lists), inactive_foo + active_foo may not be equal to the value for the foo counter, since the foo counter is type-based, not list-based.

slab_reclaimable
Part of ”slab” that might be reclaimed, such as dentries and inodes.

slab_unreclaimable
Part of ”slab” that cannot be reclaimed on memory pressure.

slab (npn)
Amount of memory used for storing in-kernel data structures.

workingset_refault_anon
Number of refaults of previously evicted anonymous pages.

workingset_refault_file
Number of refaults of previously evicted file pages.

workingset_activate_anon
Number of refaulted anonymous pages that were immediately activated.

workingset_activate_file
Number of refaulted file pages that were immediately activated.

workingset_restore_anon
Number of restored anonymous pages which have been detected as an active workingset before they got reclaimed.

workingset_restore_file
Number of restored file pages which have been detected as an active workingset before they got reclaimed.

workingset_nodereclaim
Number of times a shadow node has been reclaimed

pgscan (npn)
Amount of scanned pages (in an inactive LRU list)

pgsteal (npn)
Amount of reclaimed pages

pgscan_kswapd (npn)
Amount of scanned pages by kswapd (in an inactive LRU list)

pgscan_direct (npn)
Amount of scanned pages directly (in an inactive LRU list)

pgscan_khugepaged (npn)
Amount of scanned pages by khugepaged (in an inactive LRU list)

pgsteal_kswapd (npn)
Amount of reclaimed pages by kswapd

pgsteal_direct (npn)
Amount of reclaimed pages directly

pgsteal_khugepaged (npn)
Amount of reclaimed pages by khugepaged

pgfault (npn)
Total number of page faults incurred

pgmajfault (npn)
Number of major page faults incurred

pgrefill (npn)
Amount of scanned pages (in an active LRU list)

pgactivate (npn)
Amount of pages moved to the active LRU list

pgdeactivate (npn)
Amount of pages moved to the inactive LRU list

pglazyfree (npn)
Amount of pages postponed to be freed under memory pressure

pglazyfreed (npn)
Amount of reclaimed lazyfree pages

thp_fault_alloc (npn)
Number of transparent hugepages which were allocated to satisfy a page fault. This counter is not present when CONFIG_TRANSPARENT_HUGEPAGE is not set.

thp_collapse_alloc (npn)
Number of transparent hugepages which were allocated to allow collapsing an existing range of pages. This counter is not present when CONFIG_TRANSPARENT_HUGEPAGE is not set.

thp_swpout (npn)
Number of transparent hugepages which are swapout in one piece without splitting.

thp_swpout_fallback (npn)
Number of transparent hugepages which were split before swapout. Usually because failed to allocate some continuous swap space for the huge page.

memory.numa_stat
A read-only nested-keyed file which exists on non-root cgroups.

This breaks down the cgroup's memory footprint into different types of memory, type-specific details, and other information per node on the state of the memory management system.

This is useful for providing visibility into the NUMA locality information within an memcg since the pages are allowed to be allocated from any physical node. One of the use case is evaluating application performance by combining this information with the application's CPU allocation.

All memory amounts are in bytes.

The output format of memory.numa_stat is:

```
type N0=<bytes in node 0> N1=<bytes in node 1> ...
```

The entries are ordered to be human readable, and new entries can show up in the middle. Don't rely on items remaining in a fixed position; use the keys to look up specific values!

The entries can refer to the memory.stat.

memory.swap.current
A read-only single value file which exists on non-root cgroups.

The total amount of swap currently being used by the cgroup and its descendants.

memory.swap.high
A read-write single value file which exists on non-root cgroups. The default is "max".

Swap usage throttle limit. If a cgroup's swap usage exceeds this limit, all its further allocations will be throttled to allow userspace to implement custom out-of-memory procedures.

This limit marks a point of no return for the cgroup. It is NOT designed to manage the amount of swapping a workload does during regular operation. Compare to memory.swap.max, which prohibits swapping past a set amount, but lets the cgroup continue unimpeded as long as other memory can be reclaimed.

Healthy workloads are not expected to reach this limit.

memory.swap.peak
A read-only single value file which exists on non-root cgroups.

The max swap usage recorded for the cgroup and its descendants since the creation of the cgroup.

memory.swap.max
A read-write single value file which exists on non-root cgroups. The default is "max".

Swap usage hard limit. If a cgroup's swap usage reaches this limit, anonymous memory of the cgroup will not be swapped out.

memory.swap.events
A read-only flat-keyed file which exists on non-root cgroups. The following entries are defined. Unless specified otherwise, a value change in this file generates a file modified event.

high
The number of times the cgroup's swap usage was over the high threshold.

max
The number of times the cgroup's swap usage was about to go over the max boundary and swap allocation failed.

fail
The number of times swap allocation failed either because of running out of swap system-wide or max limit.

When reduced under the current usage, the existing swap entries are reclaimed gradually and the swap usage may stay higher than the limit for an extended period of time. This reduces the impact on the workload and memory management.

memory.zswap.current
A read-only single value file which exists on non-root cgroups.

The total amount of memory consumed by the zswap compression backend.

memory.zswap.max
A read-write single value file which exists on non-root cgroups. The default is "max".

Zswap usage hard limit. If a cgroup's zswap pool reaches this limit, it will refuse to take any more stores before existing entries fault back in or are written out to disk.

memory.zswap.writeback
A read-write single value file. The default value is "1". The initial value of the root cgroup is 1, and when a new cgroup is created, it inherits the current value of its parent.

When this is set to 0, all swapping attempts to swapping devices are disabled. This included both zswap writebacks, and swapping due to zswap store failures. If the zswap store failures are recurring (for e.g if the pages are incompressible), users can observe reclaim inefficiency after disabling writeback (because the same pages might be rejected again and again).

Note that this is subtly different from setting memory.swap.max to 0, as it still allows for pages to be written to the zswap pool.

memory.pressure
A read-only nested-keyed file.

Shows pressure stall information for memory. See Documentation/accounting/psi.rst for details.

Usage Guidelines

"memory.high" is the main mechanism to control memory usage. Over-committing on high limit (sum of high limits > available memory) and letting global memory pressure to distribute memory according to usage is a viable strategy.

Because breach of the high limit doesn't trigger the OOM killer but throttles the offending cgroup, a management agent has ample opportunities to monitor and take appropriate actions such as granting more memory or terminating the workload.

Determining whether a cgroup has enough memory is not trivial as memory usage doesn't indicate whether the workload can benefit from more memory. For example, a workload which writes data received from network to a file can use all available memory but can also operate as performant with a small amount of memory. A measure of memory pressure - how much the workload is being impacted due to lack of memory - is necessary to determine whether a workload needs more memory; unfortunately, memory pressure monitoring mechanism isn't implemented yet.

Memory Ownership

A memory area is charged to the cgroup which instantiated it and stays charged to the cgroup until the area is released. Migrating a process to a different cgroup doesn't move the memory usages that it instantiated while in the previous cgroup to the new cgroup.

A memory area may be used by processes belonging to different cgroups. To which cgroup the area will be charged is in-deterministic; however, over time, the memory area is likely to end up in a cgroup which has enough memory allowance to avoid high reclaim pressure.

If a cgroup sweeps a considerable amount of memory which is expected to be accessed repeatedly by other cgroups, it may make sense to use POSIX_FADV_DONTNEED to relinquish the ownership of memory areas belonging to the affected files to ensure correct memory ownership.

33.5.3 IO

The "io" controller regulates the distribution of IO resources. This controller implements both weight based and absolute bandwidth or IOPS limit distribution; however, weight based distribution is available only if cfq-iosched is in use and neither scheme is available for blk-mq devices.

IO Interface Files

io.stat

A read-only nested-keyed file.

Lines are keyed by $MAJ:$MIN device numbers and not ordered. The following nested keys are defined.

rbytes	Bytes read
wbytes	Bytes written
rios	Number of read IOs
wios	Number of write IOs
dbytes	Bytes discarded
dios	Number of discard IOs

An example read output follows:

```
8:16 rbytes=1459200 wbytes=314773504 rios=192 wios=353 dbytes=0
↪dios=0
8:0 rbytes=90430464 wbytes=299008000 rios=8950 wios=1252
↪dbytes=50331648 dios=3021
```

io.cost.qos

A read-write nested-keyed file which exists only on the root cgroup.

This file configures the Quality of Service of the IO cost model based controller (CONFIG_BLK_CGROUP_IOCOST) which currently implements "io.weight" proportional control. Lines are keyed by $MAJ:$MIN device numbers and not ordered. The line for a given device is populated on the first write for the device on "io.cost.qos" or "io.cost.model". The following nested keys are defined.

enable	Weight-based control enable
ctrl	”auto” or ”user”
rpct	Read latency percentile [0, 100]
rlat	Read latency threshold
wpct	Write latency percentile [0, 100]
wlat	Write latency threshold
min	Minimum scaling percentage [1, 10000]
max	Maximum scaling percentage [1, 10000]

The controller is disabled by default and can be enabled by setting ”enable” to 1. ”rpct” and ”wpct” parameters default to zero and the controller uses internal device saturation state to adjust the overall IO rate between ”min” and ”max”.

When a better control quality is needed, latency QoS parameters can be configured. For example:

```
8:16 enable=1 ctrl=auto rpct=95.00 rlat=75000 wpct=95.00␣
↪wlat=150000 min=50.00 max=150.0
```

shows that on sdb, the controller is enabled, will consider the device saturated if the 95th percentile of read completion latencies is above 75ms or write 150ms, and adjust the overall IO issue rate between 50% and 150% accordingly.

The lower the saturation point, the better the latency QoS at the cost of aggregate bandwidth. The narrower the allowed adjustment range between ”min” and ”max”, the more conformant to the cost model the IO behavior. Note that the IO issue base rate may be far off from 100% and setting ”min” and ”max” blindly can lead to a significant loss of device capacity or control quality. ”min” and ”max” are useful for regulating devices which show wide temporary behavior changes - e.g. a ssd which accepts writes at the line speed for a while and then completely stalls for multiple seconds.

When ”ctrl” is ”auto”, the parameters are controlled by the kernel and may change automatically. Setting ”ctrl” to ”user” or setting any of the percentile and latency parameters puts it into ”user” mode and disables the automatic changes. The automatic mode can be restored by setting ”ctrl” to ”auto”.

io.cost.model

A read-write nested-keyed file which exists only on the root cgroup.

This file configures the cost model of the IO cost model based controller (CONFIG_BLK_CGROUP_IOCOST) which currently implements ”io.weight” proportional control. Lines are keyed by $MAJ:$MIN device numbers and not ordered. The line for a given device is populated on the first write for the device on ”io.cost.qos” or ”io.cost.model”. The following nested keys are defined.

ctrl	”auto” or ”user”
model	The cost model in use - ”linear”

When ”ctrl” is ”auto”, the kernel may change all parameters dynamically. When ”ctrl” is set to ”user” or any other parameters are written to, ”ctrl” become ”user” and the automatic changes are disabled.

When ”model” is ”linear”, the following model parameters are defined.

[r|w]bps	The maximum sequential IO throughput
[r|w]seqiops	The maximum 4k sequential IOs per second
[r|w]randiops	The maximum 4k random IOs per second

From the above, the builtin linear model determines the base costs of a sequential and random IO and the cost coefficient for the IO size. While simple, this model can cover most common device classes acceptably.

The IO cost model isn't expected to be accurate in absolute sense and is scaled to the device behavior dynamically.

If needed, tools/cgroup/iocost_coef_gen.py can be used to generate device-specific coefficients.

io.weight

A read-write flat-keyed file which exists on non-root cgroups. The default is ”default 100”.

The first line is the default weight applied to devices without specific override. The rest are overrides keyed by $MAJ:$MIN device numbers and not ordered. The weights are in the range [1, 10000] and specifies the relative amount IO time the cgroup can use in relation to its siblings.

The default weight can be updated by writing either ”default $WEIGHT” or simply ”$WEIGHT”. Overrides can be set by writing ”$MAJ:$MIN $WEIGHT” and unset by writing ”$MAJ:$MIN default”.

An example read output follows:

```
default 100
8:16 200
8:0 50
```

io.max

A read-write nested-keyed file which exists on non-root cgroups.

BPS and IOPS based IO limit. Lines are keyed by $MAJ:$MIN device numbers and not ordered. The following nested keys are defined.

rbps	Max read bytes per second
wbps	Max write bytes per second
riops	Max read IO operations per second
wiops	Max write IO operations per second

When writing, any number of nested key-value pairs can be specified in any order. ”max” can be specified as the value to remove a specific limit. If the same key is specified multiple times, the outcome is undefined.

BPS and IOPS are measured in each IO direction and IOs are delayed if limit is reached. Temporary bursts are allowed.

Setting read limit at 2M BPS and write at 120 IOPS for 8:16:

```
echo "8:16 rbps=2097152 wiops=120" > io.max
```

Reading returns the following:

```
8:16 rbps=2097152 wbps=max riops=max wiops=120
```

Write IOPS limit can be removed by writing the following:

```
echo "8:16 wiops=max" > io.max
```

Reading now returns the following:

```
8:16 rbps=2097152 wbps=max riops=max wiops=max
```

io.pressure
A read-only nested-keyed file.

Shows pressure stall information for IO. See Documentation/accounting/psi.rst for details.

Writeback

Page cache is dirtied through buffered writes and shared mmaps and written asynchronously to the backing filesystem by the writeback mechanism. Writeback sits between the memory and IO domains and regulates the proportion of dirty memory by balancing dirtying and write IOs.

The io controller, in conjunction with the memory controller, implements control of page cache writeback IOs. The memory controller defines the memory domain that dirty memory ratio is calculated and maintained for and the io controller defines the io domain which writes out dirty pages for the memory domain. Both system-wide and per-cgroup dirty memory states are examined and the more restrictive of the two is enforced.

cgroup writeback requires explicit support from the underlying filesystem. Currently, cgroup writeback is implemented on ext2, ext4, btrfs, f2fs, and xfs. On other filesystems, all writeback IOs are attributed to the root cgroup.

There are inherent differences in memory and writeback management which affects how cgroup ownership is tracked. Memory is tracked per page while writeback per inode. For the purpose of writeback, an inode is assigned to a cgroup and all IO requests to write dirty pages from the inode are attributed to that cgroup.

As cgroup ownership for memory is tracked per page, there can be pages which are associated with different cgroups than the one the inode is associated with. These are called foreign pages. The writeback constantly keeps track of foreign pages and, if a particular foreign cgroup becomes the majority over a certain period of time, switches the ownership of the inode to that cgroup.

While this model is enough for most use cases where a given inode is mostly dirtied by a single cgroup even when the main writing cgroup changes over time, use cases where multiple cgroups write to a single inode simultaneously are not supported well. In such circumstances, a significant portion of IOs are likely to be attributed incorrectly. As memory controller assigns page ownership on the first use and doesn't update it until the page is released, even if writeback strictly follows page ownership, multiple cgroups dirtying overlapping areas wouldn't work as expected. It's recommended to avoid such usage patterns.

The sysctl knobs which affect writeback behavior are applied to cgroup writeback as follows.

vm.dirty_background_ratio, vm.dirty_ratio
These ratios apply the same to cgroup writeback with the amount of available memory capped by limits imposed by the memory controller and system-wide clean memory.

vm.dirty_background_bytes, vm.dirty_bytes
For cgroup writeback, this is calculated into ratio against total available memory and applied the same way as vm.dirty[_background]_ratio.

IO Latency

This is a cgroup v2 controller for IO workload protection. You provide a group with a latency target, and if the average latency exceeds that target the controller will throttle any peers that have a lower latency target than the protected workload.

The limits are only applied at the peer level in the hierarchy. This means that in the diagram below, only groups A, B, and C will influence each other, and groups D and F will influence each other. Group G will influence nobody:

```
          [root]
  /          |            \
  A          B            C
 /  \        |
D    F       G
```

So the ideal way to configure this is to set io.latency in groups A, B, and C. Generally you do not want to set a value lower than the latency your device supports. Experiment to find the value that works best for your workload. Start at higher than the expected latency for your device and watch the avg_lat value in io.stat for your workload group to get an idea of the latency you see during normal operation. Use the avg_lat value as a basis for your real setting, setting at 10-15% higher than the value in io.stat.

How IO Latency Throttling Works

io.latency is work conserving; so as long as everybody is meeting their latency target the controller doesn't do anything. Once a group starts missing its target it begins throttling any peer group that has a higher target than itself. This throttling takes 2 forms:

- Queue depth throttling. This is the number of outstanding IO's a group is allowed to have. We will clamp down relatively quickly, starting at no limit and going all the way down to 1 IO at a time.
- Artificial delay induction. There are certain types of IO that cannot be throttled without possibly adversely affecting higher priority groups. This includes swapping and metadata IO. These types of IO are allowed to occur normally, however they are ”charged” to the originating group. If the originating group is being throttled you will see the use_delay and delay fields in io.stat increase. The delay value is how many microseconds that are being added to any process that runs in this group. Because this number can grow quite large if there is a lot of swapping or metadata IO occurring we limit the individual delay events to 1 second at a time.

Once the victimized group starts meeting its latency target again it will start unthrottling any peer groups that were throttled previously. If the victimized group simply stops doing IO the global counter will unthrottle appropriately.

IO Latency Interface Files

io.latency
This takes a similar format as the other controllers.

"MAJOR:MINOR target=<target time in microseconds>"

io.stat
If the controller is enabled you will see extra stats in io.stat in addition to the normal ones.

depth
This is the current queue depth for the group.

avg_lat
This is an exponential moving average with a decay rate of 1/exp bound by the sampling interval. The decay rate interval can be calculated by multiplying the win value in io.stat by the corresponding number of samples based on the win value.

win
The sampling window size in milliseconds. This is the minimum duration of time between evaluation events. Windows only elapse with IO activity. Idle periods extend the most recent window.

IO Priority

A single attribute controls the behavior of the I/O priority cgroup policy, namely the io.prio.class attribute. The following values are accepted for that attribute:

no-change
Do not modify the I/O priority class.

promote-to-rt
For requests that have a non-RT I/O priority class, change it into RT. Also change the priority level of these requests to 4. Do not modify the I/O priority of requests that have priority class RT.

restrict-to-be
For requests that do not have an I/O priority class or that have I/O priority class RT, change it into BE. Also change the priority level of these requests to 0. Do not modify the I/O priority class of requests that have priority class IDLE.

idle
Change the I/O priority class of all requests into IDLE, the lowest I/O priority class.

none-to-rt
Deprecated. Just an alias for promote-to-rt.

The following numerical values are associated with the I/O priority policies:

no-change	0
promote-to-rt	1
restrict-to-be	2
idle	3

The numerical value that corresponds to each I/O priority class is as follows:

IOPRIO_CLASS_NONE	0
IOPRIO_CLASS_RT (real-time)	1
IOPRIO_CLASS_BE (best effort)	2
IOPRIO_CLASS_IDLE	3

The algorithm to set the I/O priority class for a request is as follows:

- If I/O priority class policy is promote-to-rt, change the request I/O priority class to IOPRIO_CLASS_RT and change the request I/O priority level to 4.
- If I/O priority class policy is not promote-to-rt, translate the I/O priority class policy into a number, then change the request I/O priority class into the maximum of the I/O priority class policy number and the numerical I/O priority class.

33.5.4 PID

The process number controller is used to allow a cgroup to stop any new tasks from being fork()'d or clone()'d after a specified limit is reached.

The number of tasks in a cgroup can be exhausted in ways which other controllers cannot prevent, thus warranting its own controller. For example, a fork bomb is likely to exhaust the number of tasks before hitting memory restrictions.

Note that PIDs used in this controller refer to TIDs, process IDs as used by the kernel.

PID Interface Files

pids.max
A read-write single value file which exists on non-root cgroups. The default is "max".

Hard limit of number of processes.

pids.current
A read-only single value file which exists on all cgroups.

The number of processes currently in the cgroup and its descendants.

Organisational operations are not blocked by cgroup policies, so it is possible to have pids.current > pids.max. This can be done by either setting the limit to be smaller than pids.current, or attaching enough processes to the cgroup such that pids.current is larger than pids.max. However, it is not possible to violate a cgroup PID policy through fork() or clone(). These will return -EAGAIN if the creation of a new process would cause a cgroup policy to be violated.

33.5.5 Cpuset

The ”cpuset” controller provides a mechanism for constraining the CPU and memory node placement of tasks to only the resources specified in the cpuset interface files in a task's current cgroup. This is especially valuable on large NUMA systems where placing jobs on properly sized subsets of the systems with careful processor and memory placement to reduce cross-node memory access and contention can improve overall system performance.

The ”cpuset” controller is hierarchical. That means the controller cannot use CPUs or memory nodes not allowed in its parent.

Cpuset Interface Files

cpuset.cpus

A read-write multiple values file which exists on non-root cpuset-enabled cgroups.

It lists the requested CPUs to be used by tasks within this cgroup. The actual list of CPUs to be granted, however, is subjected to constraints imposed by its parent and can differ from the requested CPUs.

The CPU numbers are comma-separated numbers or ranges. For example:

```
# cat cpuset.cpus
0-4,6,8-10
```

An empty value indicates that the cgroup is using the same setting as the nearest cgroup ancestor with a non-empty ”cpuset.cpus” or all the available CPUs if none is found.

The value of ”cpuset.cpus” stays constant until the next update and won't be affected by any CPU hotplug events.

cpuset.cpus.effective

A read-only multiple values file which exists on all cpuset-enabled cgroups.

It lists the onlined CPUs that are actually granted to this cgroup by its parent. These CPUs are allowed to be used by tasks within the current cgroup.

If ”cpuset.cpus” is empty, the ”cpuset.cpus.effective” file shows all the CPUs from the parent cgroup that can be available to be used by this cgroup. Otherwise, it should be a subset of ”cpuset.cpus” unless none of the CPUs listed in ”cpuset.cpus” can be granted. In this case, it will be treated just like an empty ”cpuset.cpus”.

Its value will be affected by CPU hotplug events.

cpuset.mems

A read-write multiple values file which exists on non-root cpuset-enabled cgroups.

It lists the requested memory nodes to be used by tasks within this cgroup. The actual list of memory nodes granted, however, is subjected to constraints imposed by its parent and can differ from the requested memory nodes.

The memory node numbers are comma-separated numbers or ranges. For example:

```
# cat cpuset.mems
0-1,3
```

An empty value indicates that the cgroup is using the same setting as the nearest cgroup ancestor with a non-empty "cpuset.mems" or all the available memory nodes if none is found.

The value of "cpuset.mems" stays constant until the next update and won't be affected by any memory nodes hotplug events.

Setting a non-empty value to "cpuset.mems" causes memory of tasks within the cgroup to be migrated to the designated nodes if they are currently using memory outside of the designated nodes.

There is a cost for this memory migration. The migration may not be complete and some memory pages may be left behind. So it is recommended that "cpuset.mems" should be set properly before spawning new tasks into the cpuset. Even if there is a need to change "cpuset.mems" with active tasks, it shouldn't be done frequently.

cpuset.mems.effective

A read-only multiple values file which exists on all cpuset-enabled cgroups.

It lists the onlined memory nodes that are actually granted to this cgroup by its parent. These memory nodes are allowed to be used by tasks within the current cgroup.

If "cpuset.mems" is empty, it shows all the memory nodes from the parent cgroup that will be available to be used by this cgroup. Otherwise, it should be a subset of "cpuset.mems" unless none of the memory nodes listed in "cpuset.mems" can be granted. In this case, it will be treated just like an empty "cpuset.mems".

Its value will be affected by memory nodes hotplug events.

cpuset.cpus.exclusive

A read-write multiple values file which exists on non-root cpuset-enabled cgroups.

It lists all the exclusive CPUs that are allowed to be used to create a new cpuset partition. Its value is not used unless the cgroup becomes a valid partition root. See the "cpuset.cpus.partition" section below for a description of what a cpuset partition is.

When the cgroup becomes a partition root, the actual exclusive CPUs that are allocated to that partition are listed in "cpuset.cpus.exclusive.effective" which may be different from "cpuset.cpus.exclusive". If "cpuset.cpus.exclusive" has previously been set, "cpuset.cpus.exclusive.effective" is always a subset of it.

Users can manually set it to a value that is different from "cpuset.cpus". The only constraint in setting it is that the list of CPUs must be exclusive with respect to its sibling.

For a parent cgroup, any one of its exclusive CPUs can only be distributed to at most one of its child cgroups. Having an exclusive CPU appearing in two or more of its child cgroups is not allowed (the exclusivity rule). A value that violates the exclusivity rule will be rejected with a write error.

The root cgroup is a partition root and all its available CPUs are in its exclusive CPU set.

cpuset.cpus.exclusive.effective
A read-only multiple values file which exists on all non-root cpuset-enabled cgroups.

This file shows the effective set of exclusive CPUs that can be used to create a partition root. The content of this file will always be a subset of "cpuset.cpus" and its parent's "cpuset.cpus.exclusive.effective" if its parent is not the root cgroup. It will also be a subset of "cpuset.cpus.exclusive" if it is set. If "cpuset.cpus.exclusive" is not set, it is treated to have an implicit value of "cpuset.cpus" in the formation of local partition.

cpuset.cpus.isolated
A read-only and root cgroup only multiple values file.

This file shows the set of all isolated CPUs used in existing isolated partitions. It will be empty if no isolated partition is created.

cpuset.cpus.partition
A read-write single value file which exists on non-root cpuset-enabled cgroups. This flag is owned by the parent cgroup and is not delegatable.

It accepts only the following input values when written to.

"member"	Non-root member of a partition
"root"	Partition root
"isolated"	Partition root without load balancing

A cpuset partition is a collection of cpuset-enabled cgroups with a partition root at the top of the hierarchy and its descendants except those that are separate partition roots themselves and their descendants. A partition has exclusive access to the set of exclusive CPUs allocated to it. Other cgroups outside of that partition cannot use any CPUs in that set.

There are two types of partitions - local and remote. A local partition is one whose parent cgroup is also a valid partition root. A remote partition is one whose parent cgroup is not a valid partition root itself. Writing to "cpuset.cpus.exclusive" is optional for the creation of a local partition as its "cpuset.cpus.exclusive" file will assume an implicit value that is the same as "cpuset.cpus" if it is not set. Writing the proper "cpuset.cpus.exclusive" values down the cgroup hierarchy before the target partition root is mandatory for the creation of a remote partition.

Currently, a remote partition cannot be created under a local partition. All the ancestors of a remote partition root except the root cgroup cannot be a partition root.

The root cgroup is always a partition root and its state cannot be changed. All other non-root cgroups start out as "member".

When set to "root", the current cgroup is the root of a new partition or scheduling domain. The set of exclusive CPUs is determined by the value of its "cpuset.cpus.exclusive.effective".

When set to ”isolated”, the CPUs in that partition will be in an isolated state without any load balancing from the scheduler and excluded from the unbound workqueues. Tasks placed in such a partition with multiple CPUs should be carefully distributed and bound to each of the individual CPUs for optimal performance.

A partition root (”root” or ”isolated”) can be in one of the two possible states - valid or invalid. An invalid partition root is in a degraded state where some state information may be retained, but behaves more like a ”member”.

All possible state transitions among ”member”, ”root” and ”isolated” are allowed.

On read, the ”cpuset.cpus.partition” file can show the following values.

”member”	Non-root member of a partition
”root”	Partition root
”isolated”	Partition root without load balancing
”root invalid (<reason>)”	Invalid partition root
”isolated invalid (<reason>)”	Invalid isolated partition root

In the case of an invalid partition root, a descriptive string on why the partition is invalid is included within parentheses.

For a local partition root to be valid, the following conditions must be met.

1) The parent cgroup is a valid partition root.
2) The ”cpuset.cpus.exclusive.effective” file cannot be empty, though it may contain offline CPUs.
3) The ”cpuset.cpus.effective” cannot be empty unless there is no task associated with this partition.

For a remote partition root to be valid, all the above conditions except the first one must be met.

External events like hotplug or changes to ”cpuset.cpus” or ”cpuset.cpus.exclusive” can cause a valid partition root to become invalid and vice versa. Note that a task cannot be moved to a cgroup with empty ”cpuset.cpus.effective”.

A valid non-root parent partition may distribute out all its CPUs to its child local partitions when there is no task associated with it.

Care must be taken to change a valid partition root to ”member” as all its child local partitions, if present, will become invalid causing disruption to tasks running in those child partitions. These inactivated partitions could be recovered if their parent is switched back to a partition root with a proper value in ”cpuset.cpus” or ”cpuset.cpus.exclusive”.

Poll and inotify events are triggered whenever the state of ”cpuset.cpus.partition” changes. That includes changes caused by write to ”cpuset.cpus.partition”, cpu hotplug or other changes that modify the validity status of the partition. This will allow user space agents to monitor unexpected changes to ”cpuset.cpus.partition” without the need to do continuous polling.

A user can pre-configure certain CPUs to an isolated state with load balancing disabled at boot time with the "isolcpus" kernel boot command line option. If those CPUs are to be put into a partition, they have to be used in an isolated partition.

33.5.6 Device controller

Device controller manages access to device files. It includes both creation of new device files (using mknod), and access to the existing device files.

Cgroup v2 device controller has no interface files and is implemented on top of cgroup BPF. To control access to device files, a user may create bpf programs of type BPF_PROG_TYPE_CGROUP_DEVICE and attach them to cgroups with BPF_CGROUP_DEVICE flag. On an attempt to access a device file, corresponding BPF programs will be executed, and depending on the return value the attempt will succeed or fail with -EPERM.

A BPF_PROG_TYPE_CGROUP_DEVICE program takes a pointer to the bpf_cgroup_dev_ctx structure, which describes the device access attempt: access type (mknod/read/write) and device (type, major and minor numbers). If the program returns 0, the attempt fails with -EPERM, otherwise it succeeds.

An example of BPF_PROG_TYPE_CGROUP_DEVICE program may be found in tools/testing/selftests/bpf/progs/dev_cgroup.c in the kernel source tree.

33.5.7 RDMA

The "rdma" controller regulates the distribution and accounting of RDMA resources.

RDMA Interface Files

rdma.max

A readwrite nested-keyed file that exists for all the cgroups except root that describes current configured resource limit for a RDMA/IB device.

Lines are keyed by device name and are not ordered. Each line contains space separated resource name and its configured limit that can be distributed.

The following nested keys are defined.

hca_handle	Maximum number of HCA Handles
hca_object	Maximum number of HCA Objects

An example for mlx4 and ocrdma device follows:

```
mlx4_0 hca_handle=2 hca_object=2000
ocrdma1 hca_handle=3 hca_object=max
```

rdma.current

A read-only file that describes current resource usage. It exists for all the cgroup except root.

An example for mlx4 and ocrdma device follows:

```
mlx4_0 hca_handle=1 hca_object=20
ocrdma1 hca_handle=1 hca_object=23
```

33.5.8 HugeTLB

The HugeTLB controller allows to limit the HugeTLB usage per control group and enforces the controller limit during page fault.

HugeTLB Interface Files

hugetlb.<hugepagesize>.current
Show current usage for "hugepagesize" hugetlb. It exists for all the cgroup except root.

hugetlb.<hugepagesize>.max
Set/show the hard limit of "hugepagesize" hugetlb usage. The default value is "max". It exists for all the cgroup except root.

hugetlb.<hugepagesize>.events
A read-only flat-keyed file which exists on non-root cgroups.

max
The number of allocation failure due to HugeTLB limit

hugetlb.<hugepagesize>.events.local
Similar to hugetlb.<hugepagesize>.events but the fields in the file are local to the cgroup i.e. not hierarchical. The file modified event generated on this file reflects only the local events.

hugetlb.<hugepagesize>.numa_stat
Similar to memory.numa_stat, it shows the numa information of the hugetlb pages of <hugepagesize> in this cgroup. Only active in use hugetlb pages are included. The per-node values are in bytes.

33.5.9 Misc

The Miscellaneous cgroup provides the resource limiting and tracking mechanism for the scalar resources which cannot be abstracted like the other cgroup resources. Controller is enabled by the CONFIG_CGROUP_MISC config option.

A resource can be added to the controller via enum misc_res_type{} in the include/linux/misc_cgroup.h file and the corresponding name via misc_res_name[] in the kernel/cgroup/misc.c file. Provider of the resource must set its capacity prior to using the resource by calling misc_cg_set_capacity().

Once a capacity is set then the resource usage can be updated using charge and uncharge APIs. All of the APIs to interact with misc controller are in include/linux/misc_cgroup.h.

Misc Interface Files

Miscellaneous controller provides 3 interface files. If two misc resources (res_a and res_b) are registered then:

misc.capacity
A read-only flat-keyed file shown only in the root cgroup. It shows miscellaneous scalar resources available on the platform along with their quantities:

```
$ cat misc.capacity
res_a 50
res_b 10
```

misc.current
A read-only flat-keyed file shown in the all cgroups. It shows the current usage of the resources in the cgroup and its children.:

```
$ cat misc.current
res_a 3
res_b 0
```

misc.max
A read-write flat-keyed file shown in the non root cgroups. Allowed maximum usage of the resources in the cgroup and its children.:

```
$ cat misc.max
res_a max
res_b 4
```

Limit can be set by:

```
# echo res_a 1 > misc.max
```

Limit can be set to max by:

```
# echo res_a max > misc.max
```

Limits can be set higher than the capacity value in the misc.capacity file.

misc.events
A read-only flat-keyed file which exists on non-root cgroups. The following entries are defined. Unless specified otherwise, a value change in this file generates a file modified event. All fields in this file are hierarchical.

max
The number of times the cgroup's resource usage was about to go over the max boundary.

Migration and Ownership

A miscellaneous scalar resource is charged to the cgroup in which it is used first, and stays charged to that cgroup until that resource is freed. Migrating a process to a different cgroup does not move the charge to the destination cgroup where the process has moved.

33.5.10 Others

perf_event

perf_event controller, if not mounted on a legacy hierarchy, is automatically enabled on the v2 hierarchy so that perf events can always be filtered by cgroup v2 path. The controller can still be moved to a legacy hierarchy after v2 hierarchy is populated.

33.5.11 Non-normative information

This section contains information that isn't considered to be a part of the stable kernel API and so is subject to change.

CPU controller root cgroup process behaviour

When distributing CPU cycles in the root cgroup each thread in this cgroup is treated as if it was hosted in a separate child cgroup of the root cgroup. This child cgroup weight is dependent on its thread nice level.

For details of this mapping see sched_prio_to_weight array in kernel/sched/core.c file (values from this array should be scaled appropriately so the neutral - nice 0 - value is 100 instead of 1024).

IO controller root cgroup process behaviour

Root cgroup processes are hosted in an implicit leaf child node. When distributing IO resources this implicit child node is taken into account as if it was a normal child cgroup of the root cgroup with a weight value of 200.

33.6 Namespace

33.6.1 Basics

cgroup namespace provides a mechanism to virtualize the view of the "/proc/$PID/cgroup" file and cgroup mounts. The CLONE_NEWCGROUP clone flag can be used with clone(2) and unshare(2) to create a new cgroup namespace. The process running inside the cgroup namespace will have its "/proc/$PID/cgroup" output restricted to cgroupns root. The cgroupns root is the cgroup of the process at the time of creation of the cgroup namespace.

Without cgroup namespace, the "/proc/$PID/cgroup" file shows the complete path of the cgroup of a process. In a container setup where a set of cgroups and namespaces are intended to

isolate processes the "/proc/$PID/cgroup" file may leak potential system level information to the isolated processes. For example:

```
# cat /proc/self/cgroup
0::/batchjobs/container_id1
```

The path '/batchjobs/container_id1' can be considered as system-data and undesirable to expose to the isolated processes. cgroup namespace can be used to restrict visibility of this path. For example, before creating a cgroup namespace, one would see:

```
# ls -l /proc/self/ns/cgroup
lrwxrwxrwx 1 root root 0 2014-07-15 10:37 /proc/self/ns/cgroup ->␣
↪cgroup:[4026531835]
# cat /proc/self/cgroup
0::/batchjobs/container_id1
```

After unsharing a new namespace, the view changes:

```
# ls -l /proc/self/ns/cgroup
lrwxrwxrwx 1 root root 0 2014-07-15 10:35 /proc/self/ns/cgroup ->␣
↪cgroup:[4026532183]
# cat /proc/self/cgroup
0::/
```

When some thread from a multi-threaded process unshares its cgroup namespace, the new cgroupns gets applied to the entire process (all the threads). This is natural for the v2 hierarchy; however, for the legacy hierarchies, this may be unexpected.

A cgroup namespace is alive as long as there are processes inside or mounts pinning it. When the last usage goes away, the cgroup namespace is destroyed. The cgroupns root and the actual cgroups remain.

33.6.2 The Root and Views

The 'cgroupns root' for a cgroup namespace is the cgroup in which the process calling unshare(2) is running. For example, if a process in /batchjobs/container_id1 cgroup calls unshare, cgroup /batchjobs/container_id1 becomes the cgroupns root. For the init_cgroup_ns, this is the real root ('/') cgroup.

The cgroupns root cgroup does not change even if the namespace creator process later moves to a different cgroup:

```
# ~/unshare -c # unshare cgroupns in some cgroup
# cat /proc/self/cgroup
0::/
# mkdir sub_cgrp_1
# echo 0 > sub_cgrp_1/cgroup.procs
# cat /proc/self/cgroup
0::/sub_cgrp_1
```

Each process gets its namespace-specific view of "/proc/$PID/cgroup"

Processes running inside the cgroup namespace will be able to see cgroup paths (in /proc/self/cgroup) only inside their root cgroup. From within an unshared cgroupns:

```
# sleep 100000 &
[1] 7353
# echo 7353 > sub_cgrp_1/cgroup.procs
# cat /proc/7353/cgroup
0::/sub_cgrp_1
```

From the initial cgroup namespace, the real cgroup path will be visible:

```
$ cat /proc/7353/cgroup
0::/batchjobs/container_id1/sub_cgrp_1
```

From a sibling cgroup namespace (that is, a namespace rooted at a different cgroup), the cgroup path relative to its own cgroup namespace root will be shown. For instance, if PID 7353's cgroup namespace root is at '/batchjobs/container_id2', then it will see:

```
# cat /proc/7353/cgroup
0::/../container_id2/sub_cgrp_1
```

Note that the relative path always starts with '/' to indicate that its relative to the cgroup namespace root of the caller.

33.6.3 Migration and setns(2)

Processes inside a cgroup namespace can move into and out of the namespace root if they have proper access to external cgroups. For example, from inside a namespace with cgroupns root at /batchjobs/container_id1, and assuming that the global hierarchy is still accessible inside cgroupns:

```
# cat /proc/7353/cgroup
0::/sub_cgrp_1
# echo 7353 > batchjobs/container_id2/cgroup.procs
# cat /proc/7353/cgroup
0::/../container_id2
```

Note that this kind of setup is not encouraged. A task inside cgroup namespace should only be exposed to its own cgroupns hierarchy.

setns(2) to another cgroup namespace is allowed when:

(a) the process has CAP_SYS_ADMIN against its current user namespace

(b) the process has CAP_SYS_ADMIN against the target cgroup namespace's userns

No implicit cgroup changes happen with attaching to another cgroup namespace. It is expected that the someone moves the attaching process under the target cgroup namespace root.

33.6.4 Interaction with Other Namespaces

Namespace specific cgroup hierarchy can be mounted by a process running inside a non-init cgroup namespace:

```
# mount -t cgroup2 none $MOUNT_POINT
```

This will mount the unified cgroup hierarchy with cgroupns root as the filesystem root. The process needs CAP_SYS_ADMIN against its user and mount namespaces.

The virtualization of /proc/self/cgroup file combined with restricting the view of cgroup hierarchy by namespace-private cgroupfs mount provides a properly isolated cgroup view inside the container.

33.7 Information on Kernel Programming

This section contains kernel programming information in the areas where interacting with cgroup is necessary. cgroup core and controllers are not covered.

33.7.1 Filesystem Support for Writeback

A filesystem can support cgroup writeback by updating address_space_operations->writepage[s]() to annotate bio's using the following two functions.

wbc_init_bio(@wbc, @bio)
Should be called for each bio carrying writeback data and associates the bio with the inode's owner cgroup and the corresponding request queue. This must be called after a queue (device) has been associated with the bio and before submission.

wbc_account_cgroup_owner(@wbc, @page, @bytes)
Should be called for each data segment being written out. While this function doesn't care exactly when it's called during the writeback session, it's the easiest and most natural to call it as data segments are added to a bio.

With writeback bio's annotated, cgroup support can be enabled per super_block by setting SB_I_CGROUPWB in ->s_iflags. This allows for selective disabling of cgroup writeback support which is helpful when certain filesystem features, e.g. journaled data mode, are incompatible.

wbc_init_bio() binds the specified bio to its cgroup. Depending on the configuration, the bio may be executed at a lower priority and if the writeback session is holding shared resources, e.g. a journal entry, may lead to priority inversion. There is no one easy solution for the problem. Filesystems can try to work around specific problem cases by skipping wbc_init_bio() and using bio_associate_blkg() directly.

33.8 Deprecated v1 Core Features

- Multiple hierarchies including named ones are not supported.
- All v1 mount options are not supported.
- The ”tasks” file is removed and ”cgroup.procs” is not sorted.
- ”cgroup.clone_children” is removed.
- /proc/cgroups is meaningless for v2. Use ”cgroup.controllers” file at the root instead.

33.9 Issues with v1 and Rationales for v2

33.9.1 Multiple Hierarchies

cgroup v1 allowed an arbitrary number of hierarchies and each hierarchy could host any number of controllers. While this seemed to provide a high level of flexibility, it wasn't useful in practice.

For example, as there is only one instance of each controller, utility type controllers such as freezer which can be useful in all hierarchies could only be used in one. The issue is exacerbated by the fact that controllers couldn't be moved to another hierarchy once hierarchies were populated. Another issue was that all controllers bound to a hierarchy were forced to have exactly the same view of the hierarchy. It wasn't possible to vary the granularity depending on the specific controller.

In practice, these issues heavily limited which controllers could be put on the same hierarchy and most configurations resorted to putting each controller on its own hierarchy. Only closely related ones, such as the cpu and cpuacct controllers, made sense to be put on the same hierarchy. This often meant that userland ended up managing multiple similar hierarchies repeating the same steps on each hierarchy whenever a hierarchy management operation was necessary.

Furthermore, support for multiple hierarchies came at a steep cost. It greatly complicated cgroup core implementation but more importantly the support for multiple hierarchies restricted how cgroup could be used in general and what controllers was able to do.

There was no limit on how many hierarchies there might be, which meant that a thread's cgroup membership couldn't be described in finite length. The key might contain any number of entries and was unlimited in length, which made it highly awkward to manipulate and led to addition of controllers which existed only to identify membership, which in turn exacerbated the original problem of proliferating number of hierarchies.

Also, as a controller couldn't have any expectation regarding the topologies of hierarchies other controllers might be on, each controller had to assume that all other controllers were attached to completely orthogonal hierarchies. This made it impossible, or at least very cumbersome, for controllers to cooperate with each other.

In most use cases, putting controllers on hierarchies which are completely orthogonal to each other isn't necessary. What usually is called for is the ability to have differing levels of granularity depending on the specific controller. In other words, hierarchy may be collapsed from leaf towards root when viewed from specific controllers. For example, a given configuration might not care about how memory is distributed beyond a certain level while still wanting to control how CPU cycles are distributed.

33.9.2 Thread Granularity

cgroup v1 allowed threads of a process to belong to different cgroups. This didn't make sense for some controllers and those controllers ended up implementing different ways to ignore such situations but much more importantly it blurred the line between API exposed to individual applications and system management interface.

Generally, in-process knowledge is available only to the process itself; thus, unlike service-level organization of processes, categorizing threads of a process requires active participation from the application which owns the target process.

cgroup v1 had an ambiguously defined delegation model which got abused in combination with thread granularity. cgroups were delegated to individual applications so that they can create and manage their own sub-hierarchies and control resource distributions along them. This effectively raised cgroup to the status of a syscall-like API exposed to lay programs.

First of all, cgroup has a fundamentally inadequate interface to be exposed this way. For a process to access its own knobs, it has to extract the path on the target hierarchy from /proc/self/cgroup, construct the path by appending the name of the knob to the path, open and then read and/or write to it. This is not only extremely clunky and unusual but also inherently racy. There is no conventional way to define transaction across the required steps and nothing can guarantee that the process would actually be operating on its own sub-hierarchy.

cgroup controllers implemented a number of knobs which would never be accepted as public APIs because they were just adding control knobs to system-management pseudo filesystem. cgroup ended up with interface knobs which were not properly abstracted or refined and directly revealed kernel internal details. These knobs got exposed to individual applications through the ill-defined delegation mechanism effectively abusing cgroup as a shortcut to implementing public APIs without going through the required scrutiny.

This was painful for both userland and kernel. Userland ended up with misbehaving and poorly abstracted interfaces and kernel exposing and locked into constructs inadvertently.

33.9.3 Competition Between Inner Nodes and Threads

cgroup v1 allowed threads to be in any cgroups which created an interesting problem where threads belonging to a parent cgroup and its children cgroups competed for resources. This was nasty as two different types of entities competed and there was no obvious way to settle it. Different controllers did different things.

The cpu controller considered threads and cgroups as equivalents and mapped nice levels to cgroup weights. This worked for some cases but fell flat when children wanted to be allocated specific ratios of CPU cycles and the number of internal threads fluctuated - the ratios constantly changed as the number of competing entities fluctuated. There also were other issues. The mapping from nice level to weight wasn't obvious or universal, and there were various other knobs which simply weren't available for threads.

The io controller implicitly created a hidden leaf node for each cgroup to host the threads. The hidden leaf had its own copies of all the knobs with `leaf_` prefixed. While this allowed equivalent control over internal threads, it was with serious drawbacks. It always added an extra layer of nesting which wouldn't be necessary otherwise, made the interface messy and significantly complicated the implementation.

The memory controller didn't have a way to control what happened between internal tasks and child cgroups and the behavior was not clearly defined. There were attempts to add ad-

hoc behaviors and knobs to tailor the behavior to specific workloads which would have led to problems extremely difficult to resolve in the long term.

Multiple controllers struggled with internal tasks and came up with different ways to deal with it; unfortunately, all the approaches were severely flawed and, furthermore, the widely different behaviors made cgroup as a whole highly inconsistent.

This clearly is a problem which needs to be addressed from cgroup core in a uniform way.

33.9.4 Other Interface Issues

cgroup v1 grew without oversight and developed a large number of idiosyncrasies and inconsistencies. One issue on the cgroup core side was how an empty cgroup was notified - a userland helper binary was forked and executed for each event. The event delivery wasn't recursive or delegatable. The limitations of the mechanism also led to in-kernel event delivery filtering mechanism further complicating the interface.

Controller interfaces were problematic too. An extreme example is controllers completely ignoring hierarchical organization and treating all cgroups as if they were all located directly under the root cgroup. Some controllers exposed a large amount of inconsistent implementation details to userland.

There also was no consistency across controllers. When a new cgroup was created, some controllers defaulted to not imposing extra restrictions while others disallowed any resource usage until explicitly configured. Configuration knobs for the same type of control used widely differing naming schemes and formats. Statistics and information knobs were named arbitrarily and used different formats and units even in the same controller.

cgroup v2 establishes common conventions where appropriate and updates controllers so that they expose minimal and consistent interfaces.

33.9.5 Controller Issues and Remedies

Memory

The original lower boundary, the soft limit, is defined as a limit that is per default unset. As a result, the set of cgroups that global reclaim prefers is opt-in, rather than opt-out. The costs for optimizing these mostly negative lookups are so high that the implementation, despite its enormous size, does not even provide the basic desirable behavior. First off, the soft limit has no hierarchical meaning. All configured groups are organized in a global rbtree and treated like equal peers, regardless where they are located in the hierarchy. This makes subtree delegation impossible. Second, the soft limit reclaim pass is so aggressive that it not just introduces high allocation latencies into the system, but also impacts system performance due to overreclaim, to the point where the feature becomes self-defeating.

The memory.low boundary on the other hand is a top-down allocated reserve. A cgroup enjoys reclaim protection when it's within its effective low, which makes delegation of subtrees possible. It also enjoys having reclaim pressure proportional to its overage when above its effective low.

The original high boundary, the hard limit, is defined as a strict limit that can not budge, even if the OOM killer has to be called. But this generally goes against the goal of making the most out of the available memory. The memory consumption of workloads varies during runtime, and

that requires users to overcommit. But doing that with a strict upper limit requires either a fairly accurate prediction of the working set size or adding slack to the limit. Since working set size estimation is hard and error prone, and getting it wrong results in OOM kills, most users tend to err on the side of a looser limit and end up wasting precious resources.

The memory.high boundary on the other hand can be set much more conservatively. When hit, it throttles allocations by forcing them into direct reclaim to work off the excess, but it never invokes the OOM killer. As a result, a high boundary that is chosen too aggressively will not terminate the processes, but instead it will lead to gradual performance degradation. The user can monitor this and make corrections until the minimal memory footprint that still gives acceptable performance is found.

In extreme cases, with many concurrent allocations and a complete breakdown of reclaim progress within the group, the high boundary can be exceeded. But even then it's mostly better to satisfy the allocation from the slack available in other groups or the rest of the system than killing the group. Otherwise, memory.max is there to limit this type of spillover and ultimately contain buggy or even malicious applications.

Setting the original memory.limit_in_bytes below the current usage was subject to a race condition, where concurrent charges could cause the limit setting to fail. memory.max on the other hand will first set the limit to prevent new charges, and then reclaim and OOM kill until the new limit is met - or the task writing to memory.max is killed.

The combined memory+swap accounting and limiting is replaced by real control over swap space.

The main argument for a combined memory+swap facility in the original cgroup design was that global or parental pressure would always be able to swap all anonymous memory of a child group, regardless of the child's own (possibly untrusted) configuration. However, untrusted groups can sabotage swapping by other means - such as referencing its anonymous memory in a tight loop - and an admin can not assume full swappability when overcommitting untrusted jobs.

For trusted jobs, on the other hand, a combined counter is not an intuitive userspace interface, and it flies in the face of the idea that cgroup controllers should account and limit specific physical resources. Swap space is a resource like all others in the system, and that's why unified hierarchy allows distributing it separately.

CHAPTER

THIRTYFOUR

CIFS

34.1 Introduction

This is the client VFS module for the SMB3 NAS protocol as well as for older dialects such as the Common Internet File System (CIFS) protocol which was the successor to the Server Message Block (SMB) protocol, the native file sharing mechanism for most early PC operating systems. New and improved versions of CIFS are now called SMB2 and SMB3. Use of SMB3 (and later, including SMB3.1.1 the most current dialect) is strongly preferred over using older dialects like CIFS due to security reasons. All modern dialects, including the most recent, SMB3.1.1, are supported by the CIFS VFS module. The SMB3 protocol is implemented and supported by all major file servers such as Windows (including Windows 2019 Server), as well as by Samba (which provides excellent CIFS/SMB2/SMB3 server support and tools for Linux and many other operating systems). Apple systems also support SMB3 well, as do most Network Attached Storage vendors, so this network filesystem client can mount to a wide variety of systems. It also supports mounting to the cloud (for example Microsoft Azure), including the necessary security features.

The intent of this module is to provide the most advanced network file system function for SMB3 compliant servers, including advanced security features, excellent parallelized high performance i/o, better POSIX compliance, secure per-user session establishment, encryption, high performance safe distributed caching (leases/oplocks), optional packet signing, large files, Unicode support and other internationalization improvements. Since both Samba server and this filesystem client support the CIFS Unix extensions, and the Linux client also suppors SMB3 POSIX extensions, the combination can provide a reasonable alternative to other network and cluster file systems for fileserving in some Linux to Linux environments, not just in Linux to Windows (or Linux to Mac) environments.

This filesystem has a mount utility (mount.cifs) and various user space tools (including smbinfo and setcifsacl) that can be obtained from

https://git.samba.org/?p=cifs-utils.git

or

git://git.samba.org/cifs-utils.git

mount.cifs should be installed in the directory with the other mount helpers.

For more information on the module see the project wiki page at

https://wiki.samba.org/index.php/LinuxCIFS

and

https://wiki.samba.org/index.php/LinuxCIFS_utils

34.2 Usage

This module supports the SMB3 family of advanced network protocols (as well as older dialects, originally called "CIFS" or SMB1).

The CIFS VFS module for Linux supports many advanced network filesystem features such as hierarchical DFS like namespace, hardlinks, locking and more. It was designed to comply with the SNIA CIFS Technical Reference (which supersedes the 1992 X/Open SMB Standard) as well as to perform best practice practical interoperability with Windows 2000, Windows XP, Samba and equivalent servers. This code was developed in participation with the Protocol Freedom Information Foundation. CIFS and now SMB3 has now become a defacto standard for interoperating between Macs and Windows and major NAS appliances.

Please see MS-SMB2 (for detailed SMB2/SMB3/SMB3.1.1 protocol specification) or https://samba.org/samba/PFIF/ for more details.

For questions or bug reports please contact:

smfrench@gmail.com

See the project page at: https://wiki.samba.org/index.php/LinuxCIFS_utils

34.2.1 Build instructions

For Linux:

1) Download the kernel (e.g. from https://www.kernel.org) and change directory into the top of the kernel directory tree (e.g. /usr/src/linux-2.5.73)
2) make menuconfig (or make xconfig)
3) select cifs from within the network filesystem choices
4) save and exit
5) make

34.2.2 Installation instructions

If you have built the CIFS vfs as module (successfully) simply type `make modules_install` (or if you prefer, manually copy the file to the modules directory e.g. /lib/modules/6.3.0-060300-generic/kernel/fs/smb/client/cifs.ko).

If you have built the CIFS vfs into the kernel itself, follow the instructions for your distribution on how to install a new kernel (usually you would simply type `make install`).

If you do not have the utility mount.cifs (in the Samba 4.x source tree and on the CIFS VFS web site) copy it to the same directory in which mount helpers reside (usually /sbin). Although the helper software is not required, mount.cifs is recommended. Most distros include a `cifs-utils` package that includes this utility so it is recommended to install this.

Note that running the Winbind pam/nss module (logon service) on all of your Linux clients is useful in mapping Uids and Gids consistently across the domain to the proper network user. The mount.cifs mount helper can be found at cifs-utils.git on git.samba.org

If cifs is built as a module, then the size and number of network buffers and maximum number of simultaneous requests to one server can be configured. Changing these from their defaults is not recommended. By executing modinfo:

```
modinfo <path to cifs.ko>
```

on kernel/fs/smb/client/cifs.ko the list of configuration changes that can be made at module initialization time (by running insmod cifs.ko) can be seen.

34.2.3 Recommendations

To improve security the SMB2.1 dialect or later (usually will get SMB3.1.1) is now the new default. To use old dialects (e.g. to mount Windows XP) use "vers=1.0" on mount (or vers=2.0 for Windows Vista). Note that the CIFS (vers=1.0) is much older and less secure than the default dialect SMB3 which includes many advanced security features such as downgrade attack detection and encrypted shares and stronger signing and authentication algorithms. There are additional mount options that may be helpful for SMB3 to get improved POSIX behavior (NB: can use vers=3 to force SMB3 or later, never 2.1):

`mfsymlinks` and either `cifsacl` or `modefromsid` (usually with `idsfromsid`)

34.2.4 Allowing User Mounts

To permit users to mount and unmount over directories they own is possible with the cifs vfs. A way to enable such mounting is to mark the mount.cifs utility as suid (e.g. `chmod +s /sbin/mount.cifs`). To enable users to umount shares they mount requires

1) mount.cifs version 1.4 or later
2) an entry for the share in /etc/fstab indicating that a user may unmount it e.g.:

```
//server/usersharename  /mnt/username cifs user 0 0
```

Note that when the mount.cifs utility is run suid (allowing user mounts), in order to reduce risks, the `nosuid` mount flag is passed in on mount to disallow execution of an suid program mounted on the remote target. When mount is executed as root, nosuid is not passed in by default, and execution of suid programs on the remote target would be enabled by default. This can be changed, as with nfs and other filesystems, by simply specifying `nosuid` among the mount options. For user mounts though to be able to pass the suid flag to mount requires rebuilding mount.cifs with the following flag: CIFS_ALLOW_USR_SUID

There is a corresponding manual page for cifs mounting in the Samba 3.0 and later source tree in docs/manpages/mount.cifs.8

34.2.5 Allowing User Unmounts

To permit users to unmount directories that they have user mounted (see above), the utility umount.cifs may be used. It may be invoked directly, or if umount.cifs is placed in /sbin, umount can invoke the cifs umount helper (at least for most versions of the umount utility) for umount of cifs mounts, unless umount is invoked with -i (which will avoid invoking a umount helper). As with mount.cifs, to enable user unmounts umount.cifs must be marked as suid (e.g. `chmod +s /sbin/umount.cifs`) or equivalent (some distributions allow adding entries to a file to the /etc/permissions file to achieve the equivalent suid effect). For this utility to succeed the target path must be a cifs mount, and the uid of the current user must match the uid of the user who mounted the resource.

Also note that the customary way of allowing user mounts and unmounts is (instead of using mount.cifs and unmount.cifs as suid) to add a line to the file /etc/fstab for each //server/share you wish to mount, but this can become unwieldy when potential mount targets include many or unpredictable UNC names.

34.2.6 Samba Considerations

Most current servers support SMB2.1 and SMB3 which are more secure, but there are useful protocol extensions for the older less secure CIFS dialect, so to get the maximum benefit if mounting using the older dialect (CIFS/SMB1), we recommend using a server that supports the SNIA CIFS Unix Extensions standard (e.g. almost any version of Samba ie version 2.2.5 or later) but the CIFS vfs works fine with a wide variety of CIFS servers. Note that uid, gid and file permissions will display default values if you do not have a server that supports the Unix extensions for CIFS (such as Samba 2.2.5 or later). To enable the Unix CIFS Extensions in the Samba server, add the line:

```
unix extensions = yes
```

to your smb.conf file on the server. Note that the following smb.conf settings are also useful (on the Samba server) when the majority of clients are Unix or Linux:

```
case sensitive = yes
delete readonly = yes
ea support = yes
```

Note that server ea support is required for supporting xattrs from the Linux cifs client, and that EA support is present in later versions of Samba (e.g. 3.0.6 and later (also EA support works in all versions of Windows, at least to shares on NTFS filesystems). Extended Attribute (xattr) support is an optional feature of most Linux filesystems which may require enabling via make menuconfig. Client support for extended attributes (user xattr) can be disabled on a per-mount basis by specifying `nouser_xattr` on mount.

The CIFS client can get and set POSIX ACLs (getfacl, setfacl) to Samba servers version 3.10 and later. Setting POSIX ACLs requires enabling both XATTR and then POSIX support in the CIFS configuration options when building the cifs module. POSIX ACL support can be disabled on a per mount basic by specifying `noacl` on mount.

Some administrators may want to change Samba's smb.conf `map archive` and `create mask` parameters from the default. Unless the create mask is changed newly created files can end up with an unnecessarily restrictive default mode, which may not be what you want, although if the CIFS Unix extensions are enabled on the server and client, subsequent setattr calls (e.g. chmod)

can fix the mode. Note that creating special devices (mknod) remotely may require specifying a mkdev function to Samba if you are not using Samba 3.0.6 or later. For more information on these see the manual pages (`man smb.conf`) on the Samba server system. Note that the cifs vfs, unlike the smbfs vfs, does not read the smb.conf on the client system (the few optional settings are passed in on mount via -o parameters instead). Note that Samba 2.2.7 or later includes a fix that allows the CIFS VFS to delete open files (required for strict POSIX compliance). Windows Servers already supported this feature. Samba server does not allow symlinks that refer to files outside of the share, so in Samba versions prior to 3.0.6, most symlinks to files with absolute paths (ie beginning with slash) such as:

```
ln -s /mnt/foo bar
```

would be forbidden. Samba 3.0.6 server or later includes the ability to create such symlinks safely by converting unsafe symlinks (ie symlinks to server files that are outside of the share) to a samba specific format on the server that is ignored by local server applications and non-cifs clients and that will not be traversed by the Samba server). This is opaque to the Linux client application using the cifs vfs. Absolute symlinks will work to Samba 3.0.5 or later, but only for remote clients using the CIFS Unix extensions, and will be invisible to Windows clients and typically will not affect local applications running on the same server as Samba.

34.2.7 Use instructions

Once the CIFS VFS support is built into the kernel or installed as a module (cifs.ko), you can use mount syntax like the following to access Samba or Mac or Windows servers:

```
mount -t cifs //9.53.216.11/e$ /mnt -o username=myname,password=mypassword
```

Before -o the option -v may be specified to make the mount.cifs mount helper display the mount steps more verbosely. After -o the following commonly used cifs vfs specific options are supported:

```
username=<username>
password=<password>
domain=<domain name>
```

Other cifs mount options are described below. Use of TCP names (in addition to ip addresses) is available if the mount helper (mount.cifs) is installed. If you do not trust the server to which are mounted, or if you do not have cifs signing enabled (and the physical network is insecure), consider use of the standard mount options `noexec` and `nosuid` to reduce the risk of running an altered binary on your local system (downloaded from a hostile server or altered by a hostile router).

Although mounting using format corresponding to the CIFS URL specification is not possible in mount.cifs yet, it is possible to use an alternate format for the server and sharename (which is somewhat similar to NFS style mount syntax) instead of the more widely used UNC format (i.e. \servershare):

```
mount -t cifs tcp_name_of_server:share_name /mnt -o user=myname,pass=mypasswd
```

When using the mount helper mount.cifs, passwords may be specified via alternate mechanisms, instead of specifying it after -o using the normal `pass=` syntax on the command line: 1) By

including it in a credential file. Specify credentials=filename as one of the mount options. Credential files contain two lines:

```
username=someuser
password=your_password
```

2) By specifying the password in the PASSWD environment variable (similarly the user name can be taken from the USER environment variable).
3) By specifying the password in a file by name via PASSWD_FILE
4) By specifying the password in a file by file descriptor via PASSWD_FD

If no password is provided, mount.cifs will prompt for password entry

34.2.8 Restrictions

Servers must support either ”pure-TCP” (port 445 TCP/IP CIFS connections) or RFC 1001/1002 support for ”Netbios-Over-TCP/IP.” This is not likely to be a problem as most servers support this.

Valid filenames differ between Windows and Linux. Windows typically restricts filenames which contain certain reserved characters (e.g.the character : which is used to delimit the beginning of a stream name by Windows), while Linux allows a slightly wider set of valid characters in filenames. Windows servers can remap such characters when an explicit mapping is specified in the Server's registry. Samba starting with version 3.10 will allow such filenames (ie those which contain valid Linux characters, which normally would be forbidden for Windows/CIFS semantics) as long as the server is configured for Unix Extensions (and the client has not disabled /proc/fs/cifs/LinuxExtensionsEnabled). In addition the mount option `mapposix` can be used on CIFS (vers=1.0) to force the mapping of illegal Windows/NTFS/SMB characters to a remap range (this mount parameter is the default for SMB3). This remap (`mapposix`) range is also compatible with Mac (and ”Services for Mac” on some older Windows).

34.2.9 CIFS VFS Mount Options

A partial list of the supported mount options follows:

username
The user name to use when trying to establish the CIFS session.

password
The user password. If the mount helper is installed, the user will be prompted for password if not supplied.

ip
The ip address of the target server

unc
The target server Universal Network Name (export) to mount.

domain
Set the SMB/CIFS workgroup name prepended to the username during CIFS session establishment

forceuid
Set the default uid for inodes to the uid passed in on mount. For mounts to servers which do support the CIFS Unix extensions, such as a properly configured Samba server, the server provides the uid, gid and mode so this parameter should not be specified unless the server and clients uid and gid numbering differ. If the server and client are in the same domain (e.g. running winbind or nss_ldap) and the server supports the Unix Extensions then the uid and gid can be retrieved from the server (and uid and gid would not have to be specified on the mount. For servers which do not support the CIFS Unix extensions, the default uid (and gid) returned on lookup of existing files will be the uid (gid) of the person who executed the mount (root, except when mount.cifs is configured setuid for user mounts) unless the `uid=` (gid) mount option is specified. Also note that permission checks (authorization checks) on accesses to a file occur at the server, but there are cases in which an administrator may want to restrict at the client as well. For those servers which do not report a uid/gid owner (such as Windows), permissions can also be checked at the client, and a crude form of client side permission checking can be enabled by specifying file_mode and dir_mode on the client. (default)

forcegid
(similar to above but for the groupid instead of uid) (default)

noforceuid
Fill in file owner information (uid) by requesting it from the server if possible. With this option, the value given in the uid= option (on mount) will only be used if the server can not support returning uids on inodes.

noforcegid
(similar to above but for the group owner, gid, instead of uid)

uid
Set the default uid for inodes, and indicate to the cifs kernel driver which local user mounted. If the server supports the unix extensions the default uid is not used to fill in the owner fields of inodes (files) unless the `forceuid` parameter is specified.

gid
Set the default gid for inodes (similar to above).

file_mode
If CIFS Unix extensions are not supported by the server this overrides the default mode for file inodes.

fsc
Enable local disk caching using FS-Cache (off by default). This option could be useful to improve performance on a slow link, heavily loaded server and/or network where reading from the disk is faster than reading from the server (over the network). This could also impact scalability positively as the number of calls to the server are reduced. However, local caching is not suitable for all workloads for e.g. read-once type workloads. So, you need to consider carefully your workload/scenario before using this option. Currently, local disk caching is functional for CIFS files opened as read-only.

dir_mode
If CIFS Unix extensions are not supported by the server this overrides the default

mode for directory inodes.

port
attempt to contact the server on this tcp port, before trying the usual ports (port 445, then 139).

iocharset
Codepage used to convert local path names to and from Unicode. Unicode is used by default for network path names if the server supports it. If iocharset is not specified then the nls_default specified during the local client kernel build will be used. If server does not support Unicode, this parameter is unused.

rsize
default read size (usually 16K). The client currently can not use rsize larger than CIFSMaxBufSize. CIFSMaxBufSize defaults to 16K and may be changed (from 8K to the maximum kmalloc size allowed by your kernel) at module install time for cifs.ko. Setting CIFSMaxBufSize to a very large value will cause cifs to use more memory and may reduce performance in some cases. To use rsize greater than 127K (the original cifs protocol maximum) also requires that the server support a new Unix Capability flag (for very large read) which some newer servers (e.g. Samba 3.0.26 or later) do. rsize can be set from a minimum of 2048 to a maximum of 130048 (127K or CIFSMaxBufSize, whichever is smaller)

wsize
default write size (default 57344) maximum wsize currently allowed by CIFS is 57344 (fourteen 4096 byte pages)

actimeo=n
attribute cache timeout in seconds (default 1 second). After this timeout, the cifs client requests fresh attribute information from the server. This option allows to tune the attribute cache timeout to suit the workload needs. Shorter timeouts mean better the cache coherency, but increased number of calls to the server. Longer timeouts mean reduced number of calls to the server at the expense of less stricter cache coherency checks (i.e. incorrect attribute cache for a short period of time).

rw
mount the network share read-write (note that the server may still consider the share read-only)

ro
mount network share read-only

version
used to distinguish different versions of the mount helper utility (not typically needed)

sep
if first mount option (after the -o), overrides the comma as the separator between the mount parameters. e.g.:

```
-o user=myname,password=mypassword,domain=mydom
```

could be passed instead with period as the separator by:

```
-o sep=.user=myname.password=mypassword.domain=mydom
```

this might be useful when comma is contained within username or password or domain. This option is less important when the cifs mount helper cifs.mount (version 1.1 or later) is used.

nosuid
Do not allow remote executables with the suid bit program to be executed. This is only meaningful for mounts to servers such as Samba which support the CIFS Unix Extensions. If you do not trust the servers in your network (your mount targets) it is recommended that you specify this option for greater security.

exec
Permit execution of binaries on the mount.

noexec
Do not permit execution of binaries on the mount.

dev
Recognize block devices on the remote mount.

nodev
Do not recognize devices on the remote mount.

suid
Allow remote files on this mountpoint with suid enabled to be executed (default for mounts when executed as root, nosuid is default for user mounts).

credentials
Although ignored by the cifs kernel component, it is used by the mount helper, mount.cifs. When mount.cifs is installed it opens and reads the credential file specified in order to obtain the userid and password arguments which are passed to the cifs vfs.

guest
Although ignored by the kernel component, the mount.cifs mount helper will not prompt the user for a password if guest is specified on the mount options. If no password is specified a null password will be used.

perm
Client does permission checks (vfs_permission check of uid and gid of the file against the mode and desired operation), Note that this is in addition to the normal ACL check on the target machine done by the server software. Client permission checking is enabled by default.

noperm
Client does not do permission checks. This can expose files on this mount to access by other users on the local client system. It is typically only needed when the server supports the CIFS Unix Extensions but the UIDs/GIDs on the client and server system do not match closely enough to allow access by the user doing the mount, but it may be useful with non CIFS Unix Extension mounts for cases in which the default mode is specified on the mount but is not to be enforced on the client (e.g. perhaps when MultiUserMount is enabled) Note that this does not affect the normal ACL check on the target machine done by the server software (of the server ACL against the user name provided at mount time).

serverino
Use server's inode numbers instead of generating automatically incrementing inode numbers on the client. Although this will make it easier to spot hardlinked files (as they will have the same inode numbers) and inode numbers may be persistent, note that the server does not guarantee that the inode numbers are unique if multiple server side mounts are exported under a single share (since inode numbers on the servers might not be unique if multiple filesystems are mounted under the same shared higher level directory). Note that some older (e.g. pre-Windows 2000) do not support returning UniqueIDs or the CIFS Unix Extensions equivalent and for those this mount option will have no effect. Exporting cifs mounts under nfsd requires this mount option on the cifs mount. This is now the default if server supports the required network operation.

noserverino
Client generates inode numbers (rather than using the actual one from the server). These inode numbers will vary after unmount or reboot which can confuse some applications, but not all server filesystems support unique inode numbers.

setuids
If the CIFS Unix extensions are negotiated with the server the client will attempt to set the effective uid and gid of the local process on newly created files, directories, and devices (create, mkdir, mknod). If the CIFS Unix Extensions are not negotiated, for newly created files and directories instead of using the default uid and gid specified on the mount, cache the new file's uid and gid locally which means that the uid for the file can change when the inode is reloaded (or the user remounts the share).

nosetuids
The client will not attempt to set the uid and gid on on newly created files, directories, and devices (create, mkdir, mknod) which will result in the server setting the uid and gid to the default (usually the server uid of the user who mounted the share). Letting the server (rather than the client) set the uid and gid is the default. If the CIFS Unix Extensions are not negotiated then the uid and gid for new files will appear to be the uid (gid) of the mounter or the uid (gid) parameter specified on the mount.

netbiosname
When mounting to servers via port 139, specifies the RFC1001 source name to use to represent the client netbios machine name when doing the RFC1001 netbios session initialize.

direct
Do not do inode data caching on files opened on this mount. This precludes mmapping files on this mount. In some cases with fast networks and little or no caching benefits on the client (e.g. when the application is doing large sequential reads bigger than page size without rereading the same data) this can provide better performance than the default behavior which caches reads (readahead) and writes (writebehind) through the local Linux client pagecache if oplock (caching token) is granted and held. Note that direct allows write operations larger than page size to be sent to the server.

strictcache
Use for switching on strict cache mode. In this mode the client read from the

cache all the time it has Oplock Level II, otherwise - read from the server. All written data are stored in the cache, but if the client doesn't have Exclusive Oplock, it writes the data to the server.

rwpidforward
Forward pid of a process who opened a file to any read or write operation on that file. This prevent applications like WINE from failing on read and write if we use mandatory brlock style.

acl
Allow setfacl and getfacl to manage posix ACLs if server supports them. (default)

noacl
Do not allow setfacl and getfacl calls on this mount

user_xattr
Allow getting and setting user xattrs (those attributes whose name begins with `user.` or `os2.`) as OS/2 EAs (extended attributes) to the server. This allows support of the setfattr and getfattr utilities. (default)

nouser_xattr
Do not allow getfattr/setfattr to get/set/list xattrs

mapchars
Translate six of the seven reserved characters (not backslash):

```
*?<>|:
```

to the remap range (above 0xF000), which also allows the CIFS client to recognize files created with such characters by Windows's POSIX emulation. This can also be useful when mounting to most versions of Samba (which also forbids creating and opening files whose names contain any of these seven characters). This has no effect if the server does not support Unicode on the wire.

nomapchars
Do not translate any of these seven characters (default).

nocase
Request case insensitive path name matching (case sensitive is the default if the server supports it). (mount option `ignorecase` is identical to `nocase`)

posixpaths
If CIFS Unix extensions are supported, attempt to negotiate posix path name support which allows certain characters forbidden in typical CIFS filenames, without requiring remapping. (default)

noposixpaths
If CIFS Unix extensions are supported, do not request posix path name support (this may cause servers to reject creatingfile with certain reserved characters).

nounix
Disable the CIFS Unix Extensions for this mount (tree connection). This is rarely needed, but it may be useful in order to turn off multiple settings all at once (ie posix acls, posix locks, posix paths, symlink support and retrieving uids/gids/mode from the server) or to work around a bug in server which implement the Unix Extensions.

nobrl
Do not send byte range lock requests to the server. This is necessary for certain applications that break with cifs style mandatory byte range locks (and most cifs servers do not yet support requesting advisory byte range locks).

forcemandatorylock
Even if the server supports posix (advisory) byte range locking, send only mandatory lock requests. For some (presumably rare) applications, originally coded for DOS/Windows, which require Windows style mandatory byte range locking, they may be able to take advantage of this option, forcing the cifs client to only send mandatory locks even if the cifs server would support posix advisory locks. `forcemand` is accepted as a shorter form of this mount option.

nostrictsync
If this mount option is set, when an application does an fsync call then the cifs client does not send an SMB Flush to the server (to force the server to write all dirty data for this file immediately to disk), although cifs still sends all dirty (cached) file data to the server and waits for the server to respond to the write. Since SMB Flush can be very slow, and some servers may be reliable enough (to risk delaying slightly flushing the data to disk on the server), turning on this option may be useful to improve performance for applications that fsync too much, at a small risk of server crash. If this mount option is not set, by default cifs will send an SMB flush request (and wait for a response) on every fsync call.

nodfs
Disable DFS (global name space support) even if the server claims to support it. This can help work around a problem with parsing of DFS paths with Samba server versions 3.0.24 and 3.0.25.

remount
remount the share (often used to change from ro to rw mounts or vice versa)

cifsacl
Report mode bits (e.g. on stat) based on the Windows ACL for the file. (EXPERIMENTAL)

servern
Specify the server 's netbios name (RFC1001 name) to use when attempting to setup a session to the server. This is needed for mounting to some older servers (such as OS/2 or Windows 98 and Windows ME) since they do not support a default server name. A server name can be up to 15 characters long and is usually uppercased.

sfu
When the CIFS Unix Extensions are not negotiated, attempt to create device files and fifos in a format compatible with Services for Unix (SFU). In addition retrieve bits 10-12 of the mode via the SETFILEBITS extended attribute (as SFU does). In the future the bottom 9 bits of the mode also will be emulated using queries of the security descriptor (ACL).

mfsymlinks
Enable support for Minshall+French symlinks (see http://wiki.samba.org/index.php/UNIX_Extensions#Minshall.2BFrench_symlinks) This option is ignored when specified together with the 'sfu' option. Minshall+French symlinks are used even if the server supports the CIFS Unix Extensions.

sign
Must use packet signing (helps avoid unwanted data modification by intermediate systems in the route). Note that signing does not work with lanman or plaintext authentication.

seal
Must seal (encrypt) all data on this mounted share before sending on the network. Requires support for Unix Extensions. Note that this differs from the sign mount option in that it causes encryption of data sent over this mounted share but other shares mounted to the same server are unaffected.

locallease
This option is rarely needed. Fcntl F_SETLEASE is used by some applications such as Samba and NFSv4 server to check to see whether a file is cacheable. CIFS has no way to explicitly request a lease, but can check whether a file is cacheable (oplocked). Unfortunately, even if a file is not oplocked, it could still be cacheable (ie cifs client could grant fcntl leases if no other local processes are using the file) for cases for example such as when the server does not support oplocks and the user is sure that the only updates to the file will be from this client. Specifying this mount option will allow the cifs client to check for leases (only) locally for files which are not oplocked instead of denying leases in that case. (EXPERIMENTAL)

sec
Security mode. Allowed values are:

none
attempt to connection as a null user (no name)

krb5
Use Kerberos version 5 authentication

krb5i
Use Kerberos authentication and packet signing

ntlm
Use NTLM password hashing (default)

ntlmi
Use NTLM password hashing with signing (if /proc/fs/cifs/PacketSigningEnabled on or if server requires signing also can be the default)

ntlmv2
Use NTLMv2 password hashing

ntlmv2i
Use NTLMv2 password hashing with packet signing

lanman
(if configured in kernel config) use older lanman hash

hard
Retry file operations if server is not responding

soft
Limit retries to unresponsive servers (usually only one retry) before returning an error. (default)

The mount.cifs mount helper also accepts a few mount options before -o including:

-S	take password from stdin (equivalent to setting the environment variable `PASSWD_FD=0`
-V	print mount.cifs version
-?	display simple usage information

With most 2.6 kernel versions of modutils, the version of the cifs kernel module can be displayed via modinfo.

34.2.10 Misc /proc/fs/cifs Flags and Debug Info

Informational pseudo-files:

DebugData	Displays information about active CIFS sessions and shares, features enabled as well as the cifs.ko version.
Stats	Lists summary resource usage information as well as per share statistics.
open_files	List all the open file handles on all active SMB sessions.
mount_params	List of all mount parameters available for the module

Configuration pseudo-files:

SecurityFlags	Flags which control security negotiation and also packet signing. Authentication (may/must) flags (e.g. for NTLM and/or NTLMv2) may be combined with the signing flags. Specifying two different password hashing mechanisms (as "must use") on the other hand does not make much sense. Default flags are:
	`0x07007`
	(NTLM, NTLMv2 and packet signing allowed). The maximum allowable flags if you want to allow mounts to servers using weaker password hashes is 0x37037 (lanman, plaintext, ntlm, ntlmv2, signing allowed). Some SecurityFlags require the corresponding menuconfig options to be enabled. Enabling plaintext authentication currently requires also enabling lanman authentication in the security flags because the cifs module only supports sending laintext passwords using the older lanman dialect form of the session setup SMB. (e.g. for authentication using plain text passwords, set the SecurityFlags to 0x30030):
	`may use packet signing 0x00001`
	`must use packet signing 0x01001`
	`may use NTLM (most common password hash) 0x00002`
	`must use NTLM 0x02002`
	`may use NTLMv2 0x00004`
	`must use NTLMv2 0x04004`
	`may use Kerberos security 0x00008`
	`must use Kerberos 0x08008`
	`may use lanman (weak) password hash 0x00010`
	`must use lanman password hash 0x10010`
	`may use plaintext passwords 0x00020`
	`must use plaintext passwords 0x20020`
	`(reserved for future packet encryption) 0x00040`

cifsFYI	If set to non-zero value, additional debug information will be logged to the system error log. This field contains three flags controlling different classes of debugging entries. The maximum value it can be set to

These experimental features and tracing can be enabled by changing flags in /proc/fs/cifs (after the cifs module has been installed or built into the kernel, e.g. insmod cifs). To enable a feature set it to 1 e.g. to enable tracing to the kernel message log type:

```
echo 7 > /proc/fs/cifs/cifsFYI
```

cifsFYI functions as a bit mask. Setting it to 1 enables additional kernel logging of various informational messages. 2 enables logging of non-zero SMB return codes while 4 enables logging of requests that take longer than one second to complete (except for byte range lock requests). Setting it to 4 requires CONFIG_CIFS_STATS2 to be set in kernel configuration (.config). Setting it to seven enables all three. Finally, tracing the start of smb requests and responses can be enabled via:

```
echo 1 > /proc/fs/cifs/traceSMB
```

Per share (per client mount) statistics are available in /proc/fs/cifs/Stats. Additional information is available if CONFIG_CIFS_STATS2 is enabled in the kernel configuration (.config). The statistics returned include counters which represent the number of attempted and failed (ie non-zero return code from the server) SMB3 (or cifs) requests grouped by request type (read, write, close etc.). Also recorded is the total bytes read and bytes written to the server for that share. Note that due to client caching effects this can be less than the number of bytes read and written by the application running on the client. Statistics can be reset to zero by `echo 0 > /proc/fs/cifs/Stats` which may be useful if comparing performance of two different scenarios.

Also note that `cat /proc/fs/cifs/DebugData` will display information about the active sessions and the shares that are mounted.

Enabling Kerberos (extended security) works but requires version 1.2 or later of the helper program cifs.upcall to be present and to be configured in the /etc/request-key.conf file. The cifs.upcall helper program is from the Samba project(https://www.samba.org). NTLM and NTLMv2 and LANMAN support do not require this helper. Note that NTLMv2 security (which does not require the cifs.upcall helper program), instead of using Kerberos, is sufficient for some use cases.

DFS support allows transparent redirection to shares in an MS-DFS name space. In addition, DFS support for target shares which are specified as UNC names which begin with host names (rather than IP addresses) requires a user space helper (such as cifs.upcall) to be present in order to translate host names to ip address, and the user space helper must also be configured in the file /etc/request-key.conf. Samba, Windows servers and many NAS appliances support DFS as a way of constructing a global name space to ease network configuration and improve reliability.

To use cifs Kerberos and DFS support, the Linux keyutils package should be installed and something like the following lines should be added to the /etc/request-key.conf file:

```
create cifs.spnego * * /usr/local/sbin/cifs.upcall %k
create dns_resolver * * /usr/local/sbin/cifs.upcall %k
```

34.2.11 CIFS kernel module parameters

These module parameters can be specified or modified either during the time of module loading or during the runtime by using the interface:

```
/sys/module/cifs/parameters/<param>
```

i.e.:

```
echo "value" > /sys/module/cifs/parameters/<param>
```

More detailed descriptions of the available module parameters and their values can be seen by doing:

modinfo cifs (or modinfo smb3)

1. enable_oplocks	Enable or disable oplocks. Oplocks are enabled by default. [Y/y/1]. To disable use any of [N/n/0].

34.3 TODO

As of 6.7 kernel. See https://wiki.samba.org/index.php/LinuxCIFSKernel for list of features added by release

34.3.1 A Partial List of Missing Features

Contributions are welcome. There are plenty of opportunities for visible, important contributions to this module. Here is a partial list of the known problems and missing features:

a) SMB3 (and SMB3.1.1) missing optional features: multichannel performance optimizations, algorithmic channel selection, directory leases optimizations, support for faster packet signing (GMAC), support for compression over the network, T10 copy offload ie "ODX" (copy chunk, and "Duplicate Extents" ioctl are currently the only two server side copy mechanisms supported)

b) Better optimized compounding and error handling for sparse file support, perhaps addition of new optional SMB3.1.1 fsctls to make collapse range and insert range more atomic

c) Support for SMB3.1.1 over QUIC (and perhaps other socket based protocols like SCTP)

d) quota support (needs minor kernel change since quota calls otherwise won't make it to network filesystems or deviceless filesystems).

e) Additional use cases can be optimized to use "compounding" (e.g. open/query/close and open/setinfo/close) to reduce the number of roundtrips to the server and improve performance. Various cases (stat, statfs, create, unlink, mkdir, xattrs) already have been improved by using compounding but more can be done. In addition we could significantly reduce redundant opens by using deferred close (with handle caching leases) and better using reference counters on file handles.

f) Finish inotify support so kde and gnome file list windows will autorefresh (partially complete by Asser). Needs minor kernel vfs change to support removing D_NOTIFY on a file.

g) Add GUI tool to configure /proc/fs/cifs settings and for display of the CIFS statistics (started)

h) implement support for security and trusted categories of xattrs (requires minor protocol extension) to enable better support for SELINUX

i) Add support for tree connect contexts (see MS-SMB2) a new SMB3.1.1 protocol feature (may be especially useful for virtualization).

j) Create UID mapping facility so server UIDs can be mapped on a per mount or a per server basis to client UIDs or nobody if no mapping exists. Also better integration with winbind for resolving SID owners

k) Add tools to take advantage of more smb3 specific ioctls and features (passthrough ioctl/fsctl is now implemented in cifs.ko to allow sending various SMB3 fsctls and query info and set info calls directly from user space) Add tools to make setting various non-POSIX metadata attributes easier from tools (e.g. extending what was done in smb-info tool).

l) encrypted file support (currently the attribute showing the file is encrypted on the server is reported, but changing the attribute is not supported).

m) improved stats gathering tools (perhaps integration with nfsometer?) to extend and make easier to use what is currently in /proc/fs/cifs/Stats

n) Add support for claims based ACLs (”DAC”)

o) mount helper GUI (to simplify the various configuration options on mount)

p) Expand support for witness protocol to allow for notification of share move, and server network adapter changes. Currently only notifications by the witness protocol for server move is supported by the Linux client.

q) Allow mount.cifs to be more verbose in reporting errors with dialect or unsupported feature errors. This would now be easier due to the implementation of the new mount API.

r) updating cifs documentation, and user guide.

s) Addressing bugs found by running a broader set of xfstests in standard file system xfstest suite.

t) split cifs and smb3 support into separate modules so legacy (and less secure) CIFS dialect can be disabled in environments that don't need it and simplify the code.

v) Additional testing of POSIX Extensions for SMB3.1.1

w) Support for the Mac SMB3.1.1 extensions to improve interop with Apple servers

x) Support for additional authentication options (e.g. IAKERB, peer-to-peer Kerberos, SCRAM and others supported by existing servers)

y) Improved tracing, more eBPF trace points, better scripts for performance analysis

34.3.2 Known Bugs

See https://bugzilla.samba.org - search on product "CifsVFS" for current bug list. Also check http://bugzilla.kernel.org (Product = File System, Component = CIFS) and xfstest results e.g. https://wiki.samba.org/index.php/Xfstest-results-smb3

34.3.3 Misc testing to do

1) check out max path names and max path name components against various server types. Try nested symlinks (8 deep). Return max path name in stat -f information
2) Improve xfstest's cifs/smb3 enablement and adapt xfstests where needed to test cifs/smb3 better
3) Additional performance testing and optimization using iozone and similar - there are some easy changes that can be done to parallelize sequential writes, and when signing is disabled to request larger read sizes (larger than negotiated size) and send larger write sizes to modern servers.
4) More exhaustively test against less common servers
5) Continue to extend the smb3 "buildbot" which does automated xfstesting against Windows, Samba and Azure currently - to add additional tests and to allow the buildbot to execute the tests faster. The URL for the buildbot is: http://smb3-test-rhel-75.southcentralus.cloudapp.azure.com
6) Address various coverity warnings (most are not bugs per-se, but the more warnings are addressed, the easier it is to spot real problems that static analyzers will point out in the future).

34.4 Changes

See https://wiki.samba.org/index.php/LinuxCIFSKernel for summary information about fixes/improvements to CIFS/SMB2/SMB3 support (changes to cifs.ko module) by kernel version (and cifs internal module version). This may be easier to read than parsing the output of "git log fs/smb/client" by release.

34.5 Authors

34.5.1 Original Author

Steve French (smfrench@gmail.com, sfrench@samba.org)

The author wishes to express his appreciation and thanks to: Andrew Tridgell (Samba team) for his early suggestions about SMB/CIFS VFS improvements. Thanks to IBM for allowing me time and test resources to pursue this project, to Jim McDonough from IBM (and the Samba Team) for his help, to the IBM Linux JFS team for explaining many esoteric Linux filesystem features. Jeremy Allison of the Samba team has done invaluable work in adding the server side of the original CIFS Unix extensions and reviewing and implementing portions of the newer CIFS POSIX extensions into the Samba 3 file server. Thank Dave Boutcher of IBM Rochester (author of the OS/400 smb/cifs filesystem client) for proving years ago that very good smb/cifs

clients could be done on Unix-like operating systems. Volker Lendecke, Andrew Tridgell, Urban Widmark, John Newbigin and others for their work on the Linux smbfs module. Thanks to the other members of the Storage Network Industry Association CIFS Technical Workgroup for their work specifying this highly complex protocol and finally thanks to the Samba team for their technical advice and encouragement.

34.5.2 Patch Contributors

- Zwane Mwaikambo
- Andi Kleen
- Amrut Joshi
- Shobhit Dayal
- Sergey Vlasov
- Richard Hughes
- Yury Umanets
- Mark Hamzy (for some of the early cifs IPv6 work)
- Domen Puncer
- Jesper Juhl (in particular for lots of whitespace/formatting cleanup)
- Vince Negri and Dave Stahl (for finding an important caching bug)
- Adrian Bunk (kcalloc cleanups)
- Miklos Szeredi
- Kazeon team for various fixes especially for 2.4 version.
- Asser Ferno (Change Notify support)
- Shaggy (Dave Kleikamp) for innumerable small fs suggestions and some good cleanup
- Gunter Kukkukk (testing and suggestions for support of old servers)
- Igor Mammedov (DFS support)
- Jeff Layton (many, many fixes, as well as great work on the cifs Kerberos code)
- Scott Lovenberg
- Pavel Shilovsky (for great work adding SMB2 support, and various SMB3 features)
- Aurelien Aptel (for DFS SMB3 work and some key bug fixes)
- Ronnie Sahlberg (for SMB3 xattr work, bug fixes, and lots of great work on compounding)
- Shirish Pargaonkar (for many ACL patches over the years)
- Sachin Prabhu (many bug fixes, including for reconnect, copy offload and security)
- Paulo Alcantara (for some excellent work in DFS, and in booting from SMB3)
- Long Li (some great work on RDMA, SMB Direct)

34.5.3 Test case and Bug Report contributors

Thanks to those in the community who have submitted detailed bug reports and debug of problems they have found: Jochen Dolze, David Blaine, Rene Scharfe, Martin Josefsson, Alexander Wild, Anthony Liguori, Lars Muller, Urban Widmark, Massimiliano Ferrero, Howard Owen, Olaf Kirch, Kieron Briggs, Nick Millington and others. Also special mention to the Stanford Checker (SWAT) which pointed out many minor bugs in error paths. Valuable suggestions also have come from Al Viro and Dave Miller.

And thanks to the IBM LTC and Power test teams and SuSE and Citrix and RedHat testers for finding multiple bugs during excellent stress test runs.

CHAPTER

THIRTYFIVE

CLEARING WARN_ONCE

WARN_ONCE / WARN_ON_ONCE / printk_once only emit a message once.

echo 1 > /sys/kernel/debug/clear_warn_once

clears the state and allows the warnings to print once again. This can be useful after test suite runs to reproduce problems.

CHAPTER

THIRTYSIX

CPU LOAD

Linux exports various bits of information via `/proc/stat` and `/proc/uptime` that userland tools, such as top(1), use to calculate the average time system spent in a particular state, for example:

```
$ iostat
Linux 2.6.18.3-exp (linmac)     02/20/2007

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
          10.01    0.00    2.92    5.44    0.00   81.63

...
```

Here the system thinks that over the default sampling period the system spent 10.01% of the time doing work in user space, 2.92% in the kernel, and was overall 81.63% of the time idle.

In most cases the `/proc/stat` information reflects the reality quite closely, however due to the nature of how/when the kernel collects this data sometimes it can not be trusted at all.

So how is this information collected? Whenever timer interrupt is signalled the kernel looks what kind of task was running at this moment and increments the counter that corresponds to this tasks kind/state. The problem with this is that the system could have switched between various states multiple times between two timer interrupts yet the counter is incremented only for the last state.

36.1 Example

If we imagine the system with one task that periodically burns cycles in the following manner:

```
 time line between two timer interrupts
|--------------------------------------|
 ^                                     ^
 |_ something begins working           |
                                       |_ something goes to sleep
                                      (only to be awaken quite soon)
```

In the above situation the system will be 0% loaded according to the `/proc/stat` (since the timer interrupt will always happen when the system is executing the idle handler), but in reality the load is closer to 99%.

One can imagine many more situations where this behavior of the kernel will lead to quite erratic information inside `/proc/stat`:

```
/* gcc -o hog smallhog.c */
#include <time.h>
#include <limits.h>
#include <signal.h>
#include <sys/time.h>
#define HIST 10

static volatile sig_atomic_t stop;

static void sighandler(int signr)
{
        (void) signr;
        stop = 1;
}

static unsigned long hog (unsigned long niters)
{
        stop = 0;
        while (!stop && --niters);
        return niters;
}

int main (void)
{
        int i;
        struct itimerval it = {
                .it_interval = { .tv_sec = 0, .tv_usec = 1 },
                .it_value    = { .tv_sec = 0, .tv_usec = 1 } };
        sigset_t set;
        unsigned long v[HIST];
        double tmp = 0.0;
        unsigned long n;
        signal(SIGALRM, &sighandler);
        setitimer(ITIMER_REAL, &it, NULL);

        hog (ULONG_MAX);
        for (i = 0; i < HIST; ++i) v[i] = ULONG_MAX - hog(ULONG_MAX);
        for (i = 0; i < HIST; ++i) tmp += v[i];
        tmp /= HIST;
        n = tmp - (tmp / 3.0);

        sigemptyset(&set);
        sigaddset(&set, SIGALRM);

        for (;;) {
                hog(n);
                sigwait(&set, &i);
        }
        return 0;
}
```

36.2 References

- https://lore.kernel.org/r/loom.20070212T063225-663@post.gmane.org
- Documentation/filesystems/proc.rst (1.8)

36.3 Thanks

Con Kolivas, Pavel Machek

CHAPTER

THIRTYSEVEN

HOW CPU TOPOLOGY INFO IS EXPORTED VIA SYSFS

CPU topology info is exported via sysfs. Items (attributes) are similar to /proc/cpuinfo output of some architectures. They reside in /sys/devices/system/cpu/cpuX/topology/. Please refer to the ABI file: Documentation/ABI/stable/sysfs-devices-system-cpu.

Architecture-neutral, drivers/base/topology.c, exports these attributes. However the die, cluster, book, and drawer hierarchy related sysfs files will only be created if an architecture provides the related macros as described below.

For an architecture to support this feature, it must define some of these macros in include/asm-XXX/topology.h:

```
#define topology_physical_package_id(cpu)
#define topology_die_id(cpu)
#define topology_cluster_id(cpu)
#define topology_core_id(cpu)
#define topology_book_id(cpu)
#define topology_drawer_id(cpu)
#define topology_sibling_cpumask(cpu)
#define topology_core_cpumask(cpu)
#define topology_cluster_cpumask(cpu)
#define topology_die_cpumask(cpu)
#define topology_book_cpumask(cpu)
#define topology_drawer_cpumask(cpu)
```

The type of `**_id macros` is int. The type of `**_cpumask macros` is `(const) struct cpumask *`. The latter correspond with appropriate `**_siblings` sysfs attributes (except for topology_sibling_cpumask() which corresponds with thread_siblings).

To be consistent on all architectures, include/linux/topology.h provides default definitions for any of the above macros that are not defined by include/asm-XXX/topology.h:

1) topology_physical_package_id: -1
2) topology_die_id: -1
3) topology_cluster_id: -1
4) topology_core_id: 0
5) topology_book_id: -1
6) topology_drawer_id: -1
7) topology_sibling_cpumask: just the given CPU

8) topology_core_cpumask: just the given CPU
9) topology_cluster_cpumask: just the given CPU
10) topology_die_cpumask: just the given CPU
11) topology_book_cpumask: just the given CPU
12) topology_drawer_cpumask: just the given CPU

Additionally, CPU topology information is provided under /sys/devices/system/cpu and includes these files. The internal source for the output is in brackets ("[]").

kernel_max:	the maximum CPU index allowed by the kernel configuration. [NR_CPUS-1]
offline:	CPUs that are not online because they have been HOTPLUGGED off or exceed the limit of CPUs allowed by the kernel configuration (kernel_max above). [~cpu_online_mask + cpus >= NR_CPUS]
online:	CPUs that are online and being scheduled [cpu_online_mask]
possible:	CPUs that have been allocated resources and can be brought online if they are present. [cpu_possible_mask]
present:	CPUs that have been identified as being present in the system. [cpu_present_mask]

The format for the above output is compatible with cpulist_parse() [see <linux/cpumask.h>]. Some examples follow.

In this example, there are 64 CPUs in the system but cpus 32-63 exceed the kernel max which is limited to 0..31 by the NR_CPUS config option being 32. Note also that CPUs 2 and 4-31 are not online but could be brought online as they are both present and possible:

```
kernel_max: 31
   offline: 2,4-31,32-63
    online: 0-1,3
  possible: 0-31
   present: 0-31
```

In this example, the NR_CPUS config option is 128, but the kernel was started with possible_cpus=144. There are 4 CPUs in the system and cpu2 was manually taken offline (and is the only CPU that can be brought online.):

```
kernel_max: 127
   offline: 2,4-127,128-143
    online: 0-1,3
  possible: 0-127
   present: 0-3
```

See Documentation/core-api/cpu_hotplug.rst for the possible_cpus=NUM kernel start parameter as well as more information on the various cpumasks.

CHAPTER

THIRTYEIGHT

DELL REMOTE BIOS UPDATE DRIVER (DELL_RBU)

38.1 Purpose

Document demonstrating the use of the Dell Remote BIOS Update driver for updating BIOS images on Dell servers and desktops.

38.2 Scope

This document discusses the functionality of the rbu driver only. It does not cover the support needed from applications to enable the BIOS to update itself with the image downloaded in to the memory.

38.3 Overview

This driver works with Dell OpenManage or Dell Update Packages for updating the BIOS on Dell servers (starting from servers sold since 1999), desktops and notebooks (starting from those sold in 2005).

Please go to http://support.dell.com register and you can find info on OpenManage and Dell Update packages (DUP).

Libsmbios can also be used to update BIOS on Dell systems go to https://linux.dell.com/libsmbios/ for details.

Dell_RBU driver supports BIOS update using the monolithic image and packetized image methods. In case of monolithic the driver allocates a contiguous chunk of physical pages having the BIOS image. In case of packetized the app using the driver breaks the image in to packets of fixed sizes and the driver would place each packet in contiguous physical memory. The driver also maintains a link list of packets for reading them back.

If the dell_rbu driver is unloaded all the allocated memory is freed.

The rbu driver needs to have an application (as mentioned above) which will inform the BIOS to enable the update in the next system reboot.

The user should not unload the rbu driver after downloading the BIOS image or updating.

The driver load creates the following directories under the /sys file system:

```
/sys/class/firmware/dell_rbu/loading
/sys/class/firmware/dell_rbu/data
/sys/devices/platform/dell_rbu/image_type
/sys/devices/platform/dell_rbu/data
/sys/devices/platform/dell_rbu/packet_size
```

The driver supports two types of update mechanism; monolithic and packetized. These update mechanism depends upon the BIOS currently running on the system. Most of the Dell systems support a monolithic update where the BIOS image is copied to a single contiguous block of physical memory.

In case of packet mechanism the single memory can be broken in smaller chunks of contiguous memory and the BIOS image is scattered in these packets.

By default the driver uses monolithic memory for the update type. This can be changed to packets during the driver load time by specifying the load parameter image_type=packet. This can also be changed later as below:

```
echo packet > /sys/devices/platform/dell_rbu/image_type
```

In packet update mode the packet size has to be given before any packets can be downloaded. It is done as below:

```
echo XXXX > /sys/devices/platform/dell_rbu/packet_size
```

In the packet update mechanism, the user needs to create a new file having packets of data arranged back to back. It can be done as follows: The user creates packets header, gets the chunk of the BIOS image and places it next to the packetheader; now, the packetheader + BIOS image chunk added together should match the specified packet_size. This makes one packet, the user needs to create more such packets out of the entire BIOS image file and then arrange all these packets back to back in to one single file.

This file is then copied to /sys/class/firmware/dell_rbu/data. Once this file gets to the driver, the driver extracts packet_size data from the file and spreads it across the physical memory in contiguous packet_sized space.

This method makes sure that all the packets get to the driver in a single operation.

In monolithic update the user simply get the BIOS image (.hdr file) and copies to the data file as is without any change to the BIOS image itself.

Do the steps below to download the BIOS image.

1) echo 1 > /sys/class/firmware/dell_rbu/loading
2) cp bios_image.hdr /sys/class/firmware/dell_rbu/data
3) echo 0 > /sys/class/firmware/dell_rbu/loading

The /sys/class/firmware/dell_rbu/ entries will remain till the following is done.

```
echo -1 > /sys/class/firmware/dell_rbu/loading
```

Until this step is completed the driver cannot be unloaded.

Also echoing either mono, packet or init in to image_type will free up the memory allocated by the driver.

If a user by accident executes steps 1 and 3 above without executing step 2; it will make the /sys/class/firmware/dell_rbu/ entries disappear.

The entries can be recreated by doing the following:

```
echo init > /sys/devices/platform/dell_rbu/image_type
```

Note: echoing init in image_type does not change its original value.

Also the driver provides /sys/devices/platform/dell_rbu/data readonly file to read back the image downloaded.

Note: After updating the BIOS image a user mode application needs to execute code which sends the BIOS update request to the BIOS. So on the next reboot the BIOS knows about the new image downloaded and it updates itself. Also don't unload the rbu driver if the image has to be updated.

CHAPTER

THIRTYNINE

DEVICE MAPPER

39.1 Guidance for writing policies

Try to keep transactionality out of it. The core is careful to avoid asking about anything that is migrating. This is a pain, but makes it easier to write the policies.

Mappings are loaded into the policy at construction time.

Every bio that is mapped by the target is referred to the policy. The policy can return a simple HIT or MISS or issue a migration.

Currently there's no way for the policy to issue background work, e.g. to start writing back dirty blocks that are going to be evicted soon.

Because we map bios, rather than requests it's easy for the policy to get fooled by many small bios. For this reason the core target issues periodic ticks to the policy. It's suggested that the policy doesn't update states (eg, hit counts) for a block more than once for each tick. The core ticks by watching bios complete, and so trying to see when the io scheduler has let the ios run.

39.1.1 Overview of supplied cache replacement policies

multiqueue (mq)

This policy is now an alias for smq (see below).

The following tunables are accepted, but have no effect:

```
'sequential_threshold <#nr_sequential_ios>'
'random_threshold <#nr_random_ios>'
'read_promote_adjustment <value>'
'write_promote_adjustment <value>'
'discard_promote_adjustment <value>'
```

Stochastic multiqueue (smq)

This policy is the default.

The stochastic multi-queue (smq) policy addresses some of the problems with the multiqueue (mq) policy.

The smq policy (vs mq) offers the promise of less memory utilization, improved performance and increased adaptability in the face of changing workloads. smq also does not have any cumbersome tuning knobs.

Users may switch from ”mq” to ”smq” simply by appropriately reloading a DM table that is using the cache target. Doing so will cause all of the mq policy's hints to be dropped. Also, performance of the cache may degrade slightly until smq recalculates the origin device's hotspots that should be cached.

Memory usage

The mq policy used a lot of memory; 88 bytes per cache block on a 64 bit machine.

smq uses 28bit indexes to implement its data structures rather than pointers. It avoids storing an explicit hit count for each block. It has a 'hotspot' queue, rather than a pre-cache, which uses a quarter of the entries (each hotspot block covers a larger area than a single cache block).

All this means smq uses ~25bytes per cache block. Still a lot of memory, but a substantial improvement nonetheless.

Level balancing

mq placed entries in different levels of the multiqueue structures based on their hit count (~ln(hit count)). This meant the bottom levels generally had the most entries, and the top ones had very few. Having unbalanced levels like this reduced the efficacy of the multiqueue.

smq does not maintain a hit count, instead it swaps hit entries with the least recently used entry from the level above. The overall ordering being a side effect of this stochastic process. With this scheme we can decide how many entries occupy each multiqueue level, resulting in better promotion/demotion decisions.

Adaptability: The mq policy maintained a hit count for each cache block. For a different block to get promoted to the cache its hit count has to exceed the lowest currently in the cache. This meant it could take a long time for the cache to adapt between varying IO patterns.

smq doesn't maintain hit counts, so a lot of this problem just goes away. In addition it tracks performance of the hotspot queue, which is used to decide which blocks to promote. If the hotspot queue is performing badly then it starts moving entries more quickly between levels. This lets it adapt to new IO patterns very quickly.

Performance

Testing smq shows substantially better performance than mq.

cleaner

The cleaner writes back all dirty blocks in a cache to decommission it.

39.1.2 Examples

The syntax for a table is:

```
cache <metadata dev> <cache dev> <origin dev> <block size>
<#feature_args> [<feature arg>]*
<policy> <#policy_args> [<policy arg>]*
```

The syntax to send a message using the dmsetup command is:

```
dmsetup message <mapped device> 0 sequential_threshold 1024
dmsetup message <mapped device> 0 random_threshold 8
```

Using dmsetup:

```
dmsetup create blah --table "0 268435456 cache /dev/sdb /dev/sdc \
    /dev/sdd 512 0 mq 4 sequential_threshold 1024 random_threshold 8"
creates a 128GB large mapped device named 'blah' with the
sequential threshold set to 1024 and the random_threshold set to 8.
```

39.2 Cache

39.2.1 Introduction

dm-cache is a device mapper target written by Joe Thornber, Heinz Mauelshagen, and Mike Snitzer.

It aims to improve performance of a block device (eg, a spindle) by dynamically migrating some of its data to a faster, smaller device (eg, an SSD).

This device-mapper solution allows us to insert this caching at different levels of the dm stack, for instance above the data device for a thin-provisioning pool. Caching solutions that are integrated more closely with the virtual memory system should give better performance.

The target reuses the metadata library used in the thin-provisioning library.

The decision as to what data to migrate and when is left to a plug-in policy module. Several of these have been written as we experiment, and we hope other people will contribute others for specific io scenarios (eg. a vm image server).

39.2.2 Glossary

Migration
Movement of the primary copy of a logical block from one device to the other.

Promotion
Migration from slow device to fast device.

Demotion
Migration from fast device to slow device.

The origin device always contains a copy of the logical block, which may be out of date or kept in sync with the copy on the cache device (depending on policy).

39.2.3 Design

Sub-devices

The target is constructed by passing three devices to it (along with other parameters detailed later):

1. An origin device - the big, slow one.
2. A cache device - the small, fast one.
3. A small metadata device - records which blocks are in the cache, which are dirty, and extra hints for use by the policy object. This information could be put on the cache device, but having it separate allows the volume manager to configure it differently, e.g. as a mirror for extra robustness. This metadata device may only be used by a single cache device.

Fixed block size

The origin is divided up into blocks of a fixed size. This block size is configurable when you first create the cache. Typically we've been using block sizes of 256KB - 1024KB. The block size must be between 64 sectors (32KB) and 2097152 sectors (1GB) and a multiple of 64 sectors (32KB).

Having a fixed block size simplifies the target a lot. But it is something of a compromise. For instance, a small part of a block may be getting hit a lot, yet the whole block will be promoted to the cache. So large block sizes are bad because they waste cache space. And small block sizes are bad because they increase the amount of metadata (both in core and on disk).

Cache operating modes

The cache has three operating modes: writeback, writethrough and passthrough.

If writeback, the default, is selected then a write to a block that is cached will go only to the cache and the block will be marked dirty in the metadata.

If writethrough is selected then a write to a cached block will not complete until it has hit both the origin and cache devices. Clean blocks should remain clean.

If passthrough is selected, useful when the cache contents are not known to be coherent with the origin device, then all reads are served from the origin device (all reads miss the cache) and all

writes are forwarded to the origin device; additionally, write hits cause cache block invalidates. To enable passthrough mode the cache must be clean. Passthrough mode allows a cache device to be activated without having to worry about coherency. Coherency that exists is maintained, although the cache will gradually cool as writes take place. If the coherency of the cache can later be verified, or established through use of the ”invalidate_cblocks” message, the cache device can be transitioned to writethrough or writeback mode while still warm. Otherwise, the cache contents can be discarded prior to transitioning to the desired operating mode.

A simple cleaner policy is provided, which will clean (write back) all dirty blocks in a cache. Useful for decommissioning a cache or when shrinking a cache. Shrinking the cache's fast device requires all cache blocks, in the area of the cache being removed, to be clean. If the area being removed from the cache still contains dirty blocks the resize will fail. Care must be taken to never reduce the volume used for the cache's fast device until the cache is clean. This is of particular importance if writeback mode is used. Writethrough and passthrough modes already maintain a clean cache. Future support to partially clean the cache, above a specified threshold, will allow for keeping the cache warm and in writeback mode during resize.

Migration throttling

Migrating data between the origin and cache device uses bandwidth. The user can set a throttle to prevent more than a certain amount of migration occurring at any one time. Currently we're not taking any account of normal io traffic going to the devices. More work needs doing here to avoid migrating during those peak io moments.

For the time being, a message ”migration_threshold <#sectors>” can be used to set the maximum number of sectors being migrated, the default being 2048 sectors (1MB).

Updating on-disk metadata

On-disk metadata is committed every time a FLUSH or FUA bio is written. If no such requests are made then commits will occur every second. This means the cache behaves like a physical disk that has a volatile write cache. If power is lost you may lose some recent writes. The metadata should always be consistent in spite of any crash.

The 'dirty' state for a cache block changes far too frequently for us to keep updating it on the fly. So we treat it as a hint. In normal operation it will be written when the dm device is suspended. If the system crashes all cache blocks will be assumed dirty when restarted.

Per-block policy hints

Policy plug-ins can store a chunk of data per cache block. It's up to the policy how big this chunk is, but it should be kept small. Like the dirty flags this data is lost if there's a crash so a safe fallback value should always be possible.

Policy hints affect performance, not correctness.

Policy messaging

Policies will have different tunables, specific to each one, so we need a generic way of getting and setting these. Device-mapper messages are used. Refer to cache-policies.txt.

Discard bitset resolution

We can avoid copying data during migration if we know the block has been discarded. A prime example of this is when mkfs discards the whole block device. We store a bitset tracking the discard state of blocks. However, we allow this bitset to have a different block size from the cache blocks. This is because we need to track the discard state for all of the origin device (compare with the dirty bitset which is just for the smaller cache device).

39.2.4 Target interface

Constructor

```
cache <metadata dev> <cache dev> <origin dev> <block size>
      <#feature args> [<feature arg>]*
      <policy> <#policy args> [policy args]*
```

metadata dev	fast device holding the persistent metadata
cache dev	fast device holding cached data blocks
origin dev	slow device holding original data blocks
block size	cache unit size in sectors
#feature args	number of feature arguments passed
feature args	writethrough or passthrough (The default is writeback.)
policy	the replacement policy to use
#policy args	an even number of arguments corresponding to key/value pairs passed to the policy
policy args	key/value pairs passed to the policy E.g. 'sequential_threshold 1024' See cache-policies.txt for details.

Optional feature arguments are:

writethrough	write through caching that prohibits cache block content from being different from origin block content. Without this argument, the default behaviour is to write back cache block contents later for performance reasons, so they may differ from the corresponding origin blocks.
passthrough	a degraded mode useful for various cache coherency situations (e.g., rolling back snapshots of underlying storage). Reads and writes always go to the origin. If a write goes to a cached origin block, then the cache block is invalidated. To enable passthrough mode the cache must be clean.
metadata2	use version 2 of the metadata. This stores the dirty bits in a separate btree, which improves speed of shutting down the cache.
no_discard_passdown	disable passing down discards from the cache to the origin's data device.

A policy called 'default' is always registered. This is an alias for the policy we currently think is giving best all round performance.

As the default policy could vary between kernels, if you are relying on the characteristics of a specific policy, always request it by name.

Status

```
<metadata block size> <#used metadata blocks>/<#total metadata blocks>
<cache block size> <#used cache blocks>/<#total cache blocks>
<#read hits> <#read misses> <#write hits> <#write misses>
<#demotions> <#promotions> <#dirty> <#features> <features>*
<#core args> <core args>* <policy name> <#policy args> <policy args>*
<cache metadata mode>
```

metadata block size	Fixed block size for each metadata block in sectors
#used metadata blocks	Number of metadata blocks used
#total metadata blocks	Total number of metadata blocks
cache block size	Configurable block size for the cache device in sectors
#used cache blocks	Number of blocks resident in the cache
#total cache blocks	Total number of cache blocks
#read hits	Number of times a READ bio has been mapped to the cache
#read misses	Number of times a READ bio has been mapped to the origin
#write hits	Number of times a WRITE bio has been mapped to the cache
#write misses	Number of times a WRITE bio has been mapped to the origin
#demotions	Number of times a block has been removed from the cache
#promotions	Number of times a block has been moved to the cache
#dirty	Number of blocks in the cache that differ from the origin
#feature args	Number of feature args to follow
feature args	'writethrough' (optional)
#core args	Number of core arguments (must be even)
core args	Key/value pairs for tuning the core e.g. migration_threshold
policy name	Name of the policy
#policy args	Number of policy arguments to follow (must be even)
policy args	Key/value pairs e.g. sequential_threshold
cache metadata mode	ro if read-only, rw if read-write In serious cases where even a read-only mode is deemed unsafe no further I/O will be permitted and the status will just contain the string 'Fail'. The userspace recovery tools should then be used.
needs_check	'needs_check' if set, '-' if not set A metadata operation has failed, resulting in the needs_check flag being set in the metadata's superblock. The metadata device must be deactivated and checked/repaired before the cache can be made fully operational again. '-' indicates needs_check is not set.

Messages

Policies will have different tunables, specific to each one, so we need a generic way of getting and setting these. Device-mapper messages are used. (A sysfs interface would also be possible.)

The message format is:

```
<key> <value>
```

E.g.:

```
dmsetup message my_cache 0 sequential_threshold 1024
```

Invalidation is removing an entry from the cache without writing it back. Cache blocks can be invalidated via the invalidate_cblocks message, which takes an arbitrary number of cblock ranges. Each cblock range's end value is "one past the end", meaning 5-10 expresses a range of values from 5 to 9. Each cblock must be expressed as a decimal value, in the future a variant message that takes cblock ranges expressed in hexadecimal may be needed to better support efficient invalidation of larger caches. The cache must be in passthrough mode when invalidate_cblocks is used:

```
invalidate_cblocks [<cblock>|<cblock begin>-<cblock end>]*
```

E.g.:

```
dmsetup message my_cache 0 invalidate_cblocks 2345 3456-4567 5678-6789
```

39.2.5 Examples

The test suite can be found here:

https://github.com/jthornber/device-mapper-test-suite

```
dmsetup create my_cache --table '0 41943040 cache /dev/mapper/metadata \
        /dev/mapper/ssd /dev/mapper/origin 512 1 writeback default 0'
dmsetup create my_cache --table '0 41943040 cache /dev/mapper/metadata \
        /dev/mapper/ssd /dev/mapper/origin 1024 1 writeback \
        mq 4 sequential_threshold 1024 random_threshold 8'
```

39.3 dm-delay

Device-Mapper's ”delay” target delays reads and/or writes and maps them to different devices.

Parameters:

```
<device> <offset> <delay> [<write_device> <write_offset> <write_delay>
                           [<flush_device> <flush_offset> <flush_delay>]]
```

With separate write parameters, the first set is only used for reads. Offsets are specified in sectors. Delays are specified in milliseconds.

39.3.1 Example scripts

```
#!/bin/sh
# Create device delaying rw operation for 500ms
echo "0 `blockdev --getsz $1` delay $1 0 500" | dmsetup create delayed
```

```
#!/bin/sh
# Create device delaying only write operation for 500ms and
# splitting reads and writes to different devices $1 $2
echo "0 `blockdev --getsz $1` delay $1 0 0 $2 0 500" | dmsetup create delayed
```

39.4 dm-clone

39.4.1 Introduction

dm-clone is a device mapper target which produces a one-to-one copy of an existing, read-only source device into a writable destination device: It presents a virtual block device which makes all data appear immediately, and redirects reads and writes accordingly.

The main use case of dm-clone is to clone a potentially remote, high-latency, read-only, archival-type block device into a writable, fast, primary-type device for fast, low-latency I/O. The cloned device is visible/mountable immediately and the copy of the source device to the destination device happens in the background, in parallel with user I/O.

For example, one could restore an application backup from a read-only copy, accessible through a network storage protocol (NBD, Fibre Channel, iSCSI, AoE, etc.), into a local SSD or NVMe device, and start using the device immediately, without waiting for the restore to complete.

When the cloning completes, the dm-clone table can be removed altogether and be replaced, e.g., by a linear table, mapping directly to the destination device.

The dm-clone target reuses the metadata library used by the thin-provisioning target.

39.4.2 Glossary

Hydration
: The process of filling a region of the destination device with data from the same region of the source device, i.e., copying the region from the source to the destination device.

Once a region gets hydrated we redirect all I/O regarding it to the destination device.

39.4.3 Design

Sub-devices

The target is constructed by passing three devices to it (along with other parameters detailed later):

1. A source device - the read-only device that gets cloned and source of the hydration.
2. A destination device - the destination of the hydration, which will become a clone of the source device.
3. A small metadata device - it records which regions are already valid in the destination device, i.e., which regions have already been hydrated, or have been written to directly, via user I/O.

The size of the destination device must be at least equal to the size of the source device.

Regions

dm-clone divides the source and destination devices in fixed sized regions. Regions are the unit of hydration, i.e., the minimum amount of data copied from the source to the destination device.

The region size is configurable when you first create the dm-clone device. The recommended region size is the same as the file system block size, which usually is 4KB. The region size must be between 8 sectors (4KB) and 2097152 sectors (1GB) and a power of two.

Reads and writes from/to hydrated regions are serviced from the destination device.

A read to a not yet hydrated region is serviced directly from the source device.

A write to a not yet hydrated region will be delayed until the corresponding region has been hydrated and the hydration of the region starts immediately.

Note that a write request with size equal to region size will skip copying of the corresponding region from the source device and overwrite the region of the destination device directly.

Discards

dm-clone interprets a discard request to a range that hasn't been hydrated yet as a hint to skip hydration of the regions covered by the request, i.e., it skips copying the region's data from the source to the destination device, and only updates its metadata.

If the destination device supports discards, then by default dm-clone will pass down discard requests to it.

Background Hydration

dm-clone copies continuously from the source to the destination device, until all of the device has been copied.

Copying data from the source to the destination device uses bandwidth. The user can set a throttle to prevent more than a certain amount of copying occurring at any one time. Moreover, dm-clone takes into account user I/O traffic going to the devices and pauses the background hydration when there is I/O in-flight.

A message *hydration_threshold <#regions>* can be used to set the maximum number of regions being copied, the default being 1 region.

dm-clone employs dm-kcopyd for copying portions of the source device to the destination device. By default, we issue copy requests of size equal to the region size. A message *hydration_batch_size <#regions>* can be used to tune the size of these copy requests. Increasing the hydration batch size results in dm-clone trying to batch together contiguous regions, so we copy the data in batches of this many regions.

When the hydration of the destination device finishes, a dm event will be sent to user space.

Updating on-disk metadata

On-disk metadata is committed every time a FLUSH or FUA bio is written. If no such requests are made then commits will occur every second. This means the dm-clone device behaves like a physical disk that has a volatile write cache. If power is lost you may lose some recent writes. The metadata should always be consistent in spite of any crash.

39.4.4 Target Interface

Constructor

```
clone <metadata dev> <destination dev> <source dev> <region␣
↪size>
     [<#feature args> [<feature arg>]* [<#core args> [<core␣
↪arg>]*]]
```

metadata dev	Fast device holding the persistent metadata
destination dev	The destination device, where the source will be cloned
source dev	Read only device containing the data that gets cloned
region size	The size of a region in sectors
#feature args	Number of feature arguments passed
feature args	no_hydration or no_discard_passdown
#core args	An even number of arguments corresponding to key/value pairs passed to dm-clone
core args	Key/value pairs passed to dm-clone, e.g. *hydration_threshold 256*

Optional feature arguments are:

no_hydration	Create a dm-clone instance with background hydration disabled
no_discard_passdown	Disable passing down discards to the destination device

Optional core arguments are:

hydration_threshold <#regions>	Maximum number of regions being copied from the source to the destination device at any one time, during background hydration.
hydration_batch_size <#regions>	During background hydration, try to batch together contiguous regions, so we copy data from the source to the destination device in batches of this many regions.

Status

```
<metadata block size> <#used metadata blocks>/<#total metadata␣
↪blocks>
<region size> <#hydrated regions>/<#total regions> <#hydrating␣
↪regions>
<#feature args> <feature args>* <#core args> <core args>*
<clone metadata mode>
```

metadata block size	Fixed block size for each metadata block in sectors
#used metadata blocks	Number of metadata blocks used
#total metadata blocks	Total number of metadata blocks
region size	Configurable region size for the device in sectors
#hydrated regions	Number of regions that have finished hydrating
#total regions	Total number of regions to hydrate
#hydrating regions	Number of regions currently hydrating
#feature args	Number of feature arguments to follow
feature args	Feature arguments, e.g. *no_hydration*
#core args	Even number of core arguments to follow
core args	Key/value pairs for tuning the core, e.g. *hydration_threshold 256*
clone metadata mode	ro if read-only, rw if read-write In serious cases where even a read-only mode is deemed unsafe no further I/O will be permitted and the status will just contain the string 'Fail'. If the metadata mode changes, a dm event will be sent to user space.

Messages

disable_hydration
Disable the background hydration of the destination device.

enable_hydration
Enable the background hydration of the destination device.

hydration_threshold <#regions>
Set background hydration threshold.

hydration_batch_size <#regions>
Set background hydration batch size.

39.4.5 Examples

Clone a device containing a file system

1. Create the dm-clone device.

```
dmsetup create clone --table "0 1048576000 clone $metadata_dev $dest_dev \
  $source_dev 8 1 no_hydration"
```

2. Mount the device and trim the file system. dm-clone interprets the discards sent by the file system and it will not hydrate the unused space.

```
mount /dev/mapper/clone /mnt/cloned-fs
fstrim /mnt/cloned-fs
```

3. Enable background hydration of the destination device.

```
dmsetup message clone 0 enable_hydration
```

4. When the hydration finishes, we can replace the dm-clone table with a linear table.

```
dmsetup suspend clone
dmsetup load clone --table "0 1048576000 linear $dest_dev 0"
dmsetup resume clone
```

   The metadata device is no longer needed and can be safely discarded or reused for other purposes.

39.4.6 Known issues

1. We redirect reads, to not-yet-hydrated regions, to the source device. If reading the source device has high latency and the user repeatedly reads from the same regions, this behaviour could degrade performance. We should use these reads as hints to hydrate the relevant regions sooner. Currently, we rely on the page cache to cache these regions, so we hopefully don't end up reading them multiple times from the source device.
2. Release in-core resources, i.e., the bitmaps tracking which regions are hydrated, after the hydration has finished.
3. During background hydration, if we fail to read the source or write to the destination device, we print an error message, but the hydration process continues indefinitely, until it succeeds. We should stop the background hydration after a number of failures and emit a dm event for user space to notice.

39.4.7 Why not...?

We explored the following alternatives before implementing dm-clone:

1. Use dm-cache with cache size equal to the source device and implement a new cloning policy:
   - The resulting cache device is not a one-to-one mirror of the source device and thus we cannot remove the cache device once cloning completes.
   - dm-cache writes to the source device, which violates our requirement that the source device must be treated as read-only.
   - Caching is semantically different from cloning.
2. Use dm-snapshot with a COW device equal to the source device:
   - dm-snapshot stores its metadata in the COW device, so the resulting device is not a one-to-one mirror of the source device.
   - No background copying mechanism.
   - dm-snapshot needs to commit its metadata whenever a pending exception completes, to ensure snapshot consistency. In the case of cloning, we don't need to be so strict and can rely on committing metadata every time a FLUSH or FUA bio is written, or periodically, like dm-thin and dm-cache do. This improves the performance significantly.
3. Use dm-mirror: The mirror target has a background copying/mirroring mechanism, but it writes to all mirrors, thus violating our requirement that the source device must be treated as read-only.
4. Use dm-thin's external snapshot functionality. This approach is the most promising among all alternatives, as the thinly-provisioned volume is a one-to-one mirror of the source device and handles reads and writes to un-provisioned/not-yet-cloned areas the same way as dm-clone does.

   Still:

   - There is no background copying mechanism, though one could be implemented.
   - Most importantly, we want to support arbitrary block devices as the destination of the cloning process and not restrict ourselves to thinly-provisioned volumes. Thin-provisioning has an inherent metadata overhead, for maintaining the thin volume mappings, which significantly degrades performance.

   Moreover, cloning a device shouldn't force the use of thin-provisioning. On the other hand, if we wish to use thin provisioning, we can just use a thin LV as dm-clone's destination device.

39.5 dm-crypt

Device-Mapper's ”crypt” target provides transparent encryption of block devices using the kernel crypto API.

For a more detailed description of supported parameters see: https://gitlab.com/cryptsetup/cryptsetup/wikis/DMCrypt

Parameters:

```
<cipher> <key> <iv_offset> <device path> \
<offset> [<#opt_params> <opt_params>]
```

<cipher>

Encryption cipher, encryption mode and Initial Vector (IV) generator.

The cipher specifications format is:

```
cipher[:keycount]-chainmode-ivmode[:ivopts]
```

Examples:

```
aes-cbc-essiv:sha256
aes-xts-plain64
serpent-xts-plain64
```

Cipher format also supports direct specification with kernel crypt API format (selected by capi: prefix). The IV specification is the same as for the first format type. This format is mainly used for specification of authenticated modes.

The crypto API cipher specifications format is:

```
capi:cipher_api_spec-ivmode[:ivopts]
```

Examples:

```
capi:cbc(aes)-essiv:sha256
capi:xts(aes)-plain64
```

Examples of authenticated modes:

```
capi:gcm(aes)-random
capi:authenc(hmac(sha256),xts(aes))-random
capi:rfc7539(chacha20,poly1305)-random
```

The /proc/crypto contains a list of currently loaded crypto modes.

<key>

Key used for encryption. It is encoded either as a hexadecimal number or it can be passed as <key_string> prefixed with single colon character (':') for keys residing in kernel keyring service. You can only use key sizes that are valid for the selected cipher in combination with the selected iv mode. Note that for some iv modes the key string can contain additional keys (for example IV seed) so the key contains more parts concatenated into a single string.

<key_string>
The kernel keyring key is identified by string in following format: <key_size>:<key_type>:<key_description>.

<key_size>
The encryption key size in bytes. The kernel key payload size must match the value passed in <key_size>.

<key_type>
Either 'logon', 'user', 'encrypted' or 'trusted' kernel key type.

<key_description>
The kernel keyring key description crypt target should look for when loading key of <key_type>.

<keycount>
Multi-key compatibility mode. You can define <keycount> keys and then sectors are encrypted according to their offsets (sector 0 uses key0; sector 1 uses key1 etc.). <keycount> must be a power of two.

<iv_offset>
The IV offset is a sector count that is added to the sector number before creating the IV.

<device path>
This is the device that is going to be used as backend and contains the encrypted data. You can specify it as a path like /dev/xxx or a device number <major>:<minor>.

<offset>
Starting sector within the device where the encrypted data begins.

<#opt_params>
Number of optional parameters. If there are no optional parameters, the optional parameters section can be skipped or #opt_params can be zero. Otherwise #opt_params is the number of following arguments.

Example of optional parameters section:
3 allow_discards same_cpu_crypt submit_from_crypt_cpus

allow_discards
Block discard requests (a.k.a. TRIM) are passed through the crypt device. The default is to ignore discard requests.

WARNING: Assess the specific security risks carefully before enabling this option. For example, allowing discards on encrypted devices may lead to the leak of information about the ciphertext device (filesystem type, used space etc.) if the discarded blocks can be located easily on the device later.

same_cpu_crypt
Perform encryption using the same cpu that IO was submitted on. The default is to use an unbound workqueue so that encryption work is automatically balanced between available CPUs.

submit_from_crypt_cpus
Disable offloading writes to a separate thread after encryption. There are some situations where offloading write bios from the encryption threads to a single thread degrades performance significantly. The default is to offload write bios to the same thread because it benefits CFQ to have writes submitted using the same context.

no_read_workqueue
Bypass dm-crypt internal workqueue and process read requests synchronously.

no_write_workqueue
Bypass dm-crypt internal workqueue and process write requests synchronously. This option is automatically enabled for host-managed zoned block devices (e.g. host-managed SMR hard-disks).

integrity:<bytes>:<type>
The device requires additional <bytes> metadata per-sector stored in per-bio integrity structure. This metadata must by provided by underlying dm-integrity target.

The <type> can be "none" if metadata is used only for persistent IV.

For Authenticated Encryption with Additional Data (AEAD) the <type> is "aead". An AEAD mode additionally calculates and verifies integrity for the encrypted device. The additional space is then used for storing authentication tag (and persistent IV if needed).

sector_size:<bytes>
Use <bytes> as the encryption unit instead of 512 bytes sectors. This option can be in range 512 - 4096 bytes and must be power of two. Virtual device will announce this size as a minimal IO and logical sector.

iv_large_sectors
IV generators will use sector number counted in <sector_size> units instead of default 512 bytes sectors.

For example, if <sector_size> is 4096 bytes, plain64 IV for the second sector will be 8 (without flag) and 1 if iv_large_sectors is present. The <iv_offset> must be multiple of <sector_size> (in 512 bytes units) if this flag is specified.

39.5.1 Example scripts

LUKS (Linux Unified Key Setup) is now the preferred way to set up disk encryption with dm-crypt using the 'cryptsetup' utility, see https://gitlab.com/cryptsetup/cryptsetup

```
#!/bin/sh
# Create a crypt device using dmsetup
dmsetup create crypt1 --table "0 `blockdev --getsz $1` crypt aes-cbc-
↪essiv:sha256 babebabebabebabebabebabebabebabe 0 $1 0"
```

```
#!/bin/sh
# Create a crypt device using dmsetup when encryption key is stored in keyring␣
↪service
dmsetup create crypt2 --table "0 `blockdev --getsize $1` crypt aes-cbc-
↪essiv:sha256 :32:logon:my_prefix:my_key 0 $1 0"
```

```
#!/bin/sh
# Create a crypt device using cryptsetup and LUKS header with default cipher
cryptsetup luksFormat $1
cryptsetup luksOpen $1 crypt1
```

39.6 dm-dust

This target emulates the behavior of bad sectors at arbitrary locations, and the ability to enable the emulation of the failures at an arbitrary time.

This target behaves similarly to a linear target. At a given time, the user can send a message to the target to start failing read requests on specific blocks (to emulate the behavior of a hard disk drive with bad sectors).

When the failure behavior is enabled (i.e.: when the output of "dmsetup status" displays "fail_read_on_bad_block"), reads of blocks in the "bad block list" will fail with EIO ("Input/output error").

Writes of blocks in the "bad block list will result in the following:

1. Remove the block from the "bad block list".
2. Successfully complete the write.

This emulates the "remapped sector" behavior of a drive with bad sectors.

Normally, a drive that is encountering bad sectors will most likely encounter more bad sectors, at an unknown time or location. With dm-dust, the user can use the "addbadblock" and "removebadblock" messages to add arbitrary bad blocks at new locations, and the "enable" and "disable" messages to modulate the state of whether the configured "bad blocks" will be treated as bad, or bypassed. This allows the pre-writing of test data and metadata prior to simulating a "failure" event where bad sectors start to appear.

39.6.1 Table parameters

<device_path> <offset> <blksz>

Mandatory parameters:

<device_path>:
Path to the block device.

<offset>:
Offset to data area from start of device_path

<blksz>:
Block size in bytes

(minimum 512, maximum 1073741824, must be a power of 2)

39.6.2 Usage instructions

First, find the size (in 512-byte sectors) of the device to be used:

```
$ sudo blockdev --getsz /dev/vdb1
33552384
```

Create the dm-dust device: (For a device with a block size of 512 bytes)

```
$ sudo dmsetup create dust1 --table '0 33552384 dust /dev/vdb1 0 512'
```

(For a device with a block size of 4096 bytes)

```
$ sudo dmsetup create dust1 --table '0 33552384 dust /dev/vdb1 0 4096'
```

Check the status of the read behavior (”bypass” indicates that all I/O will be passed through to the underlying device; ”verbose” indicates that bad block additions, removals, and remaps will be verbosely logged):

```
$ sudo dmsetup status dust1
0 33552384 dust 252:17 bypass verbose

$ sudo dd if=/dev/mapper/dust1 of=/dev/null bs=512 count=128 iflag=direct
128+0 records in
128+0 records out

$ sudo dd if=/dev/zero of=/dev/mapper/dust1 bs=512 count=128 oflag=direct
128+0 records in
128+0 records out
```

39.6.3 Adding and removing bad blocks

At any time (i.e.: whether the device has the ”bad block” emulation enabled or disabled), bad blocks may be added or removed from the device via the ”addbadblock” and ”removebadblock” messages:

```
$ sudo dmsetup message dust1 0 addbadblock 60
kernel: device-mapper: dust: badblock added at block 60

$ sudo dmsetup message dust1 0 addbadblock 67
kernel: device-mapper: dust: badblock added at block 67

$ sudo dmsetup message dust1 0 addbadblock 72
kernel: device-mapper: dust: badblock added at block 72
```

These bad blocks will be stored in the ”bad block list”. While the device is in ”bypass” mode, reads and writes will succeed:

```
$ sudo dmsetup status dust1
0 33552384 dust 252:17 bypass
```

39.6.4 Enabling block read failures

To enable the ”fail read on bad block” behavior, send the ”enable” message:

```
$ sudo dmsetup message dust1 0 enable
kernel: device-mapper: dust: enabling read failures on bad sectors

$ sudo dmsetup status dust1
0 33552384 dust 252:17 fail_read_on_bad_block
```

With the device in ”fail read on bad block” mode, attempting to read a block will encounter an ”Input/output error”:

```
$ sudo dd if=/dev/mapper/dust1 of=/dev/null bs=512 count=1 skip=67 iflag=direct
dd: error reading '/dev/mapper/dust1': Input/output error
0+0 records in
0+0 records out
0 bytes copied, 0.00040651 s, 0.0 kB/s
```

...and writing to the bad blocks will remove the blocks from the list, therefore emulating the ”remap” behavior of hard disk drives:

```
$ sudo dd if=/dev/zero of=/dev/mapper/dust1 bs=512 count=128 oflag=direct
128+0 records in
128+0 records out

kernel: device-mapper: dust: block 60 removed from badblocklist by write
kernel: device-mapper: dust: block 67 removed from badblocklist by write
kernel: device-mapper: dust: block 72 removed from badblocklist by write
kernel: device-mapper: dust: block 87 removed from badblocklist by write
```

39.6.5 Bad block add/remove error handling

Attempting to add a bad block that already exists in the list will result in an ”Invalid argument” error, as well as a helpful message:

```
$ sudo dmsetup message dust1 0 addbadblock 88
device-mapper: message ioctl on dust1  failed: Invalid argument
kernel: device-mapper: dust: block 88 already in badblocklist
```

Attempting to remove a bad block that doesn't exist in the list will result in an ”Invalid argument” error, as well as a helpful message:

```
$ sudo dmsetup message dust1 0 removebadblock 87
device-mapper: message ioctl on dust1  failed: Invalid argument
kernel: device-mapper: dust: block 87 not found in badblocklist
```

39.6.6 Counting the number of bad blocks in the bad block list

To count the number of bad blocks configured in the device, run the following message command:

```
$ sudo dmsetup message dust1 0 countbadblocks
```

A message will print with the number of bad blocks currently configured on the device:

```
countbadblocks: 895 badblock(s) found
```

39.6.7 Querying for specific bad blocks

To find out if a specific block is in the bad block list, run the following message command:

```
$ sudo dmsetup message dust1 0 queryblock 72
```

The following message will print if the block is in the list:

```
dust_query_block: block 72 found in badblocklist
```

The following message will print if the block is not in the list:

```
dust_query_block: block 72 not found in badblocklist
```

The ”queryblock” message command will work in both the ”enabled” and ”disabled” modes, allowing the verification of whether a block will be treated as ”bad” without having to issue I/O to the device, or having to ”enable” the bad block emulation.

39.6.8 Clearing the bad block list

To clear the bad block list (without needing to individually run a ”removebadblock” message command for every block), run the following message command:

```
$ sudo dmsetup message dust1 0 clearbadblocks
```

After clearing the bad block list, the following message will appear:

```
dust_clear_badblocks: badblocks cleared
```

If there were no bad blocks to clear, the following message will appear:

```
dust_clear_badblocks: no badblocks found
```

39.6.9 Listing the bad block list

To list all bad blocks in the bad block list (using an example device with blocks 1 and 2 in the bad block list), run the following message command:

```
$ sudo dmsetup message dust1 0 listbadblocks
1
2
```

If there are no bad blocks in the bad block list, the command will execute with no output:

```
$ sudo dmsetup message dust1 0 listbadblocks
```

39.6.10 Message commands list

Below is a list of the messages that can be sent to a dust device:

Operations on blocks (requires a <blknum> argument):

```
addbadblock <blknum>
queryblock <blknum>
removebadblock <blknum>
```

...where <blknum> is a block number within range of the device (corresponding to the block size of the device.)

Single argument message commands:

```
countbadblocks
clearbadblocks
listbadblocks
disable
enable
quiet
```

39.6.11 Device removal

When finished, remove the device via the "dmsetup remove" command:

```
$ sudo dmsetup remove dust1
```

39.6.12 Quiet mode

On test runs with many bad blocks, it may be desirable to avoid excessive logging (from bad blocks added, removed, or "remapped"). This can be done by enabling "quiet mode" via the following message:

```
$ sudo dmsetup message dust1 0 quiet
```

This will suppress log messages from add / remove / removed by write operations. Log messages from "countbadblocks" or "queryblock" message commands will still print in quiet mode.

The status of quiet mode can be seen by running "dmsetup status":

```
$ sudo dmsetup status dust1
0 33552384 dust 252:17 fail_read_on_bad_block quiet
```

To disable quiet mode, send the "quiet" message again:

```
$ sudo dmsetup message dust1 0 quiet

$ sudo dmsetup status dust1
0 33552384 dust 252:17 fail_read_on_bad_block verbose
```

(The presence of "verbose" indicates normal logging.)

39.6.13 ”Why not...?”

scsi_debug has a ”medium error” mode that can fail reads on one specified sector (sector 0x1234, hardcoded in the source code), but it uses RAM for the persistent storage, which drastically decreases the potential device size.

dm-flakey fails all I/O from all block locations at a specified time frequency, and not a given point in time.

When a bad sector occurs on a hard disk drive, reads to that sector are failed by the device, usually resulting in an error code of EIO (”I/O error”) or ENODATA (”No data available”). However, a write to the sector may succeed, and result in the sector becoming readable after the device controller no longer experiences errors reading the sector (or after a reallocation of the sector). However, there may be bad sectors that occur on the device in the future, in a different, unpredictable location.

This target seeks to provide a device that can exhibit the behavior of a bad sector at a known sector location, at a known time, based on a large storage device (at least tens of gigabytes, not occupying system memory).

39.7 dm-ebs

This target is similar to the linear target except that it emulates a smaller logical block size on a device with a larger logical block size. Its main purpose is to provide emulation of 512 byte sectors on devices that do not provide this emulation (i.e. 4K native disks).

Supported emulated logical block sizes 512, 1024, 2048 and 4096.

Underlying block size can be set to > 4K to test buffering larger units.

39.7.1 Table parameters

<dev path> <offset> <emulated sectors> [<underlying sectors>]

Mandatory parameters:

<dev path>:
Full pathname to the underlying block-device, or a ”major:minor” device-number.

<offset>:
Starting sector within the device; has to be a multiple of <emulated sectors>.

<emulated sectors>:
Number of sectors defining the logical block size to be emulated; 1, 2, 4, 8 sectors of 512 bytes supported.

Optional parameter:

<underlying sectors>:
Number of sectors defining the logical block size of <dev path>. 2^N supported, e.g. 8 = emulate 8 sectors of 512 bytes = 4KiB. If not provided, the logical block size of <dev path> will be used.

Examples:

Emulate 1 sector = 512 bytes logical block size on /dev/sda starting at offset 1024 sectors with underlying devices block size automatically set:

ebs /dev/sda 1024 1

Emulate 2 sector = 1KiB logical block size on /dev/sda starting at offset 128 sectors, enforce 2KiB underlying device block size. This presumes 2KiB logical blocksize on /dev/sda or less to work:

ebs /dev/sda 128 2 4

39.8 dm-flakey

This target is the same as the linear target except that it exhibits unreliable behaviour periodically. It's been found useful in simulating failing devices for testing purposes.

Starting from the time the table is loaded, the device is available for <up interval> seconds, then exhibits unreliable behaviour for <down interval> seconds, and then this cycle repeats.

Also, consider using this in combination with the dm-delay target too, which can delay reads and writes and/or send them to different underlying devices.

39.8.1 Table parameters

```
<dev path> <offset> <up interval> <down interval> \
  [<num_features> [<feature arguments>]]
```

Mandatory parameters:

<dev path>:
Full pathname to the underlying block-device, or a ”major:minor” device-number.

<offset>:
Starting sector within the device.

<up interval>:
Number of seconds device is available.

<down interval>:
Number of seconds device returns errors.

Optional feature parameters:

If no feature parameters are present, during the periods of unreliability, all I/O returns errors.

error_reads:
All read I/O is failed with an error signalled. Write I/O is handled correctly.

drop_writes:
All write I/O is silently ignored. Read I/O is handled correctly.

error_writes:
All write I/O is failed with an error signalled. Read I/O is handled correctly.

corrupt_bio_byte <Nth_byte> <direction> <value> <flags>:

During <down interval>, replace <Nth_byte> of the data of each matching bio with <value>.

<Nth_byte>:
The offset of the byte to replace. Counting starts at 1, to replace the first byte.

<direction>:
Either 'r' to corrupt reads or 'w' to corrupt writes. 'w' is incompatible with drop_writes.

<value>:
The value (from 0-255) to write.

<flags>:
Perform the replacement only if bio->bi_opf has all the selected flags set.

random_read_corrupt <probability>
During <down interval>, replace random byte in a read bio with a random value. probability is an integer between 0 and 1000000000 meaning 0% to 100% probability of corruption.

random_write_corrupt <probability>
During <down interval>, replace random byte in a write bio with a random value. probability is an integer between 0 and 1000000000 meaning 0% to 100% probability of corruption.

Examples:

Replaces the 32nd byte of READ bios with the value 1:

```
corrupt_bio_byte 32 r 1 0
```

Replaces the 224th byte of REQ_META (=32) bios with the value 0:

```
corrupt_bio_byte 224 w 0 32
```

39.9 dm-ima

For a given system, various external services/infrastructure tools (including the attestation service) interact with it - both during the setup and during rest of the system run-time. They share sensitive data and/or execute critical workload on that system. The external services may want to verify the current run-time state of the relevant kernel subsystems before fully trusting the system with business-critical data/workload.

Device mapper plays a critical role on a given system by providing various important functionalities to the block devices using various target types like crypt, verity, integrity etc. Each of these target types' functionalities can be configured with various attributes. The attributes chosen to configure these target types can significantly impact the security profile of the block device, and in-turn, of the system itself. For instance, the type of encryption algorithm and the key size determines the strength of encryption for a given block device.

Therefore, verifying the current state of various block devices as well as their various target attributes is crucial for external services before fully trusting the system with business-critical data/workload.

IMA kernel subsystem provides the necessary functionality for device mapper to measure the state and configuration of various block devices -

- by device mapper itself, from within the kernel,
- in a tamper resistant way,
- and re-measured - triggered on state/configuration change.

39.9.1 Setting the IMA Policy:

For IMA to measure the data on a given system, the IMA policy on the system needs to be updated to have following line, and the system needs to be restarted for the measurements to take effect.

```
/etc/ima/ima-policy
   measure func=CRITICAL_DATA label=device-mapper template=ima-buf
```

The measurements will be reflected in the IMA logs, which are located at:

```
/sys/kernel/security/integrity/ima/ascii_runtime_measurements
/sys/kernel/security/integrity/ima/binary_runtime_measurements
```

Then IMA ASCII measurement log has the following format:

```
<PCR> <TEMPLATE_DATA_DIGEST> <TEMPLATE_NAME> <TEMPLATE_DATA>

PCR := Platform Configuration Register, in which the values are registered.
      This is applicable if TPM chip is in use.

TEMPLATE_DATA_DIGEST := Template data digest of the IMA record.
TEMPLATE_NAME := Template name that registered the integrity value (e.g. ima-
↪buf).

TEMPLATE_DATA := <ALG> ":" <EVENT_DIGEST> <EVENT_NAME> <EVENT_DATA>
                 It contains data for the specific event to be measured,
                 in a given template data format.

ALG := Algorithm to compute event digest
EVENT_DIGEST := Digest of the event data
EVENT_NAME := Description of the event (e.g. 'dm_table_load').
EVENT_DATA := The event data to be measured.
```

NOTE #1:
The DM target data measured by IMA subsystem can alternatively be queried from userspace by setting DM_IMA_MEASUREMENT_FLAG with DM_TABLE_STATUS_CMD.

NOTE #2:
The Kernel configuration CONFIG_IMA_DISABLE_HTABLE allows measurement of duplicate records.
To support recording duplicate IMA events in the IMA log, the Kernel needs to be configured with CONFIG_IMA_DISABLE_HTABLE=y.

39.9.2 Supported Device States:

Following device state changes will trigger IMA measurements:

1. Table load
2. Device resume
3. Device remove
4. Table clear
5. Device rename

1. Table load:

When a new table is loaded in a device's inactive table slot, the device information and target specific details from the targets in the table are measured.

The IMA measurement log has the following format for 'dm_table_load':

```
EVENT_NAME := "dm_table_load"
EVENT_DATA := <dm_version_str> ";" <device_metadata> ";" <table_load_data>

dm_version_str := "dm_version=" <N> "." <N> "." <N>
                 Same as Device Mapper driver version.
device_metadata := <device_name> "," <device_uuid> "," <device_major> ","
↪<device_minor> ","
                  <minor_count> "," <num_device_targets> ";"

device_name := "name=" <dm-device-name>
device_uuid := "uuid=" <dm-device-uuid>
device_major := "major=" <N>
device_minor := "minor=" <N>
minor_count := "minor_count=" <N>
num_device_targets := "num_targets=" <N>
dm-device-name := Name of the device. If it contains special characters like '\
↪', ',', ';',
                  they are prefixed with '\'.
dm-device-uuid := UUID of the device. If it contains special characters like '\
↪', ',', ';',
                  they are prefixed with '\'.
```

```
table_load_data := <target_data>
                   Represents the data (as name=value pairs) from various␣
↪targets in the table,
                   which is being loaded into the DM device's inactive table␣
↪slot.
target_data := <target_data_row> | <target_data><target_data_row>

target_data_row := <target_index> "," <target_begin> "," <target_len> ","
↪<target_name> ","
                   <target_version> "," <target_attributes> ";"
target_index := "target_index=" <N>
                Represents nth target in the table (from 0 to N-1 targets␣
↪specified in <num_device_targets>)
                If all the data for N targets doesn't fit in the given buffer -
↪ then the data that fits
                in the buffer (say from target 0 to x) is measured in a given␣
↪IMA event.
                The remaining data from targets x+1 to N-1 is measured in the␣
↪subsequent IMA events,
                with the same format as that of 'dm_table_load'
                i.e. <dm_version_str> ";" <device_metadata> ";" <table_load_
↪data>.

target_begin := "target_begin=" <N>
target_len := "target_len=" <N>
target_name := Name of the target. 'linear', 'crypt', 'integrity' etc.
               The targets that are supported for IMA measurements are␣
↪documented below in the
               'Supported targets' section.
target_version := "target_version=" <N> "." <N> "." <N>
target_attributes := Data containing comma separated list of name=value pairs␣
↪of target specific attributes.

For instance, if a linear device is created with the following table entries,
 # dmsetup create linear1
 0 2 linear /dev/loop0 512
 2 2 linear /dev/loop0 512
 4 2 linear /dev/loop0 512
 6 2 linear /dev/loop0 512

Then IMA ASCII measurement log will have the following entry:
(converted from ASCII to text for readability)

10 a8c5ff755561c7a28146389d1514c318592af49a ima-buf␣
↪sha256:4d73481ecce5eadba8ab084640d85bb9ca899af4d0a122989252a76efadc5b72
dm_table_load
dm_version=4.45.0;
name=linear1,uuid=,major=253,minor=0,minor_count=1,num_targets=4;
target_index=0,target_begin=0,target_len=2,target_name=linear,target_version=1.
↪4.0,device_name=7:0,start=512;
```

```
target_index=1,target_begin=2,target_len=2,target_name=linear,target_version=1.
↪4.0,device_name=7:0,start=512;
target_index=2,target_begin=4,target_len=2,target_name=linear,target_version=1.
↪4.0,device_name=7:0,start=512;
target_index=3,target_begin=6,target_len=2,target_name=linear,target_version=1.
↪4.0,device_name=7:0,start=512;
```

2. Device resume:

When a suspended device is resumed, the device information and the hash of the data from previous load of an active table are measured.

The IMA measurement log has the following format for 'dm_device_resume':

```
EVENT_NAME := "dm_device_resume"
EVENT_DATA := <dm_version_str> ";" <device_metadata> ";" <active_table_hash> ";
↪" <current_device_capacity> ";"

dm_version_str := As described in the 'Table load' section above.
device_metadata := As described in the 'Table load' section above.
active_table_hash := "active_table_hash=" <table_hash_alg> ":" <table_hash>
                     Rerpresents the hash of the IMA data being measured for␣
↪the
                     active table for the device.
table_hash_alg := Algorithm used to compute the hash.
table_hash := Hash of the (<dm_version_str> ";" <device_metadata> ";" <table_
↪load_data> ";")
              as described in the 'dm_table_load' above.
              Note: If the table_load data spans across multiple IMA 'dm_table_
↪load'
              events for a given device, the hash is computed combining all␣
↪the event data
              i.e. (<dm_version_str> ";" <device_metadata> ";" <table_load_
↪data> ";")
              across all those events.
current_device_capacity := "current_device_capacity=" <N>

For instance, if a linear device is resumed with the following command,
#dmsetup resume linear1

then IMA ASCII measurement log will have an entry with:
(converted from ASCII to text for readability)

10 56c00cc062ffc24ccd9ac2d67d194af3282b934e ima-buf␣
↪sha256:e7d12c03b958b4e0e53e7363a06376be88d98a1ac191fdbd3baf5e4b77f329b6
dm_device_resume
dm_version=4.45.0;
name=linear1,uuid=,major=253,minor=0,minor_count=1,num_targets=4;
active_table_
↪hash=sha256:4d73481ecce5eadba8ab084640d85bb9ca899af4d0a122989252a76efadc5b72;
↪current_device_capacity=8;
```

3. Device remove:

When a device is removed, the device information and a sha256 hash of the data from an active and inactive table are measured.

The IMA measurement log has the following format for 'dm_device_remove':

```
EVENT_NAME := "dm_device_remove"
EVENT_DATA := <dm_version_str> ";" <device_active_metadata> ";" <device_
↪inactive_metadata> ";"
              <active_table_hash> "," <inactive_table_hash> "," <remove_all> ";
↪" <current_device_capacity> ";"

dm_version_str := As described in the 'Table load' section above.
device_active_metadata := Device metadata that reflects the currently loaded␣
↪active table.
                          The format is same as 'device_metadata' described in␣
↪the 'Table load' section above.
device_inactive_metadata := Device metadata that reflects the inactive table.
                            The format is same as 'device_metadata' described␣
↪in the 'Table load' section above.
active_table_hash := Hash of the currently loaded active table.
                     The format is same as 'active_table_hash' described in␣
↪the 'Device resume' section above.
inactive_table_hash :=  Hash of the inactive table.
                        The format is same as 'active_table_hash' described in␣
↪the 'Device resume' section above.
remove_all := "remove_all=" <yes_no>
yes_no := "y" | "n"
current_device_capacity := "current_device_capacity=" <N>

For instance, if a linear device is removed with the following command,
 #dmsetup remove l1

then IMA ASCII measurement log will have the following entry:
(converted from ASCII to text for readability)

10 790e830a3a7a31590824ac0642b3b31c2d0e8b38 ima-buf␣
↪sha256:ab9f3c959367a8f5d4403d6ce9c3627dadfa8f9f0e7ec7899299782388de3840
dm_device_remove
dm_version=4.45.0;
device_active_metadata=name=l1,uuid=,major=253,minor=2,minor_count=1,num_
↪targets=2;
device_inactive_metadata=name=l1,uuid=,major=253,minor=2,minor_count=1,num_
↪targets=1;
active_table_
↪hash=sha256:4a7e62efaebfc86af755831998b7db6f59b60d23c9534fb16a4455907957953a,
inactive_table_
↪hash=sha256:9d79c175bc2302d55a183e8f50ad4bafd60f7692fd6249e5fd213e2464384b86,
↪remove_all=n;
```

```
current_device_capacity=2048;
```

4. Table clear:

When an inactive table is cleared from the device, the device information and a sha256 hash of the data from an inactive table are measured.

The IMA measurement log has the following format for 'dm_table_clear':

```
EVENT_NAME := "dm_table_clear"
EVENT_DATA := <dm_version_str> ";" <device_inactive_metadata> ";" <inactive_
↪table_hash> ";" <current_device_capacity> ";"

dm_version_str := As described in the 'Table load' section above.
device_inactive_metadata := Device metadata that was captured during the load␣
↪time inactive table being cleared.
                            The format is same as 'device_metadata' described␣
↪in the 'Table load' section above.
inactive_table_hash := Hash of the inactive table being cleared from the␣
↪device.
                       The format is same as 'active_table_hash' described in␣
↪the 'Device resume' section above.
current_device_capacity := "current_device_capacity=" <N>

For instance, if a linear device's inactive table is cleared,
 #dmsetup clear l1

then IMA ASCII measurement log will have an entry with:
(converted from ASCII to text for readability)

10 77d347408f557f68f0041acb0072946bb2367fe5 ima-buf␣
↪sha256:42f9ca22163fdfa548e6229dece2959bc5ce295c681644240035827ada0e1db5
dm_table_clear
dm_version=4.45.0;
name=l1,uuid=,major=253,minor=2,minor_count=1,num_targets=1;
inactive_table_
↪hash=sha256:75c0dc347063bf474d28a9907037eba060bfe39d8847fc0646d75e149045d545;
↪current_device_capacity=1024;
```

5. Device rename:

When an device's NAME or UUID is changed, the device information and the new NAME and UUID are measured.

The IMA measurement log has the following format for 'dm_device_rename':

```
EVENT_NAME := "dm_device_rename"
EVENT_DATA := <dm_version_str> ";" <device_active_metadata> ";" <new_device_
↪name> "," <new_device_uuid> ";" <current_device_capacity> ";"
```

```
dm_version_str := As described in the 'Table load' section above.
device_active_metadata := Device metadata that reflects the currently loaded␣
↪active table.
                          The format is same as 'device_metadata' described in␣
↪the 'Table load' section above.
new_device_name := "new_name=" <dm-device-name>
dm-device-name := Same as <dm-device-name> described in 'Table load' section␣
↪above
new_device_uuid := "new_uuid=" <dm-device-uuid>
dm-device-uuid := Same as <dm-device-uuid> described in 'Table load' section␣
↪above
current_device_capacity := "current_device_capacity=" <N>

E.g 1: if a linear device's name is changed with the following command,
 #dmsetup rename linear1 --setuuid 1234-5678

then IMA ASCII measurement log will have an entry with:
(converted from ASCII to text for readability)

10 8b0423209b4c66ac1523f4c9848c9b51ee332f48 ima-buf␣
↪sha256:6847b7258134189531db593e9230b257c84f04038b5a18fd2e1473860e0569ac
dm_device_rename
dm_version=4.45.0;
name=linear1,uuid=,major=253,minor=2,minor_count=1,num_targets=1;new_
↪name=linear1,new_uuid=1234-5678;
current_device_capacity=1024;

E.g 2:  if a linear device's name is changed with the following command,
 # dmsetup rename linear1 linear=2

then IMA ASCII measurement log will have an entry with:
(converted from ASCII to text for readability)

10 bef70476b99c2bdf7136fae033aa8627da1bf76f ima-buf␣
↪sha256:8c6f9f53b9ef9dc8f92a2f2cca8910e622543d0f0d37d484870cb16b95111402
dm_device_rename
dm_version=4.45.0;
name=linear1,uuid=1234-5678,major=253,minor=2,minor_count=1,num_targets=1;
new_name=linear\=2,new_uuid=1234-5678;
current_device_capacity=1024;
```

39.9.3 Supported targets:

Following targets are supported to measure their data using IMA:

1. cache
2. crypt
3. integrity
4. linear
5. mirror
6. multipath
7. raid
8. snapshot
9. striped
10. verity

1. cache

The 'target_attributes' (described as part of EVENT_DATA in 'Table load' section above) has the following data format for 'cache' target.

```
target_attributes := <target_name> "," <target_version> "," <metadata_mode> ",
↪" <cache_metadata_device> ","
                     <cache_device> "," <cache_origin_device> ","
↪<writethrough> "," <writeback> ","
                     <passthrough> "," <no_discard_passdown> ";"

target_name := "target_name=cache"
target_version := "target_version=" <N> "." <N> "." <N>
metadata_mode := "metadata_mode=" <cache_metadata_mode>
cache_metadata_mode := "fail" | "ro" | "rw"
cache_device := "cache_device=" <cache_device_name_string>
cache_origin_device := "cache_origin_device=" <cache_origin_device_string>
writethrough := "writethrough=" <yes_no>
writeback := "writeback=" <yes_no>
passthrough := "passthrough=" <yes_no>
no_discard_passdown := "no_discard_passdown=" <yes_no>
yes_no := "y" | "n"

E.g.
When a 'cache' target is loaded, then IMA ASCII measurement log will have an␣
↪entry
similar to the following, depicting what 'cache' attributes are measured in␣
↪EVENT_DATA
for 'dm_table_load' event.
(converted from ASCII to text for readability)

dm_version=4.45.0;name=cache1,uuid=cache_uuid,major=253,minor=2,minor_count=1,
↪num_targets=1;
```

```
target_index=0,target_begin=0,target_len=28672,target_name=cache,target_
↪version=2.2.0,metadata_mode=rw,
cache_metadata_device=253:4,cache_device=253:3,cache_origin_device=253:5,
↪writethrough=y,writeback=n,
passthrough=n,metadata2=y,no_discard_passdown=n;
```

2. crypt

The 'target_attributes' (described as part of EVENT_DATA in 'Table load' section above) has the following data format for 'crypt' target.

```
target_attributes := <target_name> "," <target_version> "," <allow_discards> ",
↪" <same_cpu_crypt> ","
                     <submit_from_crypt_cpus> "," <no_read_workqueue> "," <no_
↪write_workqueue> ","
                     <iv_large_sectors> "," <iv_large_sectors> "," [<integrity_
↪tag_size> ","] [<cipher_auth> ","]
                     [<sector_size> ","] [<cipher_string> ","] <key_size> ","
↪<key_parts> ","
                     <key_extra_size> "," <key_mac_size> ";"

target_name := "target_name=crypt"
target_version := "target_version=" <N> "." <N> "." <N>
allow_discards := "allow_discards=" <yes_no>
same_cpu_crypt := "same_cpu_crypt=" <yes_no>
submit_from_crypt_cpus := "submit_from_crypt_cpus=" <yes_no>
no_read_workqueue := "no_read_workqueue=" <yes_no>
no_write_workqueue := "no_write_workqueue=" <yes_no>
iv_large_sectors := "iv_large_sectors=" <yes_no>
integrity_tag_size := "integrity_tag_size=" <N>
cipher_auth := "cipher_auth=" <string>
sector_size := "sector_size="  <N>
cipher_string := "cipher_string="
key_size := "key_size="  <N>
key_parts := "key_parts="  <N>
key_extra_size := "key_extra_size="  <N>
key_mac_size := "key_mac_size="  <N>
yes_no := "y" | "n"

E.g.
When a 'crypt' target is loaded, then IMA ASCII measurement log will have an␣
↪entry
similar to the following, depicting what 'crypt' attributes are measured in␣
↪EVENT_DATA
for 'dm_table_load' event.
(converted from ASCII to text for readability)

dm_version=4.45.0;
name=crypt1,uuid=crypt_uuid1,major=253,minor=0,minor_count=1,num_targets=1;
target_index=0,target_begin=0,target_len=1953125,target_name=crypt,target_
↪version=1.23.0,
```

```
allow_discards=y,same_cpu=n,submit_from_crypt_cpus=n,no_read_workqueue=n,no_
↪write_workqueue=n,
iv_large_sectors=n,cipher_string=aes-xts-plain64,key_size=32,key_parts=1,key_
↪extra_size=0,key_mac_size=0;
```

3. integrity

The 'target_attributes' (described as part of EVENT_DATA in 'Table load' section above) has the following data format for 'integrity' target.

```
target_attributes := <target_name> "," <target_version> "," <dev_name> ","
↪<start>
                     <tag_size> "," <mode> "," [<meta_device> ","] [<block_
↪size> ","] <recalculate> ","
                     <allow_discards> "," <fix_padding> "," <fix_hmac> ","
↪<legacy_recalculate> ","
                     <journal_sectors> "," <interleave_sectors> "," <buffer_
↪sectors> ";"

target_name := "target_name=integrity"
target_version := "target_version=" <N> "." <N> "." <N>
dev_name := "dev_name=" <device_name_str>
start := "start=" <N>
tag_size := "tag_size=" <N>
mode := "mode=" <integrity_mode_str>
integrity_mode_str := "J" | "B" | "D" | "R"
meta_device := "meta_device=" <meta_device_str>
block_size := "block_size=" <N>
recalculate := "recalculate=" <yes_no>
allow_discards := "allow_discards=" <yes_no>
fix_padding := "fix_padding=" <yes_no>
fix_hmac := "fix_hmac=" <yes_no>
legacy_recalculate := "legacy_recalculate=" <yes_no>
journal_sectors := "journal_sectors=" <N>
interleave_sectors := "interleave_sectors=" <N>
buffer_sectors := "buffer_sectors=" <N>
yes_no := "y" | "n"

E.g.
When a 'integrity' target is loaded, then IMA ASCII measurement log will have␣
↪an entry
similar to the following, depicting what 'integrity' attributes are measured␣
↪in EVENT_DATA
for 'dm_table_load' event.
(converted from ASCII to text for readability)

dm_version=4.45.0;
name=integrity1,uuid=,major=253,minor=1,minor_count=1,num_targets=1;
target_index=0,target_begin=0,target_len=7856,target_name=integrity,target_
↪version=1.10.0,
```

```
dev_name=253:0,start=0,tag_size=32,mode=J,recalculate=n,allow_discards=n,fix_
↪padding=n,
fix_hmac=n,legacy_recalculate=n,journal_sectors=88,interleave_sectors=32768,
↪buffer_sectors=128;
```

4. linear

The 'target_attributes' (described as part of EVENT_DATA in 'Table load' section above) has the following data format for 'linear' target.

```
target_attributes := <target_name> "," <target_version> "," <device_name> <,>
↪<start> ";"

target_name := "target_name=linear"
target_version := "target_version=" <N> "." <N> "." <N>
device_name := "device_name=" <linear_device_name_str>
start := "start=" <N>

E.g.
When a 'linear' target is loaded, then IMA ASCII measurement log will have an␣
↪entry
similar to the following, depicting what 'linear' attributes are measured in␣
↪EVENT_DATA
for 'dm_table_load' event.
(converted from ASCII to text for readability)

dm_version=4.45.0;
name=linear1,uuid=linear_uuid1,major=253,minor=2,minor_count=1,num_targets=1;
target_index=0,target_begin=0,target_len=28672,target_name=linear,target_
↪version=1.4.0,
device_name=253:1,start=2048;
```

5. mirror

The 'target_attributes' (described as part of EVENT_DATA in 'Table load' section above) has the following data format for 'mirror' target.

```
target_attributes := <target_name> "," <target_version> "," <nr_mirrors> ","
                     <mirror_device_data> "," <handle_errors> "," <keep_log> ",
↪" <log_type_status> ";"

target_name := "target_name=mirror"
target_version := "target_version=" <N> "." <N> "." <N>
nr_mirrors := "nr_mirrors=" <NR>
mirror_device_data := <mirror_device_row> | <mirror_device_data><mirror_device_
↪row>
                      mirror_device_row is repeated <NR> times - for <NR>␣
↪described in <nr_mirrors>.
mirror_device_row := <mirror_device_name> "," <mirror_device_status>
```

```
mirror_device_name := "mirror_device_" <X> "=" <mirror_device_name_str>
                      where <X> ranges from 0 to (<NR> -1) - for <NR>␣
↪described in <nr_mirrors>.
mirror_device_status := "mirror_device_" <X> "_status=" <mirror_device_status_
↪char>
                       where <X> ranges from 0 to (<NR> -1) - for <NR>␣
↪described in <nr_mirrors>.
mirror_device_status_char := "A" | "F" | "D" | "S" | "R" | "U"
handle_errors := "handle_errors=" <yes_no>
keep_log := "keep_log=" <yes_no>
log_type_status := "log_type_status=" <log_type_status_str>
yes_no := "y" | "n"

E.g.
When a 'mirror' target is loaded, then IMA ASCII measurement log will have an␣
↪entry
similar to the following, depicting what 'mirror' attributes are measured in␣
↪EVENT_DATA
for 'dm_table_load' event.
(converted from ASCII to text for readability)

dm_version=4.45.0;
name=mirror1,uuid=mirror_uuid1,major=253,minor=6,minor_count=1,num_targets=1;
target_index=0,target_begin=0,target_len=2048,target_name=mirror,target_
↪version=1.14.0,nr_mirrors=2,
   mirror_device_0=253:4,mirror_device_0_status=A,
   mirror_device_1=253:5,mirror_device_1_status=A,
handle_errors=y,keep_log=n,log_type_status=;
```

6. multipath

The 'target_attributes' (described as part of EVENT_DATA in 'Table load' section above) has the following data format for 'multipath' target.

```
target_attributes := <target_name> "," <target_version> "," <nr_priority_
↪groups>
                     ["," <pg_state> "," <priority_groups> "," <priority_group_
↪paths>] ";"

target_name := "target_name=multipath"
target_version := "target_version=" <N> "." <N> "." <N>
nr_priority_groups := "nr_priority_groups=" <NPG>
priority_groups := <priority_groups_row>|<priority_groups_row><priority_groups>
priority_groups_row := "pg_state_" <X> "=" <pg_state_str> "," "nr_pgpaths_" <X>
↪  "=" <NPGP> ","
                       "path_selector_name_" <X> "=" <string> "," <priority_
↪group_paths>
                       where <X> ranges from 0 to (<NPG> -1) - for <NPG>␣
↪described in <nr_priority_groups>.
pg_state_str := "E" | "A" | "D"
```

```
<priority_group_paths> := <priority_group_paths_row> | <priority_group_paths_
↪row><priority_group_paths>
priority_group_paths_row := "path_name_" <X> "_" <Y> "=" <string> "," "is_
↪active_" <X> "_" <Y> "=" <is_active_str>
                            "fail_count_" <X> "_" <Y> "=" <N> "," "path_
↪selector_status_" <X> "_" <Y> "=" <path_selector_status_str>
                            where <X> ranges from 0 to (<NPG> -1) - for <NPG>␣
↪described in <nr_priority_groups>,
                            and <Y> ranges from 0 to (<NPGP> -1) - for <NPGP>␣
↪described in <priority_groups_row>.
is_active_str := "A" | "F"

E.g.
When a 'multipath' target is loaded, then IMA ASCII measurement log will have␣
↪an entry
similar to the following, depicting what 'multipath' attributes are measured␣
↪in EVENT_DATA
for 'dm_table_load' event.
(converted from ASCII to text for readability)

dm_version=4.45.0;
name=mp,uuid=,major=253,minor=0,minor_count=1,num_targets=1;
target_index=0,target_begin=0,target_len=2097152,target_name=multipath,target_
↪version=1.14.0,nr_priority_groups=2,
   pg_state_0=E,nr_pgpaths_0=2,path_selector_name_0=queue-length,
       path_name_0_0=8:16,is_active_0_0=A,fail_count_0_0=0,path_selector_
↪status_0_0=,
       path_name_0_1=8:32,is_active_0_1=A,fail_count_0_1=0,path_selector_
↪status_0_1=,
   pg_state_1=E,nr_pgpaths_1=2,path_selector_name_1=queue-length,
       path_name_1_0=8:48,is_active_1_0=A,fail_count_1_0=0,path_selector_
↪status_1_0=,
       path_name_1_1=8:64,is_active_1_1=A,fail_count_1_1=0,path_selector_
↪status_1_1=;
```

7. raid

The 'target_attributes' (described as part of EVENT_DATA in 'Table load' section above) has the following data format for 'raid' target.

```
target_attributes := <target_name> "," <target_version> "," <raid_type> ","
↪<raid_disks> "," <raid_state>
                     <raid_device_status> ["," journal_dev_mode] ";"

target_name := "target_name=raid"
target_version := "target_version=" <N> "." <N> "." <N>
raid_type := "raid_type=" <raid_type_str>
raid_disks := "raid_disks=" <NRD>
raid_state := "raid_state=" <raid_state_str>
raid_state_str := "frozen" | "reshape" |"resync" | "check" | "repair" |
↪"recover" | "idle" |"undef"
```

```
raid_device_status := <raid_device_status_row> | <raid_device_status_row><raid_
↪device_status>
                      <raid_device_status_row> is repeated <NRD> times - for
↪<NRD> described in <raid_disks>.
raid_device_status_row := "raid_device_" <X> "_status=" <raid_device_status_
↪str>
                          where <X> ranges from 0 to (<NRD> -1) - for <NRD>␣
↪described in <raid_disks>.
raid_device_status_str := "A" | "D" | "a" | "-"
journal_dev_mode := "journal_dev_mode=" <journal_dev_mode_str>
journal_dev_mode_str := "writethrough" | "writeback" | "invalid"

E.g.
When a 'raid' target is loaded, then IMA ASCII measurement log will have an␣
↪entry
similar to the following, depicting what 'raid' attributes are measured in␣
↪EVENT_DATA
for 'dm_table_load' event.
(converted from ASCII to text for readability)

dm_version=4.45.0;
name=raid_LV1,uuid=uuid_raid_LV1,major=253,minor=12,minor_count=1,num_
↪targets=1;
target_index=0,target_begin=0,target_len=2048,target_name=raid,target_
↪version=1.15.1,
raid_type=raid10,raid_disks=4,raid_state=idle,
   raid_device_0_status=A,
   raid_device_1_status=A,
   raid_device_2_status=A,
   raid_device_3_status=A;
```

8. snapshot

The 'target_attributes' (described as part of EVENT_DATA in 'Table load' section above) has the following data format for 'snapshot' target.

```
target_attributes := <target_name> "," <target_version> "," <snap_origin_name>
↪","
                     <snap_cow_name> "," <snap_valid> "," <snap_merge_failed>
↪"," <snapshot_overflowed> ";"

target_name := "target_name=snapshot"
target_version := "target_version=" <N> "." <N> "." <N>
snap_origin_name := "snap_origin_name=" <string>
snap_cow_name := "snap_cow_name=" <string>
snap_valid := "snap_valid=" <yes_no>
snap_merge_failed := "snap_merge_failed=" <yes_no>
snapshot_overflowed := "snapshot_overflowed=" <yes_no>
yes_no := "y" | "n"
```

```
E.g.
When a 'snapshot' target is loaded, then IMA ASCII measurement log will have␣
↪an entry
similar to the following, depicting what 'snapshot' attributes are measured in␣
↪EVENT_DATA
for 'dm_table_load' event.
(converted from ASCII to text for readability)

dm_version=4.45.0;
name=snap1,uuid=snap_uuid1,major=253,minor=13,minor_count=1,num_targets=1;
target_index=0,target_begin=0,target_len=4096,target_name=snapshot,target_
↪version=1.16.0,
snap_origin_name=253:11,snap_cow_name=253:12,snap_valid=y,snap_merge_failed=n,
↪snapshot_overflowed=n;
```

9. striped

The 'target_attributes' (described as part of EVENT_DATA in 'Table load' section above) has the following data format for 'striped' target.

```
target_attributes := <target_name> "," <target_version> "," <stripes> ","
↪<chunk_size> ","
                     <stripe_data> ";"

target_name := "target_name=striped"
target_version := "target_version=" <N> "." <N> "." <N>
stripes := "stripes=" <NS>
chunk_size := "chunk_size=" <N>
stripe_data := <stripe_data_row>|<stripe_data><stripe_data_row>
stripe_data_row := <stripe_device_name> "," <stripe_physical_start> ","
↪<stripe_status>
stripe_device_name := "stripe_" <X> "_device_name=" <stripe_device_name_str>
                      where <X> ranges from 0 to (<NS> -1) - for <NS>␣
↪described in <stripes>.
stripe_physical_start := "stripe_" <X> "_physical_start=" <N>
                          where <X> ranges from 0 to (<NS> -1) - for <NS>␣
↪described in <stripes>.
stripe_status := "stripe_" <X> "_status=" <stripe_status_str>
                 where <X> ranges from 0 to (<NS> -1) - for <NS> described in
↪<stripes>.
stripe_status_str := "D" | "A"

E.g.
When a 'striped' target is loaded, then IMA ASCII measurement log will have an␣
↪entry
similar to the following, depicting what 'striped' attributes are measured in␣
↪EVENT_DATA
for 'dm_table_load' event.
(converted from ASCII to text for readability)
```

```
dm_version=4.45.0;
name=striped1,uuid=striped_uuid1,major=253,minor=5,minor_count=1,num_targets=1;
target_index=0,target_begin=0,target_len=640,target_name=striped,target_
↪version=1.6.0,stripes=2,chunk_size=64,
   stripe_0_device_name=253:0,stripe_0_physical_start=2048,stripe_0_status=A,
   stripe_1_device_name=253:3,stripe_1_physical_start=2048,stripe_1_status=A;
```

10. verity

The 'target_attributes' (described as part of EVENT_DATA in 'Table load' section above) has the following data format for 'verity' target.

```
target_attributes := <target_name> "," <target_version> "," <hash_failed> ","
↪<verity_version> ","
                     <data_device_name> "," <hash_device_name> "," <verity_
↪algorithm> "," <root_digest> ","
                     <salt> "," <ignore_zero_blocks> "," <check_at_most_once> [
↪"," <root_hash_sig_key_desc>]
                     ["," <verity_mode>] ";"

target_name := "target_name=verity"
target_version := "target_version=" <N> "." <N> "." <N>
hash_failed := "hash_failed=" <hash_failed_str>
hash_failed_str := "C" | "V"
verity_version := "verity_version=" <verity_version_str>
data_device_name := "data_device_name=" <data_device_name_str>
hash_device_name := "hash_device_name=" <hash_device_name_str>
verity_algorithm := "verity_algorithm=" <verity_algorithm_str>
root_digest := "root_digest=" <root_digest_str>
salt := "salt=" <salt_str>
salt_str := "-" <verity_salt_str>
ignore_zero_blocks := "ignore_zero_blocks=" <yes_no>
check_at_most_once := "check_at_most_once=" <yes_no>
root_hash_sig_key_desc := "root_hash_sig_key_desc="
verity_mode := "verity_mode=" <verity_mode_str>
verity_mode_str := "ignore_corruption" | "restart_on_corruption" | "panic_on_
↪corruption" | "invalid"
yes_no := "y" | "n"

E.g.
When a 'verity' target is loaded, then IMA ASCII measurement log will have an␣
↪entry
similar to the following, depicting what 'verity' attributes are measured in␣
↪EVENT_DATA
for 'dm_table_load' event.
(converted from ASCII to text for readability)

dm_version=4.45.0;
name=test-verity,uuid=,major=253,minor=2,minor_count=1,num_targets=1;
target_index=0,target_begin=0,target_len=1953120,target_name=verity,target_
↪version=1.8.0,hash_failed=V,
```

```
verity_version=1,data_device_name=253:1,hash_device_name=253:0,verity_
↪algorithm=sha256,
root_digest=29cb87e60ce7b12b443ba6008266f3e41e93e403d7f298f8e3f316b29ff89c5e,
salt=e48da609055204e89ae53b655ca2216dd983cf3cb829f34f63a297d106d53e2d,
ignore_zero_blocks=n,check_at_most_once=n;
```

39.10 Early creation of mapped devices

It is possible to configure a device-mapper device to act as the root device for your system in two ways.

The first is to build an initial ramdisk which boots to a minimal userspace which configures the device, then pivot_root(8) in to it.

The second is to create one or more device-mappers using the module parameter "dm-mod.create=" through the kernel boot command line argument.

The format is specified as a string of data separated by commas and optionally semi-colons, where:

- a comma is used to separate fields like name, uuid, flags and table (specifies one device)
- a semi-colon is used to separate devices.

So the format will look like this:

```
dm-mod.create=<name>,<uuid>,<minor>,<flags>,<table>[,<table>+][;<name>,<uuid>,
↪<minor>,<flags>,<table>[,<table>+]+]
```

Where:

```
<name>          ::= The device name.
<uuid>          ::= xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx | ""
<minor>         ::= The device minor number | ""
<flags>         ::= "ro" | "rw"
<table>         ::= <start_sector> <num_sectors> <target_type> <target_args>
<target_type>   ::= "verity" | "linear" | ... (see list below)
```

The dm line should be equivalent to the one used by the dmsetup tool with the *--concise* argument.

39.10.1 Target types

Not all target types are available as there are serious risks in allowing activation of certain DM targets without first using userspace tools to check the validity of associated metadata.

cache	constrained, userspace should verify cache device
crypt	allowed
delay	allowed
era	constrained, userspace should verify metadata device
flakey	constrained, meant for test
linear	allowed
log-writes	constrained, userspace should verify metadata device
mirror	constrained, userspace should verify main/mirror device
raid	constrained, userspace should verify metadata device
snapshot	constrained, userspace should verify src/dst device
snapshot-origin	allowed
snapshot-merge	constrained, userspace should verify src/dst device
striped	allowed
switch	constrained, userspace should verify dev path
thin	constrained, requires dm target message from userspace
thin-pool	constrained, requires dm target message from userspace
verity	allowed
writecache	constrained, userspace should verify cache device
zero	constrained, not meant for rootfs

If the target is not listed above, it is constrained by default (not tested).

39.10.2 Examples

An example of booting to a linear array made up of user-mode linux block devices:

```
dm-mod.create="lroot,,,rw, 0 4096 linear 98:16 0, 4096 4096 linear 98:32 0"␣
↪root=/dev/dm-0
```

This will boot to a rw dm-linear target of 8192 sectors split across two block devices identified by their major:minor numbers. After boot, udev will rename this target to /dev/mapper/lroot (depending on the rules). No uuid was assigned.

An example of multiple device-mappers, with the dm-mod.create="..." contents is shown here split on multiple lines for readability:

```
dm-linear,,1,rw,
  0 32768 linear 8:1 0,
  32768 1024000 linear 8:2 0;
dm-verity,,3,ro,
  0 1638400 verity 1 /dev/sdc1 /dev/sdc2 4096 4096 204800 1 sha256
  ac87db56303c9c1da433d7209b5a6ef3e4779df141200cbd7c157dcb8dd89c42
  5ebfe87f7df3235b80a117ebc4078e44f55045487ad4a96581d1adb564615b51
```

Other examples (per target):

"crypt":

```
dm-crypt,,8,ro,
  0 1048576 crypt aes-xts-plain64
```

```
  babebabebabebab
```

The dm-integrity target can be used with the dm-crypt target - in this situation the dm-crypt target creates the integrity data and passes them to the dm-integrity target via bio_integrity_payload attached to the bio. In this mode, the dm-crypt and dm-integrity targets provide authenticated disk encryption - if the attacker modifies the encrypted device, an I/O error is returned instead of random data.

The dm-integrity target can also be used as a standalone target, in this mode it calculates and verifies the integrity tag internally. In this mode, the dm-integrity target can be used to detect silent data corruption on the disk or in the I/O path.

There's an alternate mode of operation where dm-integrity uses a bitmap instead of a journal. If a bit in the bitmap is 1, the corresponding region's data and integrity tags are not synchronized - if the machine crashes, the unsynchronized regions will be recalculated. The bitmap mode is faster than the journal mode, because we don't have to write the data twice, but it is also less reliable, because if data corruption happens when the machine crashes, it may not be detected.

When loading the target for the first time, the kernel driver will format the device. But it will only format the device if the superblock contains zeroes. If the superblock is neither valid nor zeroed, the dm-integrity target can't be loaded.

Accesses to the on-disk metadata area containing checksums (aka tags) are buffered using dm-bufio. When an access to any given metadata area occurs, each unique metadata area gets its own buffer(s). The buffer size is capped at the size of the metadata area, but may be smaller, thereby requiring multiple buffers to represent the full metadata area. A smaller buffer size will produce a smaller resulting read/write operation to the metadata area for small reads/writes. The metadata is still read even in a full write to the data covered by a single buffer.

To use the target for the first time:

1. overwrite the superblock with zeroes
2. load the dm-integrity target with one-sector size, the kernel driver will format the device
3. unload the dm-integrity target
4. read the ”provided_data_sectors” value from the superblock
5. load the dm-integrity target with the target size ”provided_data_sectors”
6. if you want to use dm-integrity with dm-crypt, load the dm-crypt target with the size ”provided_data_sectors”

Target arguments:

1. the underlying block device
2. the number of reserved sector at the beginning of the device - the dm-integrity won't read of write these sectors
3. the size of the integrity tag (if ”-” is used, the size is taken from the internal-hash algorithm)
4. mode:

   D - direct writes (without journal)
   in this mode, journaling is not used and data sectors and integrity tags are written separately. In case of crash, it is possible that the data and integrity tag doesn't match.

   J - journaled writes
   data and integrity tags are written to the journal and atomicity is guaranteed.

In case of crash, either both data and tag or none of them are written. The journaled mode degrades write throughput twice because the data have to be written twice.

B - bitmap mode - data and metadata are written without any synchronization, the driver maintains a bitmap of dirty regions where data and metadata don't match. This mode can only be used with internal hash.

R - recovery mode - in this mode, journal is not replayed, checksums are not checked and writes to the device are not allowed. This mode is useful for data recovery if the device cannot be activated in any of the other standard modes.

5. the number of additional arguments

Additional arguments:

journal_sectors:number
The size of journal, this argument is used only if formatting the device. If the device is already formatted, the value from the superblock is used.

interleave_sectors:number (default 32768)
The number of interleaved sectors. This values is rounded down to a power of two. If the device is already formatted, the value from the superblock is used.

meta_device:device
Don't interleave the data and metadata on the device. Use a separate device for metadata.

buffer_sectors:number (default 128)
The number of sectors in one metadata buffer. The value is rounded down to a power of two.

journal_watermark:number (default 50)
The journal watermark in percents. When the size of the journal exceeds this watermark, the thread that flushes the journal will be started.

commit_time:number (default 10000)
Commit time in milliseconds. When this time passes, the journal is written. The journal is also written immediately if the FLUSH request is received.

internal_hash:algorithm(:key) (the key is optional)
Use internal hash or crc. When this argument is used, the dm-integrity target won't accept integrity tags from the upper target, but it will automatically generate and verify the integrity tags.

You can use a crc algorithm (such as crc32), then integrity target will protect the data against accidental corruption. You can also use a hmac algorithm (for example "hmac(sha256):0123456789abcdef"), in this mode it will provide cryptographic authentication of the data without encryption.

When this argument is not used, the integrity tags are accepted from an upper layer target, such as dm-crypt. The upper layer target should check the validity of the integrity tags.

recalculate
Recalculate the integrity tags automatically. It is only valid when using internal hash.

journal_crypt:algorithm(:key) (the key is optional)
Encrypt the journal using given algorithm to make sure that the attacker can't read the

journal. You can use a block cipher here (such as "cbc(aes)") or a stream cipher (for example "chacha20" or "ctr(aes)").

The journal contains history of last writes to the block device, an attacker reading the journal could see the last sector numbers that were written. From the sector numbers, the attacker can infer the size of files that were written. To protect against this situation, you can encrypt the journal.

journal_mac:algorithm(:key) (the key is optional)
Protect sector numbers in the journal from accidental or malicious modification. To protect against accidental modification, use a crc algorithm, to protect against malicious modification, use a hmac algorithm with a key.

This option is not needed when using internal-hash because in this mode, the integrity of journal entries is checked when replaying the journal. Thus, modified sector number would be detected at this stage.

block_size:number (default 512)
The size of a data block in bytes. The larger the block size the less overhead there is for per-block integrity metadata. Supported values are 512, 1024, 2048 and 4096 bytes.

sectors_per_bit:number
In the bitmap mode, this parameter specifies the number of 512-byte sectors that corresponds to one bitmap bit.

bitmap_flush_interval:number
The bitmap flush interval in milliseconds. The metadata buffers are synchronized when this interval expires.

allow_discards
Allow block discard requests (a.k.a. TRIM) for the integrity device. Discards are only allowed to devices using internal hash.

fix_padding
Use a smaller padding of the tag area that is more space-efficient. If this option is not present, large padding is used - that is for compatibility with older kernels.

fix_hmac
Improve security of internal_hash and journal_mac:

- the section number is mixed to the mac, so that an attacker can't copy sectors from one journal section to another journal section
- the superblock is protected by journal_mac
- a 16-byte salt stored in the superblock is mixed to the mac, so that the attacker can't detect that two disks have the same hmac key and also to disallow the attacker to move sectors from one disk to another

legacy_recalculate
Allow recalculating of volumes with HMAC keys. This is disabled by default for security reasons - an attacker could modify the volume, set recalc_sector to zero, and the kernel would not detect the modification.

The journal mode (D/J), buffer_sectors, journal_watermark, commit_time and allow_discards can be changed when reloading the target (load an inactive table and swap the tables with suspend and resume). The other arguments should not be changed when reloading the target because the layout of disk data depend on them and the reloaded target would be non-functional.

For example, on a device using the default interleave_sectors of 32768, a block_size of 512, and an internal_hash of crc32c with a tag size of 4 bytes, it will take 128 KiB of tags to track a full data area, requiring 256 sectors of metadata per data area. With the default buffer_sectors of 128, that means there will be 2 buffers per metadata area, or 2 buffers per 16 MiB of data.

Status line:

1. the number of integrity mismatches
2. provided data sectors - that is the number of sectors that the user could use
3. the current recalculating position (or '-' if we didn't recalculate)

The layout of the formatted block device:

- **reserved sectors**
  (they are not used by this target, they can be used for storing LUKS metadata or for other purpose), the size of the reserved area is specified in the target arguments
- **superblock (4kiB)**
  - magic string - identifies that the device was formatted
  - version
  - log2(interleave sectors)
  - integrity tag size
  - the number of journal sections
  - provided data sectors - the number of sectors that this target provides (i.e. the size of the device minus the size of all metadata and padding). The user of this target should not send bios that access data beyond the ”provided data sectors” limit.
  - **flags**

    SB_FLAG_HAVE_JOURNAL_MAC

    * a flag is set if journal_mac is used

    SB_FLAG_RECALCULATING

    * recalculating is in progress

    SB_FLAG_DIRTY_BITMAP

    * journal area contains the bitmap of dirty blocks
  - log2(sectors per block)
  - a position where recalculating finished
- **journal**
  The journal is divided into sections, each section contains:
  - metadata area (4kiB), it contains journal entries
    * every journal entry contains:
      · logical sector (specifies where the data and tag should be written)
      · last 8 bytes of data

        - integrity tag (the size is specified in the superblock)
      - every metadata sector ends with
        - mac (8-bytes), all the macs in 8 metadata sectors form a 64-byte value. It is used to store hmac of sector numbers in the journal section, to protect against a possibility that the attacker tampers with sector numbers in the journal.
        - commit id
    - data area (the size is variable; it depends on how many journal entries fit into the metadata area)
      - every sector in the data area contains:
        - data (504 bytes of data, the last 8 bytes are stored in the journal entry)
        - commit id

    To test if the whole journal section was written correctly, every 512-byte sector of the journal ends with 8-byte commit id. If the commit id matches on all sectors in a journal section, then it is assumed that the section was written correctly. If the commit id doesn't match, the section was written partially and it should not be replayed.

- **one or more runs of interleaved tags and data.**
  Each run contains:
  - tag area - it contains integrity tags. There is one tag for each sector in the data area. The size of this area is always 4KiB or greater.
  - data area - it contains data sectors. The number of data sectors in one run must be a power of two. log2 of this value is stored in the superblock.

39.12 dm-io

Dm-io provides synchronous and asynchronous I/O services. There are three types of I/O services available, and each type has a sync and an async version.

The user must set up an io_region structure to describe the desired location of the I/O. Each io_region indicates a block-device along with the starting sector and size of the region:

```
struct io_region {
   struct block_device *bdev;
   sector_t sector;
   sector_t count;
};
```

Dm-io can read from one io_region or write to one or more io_regions. Writes to multiple regions are specified by an array of io_region structures.

The first I/O service type takes a list of memory pages as the data buffer for the I/O, along with an offset into the first page:

```
struct page_list {
   struct page_list *next;
   struct page *page;
```

```
};

int dm_io_sync(unsigned int num_regions, struct io_region *where, int rw,
               struct page_list *pl, unsigned int offset,
               unsigned long *error_bits);
int dm_io_async(unsigned int num_regions, struct io_region *where, int rw,
                struct page_list *pl, unsigned int offset,
                io_notify_fn fn, void *context);
```

The second I/O service type takes an array of bio vectors as the data buffer for the I/O. This service can be handy if the caller has a pre-assembled bio, but wants to direct different portions of the bio to different devices:

```
int dm_io_sync_bvec(unsigned int num_regions, struct io_region *where,
                    int rw, struct bio_vec *bvec,
                    unsigned long *error_bits);
int dm_io_async_bvec(unsigned int num_regions, struct io_region *where,
                     int rw, struct bio_vec *bvec,
                     io_notify_fn fn, void *context);
```

The third I/O service type takes a pointer to a vmalloc'd memory buffer as the data buffer for the I/O. This service can be handy if the caller needs to do I/O to a large region but doesn't want to allocate a large number of individual memory pages:

```
int dm_io_sync_vm(unsigned int num_regions, struct io_region *where, int rw,
                  void *data, unsigned long *error_bits);
int dm_io_async_vm(unsigned int num_regions, struct io_region *where, int rw,
                   void *data, io_notify_fn fn, void *context);
```

Callers of the asynchronous I/O services must include the name of a completion callback routine and a pointer to some context data for the I/O:

```
typedef void (*io_notify_fn)(unsigned long error, void *context);
```

The ”error” parameter in this callback, as well as the **error* parameter in all of the synchronous versions, is a bitset (instead of a simple error value). In the case of an write-I/O to multiple regions, this bitset allows dm-io to indicate success or failure on each individual region.

Before using any of the dm-io services, the user should call dm_io_get() and specify the number of pages they expect to perform I/O on concurrently. Dm-io will attempt to resize its mempool to make sure enough pages are always available in order to avoid unnecessary waiting while performing I/O.

When the user is finished using the dm-io services, they should call dm_io_put() and specify the same number of pages that were given on the dm_io_get() call.

39.13 Device-Mapper Logging

The device-mapper logging code is used by some of the device-mapper RAID targets to track regions of the disk that are not consistent. A region (or portion of the address space) of the disk may be inconsistent because a RAID stripe is currently being operated on or a machine died while the region was being altered. In the case of mirrors, a region would be considered dirty/inconsistent while you are writing to it because the writes need to be replicated for all the legs of the mirror and may not reach the legs at the same time. Once all writes are complete, the region is considered clean again.

There is a generic logging interface that the device-mapper RAID implementations use to perform logging operations (see dm_dirty_log_type in include/linux/dm-dirty-log.h). Various different logging implementations are available and provide different capabilities. The list includes:

Type	Files
disk	drivers/md/dm-log.c
core	drivers/md/dm-log.c
userspace	drivers/md/dm-log-userspace* include/linux/dm-log-userspace.h

39.13.1 The ”disk” log type

This log implementation commits the log state to disk. This way, the logging state survives reboots/crashes.

39.13.2 The ”core” log type

This log implementation keeps the log state in memory. The log state will not survive a reboot or crash, but there may be a small boost in performance. This method can also be used if no storage device is available for storing log state.

39.13.3 The ”userspace” log type

This log type simply provides a way to export the log API to userspace, so log implementations can be done there. This is done by forwarding most logging requests to userspace, where a daemon receives and processes the request.

The structure used for communication between kernel and userspace are located in include/linux/dm-log-userspace.h. Due to the frequency, diversity, and 2-way communication nature of the exchanges between kernel and userspace, 'connector' is used as the interface for communication.

There are currently two userspace log implementations that leverage this framework - ”clustered-disk” and ”clustered-core”. These implementations provide a cluster-coherent log for shared-storage. Device-mapper mirroring can be used in a shared-storage environment when the cluster log implementations are employed.

39.14 dm-queue-length

dm-queue-length is a path selector module for device-mapper targets, which selects a path with the least number of in-flight I/Os. The path selector name is 'queue-length'.

Table parameters for each path: [<repeat_count>]

```
<repeat_count>: The number of I/Os to dispatch using the selected
                path before switching to the next path.
                If not given, internal default is used. To check
                the default value, see the activated table.
```

Status for each path: <status> <fail-count> <in-flight>

```
<status>: 'A' if the path is active, 'F' if the path is failed.
<fail-count>: The number of path failures.
<in-flight>: The number of in-flight I/Os on the path.
```

39.14.1 Algorithm

dm-queue-length increments/decrements 'in-flight' when an I/O is dispatched/completed respectively. dm-queue-length selects a path with the minimum 'in-flight'.

39.14.2 Examples

In case that 2 paths (sda and sdb) are used with repeat_count == 128.

```
# echo "0 10 multipath 0 0 1 1 queue-length 0 2 1 8:0 128 8:16 128" \
  dmsetup create test
#
# dmsetup table
test: 0 10 multipath 0 0 1 1 queue-length 0 2 1 8:0 128 8:16 128
#
# dmsetup status
test: 0 10 multipath 2 0 0 0 1 1 E 0 2 1 8:0 A 0 0 8:16 A 0 0
```

39.15 dm-raid

The device-mapper RAID (dm-raid) target provides a bridge from DM to MD. It allows the MD RAID drivers to be accessed using a device-mapper interface.

39.15.1 Mapping Table Interface

The target is named ”raid” and it accepts the following parameters:

```
<raid_type> <#raid_params> <raid_params> \
  <#raid_devs> <metadata_dev0> <dev0> [.. <metadata_devN> <devN>]
```

<raid_type>:

raid0	RAID0 striping (no resilience)
raid1	RAID1 mirroring
raid4	RAID4 with dedicated last parity disk
raid5_n	RAID5 with dedicated last parity disk supporting takeover Same as raid4 • Transitory layout
raid5_la	RAID5 left asymmetric • rotating parity 0 with data continuation
raid5_ra	RAID5 right asymmetric • rotating parity N with data continuation
raid5_ls	RAID5 left symmetric • rotating parity 0 with data restart
raid5_rs	RAID5 right symmetric • rotating parity N with data restart
raid6_zr	RAID6 zero restart • rotating parity zero (left-to-right) with data restart
raid6_nr	RAID6 N restart • rotating parity N (right-to-left) with data restart
raid6_nc	RAID6 N continue • rotating parity N (right-to-left) with data continuation
raid6_n_6	RAID6 with dedicate parity disks • parity and Q-syndrome on the last 2 disks; layout for takeover from/to raid4/raid5_n
raid6_la_6	Same as ”raid_la” plus dedicated last Q-syndrome disk • layout for takeover from raid5_la from/to raid6
raid6_ra_6	Same as ”raid5_ra” dedicated last Q-syndrome disk • layout for takeover from raid5_ra from/to raid6
raid6_ls_6	Same as ”raid5_ls” dedicated last Q-syndrome disk • layout for takeover from raid5_ls from/to raid6
raid6_rs_6	Same as ”raid5_rs” dedicated last Q-syndrome disk • layout for takeover from raid5_rs

Reference: Chapter 4 of https://www.snia.org/sites/default/files/SNIA_DDF_Technical_Position_v2.0.pdf

<#raid_params>: The number of parameters that follow.

<raid_params> consists of

Mandatory parameters:

<chunk_size>:
Chunk size in sectors. This parameter is often known as ”stripe size”. It is the only mandatory parameter and is placed first.

followed by optional parameters (in any order):

[sync|nosync]
Force or prevent RAID initialization.

[rebuild <idx>]
Rebuild drive number 'idx' (first drive is 0).

[daemon_sleep <ms>]
Interval between runs of the bitmap daemon that clear bits. A longer interval means less bitmap I/O but resyncing after a failure is likely to take longer.

[min_recovery_rate <kB/sec/disk>]
Throttle RAID initialization

[max_recovery_rate <kB/sec/disk>]
Throttle RAID initialization

[write_mostly <idx>]
Mark drive index 'idx' write-mostly.

[max_write_behind <sectors>]
See '--write-behind=' (man mdadm)

[stripe_cache <sectors>]
Stripe cache size (RAID 4/5/6 only)

[region_size <sectors>]
The region_size multiplied by the number of regions is the logical size of the array. The bitmap records the device synchronisation state for each region.

[raid10_copies <# copies>], [raid10_format <near|far|offset>]
These two options are used to alter the default layout of a RAID10 configuration. The number of copies is can be specified, but the default is 2. There are also three variations to how the copies are laid down - the default is ”near”. Near copies are what most people think of with respect to mirroring. If these options are left unspecified, or 'raid10_copies 2' and/or 'raid10_format near' are given, then the layouts for 2, 3 and 4 devices are:

2 drives	3 drives	4 drives
A1 A1	A1 A1 A2	A1 A1 A2 A2
A2 A2	A2 A3 A3	A3 A3 A4 A4
A3 A3	A4 A4 A5	A5 A5 A6 A6
A4 A4	A5 A6 A6	A7 A7 A8 A8

The 2-device layout is equivalent 2-way RAID1. The 4-device layout is what a traditional RAID10 would look like. The 3-device layout is what might be called a 'RAID1E - Integrated Adjacent Stripe Mirroring'.

If 'raid10_copies 2' and 'raid10_format far', then the layouts for 2, 3 and 4 devices are:

2 drives	3 drives	4 drives
A1 A2	A1 A2 A3	A1 A2 A3 A4
A3 A4	A4 A5 A6	A5 A6 A7 A8
A5 A6	A7 A8 A9	A9 A10 A11 A12
A2 A1	A3 A1 A2	A2 A1 A4 A3
A4 A3	A6 A4 A5	A6 A5 A8 A7
A6 A5	A9 A7 A8	A10 A9 A12 A11

If 'raid10_copies 2' and 'raid10_format offset', then the layouts for 2, 3 and 4 devices are:

2 drives	3 drives	4 drives
A1 A2	A1 A2 A3	A1 A2 A3 A4
A2 A1	A3 A1 A2	A2 A1 A4 A3
A3 A4	A4 A5 A6	A5 A6 A7 A8
A4 A3	A6 A4 A5	A6 A5 A8 A7
A5 A6	A7 A8 A9	A9 A10 A11 A12
A6 A5	A9 A7 A8	A10 A9 A12 A11

Here we see layouts closely akin to 'RAID1E - Integrated Offset Stripe Mirroring'.

[delta_disks <N>]
The delta_disks option value (-251 < N < +251) triggers device removal (negative value) or device addition (positive value) to any reshape supporting raid levels 4/5/6 and 10. RAID levels 4/5/6 allow for addition of devices (metadata and data device tuple), raid10_near and raid10_offset only allow for device addition. raid10_far does not support any reshaping at all. A minimum of devices have to be kept to enforce resilience, which is 3 devices for raid4/5 and 4 devices for raid6.

[data_offset <sectors>]
This option value defines the offset into each data device where the data starts. This is used to provide out-of-place reshaping space to avoid writing over data while changing the layout of stripes, hence an interruption/crash may happen at any time without the risk of losing data. E.g. when adding devices to an existing raid set during forward reshaping, the out-of-place space will be allocated at the beginning of each raid device. The kernel raid4/5/6/10 MD personalities supporting such device addition will read the data from the existing first stripes (those with smaller number of stripes) starting at data_offset to fill up a new stripe with the larger number of stripes, calculate

the redundancy blocks (CRC/Q-syndrome) and write that new stripe to offset 0. Same will be applied to all N-1 other new stripes. This out-of-place scheme is used to change the RAID type (i.e. the allocation algorithm) as well, e.g. changing from raid5_ls to raid5_n.

[journal_dev <dev>]
This option adds a journal device to raid4/5/6 raid sets and uses it to close the 'write hole' caused by the non-atomic updates to the component devices which can cause data loss during recovery. The journal device is used as writethrough thus causing writes to be throttled versus non-journaled raid4/5/6 sets. Takeover/reshape is not possible with a raid4/5/6 journal device; it has to be deconfigured before requesting these.

[journal_mode <mode>]
This option sets the caching mode on journaled raid4/5/6 raid sets (see 'journal_dev <dev>' above) to 'writethrough' or 'writeback'. If 'writeback' is selected the journal device has to be resilient and must not suffer from the 'write hole' problem itself (e.g. use raid1 or raid10) to avoid a single point of failure.

<#raid_devs>: The number of devices composing the array.
Each device consists of two entries. The first is the device containing the metadata (if any); the second is the one containing the data. A Maximum of 64 metadata/data device entries are supported up to target version 1.8.0. 1.9.0 supports up to 253 which is enforced by the used MD kernel runtime.

If a drive has failed or is missing at creation time, a '-' can be given for both the metadata and data drives for a given position.

39.15.2 Example Tables

```
# RAID4 - 4 data drives, 1 parity (no metadata devices)
# No metadata devices specified to hold superblock/bitmap info
# Chunk size of 1MiB
# (Lines separated for easy reading)

0 1960893648 raid \
        raid4 1 2048 \
        5 - 8:17 - 8:33 - 8:49 - 8:65 - 8:81

# RAID4 - 4 data drives, 1 parity (with metadata devices)
# Chunk size of 1MiB, force RAID initialization,
#       min recovery rate at 20 kiB/sec/disk

0 1960893648 raid \
        raid4 4 2048 sync min_recovery_rate 20 \
        5 8:17 8:18 8:33 8:34 8:49 8:50 8:65 8:66 8:81 8:82
```

39.15.3 Status Output

'dmsetup table' displays the table used to construct the mapping. The optional parameters are always printed in the order listed above with ”sync” or ”nosync” always output ahead of the other arguments, regardless of the order used when originally loading the table. Arguments that can be repeated are ordered by value.

'dmsetup status' yields information on the state and health of the array. The output is as follows (normally a single line, but expanded here for clarity):

```
1: <s> <l> raid \
2:      <raid_type> <#devices> <health_chars> \
3:      <sync_ratio> <sync_action> <mismatch_cnt>
```

Line 1 is the standard output produced by device-mapper.

Line 2 & 3 are produced by the raid target and are best explained by example:

```
0 1960893648 raid raid4 5 AAAAA 2/490221568 init 0
```

Here we can see the RAID type is raid4, there are 5 devices - all of which are 'A'live, and the array is 2/490221568 complete with its initial recovery. Here is a fuller description of the individual fields:

<raid_type>	Same as the <raid_type> used to create the array.
<health_chars>	One char for each device, indicating: • 'A' = alive and in-sync • 'a' = alive but not in-sync • 'D' = dead/failed.
<sync_ratio>	The ratio indicating how much of the array has undergone the process described by 'sync_action'. If the 'sync_action' is ”check” or ”repair”, then the process of ”resync” or ”recover” can be considered complete.
<sync_action>	One of the following possible states: **idle** • No synchronization action is being performed. **frozen** • The current action has been halted. **resync** • Array is undergoing its initial synchronization or is resynchronizing after an unclean shutdown (possibly aided by a bitmap). **recover** • A device in the array is being rebuilt or replaced. **check** • A user-initiated full check of the array is being performed. All blocks are read and checked for consistency. The number of discrepancies found are recorded in <mismatch_cnt>. No changes are made to the array by this action. **repair** • The same as ”check”, but discrepancies are corrected. **reshape** • The array is undergoing a reshape.
<mismatch_cnt>	The number of discrepancies found between mirror copies in RAID1/10 or wrong parity values found in RAID4/5/6. This value is valid only after a ”check” of the array is performed. A healthy array has a 'mismatch_cnt' of 0.
<data_offset>	The current data offset to the start of the user data on each component device of the raid set (see the respective raid parameter to support out-of-place reshaping).
<journal_char>	'A' = active in the raid journal device

39.15.4 Message Interface

The dm-raid target will accept certain actions through the 'message' interface. ('man dmsetup' for more information on the message interface.) These actions include:

”idle”	Halt the current sync action.
”frozen”	Freeze the current sync action.
”resync”	Initiate/continue a resync.
”recover”	Initiate/continue a recover process.
”check”	Initiate a check (i.e. a ”scrub”) of the array.
”repair”	Initiate a repair of the array.

39.15.5 Discard Support

The implementation of discard support among hardware vendors varies. When a block is discarded, some storage devices will return zeroes when the block is read. These devices set the 'discard_zeroes_data' attribute. Other devices will return random data. Confusingly, some devices that advertise 'discard_zeroes_data' will not reliably return zeroes when discarded blocks are read! Since RAID 4/5/6 uses blocks from a number of devices to calculate parity blocks and (for performance reasons) relies on 'discard_zeroes_data' being reliable, it is important that the devices be consistent. Blocks may be discarded in the middle of a RAID 4/5/6 stripe and if subsequent read results are not consistent, the parity blocks may be calculated differently at any time; making the parity blocks useless for redundancy. It is important to understand how your hardware behaves with discards if you are going to enable discards with RAID 4/5/6.

Since the behavior of storage devices is unreliable in this respect, even when reporting 'discard_zeroes_data', by default RAID 4/5/6 discard support is disabled -- this ensures data integrity at the expense of losing some performance.

Storage devices that properly support 'discard_zeroes_data' are increasingly whitelisted in the kernel and can thus be trusted.

For trusted devices, the following dm-raid module parameter can be set to safely enable discard support for RAID 4/5/6:

> 'devices_handle_discards_safely'

39.15.6 Version History

```
1.0.0  Initial version.  Support for RAID 4/5/6
1.1.0  Added support for RAID 1
1.2.0  Handle creation of arrays that contain failed devices.
1.3.0  Added support for RAID 10
1.3.1  Allow device replacement/rebuild for RAID 10
1.3.2  Fix/improve redundancy checking for RAID10
1.4.0  Non-functional change.  Removes arg from mapping function.
1.4.1  RAID10 fix redundancy validation checks (commit 55ebbb5).
1.4.2  Add RAID10 "far" and "offset" algorithm support.
1.5.0  Add message interface to allow manipulation of the sync_action.
       New status (STATUSTYPE_INFO) fields: sync_action and mismatch_cnt.
1.5.1  Add ability to restore transiently failed devices on resume.
```

```
1.5.2  'mismatch_cnt' is zero unless [last_]sync_action is "check".
1.6.0  Add discard support (and devices_handle_discard_safely module param).
1.7.0  Add support for MD RAID0 mappings.
1.8.0  Explicitly check for compatible flags in the superblock metadata
       and reject to start the raid set if any are set by a newer
       target version, thus avoiding data corruption on a raid set
       with a reshape in progress.
1.9.0  Add support for RAID level takeover/reshape/region size
       and set size reduction.
1.9.1  Fix activation of existing RAID 4/10 mapped devices
1.9.2  Don't emit '- -' on the status table line in case the constructor
       fails reading a superblock. Correctly emit 'maj:min1 maj:min2' and
       'D' on the status line.  If '- -' is passed into the constructor, emit
       '- -' on the table line and '-' as the status line health character.
1.10.0 Add support for raid4/5/6 journal device
1.10.1 Fix data corruption on reshape request
1.11.0 Fix table line argument order
       (wrong raid10_copies/raid10_format sequence)
1.11.1 Add raid4/5/6 journal write-back support via journal_mode option
1.12.1 Fix for MD deadlock between mddev_suspend() and md_write_start()␣
↪available
1.13.0 Fix dev_health status at end of "recover" (was 'a', now 'A')
1.13.1 Fix deadlock caused by early md_stop_writes().  Also fix size an
       state races.
1.13.2 Fix raid redundancy validation and avoid keeping raid set frozen
1.14.0 Fix reshape race on small devices.  Fix stripe adding reshape
       deadlock/potential data corruption.  Update superblock when
       specific devices are requested via rebuild.  Fix RAID leg
       rebuild errors.
1.15.0 Fix size extensions not being synchronized in case of new MD bitmap
       pages allocated;  also fix those not occurring after previous reductions
1.15.1 Fix argument count and arguments for rebuild/write_mostly/journal_
↪(dev|mode)
       on the status line.
```

39.16 dm-service-time

dm-service-time is a path selector module for device-mapper targets, which selects a path with the shortest estimated service time for the incoming I/O.

The service time for each path is estimated by dividing the total size of in-flight I/Os on a path with the performance value of the path. The performance value is a relative throughput value among all paths in a path-group, and it can be specified as a table argument.

The path selector name is 'service-time'.

Table parameters for each path:

[<repeat_count> [<relative_throughput>]]

<repeat_count>:

The number of I/Os to dispatch using the selected path before switching to the next path. If not given, internal default is used. To check the default value, see the activated table.

<relative_throughput>:
The relative throughput value of the path among all paths in the path-group. The valid range is 0-100. If not given, minimum value '1' is used. If '0' is given, the path isn't selected while other paths having a positive value are available.

Status for each path:

<status> <fail-count> <in-flight-size> <relative_throughput>

<status>:
'A' if the path is active, 'F' if the path is failed.

<fail-count>:
The number of path failures.

<in-flight-size>:
The size of in-flight I/Os on the path.

<relative_throughput>:
The relative throughput value of the path among all paths in the path-group.

39.16.1 Algorithm

dm-service-time adds the I/O size to 'in-flight-size' when the I/O is dispatched and subtracts when completed. Basically, dm-service-time selects a path having minimum service time which is calculated by:

```
('in-flight-size' + 'size-of-incoming-io') / 'relative_throughput'
```

However, some optimizations below are used to reduce the calculation as much as possible.

1. If the paths have the same 'relative_throughput', skip the division and just compare the 'in-flight-size'.
2. If the paths have the same 'in-flight-size', skip the division and just compare the 'relative_throughput'.
3. If some paths have non-zero 'relative_throughput' and others have zero 'relative_throughput', ignore those paths with zero 'relative_throughput'.

If such optimizations can't be applied, calculate service time, and compare service time. If calculated service time is equal, the path having maximum 'relative_throughput' may be better. So compare 'relative_throughput' then.

39.16.2 Examples

In case that 2 paths (sda and sdb) are used with repeat_count == 128 and sda has an average throughput 1GB/s and sdb has 4GB/s, 'relative_throughput' value may be '1' for sda and '4' for sdb:

```
# echo "0 10 multipath 0 0 1 1 service-time 0 2 2 8:0 128 1 8:16 128 4" \
  dmsetup create test
#
# dmsetup table
test: 0 10 multipath 0 0 1 1 service-time 0 2 2 8:0 128 1 8:16 128 4
#
# dmsetup status
test: 0 10 multipath 2 0 0 0 1 1 E 0 2 2 8:0 A 0 0 1 8:16 A 0 0 4
```

Or '2' for sda and '8' for sdb would be also true:

```
# echo "0 10 multipath 0 0 1 1 service-time 0 2 2 8:0 128 2 8:16 128 8" \
  dmsetup create test
#
# dmsetup table
test: 0 10 multipath 0 0 1 1 service-time 0 2 2 8:0 128 2 8:16 128 8
#
# dmsetup status
test: 0 10 multipath 2 0 0 0 1 1 E 0 2 2 8:0 A 0 0 2 8:16 A 0 0 8
```

39.17 device-mapper uevent

The device-mapper uevent code adds the capability to device-mapper to create and send kobject uevents (uevents). Previously device-mapper events were only available through the ioctl interface. The advantage of the uevents interface is the event contains environment attributes providing increased context for the event avoiding the need to query the state of the device-mapper device after the event is received.

There are two functions currently for device-mapper events. The first function listed creates the event and the second function sends the event(s):

```
void dm_path_uevent(enum dm_uevent_type event_type, struct dm_target *ti,
                    const char *path, unsigned nr_valid_paths)

void dm_send_uevents(struct list_head *events, struct kobject *kobj)
```

The variables added to the uevent environment are:

39.17.1 Variable Name: DM_TARGET

Uevent Action(s)
KOBJ_CHANGE

Type
string

Description
Value
Name of device-mapper target that generated the event.

39.17.2 Variable Name: DM_ACTION

Uevent Action(s)
KOBJ_CHANGE

Type
string

Description
Value
Device-mapper specific action that caused the uevent action. PATH_FAILED - A path has failed; PATH_REINSTATED - A path has been reinstated.

39.17.3 Variable Name: DM_SEQNUM

Uevent Action(s)
KOBJ_CHANGE

Type
unsigned integer

Description
A sequence number for this specific device-mapper device.

Value
Valid unsigned integer range.

39.17.4 Variable Name: DM_PATH

Uevent Action(s)
KOBJ_CHANGE

Type
string

Description
Major and minor number of the path device pertaining to this event.

Value
Path name in the form of ”Major:Minor”

39.17.5 Variable Name: DM_NR_VALID_PATHS

Uevent Action(s)
KOBJ_CHANGE

Type
unsigned integer

Description
Value
Valid unsigned integer range.

39.17.6 Variable Name: DM_NAME

Uevent Action(s)
KOBJ_CHANGE

Type
string

Description
Name of the device-mapper device.

Value
Name

39.17.7 Variable Name: DM_UUID

Uevent Action(s)
KOBJ_CHANGE

Type
string

Description
UUID of the device-mapper device.

Value
UUID. (Empty string if there isn't one.)

An example of the uevents generated as captured by udevmonitor is shown below

1.) Path failure:

```
UEVENT[1192521009.711215] change@/block/dm-3
ACTION=change
DEVPATH=/block/dm-3
SUBSYSTEM=block
DM_TARGET=multipath
DM_ACTION=PATH_FAILED
DM_SEQNUM=1
DM_PATH=8:32
DM_NR_VALID_PATHS=0
DM_NAME=mpath2
DM_UUID=mpath-35333333000002328
```

```
MINOR=3
MAJOR=253
SEQNUM=1130
```

2.) Path reinstate:

```
UEVENT[1192521132.989927] change@/block/dm-3
ACTION=change
DEVPATH=/block/dm-3
SUBSYSTEM=block
DM_TARGET=multipath
DM_ACTION=PATH_REINSTATED
DM_SEQNUM=2
DM_PATH=8:32
DM_NR_VALID_PATHS=1
DM_NAME=mpath2
DM_UUID=mpath-35333333000002328
MINOR=3
MAJOR=253
SEQNUM=1131
```

39.18 dm-zoned

The dm-zoned device mapper target exposes a zoned block device (ZBC and ZAC compliant devices) as a regular block device without any write pattern constraints. In effect, it implements a drive-managed zoned block device which hides from the user (a file system or an application doing raw block device accesses) the sequential write constraints of host-managed zoned block devices and can mitigate the potential device-side performance degradation due to excessive random writes on host-aware zoned block devices.

For a more detailed description of the zoned block device models and their constraints see (for SCSI devices):

https://www.t10.org/drafts.htm#ZBC_Family

and (for ATA devices):

http://www.t13.org/Documents/UploadedDocuments/docs2015/di537r05-Zoned_Device_ATA_Command_Set_ZAC.pdf

The dm-zoned implementation is simple and minimizes system overhead (CPU and memory usage as well as storage capacity loss). For a 10TB host-managed disk with 256 MB zones, dm-zoned memory usage per disk instance is at most 4.5 MB and as little as 5 zones will be used internally for storing metadata and performing reclaim operations.

dm-zoned target devices are formatted and checked using the dmzadm utility available at:

https://github.com/hgst/dm-zoned-tools

39.18.1 Algorithm

dm-zoned implements an on-disk buffering scheme to handle non-sequential write accesses to the sequential zones of a zoned block device. Conventional zones are used for caching as well as for storing internal metadata. It can also use a regular block device together with the zoned block device; in that case the regular block device will be split logically in zones with the same size as the zoned block device. These zones will be placed in front of the zones from the zoned block device and will be handled just like conventional zones.

The zones of the device(s) are separated into 2 types:

1) Metadata zones: these are conventional zones used to store metadata. Metadata zones are not reported as usable capacity to the user.

2) Data zones: all remaining zones, the vast majority of which will be sequential zones used exclusively to store user data. The conventional zones of the device may be used also for buffering user random writes. Data in these zones may be directly mapped to the conventional zone, but later moved to a sequential zone so that the conventional zone can be reused for buffering incoming random writes.

dm-zoned exposes a logical device with a sector size of 4096 bytes, irrespective of the physical sector size of the backend zoned block device being used. This allows reducing the amount of metadata needed to manage valid blocks (blocks written).

The on-disk metadata format is as follows:

1) The first block of the first conventional zone found contains the super block which describes the on disk amount and position of metadata blocks.

2) Following the super block, a set of blocks is used to describe the mapping of the logical device blocks. The mapping is done per chunk of blocks, with the chunk size equal to the zoned block device size. The mapping table is indexed by chunk number and each mapping entry indicates the zone number of the device storing the chunk of data. Each mapping entry may also indicate if the zone number of a conventional zone used to buffer random modification to the data zone.

3) A set of blocks used to store bitmaps indicating the validity of blocks in the data zones follows the mapping table. A valid block is defined as a block that was written and not discarded. For a buffered data chunk, a block is always valid only in the data zone mapping the chunk or in the buffer zone of the chunk.

For a logical chunk mapped to a conventional zone, all write operations are processed by directly writing to the zone. If the mapping zone is a sequential zone, the write operation is processed directly only if the write offset within the logical chunk is equal to the write pointer offset within of the sequential data zone (i.e. the write operation is aligned on the zone write pointer). Otherwise, write operations are processed indirectly using a buffer zone. In that case, an unused conventional zone is allocated and assigned to the chunk being accessed. Writing a block to the buffer zone of a chunk will automatically invalidate the same block in the sequential zone mapping the chunk. If all blocks of the sequential zone become invalid, the zone is freed and the chunk buffer zone becomes the primary zone mapping the chunk, resulting in native random write performance similar to a regular block device.

Read operations are processed according to the block validity information provided by the bitmaps. Valid blocks are read either from the sequential zone mapping a chunk, or if the chunk is buffered, from the buffer zone assigned. If the accessed chunk has no mapping, or the accessed blocks are invalid, the read buffer is zeroed and the read operation terminated.

After some time, the limited number of conventional zones available may be exhausted (all used to map chunks or buffer sequential zones) and unaligned writes to unbuffered chunks become impossible. To avoid this situation, a reclaim process regularly scans used conventional zones and tries to reclaim the least recently used zones by copying the valid blocks of the buffer zone to a free sequential zone. Once the copy completes, the chunk mapping is updated to point to the sequential zone and the buffer zone freed for reuse.

39.18.2 Metadata Protection

To protect metadata against corruption in case of sudden power loss or system crash, 2 sets of metadata zones are used. One set, the primary set, is used as the main metadata region, while the secondary set is used as a staging area. Modified metadata is first written to the secondary set and validated by updating the super block in the secondary set, a generation counter is used to indicate that this set contains the newest metadata. Once this operation completes, in place of metadata block updates can be done in the primary metadata set. This ensures that one of the set is always consistent (all modifications committed or none at all). Flush operations are used as a commit point. Upon reception of a flush request, metadata modification activity is temporarily blocked (for both incoming BIO processing and reclaim process) and all dirty metadata blocks are staged and updated. Normal operation is then resumed. Flushing metadata thus only temporarily delays write and discard requests. Read requests can be processed concurrently while metadata flush is being executed.

If a regular device is used in conjunction with the zoned block device, a third set of metadata (without the zone bitmaps) is written to the start of the zoned block device. This metadata has a generation counter of '0' and will never be updated during normal operation; it just serves for identification purposes. The first and second copy of the metadata are located at the start of the regular block device.

39.18.3 Usage

A zoned block device must first be formatted using the dmzadm tool. This will analyze the device zone configuration, determine where to place the metadata sets on the device and initialize the metadata sets.

Ex:

```
dmzadm --format /dev/sdxx
```

If two drives are to be used, both devices must be specified, with the regular block device as the first device.

Ex:

```
dmzadm --format /dev/sdxx /dev/sdyy
```

Formatted device(s) can be started with the dmzadm utility, too.:

Ex:

```
dmzadm --start /dev/sdxx /dev/sdyy
```

Information about the internal layout and current usage of the zones can be obtained with the 'status' callback from dmsetup:

Ex:

```
dmsetup status /dev/dm-X
```

will return a line

> 0 <size> zoned <nr_zones> zones <nr_unmap_rnd>/<nr_rnd> random <nr_unmap_seq>/<nr_seq> sequential

where <nr_zones> is the total number of zones, <nr_unmap_rnd> is the number of unmapped (ie free) random zones, <nr_rnd> the total number of zones, <nr_unmap_seq> the number of unmapped sequential zones, and <nr_seq> the total number of sequential zones.

Normally the reclaim process will be started once there are less than 50 percent free random zones. In order to start the reclaim process manually even before reaching this threshold the 'dmsetup message' function can be used:

Ex:

```
dmsetup message /dev/dm-X 0 reclaim
```

will start the reclaim process and random zones will be moved to sequential zones.

39.19 dm-era

39.19.1 Introduction

dm-era is a target that behaves similar to the linear target. In addition it keeps track of which blocks were written within a user defined period of time called an 'era'. Each era target instance maintains the current era as a monotonically increasing 32-bit counter.

Use cases include tracking changed blocks for backup software, and partially invalidating the contents of a cache to restore cache coherency after rolling back a vendor snapshot.

39.19.2 Constructor

era <metadata dev> <origin dev> <block size>

metadata dev	fast device holding the persistent metadata
origin dev	device holding data blocks that may change
block size	block size of origin data device, granularity that is tracked by the target

39.19.3 Messages

None of the dm messages take any arguments.

checkpoint

Possibly move to a new era. You shouldn't assume the era has incremented. After sending this message, you should check the current era via the status line.

take_metadata_snap

Create a clone of the metadata, to allow a userland process to read it.

drop_metadata_snap

Drop the metadata snapshot.

39.19.4 Status

<metadata block size> <#used metadata blocks>/<#total metadata blocks> <current era> <held metadata root | '-'>

metadata block size	Fixed block size for each metadata block in sectors
#used metadata blocks	Number of metadata blocks used
#total metadata blocks	Total number of metadata blocks
current era	The current era
held metadata root	The location, in blocks, of the metadata root that has been 'held' for userspace read access. '-' indicates there is no held root

39.19.5 Detailed use case

The scenario of invalidating a cache when rolling back a vendor snapshot was the primary use case when developing this target:

Taking a vendor snapshot

- Send a checkpoint message to the era target
- Make a note of the current era in its status line
- Take vendor snapshot (the era and snapshot should be forever associated now).

Rolling back to an vendor snapshot

- Cache enters passthrough mode (see: dm-cache's docs in cache.txt)
- Rollback vendor storage
- Take metadata snapshot
- Ascertain which blocks have been written since the snapshot was taken by checking each block's era
- Invalidate those blocks in the caching software
- Cache returns to writeback/writethrough mode

39.19.6 Memory usage

The target uses a bitset to record writes in the current era. It also has a spare bitset ready for switching over to a new era. Other than that it uses a few 4k blocks for updating metadata:

```
(4 * nr_blocks) bytes + buffers
```

39.19.7 Resilience

Metadata is updated on disk before a write to a previously unwritten block is performed. As such dm-era should not be effected by a hard crash such as power failure.

39.19.8 Userland tools

Userland tools are found in the increasingly poorly named thin-provisioning-tools project:

https://github.com/jthornber/thin-provisioning-tools

39.20 kcopyd

Kcopyd provides the ability to copy a range of sectors from one block-device to one or more other block-devices, with an asynchronous completion notification. It is used by dm-snapshot and dm-mirror.

Users of kcopyd must first create a client and indicate how many memory pages to set aside for their copy jobs. This is done with a call to kcopyd_client_create():

```
int kcopyd_client_create(unsigned int num_pages,
                         struct kcopyd_client **result);
```

To start a copy job, the user must set up io_region structures to describe the source and destinations of the copy. Each io_region indicates a block-device along with the starting sector and size of the region. The source of the copy is given as one io_region structure, and the destinations of the copy are given as an array of io_region structures:

```
struct io_region {
   struct block_device *bdev;
   sector_t sector;
   sector_t count;
};
```

To start the copy, the user calls kcopyd_copy(), passing in the client pointer, pointers to the source and destination io_regions, the name of a completion callback routine, and a pointer to some context data for the copy:

```
int kcopyd_copy(struct kcopyd_client *kc, struct io_region *from,
                unsigned int num_dests, struct io_region *dests,
                unsigned int flags, kcopyd_notify_fn fn, void *context);

typedef void (*kcopyd_notify_fn)(int read_err, unsigned int write_err,
                                 void *context);
```

When the copy completes, kcopyd will call the user's completion routine, passing back the user's context pointer. It will also indicate if a read or write error occurred during the copy.

When a user is done with all their copy jobs, they should call kcopyd_client_destroy() to delete the kcopyd client, which will release the associated memory pages:

```
void kcopyd_client_destroy(struct kcopyd_client *kc);
```

39.21 dm-linear

Device-Mapper's ”linear” target maps a linear range of the Device-Mapper device onto a linear range of another device. This is the basic building block of logical volume managers.

Parameters: <dev path> <offset>

<dev path>:
Full pathname to the underlying block-device, or a ”major:minor” device-number.

<offset>:
Starting sector within the device.

39.21.1 Example scripts

```
#!/bin/sh
# Create an identity mapping for a device
echo "0 `blockdev --getsz $1` linear $1 0" | dmsetup create identity
```

```
#!/bin/sh
# Join 2 devices together
size1=`blockdev --getsz $1`
size2=`blockdev --getsz $2`
echo "0 $size1 linear $1 0
$size1 $size2 linear $2 0" | dmsetup create joined
```

```
#!/usr/bin/perl -w
# Split a device into 4M chunks and then join them together in reverse order.

my $name = "reverse";
my $extent_size = 4 * 1024 * 2;
my $dev = $ARGV[0];
my $table = "";
my $count = 0;

if (!defined($dev)) {
        die("Please specify a device.\n");
}

my $dev_size = `blockdev --getsz $dev`;
my $extents = int($dev_size / $extent_size) -
              (($dev_size % $extent_size) ? 1 : 0);

while ($extents > 0) {
        my $this_start = $count * $extent_size;
        $extents--;
        $count++;
        my $this_offset = $extents * $extent_size;

        $table .= "$this_start $extent_size linear $dev $this_offset\n";
}

`echo \"$table\" | dmsetup create $name`;
```

39.22 dm-log-writes

This target takes 2 devices, one to pass all IO to normally, and one to log all of the write operations to. This is intended for file system developers wishing to verify the integrity of metadata or data as the file system is written to. There is a log_write_entry written for every WRITE request and the target is able to take arbitrary data from userspace to insert into the log. The data that is in the WRITE requests is copied into the log to make the replay happen exactly as it happened originally.

39.22.1 Log Ordering

We log things in order of completion once we are sure the write is no longer in cache. This means that normal WRITE requests are not actually logged until the next REQ_PREFLUSH request. This is to make it easier for userspace to replay the log in a way that correlates to what is on disk and not what is in cache, to make it easier to detect improper waiting/flushing.

This works by attaching all WRITE requests to a list once the write completes. Once we see a REQ_PREFLUSH request we splice this list onto the request and once the FLUSH request completes we log all of the WRITEs and then the FLUSH. Only completed WRITEs, at the time the REQ_PREFLUSH is issued, are added in order to simulate the worst case scenario with regard to power failures. Consider the following example (W means write, C means complete):

W1,W2,W3,C3,C2,Wflush,C1,Cflush

The log would show the following:

W3,W2,flush,W1....

Again this is to simulate what is actually on disk, this allows us to detect cases where a power failure at a particular point in time would create an inconsistent file system.

Any REQ_FUA requests bypass this flushing mechanism and are logged as soon as they complete as those requests will obviously bypass the device cache.

Any REQ_OP_DISCARD requests are treated like WRITE requests. Otherwise we would have all the DISCARD requests, and then the WRITE requests and then the FLUSH request. Consider the following example:

WRITE block 1, DISCARD block 1, FLUSH

If we logged DISCARD when it completed, the replay would look like this:

DISCARD 1, WRITE 1, FLUSH

which isn't quite what happened and wouldn't be caught during the log replay.

39.22.2 Target interface

i) Constructor

log-writes <dev_path> <log_dev_path>

dev_path	Device that all of the IO will go to normally.
log_dev_path	Device where the log entries are written to.

ii) Status

<#logged entries> <highest allocated sector>

#logged entries	Number of logged entries
highest allocated sector	Highest allocated sector

iii) Messages

mark <description>

You can use a dmsetup message to set an arbitrary mark in a log. For example say you want to fsck a file system after every write, but first you need to replay up to the mkfs to make sure we're fsck'ing something reasonable, you would do something like this:

```
mkfs.btrfs -f /dev/mapper/log
dmsetup message log 0 mark mkfs
<run test>
```

This would allow you to replay the log up to the mkfs mark and then replay from that point on doing the fsck check in the interval that you want.

Every log has a mark at the end labeled ”dm-log-writes-end”.

39.22.3 Userspace component

There is a userspace tool that will replay the log for you in various ways. It can be found here: https://github.com/josefbacik/log-writes

39.22.4 Example usage

Say you want to test fsync on your file system. You would do something like this:

```
TABLE="0 $(blockdev --getsz /dev/sdb) log-writes /dev/sdb /dev/sdc"
dmsetup create log --table "$TABLE"
mkfs.btrfs -f /dev/mapper/log
dmsetup message log 0 mark mkfs

mount /dev/mapper/log /mnt/btrfs-test
<some test that does fsync at the end>
dmsetup message log 0 mark fsync
md5sum /mnt/btrfs-test/foo
umount /mnt/btrfs-test

dmsetup remove log
replay-log --log /dev/sdc --replay /dev/sdb --end-mark fsync
mount /dev/sdb /mnt/btrfs-test
md5sum /mnt/btrfs-test/foo
<verify md5sum's are correct>

Another option is to do a complicated file system operation and verify the file
system is consistent during the entire operation.  You could do this with:

TABLE="0 $(blockdev --getsz /dev/sdb) log-writes /dev/sdb /dev/sdc"
dmsetup create log --table "$TABLE"
mkfs.btrfs -f /dev/mapper/log
dmsetup message log 0 mark mkfs

mount /dev/mapper/log /mnt/btrfs-test
<fsstress to dirty the fs>
btrfs filesystem balance /mnt/btrfs-test
umount /mnt/btrfs-test
dmsetup remove log

replay-log --log /dev/sdc --replay /dev/sdb --end-mark mkfs
btrfsck /dev/sdb
replay-log --log /dev/sdc --replay /dev/sdb --start-mark mkfs \
      --fsck "btrfsck /dev/sdb" --check fua
```

And that will replay the log until it sees a FUA request, run the fsck command and if the fsck passes it will replay to the next FUA, until it is completed or the fsck command exists abnormally.

39.23 Persistent data

39.23.1 Introduction

The more-sophisticated device-mapper targets require complex metadata that is managed in kernel. In late 2010 we were seeing that various different targets were rolling their own data structures, for example:

- Mikulas Patocka's multisnap implementation
- Heinz Mauelshagen's thin provisioning target
- Another btree-based caching target posted to dm-devel
- Another multi-snapshot target based on a design of Daniel Phillips

Maintaining these data structures takes a lot of work, so if possible we'd like to reduce the number.

The persistent-data library is an attempt to provide a re-usable framework for people who want to store metadata in device-mapper targets. It's currently used by the thin-provisioning target and an upcoming hierarchical storage target.

39.23.2 Overview

The main documentation is in the header files which can all be found under drivers/md/persistent-data.

The block manager

dm-block-manager.[hc]

This provides access to the data on disk in fixed sized-blocks. There is a read/write locking interface to prevent concurrent accesses, and keep data that is being used in the cache.

Clients of persistent-data are unlikely to use this directly.

The transaction manager

dm-transaction-manager.[hc]

This restricts access to blocks and enforces copy-on-write semantics. The only way you can get hold of a writable block through the transaction manager is by shadowing an existing block (ie. doing copy-on-write) or allocating a fresh one. Shadowing is elided within the same transaction so performance is reasonable. The commit method ensures that all data is flushed before it writes the superblock. On power failure your metadata will be as it was when last committed.

The Space Maps

dm-space-map.h dm-space-map-metadata.[hc] dm-space-map-disk.[hc]

On-disk data structures that keep track of reference counts of blocks. Also acts as the allocator of new blocks. Currently two implementations: a simpler one for managing blocks on a different device (eg. thinly-provisioned data blocks); and one for managing the metadata space. The latter is complicated by the need to store its own data within the space it's managing.

The data structures

dm-btree.[hc] dm-btree-remove.c dm-btree-spine.c dm-btree-internal.h

Currently there is only one data structure, a hierarchical btree. There are plans to add more. For example, something with an array-like interface would see a lot of use.

The btree is 'hierarchical' in that you can define it to be composed of nested btrees, and take multiple keys. For example, the thin-provisioning target uses a btree with two levels of nesting. The first maps a device id to a mapping tree, and that in turn maps a virtual block to a physical block.

Values stored in the btrees can have arbitrary size. Keys are always 64bits, although nesting allows you to use multiple keys.

39.24 Device-mapper snapshot support

Device-mapper allows you, without massive data copying:

- To create snapshots of any block device i.e. mountable, saved states of the block device which are also writable without interfering with the original content;
- To create device ”forks”, i.e. multiple different versions of the same data stream.
- To merge a snapshot of a block device back into the snapshot's origin device.

In the first two cases, dm copies only the chunks of data that get changed and uses a separate copy-on-write (COW) block device for storage.

For snapshot merge the contents of the COW storage are merged back into the origin device.

There are three dm targets available: snapshot, snapshot-origin, and snapshot-merge.

- snapshot-origin <origin>

which will normally have one or more snapshots based on it. Reads will be mapped directly to the backing device. For each write, the original data will be saved in the <COW device> of each snapshot to keep its visible content unchanged, at least until the <COW device> fills up.

- snapshot <origin> <COW device> <persistent?> <chunksize> [<# feature args> [<arg>]*]

A snapshot of the <origin> block device is created. Changed chunks of <chunksize> sectors will be stored on the <COW device>. Writes will only go to the <COW device>. Reads will come from the <COW device> or from <origin> for unchanged data. <COW device> will often be smaller than the origin and if it fills up the snapshot will become useless and be disabled,

returning errors. So it is important to monitor the amount of free space and expand the <COW device> before it fills up.

<persistent?> is P (Persistent) or N (Not persistent - will not survive after reboot). O (Overflow) can be added as a persistent store option to allow userspace to advertise its support for seeing "Overflow" in the snapshot status. So supported store types are "P", "PO" and "N".

The difference between persistent and transient is with transient snapshots less metadata must be saved on disk - they can be kept in memory by the kernel.

When loading or unloading the snapshot target, the corresponding snapshot-origin or snapshot-merge target must be suspended. A failure to suspend the origin target could result in data corruption.

Optional features:

> discard_zeroes_cow - a discard issued to the snapshot device that maps to entire chunks to will zero the corresponding exception(s) in the snapshot's exception store.
>
> discard_passdown_origin - a discard to the snapshot device is passed down to the snapshot-origin's underlying device. This doesn't cause copy-out to the snapshot exception store because the snapshot-origin target is bypassed.
>
> The discard_passdown_origin feature depends on the discard_zeroes_cow feature being enabled.

- snapshot-merge <origin> <COW device> <persistent> <chunksize> [<# feature args> [<arg>]*]

takes the same table arguments as the snapshot target except it only works with persistent snapshots. This target assumes the role of the "snapshot-origin" target and must not be loaded if the "snapshot-origin" is still present for <origin>.

Creates a merging snapshot that takes control of the changed chunks stored in the <COW device> of an existing snapshot, through a handover procedure, and merges these chunks back into the <origin>. Once merging has started (in the background) the <origin> may be opened and the merge will continue while I/O is flowing to it. Changes to the <origin> are deferred until the merging snapshot's corresponding chunk(s) have been merged. Once merging has started the snapshot device, associated with the "snapshot" target, will return -EIO when accessed.

39.24.1 How snapshot is used by LVM2

When you create the first LVM2 snapshot of a volume, four dm devices are used:

1) a device containing the original mapping table of the source volume;
2) a device used as the <COW device>;
3) a "snapshot" device, combining #1 and #2, which is the visible snapshot volume;
4) the "original" volume (which uses the device number used by the original source volume), whose table is replaced by a "snapshot-origin" mapping from device #1.

A fixed naming scheme is used, so with the following commands:

```
lvcreate -L 1G -n base volumeGroup
lvcreate -L 100M --snapshot -n snap volumeGroup/base
```

we'll have this situation (with volumes in above order):

```
# dmsetup table|grep volumeGroup

volumeGroup-base-real: 0 2097152 linear 8:19 384
volumeGroup-snap-cow: 0 204800 linear 8:19 2097536
volumeGroup-snap: 0 2097152 snapshot 254:11 254:12 P 16
volumeGroup-base: 0 2097152 snapshot-origin 254:11

# ls -lL /dev/mapper/volumeGroup-*
brw-------  1 root root 254, 11 29 ago 18:15 /dev/mapper/volumeGroup-base-real
brw-------  1 root root 254, 12 29 ago 18:15 /dev/mapper/volumeGroup-snap-cow
brw-------  1 root root 254, 13 29 ago 18:15 /dev/mapper/volumeGroup-snap
brw-------  1 root root 254, 10 29 ago 18:14 /dev/mapper/volumeGroup-base
```

39.24.2 How snapshot-merge is used by LVM2

A merging snapshot assumes the role of the ”snapshot-origin” while merging. As such the ”snapshot-origin” is replaced with ”snapshot-merge”. The ”-real” device is not changed and the ”-cow” device is renamed to <origin name>-cow to aid LVM2's cleanup of the merging snapshot after it completes. The ”snapshot” that hands over its COW device to the ”snapshot-merge” is deactivated (unless using lvchange --refresh); but if it is left active it will simply return I/O errors.

A snapshot will merge into its origin with the following command:

```
lvconvert --merge volumeGroup/snap
```

we'll now have this situation:

```
# dmsetup table|grep volumeGroup

volumeGroup-base-real: 0 2097152 linear 8:19 384
volumeGroup-base-cow: 0 204800 linear 8:19 2097536
volumeGroup-base: 0 2097152 snapshot-merge 254:11 254:12 P 16

# ls -lL /dev/mapper/volumeGroup-*
brw-------  1 root root 254, 11 29 ago 18:15 /dev/mapper/volumeGroup-base-real
brw-------  1 root root 254, 12 29 ago 18:16 /dev/mapper/volumeGroup-base-cow
brw-------  1 root root 254, 10 29 ago 18:16 /dev/mapper/volumeGroup-base
```

39.24.3 How to determine when a merging is complete

The snapshot-merge and snapshot status lines end with:

<sectors_allocated>/<total_sectors> <metadata_sectors>

Both <sectors_allocated> and <total_sectors> include both data and metadata. During merging, the number of sectors allocated gets smaller and smaller. Merging has finished when the number of sectors holding data is zero, in other words <sectors_allocated> == <metadata_sectors>.

Here is a practical example (using a hybrid of lvm and dmsetup commands):

```
# lvs
  LV      VG          Attr   LSize Origin  Snap%  Move Log Copy%  Convert
  base    volumeGroup owi-a- 4.00g
  snap    volumeGroup swi-a- 1.00g base  18.97

# dmsetup status volumeGroup-snap
0 8388608 snapshot 397896/2097152 1560
                                  ^^^^ metadata sectors

# lvconvert --merge -b volumeGroup/snap
  Merging of volume snap started.

# lvs volumeGroup/snap
  LV      VG          Attr   LSize Origin  Snap%  Move Log Copy%  Convert
  base    volumeGroup Owi-a- 4.00g          17.23

# dmsetup status volumeGroup-base
0 8388608 snapshot-merge 281688/2097152 1104

# dmsetup status volumeGroup-base
0 8388608 snapshot-merge 180480/2097152 712

# dmsetup status volumeGroup-base
0 8388608 snapshot-merge 16/2097152 16
```

Merging has finished.

```
# lvs
  LV      VG          Attr   LSize Origin  Snap%  Move Log Copy%  Convert
  base    volumeGroup owi-a- 4.00g
```

39.25 DM statistics

Device Mapper supports the collection of I/O statistics on user-defined regions of a DM device. If no regions are defined no statistics are collected so there isn't any performance impact. Only bio-based DM devices are currently supported.

Each user-defined region specifies a starting sector, length and step. Individual statistics will be collected for each step-sized area within the range specified.

The I/O statistics counters for each step-sized area of a region are in the same format as */sys/block/*/stat* or */proc/diskstats* (see: *I/O statistics fields*). But two extra counters (12 and 13) are provided: total time spent reading and writing. When the histogram argument is used, the 14th parameter is reported that represents the histogram of latencies. All these counters may be accessed by sending the @stats_print message to the appropriate DM device via dmsetup.

The reported times are in milliseconds and the granularity depends on the kernel ticks. When the option precise_timestamps is used, the reported times are in nanoseconds.

Each region has a corresponding unique identifier, which we call a region_id, that is assigned

when the region is created. The region_id must be supplied when querying statistics about the region, deleting the region, etc. Unique region_ids enable multiple userspace programs to request and process statistics for the same DM device without stepping on each other's data.

The creation of DM statistics will allocate memory via kmalloc or fallback to using vmalloc space. At most, 1/4 of the overall system memory may be allocated by DM statistics. The admin can see how much memory is used by reading:

/sys/module/dm_mod/parameters/stats_current_allocated_bytes

39.25.1 Messages

@stats_create <range> <step> [<number_of_optional_arguments> <optional_arguments>...] [<program_id> [<aux_data>]]

Create a new region and return the region_id.

<range>

"-"
whole device

"<start_sector>+<length>"
a range of <length> 512-byte sectors starting with <start_sector>.

<step>

"<area_size>"
the range is subdivided into areas each containing <area_size> sectors.

"/<number_of_areas>"
the range is subdivided into the specified number of areas.

<number_of_optional_arguments>
The number of optional arguments

<optional_arguments>
The following optional arguments are supported:

precise_timestamps
use precise timer with nanosecond resolution instead of the "jiffies" variable. When this argument is used, the resulting times are in nanoseconds instead of milliseconds. Precise timestamps are a little bit slower to obtain than jiffies-based timestamps.

histogram:n1,n2,n3,n4,...
collect histogram of latencies. The numbers n1, n2, etc are times that represent the boundaries of the histogram. If precise_timestamps is not used, the times are in milliseconds, otherwise they are in nanoseconds. For each range, the kernel will report the number of requests that completed within this range. For example, if we use "histogram:10,20,30", the kernel will report four numbers a:b:c:d. a is the number of requests that took 0-10 ms to complete, b is the number of requests that took 10-20 ms to complete, c is the number of requests that took 20-30 ms to complete and d is the number of requests that took more than 30 ms to complete.

<program_id>
An optional parameter. A name that uniquely identifies the userspace owner

of the range. This groups ranges together so that userspace programs can identify the ranges they created and ignore those created by others. The kernel returns this string back in the output of @stats_list message, but it doesn't use it for anything else. If we omit the number of optional arguments, program id must not be a number, otherwise it would be interpreted as the number of optional arguments.

<aux_data>
An optional parameter. A word that provides auxiliary data that is useful to the client program that created the range. The kernel returns this string back in the output of @stats_list message, but it doesn't use this value for anything.

@stats_delete <region_id>
Delete the region with the specified id.

<region_id>
region_id returned from @stats_create

@stats_clear <region_id>
Clear all the counters except the in-flight i/o counters.

<region_id>
region_id returned from @stats_create

@stats_list [<program_id>]
List all regions registered with @stats_create.

<program_id>
An optional parameter. If this parameter is specified, only matching regions are returned. If it is not specified, all regions are returned.

Output format:

<region_id>: <start_sector>+<length> <step> <program_id> <aux_data>
precise_timestamps histogram:n1,n2,n3,...

The strings ”precise_timestamps” and ”histogram” are printed only if they were specified when creating the region.

@stats_print <region_id> [<starting_line> <number_of_lines>]
Print counters for each step-sized area of a region.

<region_id>
region_id returned from @stats_create

<starting_line>
The index of the starting line in the output. If omitted, all lines are returned.

<number_of_lines>
The number of lines to include in the output. If omitted, all lines are returned.

Output format for each step-sized area of a region:

<start_sector>+<length>
counters

The first 11 counters have the same meaning as */sys/block/*/stat or /proc/diskstats*.

Please refer to *I/O statistics fields* for details.

1. the number of reads completed
2. the number of reads merged
3. the number of sectors read
4. the number of milliseconds spent reading
5. the number of writes completed
6. the number of writes merged
7. the number of sectors written
8. the number of milliseconds spent writing
9. the number of I/Os currently in progress
10. the number of milliseconds spent doing I/Os
11. the weighted number of milliseconds spent doing I/Os

Additional counters:

12. the total time spent reading in milliseconds
13. the total time spent writing in milliseconds

@stats_print_clear <region_id> [<starting_line> <number_of_lines>]
Atomically print and then clear all the counters except the in-flight i/o counters. Useful when the client consuming the statistics does not want to lose any statistics (those updated between printing and clearing).

<region_id>
region_id returned from @stats_create

<starting_line>
The index of the starting line in the output. If omitted, all lines are printed and then cleared.

<number_of_lines>
The number of lines to process. If omitted, all lines are printed and then cleared.

@stats_set_aux <region_id> <aux_data>
Store auxiliary data aux_data for the specified region.

<region_id>
region_id returned from @stats_create

<aux_data>
The string that identifies data which is useful to the client program that created the range. The kernel returns this string back in the output of @stats_list message, but it doesn't use this value for anything.

39.25.2 Examples

Subdivide the DM device 'vol' into 100 pieces and start collecting statistics on them:

```
dmsetup message vol 0 @stats_create - /100
```

Set the auxiliary data string to ”foo bar baz” (the escape for each space must also be escaped, otherwise the shell will consume them):

```
dmsetup message vol 0 @stats_set_aux 0 foo\\ bar\\ baz
```

List the statistics:

```
dmsetup message vol 0 @stats_list
```

Print the statistics:

```
dmsetup message vol 0 @stats_print 0
```

Delete the statistics:

```
dmsetup message vol 0 @stats_delete 0
```

39.26 dm-stripe

Device-Mapper's ”striped” target is used to create a striped (i.e. RAID-0) device across one or more underlying devices. Data is written in ”chunks”, with consecutive chunks rotating among the underlying devices. This can potentially provide improved I/O throughput by utilizing several physical devices in parallel.

Parameters: <num devs> <chunk size> [<dev path> <offset>]+

<num devs>:
Number of underlying devices.

<chunk size>:
Size of each chunk of data. Must be at least as large as the system's PAGE_SIZE.

<dev path>:
Full pathname to the underlying block-device, or a ”major:minor” device-number.

<offset>:
Starting sector within the device.

One or more underlying devices can be specified. The striped device size must be a multiple of the chunk size multiplied by the number of underlying devices.

39.26.1 Example scripts

```
#!/usr/bin/perl -w
# Create a striped device across any number of underlying devices. The device
# will be called "stripe_dev" and have a chunk-size of 128k.

my $chunk_size = 128 * 2;
my $dev_name = "stripe_dev";
my $num_devs = @ARGV;
my @devs = @ARGV;
my ($min_dev_size, $stripe_dev_size, $i);

if (!$num_devs) {
        die("Specify at least one device\n");
}

$min_dev_size = `blockdev --getsz $devs[0]`;
for ($i = 1; $i < $num_devs; $i++) {
        my $this_size = `blockdev --getsz $devs[$i]`;
        $min_dev_size = ($min_dev_size < $this_size) ?
                        $min_dev_size : $this_size;
}

$stripe_dev_size = $min_dev_size * $num_devs;
$stripe_dev_size -= $stripe_dev_size % ($chunk_size * $num_devs);

$table = "0 $stripe_dev_size striped $num_devs $chunk_size";
for ($i = 0; $i < $num_devs; $i++) {
        $table .= " $devs[$i] 0";
}

`echo $table | dmsetup create $dev_name`;
```

39.27 dm-switch

The device-mapper switch target creates a device that supports an arbitrary mapping of fixed-size regions of I/O across a fixed set of paths. The path used for any specific region can be switched dynamically by sending the target a message.

It maps I/O to underlying block devices efficiently when there is a large number of fixed-sized address regions but there is no simple pattern that would allow for a compact representation of the mapping such as dm-stripe.

39.27.1 Background

Dell EqualLogic and some other iSCSI storage arrays use a distributed frameless architecture. In this architecture, the storage group consists of a number of distinct storage arrays (”members”) each having independent controllers, disk storage and network adapters. When a LUN is created it is spread across multiple members. The details of the spreading are hidden from initiators connected to this storage system. The storage group exposes a single target discovery portal, no matter how many members are being used. When iSCSI sessions are created, each session is connected to an eth port on a single member. Data to a LUN can be sent on any iSCSI session, and if the blocks being accessed are stored on another member the I/O will be forwarded as required. This forwarding is invisible to the initiator. The storage layout is also dynamic, and the blocks stored on disk may be moved from member to member as needed to balance the load.

This architecture simplifies the management and configuration of both the storage group and initiators. In a multipathing configuration, it is possible to set up multiple iSCSI sessions to use multiple network interfaces on both the host and target to take advantage of the increased network bandwidth. An initiator could use a simple round robin algorithm to send I/O across all paths and let the storage array members forward it as necessary, but there is a performance advantage to sending data directly to the correct member.

A device-mapper table already lets you map different regions of a device onto different targets. However in this architecture the LUN is spread with an address region size on the order of 10s of MBs, which means the resulting table could have more than a million entries and consume far too much memory.

Using this device-mapper switch target we can now build a two-layer device hierarchy:

> Upper Tier - Determine which array member the I/O should be sent to. Lower Tier - Load balance amongst paths to a particular member.

The lower tier consists of a single dm multipath device for each member. Each of these multipath devices contains the set of paths directly to the array member in one priority group, and leverages existing path selectors to load balance amongst these paths. We also build a non-preferred priority group containing paths to other array members for failover reasons.

The upper tier consists of a single dm-switch device. This device uses a bitmap to look up the location of the I/O and choose the appropriate lower tier device to route the I/O. By using a bitmap we are able to use 4 bits for each address range in a 16 member group (which is very large for us). This is a much denser representation than the dm table b-tree can achieve.

Construction Parameters

<num_paths> <region_size> <num_optional_args> [<optional_args>...] [<dev_path> <offset>]+

<num_paths>
The number of paths across which to distribute the I/O.

<region_size>
The number of 512-byte sectors in a region. Each region can be redirected to any of the available paths.

<num_optional_args>
The number of optional arguments. Currently, no optional arguments are

supported and so this must be zero.

<dev_path>
The block device that represents a specific path to the device.

<offset>
The offset of the start of data on the specific <dev_path> (in units of 512-byte sectors). This number is added to the sector number when forwarding the request to the specific path. Typically it is zero.

Messages

set_region_mappings <index>:<path_nr> [<index>]:<path_nr> [<index>]:<path_nr>...

Modify the region table by specifying which regions are redirected to which paths.

<index>
The region number (region size was specified in constructor parameters). If index is omitted, the next region (previous index + 1) is used. Expressed in hexadecimal (WITHOUT any prefix like 0x).

<path_nr>
The path number in the range 0 ... (<num_paths> - 1). Expressed in hexadecimal (WITHOUT any prefix like 0x).

R<n>,<m>
This parameter allows repetitive patterns to be loaded quickly. <n> and <m> are hexadecimal numbers. The last <n> mappings are repeated in the next <m> slots.

Status

No status line is reported.

Example

Assume that you have volumes vg1/switch0 vg1/switch1 vg1/switch2 with the same size.

Create a switch device with 64kB region size:

```
dmsetup create switch --table "0 `blockdev --getsz /dev/vg1/switch0`
    switch 3 128 0 /dev/vg1/switch0 0 /dev/vg1/switch1 0 /dev/vg1/switch2 0"
```

Set mappings for the first 7 entries to point to devices switch0, switch1, switch2, switch0, switch1, switch2, switch1:

```
dmsetup message switch 0 set_region_mappings 0:0 :1 :2 :0 :1 :2 :1
```

Set repetitive mapping. This command:

```
dmsetup message switch 0 set_region_mappings 1000:1 :2 R2,10
```

is equivalent to:

```
dmsetup message switch 0 set_region_mappings 1000:1 :2 :1 :2 :1 :2 :1 :2 \
    :1 :2 :1 :2 :1 :2 :1 :2 :1 :2
```

39.28 Thin provisioning

39.28.1 Introduction

This document describes a collection of device-mapper targets that between them implement thin-provisioning and snapshots.

The main highlight of this implementation, compared to the previous implementation of snapshots, is that it allows many virtual devices to be stored on the same data volume. This simplifies administration and allows the sharing of data between volumes, thus reducing disk usage.

Another significant feature is support for an arbitrary depth of recursive snapshots (snapshots of snapshots of snapshots ...). The previous implementation of snapshots did this by chaining together lookup tables, and so performance was O(depth). This new implementation uses a single data structure to avoid this degradation with depth. Fragmentation may still be an issue, however, in some scenarios.

Metadata is stored on a separate device from data, giving the administrator some freedom, for example to:

- Improve metadata resilience by storing metadata on a mirrored volume but data on a non-mirrored one.
- Improve performance by storing the metadata on SSD.

39.28.2 Status

These targets are considered safe for production use. But different use cases will have different performance characteristics, for example due to fragmentation of the data volume.

If you find this software is not performing as expected please mail dm-devel@redhat.com with details and we'll try our best to improve things for you.

Userspace tools for checking and repairing the metadata have been fully developed and are available as 'thin_check' and 'thin_repair'. The name of the package that provides these utilities varies by distribution (on a Red Hat distribution it is named 'device-mapper-persistent-data').

39.28.3 Cookbook

This section describes some quick recipes for using thin provisioning. They use the dmsetup program to control the device-mapper driver directly. End users will be advised to use a higher-level volume manager such as LVM2 once support has been added.

Pool device

The pool device ties together the metadata volume and the data volume. It maps I/O linearly to the data volume and updates the metadata via two mechanisms:

- Function calls from the thin targets
- Device-mapper 'messages' from userspace which control the creation of new virtual devices amongst other things.

Setting up a fresh pool device

Setting up a pool device requires a valid metadata device, and a data device. If you do not have an existing metadata device you can make one by zeroing the first 4k to indicate empty metadata.

dd if=/dev/zero of=$metadata_dev bs=4096 count=1

The amount of metadata you need will vary according to how many blocks are shared between thin devices (i.e. through snapshots). If you have less sharing than average you'll need a larger-than-average metadata device.

As a guide, we suggest you calculate the number of bytes to use in the metadata device as 48 * $data_dev_size / $data_block_size but round it up to 2MB if the answer is smaller. If you're creating large numbers of snapshots which are recording large amounts of change, you may find you need to increase this.

The largest size supported is 16GB: If the device is larger, a warning will be issued and the excess space will not be used.

Reloading a pool table

You may reload a pool's table, indeed this is how the pool is resized if it runs out of space. (N.B. While specifying a different metadata device when reloading is not forbidden at the moment, things will go wrong if it does not route I/O to exactly the same on-disk location as previously.)

Using an existing pool device

```
dmsetup create pool \
    --table "0 20971520 thin-pool $metadata_dev $data_dev \
             $data_block_size $low_water_mark"
```

$data_block_size gives the smallest unit of disk space that can be allocated at a time expressed in units of 512-byte sectors. $data_block_size must be between 128 (64KB) and 2097152 (1GB) and a multiple of 128 (64KB). $data_block_size cannot be changed after the thin-pool is created. People primarily interested in thin provisioning may want to use a value such as 1024 (512KB). People doing lots of snapshotting may want a smaller value such as 128 (64KB). If you are not zeroing newly-allocated data, a larger $data_block_size in the region of 256000 (128MB) is suggested.

$low_water_mark is expressed in blocks of size $data_block_size. If free space on the data device drops below this level then a dm event will be triggered which a userspace daemon should catch allowing it to extend the pool device. Only one such event will be sent.

No special event is triggered if a just resumed device's free space is below the low water mark. However, resuming a device always triggers an event; a userspace daemon should verify that free space exceeds the low water mark when handling this event.

A low water mark for the metadata device is maintained in the kernel and will trigger a dm event if free space on the metadata device drops below it.

Updating on-disk metadata

On-disk metadata is committed every time a FLUSH or FUA bio is written. If no such requests are made then commits will occur every second. This means the thin-provisioning target behaves like a physical disk that has a volatile write cache. If power is lost you may lose some recent writes. The metadata should always be consistent in spite of any crash.

If data space is exhausted the pool will either error or queue IO according to the configuration (see: error_if_no_space). If metadata space is exhausted or a metadata operation fails: the pool will error IO until the pool is taken offline and repair is performed to 1) fix any potential inconsistencies and 2) clear the flag that imposes repair. Once the pool's metadata device is repaired it may be resized, which will allow the pool to return to normal operation. Note that if a pool is flagged as needing repair, the pool's data and metadata devices cannot be resized until repair is performed. It should also be noted that when the pool's metadata space is exhausted the current metadata transaction is aborted. Given that the pool will cache IO whose completion may have already been acknowledged to upper IO layers (e.g. filesystem) it is strongly suggested that consistency checks (e.g. fsck) be performed on those layers when repair of the pool is required.

Thin provisioning

i) Creating a new thinly-provisioned volume.

   To create a new thinly- provisioned volume you must send a message to an active pool device, /dev/mapper/pool in this example:

```
dmsetup message /dev/mapper/pool 0 "create_thin 0"
```

   Here '0' is an identifier for the volume, a 24-bit number. It's up to the caller to allocate and manage these identifiers. If the identifier is already in use, the message will fail with -EEXIST.

ii) Using a thinly-provisioned volume.

   Thinly-provisioned volumes are activated using the 'thin' target:

```
dmsetup create thin --table "0 2097152 thin /dev/mapper/pool 0"
```

   The last parameter is the identifier for the thinp device.

Internal snapshots

i) Creating an internal snapshot.

Snapshots are created with another message to the pool.

N.B. If the origin device that you wish to snapshot is active, you must suspend it before creating the snapshot to avoid corruption. This is NOT enforced at the moment, so please be careful!

```
dmsetup suspend /dev/mapper/thin
dmsetup message /dev/mapper/pool 0 "create_snap 1 0"
dmsetup resume /dev/mapper/thin
```

Here '1' is the identifier for the volume, a 24-bit number. '0' is the identifier for the origin device.

ii) Using an internal snapshot.

Once created, the user doesn't have to worry about any connection between the origin and the snapshot. Indeed the snapshot is no different from any other thinly-provisioned device and can be snapshotted itself via the same method. It's perfectly legal to have only one of them active, and there's no ordering requirement on activating or removing them both. (This differs from conventional device-mapper snapshots.)

Activate it exactly the same way as any other thinly-provisioned volume:

```
dmsetup create snap --table "0 2097152 thin /dev/mapper/pool 1"
```

External snapshots

You can use an external **read only** device as an origin for a thinly-provisioned volume. Any read to an unprovisioned area of the thin device will be passed through to the origin. Writes trigger the allocation of new blocks as usual.

One use case for this is VM hosts that want to run guests on thinly-provisioned volumes but have the base image on another device (possibly shared between many VMs).

You must not write to the origin device if you use this technique! Of course, you may write to the thin device and take internal snapshots of the thin volume.

i) Creating a snapshot of an external device

This is the same as creating a thin device. You don't mention the origin at this stage.

```
dmsetup message /dev/mapper/pool 0 "create_thin 0"
```

ii) Using a snapshot of an external device.

Append an extra parameter to the thin target specifying the origin:

```
dmsetup create snap --table "0 2097152 thin /dev/mapper/pool 0 /dev/
↪image"
```

N.B. All descendants (internal snapshots) of this snapshot require the same extra origin parameter.

Deactivation

All devices using a pool must be deactivated before the pool itself can be.

```
dmsetup remove thin
dmsetup remove snap
dmsetup remove pool
```

39.28.4 Reference

'thin-pool' target

i) Constructor

```
thin-pool <metadata dev> <data dev> <data block size (sectors)> \
          <low water mark (blocks)> [<number of feature args> [<arg>
↪]*]
```

Optional feature arguments:

skip_block_zeroing:
Skip the zeroing of newly-provisioned blocks.

ignore_discard:
Disable discard support.

no_discard_passdown:
Don't pass discards down to the underlying data device, but just remove the mapping.

read_only:
Don't allow any changes to be made to the pool metadata. This mode is only available after the thin-pool has been created and first used in full read/write mode. It cannot be specified on initial thin-pool creation.

error_if_no_space:
Error IOs, instead of queueing, if no space.

Data block size must be between 64KB (128 sectors) and 1GB (2097152 sectors) inclusive.

ii) Status

```
<transaction id> <used metadata blocks>/<total metadata blocks>
<used data blocks>/<total data blocks> <held metadata root>
ro|rw|out_of_data_space [no_]discard_passdown [error|queue]_if_no_space
needs_check|- metadata_low_watermark
```

transaction id:
A 64-bit number used by userspace to help synchronise with metadata from volume managers.

used data blocks / total data blocks
If the number of free blocks drops below the pool's low water mark a dm event will be sent to userspace. This event is edge-triggered and it will occur only once after each

resume so volume manager writers should register for the event and then check the target's status.

held metadata root:
The location, in blocks, of the metadata root that has been 'held' for userspace read access. '-' indicates there is no held root.

discard_passdown|no_discard_passdown
Whether or not discards are actually being passed down to the underlying device. When this is enabled when loading the table, it can get disabled if the underlying device doesn't support it.

ro|rw|out_of_data_space
If the pool encounters certain types of device failures it will drop into a read-only metadata mode in which no changes to the pool metadata (like allocating new blocks) are permitted.

In serious cases where even a read-only mode is deemed unsafe no further I/O will be permitted and the status will just contain the string 'Fail'. The userspace recovery tools should then be used.

error_if_no_space|queue_if_no_space
If the pool runs out of data or metadata space, the pool will either queue or error the IO destined to the data device. The default is to queue the IO until more space is added or the 'no_space_timeout' expires. The 'no_space_timeout' dm-thin-pool module parameter can be used to change this timeout -- it defaults to 60 seconds but may be disabled using a value of 0.

needs_check
A metadata operation has failed, resulting in the needs_check flag being set in the metadata's superblock. The metadata device must be deactivated and checked/repaired before the thin-pool can be made fully operational again. '-' indicates needs_check is not set.

metadata_low_watermark:
Value of metadata low watermark in blocks. The kernel sets this value internally but userspace needs to know this value to determine if an event was caused by crossing this threshold.

iii) Messages

create_thin <dev id>
Create a new thinly-provisioned device. <dev id> is an arbitrary unique 24-bit identifier chosen by the caller.

create_snap <dev id> <origin id>
Create a new snapshot of another thinly-provisioned device. <dev id> is an arbitrary unique 24-bit identifier chosen by the caller. <origin id> is the identifier of the thinly-provisioned device of which the new device will be a snapshot.

delete <dev id>
Deletes a thin device. Irreversible.

set_transaction_id <current id> <new id>
Userland volume managers, such as LVM, need a way to synchronise their external metadata with the internal metadata of the pool target. The thin-pool target

offers to store an arbitrary 64-bit transaction id and return it on the target's status line. To avoid races you must provide what you think the current transaction id is when you change it with this compare-and-swap message.

reserve_metadata_snap
Reserve a copy of the data mapping btree for use by userland. This allows userland to inspect the mappings as they were when this message was executed. Use the pool's status command to get the root block associated with the metadata snapshot.

release_metadata_snap
Release a previously reserved copy of the data mapping btree.

'thin' target

i) Constructor

```
thin <pool dev> <dev id> [<external origin dev>]
```

pool dev:
the thin-pool device, e.g. /dev/mapper/my_pool or 253:0

dev id:
the internal device identifier of the device to be activated.

external origin dev:
an optional block device outside the pool to be treated as a read-only snapshot origin: reads to unprovisioned areas of the thin target will be mapped to this device.

The pool doesn't store any size against the thin devices. If you load a thin target that is smaller than you've been using previously, then you'll have no access to blocks mapped beyond the end. If you load a target that is bigger than before, then extra blocks will be provisioned as and when needed.

ii) Status

<nr mapped sectors> <highest mapped sector>
If the pool has encountered device errors and failed, the status will just contain the string 'Fail'. The userspace recovery tools should then be used.

In the case where <nr mapped sectors> is 0, there is no highest mapped sector and the value of <highest mapped sector> is unspecified.

39.29 Device-mapper ”unstriped” target

39.29.1 Introduction

The device-mapper ”unstriped” target provides a transparent mechanism to unstripe a device-mapper ”striped” target to access the underlying disks without having to touch the true backing block-device. It can also be used to unstripe a hardware RAID-0 to access backing disks.

Parameters: <number of stripes> <chunk size> <stripe #> <dev_path> <offset>

<number of stripes>
The number of stripes in the RAID 0.

<chunk size>
The amount of 512B sectors in the chunk striping.

<dev_path>
The block device you wish to unstripe.

<stripe #>
The stripe number within the device that corresponds to physical drive you wish to unstripe. This must be 0 indexed.

39.29.2 Why use this module?

An example of undoing an existing dm-stripe

This small bash script will setup 4 loop devices and use the existing striped target to combine the 4 devices into one. It then will use the unstriped target on top of the striped device to access the individual backing loop devices. We write data to the newly exposed unstriped devices and verify the data written matches the correct underlying device on the striped array:

```
#!/bin/bash

MEMBER_SIZE=$((128 * 1024 * 1024))
NUM=4
SEQ_END=$(($(NUM)-1))
CHUNK=256
BS=4096

RAID_SIZE=$(($(MEMBER_SIZE)*$(NUM)/512))
DM_PARMS="0 $(RAID_SIZE) striped $(NUM) $(CHUNK)"
COUNT=$(($(MEMBER_SIZE) / $(BS)))

for i in $(seq 0 $(SEQ_END)); do
  dd if=/dev/zero of=member-$(i) bs=$(MEMBER_SIZE) count=1 oflag=direct
  losetup /dev/loop$(i) member-$(i)
  DM_PARMS+=" /dev/loop$(i) 0"
done

echo $DM_PARMS | dmsetup create raid0
for i in $(seq 0 $(SEQ_END)); do
  echo "0 1 unstriped $(NUM) $(CHUNK) $(i) /dev/mapper/raid0 0" | dmsetup␣
↪create set-$(i)
done;

for i in $(seq 0 $(SEQ_END)); do
  dd if=/dev/urandom of=/dev/mapper/set-$(i) bs=$(BS) count=$(COUNT)␣
↪oflag=direct
  diff /dev/mapper/set-$(i) member-$(i)
done;
```

```
for i in $(seq 0 ${SEQ_END}); do
  dmsetup remove set-${i}
done

dmsetup remove raid0

for i in $(seq 0 ${SEQ_END}); do
  losetup -d /dev/loop${i}
  rm -f member-${i}
done
```

Another example

Intel NVMe drives contain two cores on the physical device. Each core of the drive has segregated access to its LBA range. The current LBA model has a RAID 0 128k chunk on each core, resulting in a 256k stripe across the two cores:

```
 Core 0:       Core 1:

___________   ___________
| LBA 512|    | LBA 768|
| LBA 0  |    | LBA 256|
----------    ----------
```

The purpose of this unstriping is to provide better QoS in noisy neighbor environments. When two partitions are created on the aggregate drive without this unstriping, reads on one partition can affect writes on another partition. This is because the partitions are striped across the two cores. When we unstripe this hardware RAID 0 and make partitions on each new exposed device the two partitions are now physically separated.

With the dm-unstriped target we're able to segregate an fio script that has read and write jobs that are independent of each other. Compared to when we run the test on a combined drive with partitions, we were able to get a 92% reduction in read latency using this device mapper target.

39.29.3 Example dmsetup usage

unstriped on top of Intel NVMe device that has 2 cores

```
dmsetup create nvmset0 --table '0 512 unstriped 2 256 0 /dev/nvme0n1 0'
dmsetup create nvmset1 --table '0 512 unstriped 2 256 1 /dev/nvme0n1 0'
```

There will now be two devices that expose Intel NVMe core 0 and 1 respectively:

```
/dev/mapper/nvmset0
/dev/mapper/nvmset1
```

unstriped on top of striped with 4 drives using 128K chunk size

```
dmsetup create raid_disk0 --table '0 512 unstriped 4 256 0 /dev/mapper/striped␣
↪0'
dmsetup create raid_disk1 --table '0 512 unstriped 4 256 1 /dev/mapper/striped␣
↪0'
dmsetup create raid_disk2 --table '0 512 unstriped 4 256 2 /dev/mapper/striped␣
↪0'
dmsetup create raid_disk3 --table '0 512 unstriped 4 256 3 /dev/mapper/striped␣
↪0'
```

39.30 dm-verity

Device-Mapper's "verity" target provides transparent integrity checking of block devices using a cryptographic digest provided by the kernel crypto API. This target is read-only.

39.30.1 Construction Parameters

```
<version> <dev> <hash_dev>
<data_block_size> <hash_block_size>
<num_data_blocks> <hash_start_block>
<algorithm> <digest> <salt>
[<#opt_params> <opt_params>]
```

<version>
This is the type of the on-disk hash format.

0 is the original format used in the Chromium OS.
The salt is appended when hashing, digests are stored continuously and the rest of the block is padded with zeroes.

1 is the current format that should be used for new devices.
The salt is prepended when hashing and each digest is padded with zeroes to the power of two.

<dev>
This is the device containing data, the integrity of which needs to be checked. It may be specified as a path, like /dev/sdaX, or a device number, <major>:<minor>.

<hash_dev>
This is the device that supplies the hash tree data. It may be specified similarly to the device path and may be the same device. If the same device is used, the hash_start should be outside the configured dm-verity device.

<data_block_size>
The block size on a data device in bytes. Each block corresponds to one digest on the hash device.

<hash_block_size>
The size of a hash block in bytes.

<num_data_blocks>
The number of data blocks on the data device. Additional blocks are inaccessible. You can place hashes to the same partition as data, in this case hashes are placed after <num_data_blocks>.

<hash_start_block>
This is the offset, in <hash_block_size>-blocks, from the start of hash_dev to the root block of the hash tree.

<algorithm>
The cryptographic hash algorithm used for this device. This should be the name of the algorithm, like "sha1".

<digest>
The hexadecimal encoding of the cryptographic hash of the root hash block and the salt. This hash should be trusted as there is no other authenticity beyond this point.

<salt>
The hexadecimal encoding of the salt value.

<#opt_params>
Number of optional parameters. If there are no optional parameters, the optional parameters section can be skipped or #opt_params can be zero. Otherwise #opt_params is the number of following arguments.

Example of optional parameters section:
1 ignore_corruption

ignore_corruption
Log corrupted blocks, but allow read operations to proceed normally.

restart_on_corruption
Restart the system when a corrupted block is discovered. This option is not compatible with ignore_corruption and requires user space support to avoid restart loops.

panic_on_corruption
Panic the device when a corrupted block is discovered. This option is not compatible with ignore_corruption and restart_on_corruption.

ignore_zero_blocks
Do not verify blocks that are expected to contain zeroes and always return zeroes instead. This may be useful if the partition contains unused blocks that are not guaranteed to contain zeroes.

use_fec_from_device <fec_dev>
Use forward error correction (FEC) to recover from corruption if hash verification fails. Use encoding data from the specified device. This may be the same device where data and hash blocks reside, in which case fec_start must be outside data and hash areas.

If the encoding data covers additional metadata, it must be accessible on the hash device after the hash blocks.

Note: block sizes for data and hash devices must match. Also, if the verity <dev> is encrypted the <fec_dev> should be too.

fec_roots <num>
Number of generator roots. This equals to the number of parity bytes in the encoding data. For example, in RS(M, N) encoding, the number of roots is M-N.

fec_blocks <num>
The number of encoding data blocks on the FEC device. The block size for the FEC device is <data_block_size>.

fec_start <offset>
This is the offset, in <data_block_size> blocks, from the start of the FEC device to the beginning of the encoding data.

check_at_most_once
Verify data blocks only the first time they are read from the data device, rather than every time. This reduces the overhead of dm-verity so that it can be used on systems that are memory and/or CPU constrained. However, it provides a reduced level of security because only offline tampering of the data device's content will be detected, not online tampering.

Hash blocks are still verified each time they are read from the hash device, since verification of hash blocks is less performance critical than data blocks, and a hash block will not be verified any more after all the data blocks it covers have been verified anyway.

root_hash_sig_key_desc <key_description>
This is the description of the USER_KEY that the kernel will lookup to get the pkcs7 signature of the roothash. The pkcs7 signature is used to validate the root hash during the creation of the device mapper block device. Verification of roothash depends on the config DM_VERITY_VERIFY_ROOTHASH_SIG being set in the kernel. The signatures are checked against the builtin trusted keyring by default, or the secondary trusted keyring if DM_VERITY_VERIFY_ROOTHASH_SIG_SECONDARY_KEYRING is set. The secondary trusted keyring includes by default the builtin trusted keyring, and it can also gain new certificates at run time if they are signed by a certificate already in the secondary trusted keyring.

try_verify_in_tasklet
If verity hashes are in cache, verify data blocks in kernel tasklet instead of workqueue. This option can reduce IO latency.

39.30.2 Theory of operation

dm-verity is meant to be set up as part of a verified boot path. This may be anything ranging from a boot using tboot or trustedgrub to just booting from a known-good device (like a USB drive or CD).

When a dm-verity device is configured, it is expected that the caller has been authenticated in some way (cryptographic signatures, etc). After instantiation, all hashes will be verified on-demand during disk access. If they cannot be verified up to the root node of the tree, the root hash, then the I/O will fail. This should detect tampering with any data on the device and the hash data.

Cryptographic hashes are used to assert the integrity of the device on a per-block basis. This allows for a lightweight hash computation on first read into the page cache. Block hashes are stored linearly, aligned to the nearest block size.

If forward error correction (FEC) support is enabled any recovery of corrupted data will be verified using the cryptographic hash of the corresponding data. This is why combining error correction with integrity checking is essential.

Hash Tree

Each node in the tree is a cryptographic hash. If it is a leaf node, the hash of some data block on disk is calculated. If it is an intermediary node, the hash of a number of child nodes is calculated.

Each entry in the tree is a collection of neighboring nodes that fit in one block. The number is determined based on block_size and the size of the selected cryptographic digest algorithm. The hashes are linearly-ordered in this entry and any unaligned trailing space is ignored but included when calculating the parent node.

The tree looks something like:

alg = sha256, num_blocks = 32768, block_size = 4096

```
                            [   root    ]
                           /    . . .    \
                [entry_0]                 [entry_1]
              /  . . .  \                  . . .   \
    [entry_0_0]   . . .  [entry_0_127]    . . . .  [entry_1_127]
      / ... \             /   . . .  \             /           \
blk_0 ... blk_127  blk_16256   blk_16383      blk_32640 . . . blk_32767
```

39.30.3 On-disk format

The verity kernel code does not read the verity metadata on-disk header. It only reads the hash blocks which directly follow the header. It is expected that a user-space tool will verify the integrity of the verity header.

Alternatively, the header can be omitted and the dmsetup parameters can be passed via the kernel command-line in a rooted chain of trust where the command-line is verified.

Directly following the header (and with sector number padded to the next hash block boundary) are the hash blocks which are stored a depth at a time (starting from the root), sorted in order of increasing index.

The full specification of kernel parameters and on-disk metadata format is available at the cryptsetup project's wiki page

https://gitlab.com/cryptsetup/cryptsetup/wikis/DMVerity

39.30.4 Status

V (for Valid) is returned if every check performed so far was valid. If any check failed, C (for Corruption) is returned.

39.30.5 Example

Set up a device:

```
# dmsetup create vroot --readonly --table \
  "0 2097152 verity 1 /dev/sda1 /dev/sda2 4096 4096 262144 1 sha256 "\
  "4392712ba01368efdf14b05c76f9e4df0d53664630b5d48632ed17a137f39076 "\
  "1234000000000000000000000000000000000000000000000000000000000000"
```

A command line tool veritysetup is available to compute or verify the hash tree or activate the kernel device. This is available from the cryptsetup upstream repository https://gitlab.com/cryptsetup/cryptsetup/ (as a libcryptsetup extension).

Create hash on the device:

```
# veritysetup format /dev/sda1 /dev/sda2
...
Root hash: 4392712ba01368efdf14b05c76f9e4df0d53664630b5d48632ed17a137f39076
```

Activate the device:

```
# veritysetup create vroot /dev/sda1 /dev/sda2 \
  4392712ba01368efdf14b05c76f9e4df0d53664630b5d48632ed17a137f39076
```

39.31 Writecache target

The writecache target caches writes on persistent memory or on SSD. It doesn't cache reads because reads are supposed to be cached in page cache in normal RAM.

When the device is constructed, the first sector should be zeroed or the first sector should contain valid superblock from previous invocation.

Constructor parameters:

1. **type of the cache device - "p" or "s"**
   - p - persistent memory
   - s - SSD
2. the underlying device that will be cached
3. the cache device
4. block size (4096 is recommended; the maximum block size is the page size)
5. the number of optional parameters (the parameters with an argument count as two)

   start_sector n (default: 0)
   offset from the start of cache device in 512-byte sectors

   high_watermark n (default: 50)
   start writeback when the number of used blocks reach this watermark

   low_watermark x (default: 45)
   stop writeback when the number of used blocks drops below this watermark

writeback_jobs n (default: unlimited)
limit the number of blocks that are in flight during writeback. Setting this value reduces writeback throughput, but it may improve latency of read requests

autocommit_blocks n (default: 64 for pmem, 65536 for ssd)
when the application writes this amount of blocks without issuing the FLUSH request, the blocks are automatically committed

autocommit_time ms (default: 1000)
autocommit time in milliseconds. The data is automatically committed if this time passes and no FLUSH request is received

fua (by default on)
applicable only to persistent memory - use the FUA flag when writing data from persistent memory back to the underlying device

nofua
applicable only to persistent memory - don't use the FUA flag when writing back data and send the FLUSH request afterwards

- some underlying devices perform better with fua, some with nofua. The user should test it

cleaner
when this option is activated (either in the constructor arguments or by a message), the cache will not promote new writes (however, writes to already cached blocks are promoted, to avoid data corruption due to misordered writes) and it will gradually writeback any cached data. The userspace can then monitor the cleaning process with ”dmsetup status”. When the number of cached blocks drops to zero, userspace can unload the dm-writecache target and replace it with dm-linear or other targets.

max_age n
specifies the maximum age of a block in milliseconds. If a block is stored in the cache for too long, it will be written to the underlying device and cleaned up.

metadata_only
only metadata is promoted to the cache. This option improves performance for heavier REQ_META workloads.

pause_writeback n (default: 3000)
pause writeback if there was some write I/O redirected to the origin volume in the last n milliseconds

Status:

1. error indicator - 0 if there was no error, otherwise error number
2. the number of blocks
3. the number of free blocks
4. the number of blocks under writeback
5. the number of read blocks
6. the number of read blocks that hit the cache

7. the number of write blocks
8. the number of write blocks that hit uncommitted block
9. the number of write blocks that hit committed block
10. the number of write blocks that bypass the cache
11. the number of write blocks that are allocated in the cache
12. the number of write requests that are blocked on the freelist
13. the number of flush requests
14. the number of discarded blocks

Messages:

flush
Flush the cache device. The message returns successfully if the cache device was flushed without an error

flush_on_suspend
Flush the cache device on next suspend. Use this message when you are going to remove the cache device. The proper sequence for removing the cache device is:

1. send the ”flush_on_suspend” message
2. load an inactive table with a linear target that maps to the underlying device
3. suspend the device
4. ask for status and verify that there are no errors
5. resume the device, so that it will use the linear target
6. the cache device is now inactive and it can be deleted

cleaner
See above ”cleaner” constructor documentation.

clear_stats
Clear the statistics that are reported on the status line

39.32 dm-zero

Device-Mapper's ”zero” target provides a block-device that always returns zero'd data on reads and silently drops writes. This is similar behavior to /dev/zero, but as a block-device instead of a character-device.

Dm-zero has no target-specific parameters.

One very interesting use of dm-zero is for creating ”sparse” devices in conjunction with dm-snapshot. A sparse device reports a device-size larger than the amount of actual storage space available for that device. A user can write data anywhere within the sparse device and read it back like a normal device. Reads to previously unwritten areas will return a zero'd buffer. When enough data has been written to fill up the actual storage space, the sparse device is deactivated. This can be very useful for testing device and filesystem limitations.

To create a sparse device, start by creating a dm-zero device that's the desired size of the sparse device. For this example, we'll assume a 10TB sparse device:

```
TEN_TERABYTES=`expr 10 \* 1024 \* 1024 \* 1024 \* 2`   # 10 TB in sectors
echo "0 $TEN_TERABYTES zero" | dmsetup create zero1
```

Then create a snapshot of the zero device, using any available block-device as the COW device. The size of the COW device will determine the amount of real space available to the sparse device. For this example, we'll assume /dev/sdb1 is an available 10GB partition:

```
echo "0 $TEN_TERABYTES snapshot /dev/mapper/zero1 /dev/sdb1 p 128" | \
   dmsetup create sparse1
```

This will create a 10TB sparse device called /dev/mapper/sparse1 that has 10GB of actual storage space available. If more than 10GB of data is written to this device, it will start returning I/O errors.

CHAPTER

FORTY

EDID

In the good old days when graphics parameters were configured explicitly in a file called xorg.conf, even broken hardware could be managed.

Today, with the advent of Kernel Mode Setting, a graphics board is either correctly working because all components follow the standards - or the computer is unusable, because the screen remains dark after booting or it displays the wrong area. Cases when this happens are:

- The graphics board does not recognize the monitor.
- The graphics board is unable to detect any EDID data.
- The graphics board incorrectly forwards EDID data to the driver.
- The monitor sends no or bogus EDID data.
- A KVM sends its own EDID data instead of querying the connected monitor.

Adding the kernel parameter ”nomodeset” helps in most cases, but causes restrictions later on.

As a remedy for such situations, the kernel configuration item CONFIG_DRM_LOAD_EDID_FIRMWARE was introduced. It allows to provide an individually prepared or corrected EDID data set in the /lib/firmware directory from where it is loaded via the firmware interface. The code (see drivers/gpu/drm/drm_edid_load.c) contains built-in data sets for commonly used screen resolutions (800x600, 1024x768, 1280x1024, 1600x1200, 1680x1050, 1920x1080) as binary blobs, but the kernel source tree does not contain code to create these data. In order to elucidate the origin of the built-in binary EDID blobs and to facilitate the creation of individual data for a specific misbehaving monitor, commented sources and a Makefile environment are given here.

To create binary EDID and C source code files from the existing data material, simply type ”make” in tools/edid/.

If you want to create your own EDID file, copy the file 1024x768.S, replace the settings with your own data and add a new target to the Makefile. Please note that the EDID data structure expects the timing values in a different way as compared to the standard X11 format.

X11:

HTimings:
hdisp hsyncstart hsyncend htotal

VTimings:
vdisp vsyncstart vsyncend vtotal

EDID:

```
#define XPIX hdisp
#define XBLANK htotal-hdisp
#define XOFFSET hsyncstart-hdisp
#define XPULSE hsyncend-hsyncstart

#define YPIX vdisp
#define YBLANK vtotal-vdisp
#define YOFFSET vsyncstart-vdisp
#define YPULSE vsyncend-vsyncstart
```

CHAPTER

FORTYONE

THE EFI BOOT STUB

On the x86 and ARM platforms, a kernel zImage/bzImage can masquerade as a PE/COFF image, thereby convincing EFI firmware loaders to load it as an EFI executable. The code that modifies the bzImage header, along with the EFI-specific entry point that the firmware loader jumps to are collectively known as the "EFI boot stub", and live in arch/x86/boot/header.S and drivers/firmware/efi/libstub/x86-stub.c, respectively. For ARM the EFI stub is implemented in arch/arm/boot/compressed/efi-header.S and drivers/firmware/efi/libstub/arm32-stub.c. EFI stub code that is shared between architectures is in drivers/firmware/efi/libstub.

For arm64, there is no compressed kernel support, so the Image itself masquerades as a PE/COFF image and the EFI stub is linked into the kernel. The arm64 EFI stub lives in drivers/firmware/efi/libstub/arm64.c and drivers/firmware/efi/libstub/arm64-stub.c.

By using the EFI boot stub it's possible to boot a Linux kernel without the use of a conventional EFI boot loader, such as grub or elilo. Since the EFI boot stub performs the jobs of a boot loader, in a certain sense it *IS* the boot loader.

The EFI boot stub is enabled with the CONFIG_EFI_STUB kernel option.

41.1 How to install bzImage.efi

The bzImage located in arch/x86/boot/bzImage must be copied to the EFI System Partition (ESP) and renamed with the extension ".efi". Without the extension the EFI firmware loader will refuse to execute it. It's not possible to execute bzImage.efi from the usual Linux file systems because EFI firmware doesn't have support for them. For ARM the arch/arm/boot/zImage should be copied to the system partition, and it may not need to be renamed. Similarly for arm64, arch/arm64/boot/Image should be copied but not necessarily renamed.

41.2 Passing kernel parameters from the EFI shell

Arguments to the kernel can be passed after bzImage.efi, e.g.:

```
fs0:> bzImage.efi console=ttyS0 root=/dev/sda4
```

41.3 The ”initrd=” option

Like most boot loaders, the EFI stub allows the user to specify multiple initrd files using the ”initrd=” option. This is the only EFI stub-specific command line parameter, everything else is passed to the kernel when it boots.

The path to the initrd file must be an absolute path from the beginning of the ESP, relative path names do not work. Also, the path is an EFI-style path and directory elements must be separated with backslashes (). For example, given the following directory layout:

```
fs0:>
      Kernels\
                        bzImage.efi
                        initrd-large.img

      Ramdisks\
                        initrd-small.img
                        initrd-medium.img
```

to boot with the initrd-large.img file if the current working directory is fs0:Kernels, the following command must be used:

```
fs0:\Kernels> bzImage.efi initrd=\Kernels\initrd-large.img
```

Notice how bzImage.efi can be specified with a relative path. That's because the image we're executing is interpreted by the EFI shell, which understands relative paths, whereas the rest of the command line is passed to bzImage.efi.

41.4 The ”dtb=” option

For the ARM and arm64 architectures, a device tree must be provided to the kernel. Normally firmware shall supply the device tree via the EFI CONFIGURATION TABLE. However, the ”dtb=” command line option can be used to override the firmware supplied device tree, or to supply one when firmware is unable to.

Please note: Firmware adds runtime configuration information to the device tree before booting the kernel. If dtb= is used to override the device tree, then any runtime data provided by firmware will be lost. The dtb= option should only be used either as a debug tool, or as a last resort when a device tree is not provided in the EFI CONFIGURATION TABLE.

”dtb=” is processed in the same manner as the ”initrd=” option that is described above.

CHAPTER

FORTYTWO

EXT4 GENERAL INFORMATION

Ext4 is an advanced level of the ext3 filesystem which incorporates scalability and reliability enhancements for supporting large filesystems (64 bit) in keeping with increasing disk capacities and state-of-the-art feature requirements.

Mailing list: linux-ext4@vger.kernel.org Web site: http://ext4.wiki.kernel.org

42.1 Quick usage instructions

Note: More extensive information for getting started with ext4 can be found at the ext4 wiki site at the URL: http://ext4.wiki.kernel.org/index.php/Ext4_Howto

- The latest version of e2fsprogs can be found at:

  https://www.kernel.org/pub/linux/kernel/people/tytso/e2fsprogs/

  or

  http://sourceforge.net/project/showfiles.php?group_id=2406

  or grab the latest git repository from:

  https://git.kernel.org/pub/scm/fs/ext2/e2fsprogs.git

- Create a new filesystem using the ext4 filesystem type:

  ```
  # mke2fs -t ext4 /dev/hda1
  ```

  Or to configure an existing ext3 filesystem to support extents:

  ```
  # tune2fs -O extents /dev/hda1
  ```

  If the filesystem was created with 128 byte inodes, it can be converted to use 256 byte for greater efficiency via:

  ```
  # tune2fs -I 256 /dev/hda1
  ```

- Mounting:

  ```
  # mount -t ext4 /dev/hda1 /wherever
  ```

- When comparing performance with other filesystems, it's always important to try multiple workloads; very often a subtle change in a workload parameter can completely change the ranking of which filesystems do well compared to others. When comparing versus ext3, note that ext4 enables write barriers by default, while ext3 does not enable write barriers by default. So it is useful to use explicitly specify whether barriers are enabled or not when via the '-o barriers=[0|1]'

mount option for both ext3 and ext4 filesystems for a fair comparison. When tuning ext3 for best benchmark numbers, it is often worthwhile to try changing the data journaling mode; '-o data=writeback' can be faster for some workloads. (Note however that running mounted with data=writeback can potentially leave stale data exposed in recently written files in case of an unclean shutdown, which could be a security exposure in some situations.) Configuring the filesystem with a large journal can also be helpful for metadata-intensive workloads.

42.2 Features

42.2.1 Currently Available

- ability to use filesystems > 16TB (e2fsprogs support not available yet)
- extent format reduces metadata overhead (RAM, IO for access, transactions)
- extent format more robust in face of on-disk corruption due to magics,
- internal redundancy in tree
- improved file allocation (multi-block alloc)
- lift 32000 subdirectory limit imposed by i_links_count[1]
- nsec timestamps for mtime, atime, ctime, create time
- inode version field on disk (NFSv4, Lustre)
- reduced e2fsck time via uninit_bg feature
- journal checksumming for robustness, performance
- persistent file preallocation (e.g for streaming media, databases)
- ability to pack bitmaps and inode tables into larger virtual groups via the flex_bg feature
- large file support
- inode allocation using large virtual block groups via flex_bg
- delayed allocation
- large block (up to pagesize) support
- efficient new ordered mode in JBD2 and ext4 (avoid using buffer head to force the ordering)
- Case-insensitive file name lookups
- file-based encryption support (fscrypt)
- file-based verity support (fsverity)

[1] Filesystems with a block size of 1k may see a limit imposed by the directory hash tree having a maximum depth of two.

42.3 case-insensitive file name lookups

The case-insensitive file name lookup feature is supported on a per-directory basis, allowing the user to mix case-insensitive and case-sensitive directories in the same filesystem. It is enabled by flipping the +F inode attribute of an empty directory. The case-insensitive string match operation is only defined when we know how text in encoded in a byte sequence. For that reason, in order to enable case-insensitive directories, the filesystem must have the casefold feature, which stores the filesystem-wide encoding model used. By default, the charset adopted is the latest version of Unicode (12.1.0, by the time of this writing), encoded in the UTF-8 form. The comparison algorithm is implemented by normalizing the strings to the Canonical decomposition form, as defined by Unicode, followed by a byte per byte comparison.

The case-awareness is name-preserving on the disk, meaning that the file name provided by userspace is a byte-per-byte match to what is actually written in the disk. The Unicode normalization format used by the kernel is thus an internal representation, and not exposed to the userspace nor to the disk, with the important exception of disk hashes, used on large case-insensitive directories with DX feature. On DX directories, the hash must be calculated using the casefolded version of the filename, meaning that the normalization format used actually has an impact on where the directory entry is stored.

When we change from viewing filenames as opaque byte sequences to seeing them as encoded strings we need to address what happens when a program tries to create a file with an invalid name. The Unicode subsystem within the kernel leaves the decision of what to do in this case to the filesystem, which select its preferred behavior by enabling/disabling the strict mode. When Ext4 encounters one of those strings and the filesystem did not require strict mode, it falls back to considering the entire string as an opaque byte sequence, which still allows the user to operate on that file, but the case-insensitive lookups won't work.

42.4 Options

When mounting an ext4 filesystem, the following option are accepted: (*) == default

ro
Mount filesystem read only. Note that ext4 will replay the journal (and thus write to the partition) even when mounted ”read only”. The mount options ”ro,noload” can be used to prevent writes to the filesystem.

journal_checksum
Enable checksumming of the journal transactions. This will allow the recovery code in e2fsck and the kernel to detect corruption in the kernel. It is a compatible change and will be ignored by older kernels.

journal_async_commit
Commit block can be written to disk without waiting for descriptor blocks. If enabled older kernels cannot mount the device. This will enable 'journal_checksum' internally.

journal_path=path, journal_dev=devnum
When the external journal device's major/minor numbers have changed, these options allow the user to specify the new journal location. The journal device is identified through either its new major/minor numbers encoded in devnum, or via a path to the device.

norecovery, noload
Don't load the journal on mounting. Note that if the filesystem was not unmounted cleanly, skipping the journal replay will lead to the filesystem containing inconsistencies that can lead to any number of problems.

data=journal
All data are committed into the journal prior to being written into the main file system. Enabling this mode will disable delayed allocation and O_DIRECT support.

data=ordered (*)
All data are forced directly out to the main file system prior to its metadata being committed to the journal.

data=writeback
Data ordering is not preserved, data may be written into the main file system after its metadata has been committed to the journal.

commit=nrsec (*)
This setting limits the maximum age of the running transaction to 'nrsec' seconds. The default value is 5 seconds. This means that if you lose your power, you will lose as much as the latest 5 seconds of metadata changes (your filesystem will not be damaged though, thanks to the journaling). This default value (or any low value) will hurt performance, but it's good for data-safety. Setting it to 0 will have the same effect as leaving it at the default (5 seconds). Setting it to very large values will improve performance. Note that due to delayed allocation even older data can be lost on power failure since writeback of those data begins only after time set in /proc/sys/vm/dirty_expire_centisecs.

barrier=<0|1(*)>, barrier(*), nobarrier
This enables/disables the use of write barriers in the jbd code. barrier=0 disables, barrier=1 enables. This also requires an IO stack which can support barriers, and if jbd gets an error on a barrier write, it will disable again with a warning. Write barriers enforce proper on-disk ordering of journal commits, making volatile disk write caches safe to use, at some performance penalty. If your disks are battery-backed in one way or another, disabling barriers may safely improve performance. The mount options ”barrier” and ”nobarrier” can also be used to enable or disable barriers, for consistency with other ext4 mount options.

inode_readahead_blks=n
This tuning parameter controls the maximum number of inode table blocks that ext4's inode table readahead algorithm will pre-read into the buffer cache. The default value is 32 blocks.

nouser_xattr
Disables Extended User Attributes. See the attr(5) manual page for more information about extended attributes.

noacl
This option disables POSIX Access Control List support. If ACL support is enabled in the kernel configuration (CONFIG_EXT4_FS_POSIX_ACL), ACL is enabled by default on mount. See the acl(5) manual page for more information about acl.

bsddf (*)
Make 'df' act like BSD.

minixdf
Make 'df' act like Minix.

debug
Extra debugging information is sent to syslog.

abort
Simulate the effects of calling ext4_abort() for debugging purposes. This is normally used while remounting a filesystem which is already mounted.

errors=remount-ro
Remount the filesystem read-only on an error.

errors=continue
Keep going on a filesystem error.

errors=panic
Panic and halt the machine if an error occurs. (These mount options override the errors behavior specified in the superblock, which can be configured using tune2fs)

data_err=ignore(*)
Just print an error message if an error occurs in a file data buffer in ordered mode.

data_err=abort
Abort the journal if an error occurs in a file data buffer in ordered mode.

grpid | bsdgroups
New objects have the group ID of their parent.

nogrpid (*) | sysvgroups
New objects have the group ID of their creator.

resgid=n
The group ID which may use the reserved blocks.

resuid=n
The user ID which may use the reserved blocks.

sb=
Use alternate superblock at this location.

quota, noquota, grpquota, usrquota
These options are ignored by the filesystem. They are used only by quota tools to recognize volumes where quota should be turned on. See documentation in the quota-tools package for more details (http://sourceforge.net/projects/linuxquota).

jqfmt=<quota type>, usrjquota=<file>, grpjquota=<file>
These options tell filesystem details about quota so that quota information can be properly updated during journal replay. They replace the above quota options. See documentation in the quota-tools package for more details (http://sourceforge.net/projects/linuxquota).

stripe=n
Number of filesystem blocks that mballoc will try to use for allocation size and alignment. For RAID5/6 systems this should be the number of data disks * RAID chunk size in file system blocks.

delalloc (*)
Defer block allocation until just before ext4 writes out the block(s) in question. This allows ext4 to better allocation decisions more efficiently.

nodelalloc
Disable delayed allocation. Blocks are allocated when the data is copied from userspace to the page cache, either via the write(2) system call or when an mmap'ed page which was previously unallocated is written for the first time.

max_batch_time=usec
Maximum amount of time ext4 should wait for additional filesystem operations to be batch together with a synchronous write operation. Since a synchronous write operation is going to force a commit and then a wait for the I/O complete, it doesn't cost much, and can be a huge throughput win, we wait for a small amount of time to see if any other transactions can piggyback on the synchronous write. The algorithm used is designed to automatically tune for the speed of the disk, by measuring the amount of time (on average) that it takes to finish committing a transaction. Call this time the ”commit time”. If the time that the transaction has been running is less than the commit time, ext4 will try sleeping for the commit time to see if other operations will join the transaction. The commit time is capped by the max_batch_time, which defaults to 15000us (15ms). This optimization can be turned off entirely by setting max_batch_time to 0.

min_batch_time=usec
This parameter sets the commit time (as described above) to be at least min_batch_time. It defaults to zero microseconds. Increasing this parameter may improve the throughput of multi-threaded, synchronous workloads on very fast disks, at the cost of increasing latency.

journal_ioprio=prio
The I/O priority (from 0 to 7, where 0 is the highest priority) which should be used for I/O operations submitted by kjournald2 during a commit operation. This defaults to 3, which is a slightly higher priority than the default I/O priority.

auto_da_alloc(*), noauto_da_alloc
Many broken applications don't use fsync() when replacing existing files via patterns such as fd = open(”foo.new”)/write(fd,..)/close(fd)/ rename(”foo.new”, ”foo”), or worse yet, fd = open(”foo”, O_TRUNC)/write(fd,..)/close(fd). If auto_da_alloc is enabled, ext4 will detect the replace-via-rename and replace-via-truncate patterns and force that any delayed allocation blocks are allocated such that at the next journal commit, in the default data=ordered mode, the data blocks of the new file are forced to disk before the rename() operation is committed. This provides roughly the same level of guarantees as ext3, and avoids the ”zero-length” problem that can happen when a system crashes before the delayed allocation blocks are forced to disk.

noinit_itable
Do not initialize any uninitialized inode table blocks in the background. This feature may be used by installation CD's so that the install process can complete as quickly as possible; the inode table initialization process would then be deferred until the next time the file system is unmounted.

init_itable=n
The lazy itable init code will wait n times the number of milliseconds it took to zero out the previous block group's inode table. This minimizes the impact on

the system performance while file system's inode table is being initialized.

discard, nodiscard(*)
Controls whether ext4 should issue discard/TRIM commands to the underlying block device when blocks are freed. This is useful for SSD devices and sparse/thinly-provisioned LUNs, but it is off by default until sufficient testing has been done.

nouid32
Disables 32-bit UIDs and GIDs. This is for interoperability with older kernels which only store and expect 16-bit values.

block_validity(*), noblock_validity
These options enable or disable the in-kernel facility for tracking filesystem metadata blocks within internal data structures. This allows multi- block allocator and other routines to notice bugs or corrupted allocation bitmaps which cause blocks to be allocated which overlap with filesystem metadata blocks.

dioread_lock, dioread_nolock
Controls whether or not ext4 should use the DIO read locking. If the dioread_nolock option is specified ext4 will allocate uninitialized extent before buffer write and convert the extent to initialized after IO completes. This approach allows ext4 code to avoid using inode mutex, which improves scalability on high speed storages. However this does not work with data journaling and dioread_nolock option will be ignored with kernel warning. Note that dioread_nolock code path is only used for extent-based files. Because of the restrictions this options comprises it is off by default (e.g. dioread_lock).

max_dir_size_kb=n
This limits the size of directories so that any attempt to expand them beyond the specified limit in kilobytes will cause an ENOSPC error. This is useful in memory constrained environments, where a very large directory can cause severe performance problems or even provoke the Out Of Memory killer. (For example, if there is only 512mb memory available, a 176mb directory may seriously cramp the system's style.)

i_version
Enable 64-bit inode version support. This option is off by default.

dax
Use direct access (no page cache). See Documentation/filesystems/dax.rst. Note that this option is incompatible with data=journal.

inlinecrypt
When possible, encrypt/decrypt the contents of encrypted files using the blk-crypto framework rather than filesystem-layer encryption. This allows the use of inline encryption hardware. The on-disk format is unaffected. For more details, see Documentation/block/inline-encryption.rst.

42.5 Data Mode

There are 3 different data modes:

- writeback mode

  In data=writeback mode, ext4 does not journal data at all. This mode provides a similar level of journaling as that of XFS, JFS, and ReiserFS in its default mode - metadata journaling. A crash+recovery can cause incorrect data to appear in files which were written shortly before the crash. This mode will typically provide the best ext4 performance.

- ordered mode

  In data=ordered mode, ext4 only officially journals metadata, but it logically groups metadata information related to data changes with the data blocks into a single unit called a transaction. When it's time to write the new metadata out to disk, the associated data blocks are written first. In general, this mode performs slightly slower than writeback but significantly faster than journal mode.

- journal mode

  data=journal mode provides full data and metadata journaling. All new data is written to the journal first, and then to its final location. In the event of a crash, the journal can be replayed, bringing both data and metadata into a consistent state. This mode is the slowest except when data needs to be read from and written to disk at the same time where it outperforms all others modes. Enabling this mode will disable delayed allocation and O_DIRECT support.

42.6 /proc entries

Information about mounted ext4 file systems can be found in /proc/fs/ext4. Each mounted filesystem will have a directory in /proc/fs/ext4 based on its device name (i.e., /proc/fs/ext4/hdc or /proc/fs/ext4/dm-0). The files in each per-device directory are shown in table below.

Files in /proc/fs/ext4/<devname>

mb_groups
: details of multiblock allocator buddy cache of free blocks

42.7 /sys entries

Information about mounted ext4 file systems can be found in /sys/fs/ext4. Each mounted filesystem will have a directory in /sys/fs/ext4 based on its device name (i.e., /sys/fs/ext4/hdc or /sys/fs/ext4/dm-0). The files in each per-device directory are shown in table below.

Files in /sys/fs/ext4/<devname>:

(see also Documentation/ABI/testing/sysfs-fs-ext4)

delayed_allocation_blocks
: This file is read-only and shows the number of blocks that are dirty in the page cache, but which do not have their location in the filesystem allocated yet.

inode_goal
Tuning parameter which (if non-zero) controls the goal inode used by the inode allocator in preference to all other allocation heuristics. This is intended for debugging use only, and should be 0 on production systems.

inode_readahead_blks
Tuning parameter which controls the maximum number of inode table blocks that ext4's inode table readahead algorithm will pre-read into the buffer cache.

lifetime_write_kbytes
This file is read-only and shows the number of kilobytes of data that have been written to this filesystem since it was created.

max_writeback_mb_bump
The maximum number of megabytes the writeback code will try to write out before move on to another inode.

mb_group_prealloc
The multiblock allocator will round up allocation requests to a multiple of this tuning parameter if the stripe size is not set in the ext4 superblock

mb_max_to_scan
The maximum number of extents the multiblock allocator will search to find the best extent.

mb_min_to_scan
The minimum number of extents the multiblock allocator will search to find the best extent.

mb_order2_req
Tuning parameter which controls the minimum size for requests (as a power of 2) where the buddy cache is used.

mb_stats
Controls whether the multiblock allocator should collect statistics, which are shown during the unmount. 1 means to collect statistics, 0 means not to collect statistics.

mb_stream_req
Files which have fewer blocks than this tunable parameter will have their blocks allocated out of a block group specific preallocation pool, so that small files are packed closely together. Each large file will have its blocks allocated out of its own unique preallocation pool.

session_write_kbytes
This file is read-only and shows the number of kilobytes of data that have been written to this filesystem since it was mounted.

reserved_clusters
This is RW file and contains number of reserved clusters in the file system which will be used in the specific situations to avoid costly zeroout, unexpected ENOSPC, or possible data loss. The default is 2% or 4096 clusters, whichever is smaller and this can be changed however it can never exceed number of clusters in the file system. If there is not enough space for the reserved space when mounting the file mount will _not_ fail.

42.8 Ioctls

Ext4 implements various ioctls which can be used by applications to access ext4-specific functionality. An incomplete list of these ioctls is shown in the table below. This list includes truly ext4-specific ioctls (`EXT4_IOC_*`) as well as ioctls that may have been ext4-specific originally but are now supported by some other filesystem(s) too (`FS_IOC_*`).

Table of Ext4 ioctls

FS_IOC_GETFLAGS
Get additional attributes associated with inode. The ioctl argument is an integer bitfield, with bit values described in ext4.h.

FS_IOC_SETFLAGS
Set additional attributes associated with inode. The ioctl argument is an integer bitfield, with bit values described in ext4.h.

EXT4_IOC_GETVERSION, EXT4_IOC_GETVERSION_OLD
Get the inode i_generation number stored for each inode. The i_generation number is normally changed only when new inode is created and it is particularly useful for network filesystems. The '_OLD' version of this ioctl is an alias for FS_IOC_GETVERSION.

EXT4_IOC_SETVERSION, EXT4_IOC_SETVERSION_OLD
Set the inode i_generation number stored for each inode. The '_OLD' version of this ioctl is an alias for FS_IOC_SETVERSION.

EXT4_IOC_GROUP_EXTEND
This ioctl has the same purpose as the resize mount option. It allows to resize filesystem to the end of the last existing block group, further resize has to be done with resize2fs, either online, or offline. The argument points to the unsigned logn number representing the filesystem new block count.

EXT4_IOC_MOVE_EXT
Move the block extents from orig_fd (the one this ioctl is pointing to) to the donor_fd (the one specified in move_extent structure passed as an argument to this ioctl). Then, exchange inode metadata between orig_fd and donor_fd. This is especially useful for online defragmentation, because the allocator has the opportunity to allocate moved blocks better, ideally into one contiguous extent.

EXT4_IOC_GROUP_ADD
Add a new group descriptor to an existing or new group descriptor block. The new group descriptor is described by ext4_new_group_input structure, which is passed as an argument to this ioctl. This is especially useful in conjunction with EXT4_IOC_GROUP_EXTEND, which allows online resize of the filesystem to the end of the last existing block group. Those two ioctls combined is used in userspace online resize tool (e.g. resize2fs).

EXT4_IOC_MIGRATE
This ioctl operates on the filesystem itself. It converts (migrates) ext3 indirect block mapped inode to ext4 extent mapped inode by walking through indirect block mapping of the original inode and converting contiguous block ranges into ext4 extents of the temporary inode. Then, inodes are swapped. This ioctl might help, when migrating from ext3 to ext4 filesystem, however suggestion is to create fresh ext4 filesystem and copy data from the backup. Note, that filesystem

has to support extents for this ioctl to work.

EXT4_IOC_ALLOC_DA_BLKS
Force all of the delay allocated blocks to be allocated to preserve application-expected ext3 behaviour. Note that this will also start triggering a write of the data blocks, but this behaviour may change in the future as it is not necessary and has been done this way only for sake of simplicity.

EXT4_IOC_RESIZE_FS
Resize the filesystem to a new size. The number of blocks of resized filesystem is passed in via 64 bit integer argument. The kernel allocates bitmaps and inode table, the userspace tool thus just passes the new number of blocks.

EXT4_IOC_SWAP_BOOT
Swap i_blocks and associated attributes (like i_blocks, i_size, i_flags, ...) from the specified inode with inode EXT4_BOOT_LOADER_INO (#5). This is typically used to store a boot loader in a secure part of the filesystem, where it can't be changed by a normal user by accident. The data blocks of the previous boot loader will be associated with the given inode.

42.9 References

kernel source: <file:fs/ext4/>
<file:fs/jbd2/>

programs: http://e2fsprogs.sourceforge.net/

useful links: https://fedoraproject.org/wiki/ext3-devel
http://www.bullopensource.org/ext4/ http://ext4.wiki.kernel.org/index.php/Main_Page https://fedoraproject.org/wiki/Features/Ext4

CHAPTER

FORTYTHREE

FILE SYSTEM MONITORING WITH FANOTIFY

43.1 File system Error Reporting

Fanotify supports the FAN_FS_ERROR event type for file system-wide error reporting. It is meant to be used by file system health monitoring daemons, which listen for these events and take actions (notify sysadmin, start recovery) when a file system problem is detected.

By design, a FAN_FS_ERROR notification exposes sufficient information for a monitoring tool to know a problem in the file system has happened. It doesn't necessarily provide a user space application with semantics to verify an IO operation was successfully executed. That is out of scope for this feature. Instead, it is only meant as a framework for early file system problem detection and reporting recovery tools.

When a file system operation fails, it is common for dozens of kernel errors to cascade after the initial failure, hiding the original failure log, which is usually the most useful debug data to troubleshoot the problem. For this reason, FAN_FS_ERROR tries to report only the first error that occurred for a file system since the last notification, and it simply counts additional errors. This ensures that the most important pieces of information are never lost.

FAN_FS_ERROR requires the fanotify group to be setup with the FAN_REPORT_FID flag.

At the time of this writing, the only file system that emits FAN_FS_ERROR notifications is Ext4.

A FAN_FS_ERROR Notification has the following format:

```
::

   [ Notification Metadata (Mandatory) ]
   [ Generic Error Record  (Mandatory) ]
   [ FID record            (Mandatory) ]
```

The order of records is not guaranteed, and new records might be added in the future. Therefore, applications must not rely on the order and must be prepared to skip over unknown records. Please refer to `samples/fanotify/fs-monitor.c` for an example parser.

43.1.1 Generic error record

The generic error record provides enough information for a file system agnostic tool to learn about a problem in the file system, without providing any additional details about the problem. This record is identified by `struct fanotify_event_info_header.info_type` being set to FAN_EVENT_INFO_TYPE_ERROR.

```
struct fanotify_event_info_error {
     struct fanotify_event_info_header hdr;
    __s32 error;
    __u32 error_count;
};
```

The *error* field identifies the type of error using errno values. *error_count* tracks the number of errors that occurred and were suppressed to preserve the original error information, since the last notification.

43.1.2 FID record

The FID record can be used to uniquely identify the inode that triggered the error through the combination of fsid and file handle. A file system specific application can use that information to attempt a recovery procedure. Errors that are not related to an inode are reported with an empty file handle of type FILEID_INVALID.

CHAPTER

FORTYFOUR

NFS

44.1 NFS Client

44.1.1 The NFS client

The NFS version 2 protocol was first documented in RFC1094 (March 1989). Since then two more major releases of NFS have been published, with NFSv3 being documented in RFC1813 (June 1995), and NFSv4 in RFC3530 (April 2003).

The Linux NFS client currently supports all the above published versions, and work is in progress on adding support for minor version 1 of the NFSv4 protocol.

The purpose of this document is to provide information on some of the special features of the NFS client that can be configured by system administrators.

44.1.2 The nfs4_unique_id parameter

NFSv4 requires clients to identify themselves to servers with a unique string. File open and lock state shared between one client and one server is associated with this identity. To support robust NFSv4 state recovery and transparent state migration, this identity string must not change across client reboots.

Without any other intervention, the Linux client uses a string that contains the local system's node name. System administrators, however, often do not take care to ensure that node names are fully qualified and do not change over the lifetime of a client system. Node names can have other administrative requirements that require particular behavior that does not work well as part of an nfs_client_id4 string.

The nfs.nfs4_unique_id boot parameter specifies a unique string that can be used together with a system's node name when an NFS client identifies itself to a server. Thus, if the system's node name is not unique, its nfs.nfs4_unique_id can help prevent collisions with other clients.

The nfs.nfs4_unique_id string is typically a UUID, though it can contain anything that is believed to be unique across all NFS clients. An nfs4_unique_id string should be chosen when a client system is installed, just as a system's root file system gets a fresh UUID in its label at install time.

The string should remain fixed for the lifetime of the client. It can be changed safely if care is taken that the client shuts down cleanly and all outstanding NFSv4 state has expired, to prevent loss of NFSv4 state.

This string can be stored in an NFS client's grub.conf, or it can be provided via a net boot facility such as PXE. It may also be specified as an nfs.ko module parameter.

This uniquifier string will be the same for all NFS clients running in containers unless it is overridden by a value written to /sys/fs/nfs/net/nfs_client/identifier which will be local to the network namespace of the process which writes.

44.1.3 The DNS resolver

NFSv4 allows for one server to refer the NFS client to data that has been migrated onto another server by means of the special "fs_locations" attribute. See RFC3530 Section 6: Filesystem Migration and Replication and Implementation Guide for Referrals in NFSv4.

The fs_locations information can take the form of either an ip address and a path, or a DNS hostname and a path. The latter requires the NFS client to do a DNS lookup in order to mount the new volume, and hence the need for an upcall to allow userland to provide this service.

Assuming that the user has the 'rpc_pipefs' filesystem mounted in the usual /var/lib/nfs/rpc_pipefs, the upcall consists of the following steps:

(1) The process checks the dns_resolve cache to see if it contains a valid entry. If so, it returns that entry and exits.

(2) If no valid entry exists, the helper script '/sbin/nfs_cache_getent' (may be changed using the 'nfs.cache_getent' kernel boot parameter) is run, with two arguments: - the cache name, "dns_resolve" - the hostname to resolve

(3) After looking up the corresponding ip address, the helper script writes the result into the rpc_pipefs pseudo-file '/var/lib/nfs/rpc_pipefs/cache/dns_resolve/channel' in the following (text) format:

    "<ip address> <hostname> <ttl>n"

    Where <ip address> is in the usual IPv4 (123.456.78.90) or IPv6 (ffee:ddcc:bbaa:9988:7766:5544:3322:1100, ffee::1100, ...) format. <hostname> is identical to the second argument of the helper script, and <ttl> is the 'time to live' of this cache entry (in units of seconds).

    Note: If <ip address> is invalid, say the string "0", then a negative entry is created, which will cause the kernel to treat the hostname as having no valid DNS translation.

44.1.4 A basic sample /sbin/nfs_cache_getent

```
#!/bin/bash
#
ttl=600
#
cut=/usr/bin/cut
getent=/usr/bin/getent
rpc_pipefs=/var/lib/nfs/rpc_pipefs
#
die()
```

```
{
    echo "Usage: $0 cache_name entry_name"
    exit 1
}

[ $# -lt 2 ] && die
cachename="$1"
cache_path=${rpc_pipefs}/cache/${cachename}/channel

case "${cachename}" in
    dns_resolve)
        name="$2"
        result="$(${getent} hosts ${name} | ${cut} -f1 -d\ )"
        [ -z "${result}" ] && result="0"
        ;;
    *)
        die
        ;;
esac
echo "${result} ${name} ${ttl}" >${cache_path}
```

44.2 Mounting the root filesystem via NFS (nfsroot)

Authors

Written 1996 by Gero Kuhlmann <gero@gkminix.han.de>

Updated 1997 by Martin Mares <mj@atrey.karlin.mff.cuni.cz>

Updated 2006 by Nico Schottelius <nico-kernel-nfsroot@schottelius.org>

Updated 2006 by Horms <horms@verge.net.au>

Updated 2018 by Chris Novakovic <chris@chrisn.me.uk>

In order to use a diskless system, such as an X-terminal or printer server for example, it is necessary for the root filesystem to be present on a non-disk device. This may be an initramfs (see Documentation/filesystems/ramfs-rootfs-initramfs.rst), a ramdisk (see *Using the initial RAM disk (initrd)*) or a filesystem mounted via NFS. The following text describes on how to use NFS for the root filesystem. For the rest of this text 'client' means the diskless system, and 'server' means the NFS server.

44.2.1 Enabling nfsroot capabilities

In order to use nfsroot, NFS client support needs to be selected as built-in during configuration. Once this has been selected, the nfsroot option will become available, which should also be selected.

In the networking options, kernel level autoconfiguration can be selected, along with the types of autoconfiguration to support. Selecting all of DHCP, BOOTP and RARP is safe.

44.2.2 Kernel command line

When the kernel has been loaded by a boot loader (see below) it needs to be told what root fs device to use. And in the case of nfsroot, where to find both the server and the name of the directory on the server to mount as root. This can be established using the following kernel command line parameters:

root=/dev/nfs
This is necessary to enable the pseudo-NFS-device. Note that it's not a real device but just a synonym to tell the kernel to use NFS instead of a real device.

nfsroot=[<server-ip>:]<root-dir>[,<nfs-options>]
If the *nfsroot' parameter is NOT given on the command line, the default* `"/tftpboot/%s"`` will be used.

<server-ip> Specifies the IP address of the NFS server.
The default address is determined by the ip parameter (see below). This parameter allows the use of different servers for IP autoconfiguration and NFS.

<root-dir> Name of the directory on the server to mount as root.
If there is a "%s" token in the string, it will be replaced by the ASCII-representation of the client's IP address.

<nfs-options> Standard NFS options. All options are separated by commas.
The following defaults are used:

```
port            = as given by server portmap daemon
rsize           = 4096
wsize           = 4096
timeo           = 7
retrans         = 3
acregmin        = 3
acregmax        = 60
acdirmin        = 30
acdirmax        = 60
flags           = hard, nointr, noposix, cto, ac
```

ip=<client-ip>:<server-ip>:<gw-ip>:<netmask>:<hostname>:<device>:<autoconf>:<dns0-ip>:<dns1-ip>:<ntp0-ip>
This parameter tells the kernel how to configure IP addresses of devices and also how to set up the IP routing table. It was originally called nfsaddrs, but now the boot-time IP configuration works independently of NFS, so it was renamed to ip and the old name remained as an alias for compatibility reasons.

If this parameter is missing from the kernel command line, all fields are assumed to be empty, and the defaults mentioned below apply. In general this means that the kernel tries to configure everything using autoconfiguration.

The <autoconf> parameter can appear alone as the value to the ip parameter (without all the ':' characters before). If the value is "ip=off" or "ip=none", no autoconfiguration will take place, otherwise autoconfiguration will take place. The most common way to use this is "ip=dhcp".

<client-ip> IP address of the client.
Default: Determined using autoconfiguration.

<server-ip> IP address of the NFS server.
If RARP is used to determine the client address and this parameter is NOT empty only replies from the specified server are accepted.

Only required for NFS root. That is autoconfiguration will not be triggered if it is missing and NFS root is not in operation.

Value is exported to /proc/net/pnp with the prefix "bootserver " (see below).

Default: Determined using autoconfiguration. The address of the autoconfiguration server is used.

<gw-ip> IP address of a gateway if the server is on a different subnet.
Default: Determined using autoconfiguration.

<netmask> Netmask for local network interface.
If unspecified the netmask is derived from the client IP address assuming classful addressing.

Default: Determined using autoconfiguration.

<hostname> Name of the client.
If a '.' character is present, anything before the first '.' is used as the client's hostname, and anything after it is used as its NIS domain name. May be supplied by autoconfiguration, but its absence will not trigger autoconfiguration. If specified and DHCP is used, the user-provided hostname (and NIS domain name, if present) will be carried in the DHCP request; this may cause a DNS record to be created or updated for the client.

Default: Client IP address is used in ASCII notation.

<device> Name of network device to use.
Default: If the host only has one device, it is used. Otherwise the device is determined using autoconfiguration. This is done by sending autoconfiguration requests out of all devices, and using the device that received the first reply.

<autoconf> Method to use for autoconfiguration.
In the case of options which specify multiple autoconfiguration protocols, requests are sent using all protocols, and the first one to reply is used.

Only autoconfiguration protocols that have been compiled into the kernel will be used, regardless of the value of this option:

```
off or none: don't use autoconfiguration
              (do static IP assignment instead)
on or any:   use any protocol available in the kernel
             (default)
dhcp:        use DHCP
bootp:       use BOOTP
rarp:        use RARP
both:        use both BOOTP and RARP but not DHCP
             (old option kept for backwards compatibility)
```

if dhcp is used, the client identifier can be used by following format "ip=dhcp,client-id-type,client-id-value"

Default: any

<dns0-ip> IP address of primary nameserver.
Value is exported to /proc/net/pnp with the prefix ”nameserver ” (see below).

Default: None if not using autoconfiguration; determined automatically if using autoconfiguration.

<dns1-ip> IP address of secondary nameserver.
See <dns0-ip>.

<ntp0-ip> IP address of a Network Time Protocol (NTP) server.
Value is exported to /proc/net/ipconfig/ntp_servers, but is otherwise unused (see below).

Default: None if not using autoconfiguration; determined automatically if using autoconfiguration.

After configuration (whether manual or automatic) is complete, two files are created in the following format; lines are omitted if their respective value is empty following configuration:

- /proc/net/pnp:

  #PROTO: <DHCP|BOOTP|RARP|MANUAL> (depending on configuration method) domain <dns-domain> (if autoconfigured, the DNS domain) nameserver <dns0-ip> (primary name server IP) nameserver <dns1-ip> (secondary name server IP) nameserver <dns2-ip> (tertiary name server IP) bootserver <server-ip> (NFS server IP)

- /proc/net/ipconfig/ntp_servers:

  <ntp0-ip> (NTP server IP) <ntp1-ip> (NTP server IP) <ntp2-ip> (NTP server IP)

<dns-domain> and <dns2-ip> (in /proc/net/pnp) and <ntp1-ip> and <ntp2-ip> (in /proc/net/ipconfig/ntp_servers) are requested during autoconfiguration; they cannot be specified as part of the ”ip=” kernel command line parameter.

Because the ”domain” and ”nameserver” options are recognised by DNS resolvers, /etc/resolv.conf is often linked to /proc/net/pnp on systems that use an NFS root filesystem.

Note that the kernel will not synchronise the system time with any NTP servers it discovers; this is the responsibility of a user space process (e.g. an initrd/initramfs script that passes the IP addresses listed in /proc/net/ipconfig/ntp_servers to an NTP client before mounting the real root filesystem if it is on NFS).

nfsrootdebug
This parameter enables debugging messages to appear in the kernel log at boot time so that administrators can verify that the correct NFS mount options, server address, and root path are passed to the NFS client.

rdinit=<executable file>
To specify which file contains the program that starts system initialization, administrators can use this command line parameter. The default value of this parameter is ”/init”. If the specified file exists and the kernel can execute it, root filesystem related kernel command line parameters, including 'nfsroot=', are ignored.

A description of the process of mounting the root file system can be found in Documentation/driver-api/early-userspace/early_userspace_support.rst

44.2.3 Boot Loader

To get the kernel into memory different approaches can be used. They depend on various facilities being available:

- Booting from a floppy using syslinux

  When building kernels, an easy way to create a boot floppy that uses syslinux is to use the zdisk or bzdisk make targets which use zimage and bzimage images respectively. Both targets accept the FDARGS parameter which can be used to set the kernel command line.

  e.g:

  ```
  make bzdisk FDARGS="root=/dev/nfs"
  ```

  Note that the user running this command will need to have access to the floppy drive device, /dev/fd0

  For more information on syslinux, including how to create bootdisks for prebuilt kernels, see https://syslinux.zytor.com/

  Note: Previously it was possible to write a kernel directly to a floppy using dd, configure the boot device using rdev, and boot using the resulting floppy. Linux no longer supports this method of booting.

- Booting from a cdrom using isolinux

  When building kernels, an easy way to create a bootable cdrom that uses isolinux is to use the isoimage target which uses a bzimage image. Like zdisk and bzdisk, this target accepts the FDARGS parameter which can be used to set the kernel command line.

  e.g:

  ```
  make isoimage FDARGS="root=/dev/nfs"
  ```

  The resulting iso image will be arch/<ARCH>/boot/image.iso This can be written to a cdrom using a variety of tools including cdrecord.

  e.g:

  ```
  cdrecord dev=ATAPI:1,0,0 arch/x86/boot/image.iso
  ```

  For more information on isolinux, including how to create bootdisks for prebuilt kernels, see https://syslinux.zytor.com/

- Using LILO

  When using LILO all the necessary command line parameters may be specified using the 'append=' directive in the LILO configuration file.

  However, to use the 'root=' directive you also need to create a dummy root device, which may be removed after LILO is run.

  e.g:

```
mknod /dev/boot255 c 0 255
```

For information on configuring LILO, please refer to its documentation.

- Using GRUB

  When using GRUB, kernel parameter are simply appended after the kernel specification: kernel <kernel> <parameters>

- Using loadlin

  loadlin may be used to boot Linux from a DOS command prompt without requiring a local hard disk to mount as root. This has not been thoroughly tested by the authors of this document, but in general it should be possible configure the kernel command line similarly to the configuration of LILO.

  Please refer to the loadlin documentation for further information.

- Using a boot ROM

  This is probably the most elegant way of booting a diskless client. With a boot ROM the kernel is loaded using the TFTP protocol. The authors of this document are not aware of any no commercial boot ROMs that support booting Linux over the network. However, there are two free implementations of a boot ROM, netboot-nfs and etherboot, both of which are available on sunsite.unc.edu, and both of which contain everything you need to boot a diskless Linux client.

- Using pxelinux

  Pxelinux may be used to boot linux using the PXE boot loader which is present on many modern network cards.

  When using pxelinux, the kernel image is specified using "kernel <relative-path-below /tftpboot>". The nfsroot parameters are passed to the kernel by adding them to the "append" line. It is common to use serial console in conjunction with pxeliunx, see *Linux Serial Console* for more information.

  For more information on isolinux, including how to create bootdisks for prebuilt kernels, see https://syslinux.zytor.com/

44.2.4 Credits

The nfsroot code in the kernel and the RARP support have been written by Gero Kuhlmann <gero@gkminix.han.de>.

The rest of the IP layer autoconfiguration code has been written by Martin Mares <mj@atrey.karlin.mff.cuni.cz>.

In order to write the initial version of nfsroot I would like to thank Jens-Uwe Mager <jum@anubis.han.de> for his help.

44.3 Setting up NFS/RDMA

Author
NetApp and Open Grid Computing (May 29, 2008)

Warning: This document is probably obsolete.

44.3.1 Overview

This document describes how to install and setup the Linux NFS/RDMA client and server software.

The NFS/RDMA client was first included in Linux 2.6.24. The NFS/RDMA server was first included in the following release, Linux 2.6.25.

In our testing, we have obtained excellent performance results (full 10Gbit wire bandwidth at minimal client CPU) under many workloads. The code passes the full Connectathon test suite and operates over both Infiniband and iWARP RDMA adapters.

44.3.2 Getting Help

If you get stuck, you can ask questions on the nfs-rdma-devel@lists.sourceforge.net mailing list.

44.3.3 Installation

These instructions are a step by step guide to building a machine for use with NFS/RDMA.

- Install an RDMA device

  Any device supported by the drivers in drivers/infiniband/hw is acceptable.

  Testing has been performed using several Mellanox-based IB cards, the Ammasso AMS1100 iWARP adapter, and the Chelsio cxgb3 iWARP adapter.

- Install a Linux distribution and tools

  The first kernel release to contain both the NFS/RDMA client and server was Linux 2.6.25 Therefore, a distribution compatible with this and subsequent Linux kernel release should be installed.

  The procedures described in this document have been tested with distributions from Red Hat's Fedora Project (http://fedora.redhat.com/).

- Install nfs-utils-1.1.2 or greater on the client

  An NFS/RDMA mount point can be obtained by using the mount.nfs command in nfs-utils-1.1.2 or greater (nfs-utils-1.1.1 was the first nfs-utils version with support for NFS/RDMA mounts, but for various reasons we recommend using nfs-utils-1.1.2 or greater). To see which version of mount.nfs you are using, type:

  ```
  $ /sbin/mount.nfs -V
  ```

If the version is less than 1.1.2 or the command does not exist, you should install the latest version of nfs-utils.

Download the latest package from: https://www.kernel.org/pub/linux/utils/nfs

Uncompress the package and follow the installation instructions.

If you will not need the idmapper and gssd executables (you do not need these to create an NFS/RDMA enabled mount command), the installation process can be simplified by disabling these features when running configure:

```
$ ./configure --disable-gss --disable-nfsv4
```

To build nfs-utils you will need the tcp_wrappers package installed. For more information on this see the package's README and INSTALL files.

After building the nfs-utils package, there will be a mount.nfs binary in the utils/mount directory. This binary can be used to initiate NFS v2, v3, or v4 mounts. To initiate a v4 mount, the binary must be called mount.nfs4. The standard technique is to create a symlink called mount.nfs4 to mount.nfs.

This mount.nfs binary should be installed at /sbin/mount.nfs as follows:

```
$ sudo cp utils/mount/mount.nfs /sbin/mount.nfs
```

In this location, mount.nfs will be invoked automatically for NFS mounts by the system mount command.

> **Note:** mount.nfs and therefore nfs-utils-1.1.2 or greater is only needed on the NFS client machine. You do not need this specific version of nfs-utils on the server. Furthermore, only the mount.nfs command from nfs-utils-1.1.2 is needed on the client.

- Install a Linux kernel with NFS/RDMA

  The NFS/RDMA client and server are both included in the mainline Linux kernel version 2.6.25 and later. This and other versions of the Linux kernel can be found at: https://www.kernel.org/pub/linux/kernel/

  Download the sources and place them in an appropriate location.

- Configure the RDMA stack

  Make sure your kernel configuration has RDMA support enabled. Under Device Drivers -> InfiniBand support, update the kernel configuration to enable InfiniBand support [NOTE: the option name is misleading. Enabling InfiniBand support is required for all RDMA devices (IB, iWARP, etc.)].

  Enable the appropriate IB HCA support (mlx4, mthca, ehca, ipath, etc.) or iWARP adapter support (amso, cxgb3, etc.).

  If you are using InfiniBand, be sure to enable IP-over-InfiniBand support.

- Configure the NFS client and server

  Your kernel configuration must also have NFS file system support and/or NFS server support enabled. These and other NFS related configuration options can be found under File

Systems -> Network File Systems.

- Build, install, reboot

  The NFS/RDMA code will be enabled automatically if NFS and RDMA are turned on. The NFS/RDMA client and server are configured via the hidden SUNRPC_XPRT_RDMA config option that depends on SUNRPC and INFINIBAND. The value of SUNRPC_XPRT_RDMA will be:

  1. N if either SUNRPC or INFINIBAND are N, in this case the NFS/RDMA client and server will not be built
  2. M if both SUNRPC and INFINIBAND are on (M or Y) and at least one is M, in this case the NFS/RDMA client and server will be built as modules
  3. Y if both SUNRPC and INFINIBAND are Y, in this case the NFS/RDMA client and server will be built into the kernel

  Therefore, if you have followed the steps above and turned no NFS and RDMA, the NFS/RDMA client and server will be built.

  Build a new kernel, install it, boot it.

44.3.4 Check RDMA and NFS Setup

Before configuring the NFS/RDMA software, it is a good idea to test your new kernel to ensure that the kernel is working correctly. In particular, it is a good idea to verify that the RDMA stack is functioning as expected and standard NFS over TCP/IP and/or UDP/IP is working properly.

- Check RDMA Setup

  If you built the RDMA components as modules, load them at this time. For example, if you are using a Mellanox Tavor/Sinai/Arbel card:

```
$ modprobe ib_mthca
$ modprobe ib_ipoib
```

  If you are using InfiniBand, make sure there is a Subnet Manager (SM) running on the network. If your IB switch has an embedded SM, you can use it. Otherwise, you will need to run an SM, such as OpenSM, on one of your end nodes.

  If an SM is running on your network, you should see the following:

```
$ cat /sys/class/infiniband/driverX/ports/1/state
4: ACTIVE
```

  where driverX is mthca0, ipath5, ehca3, etc.

  To further test the InfiniBand software stack, use IPoIB (this assumes you have two IB hosts named host1 and host2):

```
host1$ ip link set dev ib0 up
host1$ ip address add dev ib0 a.b.c.x
host2$ ip link set dev ib0 up
host2$ ip address add dev ib0 a.b.c.y
host1$ ping a.b.c.y
host2$ ping a.b.c.x
```

For other device types, follow the appropriate procedures.

- Check NFS Setup

  For the NFS components enabled above (client and/or server), test their functionality over standard Ethernet using TCP/IP or UDP/IP.

44.3.5 NFS/RDMA Setup

We recommend that you use two machines, one to act as the client and one to act as the server.

One time configuration:

- On the server system, configure the /etc/exports file and start the NFS/RDMA server.

  Exports entries with the following formats have been tested:

```
/vol0   192.168.0.47(fsid=0,rw,async,insecure,no_root_squash)
/vol0   192.168.0.0/255.255.255.0(fsid=0,rw,async,insecure,no_root_squash)
```

  The IP address(es) is(are) the client's IPoIB address for an InfiniBand HCA or the client's iWARP address(es) for an RNIC.

  Note: The ”insecure” option must be used because the NFS/RDMA client does not use a reserved port.

Each time a machine boots:

- Load and configure the RDMA drivers

  For InfiniBand using a Mellanox adapter:

```
$ modprobe ib_mthca
$ modprobe ib_ipoib
$ ip li set dev ib0 up
$ ip addr add dev ib0 a.b.c.d
```

  Note: Please use unique addresses for the client and server!

- Start the NFS server

  If the NFS/RDMA server was built as a module (CONFIG_SUNRPC_XPRT_RDMA=m in kernel config), load the RDMA transport module:

```
$ modprobe svcrdma
```

  Regardless of how the server was built (module or built-in), start the server:

```
$ /etc/init.d/nfs start
```

or

```
$ service nfs start
```

Instruct the server to listen on the RDMA transport:

```
$ echo rdma 20049 > /proc/fs/nfsd/portlist
```

- On the client system

  If the NFS/RDMA client was built as a module (CONFIG_SUNRPC_XPRT_RDMA=m in kernel config), load the RDMA client module:

  ```
  $ modprobe xprtrdma.ko
  ```

  Regardless of how the client was built (module or built-in), use this command to mount the NFS/RDMA server:

  ```
  $ mount -o rdma,port=20049 <IPoIB-server-name-or-address>:/<export> /mnt
  ```

  To verify that the mount is using RDMA, run ”cat /proc/mounts” and check the ”proto” field for the given mount.

  Congratulations! You're using NFS/RDMA!

44.4 Administrative interfaces for nfsd

Note that normally these interfaces are used only by the utilities in nfs-utils.

nfsd is controlled mainly by pseudofiles under the ”nfsd” filesystem, which is normally mounted at /proc/fs/nfsd/.

The server is always started by the first write of a nonzero value to nfsd/threads.

Before doing that, NFSD can be told which sockets to listen on by writing to nfsd/portlist; that write may be:

- an ascii-encoded file descriptor, which should refer to a bound (and listening, for tcp) socket, or
- ”transportname port”, where transportname is currently either ”udp”, ”tcp”, or ”rdma”.

If nfsd is started without doing any of these, then it will create one udp and one tcp listener at port 2049 (see nfsd_init_socks).

On startup, nfsd and lockd grace periods start. nfsd is shut down by a write of 0 to nfsd/threads. All locks and state are thrown away at that point.

Between startup and shutdown, the number of threads may be adjusted up or down by additional writes to nfsd/threads or by writes to nfsd/pool_threads.

For more detail about files under nfsd/ and what they control, see fs/nfsd/nfsctl.c; most of them have detailed comments.

44.4.1 Implementation notes

Note that the rpc server requires the caller to serialize addition and removal of listening sockets, and startup and shutdown of the server. For nfsd this is done using nfsd_mutex.

44.5 NFS ID Mapper

Id mapper is used by NFS to translate user and group ids into names, and to translate user and group names into ids. Part of this translation involves performing an upcall to userspace to request the information. There are two ways NFS could obtain this information: placing a call to /sbin/request-key or by placing a call to the rpc.idmap daemon.

NFS will attempt to call /sbin/request-key first. If this succeeds, the result will be cached using the generic request-key cache. This call should only fail if /etc/request-key.conf is not configured for the id_resolver key type, see the "Configuring" section below if you wish to use the request-key method.

If the call to /sbin/request-key fails (if /etc/request-key.conf is not configured with the id_resolver key type), then the idmapper will ask the legacy rpc.idmap daemon for the id mapping. This result will be stored in a custom NFS idmap cache.

44.5.1 Configuring

The file /etc/request-key.conf will need to be modified so /sbin/request-key can direct the upcall. The following line should be added:

```
#OP TYPE DESCRIPTION CALLOUT INFO PROGRAM ARG1 ARG2 ARG3 ...     #====== =======
=============== =============== ================================           create
id_resolver * * /usr/sbin/nfs.idmap %k %d 600
```

This will direct all id_resolver requests to the program /usr/sbin/nfs.idmap. The last parameter, 600, defines how many seconds into the future the key will expire. This parameter is optional for /usr/sbin/nfs.idmap. When the timeout is not specified, nfs.idmap will default to 600 seconds.

id mapper uses for key descriptions:

```
  uid:  Find the UID for the given user
  gid:  Find the GID for the given group
 user:  Find the user  name for the given UID
group:  Find the group name for the given GID
```

You can handle any of these individually, rather than using the generic upcall program. If you would like to use your own program for a uid lookup then you would edit your request-key.conf so it look similar to this:

```
#OP TYPE DESCRIPTION CALLOUT INFO PROGRAM ARG1 ARG2 ARG3 ...     #====== =======
=============== =============== ================================           create
id_resolver uid:* * /some/other/program %k %d 600   create id_resolver * * /usr/
sbin/nfs.idmap %k %d 600
```

Notice that the new line was added above the line for the generic program. request-key will find the first matching line and corresponding program. In this case, /some/other/program will handle all uid lookups and /usr/sbin/nfs.idmap will handle gid, user, and group lookups.

See Documentation/security/keys/request-key.rst for more information about the request-key function.

44.5.2 nfs.idmap

nfs.idmap is designed to be called by request-key, and should not be run "by hand". This program takes two arguments, a serialized key and a key description. The serialized key is first converted into a key_serial_t, and then passed as an argument to keyctl_instantiate (both are part of keyutils.h).

The actual lookups are performed by functions found in nfsidmap.h. nfs.idmap determines the correct function to call by looking at the first part of the description string. For example, a uid lookup description will appear as "uid:user@domain".

nfs.idmap will return 0 if the key was instantiated, and non-zero otherwise.

44.6 pNFS block layout server user guide

The Linux NFS server now supports the pNFS block layout extension. In this case the NFS server acts as Metadata Server (MDS) for pNFS, which in addition to handling all the metadata access to the NFS export also hands out layouts to the clients to directly access the underlying block devices that are shared with the client.

To use pNFS block layouts with the Linux NFS server the exported file system needs to support the pNFS block layouts (currently just XFS), and the file system must sit on shared storage (typically iSCSI) that is accessible to the clients in addition to the MDS. As of now the file system needs to sit directly on the exported volume, striping or concatenation of volumes on the MDS and clients is not supported yet.

On the server, pNFS block volume support is automatically if the file system support it. On the client make sure the kernel has the CONFIG_PNFS_BLOCK option enabled, the blkmapd daemon from nfs-utils is running, and the file system is mounted using the NFSv4.1 protocol version (mount -o vers=4.1).

If the nfsd server needs to fence a non-responding client it calls /sbin/nfsd-recall-failed with the first argument set to the IP address of the client, and the second argument set to the device node without the /dev prefix for the file system to be fenced. Below is an example file that shows how to translate the device into a serial number from SCSI EVPD 0x80:

```
cat > /sbin/nfsd-recall-failed << EOF
```

```
#!/bin/sh

CLIENT="$1"
DEV="/dev/$2"
EVPD=`sg_inq --page=0x80 ${DEV} | \
        grep "Unit serial number:" | \
        awk -F ': ' '{print $2}'`

echo "fencing client ${CLIENT} serial ${EVPD}" >> /var/log/pnfsd-fence.log
EOF
```

44.7 pNFS SCSI layout server user guide

This document describes support for pNFS SCSI layouts in the Linux NFS server. With pNFS SCSI layouts, the NFS server acts as Metadata Server (MDS) for pNFS, which in addition to handling all the metadata access to the NFS export, also hands out layouts to the clients so that they can directly access the underlying SCSI LUNs that are shared with the client.

To use pNFS SCSI layouts with the Linux NFS server, the exported file system needs to support the pNFS SCSI layouts (currently just XFS), and the file system must sit on a SCSI LUN that is accessible to the clients in addition to the MDS. As of now the file system needs to sit directly on the exported LUN, striping or concatenation of LUNs on the MDS and clients is not supported yet.

On a server built with CONFIG_NFSD_SCSI, the pNFS SCSI volume support is automatically enabled if the file system is exported using the ”pnfs” option and the underlying SCSI device support persistent reservations. On the client make sure the kernel has the CONFIG_PNFS_BLOCK option enabled, and the file system is mounted using the NFSv4.1 protocol version (mount -o vers=4.1).

CHAPTER

FORTYFIVE

GPIO

45.1 GPIO Aggregator

The GPIO Aggregator provides a mechanism to aggregate GPIOs, and expose them as a new gpio_chip. This supports the following use cases.

45.1.1 Aggregating GPIOs using Sysfs

GPIO controllers are exported to userspace using /dev/gpiochip* character devices. Access control to these devices is provided by standard UNIX file system permissions, on an all-or-nothing basis: either a GPIO controller is accessible for a user, or it is not.

The GPIO Aggregator provides access control for a set of one or more GPIOs, by aggregating them into a new gpio_chip, which can be assigned to a group or user using standard UNIX file ownership and permissions. Furthermore, this simplifies and hardens exporting GPIOs to a virtual machine, as the VM can just grab the full GPIO controller, and no longer needs to care about which GPIOs to grab and which not, reducing the attack surface.

Aggregated GPIO controllers are instantiated and destroyed by writing to write-only attribute files in sysfs.

/sys/bus/platform/drivers/gpio-aggregator/

"new_device" ...

Userspace may ask the kernel to instantiate an aggregated GPIO controller by writing a string describing the GPIOs to aggregate to the "new_device" file, using the format

```
[<gpioA>] [<gpiochipB> <offsets>] ...
```

Where:

"<gpioA>" ...

is a GPIO line name,

"<gpiochipB>" ...

is a GPIO chip label, and

"<offsets>" ...

is a comma-separated list of GPIO offsets and/or GPIO offset ranges denoted by dashes.

Example: Instantiate a new GPIO aggregator by aggregating GPIO line 19 of "e6052000.gpio" and GPIO lines 20-21 of "e6050000.gpio" into a new gpio_chip:

```
$ echo 'e6052000.gpio 19 e6050000.gpio 20-21' > new_device
```

"delete_device" ...
Userspace may ask the kernel to destroy an aggregated GPIO controller after use by writing its device name to the "delete_device" file.

Example: Destroy the previously-created aggregated GPIO controller, assumed to be "gpio-aggregator.0":

```
$ echo gpio-aggregator.0 > delete_device
```

45.1.2 Generic GPIO Driver

The GPIO Aggregator can also be used as a generic driver for a simple GPIO-operated device described in DT, without a dedicated in-kernel driver. This is useful in industrial control, and is not unlike e.g. spidev, which allows the user to communicate with an SPI device from userspace.

Binding a device to the GPIO Aggregator is performed either by modifying the gpio-aggregator driver, or by writing to the "driver_override" file in Sysfs.

Example: If "door" is a GPIO-operated device described in DT, using its own compatible value:

```
door {
        compatible = "myvendor,mydoor";

        gpios = <&gpio2 19 GPIO_ACTIVE_HIGH>,
                <&gpio2 20 GPIO_ACTIVE_LOW>;
        gpio-line-names = "open", "lock";
};
```

it can be bound to the GPIO Aggregator by either:

1. Adding its compatible value to `gpio_aggregator_dt_ids[]`,
2. Binding manually using "driver_override":

```
$ echo gpio-aggregator > /sys/bus/platform/devices/door/driver_override
$ echo door > /sys/bus/platform/drivers/gpio-aggregator/bind
```

After that, a new gpiochip "door" has been created:

```
$ gpioinfo door
gpiochip12 - 2 lines:
        line   0:       "open"       unused   input  active-high
        line   1:       "lock"       unused   input  active-high
```

45.2 GPIO Sysfs Interface for Userspace

Warning: THIS ABI IS DEPRECATED, THE ABI DOCUMENTATION HAS BEEN MOVED TO Documentation/ABI/obsolete/sysfs-gpio AND NEW USERSPACE CONSUMERS ARE SUPPOSED TO USE THE CHARACTER DEVICE ABI. THIS OLD SYSFS ABI WILL NOT BE DEVELOPED (NO NEW FEATURES), IT WILL JUST BE MAINTAINED.

Refer to the examples in tools/gpio/* for an introduction to the new character device ABI. Also see the userspace header in include/uapi/linux/gpio.h

45.2.1 The deprecated sysfs ABI

Platforms which use the "gpiolib" implementors framework may choose to configure a sysfs user interface to GPIOs. This is different from the debugfs interface, since it provides control over GPIO direction and value instead of just showing a gpio state summary. Plus, it could be present on production systems without debugging support.

Given appropriate hardware documentation for the system, userspace could know for example that GPIO #23 controls the write protect line used to protect boot loader segments in flash memory. System upgrade procedures may need to temporarily remove that protection, first importing a GPIO, then changing its output state, then updating the code before re-enabling the write protection. In normal use, GPIO #23 would never be touched, and the kernel would have no need to know about it.

Again depending on appropriate hardware documentation, on some systems userspace GPIO can be used to determine system configuration data that standard kernels won't know about. And for some tasks, simple userspace GPIO drivers could be all that the system really needs.

DO NOT ABUSE SYSFS TO CONTROL HARDWARE THAT HAS PROPER KERNEL DRIVERS. PLEASE READ THE DOCUMENT AT Documentation/driver-api/gpio/drivers-on-gpio.rst TO AVOID REINVENTING KERNEL WHEELS IN USERSPACE. I MEAN IT. REALLY.

45.2.2 Paths in Sysfs

There are three kinds of entries in /sys/class/gpio:

- Control interfaces used to get userspace control over GPIOs;
- GPIOs themselves; and
- GPIO controllers ("gpio_chip" instances).

That's in addition to standard files including the "device" symlink.

The control interfaces are write-only:

/sys/class/gpio/

"export" ...
Userspace may ask the kernel to export control of a GPIO to userspace by writing its number to this file.

Example: ”echo 19 > export” will create a ”gpio19” node for GPIO #19, if that's not requested by kernel code.

”unexport” ...
Reverses the effect of exporting to userspace.

Example: ”echo 19 > unexport” will remove a ”gpio19” node exported using the ”export” file.

GPIO signals have paths like /sys/class/gpio/gpio42/ (for GPIO #42) and have the following read/write attributes:

/sys/class/gpio/gpioN/

”direction” ...
reads as either ”in” or ”out”. This value may normally be written. Writing as ”out” defaults to initializing the value as low. To ensure glitch free operation, values ”low” and ”high” may be written to configure the GPIO as an output with that initial value.

Note that this attribute *will not exist* if the kernel doesn't support changing the direction of a GPIO, or it was exported by kernel code that didn't explicitly allow userspace to reconfigure this GPIO's direction.

”value” ...
reads as either 0 (low) or 1 (high). If the GPIO is configured as an output, this value may be written; any nonzero value is treated as high.

If the pin can be configured as interrupt-generating interrupt and if it has been configured to generate interrupts (see the description of ”edge”), you can poll(2) on that file and poll(2) will return whenever the interrupt was triggered. If you use poll(2), set the events POLLPRI and POLLERR. If you use select(2), set the file descriptor in exceptfds. After poll(2) returns, either lseek(2) to the beginning of the sysfs file and read the new value or close the file and re-open it to read the value.

”edge” ...
reads as either ”none”, ”rising”, ”falling”, or ”both”. Write these strings to select the signal edge(s) that will make poll(2) on the ”value” file return.

This file exists only if the pin can be configured as an interrupt generating input pin.

”active_low” ...
reads as either 0 (false) or 1 (true). Write any nonzero value to invert the value attribute both for reading and writing. Existing and subsequent poll(2) support configuration via the edge attribute for ”rising” and ”falling” edges will follow this setting.

GPIO controllers have paths like /sys/class/gpio/gpiochip42/ (for the controller implementing GPIOs starting at #42) and have the following read-only attributes:

/sys/class/gpio/gpiochipN/

”base” ...
same as N, the first GPIO managed by this chip

"label" ...
provided for diagnostics (not always unique)

"ngpio" ...
how many GPIOs this manages (N to N + ngpio - 1)

Board documentation should in most cases cover what GPIOs are used for what purposes. However, those numbers are not always stable; GPIOs on a daughtercard might be different depending on the base board being used, or other cards in the stack. In such cases, you may need to use the gpiochip nodes (possibly in conjunction with schematics) to determine the correct GPIO number to use for a given signal.

45.2.3 Exporting from Kernel code

Kernel code can explicitly manage exports of GPIOs which have already been requested using gpio_request():

```
/* export the GPIO to userspace */
int gpiod_export(struct gpio_desc *desc, bool direction_may_change);

/* reverse gpiod_export() */
void gpiod_unexport(struct gpio_desc *desc);

/* create a sysfs link to an exported GPIO node */
int gpiod_export_link(struct device *dev, const char *name,
              struct gpio_desc *desc);
```

After a kernel driver requests a GPIO, it may only be made available in the sysfs interface by gpiod_export(). The driver can control whether the signal direction may change. This helps drivers prevent userspace code from accidentally clobbering important system state.

This explicit exporting can help with debugging (by making some kinds of experiments easier), or can provide an always-there interface that's suitable for documenting as part of a board support package.

After the GPIO has been exported, gpiod_export_link() allows creating symlinks from elsewhere in sysfs to the GPIO sysfs node. Drivers can use this to provide the interface under their own device in sysfs with a descriptive name.

45.3 GPIO Testing Driver

The GPIO Testing Driver (gpio-mockup) provides a way to create simulated GPIO chips for testing purposes. The lines exposed by these chips can be accessed using the standard GPIO character device interface as well as manipulated using the dedicated debugfs directory structure.

45.3.1 Creating simulated chips using module params

When loading the gpio-mockup driver a number of parameters can be passed to the module.

gpio_mockup_ranges

This parameter takes an argument in the form of an array of integer pairs. Each pair defines the base GPIO number (non-negative integer) and the first number after the last of this chip. If the base GPIO is -1, the gpiolib will assign it automatically. while the following parameter is the number of lines exposed by the chip.

Example: gpio_mockup_ranges=-1,8,-1,16,405,409

The line above creates three chips. The first one will expose 8 lines, the second 16 and the third 4. The base GPIO for the third chip is set to 405 while for two first chips it will be assigned automatically.

gpio_mockup_named_lines

This parameter doesn't take any arguments. It lets the driver know that GPIO lines exposed by it should be named.

The name format is: gpio-mockup-X-Y where X is mockup chip's ID and Y is the line offset.

45.3.2 Manipulating simulated lines

Each mockup chip creates its own subdirectory in /sys/kernel/debug/gpio-mockup/. The directory is named after the chip's label. A symlink is also created, named after the chip's name, which points to the label directory.

Inside each subdirectory, there's a separate attribute for each GPIO line. The name of the attribute represents the line's offset in the chip.

Reading from a line attribute returns the current value. Writing to it (0 or 1) changes the configuration of the simulated pull-up/pull-down resistor (1 - pull-up, 0 - pull-down).

45.4 Configfs GPIO Simulator

The configfs GPIO Simulator (gpio-sim) provides a way to create simulated GPIO chips for testing purposes. The lines exposed by these chips can be accessed using the standard GPIO character device interface as well as manipulated using sysfs attributes.

45.4.1 Creating simulated chips

The gpio-sim module registers a configfs subsystem called `'gpio-sim'`. For details of the configfs filesystem, please refer to the configfs documentation.

The user can create a hierarchy of configfs groups and items as well as modify values of exposed attributes. Once the chip is instantiated, this hierarchy will be translated to appropriate device properties. The general structure is:

Group: `/config/gpio-sim`

This is the top directory of the gpio-sim configfs tree.

Group: `/config/gpio-sim/gpio-device`

Attribute: `/config/gpio-sim/gpio-device/dev_name`

Attribute: `/config/gpio-sim/gpio-device/live`

This is a directory representing a GPIO platform device. The `'dev_name'` attribute is read-only and allows the user-space to read the platform device name (e.g. `'gpio-sim.0'`). The `'live'` attribute allows to trigger the actual creation of the device once it's fully configured. The accepted values are: `'1'` to enable the simulated device and `'0'` to disable and tear it down.

Group: `/config/gpio-sim/gpio-device/gpio-bankX`

Attribute: `/config/gpio-sim/gpio-device/gpio-bankX/chip_name`

Attribute: `/config/gpio-sim/gpio-device/gpio-bankX/num_lines`

This group represents a bank of GPIOs under the top platform device. The `'chip_name'` attribute is read-only and allows the user-space to read the device name of the bank device. The `'num_lines'` attribute allows to specify the number of lines exposed by this bank.

Group: `/config/gpio-sim/gpio-device/gpio-bankX/lineY`

Attribute: `/config/gpio-sim/gpio-device/gpio-bankX/lineY/name`

This group represents a single line at the offset Y. The 'name' attribute allows to set the line name as represented by the 'gpio-line-names' property.

Item: `/config/gpio-sim/gpio-device/gpio-bankX/lineY/hog`

Attribute: `/config/gpio-sim/gpio-device/gpio-bankX/lineY/hog/name`

Attribute: `/config/gpio-sim/gpio-device/gpio-bankX/lineY/hog/direction`

This item makes the gpio-sim module hog the associated line. The `'name'` attribute specifies the in-kernel consumer name to use. The `'direction'` attribute specifies the hog direction and must be one of: `'input'`, `'output-high'` and `'output-low'`.

Inside each bank directory, there's a set of attributes that can be used to configure the new chip. Additionally the user can `mkdir()` subdirectories inside the chip's directory that allow to pass additional configuration for specific lines. The name of those subdirectories must take the form of: `'line<offset>'` (e.g. `'line0'`, `'line20'`, etc.) as the name will be used by the module to assign the config to the specific line at given offset.

Once the confiuration is complete, the `'live'` attribute must be set to 1 in order to instantiate the chip. It can be set back to 0 to destroy the simulated chip. The module will synchronously

wait for the new simulated device to be successfully probed and if this doesn't happen, writing to `'live'` will result in an error.

Simulated GPIO chips can also be defined in device-tree. The compatible string must be: `"gpio-simulator"`. Supported properties are:

`"gpio-sim,label"` - chip label

Other standard GPIO properties (like `"gpio-line-names"`, `"ngpios"` or `"gpio-hog"`) are also supported. Please refer to the GPIO documentation for details.

An example device-tree code defining a GPIO simulator:

```
gpio-sim {
    compatible = "gpio-simulator";

    bank0 {
        gpio-controller;
        #gpio-cells = <2>;
        ngpios = <16>;
        gpio-sim,label = "dt-bank0";
        gpio-line-names = "", "sim-foo", "", "sim-bar";
    };

    bank1 {
        gpio-controller;
        #gpio-cells = <2>;
        ngpios = <8>;
        gpio-sim,label = "dt-bank1";

        line3 {
            gpio-hog;
            gpios = <3 0>;
            output-high;
            line-name = "sim-hog-from-dt";
        };
    };
};
```

45.4.2 Manipulating simulated lines

Each simulated GPIO chip creates a separate sysfs group under its device directory for each exposed line (e.g. `/sys/devices/platform/gpio-sim.X/gpiochipY/`). The name of each group is of the form: `'sim_gpioX'` where X is the offset of the line. Inside each group there are two attributes:

`pull` - allows to read and set the current simulated pull setting for
every line, when writing the value must be one of: `'pull-up'`, `'pull-down'`

`value` - allows to read the current value of the line which may be
different from the pull if the line is being driven from user-space

CHAPTER

FORTYSIX

NOTES ON THE CHANGE FROM 16-BIT UIDS TO 32-BIT UIDS

Author
Chris Wing <wingc@umich.edu>

Last updated
January 11, 2000

- kernel code MUST take into account __kernel_uid_t and __kernel_uid32_t when communicating between user and kernel space in an ioctl or data structure.
- kernel code should use uid_t and gid_t in kernel-private structures and code.

What's left to be done for 32-bit UIDs on all Linux architectures:

- Disk quotas have an interesting limitation that is not related to the maximum UID/GID. They are limited by the maximum file size on the underlying filesystem, because quota records are written at offsets corresponding to the UID in question. Further investigation is needed to see if the quota system can cope properly with huge UIDs. If it can deal with 64-bit file offsets on all architectures, this should not be a problem.
- Decide whether or not to keep backwards compatibility with the system accounting file, or if we should break it as the comments suggest (currently, the old 16-bit UID and GID are still written to disk, and part of the former pad space is used to store separate 32-bit UID and GID)
- Need to validate that OS emulation calls the 16-bit UID compatibility syscalls, if the OS being emulated used 16-bit UIDs, or uses the 32-bit UID system calls properly otherwise.

  This affects at least:

  - iBCS on Intel
  - sparc32 emulation on sparc64 (need to support whatever new 32-bit UID system calls are added to sparc32)
- Validate that all filesystems behave properly.

  At present, 32-bit UIDs _should_ work for:

  - ext2
  - ufs
  - isofs
  - nfs
  - coda

  - udf

  Ioctl() fixups have been made for:

  - ncpfs
  - smbfs

  Filesystems with simple fixups to prevent 16-bit UID wraparound:

  - minix
  - sysv
  - qnx4

  Other filesystems have not been checked yet.

- The ncpfs and smpfs filesystems cannot presently use 32-bit UIDs in all ioctl()s. Some new ioctl()s have been added with 32-bit UIDs, but more are needed. (as well as new user<->kernel data structures)
- The ELF core dump format only supports 16-bit UIDs on arm, i386, m68k, sh, and sparc32. Fixing this is probably not that important, but would require adding a new ELF section.
- The ioctl()s used to control the in-kernel NFS server only support 16-bit UIDs on arm, i386, m68k, sh, and sparc32.
- make sure that the UID mapping feature of AX25 networking works properly (it should be safe because it's always used a 32-bit integer to communicate between user and kernel)

CHAPTER

FORTYSEVEN

HARDWARE RANDOM NUMBER GENERATORS

47.1 Introduction

The hw_random framework is software that makes use of a special hardware feature on your CPU or motherboard, a Random Number Generator (RNG). The software has two parts: a core providing the /dev/hwrng character device and its sysfs support, plus a hardware-specific driver that plugs into that core.

To make the most effective use of these mechanisms, you should download the support software as well. Download the latest version of the ”rng-tools” package from:

https://github.com/nhorman/rng-tools

Those tools use /dev/hwrng to fill the kernel entropy pool, which is used internally and exported by the /dev/urandom and /dev/random special files.

47.2 Theory of operation

CHARACTER DEVICE. Using the standard open() and read() system calls, you can read random data from the hardware RNG device. This data is NOT CHECKED by any fitness tests, and could potentially be bogus (if the hardware is faulty or has been tampered with). Data is only output if the hardware ”has-data” flag is set, but nevertheless a security-conscious person would run fitness tests on the data before assuming it is truly random.

The rng-tools package uses such tests in ”rngd”, and lets you run them by hand with a ”rngtest” utility.

/dev/hwrng is char device major 10, minor 183.

CLASS DEVICE. There is a /sys/class/misc/hw_random node with two unique attributes, ”rng_available” and ”rng_current”. The ”rng_available” attribute lists the hardware-specific drivers available, while ”rng_current” lists the one which is currently connected to /dev/hwrng. If your system has more than one RNG available, you may change the one used by writing a name from the list in ”rng_available” into ”rng_current”.

Hardware driver for Intel/AMD/VIA Random Number Generators (RNG)

- Copyright 2000,2001 Jeff Garzik <jgarzik@pobox.com>
- Copyright 2000,2001 Philipp Rumpf <prumpf@mandrakesoft.com>

47.3 About the Intel RNG hardware, from the firmware hub datasheet

The Firmware Hub integrates a Random Number Generator (RNG) using thermal noise generated from inherently random quantum mechanical properties of silicon. When not generating new random bits the RNG circuitry will enter a low power state. Intel will provide a binary software driver to give third party software access to our RNG for use as a security feature. At this time, the RNG is only to be used with a system in an OS-present state.

47.4 Intel RNG Driver notes

FIXME: support poll(2)

Note: request_mem_region was removed, for three reasons:

1) Only one RNG is supported by this driver;
2) The location used by the RNG is a fixed location in MMIO-addressable memory;
3) users with properly working BIOS e820 handling will always have the region in which the RNG is located reserved, so request_mem_region calls always fail for proper setups. However, for people who use mem=XX, BIOS e820 information is **not** in /proc/iomem, and request_mem_region(RNG_ADDR) can succeed.

47.5 Driver details

Based on:
Intel 82802AB/82802AC Firmware Hub (FWH) Datasheet May 1999 Order Number: 290658-002 R

Intel 82802 Firmware Hub:
Random Number Generator Programmer's Reference Manual December 1999 Order Number: 298029-001 R

Intel 82802 Firmware HUB Random Number Generator Driver
Copyright (c) 2000 Matt Sottek <msottek@quiknet.com>

Special thanks to Matt Sottek. I did the ”guts”, he did the ”brains” and all the testing.

CHAPTER

FORTYEIGHT

USING THE INITIAL RAM DISK (INITRD)

Written 1996,2000 by Werner Almesberger <werner.almesberger@epfl.ch> and Hans Lermen <lermen@fgan.de>

initrd provides the capability to load a RAM disk by the boot loader. This RAM disk can then be mounted as the root file system and programs can be run from it. Afterwards, a new root file system can be mounted from a different device. The previous root (from initrd) is then moved to a directory and can be subsequently unmounted.

initrd is mainly designed to allow system startup to occur in two phases, where the kernel comes up with a minimum set of compiled-in drivers, and where additional modules are loaded from initrd.

This document gives a brief overview of the use of initrd. A more detailed discussion of the boot process can be found in[1].

48.1 Operation

When using initrd, the system typically boots as follows:

1) the boot loader loads the kernel and the initial RAM disk
2) the kernel converts initrd into a ”normal” RAM disk and frees the memory used by initrd
3) if the root device is not `/dev/ram0`, the old (deprecated) change_root procedure is followed. see the ”Obsolete root change mechanism” section below.
4) root device is mounted. if it is `/dev/ram0`, the initrd image is then mounted as root
5) /sbin/init is executed (this can be any valid executable, including shell scripts; it is run with uid 0 and can do basically everything init can do).
6) init mounts the ”real” root file system
7) init places the root file system at the root directory using the pivot_root system call
8) init execs the `/sbin/init` on the new root filesystem, performing the usual boot sequence
9) the initrd file system is removed

Note that changing the root directory does not involve unmounting it. It is therefore possible to leave processes running on initrd during that procedure. Also note that file systems mounted under initrd continue to be accessible.

[1] Almesberger, Werner; ”Booting Linux: The History and the Future” https://www.almesberger.net/cv/papers/ols2k-9.ps.gz

48.2 Boot command-line options

initrd adds the following new options:

```
initrd=<path>    (e.g. LOADLIN)

  Loads the specified file as the initial RAM disk. When using LILO, you
  have to specify the RAM disk image file in /etc/lilo.conf, using the
  INITRD configuration variable.

noinitrd

  initrd data is preserved but it is not converted to a RAM disk and
  the "normal" root file system is mounted. initrd data can be read
  from /dev/initrd. Note that the data in initrd can have any structure
  in this case and doesn't necessarily have to be a file system image.
  This option is used mainly for debugging.

  Note: /dev/initrd is read-only and it can only be used once. As soon
  as the last process has closed it, all data is freed and /dev/initrd
  can't be opened anymore.

root=/dev/ram0

  initrd is mounted as root, and the normal boot procedure is followed,
  with the RAM disk mounted as root.
```

48.3 Compressed cpio images

Recent kernels have support for populating a ramdisk from a compressed cpio archive. On such systems, the creation of a ramdisk image doesn't need to involve special block devices or loopbacks; you merely create a directory on disk with the desired initrd content, cd to that directory, and run (as an example):

```
find . | cpio --quiet -H newc -o | gzip -9 -n > /boot/imagefile.img
```

Examining the contents of an existing image file is just as simple:

```
mkdir /tmp/imagefile
cd /tmp/imagefile
gzip -cd /boot/imagefile.img | cpio -imd --quiet
```

48.4 Installation

First, a directory for the initrd file system has to be created on the "normal" root file system, e.g.:

```
# mkdir /initrd
```

The name is not relevant. More details can be found on the *pivot_root(2)* man page.

If the root file system is created during the boot procedure (i.e. if you're building an install floppy), the root file system creation procedure should create the `/initrd` directory.

If initrd will not be mounted in some cases, its content is still accessible if the following device has been created:

```
# mknod /dev/initrd b 1 250
# chmod 400 /dev/initrd
```

Second, the kernel has to be compiled with RAM disk support and with support for the initial RAM disk enabled. Also, at least all components needed to execute programs from initrd (e.g. executable format and file system) must be compiled into the kernel.

Third, you have to create the RAM disk image. This is done by creating a file system on a block device, copying files to it as needed, and then copying the content of the block device to the initrd file. With recent kernels, at least three types of devices are suitable for that:

- a floppy disk (works everywhere but it's painfully slow)
- a RAM disk (fast, but allocates physical memory)
- a loopback device (the most elegant solution)

We'll describe the loopback device method:

1) make sure loopback block devices are configured into the kernel
2) create an empty file system of the appropriate size, e.g.:

   ```
   # dd if=/dev/zero of=initrd bs=300k count=1
   # mke2fs -F -m0 initrd
   ```

   (if space is critical, you may want to use the Minix FS instead of Ext2)
3) mount the file system, e.g.:

   ```
   # mount -t ext2 -o loop initrd /mnt
   ```
4) create the console device:

   ```
   # mkdir /mnt/dev
   # mknod /mnt/dev/console c 5 1
   ```
5) copy all the files that are needed to properly use the initrd environment. Don't forget the most important file, `/sbin/init`

   Note: `/sbin/init` permissions must include "x" (execute).

6) correct operation the initrd environment can frequently be tested even without rebooting with the command:

```
# chroot /mnt /sbin/init
```

This is of course limited to initrds that do not interfere with the general system state (e.g. by reconfiguring network interfaces, overwriting mounted devices, trying to start already running demons, etc. Note however that it is usually possible to use pivot_root in such a chroot'ed initrd environment.)

7) unmount the file system:

```
# umount /mnt
```

8) the initrd is now in the file "initrd". Optionally, it can now be compressed:

```
# gzip -9 initrd
```

For experimenting with initrd, you may want to take a rescue floppy and only add a symbolic link from `/sbin/init` to `/bin/sh`. Alternatively, you can try the experimental newlib environment[2] to create a small initrd.

Finally, you have to boot the kernel and load initrd. Almost all Linux boot loaders support initrd. Since the boot process is still compatible with an older mechanism, the following boot command line parameters have to be given:

```
root=/dev/ram0 rw
```

(rw is only necessary if writing to the initrd file system.)

With LOADLIN, you simply execute:

```
LOADLIN <kernel> initrd=<disk_image>
```

e.g.:

```
LOADLIN C:\LINUX\BZIMAGE initrd=C:\LINUX\INITRD.GZ root=/dev/ram0 rw
```

With LILO, you add the option `INITRD=<path>` to either the global section or to the section of the respective kernel in `/etc/lilo.conf`, and pass the options using APPEND, e.g.:

```
image = /bzImage
  initrd = /boot/initrd.gz
  append = "root=/dev/ram0 rw"
```

and run `/sbin/lilo`

For other boot loaders, please refer to the respective documentation.

Now you can boot and enjoy using initrd.

[2] newlib package (experimental), with initrd example https://www.sourceware.org/newlib/

48.5 Changing the root device

When finished with its duties, init typically changes the root device and proceeds with starting the Linux system on the "real" root device.

The procedure involves the following steps:

- mounting the new root file system
- turning it into the root file system
- removing all accesses to the old (initrd) root file system
- unmounting the initrd file system and de-allocating the RAM disk

Mounting the new root file system is easy: it just needs to be mounted on a directory under the current root. Example:

```
# mkdir /new-root
# mount -o ro /dev/hda1 /new-root
```

The root change is accomplished with the pivot_root system call, which is also available via the `pivot_root` utility (see *`pivot_root(8)`* man page; `pivot_root` is distributed with util-linux version 2.10h or higher[3]). `pivot_root` moves the current root to a directory under the new root, and puts the new root at its place. The directory for the old root must exist before calling `pivot_root`. Example:

```
# cd /new-root
# mkdir initrd
# pivot_root . initrd
```

Now, the init process may still access the old root via its executable, shared libraries, standard input/output/error, and its current root directory. All these references are dropped by the following command:

```
# exec chroot . what-follows <dev/console >dev/console 2>&1
```

Where what-follows is a program under the new root, e.g. `/sbin/init` If the new root file system will be used with udev and has no valid `/dev` directory, udev must be initialized before invoking chroot in order to provide `/dev/console`.

Note: implementation details of pivot_root may change with time. In order to ensure compatibility, the following points should be observed:

- before calling pivot_root, the current directory of the invoking process should point to the new root directory
- use . as the first argument, and the _relative_ path of the directory for the old root as the second argument
- a chroot program must be available under the old and the new root
- chroot to the new root afterwards
- use relative paths for dev/console in the exec command

[3] util-linux: Miscellaneous utilities for Linux https://www.kernel.org/pub/linux/utils/util-linux/

Now, the initrd can be unmounted and the memory allocated by the RAM disk can be freed:

```
# umount /initrd
# blockdev --flushbufs /dev/ram0
```

It is also possible to use initrd with an NFS-mounted root, see the *pivot_root(8)* man page for details.

48.6 Usage scenarios

The main motivation for implementing initrd was to allow for modular kernel configuration at system installation. The procedure would work as follows:

1) system boots from floppy or other media with a minimal kernel (e.g. support for RAM disks, initrd, a.out, and the Ext2 FS) and loads initrd
2) `/sbin/init` determines what is needed to (1) mount the "real" root FS (i.e. device type, device drivers, file system) and (2) the distribution media (e.g. CD-ROM, network, tape, ...). This can be done by asking the user, by auto-probing, or by using a hybrid approach.
3) `/sbin/init` loads the necessary kernel modules
4) `/sbin/init` creates and populates the root file system (this doesn't have to be a very usable system yet)
5) `/sbin/init` invokes `pivot_root` to change the root file system and execs - via chroot - a program that continues the installation
6) the boot loader is installed
7) the boot loader is configured to load an initrd with the set of modules that was used to bring up the system (e.g. `/initrd` can be modified, then unmounted, and finally, the image is written from `/dev/ram0` or `/dev/rd/0` to a file)
8) now the system is bootable and additional installation tasks can be performed

The key role of initrd here is to re-use the configuration data during normal system operation without requiring the use of a bloated "generic" kernel or re-compiling or re-linking the kernel.

A second scenario is for installations where Linux runs on systems with different hardware configurations in a single administrative domain. In such cases, it is desirable to generate only a small set of kernels (ideally only one) and to keep the system-specific part of configuration information as small as possible. In this case, a common initrd could be generated with all the necessary modules. Then, only `/sbin/init` or a file read by it would have to be different.

A third scenario is more convenient recovery disks, because information like the location of the root FS partition doesn't have to be provided at boot time, but the system loaded from initrd can invoke a user-friendly dialog and it can also perform some sanity checks (or even some form of auto-detection).

Last not least, CD-ROM distributors may use it for better installation from CD, e.g. by using a boot floppy and bootstrapping a bigger RAM disk via initrd from CD; or by booting via a loader like `LOADLIN` or directly from the CD-ROM, and loading the RAM disk from CD without need of floppies.

48.7 Obsolete root change mechanism

The following mechanism was used before the introduction of pivot_root. Current kernels still support it, but you should _not_ rely on its continued availability.

It works by mounting the ”real” root device (i.e. the one set with rdev in the kernel image or with root=... at the boot command line) as the root file system when linuxrc exits. The initrd file system is then unmounted, or, if it is still busy, moved to a directory `/initrd`, if such a directory exists on the new root file system.

In order to use this mechanism, you do not have to specify the boot command options root, init, or rw. (If specified, they will affect the real root file system, not the initrd environment.)

If /proc is mounted, the ”real” root device can be changed from within linuxrc by writing the number of the new root FS device to the special file /proc/sys/kernel/real-root-dev, e.g.:

```
# echo 0x301 >/proc/sys/kernel/real-root-dev
```

Note that the mechanism is incompatible with NFS and similar file systems.

This old, deprecated mechanism is commonly called `change_root`, while the new, supported mechanism is called `pivot_root`.

48.8 Mixed change_root and pivot_root mechanism

In case you did not want to use `root=/dev/ram0` to trigger the pivot_root mechanism, you may create both `/linuxrc` and `/sbin/init` in your initrd image.

`/linuxrc` would contain only the following:

```
#! /bin/sh
mount -n -t proc proc /proc
echo 0x0100 >/proc/sys/kernel/real-root-dev
umount -n /proc
```

Once linuxrc exited, the kernel would mount again your initrd as root, this time executing `/sbin/init`. Again, it would be the duty of this init to build the right environment (maybe using the `root= device` passed on the cmdline) before the final execution of the real `/sbin/init`.

48.9 Resources

CHAPTER

FORTYNINE

I/O STATISTICS FIELDS

Since 2.4.20 (and some versions before, with patches), and 2.5.45, more extensive disk statistics have been introduced to help measure disk activity. Tools such as `sar` and `iostat` typically interpret these and do the work for you, but in case you are interested in creating your own tools, the fields are explained here.

In 2.4 now, the information is found as additional fields in `/proc/partitions`. In 2.6 and upper, the same information is found in two places: one is in the file `/proc/diskstats`, and the other is within the sysfs file system, which must be mounted in order to obtain the information. Throughout this document we'll assume that sysfs is mounted on `/sys`, although of course it may be mounted anywhere. Both `/proc/diskstats` and sysfs use the same source for the information and so should not differ.

Here are examples of these different formats:

```
2.4:
   3     0   39082680 hda 446216 784926 9550688 4382310 424847 312726 5922052␣
↪19310380 0 3376340 23705160
   3     1    9221278 hda1 35486 0 35496 38030 0 0 0 0 0 38030 38030

2.6+ sysfs:
   446216 784926 9550688 4382310 424847 312726 5922052 19310380 0 3376340␣
↪23705160
   35486    38030    38030    38030

2.6+ diskstats:
   3    0   hda 446216 784926 9550688 4382310 424847 312726 5922052 19310380 0␣
↪3376340 23705160
   3    1   hda1 35486 38030 38030 38030

4.18+ diskstats:
   3    0   hda 446216 784926 9550688 4382310 424847 312726 5922052 19310380 0␣
↪3376340 23705160 0 0 0 0
```

On 2.4 you might execute `grep 'hda ' /proc/partitions`. On 2.6+, you have a choice of `cat /sys/block/hda/stat` or `grep 'hda ' /proc/diskstats`.

The advantage of one over the other is that the sysfs choice works well if you are watching a known, small set of disks. `/proc/diskstats` may be a better choice if you are watching a large number of disks because you'll avoid the overhead of 50, 100, or 500 or more opens/closes with each snapshot of your disk statistics.

In 2.4, the statistics fields are those after the device name. In the above example, the first field of statistics would be 446216. By contrast, in 2.6+ if you look at `/sys/block/hda/stat`, you'll find just the 15 fields, beginning with 446216. If you look at `/proc/diskstats`, the 15 fields will be preceded by the major and minor device numbers, and device name. Each of these formats provides 15 fields of statistics, each meaning exactly the same things. All fields except field 9 are cumulative since boot. Field 9 should go to zero as I/Os complete; all others only increase (unless they overflow and wrap). Wrapping might eventually occur on a very busy or long-lived system; so applications should be prepared to deal with it. Regarding wrapping, the types of the fields are either unsigned int (32 bit) or unsigned long (32-bit or 64-bit, depending on your machine) as noted per-field below. Unless your observations are very spread in time, these fields should not wrap twice before you notice it.

Each set of stats only applies to the indicated device; if you want system-wide stats you'll have to find all the devices and sum them all up.

Field 1 -- # of reads completed (unsigned long)
This is the total number of reads completed successfully.

Field 2 -- # of reads merged, field 6 -- # of writes merged (unsigned long)
Reads and writes which are adjacent to each other may be merged for efficiency. Thus two 4K reads may become one 8K read before it is ultimately handed to the disk, and so it will be counted (and queued) as only one I/O. This field lets you know how often this was done.

Field 3 -- # of sectors read (unsigned long)
This is the total number of sectors read successfully.

Field 4 -- # of milliseconds spent reading (unsigned int)
This is the total number of milliseconds spent by all reads (as measured from blk_mq_alloc_request() to __blk_mq_end_request()).

Field 5 -- # of writes completed (unsigned long)
This is the total number of writes completed successfully.

Field 6 -- # of writes merged (unsigned long)
See the description of field 2.

Field 7 -- # of sectors written (unsigned long)
This is the total number of sectors written successfully.

Field 8 -- # of milliseconds spent writing (unsigned int)
This is the total number of milliseconds spent by all writes (as measured from blk_mq_alloc_request() to __blk_mq_end_request()).

Field 9 -- # of I/Os currently in progress (unsigned int)
The only field that should go to zero. Incremented as requests are given to appropriate struct request_queue and decremented as they finish.

Field 10 -- # of milliseconds spent doing I/Os (unsigned int)
This field increases so long as field 9 is nonzero.

Since 5.0 this field counts jiffies when at least one request was started or completed. If request runs more than 2 jiffies then some I/O time might be not accounted in case of concurrent requests.

Field 11 -- weighted # of milliseconds spent doing I/Os (unsigned int)
This field is incremented at each I/O start, I/O completion, I/O merge, or read of these stats by the number of I/Os in progress (field 9) times the number of milliseconds spent

doing I/O since the last update of this field. This can provide an easy measure of both I/O completion time and the backlog that may be accumulating.

Field 12 -- # of discards completed (unsigned long)
This is the total number of discards completed successfully.

Field 13 -- # of discards merged (unsigned long)
See the description of field 2

Field 14 -- # of sectors discarded (unsigned long)
This is the total number of sectors discarded successfully.

Field 15 -- # of milliseconds spent discarding (unsigned int)
This is the total number of milliseconds spent by all discards (as measured from blk_mq_alloc_request() to __blk_mq_end_request()).

Field 16 -- # of flush requests completed
This is the total number of flush requests completed successfully.

Block layer combines flush requests and executes at most one at a time. This counts flush requests executed by disk. Not tracked for partitions.

Field 17 -- # of milliseconds spent flushing
This is the total number of milliseconds spent by all flush requests.

To avoid introducing performance bottlenecks, no locks are held while modifying these counters. This implies that minor inaccuracies may be introduced when changes collide, so (for instance) adding up all the read I/Os issued per partition should equal those made to the disks ... but due to the lack of locking it may only be very close.

In 2.6+, there are counters for each CPU, which make the lack of locking almost a non-issue. When the statistics are read, the per-CPU counters are summed (possibly overflowing the unsigned long variable they are summed to) and the result given to the user. There is no convenient user interface for accessing the per-CPU counters themselves.

Since 4.19 request times are measured with nanoseconds precision and truncated to milliseconds before showing in this interface.

49.1 Disks vs Partitions

There were significant changes between 2.4 and 2.6+ in the I/O subsystem. As a result, some statistic information disappeared. The translation from a disk address relative to a partition to the disk address relative to the host disk happens much earlier. All merges and timings now happen at the disk level rather than at both the disk and partition level as in 2.4. Consequently, you'll see a different statistics output on 2.6+ for partitions from that for disks. There are only *four* fields available for partitions on 2.6+ machines. This is reflected in the examples above.

Field 1 -- # of reads issued
This is the total number of reads issued to this partition.

Field 2 -- # of sectors read
This is the total number of sectors requested to be read from this partition.

Field 3 -- # of writes issued
This is the total number of writes issued to this partition.

Field 4 -- # of sectors written

This is the total number of sectors requested to be written to this partition.

Note that since the address is translated to a disk-relative one, and no record of the partition-relative address is kept, the subsequent success or failure of the read cannot be attributed to the partition. In other words, the number of reads for partitions is counted slightly before time of queuing for partitions, and at completion for whole disks. This is a subtle distinction that is probably uninteresting for most cases.

More significant is the error induced by counting the numbers of reads/writes before merges for partitions and after for disks. Since a typical workload usually contains a lot of successive and adjacent requests, the number of reads/writes issued can be several times higher than the number of reads/writes completed.

In 2.6.25, the full statistic set is again available for partitions and disk and partition statistics are consistent again. Since we still don't keep record of the partition-relative address, an operation is attributed to the partition which contains the first sector of the request after the eventual merges. As requests can be merged across partition, this could lead to some (probably insignificant) inaccuracy.

49.2 Additional notes

In 2.6+, sysfs is not mounted by default. If your distribution of Linux hasn't added it already, here's the line you'll want to add to your `/etc/fstab`:

```
none /sys sysfs defaults 0 0
```

In 2.6+, all disk statistics were removed from `/proc/stat`. In 2.4, they appear in both `/proc/partitions` and `/proc/stat`, although the ones in `/proc/stat` take a very different format from those in `/proc/partitions` (see proc(5), if your system has it.)

-- ricklind@us.ibm.com

CHAPTER

FIFTY

JAVA(TM) BINARY KERNEL SUPPORT FOR LINUX V1.03

Linux beats them ALL! While all other OS's are TALKING about direct support of Java Binaries in the OS, Linux is doing it!

You can execute Java applications and Java Applets just like any other program after you have done the following:

1) You MUST FIRST install the Java Developers Kit for Linux. The Java on Linux HOWTO gives the details on getting and installing this. This HOWTO can be found at:

   ftp://sunsite.unc.edu/pub/Linux/docs/HOWTO/Java-HOWTO

   You should also set up a reasonable CLASSPATH environment variable to use Java applications that make use of any nonstandard classes (not included in the same directory as the application itself).

2) You have to compile BINFMT_MISC either as a module or into the kernel (`CONFIG_BINFMT_MISC`) and set it up properly. If you choose to compile it as a module, you will have to insert it manually with modprobe/insmod, as kmod cannot easily be supported with binfmt_misc. Read the file 'binfmt_misc.txt' in this directory to know more about the configuration process.

3) Add the following configuration items to binfmt_misc (you should really have read `binfmt_misc.txt` now): support for Java applications:

```
':Java:M::\xca\xfe\xba\xbe::/usr/local/bin/javawrapper:'
```

   support for executable Jar files:

```
':ExecutableJAR:E::jar::/usr/local/bin/jarwrapper:'
```

   support for Java Applets:

```
':Applet:E::html::/usr/bin/appletviewer:'
```

   or the following, if you want to be more selective:

```
':Applet:M::<!--applet::/usr/bin/appletviewer:'
```

   Of course you have to fix the path names. The path/file names given in this document match the Debian 2.1 system. (i.e. jdk installed in `/usr`, custom wrappers from this document in `/usr/local`)

   Note, that for the more selective applet support you have to modify existing html-files to contain `<!--applet-->` in the first line (< has to be the first character!) to let this work!

For the compiled Java programs you need a wrapper script like the following (this is because Java is broken in case of the filename handling), again fix the path names, both in the script and in the above given configuration string.

You, too, need the little program after the script. Compile like:

```
gcc -O2 -o javaclassname javaclassname.c
```

and stick it to `/usr/local/bin`.

Both the javawrapper shellscript and the javaclassname program were supplied by Colin J. Watson <cjw44@cam.ac.uk>.

Javawrapper shell script:

```
#!/bin/bash
# /usr/local/bin/javawrapper - the wrapper for binfmt_misc/java

if [ -z "$1" ]; then
      exec 1>&2
      echo Usage: $0 class-file
      exit 1
fi

CLASS=$1
FQCLASS=`/usr/local/bin/javaclassname $1`
FQCLASSN=`echo $FQCLASS | sed -e 's/^.*\.\([^.]*\)$/\1/'`
FQCLASSP=`echo $FQCLASS | sed -e 's-\.-/-g' -e 's-^[^/]*$--' -e 's-/[^/]*$--'`

# for example:
# CLASS=Test.class
# FQCLASS=foo.bar.Test
# FQCLASSN=Test
# FQCLASSP=foo/bar

unset CLASSBASE

declare -i LINKLEVEL=0

while :; do
      if [ "`basename $CLASS .class`" == "$FQCLASSN" ]; then
              # See if this directory works straight off
              cd -L `dirname $CLASS`
              CLASSDIR=$PWD
              cd $OLDPWD
              if echo $CLASSDIR | grep -q "$FQCLASSP$"; then
                      CLASSBASE=`echo $CLASSDIR | sed -e "s.$FQCLASSP$.."`
                      break;
              fi
              # Try dereferencing the directory name
              cd -P `dirname $CLASS`
              CLASSDIR=$PWD
              cd $OLDPWD
```

```
                if echo $CLASSDIR | grep -q "$FQCLASSP$"; then
                        CLASSBASE=`echo $CLASSDIR | sed -e "s.$FQCLASSP$.."`
                        break;
                fi
                # If no other possible filename exists
                if [ ! -L $CLASS ]; then
                        exec 1>&2
                        echo $0:
                        echo "  $CLASS should be in a" \
                             "directory tree called $FQCLASSP"
                        exit 1
                fi
        fi
        if [ ! -L $CLASS ]; then break; fi
        # Go down one more level of symbolic links
        let LINKLEVEL+=1
        if [ $LINKLEVEL -gt 5 ]; then
                exec 1>&2
                echo $0:
                echo "  Too many symbolic links encountered"
                exit 1
        fi
        CLASS=`ls --color=no -l $CLASS | sed -e 's/^.* \([^ ]*\)$/\1/'`
done

if [ -z "$CLASSBASE" ]; then
        if [ -z "$FQCLASSP" ]; then
                GOODNAME=$FQCLASSN.class
        else
                GOODNAME=$FQCLASSP/$FQCLASSN.class
        fi
        exec 1>&2
        echo $0:
        echo "  $FQCLASS should be in a file called $GOODNAME"
        exit 1
fi

if ! echo $CLASSPATH | grep -q "^\(.*:\)*$CLASSBASE\(:.*\)*"; then
        # class is not in CLASSPATH, so prepend dir of class to CLASSPATH
        if [ -z "${CLASSPATH}" ] ; then
                export CLASSPATH=$CLASSBASE
        else
                export CLASSPATH=$CLASSBASE:$CLASSPATH
        fi
fi

shift
/usr/bin/java $FQCLASS "$@"
```

javaclassname.c:

```
/* javaclassname.c
 *
 * Extracts the class name from a Java class file; intended for use in a Java
 * wrapper of the type supported by the binfmt_misc option in the Linux kernel.
 *
 * Copyright (C) 1999 Colin J. Watson <cjw44@cam.ac.uk>.
 *
 * This program is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation; either version 2 of the License, or
 * (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program; if not, write to the Free Software
 * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
 */

#include <stdlib.h>
#include <stdio.h>
#include <stdarg.h>
#include <sys/types.h>

/* From Sun's Java VM Specification, as tag entries in the constant pool. */

#define CP_UTF8 1
#define CP_INTEGER 3
#define CP_FLOAT 4
#define CP_LONG 5
#define CP_DOUBLE 6
#define CP_CLASS 7
#define CP_STRING 8
#define CP_FIELDREF 9
#define CP_METHODREF 10
#define CP_INTERFACEMETHODREF 11
#define CP_NAMEANDTYPE 12
#define CP_METHODHANDLE 15
#define CP_METHODTYPE 16
#define CP_INVOKEDYNAMIC 18

/* Define some commonly used error messages */

#define seek_error() error("%s: Cannot seek\n", program)
#define corrupt_error() error("%s: Class file corrupt\n", program)
#define eof_error() error("%s: Unexpected end of file\n", program)
#define utf8_error() error("%s: Only ASCII 1-255 supported\n", program);
```

```
char *program;

long *pool;

u_int8_t read_8(FILE *classfile);
u_int16_t read_16(FILE *classfile);
void skip_constant(FILE *classfile, u_int16_t *cur);
void error(const char *format, ...);
int main(int argc, char **argv);

/* Reads in an unsigned 8-bit integer. */
u_int8_t read_8(FILE *classfile)
{
      int b = fgetc(classfile);
      if(b == EOF)
              eof_error();
      return (u_int8_t)b;
}

/* Reads in an unsigned 16-bit integer. */
u_int16_t read_16(FILE *classfile)
{
      int b1, b2;
      b1 = fgetc(classfile);
      if(b1 == EOF)
              eof_error();
      b2 = fgetc(classfile);
      if(b2 == EOF)
              eof_error();
      return (u_int16_t)((b1 << 8) | b2);
}

/* Reads in a value from the constant pool. */
void skip_constant(FILE *classfile, u_int16_t *cur)
{
      u_int16_t len;
      int seekerr = 1;
      pool[*cur] = ftell(classfile);
      switch(read_8(classfile))
      {
      case CP_UTF8:
              len = read_16(classfile);
              seekerr = fseek(classfile, len, SEEK_CUR);
              break;
      case CP_CLASS:
      case CP_STRING:
      case CP_METHODTYPE:
              seekerr = fseek(classfile, 2, SEEK_CUR);
              break;
```

```
        case CP_METHODHANDLE:
                seekerr = fseek(classfile, 3, SEEK_CUR);
                break;
        case CP_INTEGER:
        case CP_FLOAT:
        case CP_FIELDREF:
        case CP_METHODREF:
        case CP_INTERFACEMETHODREF:
        case CP_NAMEANDTYPE:
        case CP_INVOKEDYNAMIC:
                seekerr = fseek(classfile, 4, SEEK_CUR);
                break;
        case CP_LONG:
        case CP_DOUBLE:
                seekerr = fseek(classfile, 8, SEEK_CUR);
                ++(*cur);
                break;
        default:
                corrupt_error();
        }
        if(seekerr)
                seek_error();
}

void error(const char *format, ...)
{
        va_list ap;
        va_start(ap, format);
        vfprintf(stderr, format, ap);
        va_end(ap);
        exit(1);
}

int main(int argc, char **argv)
{
        FILE *classfile;
        u_int16_t cp_count, i, this_class, classinfo_ptr;
        u_int8_t length;

        program = argv[0];

        if(!argv[1])
                error("%s: Missing input file\n", program);
        classfile = fopen(argv[1], "rb");
        if(!classfile)
                error("%s: Error opening %s\n", program, argv[1]);

        if(fseek(classfile, 8, SEEK_SET))  /* skip magic and version numbers */
                seek_error();
        cp_count = read_16(classfile);
```

```
    pool = calloc(cp_count, sizeof(long));
    if(!pool)
            error("%s: Out of memory for constant pool\n", program);

    for(i = 1; i < cp_count; ++i)
            skip_constant(classfile, &i);
    if(fseek(classfile, 2, SEEK_CUR))       /* skip access flags */
            seek_error();

    this_class = read_16(classfile);
    if(this_class < 1 || this_class >= cp_count)
            corrupt_error();
    if(!pool[this_class] || pool[this_class] == -1)
            corrupt_error();
    if(fseek(classfile, pool[this_class] + 1, SEEK_SET))
            seek_error();

    classinfo_ptr = read_16(classfile);
    if(classinfo_ptr < 1 || classinfo_ptr >= cp_count)
            corrupt_error();
    if(!pool[classinfo_ptr] || pool[classinfo_ptr] == -1)
            corrupt_error();
    if(fseek(classfile, pool[classinfo_ptr] + 1, SEEK_SET))
            seek_error();

    length = read_16(classfile);
    for(i = 0; i < length; ++i)
    {
            u_int8_t x = read_8(classfile);
            if((x & 0x80) || !x)
            {
                    if((x & 0xE0) == 0xC0)
                    {
                            u_int8_t y = read_8(classfile);
                            if((y & 0xC0) == 0x80)
                            {
                                    int c = ((x & 0x1f) << 6) + (y & 0x3f);
                                    if(c) putchar(c);
                                    else utf8_error();
                            }
                            else utf8_error();
                    }
                    else utf8_error();
            }
            else if(x == '/') putchar('.');
            else putchar(x);
    }
    putchar('\n');
    free(pool);
    fclose(classfile);
```

```
      return 0;
}
```

jarwrapper:

```
#!/bin/bash
# /usr/local/java/bin/jarwrapper - the wrapper for binfmt_misc/jar

java -jar $1
```

Now simply `chmod +x` the `.class`, `.jar` and/or `.html` files you want to execute.

To add a Java program to your path best put a symbolic link to the main .class file into /usr/bin (or another place you like) omitting the .class extension. The directory containing the original .class file will be added to your CLASSPATH during execution.

To test your new setup, enter in the following simple Java app, and name it "HelloWorld.java":

```
class HelloWorld {
        public static void main(String args[]) {
                System.out.println("Hello World!");
        }
}
```

Now compile the application with:

```
javac HelloWorld.java
```

Set the executable permissions of the binary file, with:

```
chmod 755 HelloWorld.class
```

And then execute it:

```
./HelloWorld.class
```

To execute Java Jar files, simple chmod the `*.jar` files to include the execution bit, then just do:

```
./Application.jar
```

To execute Java Applets, simple chmod the `*.html` files to include the execution bit, then just do:

```
./Applet.html
```

originally by Brian A. Lantz, brian@lantz.com heavily edited for binfmt_misc by Richard Günther new scripts by Colin J. Watson <cjw44@cam.ac.uk> added executable Jar file support by Kurt Huwig <kurt@iku-netz.de>

CHAPTER

FIFTYONE

IBM'S JOURNALED FILE SYSTEM (JFS) FOR LINUX

JFS Homepage: http://jfs.sourceforge.net/

The following mount options are supported:

(*) == default

iocharset=name
Character set to use for converting from Unicode to ASCII. The default is to do no conversion. Use iocharset=utf8 for UTF-8 translations. This requires CONFIG_NLS_UTF8 to be set in the kernel .config file. iocharset=none specifies the default behavior explicitly.

resize=value
Resize the volume to <value> blocks. JFS only supports growing a volume, not shrinking it. This option is only valid during a remount, when the volume is mounted read-write. The resize keyword with no value will grow the volume to the full size of the partition.

nointegrity
Do not write to the journal. The primary use of this option is to allow for higher performance when restoring a volume from backup media. The integrity of the volume is not guaranteed if the system abnormally abends.

integrity(*)
Commit metadata changes to the journal. Use this option to remount a volume where the nointegrity option was previously specified in order to restore normal behavior.

errors=continue
Keep going on a filesystem error.

errors=remount-ro(*)
Remount the filesystem read-only on an error.

errors=panic
Panic and halt the machine if an error occurs.

uid=value
Override on-disk uid with specified value

gid=value
Override on-disk gid with specified value

umask=value
Override on-disk umask with specified octal value. For directories, the execute bit will be set if the corresponding read bit is set.

discard=minlen, discard/nodiscard(*)
This enables/disables the use of discard/TRIM commands. The discard/TRIM commands

are sent to the underlying block device when blocks are freed. This is useful for SSD devices and sparse/thinly-provisioned LUNs. The FITRIM ioctl command is also available together with the nodiscard option. The value of minlen specifies the minimum blockcount, when a TRIM command to the block device is considered useful. When no value is given to the discard option, it defaults to 64 blocks, which means 256KiB in JFS. The minlen value of discard overrides the minlen value given on an FITRIM ioctl().

The JFS mailing list can be subscribed to by using the link labeled ”Mail list Subscribe” at our web page http://jfs.sourceforge.net/

CHAPTER

FIFTYTWO

REDUCING OS JITTER DUE TO PER-CPU KTHREADS

This document lists per-CPU kthreads in the Linux kernel and presents options to control their OS jitter. Note that non-per-CPU kthreads are not listed here. To reduce OS jitter from non-per-CPU kthreads, bind them to a ”housekeeping” CPU dedicated to such work.

52.1 References

- Documentation/core-api/irq/irq-affinity.rst: Binding interrupts to sets of CPUs.
- Documentation/admin-guide/cgroup-v1: Using cgroups to bind tasks to sets of CPUs.
- man taskset: Using the taskset command to bind tasks to sets of CPUs.
- man sched_setaffinity: Using the sched_setaffinity() system call to bind tasks to sets of CPUs.
- /sys/devices/system/cpu/cpuN/online: Control CPU N's hotplug state, writing ”0” to offline and ”1” to online.
- In order to locate kernel-generated OS jitter on CPU N:

  cd /sys/kernel/tracing echo 1 > max_graph_depth # Increase the ”1” for more detail echo function_graph > current_tracer # run workload cat per_cpu/cpuN/trace

52.2 kthreads

Name:
ehca_comp/%u

Purpose:
Periodically process Infiniband-related work.

To reduce its OS jitter, do any of the following:

1. Don't use eHCA Infiniband hardware, instead choosing hardware that does not require per-CPU kthreads. This will prevent these kthreads from being created in the first place. (This will work for most people, as this hardware, though important, is relatively old and is produced in relatively low unit volumes.)
2. Do all eHCA-Infiniband-related work on other CPUs, including interrupts.
3. Rework the eHCA driver so that its per-CPU kthreads are provisioned only on selected CPUs.

Name:
irq/%d-%s

Purpose:
Handle threaded interrupts.

To reduce its OS jitter, do the following:

1. Use irq affinity to force the irq threads to execute on some other CPU.

Name:
kcmtpd_ctr_%d

Purpose:
Handle Bluetooth work.

To reduce its OS jitter, do one of the following:

1. Don't use Bluetooth, in which case these kthreads won't be created in the first place.
2. Use irq affinity to force Bluetooth-related interrupts to occur on some other CPU and furthermore initiate all Bluetooth activity on some other CPU.

Name:
ksoftirqd/%u

Purpose:
Execute softirq handlers when threaded or when under heavy load.

To reduce its OS jitter, each softirq vector must be handled separately as follows:

52.2.1 TIMER_SOFTIRQ

Do all of the following:

1. To the extent possible, keep the CPU out of the kernel when it is non-idle, for example, by avoiding system calls and by forcing both kernel threads and interrupts to execute elsewhere.
2. Build with CONFIG_HOTPLUG_CPU=y. After boot completes, force the CPU offline, then bring it back online. This forces recurring timers to migrate elsewhere. If you are concerned with multiple CPUs, force them all offline before bringing the first one back online. Once you have onlined the CPUs in question, do not offline any other CPUs, because doing so could force the timer back onto one of the CPUs in question.

52.2.2 NET_TX_SOFTIRQ and NET_RX_SOFTIRQ

Do all of the following:

1. Force networking interrupts onto other CPUs.
2. Initiate any network I/O on other CPUs.
3. Once your application has started, prevent CPU-hotplug operations from being initiated from tasks that might run on the CPU to be de-jittered. (It is OK to force this CPU offline and then bring it back online before you start your application.)

52.2.3 BLOCK_SOFTIRQ

Do all of the following:

1. Force block-device interrupts onto some other CPU.
2. Initiate any block I/O on other CPUs.
3. Once your application has started, prevent CPU-hotplug operations from being initiated from tasks that might run on the CPU to be de-jittered. (It is OK to force this CPU offline and then bring it back online before you start your application.)

52.2.4 IRQ_POLL_SOFTIRQ

Do all of the following:

1. Force block-device interrupts onto some other CPU.
2. Initiate any block I/O and block-I/O polling on other CPUs.
3. Once your application has started, prevent CPU-hotplug operations from being initiated from tasks that might run on the CPU to be de-jittered. (It is OK to force this CPU offline and then bring it back online before you start your application.)

52.2.5 TASKLET_SOFTIRQ

Do one or more of the following:

1. Avoid use of drivers that use tasklets. (Such drivers will contain calls to things like tasklet_schedule().)
2. Convert all drivers that you must use from tasklets to workqueues.
3. Force interrupts for drivers using tasklets onto other CPUs, and also do I/O involving these drivers on other CPUs.

52.2.6 SCHED_SOFTIRQ

Do all of the following:

1. Avoid sending scheduler IPIs to the CPU to be de-jittered, for example, ensure that at most one runnable kthread is present on that CPU. If a thread that expects to run on the de-jittered CPU awakens, the scheduler will send an IPI that can result in a subsequent SCHED_SOFTIRQ.
2. CONFIG_NO_HZ_FULL=y and ensure that the CPU to be de-jittered is marked as an adaptive-ticks CPU using the ”nohz_full=” boot parameter. This reduces the number of scheduler-clock interrupts that the de-jittered CPU receives, minimizing its chances of being selected to do the load balancing work that runs in SCHED_SOFTIRQ context.
3. To the extent possible, keep the CPU out of the kernel when it is non-idle, for example, by avoiding system calls and by forcing both kernel threads and interrupts to execute elsewhere. This further reduces the number of scheduler-clock interrupts received by the de-jittered CPU.

52.2.7 HRTIMER_SOFTIRQ

Do all of the following:

1. To the extent possible, keep the CPU out of the kernel when it is non-idle. For example, avoid system calls and force both kernel threads and interrupts to execute elsewhere.
2. Build with CONFIG_HOTPLUG_CPU=y. Once boot completes, force the CPU offline, then bring it back online. This forces recurring timers to migrate elsewhere. If you are concerned with multiple CPUs, force them all offline before bringing the first one back online. Once you have onlined the CPUs in question, do not offline any other CPUs, because doing so could force the timer back onto one of the CPUs in question.

52.2.8 RCU_SOFTIRQ

Do at least one of the following:

1. Offload callbacks and keep the CPU in either dyntick-idle or adaptive-ticks state by doing all of the following:
   a. CONFIG_NO_HZ_FULL=y and ensure that the CPU to be de-jittered is marked as an adaptive-ticks CPU using the "nohz_full=" boot parameter. Bind the rcuo kthreads to housekeeping CPUs, which can tolerate OS jitter.
   b. To the extent possible, keep the CPU out of the kernel when it is non-idle, for example, by avoiding system calls and by forcing both kernel threads and interrupts to execute elsewhere.
2. Enable RCU to do its processing remotely via dyntick-idle by doing all of the following:
   a. Build with CONFIG_NO_HZ=y.
   b. Ensure that the CPU goes idle frequently, allowing other CPUs to detect that it has passed through an RCU quiescent state. If the kernel is built with CONFIG_NO_HZ_FULL=y, userspace execution also allows other CPUs to detect that the CPU in question has passed through a quiescent state.
   c. To the extent possible, keep the CPU out of the kernel when it is non-idle, for example, by avoiding system calls and by forcing both kernel threads and interrupts to execute elsewhere.

Name:
kworker/%u:%d%s (cpu, id, priority)

Purpose:
Execute workqueue requests

To reduce its OS jitter, do any of the following:

1. Run your workload at a real-time priority, which will allow preempting the kworker daemons.
2. A given workqueue can be made visible in the sysfs filesystem by passing the WQ_SYSFS to that workqueue's alloc_workqueue(). Such a workqueue can be confined to a given subset of the CPUs using the `/sys/devices/virtual/workqueue/*/cpumask` sysfs files. The set of WQ_SYSFS workqueues can be displayed using "ls /sys/devices/virtual/workqueue". That said, the workqueues maintainer would like to caution people against indiscriminately sprinkling WQ_SYSFS across all the workqueues. The reason for caution is that it is easy

to add WQ_SYSFS, but because sysfs is part of the formal user/kernel API, it can be nearly impossible to remove it, even if its addition was a mistake.

3. Do any of the following needed to avoid jitter that your application cannot tolerate:

   a. Avoid using oprofile, thus avoiding OS jitter from wq_sync_buffer().

   b. Limit your CPU frequency so that a CPU-frequency governor is not required, possibly enlisting the aid of special heatsinks or other cooling technologies. If done correctly, and if you CPU architecture permits, you should be able to build your kernel with CONFIG_CPU_FREQ=n to avoid the CPU-frequency governor periodically running on each CPU, including cs_dbs_timer() and od_dbs_timer().

      WARNING: Please check your CPU specifications to make sure that this is safe on your particular system.

   c. As of v3.18, Christoph Lameter's on-demand vmstat workers commit prevents OS jitter due to vmstat_update() on CONFIG_SMP=y systems. Before v3.18, is not possible to entirely get rid of the OS jitter, but you can decrease its frequency by writing a large value to /proc/sys/vm/stat_interval. The default value is HZ, for an interval of one second. Of course, larger values will make your virtual-memory statistics update more slowly. Of course, you can also run your workload at a real-time priority, thus preempting vmstat_update(), but if your workload is CPU-bound, this is a bad idea. However, there is an RFC patch from Christoph Lameter (based on an earlier one from Gilad Ben-Yossef) that reduces or even eliminates vmstat overhead for some workloads at https://lore.kernel.org/r/00000140e9dfd6bd-40db3d4f-c1be-434f-8132-7820f81bb586-000000@email.amazonses.com.

   d. If running on high-end powerpc servers, build with CONFIG_PPC_RTAS_DAEMON=n. This prevents the RTAS daemon from running on each CPU every second or so. (This will require editing Kconfig files and will defeat this platform's RAS functionality.) This avoids jitter due to the rtas_event_scan() function. WARNING: Please check your CPU specifications to make sure that this is safe on your particular system.

   e. If running on Cell Processor, build your kernel with CBE_CPUFREQ_SPU_GOVERNOR=n to avoid OS jitter from spu_gov_work(). WARNING: Please check your CPU specifications to make sure that this is safe on your particular system.

   f. If running on PowerMAC, build your kernel with CONFIG_PMAC_RACKMETER=n to disable the CPU-meter, avoiding OS jitter from rackmeter_do_timer().

Name:
rcuc/%u

Purpose:
Execute RCU callbacks in CONFIG_RCU_BOOST=y kernels.

To reduce its OS jitter, do at least one of the following:

1. Build the kernel with CONFIG_PREEMPT=n. This prevents these kthreads from being created in the first place, and also obviates the need for RCU priority boosting. This approach is feasible for workloads that do not require high degrees of responsiveness.

2. Build the kernel with CONFIG_RCU_BOOST=n. This prevents these kthreads from being created in the first place. This approach is feasible only if your workload never requires

RCU priority boosting, for example, if you ensure frequent idle time on all CPUs that might execute within the kernel.

3. Build with CONFIG_RCU_NOCB_CPU=y and boot with the rcu_nocbs= boot parameter offloading RCU callbacks from all CPUs susceptible to OS jitter. This approach prevents the rcuc/%u kthreads from having any work to do, so that they are never awakened.
4. Ensure that the CPU never enters the kernel, and, in particular, avoid initiating any CPU hotplug operations on this CPU. This is another way of preventing any callbacks from being queued on the CPU, again preventing the rcuc/%u kthreads from having any work to do.

Name:
rcuop/%d and rcuos/%d

Purpose:
Offload RCU callbacks from the corresponding CPU.

To reduce its OS jitter, do at least one of the following:

1. Use affinity, cgroups, or other mechanism to force these kthreads to execute on some other CPU.
2. Build with CONFIG_RCU_NOCB_CPU=n, which will prevent these kthreads from being created in the first place. However, please note that this will not eliminate OS jitter, but will instead shift it to RCU_SOFTIRQ.

CHAPTER

FIFTYTHREE

LAPTOP DRIVERS

53.1 Asus Laptop Extras

Version 0.1

August 6, 2009

Corentin Chary <corentincj@iksaif.net> http://acpi4asus.sf.net/

This driver provides support for extra features of ACPI-compatible ASUS laptops. It may also support some MEDION, JVC or VICTOR laptops (such as MEDION 9675 or VICTOR XP7210 for example). It makes all the extra buttons generate input events (like keyboards).

On some models adds support for changing the display brightness and output, switching the LCD backlight on and off, and most importantly, allows you to blink those fancy LEDs intended for reporting mail and wireless status.

This driver supersedes the old asus_acpi driver.

53.1.1 Requirements

Kernel 2.6.X sources, configured for your computer, with ACPI support. You also need CONFIG_INPUT and CONFIG_ACPI.

53.1.2 Status

The features currently supported are the following (see below for detailed description):

- Fn key combinations
- Bluetooth enable and disable
- Wlan enable and disable
- GPS enable and disable
- Video output switching
- Ambient Light Sensor on and off
- LED control
- LED Display control

- LCD brightness control
- LCD on and off

A compatibility table by model and feature is maintained on the web site, http://acpi4asus.sf.net/.

53.1.3 Usage

Try "modprobe asus-laptop". Check your dmesg (simply type dmesg). You should see some lines like this :

Asus Laptop Extras version 0.42

- L2D model detected.

If it is not the output you have on your laptop, send it (and the laptop's DSDT) to me.

That's all, now, all the events generated by the hotkeys of your laptop should be reported via netlink events. You can check with "acpi_genl monitor" (part of the acpica project).

Hotkeys are also reported as input keys (like keyboards) you can check which key are supported using "xev" under X11.

You can get information on the version of your DSDT table by reading the /sys/devices/platform/asus-laptop/infos entry. If you have a question or a bug report to do, please include the output of this entry.

53.1.4 LEDs

You can modify LEDs be echoing values to */sys/class/leds/asus/*/brightness*:

```
echo 1 >  /sys/class/leds/asus::mail/brightness
```

will switch the mail LED on.

You can also know if they are on/off by reading their content and use kernel triggers like disk-activity or heartbeat.

53.1.5 Backlight

You can control lcd backlight power and brightness with /sys/class/backlight/asus-laptop/. Brightness Values are between 0 and 15.

53.1.6 Wireless devices

You can turn the internal Bluetooth adapter on/off with the bluetooth entry (only on models with Bluetooth). This usually controls the associated LED. Same for Wlan adapter.

53.1.7 Display switching

Note: the display switching code is currently considered EXPERIMENTAL.

Switching works for the following models:

- L3800C
- A2500H
- L5800C
- M5200N
- W1000N (albeit with some glitches)
- M6700R
- A6JC
- F3J

Switching doesn't work for the following:

- M3700N
- L2X00D (locks the laptop under certain conditions)

To switch the displays, echo values from 0 to 15 to /sys/devices/platform/asus-laptop/display. The significance of those values is as follows:

Bin	Val	DVI	TV	CRT	LCD
0000	0				
0001	1				X
0010	2			X	
0011	3			X	X
0100	4		X		
0101	5		X		X
0110	6		X	X	
0111	7		X	X	X
1000	8	X			
1001	9	X			X
1010	10	X		X	
1011	11	X		X	X
1100	12	X	X		
1101	13	X	X		X
1110	14	X	X	X	
1111	15	X	X	X	X

In most cases, the appropriate displays must be plugged in for the above combinations to work. TV-Out may need to be initialized at boot time.

Debugging:

1) Check whether the Fn+F8 key:

   a) does not lock the laptop (try a boot with noapic / nolapic if it does)

   b) generates events (0x6n, where n is the value corresponding to the configuration above)

   c) actually works

   Record the disp value at every configuration.

2) Echo values from 0 to 15 to /sys/devices/platform/asus-laptop/display. Record its value, note any change. If nothing changes, try a broader range, up to 65535.

3) Send ANY output (both positive and negative reports are needed, unless your machine is already listed above) to the acpi4asus-user mailing list.

Note: on some machines (e.g. L3C), after the module has been loaded, only 0x6n events are generated and no actual switching occurs. In such a case, a line like:

```
echo $((10#$arg-60)) > /sys/devices/platform/asus-laptop/display
```

will usually do the trick ($arg is the 0000006n-like event passed to acpid).

Note: there is currently no reliable way to read display status on xxN (Centrino) models.

53.1.8 LED display

Some models like the W1N have a LED display that can be used to display several items of information.

LED display works for the following models:

- W1000N
- W1J

To control the LED display, use the following:

```
echo 0x0T000DDD > /sys/devices/platform/asus-laptop/
```

where T control the 3 letters display, and DDD the 3 digits display, according to the tables below:

```
DDD (digits)
000 to 999 = display digits
AAA        = ---
BBB to FFF = turn-off

T  (type)
0 = off
1 = dvd
```

```
2 = vcd
3 = mp3
4 = cd
5 = tv
6 = cpu
7 = vol
```

For example ”echo 0x01000001 >/sys/devices/platform/asus-laptop/ledd” would display ”DVD001”.

53.1.9 Driver options

Options can be passed to the asus-laptop driver using the standard module argument syntax (<param>=<value> when passing the option to the module or asus-laptop.<param>=<value> on the kernel boot line when asus-laptop is statically linked into the kernel).

wapf: WAPF defines the behavior of the Fn+Fx wlan key
The significance of values is yet to be found, but most of the time:

- 0x0 should do nothing
- 0x1 should allow to control the device with Fn+Fx key.
- 0x4 should send an ACPI event (0x88) while pressing the Fn+Fx key
- 0x5 like 0x1 or 0x4

The default value is 0x1.

53.1.10 Unsupported models

These models will never be supported by this module, as they use a completely different mechanism to handle LEDs and extra stuff (meaning we have no clue how it works):

- ASUS A1300 (A1B), A1370D
- ASUS L7300G
- ASUS L8400

53.1.11 Patches, Errors, Questions

I appreciate any success or failure reports, especially if they add to or correct the compatibility table. Please include the following information in your report:

- Asus model name
- a copy of your ACPI tables, using the ”acpidump” utility
- a copy of /sys/devices/platform/asus-laptop/infos
- which driver features work and which don't
- the observed behavior of non-working features

Any other comments or patches are also more than welcome.

acpi4asus-user@lists.sourceforge.net

http://sourceforge.net/projects/acpi4asus

53.2 Hard disk shock protection

Author: Elias Oltmanns <eo@nebensachen.de>

Last modified: 2008-10-03

53.2.1 1. Intro

ATA/ATAPI-7 specifies the IDLE IMMEDIATE command with unload feature. Issuing this command should cause the drive to switch to idle mode and unload disk heads. This feature is being used in modern laptops in conjunction with accelerometers and appropriate software to implement a shock protection facility. The idea is to stop all I/O operations on the internal hard drive and park its heads on the ramp when critical situations are anticipated. The desire to have such a feature available on GNU/Linux systems has been the original motivation to implement a generic disk head parking interface in the Linux kernel. Please note, however, that other components have to be set up on your system in order to get disk shock protection working (see section 3. References below for pointers to more information about that).

53.2.2 2. The interface

For each ATA device, the kernel exports the file *block/*/device/unload_heads* in sysfs (here assumed to be mounted under /sys). Access to */sys/block/*/device/unload_heads* is denied with -EOPNOTSUPP if the device does not support the unload feature. Otherwise, writing an integer value to this file will take the heads of the respective drive off the platter and block all I/O operations for the specified number of milliseconds. When the timeout expires and no further disk head park request has been issued in the meantime, normal operation will be resumed. The maximal value accepted for a timeout is 30000 milliseconds. Exceeding this limit will return -EOVERFLOW, but heads will be parked anyway and the timeout will be set to 30 seconds. However, you can always change a timeout to any value between 0 and 30000 by issuing a subsequent head park request before the timeout of the previous one has expired. In particular, the total timeout can exceed 30 seconds and, more importantly, you can cancel a previously set timeout and resume normal operation immediately by specifying a timeout of 0. Values below -2 are rejected with -EINVAL (see below for the special meaning of -1 and -2). If the timeout specified for a recent head park request has not yet expired, reading from */sys/block/*/device/unload_heads* will report the number of milliseconds remaining until normal operation will be resumed; otherwise, reading the unload_heads attribute will return 0.

For example, do the following in order to park the heads of drive /dev/sda and stop all I/O operations for five seconds:

```
# echo 5000 > /sys/block/sda/device/unload_heads
```

A simple:

```
# cat /sys/block/sda/device/unload_heads
```

will show you how many milliseconds are left before normal operation will be resumed.

A word of caution: The fact that the interface operates on a basis of milliseconds may raise expectations that cannot be satisfied in reality. In fact, the ATA specs clearly state that the time for an unload operation to complete is vendor specific. The hint in ATA-7 that this will typically be within 500 milliseconds apparently has been dropped in ATA-8.

There is a technical detail of this implementation that may cause some confusion and should be discussed here. When a head park request has been issued to a device successfully, all I/O operations on the controller port this device is attached to will be deferred. That is to say, any other device that may be connected to the same port will be affected too. The only exception is that a subsequent head unload request to that other device will be executed immediately. Further operations on that port will be deferred until the timeout specified for either device on the port has expired. As far as PATA (old style IDE) configurations are concerned, there can only be two devices attached to any single port. In SATA world we have port multipliers which means that a user-issued head parking request to one device may actually result in stopping I/O to a whole bunch of devices. However, since this feature is supposed to be used on laptops and does not seem to be very useful in any other environment, there will be mostly one device per port. Even if the CD/DVD writer happens to be connected to the same port as the hard drive, it generally *should* recover just fine from the occasional buffer under-run incurred by a head park request to the HD. Actually, when you are using an ide driver rather than its libata counterpart (i.e. your disk is called /dev/hda instead of /dev/sda), then parking the heads of one drive (drive X) will generally not affect the mode of operation of another drive (drive Y) on the same port as described above. It is only when a port reset is required to recover from an exception on drive Y that further I/O operations on that drive (and the reset itself) will be delayed until drive X is no longer in the parked state.

Finally, there are some hard drives that only comply with an earlier version of the ATA standard than ATA-7, but do support the unload feature nonetheless. Unfortunately, there is no safe way Linux can detect these devices, so you won't be able to write to the unload_heads attribute. If you know that your device really does support the unload feature (for instance, because the vendor of your laptop or the hard drive itself told you so), then you can tell the kernel to enable the usage of this feature for that drive by writing the special value -1 to the unload_heads attribute:

```
# echo -1 > /sys/block/sda/device/unload_heads
```

will enable the feature for /dev/sda, and giving -2 instead of -1 will disable it again.

53.2.3 3. References

There are several laptops from different vendors featuring shock protection capabilities. As manufacturers have refused to support open source development of the required software components so far, Linux support for shock protection varies considerably between different hardware implementations. Ideally, this section should contain a list of pointers at different projects aiming at an implementation of shock protection on different systems. Unfortunately, I only know of a single project which, although still considered experimental, is fit for use. Please feel free to add projects that have been the victims of my ignorance.

- https://www.thinkwiki.org/wiki/HDAPS

See this page for information about Linux support of the hard disk active protection system as implemented in IBM/Lenovo Thinkpads.

53.2.4 4. CREDITS

This implementation of disk head parking has been inspired by a patch originally published by Jon Escombe <lists@dresco.co.uk>. My efforts to develop an implementation of this feature that is fit to be merged into mainline have been aided by various kernel developers, in particular by Tejun Heo and Bartlomiej Zolnierkiewicz.

53.3 How to conserve battery power using laptop-mode

Document Author: Bart Samwel (bart@samwel.tk)

Date created: January 2, 2004

Last modified: December 06, 2004

53.3.1 Introduction

Laptop mode is used to minimize the time that the hard disk needs to be spun up, to conserve battery power on laptops. It has been reported to cause significant power savings.

53.3.2 Installation

To use laptop mode, you don't need to set any kernel configuration options or anything. Simply install all the files included in this document, and laptop mode will automatically be started when you're on battery. For your convenience, a tarball containing an installer can be downloaded at:

http://www.samwel.tk/laptop_mode/laptop_mode/

To configure laptop mode, you need to edit the configuration file, which is located in /etc/default/laptop-mode on Debian-based systems, or in /etc/sysconfig/laptop-mode on other systems.

Unfortunately, automatic enabling of laptop mode does not work for laptops that don't have ACPI. On those laptops, you need to start laptop mode manually. To start laptop mode, run ”laptop_mode start”, and to stop it, run ”laptop_mode stop”. (Note: The laptop mode tools package now has experimental support for APM, you might want to try that first.)

53.3.3 Caveats

- The downside of laptop mode is that you have a chance of losing up to 10 minutes of work. If you cannot afford this, don't use it! The supplied ACPI scripts automatically turn off laptop mode when the battery almost runs out, so that you won't lose any data at the end of your battery life.
- Most desktop hard drives have a very limited lifetime measured in spindown cycles, typically about 50.000 times (it's usually listed on the spec sheet). Check your drive's rating, and don't wear down your drive's lifetime if you don't need to.
- If you mount some of your ext3/reiserfs filesystems with the -n option, then the control script will not be able to remount them correctly. You must set DO_REMOUNTS=0 in the control script, otherwise it will remount them with the wrong options -- or it will fail because it cannot write to /etc/mtab.
- If you have your filesystems listed as type ”auto” in fstab, like I did, then the control script will not recognize them as filesystems that need remounting. You must list the filesystems with their true type instead.
- It has been reported that some versions of the mutt mail client use file access times to determine whether a folder contains new mail. If you use mutt and experience this, you must disable the noatime remounting by setting the option DO_REMOUNT_NOATIME to 0 in the configuration file.

53.3.4 The Details

Laptop mode is controlled by the knob /proc/sys/vm/laptop_mode. This knob is present for all kernels that have the laptop mode patch, regardless of any configuration options. When the knob is set, any physical disk I/O (that might have caused the hard disk to spin up) causes Linux to flush all dirty blocks. The result of this is that after a disk has spun down, it will not be spun up anymore to write dirty blocks, because those blocks had already been written immediately after the most recent read operation. The value of the laptop_mode knob determines the time between the occurrence of disk I/O and when the flush is triggered. A sensible value for the knob is 5 seconds. Setting the knob to 0 disables laptop mode.

To increase the effectiveness of the laptop_mode strategy, the laptop_mode control script increases dirty_expire_centisecs and dirty_writeback_centisecs in /proc/sys/vm to about 10 minutes (by default), which means that pages that are dirtied are not forced to be written to disk as often. The control script also changes the dirty background ratio, so that background writeback of dirty pages is not done anymore. Combined with a higher commit value (also 10 minutes) for ext3 or ReiserFS filesystems (also done automatically by the control script), this results in concentration of disk activity in a small time interval which occurs only once every 10 minutes, or whenever the disk is forced to spin up by a cache miss. The disk can then be spun down in the periods of inactivity.

53.3.5 Configuration

The laptop mode configuration file is located in /etc/default/laptop-mode on Debian-based systems, or in /etc/sysconfig/laptop-mode on other systems. It contains the following options:

MAX_AGE:

Maximum time, in seconds, of hard drive spindown time that you are comfortable with. Worst case, it's possible that you could lose this amount of work if your battery fails while you're in laptop mode.

MINIMUM_BATTERY_MINUTES:

Automatically disable laptop mode if the remaining number of minutes of battery power is less than this value. Default is 10 minutes.

AC_HD/BATT_HD:

The idle timeout that should be set on your hard drive when laptop mode is active (BATT_HD) and when it is not active (AC_HD). The defaults are 20 seconds (value 4) for BATT_HD and 2 hours (value 244) for AC_HD. The possible values are those listed in the manual page for ”hdparm” for the ”-S” option.

HD:

The devices for which the spindown timeout should be adjusted by laptop mode. Default is /dev/hda. If you specify multiple devices, separate them by a space.

READAHEAD:

Disk readahead, in 512-byte sectors, while laptop mode is active. A large readahead can prevent disk accesses for things like executable pages (which are loaded on demand while the application executes) and sequentially accessed data (MP3s).

DO_REMOUNTS:

The control script automatically remounts any mounted journaled filesystems with appropriate commit interval options. When this option is set to 0, this feature is disabled.

DO_REMOUNT_NOATIME:

When remounting, should the filesystems be remounted with the noatime option? Normally, this is set to ”1” (enabled), but there may be programs that require access time recording.

DIRTY_RATIO:

The percentage of memory that is allowed to contain ”dirty” or unsaved data before a writeback is forced, while laptop mode is active. Corresponds to the /proc/sys/vm/dirty_ratio sysctl.

DIRTY_BACKGROUND_RATIO:

The percentage of memory that is allowed to contain ”dirty” or unsaved data after a forced writeback is done due to an exceeding of DIRTY_RATIO. Set this nice and low. This corresponds to the /proc/sys/vm/dirty_background_ratio sysctl.

Note that the behaviour of dirty_background_ratio is quite different when laptop mode is active and when it isn't. When laptop mode is inactive, dirty_background_ratio is the threshold percentage at which background writeouts start taking place. When laptop mode is active, however, background writeouts are disabled, and the dirty_background_ratio only determines how much writeback is done when dirty_ratio is reached.

DO_CPU:

Enable CPU frequency scaling when in laptop mode. (Requires CPUFreq to be setup. See *CPU Performance Scaling* for more info. Disabled by default.)

CPU_MAXFREQ:

When on battery, what is the maximum CPU speed that the system should use? Legal values are "slowest" for the slowest speed that your CPU is able to operate at, or a value listed in /sys/devices/system/cpu/cpu0/cpufreq/scaling_available_frequencies.

53.3.6 Tips & Tricks

- Bartek Kania reports getting up to 50 minutes of extra battery life (on top of his regular 3 to 3.5 hours) using a spindown time of 5 seconds (BATT_HD=1).
- You can spin down the disk while playing MP3, by setting disk readahead to 8MB (READAHEAD=16384). Effectively, the disk will read a complete MP3 at once, and will then spin down while the MP3 is playing. (Thanks to Bartek Kania.)
- Drew Scott Daniels observed: "I don't know why, but when I decrease the number of colours that my display uses it consumes less battery power. I've seen this on powerbooks too. I hope that this is a piece of information that might be useful to the Laptop Mode patch or its users."
- In syslog.conf, you can prefix entries with a dash - to omit syncing the file after every logging. When you're using laptop-mode and your disk doesn't spin down, this is a likely culprit.
- Richard Atterer observed that laptop mode does not work well with noflushd (http://noflushd.sourceforge.net/), it seems that noflushd prevents laptop-mode from doing its thing.
- If you're worried about your data, you might want to consider using a USB memory stick or something like that as a "working area". (Be aware though that flash memory can only handle a limited number of writes, and overuse may wear out your memory stick pretty quickly. Do _not_ use journalling filesystems on flash memory sticks.)

53.3.7 Configuration file for control and ACPI battery scripts

This allows the tunables to be changed for the scripts via an external configuration file

It should be installed as /etc/default/laptop-mode on Debian, and as /etc/sysconfig/laptop-mode on Red Hat, SUSE, Mandrake, and other work-alikes.

Config file:

```
# Maximum time, in seconds, of hard drive spindown time that you are
# comfortable with. Worst case, it's possible that you could lose this
# amount of work if your battery fails you while in laptop mode.
#MAX_AGE=600

# Automatically disable laptop mode when the number of minutes of battery
# that you have left goes below this threshold.
MINIMUM_BATTERY_MINUTES=10
```

```
# Read-ahead, in 512-byte sectors. You can spin down the disk while playing␣
↪MP3/OGG
# by setting the disk readahead to 8MB (READAHEAD=16384). Effectively, the disk
# will read a complete MP3 at once, and will then spin down while the MP3/OGG␣
↪is
# playing.
#READAHEAD=4096

# Shall we remount journaled fs. with appropriate commit interval? (1=yes)
#DO_REMOUNTS=1

# And shall we add the "noatime" option to that as well? (1=yes)
#DO_REMOUNT_NOATIME=1

# Dirty synchronous ratio.  At this percentage of dirty pages the process
# which
# calls write() does its own writeback
#DIRTY_RATIO=40

#
# Allowed dirty background ratio, in percent.  Once DIRTY_RATIO has been
# exceeded, the kernel will wake flusher threads which will then reduce the
# amount of dirty memory to dirty_background_ratio.  Set this nice and low,
# so once some writeout has commenced, we do a lot of it.
#
#DIRTY_BACKGROUND_RATIO=5

# kernel default dirty buffer age
#DEF_AGE=30
#DEF_UPDATE=5
#DEF_DIRTY_BACKGROUND_RATIO=10
#DEF_DIRTY_RATIO=40
#DEF_XFS_AGE_BUFFER=15
#DEF_XFS_SYNC_INTERVAL=30
#DEF_XFS_BUFD_INTERVAL=1

# This must be adjusted manually to the value of HZ in the running kernel
# on 2.4, until the XFS people change their 2.4 external interfaces to work in
# centisecs. This can be automated, but it's a work in progress that still
# needs# some fixes. On 2.6 kernels, XFS uses USER_HZ instead of HZ for
# external interfaces, and that is currently always set to 100. So you don't
# need to change this on 2.6.
#XFS_HZ=100

# Should the maximum CPU frequency be adjusted down while on battery?
# Requires CPUFreq to be setup.
# See Documentation/admin-guide/pm/cpufreq.rst for more info
#DO_CPU=0
```

```
# When on battery what is the maximum CPU speed that the system should
# use? Legal values are "slowest" for the slowest speed that your
# CPU is able to operate at, or a value listed in:
# /sys/devices/system/cpu/cpu0/cpufreq/scaling_available_frequencies
# Only applicable if DO_CPU=1.
#CPU_MAXFREQ=slowest

# Idle timeout for your hard drive (man hdparm for valid values, -S option)
# Default is 2 hours on AC (AC_HD=244) and 20 seconds for battery (BATT_HD=4).
#AC_HD=244
#BATT_HD=4

# The drives for which to adjust the idle timeout. Separate them by a space,
# e.g. HD="/dev/hda /dev/hdb".
#HD="/dev/hda"

# Set the spindown timeout on a hard drive?
#DO_HD=1
```

53.3.8 Control script

Please note that this control script works for the Linux 2.4 and 2.6 series (thanks to Kiko Piris).

Control script:

```
#!/bin/bash

# start or stop laptop_mode, best run by a power management daemon when
# ac gets connected/disconnected from a laptop
#
# install as /sbin/laptop_mode
#
# Contributors to this script:   Kiko Piris
#                               Bart Samwel
#                               Micha Feigin
#                               Andrew Morton
#                               Herve Eychenne
#                               Dax Kelson
#
# Original Linux 2.4 version by: Jens Axboe

###############################################################################

# Source config
if [ -f /etc/default/laptop-mode ] ; then
      # Debian
      . /etc/default/laptop-mode
elif [ -f /etc/sysconfig/laptop-mode ] ; then
      # Others
        . /etc/sysconfig/laptop-mode
```

```
fi

# Don't raise an error if the config file is incomplete
# set defaults instead:

# Maximum time, in seconds, of hard drive spindown time that you are
# comfortable with. Worst case, it's possible that you could lose this
# amount of work if your battery fails you while in laptop mode.
MAX_AGE=${MAX_AGE:-'600'}

# Read-ahead, in kilobytes
READAHEAD=${READAHEAD:-'4096'}

# Shall we remount journaled fs. with appropriate commit interval? (1=yes)
DO_REMOUNTS=${DO_REMOUNTS:-'1'}

# And shall we add the "noatime" option to that as well? (1=yes)
DO_REMOUNT_NOATIME=${DO_REMOUNT_NOATIME:-'1'}

# Shall we adjust the idle timeout on a hard drive?
DO_HD=${DO_HD:-'1'}

# Adjust idle timeout on which hard drive?
HD="${HD:-'/dev/hda'}"

# spindown time for HD (hdparm -S values)
AC_HD=${AC_HD:-'244'}
BATT_HD=${BATT_HD:-'4'}

# Dirty synchronous ratio.  At this percentage of dirty pages the process which
# calls write() does its own writeback
DIRTY_RATIO=${DIRTY_RATIO:-'40'}

# cpu frequency scaling
# See Documentation/admin-guide/pm/cpufreq.rst for more info
DO_CPU=${CPU_MANAGE:-'0'}
CPU_MAXFREQ=${CPU_MAXFREQ:-'slowest'}

#
# Allowed dirty background ratio, in percent.  Once DIRTY_RATIO has been
# exceeded, the kernel will wake flusher threads which will then reduce the
# amount of dirty memory to dirty_background_ratio.  Set this nice and low,
# so once some writeout has commenced, we do a lot of it.
#
DIRTY_BACKGROUND_RATIO=${DIRTY_BACKGROUND_RATIO:-'5'}

# kernel default dirty buffer age
DEF_AGE=${DEF_AGE:-'30'}
DEF_UPDATE=${DEF_UPDATE:-'5'}
DEF_DIRTY_BACKGROUND_RATIO=${DEF_DIRTY_BACKGROUND_RATIO:-'10'}
```

```
DEF_DIRTY_RATIO=${DEF_DIRTY_RATIO:-'40'}
DEF_XFS_AGE_BUFFER=${DEF_XFS_AGE_BUFFER:-'15'}
DEF_XFS_SYNC_INTERVAL=${DEF_XFS_SYNC_INTERVAL:-'30'}
DEF_XFS_BUFD_INTERVAL=${DEF_XFS_BUFD_INTERVAL:-'1'}

# This must be adjusted manually to the value of HZ in the running kernel
# on 2.4, until the XFS people change their 2.4 external interfaces to work in
# centisecs. This can be automated, but it's a work in progress that still␣
↪needs
# some fixes. On 2.6 kernels, XFS uses USER_HZ instead of HZ for external
# interfaces, and that is currently always set to 100. So you don't need to
# change this on 2.6.
XFS_HZ=${XFS_HZ:-'100'}

##########################################################################

KLEVEL="$(uname -r |
             {
             IFS='.' read a b c
             echo $a.$b
           }
)"
case "$KLEVEL" in
      "2.4"|"2.6")
               ;;
      *)
               echo "Unhandled kernel version: $KLEVEL ('uname -r' = '$(uname -
↪r)')" >&2
               exit 1
               ;;
esac

if [ ! -e /proc/sys/vm/laptop_mode ] ; then
      echo "Kernel is not patched with laptop_mode patch." >&2
      exit 1
fi

if [ ! -w /proc/sys/vm/laptop_mode ] ; then
      echo "You do not have enough privileges to enable laptop_mode." >&2
      exit 1
fi

# Remove an option (the first parameter) of the form option=<number> from
# a mount options string (the rest of the parameters).
parse_mount_opts () {
      OPT="$1"
      shift
      echo ",$*," | sed                \
       -e 's/,'"$OPT"'=[0-9]*,/,/g'    \
       -e 's/,,*/,/g'                  \
```

```
       -e 's/^,//'                    \
       -e 's/,$//'
}

# Remove an option (the first parameter) without any arguments from
# a mount option string (the rest of the parameters).
parse_nonumber_mount_opts () {
      OPT="$1"
      shift
      echo ",$*," | sed               \
       -e 's/,'"$OPT"',/,/g'          \
       -e 's/,,*/,/g'                 \
       -e 's/^,//'                    \
       -e 's/,$//'
}

# Find out the state of a yes/no option (e.g. "atime"/"noatime") in
# fstab for a given filesystem, and use this state to replace the
# value of the option in another mount options string. The device
# is the first argument, the option name the second, and the default
# value the third. The remainder is the mount options string.
#
# Example:
# parse_yesno_opts_wfstab /dev/hda1 atime atime defaults,noatime
#
# If fstab contains, say, "rw" for this filesystem, then the result
# will be "defaults,atime".
parse_yesno_opts_wfstab () {
      L_DEV="$1"
      OPT="$2"
      DEF_OPT="$3"
      shift 3
      L_OPTS="$*"
      PARSEDOPTS1="$(parse_nonumber_mount_opts $OPT $L_OPTS)"
      PARSEDOPTS1="$(parse_nonumber_mount_opts no$OPT $PARSEDOPTS1)"
      # Watch for a default atime in fstab
      FSTAB_OPTS="$(awk '$1 == "'$L_DEV'" { print $4 }' /etc/fstab)"
      if echo "$FSTAB_OPTS" | grep "$OPT" > /dev/null ; then
              # option specified in fstab: extract the value and use it
              if echo "$FSTAB_OPTS" | grep "no$OPT" > /dev/null ; then
                      echo "$PARSEDOPTS1,no$OPT"
              else
                      # no$OPT not found -- so we must have $OPT.
                      echo "$PARSEDOPTS1,$OPT"
              fi
      else
              # option not specified in fstab -- choose the default.
              echo "$PARSEDOPTS1,$DEF_OPT"
      fi
}
```

```
# Find out the state of a numbered option (e.g. "commit=NNN") in
# fstab for a given filesystem, and use this state to replace the
# value of the option in another mount options string. The device
# is the first argument, and the option name the second. The
# remainder is the mount options string in which the replacement
# must be done.
#
# Example:
# parse_mount_opts_wfstab /dev/hda1 commit defaults,commit=7
#
# If fstab contains, say, "commit=3,rw" for this filesystem, then the
# result will be "rw,commit=3".
parse_mount_opts_wfstab () {
      L_DEV="$1"
      OPT="$2"
      shift 2
      L_OPTS="$*"
      PARSEDOPTS1="$(parse_mount_opts $OPT $L_OPTS)"
      # Watch for a default commit in fstab
      FSTAB_OPTS="$(awk '$1 == "'$L_DEV'" { print $4 }' /etc/fstab)"
      if echo "$FSTAB_OPTS" | grep "$OPT=" > /dev/null ; then
              # option specified in fstab: extract the value, and use it
              echo -n "$PARSEDOPTS1,$OPT="
              echo ",$FSTAB_OPTS," | sed \
               -e 's/.*,'"$OPT"'=//'  \
               -e 's/,.*//'
      else
              # option not specified in fstab: set it to 0
              echo "$PARSEDOPTS1,$OPT=0"
      fi
}

deduce_fstype () {
      MP="$1"
      # My root filesystem unfortunately has
      # type "unknown" in /etc/mtab. If we encounter
      # "unknown", we try to get the type from fstab.
      cat /etc/fstab |
      grep -v '^#' |
      while read FSTAB_DEV FSTAB_MP FSTAB_FST FSTAB_OPTS FSTAB_DUMP FSTAB_DUMP␣
↪; do
              if [ "$FSTAB_MP" = "$MP" ]; then
                      echo $FSTAB_FST
                      exit 0
              fi
      done
}

if [ $DO_REMOUNT_NOATIME -eq 1 ] ; then
```

```
      NOATIME_OPT=",noatime"
fi

case "$1" in
      start)
              AGE=$((100*$MAX_AGE))
              XFS_AGE=$(($XFS_HZ*$MAX_AGE))
              echo -n "Starting laptop_mode"

              if [ -d /proc/sys/vm/pagebuf ] ; then
                      # (For 2.4 and early 2.6.)
                      # This only needs to be set, not reset -- it is only␣
↪used when
                      # laptop mode is enabled.
                      echo $XFS_AGE > /proc/sys/vm/pagebuf/lm_flush_age
                      echo $XFS_AGE > /proc/sys/fs/xfs/lm_sync_interval
              elif [ -f /proc/sys/fs/xfs/lm_age_buffer ] ; then
                      # (A couple of early 2.6 laptop mode patches had these.)
                      # The same goes for these.
                      echo $XFS_AGE > /proc/sys/fs/xfs/lm_age_buffer
                      echo $XFS_AGE > /proc/sys/fs/xfs/lm_sync_interval
              elif [ -f /proc/sys/fs/xfs/age_buffer ] ; then
                      # (2.6.6)
                      # But not for these -- they are also used in normal
                      # operation.
                      echo $XFS_AGE > /proc/sys/fs/xfs/age_buffer
                      echo $XFS_AGE > /proc/sys/fs/xfs/sync_interval
              elif [ -f /proc/sys/fs/xfs/age_buffer_centisecs ] ; then
                      # (2.6.7 upwards)
                      # And not for these either. These are in centisecs,
                      # not USER_HZ, so we have to use $AGE, not $XFS_AGE.
                      echo $AGE > /proc/sys/fs/xfs/age_buffer_centisecs
                      echo $AGE > /proc/sys/fs/xfs/xfssyncd_centisecs
                      echo 3000 > /proc/sys/fs/xfs/xfsbufd_centisecs
              fi

              case "$KLEVEL" in
                      "2.4")
                              echo 1                                  > /proc/
↪sys/vm/laptop_mode
                              echo "30 500 0 0 $AGE $AGE 60 20 0"     > /proc/
↪sys/vm/bdflush
                              ;;
                      "2.6")
                              echo 5                                  > /proc/
↪sys/vm/laptop_mode
                              echo "$AGE"                             > /proc/
↪sys/vm/dirty_writeback_centisecs
                              echo "$AGE"                             > /proc/
↪sys/vm/dirty_expire_centisecs
```

```
                                echo "$DIRTY_RATIO"                    > /proc/
↪sys/vm/dirty_ratio
                                echo "$DIRTY_BACKGROUND_RATIO"         > /proc/
↪sys/vm/dirty_background_ratio
                                ;;
                esac
                if [ $DO_REMOUNTS -eq 1 ]; then
                        cat /etc/mtab | while read DEV MP FST OPTS DUMP PASS ; do
                                PARSEDOPTS="$(parse_mount_opts "$OPTS")"
                                if [ "$FST" = 'unknown' ]; then
                                        FST=$(deduce_fstype $MP)
                                fi
                                case "$FST" in
                                        "ext3"|"reiserfs")
                                                PARSEDOPTS="$(parse_mount_opts␣
↪commit "$OPTS")"
                                                mount $DEV -t $FST $MP -o␣
↪remount,$PARSEDOPTS,commit=$MAX_AGE$NOATIME_OPT
                                                ;;
                                        "xfs")
                                                mount $DEV -t $FST $MP -o␣
↪remount,$OPTS$NOATIME_OPT
                                                ;;
                                esac
                                if [ -b $DEV ] ; then
                                        blockdev --setra $(($READAHEAD * 2)) $DEV
                                fi
                        done
                fi
                if [ $DO_HD -eq 1 ] ; then
                        for THISHD in $HD ; do
                                /sbin/hdparm -S $BATT_HD $THISHD > /dev/null 2>&1
                                /sbin/hdparm -B 1 $THISHD > /dev/null 2>&1
                        done
                fi
                if [ $DO_CPU -eq 1 -a -e /sys/devices/system/cpu/cpu0/cpufreq/
↪cpuinfo_min_freq ]; then
                        if [ $CPU_MAXFREQ = 'slowest' ]; then
                                CPU_MAXFREQ=`cat /sys/devices/system/cpu/cpu0/
↪cpufreq/cpuinfo_min_freq`
                        fi
                        echo $CPU_MAXFREQ > /sys/devices/system/cpu/cpu0/cpufreq/
↪scaling_max_freq
                fi
                echo "."
                ;;
        stop)
                U_AGE=$((100*$DEF_UPDATE))
                B_AGE=$((100*$DEF_AGE))
                echo -n "Stopping laptop_mode"
```

```
            echo 0 > /proc/sys/vm/laptop_mode
            if [ -f /proc/sys/fs/xfs/age_buffer -a ! -f /proc/sys/fs/xfs/lm_
↪age_buffer ] ; then
                    # These need to be restored, if there are no lm_*.
                    echo $(($XFS_HZ*$DEF_XFS_AGE_BUFFER))           > /proc/
↪sys/fs/xfs/age_buffer
                    echo $(($XFS_HZ*$DEF_XFS_SYNC_INTERVAL))        > /proc/
↪sys/fs/xfs/sync_interval
            elif [ -f /proc/sys/fs/xfs/age_buffer_centisecs ] ; then
                    # These need to be restored as well.
                    echo $((100*$DEF_XFS_AGE_BUFFER))       > /proc/sys/fs/
↪xfs/age_buffer_centisecs
                    echo $((100*$DEF_XFS_SYNC_INTERVAL))    > /proc/sys/fs/
↪xfs/xfssyncd_centisecs
                    echo $((100*$DEF_XFS_BUFD_INTERVAL))    > /proc/sys/fs/
↪xfs/xfsbufd_centisecs
            fi
            case "$KLEVEL" in
                    "2.4")
                            echo "30 500 0 0 $U_AGE $B_AGE 60 20 0" > /proc/
↪sys/vm/bdflush
                            ;;
                    "2.6")
                            echo "$U_AGE"                           > /proc/
↪sys/vm/dirty_writeback_centisecs
                            echo "$B_AGE"                           > /proc/
↪sys/vm/dirty_expire_centisecs
                            echo "$DEF_DIRTY_RATIO"                 > /proc/
↪sys/vm/dirty_ratio
                            echo "$DEF_DIRTY_BACKGROUND_RATIO"      > /proc/
↪sys/vm/dirty_background_ratio
                            ;;
            esac
            if [ $DO_REMOUNTS -eq 1 ] ; then
                    cat /etc/mtab | while read DEV MP FST OPTS DUMP PASS ; do
                            # Reset commit and atime options to defaults.
                            if [ "$FST" = 'unknown' ]; then
                                    FST=$(deduce_fstype $MP)
                            fi
                            case "$FST" in
                                    "ext3"|"reiserfs")
                                            PARSEDOPTS="$(parse_mount_opts_
↪wfstab $DEV commit $OPTS)"
                                            PARSEDOPTS="$(parse_yesno_opts_
↪wfstab $DEV atime atime $PARSEDOPTS)"
                                            mount $DEV -t $FST $MP -o␣
↪remount,$PARSEDOPTS
                                            ;;
                                    "xfs")
                                            PARSEDOPTS="$(parse_yesno_opts_
↪wfstab $DEV atime atime $OPTS)"
```

```
                                                mount $DEV -t $FST $MP -o␣
↪remount,$PARSEDOPTS
                                                ;;
                                esac
                                if [ -b $DEV ] ; then
                                        blockdev --setra 256 $DEV
                                fi
                        done
                fi
                if [ $DO_HD -eq 1 ] ; then
                        for THISHD in $HD ; do
                                /sbin/hdparm -S $AC_HD $THISHD > /dev/null 2>&1
                                /sbin/hdparm -B 255 $THISHD > /dev/null 2>&1
                        done
                fi
                if [ $DO_CPU -eq 1 -a -e /sys/devices/system/cpu/cpu0/cpufreq/
↪cpuinfo_min_freq ]; then
                        echo `cat /sys/devices/system/cpu/cpu0/cpufreq/cpuinfo_
↪max_freq` > /sys/devices/system/cpu/cpu0/cpufreq/scaling_max_freq
                fi
                echo "."
                ;;
        *)
                echo "Usage: $0 {start|stop}" 2>&1
                exit 1
                ;;

esac

exit 0
```

53.3.9 ACPI integration

Dax Kelson submitted this so that the ACPI acpid daemon will kick off the laptop_mode script and run hdparm. The part that automatically disables laptop mode when the battery is low was written by Jan Topinski.

/etc/acpi/events/ac_adapter:

```
event=ac_adapter
action=/etc/acpi/actions/ac.sh %e
```

/etc/acpi/events/battery:

```
event=battery.*
action=/etc/acpi/actions/battery.sh %e
```

/etc/acpi/actions/ac.sh:

```
#!/bin/bash
```

```
# ac on/offline event handler

status=`awk '/^state: / { print $2 }' /proc/acpi/ac_adapter/$2/state`

case $status in
        "on-line")
                /sbin/laptop_mode stop
                exit 0
        ;;
        "off-line")
                /sbin/laptop_mode start
                exit 0
        ;;
esac
```

/etc/acpi/actions/battery.sh:

```
#! /bin/bash

# Automatically disable laptop mode when the battery almost runs out.

BATT_INFO=/proc/acpi/battery/$2/state

if [[ -f /proc/sys/vm/laptop_mode ]]
then
   LM=`cat /proc/sys/vm/laptop_mode`
   if [[ $LM -gt 0 ]]
   then
     if [[ -f $BATT_INFO ]]
     then
        # Source the config file only now that we know we need
        if [ -f /etc/default/laptop-mode ] ; then
                # Debian
                . /etc/default/laptop-mode
        elif [ -f /etc/sysconfig/laptop-mode ] ; then
                # Others
                . /etc/sysconfig/laptop-mode
        fi
        MINIMUM_BATTERY_MINUTES=${MINIMUM_BATTERY_MINUTES:-'10'}

        ACTION="`cat $BATT_INFO | grep charging | cut -c 26-`"
        if [[ ACTION -eq "discharging" ]]
        then
           PRESENT_RATE=`cat $BATT_INFO | grep "present rate:" | sed  "s/.* \
↪([0-9][0-9]* \).*/\1/" `
           REMAINING=`cat $BATT_INFO | grep "remaining capacity:" | sed  "s/.*␣
↪\([0-9][0-9]* \).*/\1/" `
        fi
        if (($REMAINING * 60 / $PRESENT_RATE < $MINIMUM_BATTERY_MINUTES))
        then
           /sbin/laptop_mode stop
```

```
        fi
     else
       logger -p daemon.warning "You are using laptop mode and your battery␣
↪interface $BATT_INFO is missing. This may lead to loss of data when the␣
↪battery runs out. Check kernel ACPI support and /proc/acpi/battery folder,␣
↪and edit /etc/acpi/battery.sh to set BATT_INFO to the correct path."
     fi
   fi
fi
```

53.3.10 Monitoring tool

Bartek Kania submitted this, it can be used to measure how much time your disk spends spun up/down. See tools/laptop/dslm/dslm.c

53.4 LG Gram laptop extra features

By Matan Ziv-Av <matan@svgalib.org>

53.4.1 Hotkeys

The following FN keys are ignored by the kernel without this driver:

- FN-F1 (LG control panel) - Generates F15
- FN-F5 (Touchpad toggle) - Generates F21
- FN-F6 (Airplane mode) - Generates RFKILL
- FN-F9 (Reader mode) - Generates F14

The rest of the FN keys work without a need for a special driver.

53.4.2 Reader mode

Writing 0/1 to /sys/devices/platform/lg-laptop/reader_mode disables/enables reader mode. In this mode the screen colors change (blue color reduced), and the reader mode indicator LED (on F9 key) turns on.

53.4.3 FN Lock

Writing 0/1 to /sys/devices/platform/lg-laptop/fn_lock disables/enables FN lock.

53.4.4 Battery care limit

Writing 80/100 to /sys/class/power_supply/CMB0/charge_control_end_threshold sets the maximum capacity to charge the battery. Limiting the charge reduces battery capacity loss over time.

This value is reset to 100 when the kernel boots.

53.4.5 Fan mode

Writing 1/0 to /sys/devices/platform/lg-laptop/fan_mode disables/enables the fan silent mode.

53.4.6 USB charge

Writing 0/1 to /sys/devices/platform/lg-laptop/usb_charge disables/enables charging another device from the USB port while the device is turned off.

This value is reset to 0 when the kernel boots.

LEDs

The are two LED devices supported by the driver:

53.4.7 Keyboard backlight

A led device named kbd_led controls the keyboard backlight. There are three lighting level: off (0), low (127) and high (255).

The keyboard backlight is also controlled by the key combination FN-F8 which cycles through those levels.

53.4.8 Touchpad indicator LED

On the F5 key. Controlled by led device names tpad_led.

53.5 Sony Notebook Control Driver (SNC) Readme

- Copyright (C) 2004- 2005 Stelian Pop <stelian@popies.net>
- Copyright (C) 2007 Mattia Dongili <malattia@linux.it>

This mini-driver drives the SNC and SPIC device present in the ACPI BIOS of the Sony Vaio laptops. This driver mixes both devices functions under the same (hopefully consistent) interface. This also means that the sonypi driver is obsoleted by sony-laptop now.

53.5.1 Fn keys (hotkeys):

Some models report hotkeys through the SNC or SPIC devices, such events are reported both through the ACPI subsystem as acpi events and through the INPUT subsystem. See the logs of /proc/bus/input/devices to find out what those events are and which input devices are created by the driver. Additionally, loading the driver with the debug option will report all events in the kernel log.

The "scancodes" passed to the input system (that can be remapped with udev) are indexes to the table "sony_laptop_input_keycode_map" in the sony-laptop.c module. For example the "FN/E" key combination (EJECTCD on some models) generates the scancode 20 (0x14).

53.5.2 Backlight control:

If your laptop model supports it, you will find sysfs files in the /sys/class/backlight/sony/ directory. You will be able to query and set the current screen brightness:

brightness	get/set screen brightness (an integer between 0 and 7)
actual_brightness	reading from this file will query the HW to get real brightness value
max_brightness	the maximum brightness value

53.5.3 Platform specific:

Loading the sony-laptop module will create a /sys/devices/platform/sony-laptop/ directory populated with some files.

You then read/write integer values from/to those files by using standard UNIX tools.

The files are:

brightness_default	screen brightness which will be set when the laptop will be rebooted
cdpower	power on/off the internal CD drive
audiopower	power on/off the internal sound card
lanpower	power on/off the internal ethernet card (only in debug mode)
bluetoothpower	power on/off the internal bluetooth device
fanspeed	get/set the fan speed

Note that some files may be missing if they are not supported by your particular laptop model.

Example usage:

```
# echo "1" > /sys/devices/platform/sony-laptop/brightness_default
```

sets the lowest screen brightness for the next and later reboots

```
# echo "8" > /sys/devices/platform/sony-laptop/brightness_default
```

sets the highest screen brightness for the next and later reboots

```
# cat /sys/devices/platform/sony-laptop/brightness_default
```

retrieves the value

```
# echo "0" > /sys/devices/platform/sony-laptop/audiopower
```

powers off the sound card

```
# echo "1" > /sys/devices/platform/sony-laptop/audiopower
```

powers on the sound card.

53.5.4 RFkill control:

More recent Vaio models expose a consistent set of ACPI methods to control radio frequency emitting devices. If you are a lucky owner of such a laptop you will find the necessary rfkill devices under /sys/class/rfkill. Check those starting with sony-* in:

```
# grep . /sys/class/rfkill/*/{state,name}
```

53.5.5 Development:

If you want to help with the development of this driver (and you are not afraid of any side effects doing strange things with your ACPI BIOS could have on your laptop), load the driver and pass the option 'debug=1'.

REPEAT:
DON'T DO THIS IF YOU DON'T LIKE RISKY BUSINESS.

In your kernel logs you will find the list of all ACPI methods the SNC device has on your laptop.

- For new models you will see a long list of meaningless method names, reading the DSDT table source should reveal that:

(1) the SNC device uses an internal capability lookup table

(2) SN00 is used to find values in the lookup table

(3) SN06 and SN07 are used to call into the real methods based on offsets you can obtain iterating the table using SN00

(4) SN02 used to enable events.

Some values in the capability lookup table are more or less known, see the code for all sony_call_snc_handle calls, others are more obscure.

- For old models you can see the GCDP/GCDP methods used to pwer on/off the CD drive, but there are others and they are usually different from model to model.

I HAVE NO IDEA WHAT THOSE METHODS DO.

The sony-laptop driver creates, for some of those methods (the most current ones found on several Vaio models), an entry under /sys/devices/platform/sony-laptop, just like the 'cdpower' one. You can create other entries corresponding to your own laptop methods by further editing

the source (see the 'sony_nc_values' table, and add a new entry to this table with your get/set method names using the SNC_HANDLE_NAMES macro).

Your mission, should you accept it, is to try finding out what those entries are for, by reading/writing random values from/to those files and find out what is the impact on your laptop.

Should you find anything interesting, please report it back to me, I will not disavow all knowledge of your actions :)

See also http://www.linux.it/~malattia/wiki/index.php/Sony_drivers for other useful info.

53.5.6 Bugs/Limitations:

- This driver is not based on official documentation from Sony (because there is none), so there is no guarantee this driver will work at all, or do the right thing. Although this hasn't happened to me, this driver could do very bad things to your laptop, including permanent damage.
- The sony-laptop and sonypi drivers do not interact at all. In the future, sonypi will be removed and replaced by sony-laptop.
- spicctrl, which is the userspace tool used to communicate with the sonypi driver (through /dev/sonypi) is deprecated as well since all its features are now available under the sysfs tree via sony-laptop.

53.6 Sony Programmable I/O Control Device Driver Readme

- Copyright (C) 2001-2004 Stelian Pop <stelian@popies.net>
- Copyright (C) 2001-2002 Alcôve <www.alcove.com>
- Copyright (C) 2001 Michael Ashley <m.ashley@unsw.edu.au>
- Copyright (C) 2001 Junichi Morita <jun1m@mars.dti.ne.jp>
- Copyright (C) 2000 Takaya Kinjo <t-kinjo@tc4.so-net.ne.jp>
- Copyright (C) 2000 Andrew Tridgell <tridge@samba.org>

This driver enables access to the Sony Programmable I/O Control Device which can be found in many Sony Vaio laptops. Some newer Sony laptops (seems to be limited to new FX series laptops, at least the FX501 and the FX702) lack a sonypi device and are not supported at all by this driver.

It will give access (through a user space utility) to some events those laptops generate, like:

- jogdial events (the small wheel on the side of Vaios)
- capture button events (only on Vaio Picturebook series)
- Fn keys
- bluetooth button (only on C1VR model)
- programmable keys, back, help, zoom, thumbphrase buttons, etc. (when available)

Those events (see linux/sonypi.h) can be polled using the character device node /dev/sonypi (major 10, minor auto allocated or specified as a option). A simple daemon which translates the jogdial movements into mouse wheel events can be downloaded at: <http://popies.net/sonypi/>

Another option to intercept the events is to get them directly through the input layer.

This driver supports also some ioctl commands for setting the LCD screen brightness and querying the batteries charge information (some more commands may be added in the future).

This driver can also be used to set the camera controls on Picturebook series (brightness, contrast etc), and is used by the video4linux driver for the Motion Eye camera.

Please note that this driver was created by reverse engineering the Windows driver and the ACPI BIOS, because Sony doesn't agree to release any programming specs for its laptops. If someone convinces them to do so, drop me a note.

53.6.1 Driver options:

Several options can be passed to the sonypi driver using the standard module argument syntax (<param>=<value> when passing the option to the module or sonypi.<param>=<value> on the kernel boot line when sonypi is statically linked into the kernel). Those options are:

minor:	minor number of the misc device /dev/sonypi, default is -1 (automatic allocation, see /proc/misc or kernel logs)
camera:	if you have a PictureBook series Vaio (with the integrated MotionEye camera), set this parameter to 1 in order to let the driver access to the camera
fnkeyinit:	on some Vaios (C1VE, C1VR etc), the Fn key events don't get enabled unless you set this parameter to 1. Do not use this option unless it's actually necessary, some Vaio models don't deal well with this option. This option is available only if the kernel is compiled without ACPI support (since it conflicts with it and it shouldn't be required anyway if ACPI is already enabled).
verbose:	set to 1 to print unknown events received from the sonypi device. set to 2 to print all events received from the sonypi device.
compat:	uses some compatibility code for enabling the sonypi events. If the driver worked for you in the past (prior to version 1.5) and does not work anymore, add this option and report to the author.
mask:	event mask telling the driver what events will be reported to the user. This parameter is required for some Vaio models where the hardware reuses values used in other Vaio models (like the FX series who does not have a jogdial but reuses the jogdial events for programmable keys events). The default event mask is set to 0xffffffff, meaning that all possible events will be tried. You can use the following bits to construct your own event mask (from drivers/char/sonypi.h):

```
SONYPI_JOGGER_MASK                                  ␣
↪0x0001
SONYPI_CAPTURE_MASK                                 ␣
↪0x0002
SONYPI_FNKEY_MASK                                   ␣
↪0x0004
SONYPI_BLUETOOTH_MASK                               ␣
↪0x0008
SONYPI_PKEY_MASK                                    ␣
↪0x0010
SONYPI_BACK_MASK                                    ␣
↪0x0020
SONYPI_HELP_MASK                                    ␣
↪0x0040
SONYPI_LID_MASK                                     ␣
↪0x0080
```

53.6.2 Module use:

In order to automatically load the sonypi module on use, you can put those lines a configuration file in /etc/modprobe.d/:

```
alias char-major-10-250 sonypi
options sonypi minor=250
```

This supposes the use of minor 250 for the sonypi device:

```
# mknod /dev/sonypi c 10 250
```

53.6.3 Bugs:

- several users reported that this driver disables the BIOS-managed Fn-keys which put the laptop in sleeping state, or switch the external monitor on/off. There is no workaround yet, since this driver disables all APM management for those keys, by enabling the ACPI management (and the ACPI core stuff is not complete yet). If you have one of those laptops with working Fn keys and want to continue to use them, don't use this driver.
- some users reported that the laptop speed is lower (dhrystone tested) when using the driver with the fnkeyinit parameter. I cannot reproduce it on my laptop and not all users have this problem. This happens because the fnkeyinit parameter enables the ACPI mode (but without additional ACPI control, like processor speed handling etc). Use ACPI instead of APM if it works on your laptop.
- sonypi lacks the ability to distinguish between certain key events on some models.
- some models with the nvidia card (geforce go 6200 tc) uses a different way to adjust the backlighting of the screen. There is a userspace utility to adjust the brightness on those models, which can be downloaded from https://www.acc.umu.se/~erikw/program/smartdimmer-0.1.tar.bz2
- since all development was done by reverse engineering, there is *absolutely no guarantee* that this driver will not crash your laptop. Permanently.

53.7 ThinkPad ACPI Extras Driver

Version 0.25

October 16th, 2013

- Borislav Deianov <borislav@users.sf.net>
- Henrique de Moraes Holschuh <hmh@hmh.eng.br>

http://ibm-acpi.sf.net/

This is a Linux driver for the IBM and Lenovo ThinkPad laptops. It supports various features of these laptops which are accessible through the ACPI and ACPI EC framework, but not otherwise fully supported by the generic Linux ACPI drivers.

This driver used to be named ibm-acpi until kernel 2.6.21 and release 0.13-20070314. It used to be in the drivers/acpi tree, but it was moved to the drivers/misc tree and renamed to thinkpad-acpi for kernel 2.6.22, and release 0.14. It was moved to drivers/platform/x86 for kernel 2.6.29 and release 0.22.

The driver is named ”thinkpad-acpi”. In some places, like module names and log messages, ”thinkpad_acpi” is used because of userspace issues.

”tpacpi” is used as a shorthand where ”thinkpad-acpi” would be too long due to length limitations on some Linux kernel versions.

53.7.1 Status

The features currently supported are the following (see below for detailed description):

- Fn key combinations
- Bluetooth enable and disable
- video output switching, expansion control
- ThinkLight on and off
- CMOS/UCMS control
- LED control
- ACPI sounds
- temperature sensors
- Experimental: embedded controller register dump
- LCD brightness control
- Volume control
- Fan control and monitoring: fan speed, fan enable/disable
- WAN enable and disable
- UWB enable and disable
- LCD Shadow (PrivacyGuard) enable and disable
- Lap mode sensor
- Setting keyboard language
- WWAN Antenna type
- Auxmac

A compatibility table by model and feature is maintained on the web site, http://ibm-acpi.sf.net/. I appreciate any success or failure reports, especially if they add to or correct the compatibility table. Please include the following information in your report:

- ThinkPad model name
- a copy of your ACPI tables, using the ”acpidump” utility
- a copy of the output of dmidecode, with serial numbers and UUIDs masked off
- which driver features work and which don't

- the observed behavior of non-working features

Any other comments or patches are also more than welcome.

53.7.2 Installation

If you are compiling this driver as included in the Linux kernel sources, look for the CONFIG_THINKPAD_ACPI Kconfig option. It is located on the menu path: ”Device Drivers” -> ”X86 Platform Specific Device Drivers” -> ”ThinkPad ACPI Laptop Extras”.

53.7.3 Features

The driver exports two different interfaces to userspace, which can be used to access the features it provides. One is a legacy procfs-based interface, which will be removed at some time in the future. The other is a new sysfs-based interface which is not complete yet.

The procfs interface creates the /proc/acpi/ibm directory. There is a file under that directory for each feature it supports. The procfs interface is mostly frozen, and will change very little if at all: it will not be extended to add any new functionality in the driver, instead all new functionality will be implemented on the sysfs interface.

The sysfs interface tries to blend in the generic Linux sysfs subsystems and classes as much as possible. Since some of these subsystems are not yet ready or stabilized, it is expected that this interface will change, and any and all userspace programs must deal with it.

Notes about the sysfs interface

Unlike what was done with the procfs interface, correctness when talking to the sysfs interfaces will be enforced, as will correctness in the thinkpad-acpi's implementation of sysfs interfaces.

Also, any bugs in the thinkpad-acpi sysfs driver code or in the thinkpad-acpi's implementation of the sysfs interfaces will be fixed for maximum correctness, even if that means changing an interface in non-compatible ways. As these interfaces mature both in the kernel and in thinkpad-acpi, such changes should become quite rare.

Applications interfacing to the thinkpad-acpi sysfs interfaces must follow all sysfs guidelines and correctly process all errors (the sysfs interface makes extensive use of errors). File descriptors and open / close operations to the sysfs inodes must also be properly implemented.

The version of thinkpad-acpi's sysfs interface is exported by the driver as a driver attribute (see below).

Sysfs driver attributes are on the driver's sysfs attribute space, for 2.6.23+ this is /sys/bus/platform/drivers/thinkpad_acpi/ and /sys/bus/platform/drivers/thinkpad_hwmon/

Sysfs device attributes are on the thinkpad_acpi device sysfs attribute space, for 2.6.23+ this is /sys/devices/platform/thinkpad_acpi/.

Sysfs device attributes for the sensors and fan are on the thinkpad_hwmon device's sysfs attribute space, but you should locate it looking for a hwmon device with the name attribute of ”thinkpad”, or better yet, through libsensors. For 4.14+ sysfs attributes were moved to the hwmon device (/sys/bus/platform/devices/thinkpad_hwmon/hwmon/hwmon? or /sys/class/hwmon/hwmon?).

53.7.4 Driver version

procfs: /proc/acpi/ibm/driver

sysfs driver attribute: version

The driver name and version. No commands can be written to this file.

53.7.5 Sysfs interface version

sysfs driver attribute: interface_version

Version of the thinkpad-acpi sysfs interface, as an unsigned long (output in hex format: 0xAAAABBCC), where:

AAAA

- major revision

BB

- minor revision

CC

- bugfix revision

The sysfs interface version changelog for the driver can be found at the end of this document. Changes to the sysfs interface done by the kernel subsystems are not documented here, nor are they tracked by this attribute.

Changes to the thinkpad-acpi sysfs interface are only considered non-experimental when they are submitted to Linux mainline, at which point the changes in this interface are documented and interface_version may be updated. If you are using any thinkpad-acpi features not yet sent to mainline for merging, you do so on your own risk: these features may disappear, or be implemented in a different and incompatible way by the time they are merged in Linux mainline.

Changes that are backwards-compatible by nature (e.g. the addition of attributes that do not change the way the other attributes work) do not always warrant an update of interface_version. Therefore, one must expect that an attribute might not be there, and deal with it properly (an attribute not being there *is* a valid way to make it clear that a feature is not available in sysfs).

53.7.6 Hot keys

procfs: /proc/acpi/ibm/hotkey

sysfs device attribute: hotkey_*

In a ThinkPad, the ACPI HKEY handler is responsible for communicating some important events and also keyboard hot key presses to the operating system. Enabling the hotkey functionality of thinkpad-acpi signals the firmware that such a driver is present, and modifies how the ThinkPad firmware will behave in many situations.

The driver enables the HKEY (”hot key”) event reporting automatically when loaded, and disables it when it is removed.

The driver will report HKEY events in the following format:

```
ibm/hotkey HKEY 00000080 0000xxxx
```

Some of these events refer to hot key presses, but not all of them.

The driver will generate events over the input layer for hot keys and radio switches, and over the ACPI netlink layer for other events. The input layer support accepts the standard IOCTLs to remap the keycodes assigned to each hot key.

The hot key bit mask allows some control over which hot keys generate events. If a key is "masked" (bit set to 0 in the mask), the firmware will handle it. If it is "unmasked", it signals the firmware that thinkpad-acpi would prefer to handle it, if the firmware would be so kind to allow it (and it often doesn't!).

Not all bits in the mask can be modified. Not all bits that can be modified do anything. Not all hot keys can be individually controlled by the mask. Some models do not support the mask at all. The behaviour of the mask is, therefore, highly dependent on the ThinkPad model.

The driver will filter out any unmasked hotkeys, so even if the firmware doesn't allow disabling an specific hotkey, the driver will not report events for unmasked hotkeys.

Note that unmasking some keys prevents their default behavior. For example, if Fn+F5 is unmasked, that key will no longer enable/disable Bluetooth by itself in firmware.

Note also that not all Fn key combinations are supported through ACPI depending on the ThinkPad model and firmware version. On those ThinkPads, it is still possible to support some extra hotkeys by polling the "CMOS NVRAM" at least 10 times per second. The driver attempts to enables this functionality automatically when required.

procfs notes

The following commands can be written to the /proc/acpi/ibm/hotkey file:

```
echo 0xffffffff > /proc/acpi/ibm/hotkey -- enable all hot keys
echo 0 > /proc/acpi/ibm/hotkey -- disable all possible hot keys
... any other 8-hex-digit mask ...
echo reset > /proc/acpi/ibm/hotkey -- restore the recommended mask
```

The following commands have been deprecated and will cause the kernel to log a warning:

```
echo enable > /proc/acpi/ibm/hotkey -- does nothing
echo disable > /proc/acpi/ibm/hotkey -- returns an error
```

The procfs interface does not support NVRAM polling control. So as to maintain maximum bug-to-bug compatibility, it does not report any masks, nor does it allow one to manipulate the hot key mask when the firmware does not support masks at all, even if NVRAM polling is in use.

sysfs notes

hotkey_bios_enabled:
DEPRECATED, WILL BE REMOVED SOON.

Returns 0.

hotkey_bios_mask:
DEPRECATED, DON'T USE, WILL BE REMOVED IN THE FUTURE.

Returns the hot keys mask when thinkpad-acpi was loaded. Upon module unload, the hot keys mask will be restored to this value. This is always 0x80c, because those are the hotkeys that were supported by ancient firmware without mask support.

hotkey_enable:
DEPRECATED, WILL BE REMOVED SOON.

0: returns -EPERM 1: does nothing

hotkey_mask:
bit mask to enable reporting (and depending on the firmware, ACPI event generation) for each hot key (see above). Returns the current status of the hot keys mask, and allows one to modify it.

hotkey_all_mask:
bit mask that should enable event reporting for all supported hot keys, when echoed to hotkey_mask above. Unless you know which events need to be handled passively (because the firmware *will* handle them anyway), do *not* use hotkey_all_mask. Use hotkey_recommended_mask, instead. You have been warned.

hotkey_recommended_mask:
bit mask that should enable event reporting for all supported hot keys, except those which are always handled by the firmware anyway. Echo it to hotkey_mask above, to use. This is the default mask used by the driver.

hotkey_source_mask:
bit mask that selects which hot keys will the driver poll the NVRAM for. This is auto-detected by the driver based on the capabilities reported by the ACPI firmware, but it can be overridden at runtime.

Hot keys whose bits are set in hotkey_source_mask are polled for in NVRAM, and reported as hotkey events if enabled in hotkey_mask. Only a few hot keys are available through CMOS NVRAM polling.

Warning: when in NVRAM mode, the volume up/down/mute keys are synthesized according to changes in the mixer, which uses a single volume up or volume down hotkey press to unmute, as per the ThinkPad volume mixer user interface. When in ACPI event mode, volume up/down/mute events are reported by the firmware and can behave differently (and that behaviour changes with firmware version -- not just with firmware models -- as well as OSI(Linux) state).

hotkey_poll_freq:
frequency in Hz for hot key polling. It must be between 0 and 25 Hz. Polling is only carried out when strictly needed.

Setting hotkey_poll_freq to zero disables polling, and will cause hot key presses that require NVRAM polling to never be reported.

Setting hotkey_poll_freq too low may cause repeated pressings of the same hot key to be misreported as a single key press, or to not even be detected at all. The recommended polling frequency is 10Hz.

hotkey_radio_sw:
If the ThinkPad has a hardware radio switch, this attribute will read 0 if the switch is in the "radios disabled" position, and 1 if the switch is in the "radios enabled" position.

This attribute has poll()/select() support.

hotkey_tablet_mode:
If the ThinkPad has tablet capabilities, this attribute will read 0 if the ThinkPad is in normal mode, and 1 if the ThinkPad is in tablet mode.

This attribute has poll()/select() support.

wakeup_reason:
Set to 1 if the system is waking up because the user requested a bay ejection. Set to 2 if the system is waking up because the user requested the system to undock. Set to zero for normal wake-ups or wake-ups due to unknown reasons.

This attribute has poll()/select() support.

wakeup_hotunplug_complete:
Set to 1 if the system was waken up because of an undock or bay ejection request, and that request was successfully completed. At this point, it might be useful to send the system back to sleep, at the user's choice. Refer to HKEY events 0x4003 and 0x3003, below.

This attribute has poll()/select() support.

input layer notes

A Hot key is mapped to a single input layer EV_KEY event, possibly followed by an EV_MSC MSC_SCAN event that shall contain that key's scan code. An EV_SYN event will always be generated to mark the end of the event block.

Do not use the EV_MSC MSC_SCAN events to process keys. They are to be used as a helper to remap keys, only. They are particularly useful when remapping KEY_UNKNOWN keys.

The events are available in an input device, with the following id:

Bus	BUS_HOST
vendor	0x1014 (PCI_VENDOR_ID_IBM) or 0x17aa (PCI_VENDOR_ID_LENOVO)
product	0x5054 ("TP")
version	0x4101

The version will have its LSB incremented if the keymap changes in a backwards-compatible way. The MSB shall always be 0x41 for this input device. If the MSB is not 0x41, do not use the device as described in this section, as it is either something else (e.g. another input device exported by a thinkpad driver, such as HDAPS) or its functionality has been changed in a non-backwards compatible way.

Adding other event types for other functionalities shall be considered a backwards-compatible change for this input device.

Thinkpad-acpi Hot Key event map (version 0x4101):

ACPI event	Scan code	Key	Notes
0x1001	0x00	FN+F1	•
0x1002	0x01	FN+F2	IBM: battery (rare) Lenovo: Screen lock
0x1003	0x02	FN+F3	Many IBM models always report this hot key, even with hot keys disabled or with Fn+F3 masked off IBM: screen lock, often turns off the ThinkLight as side-effect Lenovo: battery
0x1004	0x03	FN+F4	Sleep button (ACPI sleep button semantics, i.e. sleep-to-RAM). It always generates some kind of event, either the hot key event or an ACPI sleep button event. The firmware may refuse to generate further FN+F4 key presses until a S3 or S4 ACPI sleep cycle is performed or some time passes.
0x1005	0x04	FN+F5	Radio. Enables/disables the internal Bluetooth hardware and W-WAN card if left in control of the firmware. Does not affect the WLAN card. Should be used to turn on/off all radios (Bluetooth+W-WAN+WLAN), really.
0x1006	0x05	FN+F6	•
0x1007	0x06	FN+F7	Video output cycle. Do you feel lucky today?
0x1008	0x07	FN+F8	IBM: toggle screen expand Lenovo: configure UltraNav, or toggle screen expand
0x1009	0x08	FN+F9	•

The ThinkPad firmware does not allow one to differentiate when most hot keys are pressed or released (either that, or we don't know how to, yet). For these keys, the driver generates a set of events for a key press and immediately issues the same set of events for a key release. It is unknown by the driver if the ThinkPad firmware triggered these events on hot key press or release, but the firmware will do it for either one, not both.

If a key is mapped to KEY_RESERVED, it generates no input events at all. If a key is mapped to KEY_UNKNOWN, it generates an input event that includes an scan code. If a key is mapped to anything else, it will generate input device EV_KEY events.

In addition to the EV_KEY events, thinkpad-acpi may also issue EV_SW events for switches:

SW_RFKILL_ALL	T60 and later hardware rfkill rocker switch
SW_TABLET_MODE	Tablet ThinkPads HKEY events 0x5009 and 0x500A

53.7.7 Non hotkey ACPI HKEY event map

Events that are never propagated by the driver:

0x2304	System is waking up from suspend to undock
0x2305	System is waking up from suspend to eject bay
0x2404	System is waking up from hibernation to undock
0x2405	System is waking up from hibernation to eject bay
0x5001	Lid closed
0x5002	Lid opened
0x5009	Tablet swivel: switched to tablet mode
0x500A	Tablet swivel: switched to normal mode
0x5010	Brightness level changed/control event
0x6000	KEYBOARD: Numlock key pressed
0x6005	KEYBOARD: Fn key pressed (TO BE VERIFIED)
0x7000	Radio Switch may have changed state

Events that are propagated by the driver to userspace:

0x2313	ALARM: System is waking up from suspend because the battery is nearly empty
0x2413	ALARM: System is waking up from hibernation because the battery is nearly empty
0x3003	Bay ejection (see 0x2x05) complete, can sleep again
0x3006	Bay hotplug request (hint to power up SATA link when the optical drive tray is ejected)
0x4003	Undocked (see 0x2x04), can sleep again
0x4010	Docked into hotplug port replicator (non-ACPI dock)
0x4011	Undocked from hotplug port replicator (non-ACPI dock)
0x500B	Tablet pen inserted into its storage bay
0x500C	Tablet pen removed from its storage bay
0x6011	ALARM: battery is too hot
0x6012	ALARM: battery is extremely hot
0x6021	ALARM: a sensor is too hot
0x6022	ALARM: a sensor is extremely hot
0x6030	System thermal table changed
0x6032	Thermal Control command set completion (DYTC, Windows)
0x6040	Nvidia Optimus/AC adapter related (TO BE VERIFIED)
0x60C0	X1 Yoga 2016, Tablet mode status changed
0x60F0	Thermal Transformation changed (GMTS, Windows)

Battery nearly empty alarms are a last resort attempt to get the operating system to hibernate or shutdown cleanly (0x2313), or shutdown cleanly (0x2413) before power is lost. They must be acted upon, as the wake up caused by the firmware will have negated most safety nets...

When any of the ”too hot” alarms happen, according to Lenovo the user should suspend or hibernate the laptop (and in the case of battery alarms, unplug the AC adapter) to let it cool down. These alarms do signal that something is wrong, they should never happen on normal operating conditions.

The ”extremely hot” alarms are emergencies. According to Lenovo, the operating system is to force either an immediate suspend or hibernate cycle, or a system shutdown. Obviously, something is very wrong if this happens.

Brightness hotkey notes

Don't mess with the brightness hotkeys in a Thinkpad. If you want notifications for OSD, use the sysfs backlight class event support.

The driver will issue KEY_BRIGHTNESS_UP and KEY_BRIGHTNESS_DOWN events automatically for the cases were userspace has to do something to implement brightness changes. When you override these events, you will either fail to handle properly the ThinkPads that require explicit action to change backlight brightness, or the ThinkPads that require that no action be taken to work properly.

53.7.8 Bluetooth

procfs: /proc/acpi/ibm/bluetooth

sysfs device attribute: bluetooth_enable (deprecated)

sysfs rfkill class: switch "tpacpi_bluetooth_sw"

This feature shows the presence and current state of a ThinkPad Bluetooth device in the internal ThinkPad CDC slot.

If the ThinkPad supports it, the Bluetooth state is stored in NVRAM, so it is kept across reboots and power-off.

Procfs notes

If Bluetooth is installed, the following commands can be used:

```
echo enable > /proc/acpi/ibm/bluetooth
echo disable > /proc/acpi/ibm/bluetooth
```

Sysfs notes

If the Bluetooth CDC card is installed, it can be enabled / disabled through the "bluetooth_enable" thinkpad-acpi device attribute, and its current status can also be queried.

enable:

- 0: disables Bluetooth / Bluetooth is disabled
- 1: enables Bluetooth / Bluetooth is enabled.

Note: this interface has been superseded by the generic rfkill class. It has been deprecated, and it will be removed in year 2010.

rfkill controller switch "tpacpi_bluetooth_sw": refer to Documentation/driver-api/rfkill.rst for details.

53.7.9 Video output control -- /proc/acpi/ibm/video

This feature allows control over the devices used for video output - LCD, CRT or DVI (if available). The following commands are available:

```
echo lcd_enable > /proc/acpi/ibm/video
echo lcd_disable > /proc/acpi/ibm/video
echo crt_enable > /proc/acpi/ibm/video
echo crt_disable > /proc/acpi/ibm/video
echo dvi_enable > /proc/acpi/ibm/video
echo dvi_disable > /proc/acpi/ibm/video
echo auto_enable > /proc/acpi/ibm/video
echo auto_disable > /proc/acpi/ibm/video
echo expand_toggle > /proc/acpi/ibm/video
echo video_switch > /proc/acpi/ibm/video
```

NOTE:
Access to this feature is restricted to processes owning the CAP_SYS_ADMIN capability for safety reasons, as it can interact badly enough with some versions of X.org to crash it.

Each video output device can be enabled or disabled individually. Reading /proc/acpi/ibm/video shows the status of each device.

Automatic video switching can be enabled or disabled. When automatic video switching is enabled, certain events (e.g. opening the lid, docking or undocking) cause the video output device to change automatically. While this can be useful, it also causes flickering and, on the X40, video corruption. By disabling automatic switching, the flickering or video corruption can be avoided.

The video_switch command cycles through the available video outputs (it simulates the behavior of Fn-F7).

Video expansion can be toggled through this feature. This controls whether the display is expanded to fill the entire LCD screen when a mode with less than full resolution is used. Note that the current video expansion status cannot be determined through this feature.

Note that on many models (particularly those using Radeon graphics chips) the X driver configures the video card in a way which prevents Fn-F7 from working. This also disables the video output switching features of this driver, as it uses the same ACPI methods as Fn-F7. Video switching on the console should still work.

UPDATE: refer to https://bugs.freedesktop.org/show_bug.cgi?id=2000

53.7.10 ThinkLight control

procfs: /proc/acpi/ibm/light

sysfs attributes: as per LED class, for the "tpacpi::thinklight" LED

procfs notes

The ThinkLight status can be read and set through the procfs interface. A few models which do not make the status available will show the ThinkLight status as "unknown". The available commands are:

```
echo on  > /proc/acpi/ibm/light
echo off > /proc/acpi/ibm/light
```

sysfs notes

The ThinkLight sysfs interface is documented by the LED class documentation, in Documentation/leds/leds-class.rst. The ThinkLight LED name is ”tpacpi::thinklight”.

Due to limitations in the sysfs LED class, if the status of the ThinkLight cannot be read or if it is unknown, thinkpad-acpi will report it as ”off”. It is impossible to know if the status returned through sysfs is valid.

53.7.11 CMOS/UCMS control

procfs: /proc/acpi/ibm/cmos

sysfs device attribute: cmos_command

This feature is mostly used internally by the ACPI firmware to keep the legacy CMOS NVRAM bits in sync with the current machine state, and to record this state so that the ThinkPad will retain such settings across reboots.

Some of these commands actually perform actions in some ThinkPad models, but this is expected to disappear more and more in newer models. As an example, in a T43 and in a X40, commands 12 and 13 still control the ThinkLight state for real, but commands 0 to 2 don't control the mixer anymore (they have been phased out) and just update the NVRAM.

The range of valid cmos command numbers is 0 to 21, but not all have an effect and the behavior varies from model to model. Here is the behavior on the X40 (tpb is the ThinkPad Buttons utility):

- 0 - Related to ”Volume down” key press
- 1 - Related to ”Volume up” key press
- 2 - Related to ”Mute on” key press
- 3 - Related to ”Access IBM” key press
- 4 - Related to ”LCD brightness up” key press
- 5 - Related to ”LCD brightness down” key press
- 11 - Related to ”toggle screen expansion” key press/function
- 12 - Related to ”ThinkLight on”
- 13 - Related to ”ThinkLight off”
- 14 - Related to ”ThinkLight” key press (toggle ThinkLight)

The cmos command interface is prone to firmware split-brain problems, as in newer ThinkPads it is just a compatibility layer. Do not use it, it is exported just as a debug tool.

53.7.12 LED control

procfs: /proc/acpi/ibm/led sysfs attributes: as per LED class, see below for names

Some of the LED indicators can be controlled through this feature. On some older ThinkPad models, it is possible to query the status of the LED indicators as well. Newer ThinkPads cannot query the real status of the LED indicators.

Because misuse of the LEDs could induce an unaware user to perform dangerous actions (like undocking or ejecting a bay device while the buses are still active), or mask an important alarm (such as a nearly empty battery, or a broken battery), access to most LEDs is restricted.

Unrestricted access to all LEDs requires that thinkpad-acpi be compiled with the CONFIG_THINKPAD_ACPI_UNSAFE_LEDS option enabled. Distributions must never enable this option. Individual users that are aware of the consequences are welcome to enabling it.

Audio mute and microphone mute LEDs are supported, but currently not visible to userspace. They are used by the snd-hda-intel audio driver.

procfs notes

The available commands are:

```
echo '<LED number> on' >/proc/acpi/ibm/led
echo '<LED number> off' >/proc/acpi/ibm/led
echo '<LED number> blink' >/proc/acpi/ibm/led
```

The <LED number> range is 0 to 15. The set of LEDs that can be controlled varies from model to model. Here is the common ThinkPad mapping:

- 0 - power
- 1 - battery (orange)
- 2 - battery (green)
- 3 - UltraBase/dock
- 4 - UltraBay
- 5 - UltraBase battery slot
- 6 - (unknown)
- 7 - standby
- 8 - dock status 1
- 9 - dock status 2
- 10, 11 - (unknown)
- 12 - thinkvantage
- 13, 14, 15 - (unknown)

All of the above can be turned on and off and can be made to blink.

sysfs notes

The ThinkPad LED sysfs interface is described in detail by the LED class documentation, in Documentation/leds/leds-class.rst.

The LEDs are named (in LED ID order, from 0 to 12): "tpacpi::power", "tpacpi:orange:batt", "tpacpi:green:batt", "tpacpi::dock_active", "tpacpi::bay_active", "tpacpi::dock_batt", "tpacpi::unknown_led", "tpacpi::standby", "tpacpi::dock_status1", "tpacpi::dock_status2", "tpacpi::unknown_led2", "tpacpi::unknown_led3", "tpacpi::thinkvantage".

Due to limitations in the sysfs LED class, if the status of the LED indicators cannot be read due to an error, thinkpad-acpi will report it as a brightness of zero (same as LED off).

If the thinkpad firmware doesn't support reading the current status, trying to read the current LED brightness will just return whatever brightness was last written to that attribute.

These LEDs can blink using hardware acceleration. To request that a ThinkPad indicator LED should blink in hardware accelerated mode, use the "timer" trigger, and leave the delay_on and delay_off parameters set to zero (to request hardware acceleration autodetection).

LEDs that are known not to exist in a given ThinkPad model are not made available through the sysfs interface. If you have a dock and you notice there are LEDs listed for your ThinkPad that do not exist (and are not in the dock), or if you notice that there are missing LEDs, a report to ibm-acpi-devel@lists.sourceforge.net is appreciated.

53.7.13 ACPI sounds -- /proc/acpi/ibm/beep

The BEEP method is used internally by the ACPI firmware to provide audible alerts in various situations. This feature allows the same sounds to be triggered manually.

The commands are non-negative integer numbers:

```
echo <number> >/proc/acpi/ibm/beep
```

The valid <number> range is 0 to 17. Not all numbers trigger sounds and the sounds vary from model to model. Here is the behavior on the X40:

- 0 - stop a sound in progress (but use 17 to stop 16)
- 2 - two beeps, pause, third beep ("low battery")
- 3 - single beep
- 4 - high, followed by low-pitched beep ("unable")
- 5 - single beep
- 6 - very high, followed by high-pitched beep ("AC/DC")
- 7 - high-pitched beep
- 9 - three short beeps
- 10 - very long beep
- 12 - low-pitched beep
- 15 - three high-pitched beeps repeating constantly, stop with 0
- 16 - one medium-pitched beep repeating constantly, stop with 17

- 17 - stop 16

53.7.14 Temperature sensors

procfs: /proc/acpi/ibm/thermal

sysfs device attributes: (hwmon "thinkpad") temp*_input

Most ThinkPads include six or more separate temperature sensors but only expose the CPU temperature through the standard ACPI methods. This feature shows readings from up to eight different sensors on older ThinkPads, and up to sixteen different sensors on newer ThinkPads.

For example, on the X40, a typical output may be:

temperatures:
42 42 45 41 36 -128 33 -128

On the T43/p, a typical output may be:

temperatures:
48 48 36 52 38 -128 31 -128 48 52 48 -128 -128 -128 -128 -128

The mapping of thermal sensors to physical locations varies depending on system-board model (and thus, on ThinkPad model).

https://thinkwiki.org/wiki/Thermal_Sensors is a public wiki page that tries to track down these locations for various models.

Most (newer?) models seem to follow this pattern:

- 1: CPU
- 2: (depends on model)
- 3: (depends on model)
- 4: GPU
- 5: Main battery: main sensor
- 6: Bay battery: main sensor
- 7: Main battery: secondary sensor
- 8: Bay battery: secondary sensor
- 9-15: (depends on model)

For the R51 (source: Thomas Gruber):

- 2: Mini-PCI
- 3: Internal HDD

For the T43, T43/p (source: Shmidoax/Thinkwiki.org) https://thinkwiki.org/wiki/Thermal_Sensors#ThinkPad_T43.2C_T43p

- 2: System board, left side (near PCMCIA slot), reported as HDAPS temp
- 3: PCMCIA slot
- 9: MCH (northbridge) to DRAM Bus

- **10: Clock-generator, mini-pci card and ICH (southbridge), under Mini-PCI** card, under touchpad
- 11: Power regulator, underside of system board, below F2 key

The A31 has a very atypical layout for the thermal sensors (source: Milos Popovic, https://thinkwiki.org/wiki/Thermal_Sensors#ThinkPad_A31)

- 1: CPU
- 2: Main Battery: main sensor
- 3: Power Converter
- 4: Bay Battery: main sensor
- 5: MCH (northbridge)
- 6: PCMCIA/ambient
- 7: Main Battery: secondary sensor
- 8: Bay Battery: secondary sensor

Procfs notes

Readings from sensors that are not available return -128. No commands can be written to this file.

Sysfs notes

Sensors that are not available return the ENXIO error. This status may change at runtime, as there are hotplug thermal sensors, like those inside the batteries and docks.

thinkpad-acpi thermal sensors are reported through the hwmon subsystem, and follow all of the hwmon guidelines at Documentation/hwmon.

53.7.15 EXPERIMENTAL: Embedded controller register dump

This feature is not included in the thinkpad driver anymore. Instead the EC can be accessed through /sys/kernel/debug/ec with a userspace tool which can be found here: ftp://ftp.suse.com/pub/people/trenn/sources/ec

Use it to determine the register holding the fan speed on some models. To do that, do the following:

- make sure the battery is fully charged
- make sure the fan is running
- use above mentioned tool to read out the EC

Often fan and temperature values vary between readings. Since temperatures don't change vary fast, you can take several quick dumps to eliminate them.

You can use a similar method to figure out the meaning of other embedded controller registers - e.g. make sure nothing else changes except the charging or discharging battery to determine

which registers contain the current battery capacity, etc. If you experiment with this, do send me your results (including some complete dumps with a description of the conditions when they were taken.)

53.7.16 LCD brightness control

procfs: /proc/acpi/ibm/brightness

sysfs backlight device ”thinkpad_screen”

This feature allows software control of the LCD brightness on ThinkPad models which don't have a hardware brightness slider.

It has some limitations: the LCD backlight cannot be actually turned on or off by this interface, it just controls the backlight brightness level.

On IBM (and some of the earlier Lenovo) ThinkPads, the backlight control has eight brightness levels, ranging from 0 to 7. Some of the levels may not be distinct. Later Lenovo models that implement the ACPI display backlight brightness control methods have 16 levels, ranging from 0 to 15.

For IBM ThinkPads, there are two interfaces to the firmware for direct brightness control, EC and UCMS (or CMOS). To select which one should be used, use the brightness_mode module parameter: brightness_mode=1 selects EC mode, brightness_mode=2 selects UCMS mode, brightness_mode=3 selects EC mode with NVRAM backing (so that brightness changes are remembered across shutdown/reboot).

The driver tries to select which interface to use from a table of defaults for each ThinkPad model. If it makes a wrong choice, please report this as a bug, so that we can fix it.

Lenovo ThinkPads only support brightness_mode=2 (UCMS).

When display backlight brightness controls are available through the standard ACPI interface, it is best to use it instead of this direct ThinkPad-specific interface. The driver will disable its native backlight brightness control interface if it detects that the standard ACPI interface is available in the ThinkPad.

If you want to use the thinkpad-acpi backlight brightness control instead of the generic ACPI video backlight brightness control for some reason, you should use the acpi_backlight=vendor kernel parameter.

The brightness_enable module parameter can be used to control whether the LCD brightness control feature will be enabled when available. brightness_enable=0 forces it to be disabled. brightness_enable=1 forces it to be enabled when available, even if the standard ACPI interface is also available.

Procfs notes

The available commands are:

```
echo up   >/proc/acpi/ibm/brightness
echo down >/proc/acpi/ibm/brightness
echo 'level <level>' >/proc/acpi/ibm/brightness
```

Sysfs notes

The interface is implemented through the backlight sysfs class, which is poorly documented at this time.

Locate the thinkpad_screen device under /sys/class/backlight, and inside it there will be the following attributes:

max_brightness:
Reads the maximum brightness the hardware can be set to. The minimum is always zero.

actual_brightness:
Reads what brightness the screen is set to at this instant.

brightness:
Writes request the driver to change brightness to the given value. Reads will tell you what brightness the driver is trying to set the display to when "power" is set to zero and the display has not been dimmed by a kernel power management event.

power:
power management mode, where 0 is "display on", and 1 to 3 will dim the display backlight to brightness level 0 because thinkpad-acpi cannot really turn the backlight off. Kernel power management events can temporarily increase the current power management level, i.e. they can dim the display.

WARNING:

Whatever you do, do NOT ever call thinkpad-acpi backlight-level change interface and the ACPI-based backlight level change interface (available on newer BIOSes, and driven by the Linux ACPI video driver) at the same time. The two will interact in bad ways, do funny things, and maybe reduce the life of the backlight lamps by needlessly kicking its level up and down at every change.

53.7.17 Volume control (Console Audio control)

procfs: /proc/acpi/ibm/volume

ALSA: "ThinkPad Console Audio Control", default ID: "ThinkPadEC"

NOTE: by default, the volume control interface operates in read-only mode, as it is supposed to be used for on-screen-display purposes. The read/write mode can be enabled through the use of the "volume_control=1" module parameter.

NOTE: distros are urged to not enable volume_control by default, this should be done by the local admin only. The ThinkPad UI is for the console audio control to be done through the

volume keys only, and for the desktop environment to just provide on-screen-display feedback. Software volume control should be done only in the main AC97/HDA mixer.

About the ThinkPad Console Audio control

ThinkPads have a built-in amplifier and muting circuit that drives the console headphone and speakers. This circuit is after the main AC97 or HDA mixer in the audio path, and under exclusive control of the firmware.

ThinkPads have three special hotkeys to interact with the console audio control: volume up, volume down and mute.

It is worth noting that the normal way the mute function works (on ThinkPads that do not have a ”mute LED”) is:

1. Press mute to mute. It will *always* mute, you can press it as many times as you want, and the sound will remain mute.
2. Press either volume key to unmute the ThinkPad (it will _not_ change the volume, it will just unmute).

This is a very superior design when compared to the cheap software-only mute-toggle solution found on normal consumer laptops: you can be absolutely sure the ThinkPad will not make noise if you press the mute button, no matter the previous state.

The IBM ThinkPads, and the earlier Lenovo ThinkPads have variable-gain amplifiers driving the speakers and headphone output, and the firmware also handles volume control for the headphone and speakers on these ThinkPads without any help from the operating system (this volume control stage exists after the main AC97 or HDA mixer in the audio path).

The newer Lenovo models only have firmware mute control, and depend on the main HDA mixer to do volume control (which is done by the operating system). In this case, the volume keys are filtered out for unmute key press (there are some firmware bugs in this area) and delivered as normal key presses to the operating system (thinkpad-acpi is not involved).

The ThinkPad-ACPI volume control

The preferred way to interact with the Console Audio control is the ALSA interface.

The legacy procfs interface allows one to read the current state, and if volume control is enabled, accepts the following commands:

```
echo up   >/proc/acpi/ibm/volume
echo down >/proc/acpi/ibm/volume
echo mute >/proc/acpi/ibm/volume
echo unmute >/proc/acpi/ibm/volume
echo 'level <level>' >/proc/acpi/ibm/volume
```

The <level> number range is 0 to 14 although not all of them may be distinct. To unmute the volume after the mute command, use either the up or down command (the level command will not unmute the volume), or the unmute command.

You can use the volume_capabilities parameter to tell the driver whether your thinkpad has volume control or mute-only control: volume_capabilities=1 for mixers with mute and volume control, volume_capabilities=2 for mixers with only mute control.

If the driver misdetects the capabilities for your ThinkPad model, please report this to ibm-acpi-devel@lists.sourceforge.net, so that we can update the driver.

There are two strategies for volume control. To select which one should be used, use the volume_mode module parameter: volume_mode=1 selects EC mode, and volume_mode=3 selects EC mode with NVRAM backing (so that volume/mute changes are remembered across shutdown/reboot).

The driver will operate in volume_mode=3 by default. If that does not work well on your ThinkPad model, please report this to ibm-acpi-devel@lists.sourceforge.net.

The driver supports the standard ALSA module parameters. If the ALSA mixer is disabled, the driver will disable all volume functionality.

53.7.18 Fan control and monitoring: fan speed, fan enable/disable

procfs: /proc/acpi/ibm/fan

sysfs device attributes: (hwmon "thinkpad") fan1_input, pwm1, pwm1_enable, fan2_input

sysfs hwmon driver attributes: fan_watchdog

NOTE NOTE NOTE:
fan control operations are disabled by default for safety reasons. To enable them, the module parameter "fan_control=1" must be given to thinkpad-acpi.

This feature attempts to show the current fan speed, control mode and other fan data that might be available. The speed is read directly from the hardware registers of the embedded controller. This is known to work on later R, T, X and Z series ThinkPads but may show a bogus value on other models.

Some Lenovo ThinkPads support a secondary fan. This fan cannot be controlled separately, it shares the main fan control.

Fan levels

Most ThinkPad fans work in "levels" at the firmware interface. Level 0 stops the fan. The higher the level, the higher the fan speed, although adjacent levels often map to the same fan speed. 7 is the highest level, where the fan reaches the maximum recommended speed.

Level "auto" means the EC changes the fan level according to some internal algorithm, usually based on readings from the thermal sensors.

There is also a "full-speed" level, also known as "disengaged" level. In this level, the EC disables the speed-locked closed-loop fan control, and drives the fan as fast as it can go, which might exceed hardware limits, so use this level with caution.

The fan usually ramps up or down slowly from one speed to another, and it is normal for the EC to take several seconds to react to fan commands. The full-speed level may take up to two minutes to ramp up to maximum speed, and in some ThinkPads, the tachometer readings go stale while the EC is transitioning to the full-speed level.

WARNING WARNING WARNING: do not leave the fan disabled unless you are monitoring all of the temperature sensor readings and you are ready to enable it if necessary to avoid overheating.

An enabled fan in level ”auto” may stop spinning if the EC decides the ThinkPad is cool enough and doesn't need the extra airflow. This is normal, and the EC will spin the fan up if the various thermal readings rise too much.

On the X40, this seems to depend on the CPU and HDD temperatures. Specifically, the fan is turned on when either the CPU temperature climbs to 56 degrees or the HDD temperature climbs to 46 degrees. The fan is turned off when the CPU temperature drops to 49 degrees and the HDD temperature drops to 41 degrees. These thresholds cannot currently be controlled.

The ThinkPad's ACPI DSDT code will reprogram the fan on its own when certain conditions are met. It will override any fan programming done through thinkpad-acpi.

The thinkpad-acpi kernel driver can be programmed to revert the fan level to a safe setting if userspace does not issue one of the procfs fan commands: ”enable”, ”disable”, ”level” or ”watchdog”, or if there are no writes to pwm1_enable (or to pwm1 *if and only if* pwm1_enable is set to 1, manual mode) within a configurable amount of time of up to 120 seconds. This functionality is called fan safety watchdog.

Note that the watchdog timer stops after it enables the fan. It will be rearmed again automatically (using the same interval) when one of the above mentioned fan commands is received. The fan watchdog is, therefore, not suitable to protect against fan mode changes made through means other than the ”enable”, ”disable”, and ”level” procfs fan commands, or the hwmon fan control sysfs interface.

Procfs notes

The fan may be enabled or disabled with the following commands:

```
echo enable  >/proc/acpi/ibm/fan
echo disable >/proc/acpi/ibm/fan
```

Placing a fan on level 0 is the same as disabling it. Enabling a fan will try to place it in a safe level if it is too slow or disabled.

The fan level can be controlled with the command:

```
echo 'level <level>' > /proc/acpi/ibm/fan
```

Where <level> is an integer from 0 to 7, or one of the words ”auto” or ”full-speed” (without the quotes). Not all ThinkPads support the ”auto” and ”full-speed” levels. The driver accepts ”disengaged” as an alias for ”full-speed”, and reports it as ”disengaged” for backwards compatibility.

On the X31 and X40 (and ONLY on those models), the fan speed can be controlled to a certain degree. Once the fan is running, it can be forced to run faster or slower with the following command:

```
echo 'speed <speed>' > /proc/acpi/ibm/fan
```

The sustainable range of fan speeds on the X40 appears to be from about 3700 to about 7350. Values outside this range either do not have any effect or the fan speed eventually settles somewhere in that range. The fan cannot be stopped or started with this command. This functionality is incomplete, and not available through the sysfs interface.

To program the safety watchdog, use the ”watchdog” command:

```
echo 'watchdog <interval in seconds>' > /proc/acpi/ibm/fan
```

If you want to disable the watchdog, use 0 as the interval.

Sysfs notes

The sysfs interface follows the hwmon subsystem guidelines for the most part, and the exception is the fan safety watchdog.

Writes to any of the sysfs attributes may return the EINVAL error if that operation is not supported in a given ThinkPad or if the parameter is out-of-bounds, and EPERM if it is forbidden. They may also return EINTR (interrupted system call), and EIO (I/O error while trying to talk to the firmware).

Features not yet implemented by the driver return ENOSYS.

hwmon device attribute pwm1_enable:

- 0: PWM offline (fan is set to full-speed mode)
- 1: Manual PWM control (use pwm1 to set fan level)
- 2: Hardware PWM control (EC "auto" mode)
- 3: reserved (Software PWM control, not implemented yet)

Modes 0 and 2 are not supported by all ThinkPads, and the driver is not always able to detect this. If it does know a mode is unsupported, it will return -EINVAL.

hwmon device attribute pwm1:
Fan level, scaled from the firmware values of 0-7 to the hwmon scale of 0-255. 0 means fan stopped, 255 means highest normal speed (level 7).

This attribute only commands the fan if pmw1_enable is set to 1 (manual PWM control).

hwmon device attribute fan1_input:
Fan tachometer reading, in RPM. May go stale on certain ThinkPads while the EC transitions the PWM to offline mode, which can take up to two minutes. May return rubbish on older ThinkPads.

hwmon device attribute fan2_input:
Fan tachometer reading, in RPM, for the secondary fan. Available only on some ThinkPads. If the secondary fan is not installed, will always read 0.

hwmon driver attribute fan_watchdog:
Fan safety watchdog timer interval, in seconds. Minimum is 1 second, maximum is 120 seconds. 0 disables the watchdog.

To stop the fan: set pwm1 to zero, and pwm1_enable to 1.

To start the fan in a safe mode: set pwm1_enable to 2. If that fails with EINVAL, try to set pwm1_enable to 1 and pwm1 to at least 128 (255 would be the safest choice, though).

53.7.19 WAN

procfs: /proc/acpi/ibm/wan

sysfs device attribute: wwan_enable (deprecated)

sysfs rfkill class: switch "tpacpi_wwan_sw"

This feature shows the presence and current state of the built-in Wireless WAN device.

If the ThinkPad supports it, the WWAN state is stored in NVRAM, so it is kept across reboots and power-off.

It was tested on a Lenovo ThinkPad X60. It should probably work on other ThinkPad models which come with this module installed.

Procfs notes

If the W-WAN card is installed, the following commands can be used:

```
echo enable > /proc/acpi/ibm/wan
echo disable > /proc/acpi/ibm/wan
```

Sysfs notes

If the W-WAN card is installed, it can be enabled / disabled through the "wwan_enable" thinkpad-acpi device attribute, and its current status can also be queried.

enable:

- 0: disables WWAN card / WWAN card is disabled
- 1: enables WWAN card / WWAN card is enabled.

Note: this interface has been superseded by the generic rfkill class. It has been deprecated, and it will be removed in year 2010.

rfkill controller switch "tpacpi_wwan_sw": refer to Documentation/driver-api/rfkill.rst for details.

53.7.20 LCD Shadow control

procfs: /proc/acpi/ibm/lcdshadow

Some newer T480s and T490s ThinkPads provide a feature called PrivacyGuard. By turning this feature on, the usable vertical and horizontal viewing angles of the LCD can be limited (as if some privacy screen was applied manually in front of the display).

procfs notes

The available commands are:

```
echo '0' >/proc/acpi/ibm/lcdshadow
echo '1' >/proc/acpi/ibm/lcdshadow
```

The first command ensures the best viewing angle and the latter one turns on the feature, restricting the viewing angles.

53.7.21 DYTC Lapmode sensor

sysfs: dytc_lapmode

Newer thinkpads and mobile workstations have the ability to determine if the device is in deskmode or lapmode. This feature is used by user space to decide if WWAN transmission can be increased to maximum power and is also useful for understanding the different thermal modes available as they differ between desk and lap mode.

The property is read-only. If the platform doesn't have support the sysfs class is not created.

53.7.22 EXPERIMENTAL: UWB

This feature is considered EXPERIMENTAL because it has not been extensively tested and validated in various ThinkPad models yet. The feature may not work as expected. USE WITH CAUTION! To use this feature, you need to supply the experimental=1 parameter when loading the module.

sysfs rfkill class: switch ”tpacpi_uwb_sw”

This feature exports an rfkill controller for the UWB device, if one is present and enabled in the BIOS.

Sysfs notes

> rfkill controller switch ”tpacpi_uwb_sw”: refer to Documentation/driver-api/rfkill.rst for details.

53.7.23 Setting keyboard language

sysfs: keyboard_lang

This feature is used to set keyboard language to ECFW using ASL interface. Fewer thinkpads models like T580 , T590 , T15 Gen 1 etc.. has ”=”, ”(', ”)” numeric keys, which are not displaying correctly, when keyboard language is other than ”english”. This is because the default keyboard language in ECFW is set as ”english”. Hence using this sysfs, user can set the correct keyboard language to ECFW and then these key's will work correctly.

Example of command to set keyboard language is mentioned below:

```
echo jp > /sys/devices/platform/thinkpad_acpi/keyboard_lang
```

Text corresponding to keyboard layout to be set in sysfs are: be(Belgian), cz(Czech), da(Danish), de(German), en(English), es(Spain), et(Estonian), fr(French), fr-ch(French(Switzerland)), hu(Hungarian), it(Italy), jp (Japan), nl(Dutch), nn(Norway), pl(Polish), pt(portuguese), sl(Slovenian), sv(Sweden), tr(Turkey)

53.7.24 WWAN Antenna type

sysfs: wwan_antenna_type

On some newer Thinkpads we need to set SAR value based on the antenna type. This interface will be used by userspace to get the antenna type and set the corresponding SAR value, as is required for FCC certification.

The available commands are:

```
cat /sys/devices/platform/thinkpad_acpi/wwan_antenna_type
```

Currently 2 antenna types are supported as mentioned below: - type a - type b

The property is read-only. If the platform doesn't have support the sysfs class is not created.

53.7.25 Auxmac

sysfs: auxmac

Some newer Thinkpads have a feature called MAC Address Pass-through. This feature is implemented by the system firmware to provide a system unique MAC, that can override a dock or USB ethernet dongle MAC, when connected to a network. This property enables user-space to easily determine the MAC address if the feature is enabled.

The values of this auxiliary MAC are:

cat /sys/devices/platform/thinkpad_acpi/auxmac

If the feature is disabled, the value will be 'disabled'.

This property is read-only.

53.7.26 Adaptive keyboard

sysfs device attribute: adaptive_kbd_mode

This sysfs attribute controls the keyboard ”face” that will be shown on the Lenovo X1 Carbon 2nd gen (2014)'s adaptive keyboard. The value can be read and set.

- 0 = Home mode
- 1 = Web-browser mode
- 2 = Web-conference mode
- 3 = Function mode
- 4 = Layflat mode

For more details about which buttons will appear depending on the mode, please review the laptop's user guide: https://download.lenovo.com/ibmdl/pub/pc/pccbbs/mobiles_pdf/x1carbon_2_ug_en.pdf

53.7.27 Battery charge control

sysfs attributes: /sys/class/power_supply/BAT*/charge_control_{start,end}_threshold

These two attributes are created for those batteries that are supported by the driver. They enable the user to control the battery charge thresholds of the given battery. Both values may be read and set. *charge_control_start_threshold* accepts an integer between 0 and 99 (inclusive); this value represents a battery percentage level, below which charging will begin. *charge_control_end_threshold* accepts an integer between 1 and 100 (inclusive); this value represents a battery percentage level, above which charging will stop.

The exact semantics of the attributes may be found in Documentation/ABI/testing/sysfs-class-power.

53.7.28 Multiple Commands, Module Parameters

Multiple commands can be written to the proc files in one shot by separating them with commas, for example:

```
echo enable,0xffff > /proc/acpi/ibm/hotkey
echo lcd_disable,crt_enable > /proc/acpi/ibm/video
```

Commands can also be specified when loading the thinkpad-acpi module, for example:

```
modprobe thinkpad_acpi hotkey=enable,0xffff video=auto_disable
```

53.7.29 Enabling debugging output

The module takes a debug parameter which can be used to selectively enable various classes of debugging output, for example:

```
modprobe thinkpad_acpi debug=0xffff
```

will enable all debugging output classes. It takes a bitmask, so to enable more than one output class, just add their values.

Debug bitmask	Description
0x8000	Disclose PID of userspace programs accessing some functions of the driver
0x0001	Initialization and probing
0x0002	Removal
0x0004	RF Transmitter control (RFKILL) (bluetooth, WWAN, UWB...)
0x0008	HKEY event interface, hotkeys
0x0010	Fan control
0x0020	Backlight brightness
0x0040	Audio mixer/volume control

There is also a kernel build option to enable more debugging information, which may be necessary to debug driver problems.

The level of debugging information output by the driver can be changed at runtime through sysfs, using the driver attribute debug_level. The attribute takes the same bitmask as the debug module parameter above.

53.7.30 Force loading of module

If thinkpad-acpi refuses to detect your ThinkPad, you can try to specify the module parameter force_load=1. Regardless of whether this works or not, please contact ibm-acpi-devel@lists.sourceforge.net with a report.

Sysfs interface changelog

0x000100:	Initial sysfs support, as a single platform driver and device.
0x000200:	Hot key support for 32 hot keys, and radio slider switch support.
0x010000:	Hot keys are now handled by default over the input layer, the radio switch generates input event EV_RADIO, and the driver enables hot key handling by default in the firmware.
0x020000:	ABI fix: added a separate hwmon platform device and driver, which must be located by name (thinkpad) and the hwmon class for libsensors4 (lm-sensors 3) compatibility. Moved all hwmon attributes to this new platform device.
0x020100:	Marker for thinkpad-acpi with hot key NVRAM polling support. If you must, use it to know you should not start a userspace NVRAM poller (allows to detect when NVRAM is compiled out by the user because it is unneeded/undesired in the first place).
0x020101:	Marker for thinkpad-acpi with hot key NVRAM polling and proper hotkey_mask semantics (version 8 of the NVRAM polling patch). Some development snapshots of 0.18 had an earlier version that did strange things to hotkey_mask.
0x020200:	Add poll()/select() support to the following attributes: hotkey_radio_sw, wakeup_hotunplug_complete, wakeup_reason
0x020300:	hotkey enable/disable support removed, attributes hotkey_bios_enabled and hotkey_enable deprecated and marked for removal.
0x020400:	Marker for 16 LEDs support. Also, LEDs that are known to not exist in a given model are not registered with the LED sysfs class anymore.
0x020500:	Updated hotkey driver, hotkey_mask is always available and it is always able to disable hot keys. Very old thinkpads are properly supported. hotkey_bios_mask is deprecated and marked for removal.
0x020600:	Marker for backlight change event support.
0x020700:	Support for mute-only mixers. Volume control in read-only mode by default. Marker for ALSA mixer support.
0x030000:	Thermal and fan sysfs attributes were moved to the hwmon device instead of being attached to the backing platform device.

53.8 Toshiba HDD Active Protection Sensor

Kernel driver: toshiba_haps

Author: Azael Avalos <coproscefalo@gmail.com>

53.8.1 1. Description

This driver provides support for the accelerometer found in various Toshiba laptops, being called "Toshiba HDD Protection - Shock Sensor" officially, and detects laptops automatically with this device. On Windows, Toshiba provided software monitors this device and provides automatic HDD protection (head unload) on sudden moves or harsh vibrations, however, this driver only provides a notification via a sysfs file to let userspace tools or daemons act accordingly, as well as providing a sysfs file to set the desired protection level or sensor sensibility.

53.8.2 2. Interface

This device comes with 3 methods:

_STA	Checks existence of the device, returning Zero if the device does not exists or is not supported.
PTLV	Sets the desired protection level.
RSSS	Shuts down the HDD protection interface for a few seconds, then restores normal operation.

Note:
The presence of Solid State Drives (SSD) can make this driver to fail loading, given the fact that such drives have no movable parts, and thus, not requiring any "protection" as well as failing during the evaluation of the _STA method found under this device.

53.8.3 3. Accelerometer axes

This device does not report any axes, however, to query the sensor position a couple HCI (Hardware Configuration Interface) calls (0x6D and 0xA6) are provided to query such information, handled by the kernel module toshiba_acpi since kernel version 3.15.

53.8.4 4. Supported devices

This driver binds itself to the ACPI device TOS620A, and any Toshiba laptop with this device is supported, given the fact that they have the presence of conventional HDD and not only SSD, or a combination of both HDD and SSD.

53.8.5 5. Usage

The sysfs files under /sys/devices/LNXSYSTM:00/LNXSYBUS:00/TOS620A:00/ are:

protection_level	The protection_level is readable and writeable, and provides a way to let userspace query the current protection level, as well as set the desired protection level, the available protection levels are:
	`============  =======  ========== ␣ ↪ ========`
	`0 - Disabled  1 - Low  2 - Medium ␣ ↪ 3 - High`
	`============  =======  ========== ␣ ↪ ========`

reset_protection	The reset_protection entry is writeable only, being "1" the only parameter it accepts, it is used to trigger a reset of the protection interface.

CHAPTER

FIFTYFOUR

PARALLEL PORT LCD/KEYPAD PANEL SUPPORT

Some LCDs allow you to define up to 8 characters, mapped to ASCII characters 0 to 7. The escape code to define a new character is 'e[LG' followed by one digit from 0 to 7, representing the character number, and up to 8 couples of hex digits terminated by a semi-colon (';'). Each couple of digits represents a line, with 1-bits for each illuminated pixel with LSB on the right. Lines are numbered from the top of the character to the bottom. On a 5x7 matrix, only the 5 lower bits of the 7 first bytes are used for each character. If the string is incomplete, only complete lines will be redefined. Here are some examples:

```
printf "\e[LG0010101050D1F0C04;"  => 0 = [enter]
printf "\e[LG1040E1F0000000000;"  => 1 = [up]
printf "\e[LG2000000001F0E0400;"  => 2 = [down]
printf "\e[LG3040E1F001F0E0400;"  => 3 = [up-down]
printf "\e[LG40002060E1E0E0602;"  => 4 = [left]
printf "\e[LG500080C0E0F0E0C08;"  => 5 = [right]
printf "\e[LG60016051516141400;"  => 6 = "IP"

printf "\e[LG00103071F1F070301;"  => big speaker
printf "\e[LG00002061E1E060200;"  => small speaker
```

Willy

CHAPTER

FIFTYFIVE

LDM - LOGICAL DISK MANAGER (DYNAMIC DISKS)

Author
Originally Written by FlatCap - Richard Russon <ldm@flatcap.org>.

Last Updated
Anton Altaparmakov on 30 March 2007 for Windows Vista.

55.1 Overview

Windows 2000, XP, and Vista use a new partitioning scheme. It is a complete replacement for the MSDOS style partitions. It stores its information in a 1MiB journalled database at the end of the physical disk. The size of partitions is limited only by disk space. The maximum number of partitions is nearly 2000.

Any partitions created under the LDM are called ”Dynamic Disks”. There are no longer any primary or extended partitions. Normal MSDOS style partitions are now known as Basic Disks.

If you wish to use Spanned, Striped, Mirrored or RAID 5 Volumes, you must use Dynamic Disks. The journalling allows Windows to make changes to these partitions and filesystems without the need to reboot.

Once the LDM driver has divided up the disk, you can use the MD driver to assemble any multi-partition volumes, e.g. Stripes, RAID5.

To prevent legacy applications from repartitioning the disk, the LDM creates a dummy MSDOS partition containing one disk-sized partition. This is what is supported with the Linux LDM driver.

A newer approach that has been implemented with Vista is to put LDM on top of a GPT label disk. This is not supported by the Linux LDM driver yet.

55.2 Example

Below we have a 50MiB disk, divided into seven partitions.

Note: The missing 1MiB at the end of the disk is where the LDM database is stored.

Device		Offset Bytes	Sectors	MiB		Size Bytes	Sectors	MiB
hda		0	0	0		52428800	102400	50
hda1		51380224	100352	49		1048576	2048	1
hda2		16384	32	0		6979584	13632	6
hda3		6995968	13664	6		10485760	20480	10
hda4		17481728	34144	16		4194304	8192	4
hda5		21676032	42336	20		5242880	10240	5
hda6		26918912	52576	25		10485760	20480	10
hda7		37404672	73056	35		13959168	27264	13

The LDM Database may not store the partitions in the order that they appear on disk, but the driver will sort them.

When Linux boots, you will see something like:

```
hda: 102400 sectors w/32KiB Cache, CHS=50/64/32
hda: [LDM] hda1 hda2 hda3 hda4 hda5 hda6 hda7
```

55.3 Compiling LDM Support

To enable LDM, choose the following two options:

- ”Advanced partition selection” CONFIG_PARTITION_ADVANCED
- ”Windows Logical Disk Manager (Dynamic Disk) support” CONFIG_LDM_PARTITION

If you believe the driver isn't working as it should, you can enable the extra debugging code. This will produce a LOT of output. The option is:

- ”Windows LDM extra logging” CONFIG_LDM_DEBUG

N.B. The partition code cannot be compiled as a module.

As with all the partition code, if the driver doesn't see signs of its type of partition, it will pass control to another driver, so there is no harm in enabling it.

If you have Dynamic Disks but don't enable the driver, then all you will see is a dummy MSDOS partition filling the whole disk. You won't be able to mount any of the volumes on the disk.

55.4 Booting

If you enable LDM support, then lilo is capable of booting from any of the discovered partitions. However, grub does not understand the LDM partitioning and cannot boot from a Dynamic Disk.

55.5 More Documentation

There is an Overview of the LDM together with complete Technical Documentation. It is available for download.

http://www.linux-ntfs.org/

If you have any LDM questions that aren't answered in the documentation, email me.

Cheers,

FlatCap - Richard Russon ldm@flatcap.org

CHAPTER

FIFTYSIX

SOFTLOCKUP DETECTOR AND HARDLOCKUP DETECTOR (AKA NMI_WATCHDOG)

The Linux kernel can act as a watchdog to detect both soft and hard lockups.

A 'softlockup' is defined as a bug that causes the kernel to loop in kernel mode for more than 20 seconds (see "Implementation" below for details), without giving other tasks a chance to run. The current stack trace is displayed upon detection and, by default, the system will stay locked up. Alternatively, the kernel can be configured to panic; a sysctl, "kernel.softlockup_panic", a kernel parameter, "softlockup_panic" (see "*The kernel's command-line parameters*" for details), and a compile option, "BOOTPARAM_SOFTLOCKUP_PANIC", are provided for this.

A 'hardlockup' is defined as a bug that causes the CPU to loop in kernel mode for more than 10 seconds (see "Implementation" below for details), without letting other interrupts have a chance to run. Similarly to the softlockup case, the current stack trace is displayed upon detection and the system will stay locked up unless the default behavior is changed, which can be done through a sysctl, 'hardlockup_panic', a compile time knob, "BOOTPARAM_HARDLOCKUP_PANIC", and a kernel parameter, "nmi_watchdog" (see "*The kernel's command-line parameters*" for details).

The panic option can be used in combination with panic_timeout (this timeout is set through the confusingly named "kernel.panic" sysctl), to cause the system to reboot automatically after a specified amount of time.

56.1 Implementation

The soft and hard lockup detectors are built on top of the hrtimer and perf subsystems, respectively. A direct consequence of this is that, in principle, they should work in any architecture where these subsystems are present.

A periodic hrtimer runs to generate interrupts and kick the watchdog job. An NMI perf event is generated every "watchdog_thresh" (compile-time initialized to 10 and configurable through sysctl of the same name) seconds to check for hardlockups. If any CPU in the system does not receive any hrtimer interrupt during that time the 'hardlockup detector' (the handler for the NMI perf event) will generate a kernel warning or call panic, depending on the configuration.

The watchdog job runs in a stop scheduling thread that updates a timestamp every time it is scheduled. If that timestamp is not updated for 2*watchdog_thresh seconds (the softlockup threshold) the 'softlockup detector' (coded inside the hrtimer callback function) will dump useful debug information to the system log, after which it will call panic if it was instructed to do so or resume execution of other kernel code.

The period of the hrtimer is 2*watchdog_thresh/5, which means it has two or three chances to generate an interrupt before the hardlockup detector kicks in.

As explained above, a kernel knob is provided that allows administrators to configure the period of the hrtimer and the perf event. The right value for a particular environment is a trade-off between fast response to lockups and detection overhead.

By default, the watchdog runs on all online cores. However, on a kernel configured with NO_HZ_FULL, by default the watchdog runs only on the housekeeping cores, not the cores specified in the "nohz_full" boot argument. If we allowed the watchdog to run by default on the "nohz_full" cores, we would have to run timer ticks to activate the scheduler, which would prevent the "nohz_full" functionality from protecting the user code on those cores from the kernel. Of course, disabling it by default on the nohz_full cores means that when those cores do enter the kernel, by default we will not be able to detect if they lock up. However, allowing the watchdog to continue to run on the housekeeping (non-tickless) cores means that we will continue to detect lockups properly on those cores.

In either case, the set of cores excluded from running the watchdog may be adjusted via the kernel.watchdog_cpumask sysctl. For nohz_full cores, this may be useful for debugging a case where the kernel seems to be hanging on the nohz_full cores.

CHAPTER

FIFTYSEVEN

LINUX SECURITY MODULE USAGE

The Linux Security Module (LSM) framework provides a mechanism for various security checks to be hooked by new kernel extensions. The name ”module” is a bit of a misnomer since these extensions are not actually loadable kernel modules. Instead, they are selectable at build-time via CONFIG_DEFAULT_SECURITY and can be overridden at boot-time via the `"security=..."` kernel command line argument, in the case where multiple LSMs were built into a given kernel.

The primary users of the LSM interface are Mandatory Access Control (MAC) extensions which provide a comprehensive security policy. Examples include SELinux, Smack, Tomoyo, and AppArmor. In addition to the larger MAC extensions, other extensions can be built using the LSM to provide specific changes to system operation when these tweaks are not available in the core functionality of Linux itself.

The Linux capabilities modules will always be included. This may be followed by any number of ”minor” modules and at most one ”major” module. For more details on capabilities, see `capabilities(7)` in the Linux man-pages project.

A list of the active security modules can be found by reading `/sys/kernel/security/lsm`. This is a comma separated list, and will always include the capability module. The list reflects the order in which checks are made. The capability module will always be first, followed by any ”minor” modules (e.g. Yama) and then the one ”major” module (e.g. SELinux) if there is one configured.

Process attributes associated with ”major” security modules should be accessed and maintained using the special files in `/proc/.../attr`. A security module may maintain a module specific subdirectory there, named after the module. `/proc/.../attr/smack` is provided by the Smack security module and contains all its special files. The files directly in `/proc/.../attr` remain as legacy interfaces for modules that provide subdirectories.

57.1 AppArmor

57.1.1 What is AppArmor?

AppArmor is MAC style security extension for the Linux kernel. It implements a task centered policy, with task ”profiles” being created and loaded from user space. Tasks on the system that do not have a profile defined for them run in an unconfined state which is equivalent to standard Linux DAC permissions.

57.1.2 How to enable/disable

set `CONFIG_SECURITY_APPARMOR=y`

If AppArmor should be selected as the default security module then set:

```
CONFIG_DEFAULT_SECURITY="apparmor"
CONFIG_SECURITY_APPARMOR_BOOTPARAM_VALUE=1
```

Build the kernel

If AppArmor is not the default security module it can be enabled by passing `security=apparmor` on the kernel's command line.

If AppArmor is the default security module it can be disabled by passing `apparmor=0`, `security=XXXX` (where `XXXX` is valid security module), on the kernel's command line.

For AppArmor to enforce any restrictions beyond standard Linux DAC permissions policy must be loaded into the kernel from user space (see the Documentation and tools links).

57.1.3 Documentation

Documentation can be found on the wiki, linked below.

57.1.4 Links

Mailing List - apparmor@lists.ubuntu.com

Wiki - http://wiki.apparmor.net

User space tools - https://gitlab.com/apparmor

Kernel module - git://git.kernel.org/pub/scm/linux/kernel/git/jj/linux-apparmor

57.2 LoadPin

LoadPin is a Linux Security Module that ensures all kernel-loaded files (modules, firmware, etc) all originate from the same filesystem, with the expectation that such a filesystem is backed by a read-only device such as dm-verity or CDROM. This allows systems that have a verified and/or unchangeable filesystem to enforce module and firmware loading restrictions without needing to sign the files individually.

The LSM is selectable at build-time with `CONFIG_SECURITY_LOADPIN`, and can be controlled at boot-time with the kernel command line option "`loadpin.enforce`". By default, it is enabled, but can be disabled at boot ("`loadpin.enforce=0`").

LoadPin starts pinning when it sees the first file loaded. If the block device backing the filesystem is not read-only, a sysctl is created to toggle pinning: `/proc/sys/kernel/loadpin/enabled`. (Having a mutable filesystem means pinning is mutable too, but having the sysctl allows for easy testing on systems with a mutable filesystem.)

It's also possible to exclude specific file types from LoadPin using kernel command line option "`loadpin.exclude`". By default, all files are included, but they can be excluded using kernel command line option such as "`loadpin.exclude=kernel-module,kexec-image`". This allows to

use different mechanisms such as `CONFIG_MODULE_SIG` and `CONFIG_KEXEC_VERIFY_SIG` to verify kernel module and kernel image while still use LoadPin to protect the integrity of other files kernel loads. The full list of valid file types can be found in `kernel_read_file_str` defined in `include/linux/kernel_read_file.h`.

57.3 SELinux

If you want to use SELinux, chances are you will want to use the distro-provided policies, or install the latest reference policy release from

https://github.com/SELinuxProject/refpolicy

However, if you want to install a dummy policy for testing, you can do using `mdp` provided under scripts/selinux. Note that this requires the selinux userspace to be installed - in particular you will need checkpolicy to compile a kernel, and setfiles and fixfiles to label the filesystem.

1. Compile the kernel with selinux enabled.
2. Type `make` to compile `mdp`.
3. Make sure that you are not running with SELinux enabled and a real policy. If you are, reboot with selinux disabled before continuing.
4. Run install_policy.sh:

```
cd scripts/selinux
sh install_policy.sh
```

Step 4 will create a new dummy policy valid for your kernel, with a single selinux user, role, and type. It will compile the policy, will set your `SELINUXTYPE` to `dummy` in `/etc/selinux/config`, install the compiled policy as `dummy`, and relabel your filesystem.

57.4 Smack

> "Good for you, you've decided to clean the elevator!" - The Elevator, from Dark Star

Smack is the Simplified Mandatory Access Control Kernel. Smack is a kernel based implementation of mandatory access control that includes simplicity in its primary design goals.

Smack is not the only Mandatory Access Control scheme available for Linux. Those new to Mandatory Access Control are encouraged to compare Smack with the other mechanisms available to determine which is best suited to the problem at hand.

Smack consists of three major components:

- The kernel
- Basic utilities, which are helpful but not required
- Configuration data

The kernel component of Smack is implemented as a Linux Security Modules (LSM) module. It requires netlabel and works best with file systems that support extended attributes, although xattr support is not strictly required. It is safe to run a Smack kernel under a "vanilla" distribution.

Smack kernels use the CIPSO IP option. Some network configurations are intolerant of IP options and can impede access to systems that use them as Smack does.

Smack is used in the Tizen operating system. Please go to http://wiki.tizen.org for information about how Smack is used in Tizen.

The current git repository for Smack user space is:

git://github.com/smack-team/smack.git

This should make and install on most modern distributions. There are five commands included in smackutil:

chsmack:
display or set Smack extended attribute values

smackctl:
load the Smack access rules

smackaccess:
report if a process with one label has access to an object with another

These two commands are obsolete with the introduction of the smackfs/load2 and smackfs/cipso2 interfaces.

smackload:
properly formats data for writing to smackfs/load

smackcipso:
properly formats data for writing to smackfs/cipso

In keeping with the intent of Smack, configuration data is minimal and not strictly required. The most important configuration step is mounting the smackfs pseudo filesystem. If smackutil is installed the startup script will take care of this, but it can be manually as well.

Add this line to `/etc/fstab`:

```
smackfs /sys/fs/smackfs smackfs defaults 0 0
```

The `/sys/fs/smackfs` directory is created by the kernel.

Smack uses extended attributes (xattrs) to store labels on filesystem objects. The attributes are stored in the extended attribute security name space. A process must have `CAP_MAC_ADMIN` to change any of these attributes.

The extended attributes that Smack uses are:

SMACK64
Used to make access control decisions. In almost all cases the label given to a new filesystem object will be the label of the process that created it.

SMACK64EXEC
The Smack label of a process that execs a program file with this attribute set will run with this attribute's value.

SMACK64MMAP
Don't allow the file to be mmapped by a process whose Smack label does not allow all of the access permitted to a process with the label contained in this attribute. This is a very specific use case for shared libraries.

SMACK64TRANSMUTE
Can only have the value "TRUE". If this attribute is present on a directory when an object is created in the directory and the Smack rule (more below) that permitted the write access to the directory includes the transmute ("t") mode the object gets the label of the directory instead of the label of the creating process. If the object being created is a directory the SMACK64TRANSMUTE attribute is set as well.

SMACK64IPIN
This attribute is only available on file descriptors for sockets. Use the Smack label in this attribute for access control decisions on packets being delivered to this socket.

SMACK64IPOUT
This attribute is only available on file descriptors for sockets. Use the Smack label in this attribute for access control decisions on packets coming from this socket.

There are multiple ways to set a Smack label on a file:

```
# attr -S -s SMACK64 -V "value" path
# chsmack -a value path
```

A process can see the Smack label it is running with by reading `/proc/self/attr/current`. A process with `CAP_MAC_ADMIN` can set the process Smack by writing there.

Most Smack configuration is accomplished by writing to files in the smackfs filesystem. This pseudo-filesystem is mounted on `/sys/fs/smackfs`.

access
Provided for backward compatibility. The access2 interface is preferred and should be used instead. This interface reports whether a subject with the specified Smack label has a particular access to an object with a specified Smack label. Write a fixed format access rule to this file. The next read will indicate whether the access would be permitted. The text will be either "1" indicating access, or "0" indicating denial.

access2
This interface reports whether a subject with the specified Smack label has a particular access to an object with a specified Smack label. Write a long format access rule to this file. The next read will indicate whether the access would be permitted. The text will be either "1" indicating access, or "0" indicating denial.

ambient
This contains the Smack label applied to unlabeled network packets.

change-rule
This interface allows modification of existing access control rules. The format accepted on write is:

```
"%s %s %s %s"
```

where the first string is the subject label, the second the object label, the third the access to allow and the fourth the access to deny. The access strings may contain only the characters "rwxat-". If a rule for a given subject and object exists it will be modified by enabling the permissions in the third string and disabling those in the fourth string. If there is no such rule it will be created using the access specified in the third and the fourth strings.

cipso
Provided for backward compatibility. The cipso2 interface is preferred and should be used

instead. This interface allows a specific CIPSO header to be assigned to a Smack label. The format accepted on write is:

```
"%24s%4d%4d"["%4d"]...
```

The first string is a fixed Smack label. The first number is the level to use. The second number is the number of categories. The following numbers are the categories:

```
"level-3-cats-5-19          3   2   5  19"
```

cipso2
This interface allows a specific CIPSO header to be assigned to a Smack label. The format accepted on write is:

```
"%s%4d%4d"["%4d"]...
```

The first string is a long Smack label. The first number is the level to use. The second number is the number of categories. The following numbers are the categories:

```
"level-3-cats-5-19   3   2   5  19"
```

direct
This contains the CIPSO level used for Smack direct label representation in network packets.

doi
This contains the CIPSO domain of interpretation used in network packets.

ipv6host
This interface allows specific IPv6 internet addresses to be treated as single label hosts. Packets are sent to single label hosts only from processes that have Smack write access to the host label. All packets received from single label hosts are given the specified label. The format accepted on write is:

```
"%h:%h:%h:%h:%h:%h:%h:%h label" or
"%h:%h:%h:%h:%h:%h:%h:%h/%d label".
```

The "::" address shortcut is not supported. If label is "-DELETE" a matched entry will be deleted.

load
Provided for backward compatibility. The load2 interface is preferred and should be used instead. This interface allows access control rules in addition to the system defined rules to be specified. The format accepted on write is:

```
"%24s%24s%5s"
```

where the first string is the subject label, the second the object label, and the third the requested access. The access string may contain only the characters "rwxat-", and specifies which sort of access is allowed. The "-" is a placeholder for permissions that are not allowed. The string "r-x--" would specify read and execute access. Labels are limited to 23 characters in length.

load2
This interface allows access control rules in addition to the system defined rules to be

specified. The format accepted on write is:

```
"%s %s %s"
```

where the first string is the subject label, the second the object label, and the third the requested access. The access string may contain only the characters ”rwxat-”, and specifies which sort of access is allowed. The ”-” is a placeholder for permissions that are not allowed. The string ”r-x--” would specify read and execute access.

load-self
Provided for backward compatibility. The load-self2 interface is preferred and should be used instead. This interface allows process specific access rules to be defined. These rules are only consulted if access would otherwise be permitted, and are intended to provide additional restrictions on the process. The format is the same as for the load interface.

load-self2
This interface allows process specific access rules to be defined. These rules are only consulted if access would otherwise be permitted, and are intended to provide additional restrictions on the process. The format is the same as for the load2 interface.

logging
This contains the Smack logging state.

mapped
This contains the CIPSO level used for Smack mapped label representation in network packets.

netlabel
This interface allows specific internet addresses to be treated as single label hosts. Packets are sent to single label hosts without CIPSO headers, but only from processes that have Smack write access to the host label. All packets received from single label hosts are given the specified label. The format accepted on write is:

```
"%d.%d.%d.%d label" or "%d.%d.%d.%d/%d label".
```

If the label specified is ”-CIPSO” the address is treated as a host that supports CIPSO headers.

onlycap
This contains labels processes must have for CAP_MAC_ADMIN and `CAP_MAC_OVERRIDE` to be effective. If this file is empty these capabilities are effective at for processes with any label. The values are set by writing the desired labels, separated by spaces, to the file or cleared by writing ”-” to the file.

ptrace
This is used to define the current ptrace policy

0 - default:
this is the policy that relies on Smack access rules. For the `PTRACE_READ` a subject needs to have a read access on object. For the `PTRACE_ATTACH` a read-write access is required.

1 - exact:
this is the policy that limits `PTRACE_ATTACH`. Attach is only allowed when subject's and object's labels are equal. `PTRACE_READ` is not affected. Can be overridden with `CAP_SYS_PTRACE`.

2 - draconian:
this policy behaves like the 'exact' above with an exception that it can't be overridden with `CAP_SYS_PTRACE`.

revoke-subject
Writing a Smack label here sets the access to '-' for all access rules with that subject label.

unconfined
If the kernel is configured with `CONFIG_SECURITY_SMACK_BRINGUP` a process with `CAP_MAC_ADMIN` can write a label into this interface. Thereafter, accesses that involve that label will be logged and the access permitted if it wouldn't be otherwise. Note that this is dangerous and can ruin the proper labeling of your system. It should never be used in production.

relabel-self
This interface contains a list of labels to which the process can transition to, by writing to `/proc/self/attr/current`. Normally a process can change its own label to any legal value, but only if it has `CAP_MAC_ADMIN`. This interface allows a process without `CAP_MAC_ADMIN` to relabel itself to one of labels from predefined list. A process without `CAP_MAC_ADMIN` can change its label only once. When it does, this list will be cleared. The values are set by writing the desired labels, separated by spaces, to the file or cleared by writing "-" to the file.

If you are using the smackload utility you can add access rules in `/etc/smack/accesses`. They take the form:

```
subjectlabel objectlabel access
```

access is a combination of the letters rwxatb which specify the kind of access permitted a subject with subjectlabel on an object with objectlabel. If there is no rule no access is allowed.

Look for additional programs on http://schaufler-ca.com

57.4.1 The Simplified Mandatory Access Control Kernel (Whitepaper)

Casey Schaufler casey@schaufler-ca.com

Mandatory Access Control

Computer systems employ a variety of schemes to constrain how information is shared among the people and services using the machine. Some of these schemes allow the program or user to decide what other programs or users are allowed access to pieces of data. These schemes are called discretionary access control mechanisms because the access control is specified at the discretion of the user. Other schemes do not leave the decision regarding what a user or program can access up to users or programs. These schemes are called mandatory access control mechanisms because you don't have a choice regarding the users or programs that have access to pieces of data.

Bell & LaPadula

From the middle of the 1980's until the turn of the century Mandatory Access Control (MAC) was very closely associated with the Bell & LaPadula security model, a mathematical description of the United States Department of Defense policy for marking paper documents. MAC in this form enjoyed a following within the Capital Beltway and Scandinavian supercomputer centers but was often sited as failing to address general needs.

Domain Type Enforcement

Around the turn of the century Domain Type Enforcement (DTE) became popular. This scheme organizes users, programs, and data into domains that are protected from each other. This scheme has been widely deployed as a component of popular Linux distributions. The administrative overhead required to maintain this scheme and the detailed understanding of the whole system necessary to provide a secure domain mapping leads to the scheme being disabled or used in limited ways in the majority of cases.

Smack

Smack is a Mandatory Access Control mechanism designed to provide useful MAC while avoiding the pitfalls of its predecessors. The limitations of Bell & LaPadula are addressed by providing a scheme whereby access can be controlled according to the requirements of the system and its purpose rather than those imposed by an arcane government policy. The complexity of Domain Type Enforcement and avoided by defining access controls in terms of the access modes already in use.

Smack Terminology

The jargon used to talk about Smack will be familiar to those who have dealt with other MAC systems and shouldn't be too difficult for the uninitiated to pick up. There are four terms that are used in a specific way and that are especially important:

Subject:
A subject is an active entity on the computer system. On Smack a subject is a task, which is in turn the basic unit of execution.

Object:
An object is a passive entity on the computer system. On Smack files of all types, IPC, and tasks can be objects.

Access:
Any attempt by a subject to put information into or get information from an object is an access.

Label:
Data that identifies the Mandatory Access Control characteristics of a subject or an object.

These definitions are consistent with the traditional use in the security community. There are also some terms from Linux that are likely to crop up:

Capability:
A task that possesses a capability has permission to violate an aspect of the system security policy, as identified by the specific capability. A task that possesses one or more capabilities is a privileged task, whereas a task with no capabilities is an unprivileged task.

Privilege:
A task that is allowed to violate the system security policy is said to have privilege. As of this writing a task can have privilege either by possessing capabilities or by having an effective user of root.

Smack Basics

Smack is an extension to a Linux system. It enforces additional restrictions on what subjects can access which objects, based on the labels attached to each of the subject and the object.

Labels

Smack labels are ASCII character strings. They can be up to 255 characters long, but keeping them to twenty-three characters is recommended. Single character labels using special characters, that being anything other than a letter or digit, are reserved for use by the Smack development team. Smack labels are unstructured, case sensitive, and the only operation ever performed on them is comparison for equality. Smack labels cannot contain unprintable characters, the ”/” (slash), the ”” (backslash), the ”'” (quote) and '”' (double-quote) characters. Smack labels cannot begin with a '-'. This is reserved for special options.

There are some predefined labels:

```
_       Pronounced "floor", a single underscore character.
^       Pronounced "hat", a single circumflex character.
*       Pronounced "star", a single asterisk character.
?       Pronounced "huh", a single question mark character.
@       Pronounced "web", a single at sign character.
```

Every task on a Smack system is assigned a label. The Smack label of a process will usually be assigned by the system initialization mechanism.

Access Rules

Smack uses the traditional access modes of Linux. These modes are read, execute, write, and occasionally append. There are a few cases where the access mode may not be obvious. These include:

Signals:
A signal is a write operation from the subject task to the object task.

Internet Domain IPC:
Transmission of a packet is considered a write operation from the source task to the destination task.

Smack restricts access based on the label attached to a subject and the label attached to the object it is trying to access. The rules enforced are, in order:

1. Any access requested by a task labeled "*" is denied.
2. A read or execute access requested by a task labeled "^" is permitted.
3. A read or execute access requested on an object labeled "_" is permitted.
4. Any access requested on an object labeled "*" is permitted.
5. Any access requested by a task on an object with the same label is permitted.
6. Any access requested that is explicitly defined in the loaded rule set is permitted.
7. Any other access is denied.

Smack Access Rules

With the isolation provided by Smack access separation is simple. There are many interesting cases where limited access by subjects to objects with different labels is desired. One example is the familiar spy model of sensitivity, where a scientist working on a highly classified project would be able to read documents of lower classifications and anything she writes will be "born" highly classified. To accommodate such schemes Smack includes a mechanism for specifying rules allowing access between labels.

Access Rule Format

The format of an access rule is:

```
subject-label object-label access
```

Where subject-label is the Smack label of the task, object-label is the Smack label of the thing being accessed, and access is a string specifying the sort of access allowed. The access specification is searched for letters that describe access modes:

> a: indicates that append access should be granted. r: indicates that read access should be granted. w: indicates that write access should be granted. x: indicates that execute access should be granted. t: indicates that the rule requests transmutation. b: indicates that the rule should be reported for bring-up.

Uppercase values for the specification letters are allowed as well. Access mode specifications can be in any order. Examples of acceptable rules are:

```
TopSecret Secret  rx
Secret    Unclass R
Manager   Game    x
User      HR      w
Snap      Crackle rwxatb
New       Old     rRrRr
Closed    Off     -
```

Examples of unacceptable rules are:

```
Top Secret Secret     rx
Ace        Ace        r
Odd        spells     waxbeans
```

Spaces are not allowed in labels. Since a subject always has access to files with the same label specifying a rule for that case is pointless. Only valid letters (rwxatbRWXATB) and the dash ('-') character are allowed in access specifications. The dash is a placeholder, so ”a-r” is the same as ”ar”. A lone dash is used to specify that no access should be allowed.

Applying Access Rules

The developers of Linux rarely define new sorts of things, usually importing schemes and concepts from other systems. Most often, the other systems are variants of Unix. Unix has many endearing properties, but consistency of access control models is not one of them. Smack strives to treat accesses as uniformly as is sensible while keeping with the spirit of the underlying mechanism.

File system objects including files, directories, named pipes, symbolic links, and devices require access permissions that closely match those used by mode bit access. To open a file for reading read access is required on the file. To search a directory requires execute access. Creating a file with write access requires both read and write access on the containing directory. Deleting a file requires read and write access to the file and to the containing directory. It is possible that a user may be able to see that a file exists but not any of its attributes by the circumstance of having read access to the containing directory but not to the differently labeled file. This is an artifact of the file name being data in the directory, not a part of the file.

If a directory is marked as transmuting (SMACK64TRANSMUTE=TRUE) and the access rule that allows a process to create an object in that directory includes 't' access the label assigned to the new object will be that of the directory, not the creating process. This makes it much easier for two processes with different labels to share data without granting access to all of their files.

IPC objects, message queues, semaphore sets, and memory segments exist in flat namespaces and access requests are only required to match the object in question.

Process objects reflect tasks on the system and the Smack label used to access them is the same Smack label that the task would use for its own access attempts. Sending a signal via the kill() system call is a write operation from the signaler to the recipient. Debugging a process requires both reading and writing. Creating a new task is an internal operation that results in two tasks with identical Smack labels and requires no access checks.

Sockets are data structures attached to processes and sending a packet from one process to another requires that the sender have write access to the receiver. The receiver is not required to have read access to the sender.

Setting Access Rules

The configuration file /etc/smack/accesses contains the rules to be set at system startup. The contents are written to the special file /sys/fs/smackfs/load2. Rules can be added at any time and take effect immediately. For any pair of subject and object labels there can be only one rule, with the most recently specified overriding any earlier specification.

Task Attribute

The Smack label of a process can be read from /proc/<pid>/attr/current. A process can read its own Smack label from /proc/self/attr/current. A privileged process can change its own Smack label by writing to /proc/self/attr/current but not the label of another process.

File Attribute

The Smack label of a filesystem object is stored as an extended attribute named SMACK64 on the file. This attribute is in the security namespace. It can only be changed by a process with privilege.

Privilege

A process with CAP_MAC_OVERRIDE or CAP_MAC_ADMIN is privileged. CAP_MAC_OVERRIDE allows the process access to objects it would be denied otherwise. CAP_MAC_ADMIN allows a process to change Smack data, including rules and attributes.

Smack Networking

As mentioned before, Smack enforces access control on network protocol transmissions. Every packet sent by a Smack process is tagged with its Smack label. This is done by adding a CIPSO tag to the header of the IP packet. Each packet received is expected to have a CIPSO tag that identifies the label and if it lacks such a tag the network ambient label is assumed. Before the packet is delivered a check is made to determine that a subject with the label on the packet has write access to the receiving process and if that is not the case the packet is dropped.

CIPSO Configuration

It is normally unnecessary to specify the CIPSO configuration. The default values used by the system handle all internal cases. Smack will compose CIPSO label values to match the Smack labels being used without administrative intervention. Unlabeled packets that come into the system will be given the ambient label.

Smack requires configuration in the case where packets from a system that is not Smack that speaks CIPSO may be encountered. Usually this will be a Trusted Solaris system, but there are other, less widely deployed systems out there. CIPSO provides 3 important values, a Domain Of Interpretation (DOI), a level, and a category set with each packet. The DOI is intended to identify a group of systems that use compatible labeling schemes, and the DOI specified on the Smack system must match that of the remote system or packets will be discarded. The DOI is 3 by default. The value can be read from /sys/fs/smackfs/doi and can be changed by writing to /sys/fs/smackfs/doi.

The label and category set are mapped to a Smack label as defined in /etc/smack/cipso.

A Smack/CIPSO mapping has the form:

```
smack level [category [category]*]
```

Smack does not expect the level or category sets to be related in any particular way and does not assume or assign accesses based on them. Some examples of mappings:

```
TopSecret 7
TS:A,B    7 1 2
SecBDE    5 2 4 6
RAFTERS   7 12 26
```

The ":" and "," characters are permitted in a Smack label but have no special meaning.

The mapping of Smack labels to CIPSO values is defined by writing to /sys/fs/smackfs/cipso2.

In addition to explicit mappings Smack supports direct CIPSO mappings. One CIPSO level is used to indicate that the category set passed in the packet is in fact an encoding of the Smack label. The level used is 250 by default. The value can be read from /sys/fs/smackfs/direct and changed by writing to /sys/fs/smackfs/direct.

Socket Attributes

There are two attributes that are associated with sockets. These attributes can only be set by privileged tasks, but any task can read them for their own sockets.

SMACK64IPIN:
The Smack label of the task object. A privileged program that will enforce policy may set this to the star label.

SMACK64IPOUT:
The Smack label transmitted with outgoing packets. A privileged program may set this to match the label of another task with which it hopes to communicate.

Smack Netlabel Exceptions

You will often find that your labeled application has to talk to the outside, unlabeled world. To do this there's a special file /sys/fs/smackfs/netlabel where you can add some exceptions in the form of:

```
@IP1       LABEL1 or
@IP2/MASK  LABEL2
```

It means that your application will have unlabeled access to @IP1 if it has write access on LABEL1, and access to the subnet @IP2/MASK if it has write access on LABEL2.

Entries in the /sys/fs/smackfs/netlabel file are matched by longest mask first, like in classless IPv4 routing.

A special label '@' and an option '-CIPSO' can be used there:

```
@      means Internet, any application with any label has access to it
-CIPSO means standard CIPSO networking
```

If you don't know what CIPSO is and don't plan to use it, you can just do:

```
echo 127.0.0.1 -CIPSO > /sys/fs/smackfs/netlabel
echo 0.0.0.0/0 @      > /sys/fs/smackfs/netlabel
```

If you use CIPSO on your 192.168.0.0/16 local network and need also unlabeled Internet access, you can have:

```
echo 127.0.0.1      -CIPSO > /sys/fs/smackfs/netlabel
echo 192.168.0.0/16 -CIPSO > /sys/fs/smackfs/netlabel
echo 0.0.0.0/0      @      > /sys/fs/smackfs/netlabel
```

Writing Applications for Smack

There are three sorts of applications that will run on a Smack system. How an application interacts with Smack will determine what it will have to do to work properly under Smack.

Smack Ignorant Applications

By far the majority of applications have no reason whatever to care about the unique properties of Smack. Since invoking a program has no impact on the Smack label associated with the process the only concern likely to arise is whether the process has execute access to the program.

Smack Relevant Applications

Some programs can be improved by teaching them about Smack, but do not make any security decisions themselves. The utility ls(1) is one example of such a program.

Smack Enforcing Applications

These are special programs that not only know about Smack, but participate in the enforcement of system policy. In most cases these are the programs that set up user sessions. There are also network services that provide information to processes running with various labels.

File System Interfaces

Smack maintains labels on file system objects using extended attributes. The Smack label of a file, directory, or other file system object can be obtained using getxattr(2):

```
len = getxattr("/", "security.SMACK64", value, sizeof (value));
```

will put the Smack label of the root directory into value. A privileged process can set the Smack label of a file system object with setxattr(2):

```
len = strlen("Rubble");
rc = setxattr("/foo", "security.SMACK64", "Rubble", len, 0);
```

will set the Smack label of /foo to ”Rubble” if the program has appropriate privilege.

Socket Interfaces

The socket attributes can be read using fgetxattr(2).

A privileged process can set the Smack label of outgoing packets with fsetxattr(2):

```
len = strlen("Rubble");
rc = fsetxattr(fd, "security.SMACK64IPOUT", "Rubble", len, 0);
```

will set the Smack label ”Rubble” on packets going out from the socket if the program has appropriate privilege:

```
rc = fsetxattr(fd, "security.SMACK64IPIN, "*", strlen("*"), 0);
```

will set the Smack label ”*” as the object label against which incoming packets will be checked if the program has appropriate privilege.

Administration

Smack supports some mount options:

smackfsdef=label:
specifies the label to give files that lack the Smack label extended attribute.

smackfsroot=label:
specifies the label to assign the root of the file system if it lacks the Smack extended attribute.

smackfshat=label:
specifies a label that must have read access to all labels set on the filesystem. Not yet enforced.

smackfsfloor=label:
specifies a label to which all labels set on the filesystem must have read access. Not yet enforced.

smackfstransmute=label:
behaves exactly like smackfsroot except that it also sets the transmute flag on the root of the mount

These mount options apply to all file system types.

Smack auditing

If you want Smack auditing of security events, you need to set CONFIG_AUDIT in your kernel configuration. By default, all denied events will be audited. You can change this behavior by writing a single character to the /sys/fs/smackfs/logging file:

```
0 : no logging
1 : log denied (default)
2 : log accepted
3 : log denied & accepted
```

Events are logged as 'key=value' pairs, for each event you at least will get the subject, the object, the rights requested, the action, the kernel function that triggered the event, plus other pairs depending on the type of event audited.

Bringup Mode

Bringup mode provides logging features that can make application configuration and system bringup easier. Configure the kernel with CONFIG_SECURITY_SMACK_BRINGUP to enable these features. When bringup mode is enabled accesses that succeed due to rules marked with the ”b” access mode will logged. When a new label is introduced for processes rules can be added aggressively, marked with the ”b”. The logging allows tracking of which rules actual get used for that label.

Another feature of bringup mode is the ”unconfined” option. Writing a label to /sys/fs/smackfs/unconfined makes subjects with that label able to access any object, and objects with that label accessible to all subjects. Any access that is granted because a label is unconfined is logged. This feature is dangerous, as files and directories may be created in places they couldn't if the policy were being enforced.

57.5 TOMOYO

57.5.1 What is TOMOYO?

TOMOYO is a name-based MAC extension (LSM module) for the Linux kernel.

LiveCD-based tutorials are available at

http://tomoyo.sourceforge.jp/1.8/ubuntu12.04-live.html http://tomoyo.sourceforge.jp/1.8/centos6-live.html

Though these tutorials use non-LSM version of TOMOYO, they are useful for you to know what TOMOYO is.

57.5.2 How to enable TOMOYO?

Build the kernel with `CONFIG_SECURITY_TOMOYO=y` and pass `security=tomoyo` on kernel's command line.

Please see http://tomoyo.osdn.jp/2.5/ for details.

57.5.3 Where is documentation?

User <-> Kernel interface documentation is available at https://tomoyo.osdn.jp/2.5/policy-specification/index.html .

Materials we prepared for seminars and symposiums are available at https://osdn.jp/projects/tomoyo/docs/?category_id=532&language_id=1 . Below lists are chosen from three aspects.

What is TOMOYO?

TOMOYO Linux Overview
https://osdn.jp/projects/tomoyo/docs/lca2009-takeda.pdf

TOMOYO Linux: pragmatic and manageable security for Linux
https://osdn.jp/projects/tomoyo/docs/freedomhectaipei-tomoyo.pdf

TOMOYO Linux: A Practical Method to Understand and Protect Your Own Linux Box
https://osdn.jp/projects/tomoyo/docs/PacSec2007-en-no-demo.pdf

What can TOMOYO do?

Deep inside TOMOYO Linux
https://osdn.jp/projects/tomoyo/docs/lca2009-kumaneko.pdf

The role of "pathname based access control" in security.
https://osdn.jp/projects/tomoyo/docs/lfj2008-bof.pdf

History of TOMOYO?

Realities of Mainlining
https://osdn.jp/projects/tomoyo/docs/lfj2008.pdf

57.5.4 What is future plan?

We believe that inode based security and name based security are complementary and both should be used together. But unfortunately, so far, we cannot enable multiple LSM modules at the same time. We feel sorry that you have to give up SELinux/SMACK/AppArmor etc. when you want to use TOMOYO.

We hope that LSM becomes stackable in future. Meanwhile, you can use non-LSM version of TOMOYO, available at http://tomoyo.osdn.jp/1.8/ . LSM version of TOMOYO is a subset of non-LSM version of TOMOYO. We are planning to port non-LSM version's functionalities to LSM versions.

57.6 Yama

Yama is a Linux Security Module that collects system-wide DAC security protections that are not handled by the core kernel itself. This is selectable at build-time with `CONFIG_SECURITY_YAMA`, and can be controlled at run-time through sysctls in `/proc/sys/kernel/yama`:

57.6.1 ptrace_scope

As Linux grows in popularity, it will become a larger target for malware. One particularly troubling weakness of the Linux process interfaces is that a single user is able to examine the memory and running state of any of their processes. For example, if one application (e.g. Pidgin) was compromised, it would be possible for an attacker to attach to other running processes (e.g. Firefox, SSH sessions, GPG agent, etc) to extract additional credentials and continue to expand the scope of their attack without resorting to user-assisted phishing.

This is not a theoretical problem. SSH session hijacking and arbitrary code injection attacks already exist and remain possible if ptrace is allowed to operate as before. Since ptrace is not commonly used by non-developers and non-admins, system builders should be allowed the option to disable this debugging system.

For a solution, some applications use `prctl(PR_SET_DUMPABLE, ...)` to specifically disallow such ptrace attachment (e.g. ssh-agent), but many do not. A more general solution is to only allow ptrace directly from a parent to a child process (i.e. direct "gdb EXE" and "strace EXE" still work), or with `CAP_SYS_PTRACE` (i.e. "gdb --pid=PID", and "strace -p PID" still work as root).

In mode 1, software that has defined application-specific relationships between a debugging process and its inferior (crash handlers, etc), `prctl(PR_SET_PTRACER, pid, ...)` can be used. An inferior can declare which other process (and its descendants) are allowed to call `PTRACE_ATTACH` against it. Only one such declared debugging process can exists for each inferior at a time. For example, this is used by KDE, Chromium, and Firefox's crash handlers, and by Wine for allowing only Wine processes to ptrace each other. If a process wishes to entirely disable these ptrace restrictions, it can call `prctl(PR_SET_PTRACER, PR_SET_PTRACER_ANY, ...)` so that any otherwise allowed process (even those in external pid namespaces) may attach.

The sysctl settings (writable only with `CAP_SYS_PTRACE`) are:

0 - classic ptrace permissions:
a process can `PTRACE_ATTACH` to any other process running under the same uid, as long as it is dumpable (i.e. did not transition uids, start privileged, or have called `prctl(PR_SET_DUMPABLE...)` already). Similarly, `PTRACE_TRACEME` is unchanged.

1 - restricted ptrace:
a process must have a predefined relationship with the inferior it wants to call `PTRACE_ATTACH` on. By default, this relationship is that of only its descendants when the above classic criteria is also met. To change the relationship, an inferior can call `prctl(PR_SET_PTRACER, debugger, ...)` to declare an allowed debugger PID to call `PTRACE_ATTACH` on the inferior. Using `PTRACE_TRACEME` is unchanged.

2 - admin-only attach:
only processes with `CAP_SYS_PTRACE` may use ptrace, either with `PTRACE_ATTACH` or through children calling `PTRACE_TRACEME`.

3 - no attach:
no processes may use ptrace with `PTRACE_ATTACH` nor via `PTRACE_TRACEME`. Once set, this sysctl value cannot be changed.

The original children-only logic was based on the restrictions in grsecurity.

57.7 SafeSetID

SafeSetID is an LSM module that gates the setid family of syscalls to restrict UID/GID transitions from a given UID/GID to only those approved by a system-wide allowlist. These restrictions also prohibit the given UIDs/GIDs from obtaining auxiliary privileges associated with CAP_SET{U/G}ID, such as allowing a user to set up user namespace UID/GID mappings.

57.7.1 Background

In absence of file capabilities, processes spawned on a Linux system that need to switch to a different user must be spawned with CAP_SETUID privileges. CAP_SETUID is granted to programs running as root or those running as a non-root user that have been explicitly given the CAP_SETUID runtime capability. It is often preferable to use Linux runtime capabilities rather than file capabilities, since using file capabilities to run a program with elevated privileges opens up possible security holes since any user with access to the file can exec() that program to gain the elevated privileges.

While it is possible to implement a tree of processes by giving full CAP_SET{U/G}ID capabilities, this is often at odds with the goals of running a tree of processes under non-root user(s) in the first place. Specifically, since CAP_SETUID allows changing to any user on the system, including the root user, it is an overpowered capability for what is needed in this scenario, especially since programs often only call setuid() to drop privileges to a lesser-privileged user -- not elevate privileges. Unfortunately, there is no generally feasible way in Linux to restrict the potential UIDs that a user can switch to through setuid() beyond allowing a switch to any user on the system. This SafeSetID LSM seeks to provide a solution for restricting setid capabilities in such a way.

The main use case for this LSM is to allow a non-root program to transition to other untrusted uids without full blown CAP_SETUID capabilities. The non-root program would still need CAP_SETUID to do any kind of transition, but the additional restrictions imposed by this LSM would mean it is a ”safer” version of CAP_SETUID since the non-root program cannot take advantage of CAP_SETUID to do any unapproved actions (e.g. setuid to uid 0 or create/enter new user namespace). The higher level goal is to allow for uid-based sandboxing of system services without having to give out CAP_SETUID all over the place just so that non-root programs can drop to even-lesser-privileged uids. This is especially relevant when one non-root daemon on the system should be allowed to spawn other processes as different uids, but its undesirable to give the daemon a basically-root-equivalent CAP_SETUID.

57.7.2 Other Approaches Considered

Solve this problem in userspace

For candidate applications that would like to have restricted setid capabilities as implemented in this LSM, an alternative option would be to simply take away setid capabilities from the application completely and refactor the process spawning semantics in the application (e.g. by using a privileged helper program to do process spawning and UID/GID transitions). Unfortunately, there are a number of semantics around process spawning that would be affected by this, such as fork() calls where the program doesn't immediately call exec() after the fork(), parent processes specifying custom environment variables or command line args for spawned

child processes, or inheritance of file handles across a fork()/exec(). Because of this, as solution that uses a privileged helper in userspace would likely be less appealing to incorporate into existing projects that rely on certain process-spawning semantics in Linux.

Use user namespaces

Another possible approach would be to run a given process tree in its own user namespace and give programs in the tree setid capabilities. In this way, programs in the tree could change to any desired UID/GID in the context of their own user namespace, and only approved UIDs/GIDs could be mapped back to the initial system user namespace, affectively preventing privilege escalation. Unfortunately, it is not generally feasible to use user namespaces in isolation, without pairing them with other namespace types, which is not always an option. Linux checks for capabilities based off of the user namespace that ”owns” some entity. For example, Linux has the notion that network namespaces are owned by the user namespace in which they were created. A consequence of this is that capability checks for access to a given network namespace are done by checking whether a task has the given capability in the context of the user namespace that owns the network namespace -- not necessarily the user namespace under which the given task runs. Therefore spawning a process in a new user namespace effectively prevents it from accessing the network namespace owned by the initial namespace. This is a deal-breaker for any application that expects to retain the CAP_NET_ADMIN capability for the purpose of adjusting network configurations. Using user namespaces in isolation causes problems regarding other system interactions, including use of pid namespaces and device creation.

Use an existing LSM

None of the other in-tree LSMs have the capability to gate setid transitions, or even employ the security_task_fix_setuid hook at all. SELinux says of that hook: ”Since setuid only affects the current process, and since the SELinux controls are not based on the Linux identity attributes, SELinux does not need to control this operation.”

57.7.3 Directions for use

This LSM hooks the setid syscalls to make sure transitions are allowed if an applicable restriction policy is in place. Policies are configured through securityfs by writing to the safesetid/uid_allowlist_policy and safesetid/gid_allowlist_policy files at the location where securityfs is mounted. The format for adding a policy is '<UID>:<UID>' or '<GID>:<GID>', using literal numbers, and ending with a newline character such as '123:456n'. Writing an empty string ”” will flush the policy. Again, configuring a policy for a UID/GID will prevent that UID/GID from obtaining auxiliary setid privileges, such as allowing a user to set up user namespace UID/GID mappings.

57.7.4 Note on GID policies and setgroups()

In v5.9 we are adding support for limiting CAP_SETGID privileges as was done previously for CAP_SETUID. However, for compatibility with common sandboxing related code conventions in userspace, we currently allow arbitrary setgroups() calls for processes with CAP_SETGID restrictions. Until we add support in a future release for restricting setgroups() calls, these GID policies add no meaningful security. setgroups() restrictions will be enforced once we have the policy checking code in place, which will rely on GID policy configuration code added in v5.9.

CHAPTER

FIFTYEIGHT

RAID ARRAYS

58.1 Boot time assembly of RAID arrays

Tools that manage md devices can be found at
https://www.kernel.org/pub/linux/utils/raid/

You can boot with your md device with the following kernel command lines:

for old raid arrays without persistent superblocks:

```
md=<md device no.>,<raid level>,<chunk size factor>,<fault level>,dev0,dev1,...
↪,devn
```

for raid arrays with persistent superblocks:

```
md=<md device no.>,dev0,dev1,...,devn
```

or, to assemble a partitionable array:

```
md=d<md device no.>,dev0,dev1,...,devn
```

58.1.1 md device no.

The number of the md device

md device no.	device
0	md0
1	md1
2	md2
3	md3
4	md4

58.1.2 raid level

level of the RAID array

raid level	level
-1	linear mode
0	striped mode

other modes are only supported with persistent super blocks

58.1.3 chunk size factor

(raid-0 and raid-1 only)

Set the chunk size as 4k << n.

58.1.4 fault level

Totally ignored

58.1.5 dev0 to devn

e.g. `/dev/hda1`, `/dev/hdc1`, `/dev/sda1`, `/dev/sdb1`

A possible loadlin line (Harald Hoyer <HarryH@Royal.Net>) looks like this:

```
e:\loadlin\loadlin e:\zimage root=/dev/md0 md=0,0,4,0,/dev/hdb2,/dev/hdc3 ro
```

58.2 Boot time autodetection of RAID arrays

When md is compiled into the kernel (not as module), partitions of type 0xfd are scanned and automatically assembled into RAID arrays. This autodetection may be suppressed with the kernel parameter `raid=noautodetect`. As of kernel 2.6.9, only drives with a type 0 superblock can be autodetected and run at boot time.

The kernel parameter `raid=partitionable` (or `raid=part`) means that all auto-detected arrays are assembled as partitionable.

58.3 Boot time assembly of degraded/dirty arrays

If a raid5 or raid6 array is both dirty and degraded, it could have undetectable data corruption. This is because the fact that it is `dirty` means that the parity cannot be trusted, and the fact that it is degraded means that some datablocks are missing and cannot reliably be reconstructed (due to no parity).

For this reason, md will normally refuse to start such an array. This requires the sysadmin to take action to explicitly start the array despite possible corruption. This is normally done with:

```
mdadm --assemble --force ....
```

This option is not really available if the array has the root filesystem on it. In order to support this booting from such an array, md supports a module parameter `start_dirty_degraded` which, when set to 1, bypassed the checks and will allows dirty degraded arrays to be started.

So, to boot with a root filesystem of a dirty degraded raid 5 or 6, use:

```
md-mod.start_dirty_degraded=1
```

58.4 Superblock formats

The md driver can support a variety of different superblock formats. Currently, it supports superblock formats `0.90.0` and the `md-1` format introduced in the 2.5 development series.

The kernel will autodetect which format superblock is being used.

Superblock format `0` is treated differently to others for legacy reasons - it is the original superblock format.

58.5 General Rules - apply for all superblock formats

An array is `created` by writing appropriate superblocks to all devices.

It is `assembled` by associating each of these devices with an particular md virtual device. Once it is completely assembled, it can be accessed.

An array should be created by a user-space tool. This will write superblocks to all devices. It will usually mark the array as `unclean`, or with some devices missing so that the kernel md driver can create appropriate redundancy (copying in raid 1, parity calculation in raid 4/5).

When an array is assembled, it is first initialized with the SET_ARRAY_INFO ioctl. This contains, in particular, a major and minor version number. The major version number selects which superblock format is to be used. The minor number might be used to tune handling of the format, such as suggesting where on each device to look for the superblock.

Then each device is added using the ADD_NEW_DISK ioctl. This provides, in particular, a major and minor number identifying the device to add.

The array is started with the RUN_ARRAY ioctl.

Once started, new devices can be added. They should have an appropriate superblock written to them, and then be passed in with ADD_NEW_DISK.

Devices that have failed or are not yet active can be detached from an array using HOT_REMOVE_DISK.

58.6 Specific Rules that apply to format-0 super block arrays, and arrays with no superblock (non-persistent)

An array can be `created` by describing the array (level, chunksize etc) in a SET_ARRAY_INFO ioctl. This must have `major_version==0` and `raid_disks != 0`.

Then uninitialized devices can be added with ADD_NEW_DISK. The structure passed to ADD_NEW_DISK must specify the state of the device and its role in the array.

Once started with RUN_ARRAY, uninitialized spares can be added with HOT_ADD_DISK.

58.7 MD devices in sysfs

md devices appear in sysfs (`/sys`) as regular block devices, e.g.:

```
/sys/block/md0
```

Each `md` device will contain a subdirectory called `md` which contains further md-specific information about the device.

All md devices contain:

level
a text file indicating the `raid level`. e.g. raid0, raid1, raid5, linear, multipath, faulty. If no raid level has been set yet (array is still being assembled), the value will reflect whatever has been written to it, which may be a name like the above, or may be a number such as `0`, `5`, etc.

raid_disks
a text file with a simple number indicating the number of devices in a fully functional array. If this is not yet known, the file will be empty. If an array is being resized this will contain the new number of devices. Some raid levels allow this value to be set while the array is active. This will reconfigure the array. Otherwise it can only be set while assembling an array. A change to this attribute will not be permitted if it would reduce the size of the array. To reduce the number of drives in an e.g. raid5, the array size must first be reduced by setting the `array_size` attribute.

chunk_size
This is the size in bytes for `chunks` and is only relevant to raid levels that involve striping (0,4,5,6,10). The address space of the array is conceptually divided into chunks and consecutive chunks are striped onto neighbouring devices. The size should be at least PAGE_SIZE (4k) and should be a power of 2. This can only be set while assembling an array

layout
The `layout` for the array for the particular level. This is simply a number that is interpreted differently by different levels. It can be written while assembling an array.

array_size
This can be used to artificially constrain the available space in the array to be less than is actually available on the combined devices. Writing a number (in

Kilobytes) which is less than the available size will set the size. Any reconfiguration of the array (e.g. adding devices) will not cause the size to change. Writing the word `default` will cause the effective size of the array to be whatever size is actually available based on `level`, `chunk_size` and `component_size`.

This can be used to reduce the size of the array before reducing the number of devices in a raid4/5/6, or to support external metadata formats which mandate such clipping.

reshape_position
This is either `none` or a sector number within the devices of the array where `reshape` is up to. If this is set, the three attributes mentioned above (raid_disks, chunk_size, layout) can potentially have 2 values, an old and a new value. If these values differ, reading the attribute returns:

```
new (old)
```

and writing will effect the `new` value, leaving the `old` unchanged.

component_size
For arrays with data redundancy (i.e. not raid0, linear, faulty, multipath), all components must be the same size - or at least there must a size that they all provide space for. This is a key part or the geometry of the array. It is measured in sectors and can be read from here. Writing to this value may resize the array if the personality supports it (raid1, raid5, raid6), and if the component drives are large enough.

metadata_version
This indicates the format that is being used to record metadata about the array. It can be 0.90 (traditional format), 1.0, 1.1, 1.2 (newer format in varying locations) or `none` indicating that the kernel isn't managing metadata at all. Alternately it can be `external:` followed by a string which is set by user-space. This indicates that metadata is managed by a user-space program. Any device failure or other event that requires a metadata update will cause array activity to be suspended until the event is acknowledged.

resync_start
The point at which resync should start. If no resync is needed, this will be a very large number (or `none` since 2.6.30-rc1). At array creation it will default to 0, though starting the array as `clean` will set it much larger.

new_dev
This file can be written but not read. The value written should be a block device number as major:minor. e.g. 8:0 This will cause that device to be attached to the array, if it is available. It will then appear at md/dev-XXX (depending on the name of the device) and further configuration is then possible.

safe_mode_delay
When an md array has seen no write requests for a certain period of time, it will be marked as `clean`. When another write request arrives, the array is marked as `dirty` before the write commences. This is known as `safe_mode`. The `certain period` is controlled by this file which stores the period as a number of seconds. The default is 200msec (0.200). Writing a value of 0 disables safemode.

array_state

This file contains a single word which describes the current state of the array. In many cases, the state can be set by writing the word for the desired state, however some states cannot be explicitly set, and some transitions are not allowed.

Select/poll works on this file. All changes except between Active_idle and active (which can be frequent and are not very interesting) are notified. active->active_idle is reported if the metadata is externally managed.

clear
No devices, no size, no level

Writing is equivalent to STOP_ARRAY ioctl

inactive
May have some settings, but array is not active all IO results in error

When written, doesn't tear down array, but just stops it

suspended (not supported yet)
All IO requests will block. The array can be reconfigured.

Writing this, if accepted, will block until array is quiescent

readonly
no resync can happen. no superblocks get written.

Write requests fail

read-auto
like readonly, but behaves like `clean` on a write request.

clean
no pending writes, but otherwise active.

When written to inactive array, starts without resync

If a write request arrives then if metadata is known, mark `dirty` and switch to `active`. if not known, block and switch to write-pending

If written to an active array that has pending writes, then fails.

active
fully active: IO and resync can be happening. When written to inactive array, starts with resync

write-pending
clean, but writes are blocked waiting for `active` to be written.

active-idle
like active, but no writes have been seen for a while (safe_mode_delay).

bitmap/location
This indicates where the write-intent bitmap for the array is stored.

It can be one of `none`, `file` or `[+-]N`. `file` may later be extended to `file:/file/name` `[+-]N` means that many sectors from the start of the metadata.

This is replicated on all devices. For arrays with externally managed metadata, the offset is from the beginning of the device.

bitmap/chunksize
The size, in bytes, of the chunk which will be represented by a single bit. For

RAID456, it is a portion of an individual device. For RAID10, it is a portion of the array. For RAID1, it is both (they come to the same thing).

bitmap/time_base
The time, in seconds, between looking for bits in the bitmap to be cleared. In the current implementation, a bit will be cleared between 2 and 3 times `time_base` after all the covered blocks are known to be in-sync.

bitmap/backlog
When write-mostly devices are active in a RAID1, write requests to those devices proceed in the background - the filesystem (or other user of the device) does not have to wait for them. `backlog` sets a limit on the number of concurrent background writes. If there are more than this, new writes will by synchronous.

bitmap/metadata
This can be either `internal` or `external`.

`internal`
is the default and means the metadata for the bitmap is stored in the first 256 bytes of the allocated space and is managed by the md module.

`external`
means that bitmap metadata is managed externally to the kernel (i.e. by some userspace program)

bitmap/can_clear
This is either `true` or `false`. If `true`, then bits in the bitmap will be cleared when the corresponding blocks are thought to be in-sync. If `false`, bits will never be cleared. This is automatically set to `false` if a write happens on a degraded array, or if the array becomes degraded during a write. When metadata is managed externally, it should be set to true once the array becomes non-degraded, and this fact has been recorded in the metadata.

consistency_policy
This indicates how the array maintains consistency in case of unexpected shutdown. It can be:

none
Array has no redundancy information, e.g. raid0, linear.

resync
Full resync is performed and all redundancy is regenerated when the array is started after unclean shutdown.

bitmap
Resync assisted by a write-intent bitmap.

journal
For raid4/5/6, journal device is used to log transactions and replay after unclean shutdown.

ppl
For raid5 only, Partial Parity Log is used to close the write hole and eliminate resync.

The accepted values when writing to this file are `ppl` and `resync`, used to enable and disable PPL.

uuid
This indicates the UUID of the array in the following format: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx

As component devices are added to an md array, they appear in the `md` directory as new directories named:

```
dev-XXX
```

where `XXX` is a name that the kernel knows for the device, e.g. hdb1. Each directory contains:

block
a symlink to the block device in /sys/block, e.g.:

```
/sys/block/md0/md/dev-hdb1/block -> ../../../../block/hdb/hdb1
```

super
A file containing an image of the superblock read from, or written to, that device.

state
A file recording the current state of the device in the array which can be a comma separated list of:

faulty
device has been kicked from active use due to a detected fault, or it has unacknowledged bad blocks

in_sync
device is a fully in-sync member of the array

writemostly
device will only be subject to read requests if there are no other options.

This applies only to raid1 arrays.

blocked
device has failed, and the failure hasn't been acknowledged yet by the metadata handler.

Writes that would write to this device if it were not faulty are blocked.

spare
device is working, but not a full member.

This includes spares that are in the process of being recovered to

write_error
device has ever seen a write error.

want_replacement
device is (mostly) working but probably should be replaced, either due to errors or due to user request.

replacement
device is a replacement for another active device with same raid_disk.

This list may grow in future.

This can be written to.

Writing `faulty` simulates a failure on the device.

Writing `remove` removes the device from the array.

Writing `writemostly` sets the writemostly flag.

Writing `-writemostly` clears the writemostly flag.

Writing `blocked` sets the `blocked` flag.

Writing `-blocked` clears the `blocked` flags and allows writes to complete and possibly simulates an error.

Writing `in_sync` sets the in_sync flag.

Writing `write_error` sets writeerrorseen flag.

Writing `-write_error` clears writeerrorseen flag.

Writing `want_replacement` is allowed at any time except to a replacement device or a spare. It sets the flag.

Writing `-want_replacement` is allowed at any time. It clears the flag.

Writing `replacement` or `-replacement` is only allowed before starting the array. It sets or clears the flag.

This file responds to select/poll. Any change to `faulty` or `blocked` causes an event.

errors
An approximate count of read errors that have been detected on this device but have not caused the device to be evicted from the array (either because they were corrected or because they happened while the array was read-only). When using version-1 metadata, this value persists across restarts of the array.

This value can be written while assembling an array thus providing an ongoing count for arrays with metadata managed by userspace.

slot
This gives the role that the device has in the array. It will either be `none` if the device is not active in the array (i.e. is a spare or has failed) or an integer less than the `raid_disks` number for the array indicating which position it currently fills. This can only be set while assembling an array. A device for which this is set is assumed to be working.

offset
This gives the location in the device (in sectors from the start) where data from the array will be stored. Any part of the device before this offset is not touched, unless it is used for storing metadata (Formats 1.1 and 1.2).

size
The amount of the device, after the offset, that can be used for storage of data. This will normally be the same as the component_size. This can be written while assembling an array. If a value less than the current component_size is written, it will be rejected.

recovery_start
When the device is not `in_sync`, this records the number of sectors from the start of the device which are known to be correct. This is normally zero, but during

a recovery operation it will steadily increase, and if the recovery is interrupted, restoring this value can cause recovery to avoid repeating the earlier blocks. With v1.x metadata, this value is saved and restored automatically.

This can be set whenever the device is not an active member of the array, either before the array is activated, or before the `slot` is set.

Setting this to `none` is equivalent to setting `in_sync`. Setting to any other value also clears the `in_sync` flag.

bad_blocks
This gives the list of all known bad blocks in the form of start address and length (in sectors respectively). If output is too big to fit in a page, it will be truncated. Writing `sector length` to this file adds new acknowledged (i.e. recorded to disk safely) bad blocks.

unacknowledged_bad_blocks
This gives the list of known-but-not-yet-saved-to-disk bad blocks in the same form of `bad_blocks`. If output is too big to fit in a page, it will be truncated. Writing to this file adds bad blocks without acknowledging them. This is largely for testing.

ppl_sector, ppl_size
Location and size (in sectors) of the space used for Partial Parity Log on this device.

An active md device will also contain an entry for each active device in the array. These are named:

```
rdNN
```

where `NN` is the position in the array, starting from 0. So for a 3 drive array there will be rd0, rd1, rd2. These are symbolic links to the appropriate `dev-XXX` entry. Thus, for example:

```
cat /sys/block/md*/md/rd*/state
```

will show `in_sync` on every line.

Active md devices for levels that support data redundancy (1,4,5,6,10) also have

sync_action
a text file that can be used to monitor and control the rebuild process. It contains one word which can be one of:

resync
redundancy is being recalculated after unclean shutdown or creation

recover
a hot spare is being built to replace a failed/missing device

idle
nothing is happening

check
A full check of redundancy was requested and is happening. This reads all blocks and checks them. A repair may also happen for some raid levels.

repair
A full check and repair is happening. This is similar to `resync`, but was requested by the user, and the write-intent bitmap is NOT used to optimise the process.

This file is writable, and each of the strings that could be read are meaningful for writing.

`idle` will stop an active resync/recovery etc. There is no guarantee that another resync/recovery may not be automatically started again, though some event will be needed to trigger this.

`resync` or `recovery` can be used to restart the corresponding operation if it was stopped with `idle`.

`check` and `repair` will start the appropriate process providing the current state is `idle`.

This file responds to select/poll. Any important change in the value triggers a poll event. Sometimes the value will briefly be `recover` if a recovery seems to be needed, but cannot be achieved. In that case, the transition to `recover` isn't notified, but the transition away is.

degraded
This contains a count of the number of devices by which the arrays is degraded. So an optimal array will show `0`. A single failed/missing drive will show `1`, etc.

This file responds to select/poll, any increase or decrease in the count of missing devices will trigger an event.

mismatch_count
When performing `check` and `repair`, and possibly when performing `resync`, md will count the number of errors that are found. The count in `mismatch_cnt` is the number of sectors that were re-written, or (for `check`) would have been re-written. As most raid levels work in units of pages rather than sectors, this may be larger than the number of actual errors by a factor of the number of sectors in a page.

bitmap_set_bits
If the array has a write-intent bitmap, then writing to this attribute can set bits in the bitmap, indicating that a resync would need to check the corresponding blocks. Either individual numbers or start-end pairs can be written. Multiple numbers can be separated by a space.

Note that the numbers are `bit` numbers, not `block` numbers. They should be scaled by the bitmap_chunksize.

sync_speed_min, sync_speed_max
This are similar to `/proc/sys/dev/raid/speed_limit_{min,max}` however they only apply to the particular array.

If no value has been written to these, or if the word `system` is written, then the system-wide value is used. If a value, in kibibytes-per-second is written, then it is used.

When the files are read, they show the currently active value followed by `(local)` or `(system)` depending on whether it is a locally set or system-wide value.

sync_completed
This shows the number of sectors that have been completed of whatever the current sync_action is, followed by the number of sectors in total that could need to be processed. The two numbers are separated by a `/` thus effectively showing one value, a fraction of the process that is complete.

A `select` on this attribute will return when resync completes, when it reaches the current sync_max (below) and possibly at other times.

sync_speed
This shows the current actual speed, in K/sec, of the current sync_action. It is averaged over the last 30 seconds.

suspend_lo, suspend_hi
The two values, given as numbers of sectors, indicate a range within the array where IO will be blocked. This is currently only supported for raid4/5/6.

sync_min, sync_max
The two values, given as numbers of sectors, indicate a range within the array where `check/repair` will operate. Must be a multiple of chunk_size. When it reaches `sync_max` it will pause, rather than complete. You can use `select` or `poll` on `sync_completed` to wait for that number to reach sync_max. Then you can either increase `sync_max`, or can write `idle` to `sync_action`.

The value of `max` for `sync_max` effectively disables the limit. When a resync is active, the value can only ever be increased, never decreased. The value of `0` is the minimum for `sync_min`.

Each active md device may also have attributes specific to the personality module that manages it. These are specific to the implementation of the module and could change substantially if the implementation changes.

These currently include:

stripe_cache_size (currently raid5 only)
number of entries in the stripe cache. This is writable, but there are upper and lower limits (32768, 17). Default is 256.

strip_cache_active (currently raid5 only)
number of active entries in the stripe cache

preread_bypass_threshold (currently raid5 only)
number of times a stripe requiring preread will be bypassed by a stripe that does not require preread. For fairness defaults to 1. Setting this to 0 disables bypass accounting and requires preread stripes to wait until all full-width stripe- writes are complete. Valid values are 0 to stripe_cache_size.

journal_mode (currently raid5 only)
The cache mode for raid5. raid5 could include an extra disk for caching. The mode can be ”write-throuth” and ”write-back”. The default is ”write-through”.

ppl_write_hint
NVMe stream ID to be set for each PPL write request.

CHAPTER

FIFTYNINE

MEDIA SUBSYSTEM ADMIN AND USER GUIDE

This section contains usage information about media subsystem and its supported drivers.

Please see:

Documentation/userspace-api/media/index.rst

- for the userspace APIs used on media devices.

Documentation/driver-api/media/index.rst

- for driver development information and Kernel APIs used by media devices;

59.1 Introduction

The media subsystem consists on Linux support for several different types of devices:

- Audio and video grabbers;
- PC and Laptop Cameras;
- Complex cameras found on Embedded hardware;
- Analog and digital TV;
- HDMI Customer Electronics Control (CEC);
- Multi-touch input devices;
- Remote Controllers;
- Media encoders and decoders.

Due to the diversity of devices, the subsystem provides several different APIs:

- Remote Controller API;
- HDMI CEC API;
- Video4Linux API;
- Media controller API;
- Video4Linux Request API (experimental);
- Digital TV API (also known as DVB API).

59.2 Building support for a media device

The first step is to download the Kernel's source code, either via a distribution-specific source file or via the Kernel's main git tree[1].

Please notice, however, that, if:

- you're a braveheart and want to experiment with new stuff;
- if you want to report a bug;
- if you're developing new patches

you should use the main media development tree `master` branch:

https://git.linuxtv.org/media_tree.git/

In this case, you may find some useful information at the LinuxTv wiki pages:

https://linuxtv.org/wiki/index.php/How_to_Obtain,_Build_and_Install_V4L-DVB_Device_Drivers

59.2.1 Configuring the Linux Kernel

You can access a menu of Kernel building options with:

```
$ make menuconfig
```

Then, select all desired options and exit it, saving the configuration.

The changed configuration will be at the `.config` file. It would look like:

```
...
# CONFIG_RC_CORE is not set
# CONFIG_CEC_CORE is not set
CONFIG_MEDIA_SUPPORT=m
CONFIG_MEDIA_SUPPORT_FILTER=y
...
```

The media subsystem is controlled by those menu configuration options:

```
Device Drivers --->
    <M> Remote Controller support  --->
    [ ] HDMI CEC RC integration
    [ ] Enable CEC error injection support
    [*] HDMI CEC drivers  --->
    <*> Multimedia support  --->
```

The `Remote Controller support` option enables the core support for remote controllers[2].

[1] The upstream Linux Kernel development tree is located at https://git.kernel.org/pub/scm/li nux/kernel/git/torvalds/linux.git/

[2] `Remote Controller support` should also be enabled if you want to use some TV card drivers that may depend on the remote controller core support.

The `HDMI CEC RC integration` option enables integration of HDMI CEC with Linux, allowing to receive data via HDMI CEC as if it were produced by a remote controller directly connected to the machine.

The `HDMI CEC drivers` option allow selecting platform and USB drivers that receives and/or transmits CEC codes via HDMI interfaces[3].

The last option (`Multimedia support`) enables support for cameras, audio/video grabbers and TV.

The media subsystem support can either be built together with the main Kernel or as a module. For most use cases, it is preferred to have it built as modules.

Note: Instead of using a menu, the Kernel provides a script with allows enabling configuration options directly. To enable media support and remote controller support using Kernel modules, you could use:

```
$ scripts/config -m RC_CORE
$ scripts/config -m MEDIA_SUPPORT
```

Media dependencies

It should be noticed that enabling the above from a clean config is usually not enough. The media subsystem depends on several other Linux core support in order to work.

For example, most media devices use a serial communication bus in order to talk with some peripherals. Such bus is called I^2C (Inter-Integrated Circuit). In order to be able to build support for such hardware, the I^2C bus support should be enabled, either via menu or with:

```
./scripts/config -m I2C
```

Another example: the remote controller core requires support for input devices, with can be enabled with:

```
./scripts/config -m INPUT
```

Other core functionality may also be needed (like PCI and/or USB support), depending on the specific driver(s) you would like to enable.

[3] Please notice that the DRM subsystem also have drivers for GPUs that use the media HDMI CEC support. Those GPU-specific drivers are selected via the `Graphics support` menu, under `Device Drivers`. When a GPU driver supports HDMI CEC, it will automatically enable the CEC core support at the media subsystem.

Enabling Remote Controller Support

The remote controller menu allows selecting drivers for specific devices. It's menu looks like this:

```
--- Remote Controller support
<M>   Compile Remote Controller keymap modules
[*]   LIRC user interface
[*]     Support for eBPF programs attached to lirc devices
[*]   Remote controller decoders  --->
[*]   Remote Controller devices  --->
```

The `Compile Remote Controller keymap modules` option creates key maps for several popular remote controllers.

The `LIRC user interface` option adds enhanced functionality when using the `lirc` program, by enabling an API that allows userspace to receive raw data from remote controllers.

The `Support for eBPF programs attached to lirc devices` option allows the usage of special programs (called eBPF) that would allow applications to add extra remote controller decoding functionality to the Linux Kernel.

The `Remote controller decoders` option allows selecting the protocols that will be recognized by the Linux Kernel. Except if you want to disable some specific decoder, it is suggested to keep all sub-options enabled.

The `Remote Controller devices` allows you to select the drivers that would be needed to support your device.

The same configuration can also be set via the `script/config` script. So, for instance, in order to support the ITE remote controller driver (found on Intel NUCs and on some ASUS x86 desktops), you could do:

```
$ scripts/config -e INPUT
$ scripts/config -e ACPI
$ scripts/config -e MODULES
$ scripts/config -m RC_CORE
$ scripts/config -e RC_DEVICES
$ scripts/config -e RC_DECODERS
$ scripts/config -m IR_RC5_DECODER
$ scripts/config -m IR_ITE_CIR
```

Enabling HDMI CEC Support

The HDMI CEC support is set automatically when a driver requires it. So, all you need to do is to enable support either for a graphics card that needs it or by one of the existing HDMI drivers.

The HDMI-specific drivers are available at the `HDMI CEC drivers` menu[4]:

[4] The above contents is just an example. The actual options for HDMI devices depends on the system's architecture and may vary on new Kernels.

```
--- HDMI CEC drivers
< >   ChromeOS EC CEC driver
< >   Amlogic Meson AO CEC driver
< >   Amlogic Meson G12A AO CEC driver
< >   Generic GPIO-based CEC driver
< >   Samsung S5P CEC driver
< >   STMicroelectronics STiH4xx HDMI CEC driver
< >   STMicroelectronics STM32 HDMI CEC driver
< >   Tegra HDMI CEC driver
< >   SECO Boards HDMI CEC driver
[ ]     SECO Boards IR RC5 support
< >   Pulse Eight HDMI CEC
< >   RainShadow Tech HDMI CEC
```

Enabling Media Support

The Media menu has a lot more options than the remote controller menu. Once selected, you should see the following options:

```
--- Media support
[ ] Filter media drivers
[*] Autoselect ancillary drivers
    Media device types --->
    Media core support --->
    Video4Linux options --->
    Media controller options --->
    Digital TV options --->
    HDMI CEC options --->
    Media drivers --->
    Media ancillary drivers --->
```

Except if you know exactly what you're doing, or if you want to build a driver for a SoC platform, it is strongly recommended to keep the `Autoselect ancillary drivers` option turned on, as it will auto-select the needed I^2C ancillary drivers.

There are now two ways to select media device drivers, as described below.

`Filter media drivers` menu

This menu is meant to easy setup for PC and Laptop hardware. It works by letting the user to specify what kind of media drivers are desired, with those options:

```
[ ] Cameras and video grabbers
[ ] Analog TV
[ ] Digital TV
[ ] AM/FM radio receivers/transmitters
[ ] Software defined radio
[ ] Platform-specific devices
[ ] Test drivers
```

So, if you want to add support to a camera or video grabber only, select just the first option. Multiple options are allowed.

Once the options on this menu are selected, the building system will auto-select the needed core drivers in order to support the selected functionality.

Note: Most TV cards are hybrid: they support both Analog TV and Digital TV.

If you have an hybrid card, you may need to enable both `Analog TV` and `Digital TV` at the menu.

When using this option, the defaults for the media support core functionality are usually good enough to provide the basic functionality for the driver. Yet, you could manually enable some desired extra (optional) functionality using the settings under each of the following `Media support` sub-menus:

```
Media core support --->
Video4Linux options --->
Media controller options --->
Digital TV options --->
HDMI CEC options --->
```

Once you select the desired filters, the drivers that matches the filtering criteria will be available at the `Media support->Media drivers` sub-menu.

`Media Core Support` menu without filtering

If you disable the `Filter media drivers` menu, all drivers available for your system whose dependencies are met should be shown at the `Media drivers` menu.

Please notice, however, that you should first ensure that the `Media Core Support` menu has all the core functionality your drivers would need, as otherwise the corresponding device drivers won't be shown.

Example

In order to enable modular support for one of the boards listed on *this table*, with modular media core modules, the `.config` file should contain those lines:

```
CONFIG_MODULES=y
CONFIG_USB=y
CONFIG_I2C=y
CONFIG_INPUT=y
CONFIG_RC_CORE=m
CONFIG_MEDIA_SUPPORT=m
CONFIG_MEDIA_SUPPORT_FILTER=y
CONFIG_MEDIA_ANALOG_TV_SUPPORT=y
CONFIG_MEDIA_DIGITAL_TV_SUPPORT=y
CONFIG_MEDIA_USB_SUPPORT=y
CONFIG_VIDEO_CX231XX=y
CONFIG_VIDEO_CX231XX_DVB=y
```

59.2.2 Building and installing a new Kernel

Once the `.config` file has everything needed, all it takes to build is to run the `make` command:

```
$ make
```

And then install the new Kernel and its modules:

```
$ sudo make modules_install
$ sudo make install
```

59.2.3 Building just the new media drivers and core

Running a new development Kernel from the development tree is usually risky, because it may have experimental changes that may have bugs. So, there are some ways to build just the new drivers, using alternative trees.

There is the Linux Kernel backports project, with contains newer drivers meant to be compiled against stable Kernels.

The LinuxTV developers, with are responsible for maintaining the media subsystem also maintains a backport tree, with just the media drivers daily updated from the newest kernel. Such tree is available at:

https://git.linuxtv.org/media_build.git/

It should be noticed that, while it should be relatively safe to use the `media_build` tree for testing purposes, there are not warranties that it would work (or even build) on a random Kernel. This tree is maintained using a ”best-efforts” principle, as time permits us to fix issues there.

If you notice anything wrong on it, feel free to submit patches at the Linux media subsystem's mailing list: media@vger.kernel.org. Please add `[PATCH media-build]` at the e-mail's subject if you submit a new patch for the media-build.

Before using it, you should run:

```
$ ./build
```

Note:

1) you may need to run it twice if the `media-build` tree gets updated;
2) you may need to do a `make distclean` if you had built it in the past for a different Kernel version than the one you're currently using;
3) by default, it will use the same config options for media as the ones defined on the Kernel you're running.

In order to select different drivers or different config options, use:

```
$ make menuconfig
```

Then, you can build and install the new drivers:

```
$ make && sudo make install
```

This will override the previous media drivers that your Kernel were using.

59.3 Infrared remote control support in video4linux drivers

Authors: Gerd Hoffmann, Mauro Carvalho Chehab

59.3.1 Basics

Most analog and digital TV boards support remote controllers. Several of them have a microprocessor that receives the IR carriers, convert into pulse/space sequences and then to scan codes, returning such codes to userspace (”scancode mode”). Other boards return just the pulse/space sequences (”raw mode”).

The support for remote controller in scancode mode is provided by the standard Linux input layer. The support for raw mode is provided via LIRC.

In order to check the support and test it, it is suggested to download the v4l-utils. It provides two tools to handle remote controllers:

- ir-keytable: provides a way to query the remote controller, list the protocols it supports, enable in-kernel support for IR decoder or switch the protocol and to test the reception of scan codes;
- ir-ctl: provide tools to handle remote controllers that support raw mode via LIRC interface.

Usually, the remote controller module is auto-loaded when the TV card is detected. However, for a few devices, you need to manually load the ir-kbd-i2c module.

59.3.2 How it works

The modules register the remote as keyboard within the linux input layer, i.e. you'll see the keys of the remote as normal key strokes (if CONFIG_INPUT_KEYBOARD is enabled).

Using the event devices (CONFIG_INPUT_EVDEV) it is possible for applications to access the remote via /dev/input/event<n> devices. The udev/systemd will automatically create the devices. If you install the v4l-utils, it may also automatically load a different keytable than the default one. Please see v4l-utils ir-keytable.1 man page for details.

The ir-keytable tool is nice for trouble shooting, i.e. to check whenever the input device is really present, which of the devices it is, check whenever pressing keys on the remote actually generates events and the like. You can also use any other input utility that changes the keymaps, like the input kbd utility.

Using with lircd

The latest versions of the lircd daemon supports reading events from the linux input layer (via event device). It also supports receiving IR codes in lirc mode.

Using without lircd

Xorg recognizes several IR keycodes that have its numerical value lower than 247. With the advent of Wayland, the input driver got updated too, and should now accept all keycodes. Yet, you may want to just reassign the keycodes to something that your favorite media application likes.

This can be done by setting v4l-utils to load your own keytable in runtime. Please read ir-keytable.1 man page for details.

59.4 HDMI CEC

59.4.1 Supported hardware in mainline

HDMI Transmitters:

- Exynos4
- Exynos5
- STIH4xx HDMI CEC
- V4L2 adv7511 (same HW, but a different driver from the drm adv7511)
- stm32
- Allwinner A10 (sun4i)
- Raspberry Pi
- dw-hdmi (Synopsis IP)
- amlogic (meson ao-cec and ao-cec-g12a)
- drm adv7511/adv7533
- omap4
- tegra
- rk3288, rk3399
- tda998x
- DisplayPort CEC-Tunneling-over-AUX on i915, nouveau and amdgpu
- ChromeOS EC CEC
- CEC for SECO boards (UDOO x86).
- Chrontel CH7322

HDMI Receivers:

- adv7604/11/12

- adv7842
- tc358743

USB Dongles (see below for additional information on how to use these dongles):

- Pulse-Eight: the pulse8-cec driver implements the following module option: `persistent_config`: by default this is off, but when set to 1 the driver will store the current settings to the device's internal eeprom and restore it the next time the device is connected to the USB port.
- RainShadow Tech. Note: this driver does not support the persistent_config module option of the Pulse-Eight driver. The hardware supports it, but I have no plans to add this feature. But I accept patches :-)

Miscellaneous:

- vivid: emulates a CEC receiver and CEC transmitter. Can be used to test CEC applications without actual CEC hardware.
- cec-gpio. If the CEC pin is hooked up to a GPIO pin then you can control the CEC line through this driver. This supports error injection as well.
- cec-gpio and Allwinner A10 (or any other driver that uses the CEC pin framework to drive the CEC pin directly): the CEC pin framework uses high-resolution timers. These timers are affected by NTP daemons that speed up or slow down the clock to sync with the official time. The chronyd server will by default increase or decrease the clock by 1/12th. This will cause the CEC timings to go out of spec. To fix this, add a 'maxslewrate 40000' line to chronyd.conf. This limits the clock frequency change to 1/25th, which keeps the CEC timings within spec.

59.4.2 Utilities

Utilities are available here: https://git.linuxtv.org/v4l-utils.git

`utils/cec-ctl`: control a CEC device

`utils/cec-compliance`: test compliance of a remote CEC device

`utils/cec-follower`: emulate a CEC follower device

Note that `cec-ctl` has support for the CEC Hospitality Profile as is used in some hotel displays. See http://www.htng.org.

Note that the libcec library (https://github.com/Pulse-Eight/libcec) supports the linux CEC framework.

If you want to get the CEC specification, then look at the References of the HDMI wikipedia page: https://en.wikipedia.org/wiki/HDMI. CEC is part of the HDMI specification. HDMI 1.3 is freely available (very similar to HDMI 1.4 w.r.t. CEC) and should be good enough for most things.

59.4.3 DisplayPort to HDMI Adapters with working CEC

Background: most adapters do not support the CEC Tunneling feature, and of those that do many did not actually connect the CEC pin. Unfortunately, this means that while a CEC device is created, it is actually all alone in the world and will never be able to see other CEC devices.

This is a list of known working adapters that have CEC Tunneling AND that properly connected the CEC pin. If you find adapters that work but are not in this list, then drop me a note.

To test: hook up your DP-to-HDMI adapter to a CEC capable device (typically a TV), then run:

```
cec-ctl --playback      # Configure the PC as a CEC Playback device
cec-ctl -S              # Show the CEC topology
```

The `cec-ctl -S` command should show at least two CEC devices, ourselves and the CEC device you are connected to (i.e. typically the TV).

General note: I have only seen this work with the Parade PS175, PS176 and PS186 chipsets and the MegaChips 2900. While MegaChips 28x0 claims CEC support, I have never seen it work.

USB-C to HDMI

Samsung Multiport Adapter EE-PW700: https://www.samsung.com/ie/support/model/EE-PW700BBEGWW/

Kramer ADC-U31C/HF: https://www.kramerav.com/product/ADC-U31C/HF

Club3D CAC-2504: https://www.club-3d.com/en/detail/2449/usb_3.1_type_c_to_hdmi_2.0_uhd_4k_60hz_active_adapter/

DisplayPort to HDMI

Club3D CAC-1080: https://www.club-3d.com/en/detail/2442/displayport_1.4_to_hdmi_2.0b_hdr/

CableCreation (SKU: CD0712): https://www.cablecreation.com/products/active-displayport-to-hdmi-adapter-4k-hdr

HP DisplayPort to HDMI True 4k Adapter (P/N 2JA63AA): https://www.hp.com/us-en/shop/pdp/hp-displayport-to-hdmi-true-4k-adapter

Mini-DisplayPort to HDMI

Club3D CAC-1180: https://www.club-3d.com/en/detail/2443/mini_displayport_1.4_to_hdmi_2.0b_hdr/

Note that passive adapters will never work, you need an active adapter.

The Club3D adapters in this list are all MegaChips 2900 based. Other Club3D adapters are PS176 based and do NOT have the CEC pin hooked up, so only the three Club3D adapters above are known to work.

I suspect that MegaChips 2900 based designs in general are likely to work whereas with the PS176 it is more hit-and-miss (mostly miss). The PS186 is likely to have the CEC pin hooked up, it looks like they changed the reference design for that chipset.

59.4.4 USB CEC Dongles

These dongles appear as `/dev/ttyACMX` devices and need the `inputattach` utility to create the `/dev/cecX` devices. Support for the Pulse-Eight has been added to `inputattach` 1.6.0. Support for the Rainshadow Tech has been added to `inputattach` 1.6.1.

You also need udev rules to automatically start systemd services:

```
SUBSYSTEM=="tty", KERNEL=="ttyACM[0-9]*", ATTRS{idVendor}=="2548", ATTRS
↪{idProduct}=="1002", ACTION=="add", TAG+="systemd", ENV{SYSTEMD_WANTS}+=
↪"pulse8-cec-inputattach@%k.service"
SUBSYSTEM=="tty", KERNEL=="ttyACM[0-9]*", ATTRS{idVendor}=="2548", ATTRS
↪{idProduct}=="1001", ACTION=="add", TAG+="systemd", ENV{SYSTEMD_WANTS}+=
↪"pulse8-cec-inputattach@%k.service"
SUBSYSTEM=="tty", KERNEL=="ttyACM[0-9]*", ATTRS{idVendor}=="04d8", ATTRS
↪{idProduct}=="ff59", ACTION=="add", TAG+="systemd", ENV{SYSTEMD_WANTS}+=
↪"rainshadow-cec-inputattach@%k.service"
```

and these systemd services:

For Pulse-Eight make /lib/systemd/system/pulse8-cec-inputattach@.service:

```
[Unit]
Description=inputattach for pulse8-cec device on %I

[Service]
Type=simple
ExecStart=/usr/bin/inputattach --pulse8-cec /dev/%I
```

For the RainShadow Tech make /lib/systemd/system/rainshadow-cec-inputattach@.service:

```
[Unit]
Description=inputattach for rainshadow-cec device on %I

[Service]
Type=simple
ExecStart=/usr/bin/inputattach --rainshadow-cec /dev/%I
```

For proper suspend/resume support create: /lib/systemd/system/restart-cec-inputattach.service:

```
[Unit]
Description=restart inputattach for cec devices
After=suspend.target

[Service]
Type=forking
ExecStart=/bin/bash -c 'for d in /dev/serial/by-id/usb-Pulse-Eight*; do /usr/
↪bin/inputattach --daemon --pulse8-cec $d; done; for d in /dev/serial/by-id/
↪usb-RainShadow_Tech*; do /usr/bin/inputattach --daemon --rainshadow-cec $d;␣
↪done'
```

```
[Install]
WantedBy=suspend.target
```

And run `systemctl enable restart-cec-inputattach`.

To automatically set the physical address of the CEC device whenever the EDID changes, you can use `cec-ctl` with the `-E` option:

```
cec-ctl -E /sys/class/drm/card0-DP-1/edid
```

This assumes the dongle is connected to the card0-DP-1 output (`xrandr` will tell you which output is used) and it will poll for changes to the EDID and update the Physical Address whenever they occur.

To automatically run this command you can use cron. Edit crontab with `crontab -e` and add this line:

```
@reboot /usr/local/bin/cec-ctl -E /sys/class/drm/card0-DP-1/edid
```

This only works for display drivers that expose the EDID in `/sys/class/drm`, such as the i915 driver.

59.4.5 CEC Without HPD

Some displays when in standby mode have no HDMI Hotplug Detect signal, but CEC is still enabled so connected devices can send an <Image View On> CEC message in order to wake up such displays. Unfortunately, not all CEC adapters can support this. An example is the Odroid-U3 SBC that has a level-shifter that is powered off when the HPD signal is low, thus blocking the CEC pin. Even though the SoC can use CEC without a HPD, the level-shifter will prevent this from functioning.

There is a CEC capability flag to signal this: `CEC_CAP_NEEDS_HPD`. If set, then the hardware cannot wake up displays with this behavior.

Note for CEC application implementers: the <Image View On> message must be the first message you send, don't send any other messages before. Certain very bad but unfortunately not uncommon CEC implementations get very confused if they receive anything else but this message and they won't wake up.

When writing a driver it can be tricky to test this. There are two ways to do this:

1) Get a Pulse-Eight USB CEC dongle, connect an HDMI cable from your device to the Pulse-Eight, but do not connect the Pulse-Eight to the display.

   Now configure the Pulse-Eight dongle:

   ```
   cec-ctl -p0.0.0.0 --tv
   ```

   and start monitoring:

   ```
   sudo cec-ctl -M
   ```

   On the device you are testing run:

```
cec-ctl --playback
```

It should report a physical address of f.f.f.f. Now run this command:

```
cec-ctl -t0 --image-view-on
```

The Pulse-Eight should see the <Image View On> message. If not, then something (hardware and/or software) is preventing the CEC message from going out.

To make sure you have the wiring correct just connect the Pulse-Eight to a CEC-enabled display and run the same command on your device: now there is a HPD, so you should see the command arriving at the Pulse-Eight.

2) If you have another linux device supporting CEC without HPD, then you can just connect your device to that device. Yes, you can connect two HDMI outputs together. You won't have a HPD (which is what we want for this test), but the second device can monitor the CEC pin.

   Otherwise use the same commands as in 1.

If CEC messages do not come through when there is no HPD, then you need to figure out why. Typically it is either a hardware restriction or the software powers off the CEC core when the HPD goes low. The first cannot be corrected of course, the second will likely required driver changes.

59.4.6 Microcontrollers & CEC

We have seen some CEC implementations in displays that use a microcontroller to sample the bus. This does not have to be a problem, but some implementations have timing issues. This is hard to discover unless you can hook up a low-level CEC debugger (see the next section).

You will see cases where the CEC transmitter holds the CEC line high or low for a longer time than is allowed. For directed messages this is not a problem since if that happens the message will not be Acked and it will be retransmitted. For broadcast messages no such mechanism exists.

It's not clear what to do about this. It is probably wise to transmit some broadcast messages twice to reduce the chance of them being lost. Specifically <Standby> and <Active Source> are candidates for that.

59.4.7 Making a CEC debugger

By using a Raspberry Pi 4B and some cheap components you can make your own low-level CEC debugger.

The critical component is one of these HDMI female-female passthrough connectors (full soldering type 1):

https://elabbay.myshopify.com/collections/camera/products/hdmi-af-af-v1a-hdmi-type-a-female-to-hd
variant=45533926147

The video quality is variable and certainly not enough to pass-through 4kp60 (594 MHz) video. You might be able to support 4kp30, but more likely you will be limited to 1080p60 (148.5 MHz). But for CEC testing that is fine.

You need a breadboard and some breadboard wires:

http://www.dx.com/p/diy-40p-male-to-female-male-to-male-female-to-female-dupont-line-wire-3pcs-3
.WYLOOXWGN7I

If you want to monitor the HPD and/or 5V lines as well, then you need one of these 5V to 3.3V level shifters:

https://www.adafruit.com/product/757

(This is just where I got these components, there are many other places you can get similar things).

The ground pin of the HDMI connector needs to be connected to a ground pin of the Raspberry Pi, of course.

The CEC pin of the HDMI connector needs to be connected to these pins: GPIO 6 and GPIO 7. The optional HPD pin of the HDMI connector should be connected via the level shifter to these pins: GPIO 23 and GPIO 12. The optional 5V pin of the HDMI connector should be connected via the level shifter to these pins: GPIO 25 and GPIO 22. Monitoring the HPD and 5V lines is not necessary, but it is helpful.

This device tree addition in `arch/arm/boot/dts/bcm2711-rpi-4-b.dts` will hook up the cec-gpio driver correctly:

```
cec@6 {
        compatible = "cec-gpio";
        cec-gpios = <&gpio 6 (GPIO_ACTIVE_HIGH|GPIO_OPEN_DRAIN)>;
        hpd-gpios = <&gpio 23 GPIO_ACTIVE_HIGH>;
        v5-gpios = <&gpio 25 GPIO_ACTIVE_HIGH>;
};

cec@7 {
        compatible = "cec-gpio";
        cec-gpios = <&gpio 7 (GPIO_ACTIVE_HIGH|GPIO_OPEN_DRAIN)>;
        hpd-gpios = <&gpio 12 GPIO_ACTIVE_HIGH>;
        v5-gpios = <&gpio 22 GPIO_ACTIVE_HIGH>;
};
```

If you haven't hooked up the HPD and/or 5V lines, then just delete those lines.

This dts change will enable two cec GPIO devices: I typically use one to send/receive CEC commands and the other to monitor. If you monitor using an unconfigured CEC adapter then it will use GPIO interrupts which makes monitoring very accurate.

If you just want to monitor traffic, then a single instance is sufficient. The minimum configuration is one HDMI female-female passthrough connector and two female-female breadboard wires: one for connecting the HDMI ground pin to a ground pin on the Raspberry Pi, and the other to connect the HDMI CEC pin to GPIO 6 on the Raspberry Pi.

The documentation on how to use the error injection is here: cec_pin_error_inj.

`cec-ctl --monitor-pin` will do low-level CEC bus sniffing and analysis. You can also store the CEC traffic to file using `--store-pin` and analyze it later using `--analyze-pin`.

You can also use this as a full-fledged CEC device by configuring it using `cec-ctl --tv -p0.0.0.0` or `cec-ctl --playback -p1.0.0.0`.

59.5 Digital TV

59.5.1 Using the Digital TV Framework

Introduction

One significant difference between Digital TV and Analogue TV that the unwary (like myself) should consider is that, although the component structure of DVB-T cards are substantially similar to Analogue TV cards, they function in substantially different ways.

The purpose of an Analogue TV is to receive and display an Analogue Television signal. An Analogue TV signal (otherwise known as composite video) is an analogue encoding of a sequence of image frames (25 frames per second in Europe) rasterised using an interlacing technique. Interlacing takes two fields to represent one frame. Therefore, an Analogue TV card for a PC has the following purpose:

- Tune the receiver to receive a broadcast signal
- demodulate the broadcast signal
- demultiplex the analogue video signal and analogue audio signal.

  Note: some countries employ a digital audio signal embedded within the modulated composite analogue signal - using NICAM signaling.)

- digitize the analogue video signal and make the resulting datastream available to the data bus.

The digital datastream from an Analogue TV card is generated by circuitry on the card and is often presented uncompressed. For a PAL TV signal encoded at a resolution of 768x576 24-bit color pixels over 25 frames per second - a fair amount of data is generated and must be processed by the PC before it can be displayed on the video monitor screen. Some Analogue TV cards for PCs have onboard MPEG2 encoders which permit the raw digital data stream to be presented to the PC in an encoded and compressed form - similar to the form that is used in Digital TV.

The purpose of a simple budget digital TV card (DVB-T,C or S) is to simply:

- Tune the received to receive a broadcast signal. * Extract the encoded digital datastream from the broadcast signal.
- Make the encoded digital datastream (MPEG2) available to the data bus.

The significant difference between the two is that the tuner on the analogue TV card spits out an Analogue signal, whereas the tuner on the digital TV card spits out a compressed encoded digital datastream. As the signal is already digitised, it is trivial to pass this datastream to the PC databus with minimal additional processing and then extract the digital video and audio datastreams passing them to the appropriate software or hardware for decoding and viewing.

Getting the card going

The Device Driver API for DVB under Linux will the following device nodes via the devfs filesystem:

- /dev/dvb/adapter0/demux0
- /dev/dvb/adapter0/dvr0
- /dev/dvb/adapter0/frontend0

The `/dev/dvb/adapter0/dvr0` device node is used to read the MPEG2 Data Stream and the `/dev/dvb/adapter0/frontend0` device node is used to tune the frontend tuner module. The `/dev/dvb/adapter0/demux0` is used to control what programs will be received.

Depending on the card's feature set, the Device Driver API could also expose other device nodes:

- /dev/dvb/adapter0/ca0
- /dev/dvb/adapter0/audio0
- /dev/dvb/adapter0/net0
- /dev/dvb/adapter0/osd0
- /dev/dvb/adapter0/video0

The `/dev/dvb/adapter0/ca0` is used to decode encrypted channels. The other device nodes are found only on devices that use the av7110 driver, with is now obsoleted, together with the extra API whose such devices use.

Receiving a digital TV channel

This section attempts to explain how it works and how this affects the configuration of a Digital TV card.

On this example, we're considering tuning into DVB-T channels in Australia, at the Melbourne region.

The frequencies broadcast by Mount Dandenong transmitters are, currently:

Table 1. Transponder Frequencies Mount Dandenong, Vic, Aus.

Broadcaster	Frequency
Seven	177.500 Mhz
SBS	184.500 Mhz
Nine	191.625 Mhz
Ten	219.500 Mhz
ABC	226.500 Mhz
Channel 31	557.625 Mhz

The digital TV Scan utilities (like dvbv5-scan) have use a set of compiled-in defaults for various countries and regions. Those are currently provided as a separate package, called dtv-scan-tables. It's git tree is located at LinuxTV.org:

https://git.linuxtv.org/dtv-scan-tables.git/

If none of the tables there suit, you can specify a data file on the command line which contains the transponder frequencies. Here is a sample file for the above channel transponders, in the old ”channel” format:

```
# Data file for DVB scan program
#
# C Frequency SymbolRate FEC QAM
# S Frequency Polarisation SymbolRate FEC
# T Frequency Bandwidth FEC FEC2 QAM Mode Guard Hier

T 177500000 7MHz AUTO AUTO QAM64 8k 1/16 NONE
T 184500000 7MHz AUTO AUTO QAM64 8k 1/8 NONE
T 191625000 7MHz AUTO AUTO QAM64 8k 1/16 NONE
T 219500000 7MHz AUTO AUTO QAM64 8k 1/16 NONE
T 226500000 7MHz AUTO AUTO QAM64 8k 1/16 NONE
T 557625000 7MHz AUTO AUTO QPSK 8k 1/16 NONE
```

Nowadays, we prefer to use a newer format, with is more verbose and easier to understand. With the new format, the ”Seven” channel transponder's data is represented by:

```
[Seven]
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 177500000
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = AUTO
        CODE_RATE_LP = AUTO
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/16
        HIERARCHY = NONE
        INVERSION = AUTO
```

For an updated version of the complete table, please see:

https://git.linuxtv.org/dtv-scan-tables.git/tree/dvb-t/au-Melbourne

When the Digital TV scanning utility runs, it will output a file containing the information for all the audio and video programs that exists into each channel's transponders which the card's frontend can lock onto. (i.e. any whose signal is strong enough at your antenna).

Here's the output of the dvbv5 tools from a channel scan took from Melburne:

```
[ABC HDTV]
        SERVICE_ID = 560
        VIDEO_PID = 2307
        AUDIO_PID = 0
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 226500000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 3/4
        CODE_RATE_LP = 3/4
        MODULATION = QAM/64
```

```
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/16
        HIERARCHY = NONE

[ABC TV Melbourne]
        SERVICE_ID = 561
        VIDEO_PID = 512
        AUDIO_PID = 650
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 226500000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 3/4
        CODE_RATE_LP = 3/4
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/16
        HIERARCHY = NONE

[ABC TV 2]
        SERVICE_ID = 562
        VIDEO_PID = 512
        AUDIO_PID = 650
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 226500000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 3/4
        CODE_RATE_LP = 3/4
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/16
        HIERARCHY = NONE

[ABC TV 3]
        SERVICE_ID = 563
        VIDEO_PID = 512
        AUDIO_PID = 650
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 226500000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 3/4
        CODE_RATE_LP = 3/4
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/16
        HIERARCHY = NONE

[ABC TV 4]
```

```
        SERVICE_ID = 564
        VIDEO_PID = 512
        AUDIO_PID = 650
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 226500000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 3/4
        CODE_RATE_LP = 3/4
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/16
        HIERARCHY = NONE

[ABC DiG Radio]
        SERVICE_ID = 566
        VIDEO_PID = 0
        AUDIO_PID = 2311
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 226500000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 3/4
        CODE_RATE_LP = 3/4
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/16
        HIERARCHY = NONE

[TEN Digital]
        SERVICE_ID = 1585
        VIDEO_PID = 512
        AUDIO_PID = 650
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 219500000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 3/4
        CODE_RATE_LP = 1/2
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/16
        HIERARCHY = NONE

[TEN Digital 1]
        SERVICE_ID = 1586
        VIDEO_PID = 512
        AUDIO_PID = 650
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 219500000
```

```
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 3/4
        CODE_RATE_LP = 1/2
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/16
        HIERARCHY = NONE

[TEN Digital 2]
        SERVICE_ID = 1587
        VIDEO_PID = 512
        AUDIO_PID = 650
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 219500000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 3/4
        CODE_RATE_LP = 1/2
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/16
        HIERARCHY = NONE

[TEN Digital 3]
        SERVICE_ID = 1588
        VIDEO_PID = 512
        AUDIO_PID = 650
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 219500000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 3/4
        CODE_RATE_LP = 1/2
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/16
        HIERARCHY = NONE

[TEN Digital]
        SERVICE_ID = 1589
        VIDEO_PID = 512
        AUDIO_PID = 650
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 219500000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 3/4
        CODE_RATE_LP = 1/2
        MODULATION = QAM/64
```

```
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/16
        HIERARCHY = NONE

[TEN Digital 4]
        SERVICE_ID = 1590
        VIDEO_PID = 512
        AUDIO_PID = 650
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 219500000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 3/4
        CODE_RATE_LP = 1/2
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/16
        HIERARCHY = NONE

[TEN Digital]
        SERVICE_ID = 1591
        VIDEO_PID = 512
        AUDIO_PID = 650
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 219500000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 3/4
        CODE_RATE_LP = 1/2
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/16
        HIERARCHY = NONE

[TEN HD]
        SERVICE_ID = 1592
        VIDEO_PID = 514
        AUDIO_PID = 0
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 219500000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 3/4
        CODE_RATE_LP = 1/2
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/16
        HIERARCHY = NONE

[TEN Digital]
```

```
        SERVICE_ID = 1593
        VIDEO_PID = 512
        AUDIO_PID = 650
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 219500000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 3/4
        CODE_RATE_LP = 1/2
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/16
        HIERARCHY = NONE

[Nine Digital]
        SERVICE_ID = 1072
        VIDEO_PID = 513
        AUDIO_PID = 660
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 191625000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 3/4
        CODE_RATE_LP = 1/2
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/16
        HIERARCHY = NONE

[Nine Digital HD]
        SERVICE_ID = 1073
        VIDEO_PID = 512
        AUDIO_PID = 0
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 191625000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 3/4
        CODE_RATE_LP = 1/2
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/16
        HIERARCHY = NONE

[Nine Guide]
        SERVICE_ID = 1074
        VIDEO_PID = 514
        AUDIO_PID = 670
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 191625000
```

```
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 3/4
        CODE_RATE_LP = 1/2
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/16
        HIERARCHY = NONE

[7 Digital]
        SERVICE_ID = 1328
        VIDEO_PID = 769
        AUDIO_PID = 770
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 177500000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 2/3
        CODE_RATE_LP = 2/3
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/8
        HIERARCHY = NONE

[7 Digital 1]
        SERVICE_ID = 1329
        VIDEO_PID = 769
        AUDIO_PID = 770
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 177500000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 2/3
        CODE_RATE_LP = 2/3
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/8
        HIERARCHY = NONE

[7 Digital 2]
        SERVICE_ID = 1330
        VIDEO_PID = 769
        AUDIO_PID = 770
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 177500000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 2/3
        CODE_RATE_LP = 2/3
        MODULATION = QAM/64
```

```
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/8
        HIERARCHY = NONE

[7 Digital 3]
        SERVICE_ID = 1331
        VIDEO_PID = 769
        AUDIO_PID = 770
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 177500000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 2/3
        CODE_RATE_LP = 2/3
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/8
        HIERARCHY = NONE

[7 HD Digital]
        SERVICE_ID = 1332
        VIDEO_PID = 833
        AUDIO_PID = 834
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 177500000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 2/3
        CODE_RATE_LP = 2/3
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/8
        HIERARCHY = NONE

[7 Program Guide]
        SERVICE_ID = 1334
        VIDEO_PID = 865
        AUDIO_PID = 866
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 177500000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 2/3
        CODE_RATE_LP = 2/3
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/8
        HIERARCHY = NONE

[SBS HD]
```

```
        SERVICE_ID = 784
        VIDEO_PID = 102
        AUDIO_PID = 103
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 536500000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 2/3
        CODE_RATE_LP = 2/3
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/8
        HIERARCHY = NONE

[SBS DIGITAL 1]
        SERVICE_ID = 785
        VIDEO_PID = 161
        AUDIO_PID = 81
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 536500000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 2/3
        CODE_RATE_LP = 2/3
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/8
        HIERARCHY = NONE

[SBS DIGITAL 2]
        SERVICE_ID = 786
        VIDEO_PID = 162
        AUDIO_PID = 83
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 536500000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 2/3
        CODE_RATE_LP = 2/3
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/8
        HIERARCHY = NONE

[SBS EPG]
        SERVICE_ID = 787
        VIDEO_PID = 163
        AUDIO_PID = 85
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 536500000
```

```
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 2/3
        CODE_RATE_LP = 2/3
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/8
        HIERARCHY = NONE

[SBS RADIO 1]
        SERVICE_ID = 798
        VIDEO_PID = 0
        AUDIO_PID = 201
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 536500000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 2/3
        CODE_RATE_LP = 2/3
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/8
        HIERARCHY = NONE

[SBS RADIO 2]
        SERVICE_ID = 799
        VIDEO_PID = 0
        AUDIO_PID = 202
        DELIVERY_SYSTEM = DVBT
        FREQUENCY = 536500000
        INVERSION = OFF
        BANDWIDTH_HZ = 7000000
        CODE_RATE_HP = 2/3
        CODE_RATE_LP = 2/3
        MODULATION = QAM/64
        TRANSMISSION_MODE = 8K
        GUARD_INTERVAL = 1/8
        HIERARCHY = NONE
```

59.5.2 Digital TV Conditional Access Interface

Note: This documentation is outdated.

This document describes the usage of the high level CI API as in accordance to the Linux DVB API. This is a not a documentation for the, existing low level CI API.

Note: For the Twinhan/Twinhan clones, the dst_ca module handles the CI hardware handling. This module is loaded automatically if a CI (Common Interface, that holds the CAM (Conditional

Access Module) is detected.

ca_zap

A userspace application, like `ca_zap` is required to handle encrypted MPEG-TS streams.

The `ca_zap` userland application is in charge of sending the descrambling related information to the Conditional Access Module (CAM).

This application requires the following to function properly as of now.

a) Tune to a valid channel, with szap.

   eg: $ szap -c channels.conf -r "TMC" -x

b) a channels.conf containing a valid PMT PID

   eg: TMC:11996:h:0:27500:278:512:650:321

   here 278 is a valid PMT PID. the rest of the values are the same ones that szap uses.

c) after running a szap, you have to run ca_zap, for the descrambler to function,

   eg: $ ca_zap channels.conf "TMC"

d) Hopefully enjoy your favourite subscribed channel as you do with a FTA card.

Note: Currently ca_zap, and dst_test, both are meant for demonstration purposes only, they can become full fledged applications if necessary.

Cards that fall in this category

At present the cards that fall in this category are the Twinhan and its clones, these cards are available as VVMER, Tomato, Hercules, Orange and so on.

CI modules that are supported

The CI module support is largely dependent upon the firmware on the cards Some cards do support almost all of the available CI modules. There is nothing much that can be done in order to make additional CI modules working with these cards.

Modules that have been tested by this driver at present are

(1) Irdeto 1 and 2 from SCM

(2) Viaccess from SCM

(3) Dragoncam

59.5.3 FAQ

Note:

1. With Digital TV, a single physical channel may have different contents inside it. The specs call each one as a *service*. This is what a TV user would call ”channel”. So, in order to avoid confusion, we're calling *transponders* as the physical channel on this FAQ, and *services* for the logical channel.
2. The LinuxTV community maintains some Wiki pages with contain a lot of information related to the media subsystem. If you don't find an answer for your needs here, it is likely that you'll be able to get something useful there. It is hosted at:

   https://www.linuxtv.org/wiki/

Some very frequently asked questions about Linux Digital TV support

1. The signal seems to die a few seconds after tuning.

   It's not a bug, it's a feature. Because the frontends have significant power requirements (and hence get very hot), they are powered down if they are unused (i.e. if the frontend device is closed). The `dvb-core` module parameter `dvb_shutdown_timeout` allow you to change the timeout (default 5 seconds). Setting the timeout to 0 disables the timeout feature.

2. How can I watch TV?

   Together with the Linux Kernel, the Digital TV developers support some simple utilities which are mainly intended for testing and to demonstrate how the DVB API works. This is called DVB v5 tools and are grouped together with the `v4l-utils` git repository:

   https://git.linuxtv.org/v4l-utils.git/

   You can find more information at the LinuxTV wiki:

   https://www.linuxtv.org/wiki/index.php/DVBv5_Tools

   The first step is to get a list of services that are transmitted.

   This is done by using several existing tools. You can use for example the `dvbv5-scan` tool. You can find more information about it at:

   https://www.linuxtv.org/wiki/index.php/Dvbv5-scan

   There are some other applications like `w_scan`[1] that do a blind scan, trying hard to find all possible channels, but those consumes a large amount of time to run.

   Also, some applications like `kaffeine` have their own code to scan for services. So, you don't need to use an external application to obtain such list.

   Most of such tools need a file containing a list of channel transponders available on your area. So, LinuxTV developers maintain tables of Digital TV channel transponders, receiving patches from the community to keep them updated.

   This list is hosted at:

[1] https://www.linuxtv.org/wiki/index.php/W_scan

https://git.linuxtv.org/dtv-scan-tables.git

And packaged on several distributions.

Kaffeine has some blind scan support for some terrestrial standards. It also relies on DTV scan tables, although it contains a copy of it internally (and, if requested by the user, it will download newer versions of it).

If you are lucky you can just use one of the supplied channel transponders. If not, you may need to seek for such info at the Internet and create a new file. There are several sites with contains physical channel lists. For cable and satellite, usually knowing how to tune into a single channel is enough for the scanning tool to identify the other channels. On some places, this could also work for terrestrial transmissions.

Once you have a transponders list, you need to generate a services list with a tool like `dvbv5-scan`.

Almost all modern Digital TV cards don't have built-in hardware MPEG-decoders. So, it is up to the application to get a MPEG-TS stream provided by the board, split it into audio, video and other data and decode.

3. Which Digital TV applications exist?

   Several media player applications are capable of tuning into digital TV channels, including Kaffeine, Vlc, mplayer and MythTV.

   Kaffeine aims to be very user-friendly, and it is maintained by one of the Kernel driver developers.

   A comprehensive list of those and other apps can be found at:

   https://www.linuxtv.org/wiki/index.php/TV_Related_Software

   Some of the most popular ones are linked below:

   https://kde.org/applications/multimedia/org.kde.kaffeine
   KDE media player, focused on Digital TV support

   https://www.linuxtv.org/vdrwiki/index.php/Main_Page
   Klaus Schmidinger's Video Disk Recorder

   https://linuxtv.org/downloads and https://git.linuxtv.org/
   Digital TV and other media-related applications and Kernel drivers. The `v4l-utils` package there contains several swiss knife tools for using with Digital TV.

   http://sourceforge.net/projects/dvbtools/
   Dave Chapman's dvbtools package, including dvbstream and dvbtune

   http://www.dbox2.info/
   LinuxDVB on the dBox2

   http://www.tuxbox.org/
   the TuxBox CVS many interesting DVB applications and the dBox2 DVB source

   http://www.nenie.org/misc/mpsys/
   MPSYS: a MPEG2 system library and tools

   https://www.videolan.org/vlc/index.pt.html
   Vlc

http://mplayerhq.hu/
MPlayer

http://xine.sourceforge.net/ and http://xinehq.de/
Xine

http://www.mythtv.org/
MythTV - analog TV and digital TV PVR

http://dvbsnoop.sourceforge.net/
DVB sniffer program to monitor, analyze, debug, dump or view dvb/mpeg/dsm-cc/mhp stream information (TS, PES, SECTION)

4. Can't get a signal tuned correctly

   That could be due to a lot of problems. On my personal experience, usually TV cards need stronger signals than TV sets, and are more sensitive to noise. So, perhaps you just need a better antenna or cabling. Yet, it could also be some hardware or driver issue.

   For example, if you are using a Technotrend/Hauppauge DVB-C card *without* analog module, you might have to use module parameter adac=-1 (dvb-ttpci.o).

   Please see the FAQ page at linuxtv.org, as it could contain some valuable information:

   https://www.linuxtv.org/wiki/index.php/FAQ_%26_Troubleshooting

   If that doesn't work, check at the linux-media ML archives, to see if someone else had a similar problem with your hardware and/or digital TV service provider:

   https://lore.kernel.org/linux-media/

   If none of this works, you can try sending an e-mail to the linux-media ML and see if someone else could shed some light. The e-mail is linux-media AT vger.kernel.org.

5. The dvb_net device doesn't give me any packets at all

   Run `tcpdump` on the `dvb0_0` interface. This sets the interface into promiscuous mode so it accepts any packets from the PID you have configured with the `dvbnet` utility. Check if there are any packets with the IP addr and MAC addr you have configured with `ifconfig` or with `ip addr`.

   If `tcpdump` doesn't give you any output, check the statistics which `ifconfig` or `netstat -ni` outputs. (Note: If the MAC address is wrong, `dvb_net` won't get any input; thus you have to run `tcpdump` before checking the statistics.) If there are no packets at all then maybe the PID is wrong. If there are error packets, then either the PID is wrong or the stream does not conform to the MPE standard (EN 301 192, http://www.etsi.org/). You can use e.g. `dvbsnoop` for debugging.

6. The `dvb_net` device doesn't give me any multicast packets

   Check your routes if they include the multicast address range. Additionally make sure that "source validation by reversed path lookup" is disabled:

   ```
   $ "echo 0 > /proc/sys/net/ipv4/conf/dvb0/rp_filter"
   ```

7. What are all those modules that need to be loaded?

In order to make it more flexible and support different hardware combinations, the media subsystem is written on a modular way.

So, besides the Digital TV hardware module for the main chipset, it also needs to load a frontend driver, plus the Digital TV core. If the board also has remote controller, it will also need the remote controller core and the remote controller tables. The same happens if the board has support for analog TV: the core support for video4linux need to be loaded.

The actual module names are Linux-kernel version specific, as, from time to time, things change, in order to make the media support more flexible.

59.5.4 References

The main development site and GIT repository for Digital TV drivers is https://linuxtv.org.

The DVB mailing list linux-dvb is hosted at vger. Please see http://vger.kernel.org/vger-lists.html#linux-media for details.

There are also some other old lists hosted at: https://linuxtv.org/lists.php. If you're interested on that for historic reasons, please check the archive at https://linuxtv.org/pipermail/linux-dvb/.

The media subsystem Wiki is hosted at https://linuxtv.org/wiki/. There, you'll find lots of information, from both development and usage of media boards. Please check it before asking newbie questions on the mailing list or IRC channels.

The API documentation is documented at the Kernel tree. You can find it in both html and pdf formats, together with other useful documentation at:

- https://linuxtv.org/docs.php.

You may also find useful material at https://linuxtv.org/downloads/.

In order to get the needed firmware for some drivers to work, there's a script at the kernel tree, at scripts/get_dvb_firmware.

59.6 Cards List

The media subsystem provide support for lots of PCI and USB drivers, plus platform-specific drivers. It also contains several ancillary I^2C drivers.

The platform-specific drivers are usually present on embedded systems, or are supported by the main board. Usually, setting them is done via OpenFirmware or ACPI.

The PCI and USB drivers, however, are independent of the system's board, and may be added/removed by the user.

You may also take a look at https://linuxtv.org/wiki/index.php/Hardware_Device_Information for more details about supported cards.

59.6.1 USB drivers

The USB boards are identified by an identification called USB ID.

The `lsusb` command allows identifying the USB IDs:

```
$ lsusb
...
Bus 001 Device 015: ID 046d:082d Logitech, Inc. HD Pro Webcam C920
Bus 001 Device 074: ID 2040:b131 Hauppauge
Bus 001 Device 075: ID 2013:024f PCTV Systems nanoStick T2 290e
...
```

Newer camera devices use a standard way to expose themselves as such, via USB Video Class. Those cameras are automatically supported by the `uvc-driver`.

Older cameras and TV USB devices uses USB Vendor Classes: each vendor defines its own way to access the device. This section contains card lists for such vendor-class devices.

While this is not as common as on PCI, sometimes the same USB ID is used by different products. So, several media drivers allow passing a `card=` parameter, in order to setup a card number that would match the correct settings for an specific product type.

The current supported USB cards (not including staging drivers) are listed below[1].

Driver	Name
airspy	AirSpy
au0828	Auvitek AU0828
b2c2-flexcop-usb	Technisat/B2C2 Air/Sky/Cable2PC USB
cx231xx	Conexant cx231xx USB video capture
dvb-as102	Abilis AS102 DVB receiver
dvb-ttusb-budget	Technotrend/Hauppauge Nova - USB devices
dvb-usb-a800	AVerMedia AverTV DVB-T USB 2.0 (A800)
dvb-usb-af9005	Afatech AF9005 DVB-T USB1.1
dvb-usb-af9015	Afatech AF9015 DVB-T USB2.0
dvb-usb-af9035	Afatech AF9035 DVB-T USB2.0
dvb-usb-anysee	Anysee DVB-T/C USB2.0
dvb-usb-au6610	Alcor Micro AU6610 USB2.0
dvb-usb-az6007	AzureWave 6007 and clones DVB-T/C USB2.0
dvb-usb-az6027	Azurewave DVB-S/S2 USB2.0 AZ6027
dvb-usb-ce6230	Intel CE6230 DVB-T USB2.0
dvb-usb-cinergyT2	Terratec CinergyT2/qanu USB 2.0 DVB-T
dvb-usb-cxusb	Conexant USB2.0 hybrid
dvb-usb-dib0700	DiBcom DiB0700
dvb-usb-dibusb-common	DiBcom DiB3000M-B
dvb-usb-dibusb-mc	DiBcom DiB3000M-C/P
dvb-usb-digitv	Nebula Electronics uDigiTV DVB-T USB2.0
dvb-usb-dtt200u	WideView WT-200U and WT-220U (pen) DVB-T
dvb-usb-dtv5100	AME DTV-5100 USB2.0 DVB-T

continues on next page

[1] some of the drivers have sub-drivers, not shown at this table. In particular, gspca driver has lots of sub-drivers, for cameras not supported by the USB Video Class (UVC) driver, as shown at *gspca card list*.

Table 1 – continued from previous page

Driver	Name
dvb-usb-dvbsky	DVBSky USB
dvb-usb-dw2102	DvbWorld & TeVii DVB-S/S2 USB2.0
dvb-usb-ec168	E3C EC168 DVB-T USB2.0
dvb-usb-gl861	Genesys Logic GL861 USB2.0
dvb-usb-gp8psk	GENPIX 8PSK->USB module
dvb-usb-lmedm04	LME DM04/QQBOX DVB-S USB2.0
dvb-usb-m920x	Uli m920x DVB-T USB2.0
dvb-usb-nova-t-usb2	Hauppauge WinTV-NOVA-T usb2 DVB-T USB2.0
dvb-usb-opera	Opera1 DVB-S USB2.0 receiver
dvb-usb-pctv452e	Pinnacle PCTV HDTV Pro USB device/TT Connect S2-3600
dvb-usb-rtl28xxu	Realtek RTL28xxU DVB USB
dvb-usb-technisat-usb2	Technisat DVB-S/S2 USB2.0
dvb-usb-ttusb2	Pinnacle 400e DVB-S USB2.0
dvb-usb-umt-010	HanfTek UMT-010 DVB-T USB2.0
dvb_usb_v2	Support for various USB DVB devices v2
dvb-usb-vp702x	TwinhanDTV StarBox and clones DVB-S USB2.0
dvb-usb-vp7045	TwinhanDTV Alpha/MagicBoxII, DNTV tinyUSB2, Beetle USB2.0
em28xx	Empia EM28xx USB devices
go7007	WIS GO7007 MPEG encoder
gspca	Drivers for several USB Cameras
hackrf	HackRF
hdpvr	Hauppauge HD PVR
msi2500	Mirics MSi2500
mxl111sf-tuner	MxL111SF DTV USB2.0
pvrusb2	Hauppauge WinTV-PVR USB2
pwc	USB Philips Cameras
s2250	Sensoray 2250/2251
s2255drv	USB Sensoray 2255 video capture device
smsusb	Siano SMS1xxx based MDTV receiver
ttusb_dec	Technotrend/Hauppauge USB DEC devices
usbtv	USBTV007 video capture
uvcvideo	USB Video Class (UVC)
zd1301	ZyDAS ZD1301

AU0828 cards list

Card number	Card name	USB IDs
0	Unknown board	
1	Hauppauge HVR950Q	2040:7200, 2040:7210, 2040:7217, 2040:721b, 2040:721e, 2040:721f, 2040:7280, 0fd9:0008, 2040:7260, 2040:7213, 2040:7270
2	Hauppauge HVR850	2040:7240
3	DViCO FusionHDTV USB	0fe9:d620
4	Hauppauge HVR950Q rev xxF8	2040:7201, 2040:7211, 2040:7281
5	Hauppauge Woodbury	05e1:0480, 2040:8200

cx231xx cards list

Card number	Card name	USB IDs
0	Unknown CX231xx video grabber	0572:5A3C
1	Conexant Hybrid TV - CARRAERA	0572:58A2
2	Conexant Hybrid TV - SHELBY	0572:58A1
3	Conexant Hybrid TV - RDE253S	0572:58A4
4	Conexant Hybrid TV - RDU253S	0572:58A5
5	Conexant VIDEO GRABBER	0572:58A6, 07ca:c039
6	Conexant Hybrid TV - rde 250	0572:589E
7	Conexant Hybrid TV - RDU 250	0572:58A0
8	Hauppauge EXETER	2040:b120, 2040:b140
9	Hauppauge USB Live 2	2040:c200
10	Pixelview PlayTV USB Hybrid	4000:4001
11	Pixelview Xcapture USB	1D19:6109, 4000:4001
12	Kworld UB430 USB Hybrid	1b80:e424
13	Iconbit Analog Stick U100 FM	1f4d:0237
14	Hauppauge WinTV USB2 FM (PAL)	2040:b110
15	Hauppauge WinTV USB2 FM (NTSC)	2040:b111
16	Elgato Video Capture V2	0fd9:0037
17	Geniatech OTG102	1f4d:0102
18	Kworld UB445 USB Hybrid	1b80:e421
19	Hauppauge WinTV 930C-HD (1113xx) / HVR-900H (111xxx) / PCTV QuatroStick 521e	2040:b130, 2040:b138, 2013:0259
20	Hauppauge WinTV 930C-HD (1114xx) / HVR-901H (1114xx) / PCTV QuatroStick 522e	2040:b131, 2040:b139, 2013:025e
21	Hauppauge WinTV-HVR-955Q (111401)	2040:b123, 2040:b124
22	Terratec Grabby	1f4d:0102
23	Evromedia USB Full Hybrid Full HD	1b80:d3b2
24	Astrometa T2hybrid	15f4:0135
25	The Imaging Source DFG/USB2pro	199e:8002
26	Hauppauge WinTV-HVR-935C	2040:b151
27	Hauppauge WinTV-HVR-975	2040:b150

EM28xx cards list

Card number	Card name	Empia Chip	USB IDs
0	Unknown EM2800 video grabber	em2800	eb1a:2800

continues on next page

Table 2 – continued from previous page

Card number	Card name	Empia Chip	USB IDs
1	Unknown EM2750/28xx video grabber	em2820 or em2840	eb1a:2710, eb1a:2820, eb1a:2821, eb1a:2860, eb1a:2861, eb1a:2862, eb1a:2863, eb1a:2870, eb1a:2881, eb1a:2883, eb1a:2868, eb1a:2875
2	Terratec Cinergy 250 USB	em2820 or em2840	0ccd:0036
3	Pinnacle PCTV USB 2	em2820 or em2840	2304:0208
4	Hauppauge WinTV USB 2	em2820 or em2840	2040:4200, 2040:4201
5	MSI VOX USB 2.0	em2820 or em2840	
6	Terratec Cinergy 200 USB	em2800	
7	Leadtek Winfast USB II	em2800	0413:6023
8	Kworld USB2800	em2800	
9	Pinnacle Dazzle DVC 90/100/101/107 / Kaiser Baas Video to DVD maker / Kworld DVD Maker 2 / Plextor ConvertX PX-AV100U	em2820 or em2840	1b80:e302, 1b80:e304, 2304:0207, 2304:021a, 093b:a003
10	Hauppauge WinTV HVR 900	em2880	2040:6500
11	Terratec Hybrid XS	em2880	
12	Kworld PVR TV 2800 RF	em2820 or em2840	
13	Terratec Prodigy XS	em2880	
14	SIIG AVTuner-PVR / Pixelview Prolink PlayTV USB 2.0	em2820 or em2840	
15	V-Gear PocketTV	em2800	
16	Hauppauge WinTV HVR 950	em2883	2040:6513, 2040:6517, 2040:651b
17	Pinnacle PCTV HD Pro Stick	em2880	2304:0227

continues on next page

Table 2 – continued from previous page

Card number	Card name	Empia Chip	USB IDs
18	Hauppauge WinTV HVR 900 (R2)	em2880	2040:6502
19	EM2860/SAA711X Reference Design	em2860	
20	AMD ATI TV Wonder HD 600	em2880	0438:b002
21	eMPIA Technology, Inc. GrabBeeX+ Video Encoder	em2800	eb1a:2801
22	EM2710/EM2750/EM2751 webcam grabber	em2750	eb1a:2750, eb1a:2751
23	Huaqi DLCW-130	em2750	
24	D-Link DUB-T210 TV Tuner	em2820 or em2840	2001:f112
25	Gadmei UTV310	em2820 or em2840	
26	Hercules Smart TV USB 2.0	em2820 or em2840	
27	Pinnacle PCTV USB 2 (Philips FM1216ME)	em2820 or em2840	
28	Leadtek Winfast USB II Deluxe	em2820 or em2840	
29	EM2860/TVP5150 Reference Design	em2860	eb1a:5051
30	Videology 20K14XUSB USB2.0	em2820 or em2840	
31	Usbgear VD204v9	em2821	
32	Supercomp USB 2.0 TV	em2821	
33	Elgato Video Capture	em2860	0fd9:0033
34	Terratec Cinergy A Hybrid XS	em2860	0ccd:004f
35	Typhoon DVD Maker	em2860	
36	NetGMBH Cam	em2860	
37	Gadmei UTV330	em2860	eb1a:50a6
38	Yakumo MovieMixer	em2861	
39	KWorld PVRTV 300U	em2861	eb1a:e300
40	Plextor ConvertX PX-TV100U	em2861	093b:a005
41	Kworld 350 U DVB-T	em2870	eb1a:e350
42	Kworld 355 U DVB-T	em2870	eb1a:e355, eb1a:e357, eb1a:e359
43	Terratec Cinergy T XS	em2870	
44	Terratec Cinergy T XS (MT2060)	em2870	0ccd:0043
45	Pinnacle PCTV DVB-T	em2870	
46	Compro, VideoMate U3	em2870	185b:2870
47	KWorld DVB-T 305U	em2880	eb1a:e305

continues on next page

Table 2 – continued from previous page

Card number	Card name	Empia Chip	USB IDs
48	KWorld DVB-T 310U	em2880	
49	MSI DigiVox A/D	em2880	eb1a:e310
50	MSI DigiVox A/D II	em2880	eb1a:e320
51	Terratec Hybrid XS Secam	em2880	0ccd:004c
52	DNT DA2 Hybrid	em2881	
53	Pinnacle Hybrid Pro	em2881	
54	Kworld VS-DVB-T 323UR	em2882	eb1a:e323
55	Terratec Cinergy Hybrid T USB XS (em2882)	em2882	0ccd:005e, 0ccd:0042
56	Pinnacle Hybrid Pro (330e)	em2882	2304:0226
57	Kworld PlusTV HD Hybrid 330	em2883	eb1a:a316
58	Compro VideoMate ForYou/Stereo	em2820 or em2840	185b:2041
59	Pinnacle PCTV HD Mini	em2874	2304:023f
60	Hauppauge WinTV HVR 850	em2883	2040:651f
61	Pixelview PlayTV Box 4 USB 2.0	em2820 or em2840	
62	Gadmei TVR200	em2820 or em2840	
63	Kaiomy TVnPC U2	em2860	eb1a:e303
64	Easy Cap Capture DC-60	em2860	1b80:e309
65	IO-DATA GV-MVP/SZ	em2820 or em2840	04bb:0515
66	Empire dual TV	em2880	
67	Terratec Grabby	em2860	0ccd:0096, 0ccd:10AF
68	Terratec AV350	em2860	0ccd:0084
69	KWorld ATSC 315U HDTV TV Box	em2882	eb1a:a313
70	Evga inDtube	em2882	
71	Silvercrest Webcam 1.3mpix	em2820 or em2840	
72	Gadmei UTV330+	em2861	
73	Reddo DVB-C USB TV Box	em2870	
74	Actionmaster/LinXcel/Digitus VC211A	em2800	
75	Dikom DK300	em2882	
76	KWorld PlusTV 340U or UB435-Q (ATSC)	em2870	1b80:a340
77	EM2874 Leadership ISDBT	em2874	
78	PCTV nanoStick T2 290e	em28174	2013:024f

continues on next page

Table 2 – continued from previous page

Card number	Card name	Empia Chip	USB IDs
79	Terratec Cinergy H5	em2884	eb1a:2885, 0ccd:10a2, 0ccd:10ad, 0ccd:10b6
80	PCTV DVB-S2 Stick (460e)	em28174	2013:024c
81	Hauppauge WinTV HVR 930C	em2884	2040:1605
82	Terratec Cinergy HTC Stick	em2884	0ccd:00b2
83	Honestech Vidbox NW03	em2860	eb1a:5006
84	MaxMedia UB425-TC	em2874	1b80:e425
85	PCTV QuatroStick (510e)	em2884	2304:0242
86	PCTV QuatroStick nano (520e)	em2884	2013:0251
87	Terratec Cinergy HTC USB XS	em2884	0ccd:008e, 0ccd:00ac
88	C3 Tech Digital Duo HDTV/SDTV USB	em2884	1b80:e755
89	Delock 61959	em2874	1b80:e1cc
90	KWorld USB ATSC TV Stick UB435-Q V2	em2874	1b80:e346
91	SpeedLink Vicious And Devine Laplace webcam	em2765	1ae7:9003, 1ae7:9004
92	PCTV DVB-S2 Stick (461e)	em28178	2013:0258
93	KWorld USB ATSC TV Stick UB435-Q V3	em2874	1b80:e34c
94	PCTV tripleStick (292e)	em28178	2013:025f, 2013:0264, 2040:0264, 2040:8264, 2040:8268
95	Leadtek VC100	em2861	0413:6f07
96	Terratec Cinergy T2 Stick HD	em28178	eb1a:8179
97	Elgato EyeTV Hybrid 2008 INT	em2884	0fd9:0018
98	PLEX PX-BCUD	em28178	3275:0085
99	Hauppauge WinTV-dualHD DVB	em28174	2040:0265, 2040:8265
100	Hauppauge WinTV-dualHD 01595 ATSC/QAM	em28174	2040:026d, 2040:826d
101	Terratec Cinergy H6 rev. 2	em2884	0ccd:10b2
102	:ZOLID HYBRID TV STICK	em2882	
103	Magix USB Videowandler-2	em2861	1b80:e349
104	PCTV DVB-S2 Stick (461e v2)	em28178	2013:0461, 2013:0259
105	MyGica iGrabber	em2860	1f4d:1abe

Siano cards list

Card name	USB IDs
Hauppauge Catamount	2040:1700
Hauppauge Okemo-A	2040:1800
Hauppauge Okemo-B	2040:1801
Hauppauge WinTV MiniCard	2040:2000, 2040:200a, 2040:2010, 2040:2011, 2040:2019
Hauppauge WinTV MiniCard Rev 2	2040:2009
Hauppauge WinTV MiniStick	2040:5500, 2040:5510, 2040:5520, 2040:5530, 2040:5580, 2040:5590, 2040:b900, 2040:b910, 2040:b980, 2040:b990, 2040:c000, 2040:c010, 2040:c080, 2040:c090, 2040:c0a0, 2040:f5a0
Hauppauge microStick 77e	2013:0257
ONDA Data Card Digital Receiver	19D2:0078
Siano Denver (ATSC-M/H) Digital Receiver	187f:0800
Siano Denver (TDMB) Digital Receiver	187f:0700
Siano Ming Digital Receiver	187f:0310
Siano Nice Digital Receiver	187f:0202, 187f:0202
Siano Nova A Digital Receiver	187f:0200
Siano Nova B Digital Receiver	187f:0201
Siano Pele Digital Receiver	187f:0500
Siano Rio Digital Receiver	187f:0600, 3275:0080
Siano Stellar Digital Receiver	187f:0100
Siano Stellar Digital Receiver ROM	187f:0010
Siano Vega Digital Receiver	187f:0300
Siano Venice Digital Receiver	187f:0301, 187f:0301, 187f:0302
ZTE Data Card Digital Receiver	19D2:0086

The gspca cards list

The modules for the gspca webcam drivers are:

- gspca_main: main driver
- gspca_*driver*: subdriver module with *driver* as follows

driver	vend:prod	Device
spca501	0000:0000	MystFromOri Unknown Camera
spca508	0130:0130	Clone Digital Webcam 11043
se401	03e8:0004	Endpoints/AoxSE401
zc3xx	03f0:1b07	HP Premium Starter Cam
m5602	0402:5602	ALi Video Camera Controller
spca501	040a:0002	Kodak DVC-325
spca500	040a:0300	Kodak EZ200
zc3xx	041e:041e	Creative WebCam Live!
ov519	041e:4003	Video Blaster WebCam Go Plus
stv0680	041e:4007	Go Mini
spca500	041e:400a	Creative PC-CAM 300
sunplus	041e:400b	Creative PC-CAM 600
sunplus	041e:4012	PC-Cam350
sunplus	041e:4013	Creative Pccam750
zc3xx	041e:4017	Creative Webcam Mobile PD1090
spca508	041e:4018	Creative Webcam Vista (PD1100)
spca561	041e:401a	Creative Webcam Vista (PD1100)
zc3xx	041e:401c	Creative NX
spca505	041e:401d	Creative Webcam NX ULTRA
zc3xx	041e:401e	Creative Nx Pro
zc3xx	041e:401f	Creative Webcam Notebook PD1171
zc3xx	041e:4022	Webcam NX Pro
pac207	041e:4028	Creative Webcam Vista Plus
zc3xx	041e:4029	Creative WebCam Vista Pro
zc3xx	041e:4034	Creative Instant P0620
zc3xx	041e:4035	Creative Instant P0620D
zc3xx	041e:4036	Creative Live !
sq930x	041e:4038	Creative Joy-IT
zc3xx	041e:403a	Creative Nx Pro 2
spca561	041e:403b	Creative Webcam Vista (VF0010)
sq930x	041e:403c	Creative Live! Ultra
sq930x	041e:403d	Creative Live! Ultra for Notebooks
sq930x	041e:4041	Creative Live! Motion
zc3xx	041e:4051	Creative Live!Cam Notebook Pro (VF0250)
ov519	041e:4052	Creative Live! VISTA IM
zc3xx	041e:4053	Creative Live!Cam Video IM
vc032x	041e:405b	Creative Live! Cam Notebook Ultra (VC0130)
ov519	041e:405f	Creative Live! VISTA VF0330
ov519	041e:4060	Creative Live! VISTA VF0350
ov519	041e:4061	Creative Live! VISTA VF0400
ov519	041e:4064	Creative Live! VISTA VF0420

continues on next page

Table 3 – continued from previous page

driver	vend:prod	Device
ov519	041e:4067	Creative Live! Cam Video IM (VF0350)
ov519	041e:4068	Creative Live! VISTA VF0470
sn9c2028	0458:7003	GeniusVideocam Live v2
spca561	0458:7004	Genius VideoCAM Express V2
sn9c2028	0458:7005	Genius Smart 300, version 2
sunplus	0458:7006	Genius Dsc 1.3 Smart
zc3xx	0458:7007	Genius VideoCam V2
zc3xx	0458:700c	Genius VideoCam V3
zc3xx	0458:700f	Genius VideoCam Web V2
sonixj	0458:7025	Genius Eye 311Q
sn9c20x	0458:7029	Genius Look 320s
sonixj	0458:702e	Genius Slim 310 NB
sn9c20x	0458:7045	Genius Look 1320 V2
sn9c20x	0458:704a	Genius Slim 1320
sn9c20x	0458:704c	Genius i-Look 1321
sn9c20x	045e:00f4	LifeCam VX-6000 (SN9C20x + OV9650)
sonixj	045e:00f5	MicroSoft VX3000
sonixj	045e:00f7	MicroSoft VX1000
ov519	045e:028c	Micro$oft xbox cam
kinect	045e:02ae	Xbox NUI Camera
kinect	045e:02bf	Kinect for Windows NUI Camera
spca561	0461:0815	Micro Innovations IC200 Webcam
sunplus	0461:0821	Fujifilm MV-1
zc3xx	0461:0a00	MicroInnovation WebCam320
stv06xx	046D:08F0	QuickCamMessenger
stv06xx	046D:08F5	QuickCamCommunicate
stv06xx	046D:08F6	QuickCamMessenger (new)
stv06xx	046d:0840	QuickCamExpress
stv06xx	046d:0850	LEGOcam / QuickCam Web
stv06xx	046d:0870	DexxaWebCam USB
spca500	046d:0890	Logitech QuickCam traveler
vc032x	046d:0892	Logitech Orbicam
vc032x	046d:0896	Logitech Orbicam
vc032x	046d:0897	Logitech QuickCam for Dell notebooks
zc3xx	046d:089d	Logitech QuickCam E2500
zc3xx	046d:08a0	Logitech QC IM
zc3xx	046d:08a1	Logitech QC IM 0x08A1 +sound
zc3xx	046d:08a2	Labtec Webcam Pro
zc3xx	046d:08a3	Logitech QC Chat
zc3xx	046d:08a6	Logitech QCim
zc3xx	046d:08a7	Logitech QuickCam Image
zc3xx	046d:08a9	Logitech Notebook Deluxe
zc3xx	046d:08aa	Labtec Webcam Notebook
zc3xx	046d:08ac	Logitech QuickCam Cool
zc3xx	046d:08ad	Logitech QCCommunicate STX
zc3xx	046d:08ae	Logitech QuickCam for Notebooks
zc3xx	046d:08af	Logitech QuickCam Cool

continues on next page

Table 3 – continued from previous page

driver	vend:prod	Device
zc3xx	046d:08b9	Logitech QuickCam Express
zc3xx	046d:08d7	Logitech QCam STX
zc3xx	046d:08d8	Logitech Notebook Deluxe
zc3xx	046d:08d9	Logitech QuickCam IM/Connect
zc3xx	046d:08da	Logitech QuickCam Messenger
zc3xx	046d:08dd	Logitech QuickCam for Notebooks
spca500	046d:0900	Logitech Inc. ClickSmart 310
spca500	046d:0901	Logitech Inc. ClickSmart 510
sunplus	046d:0905	Logitech ClickSmart 820
tv8532	046d:0920	Logitech QuickCam Express
tv8532	046d:0921	Labtec Webcam
spca561	046d:0928	Logitech QC Express Etch2
spca561	046d:0929	Labtec Webcam Elch2
spca561	046d:092a	Logitech QC for Notebook
spca561	046d:092b	Labtec Webcam Plus
spca561	046d:092c	Logitech QC chat Elch2
spca561	046d:092d	Logitech QC Elch2
spca561	046d:092e	Logitech QC Elch2
spca561	046d:092f	Logitech QuickCam Express Plus
sunplus	046d:0960	Logitech ClickSmart 420
nw80x	046d:d001	Logitech QuickCam Pro (dark focus ring)
se401	0471:030b	PhilipsPCVC665K
sunplus	0471:0322	Philips DMVC1300K
zc3xx	0471:0325	Philips SPC 200 NC
zc3xx	0471:0326	Philips SPC 300 NC
sonixj	0471:0327	Philips SPC 600 NC
sonixj	0471:0328	Philips SPC 700 NC
zc3xx	0471:032d	Philips SPC 210 NC
zc3xx	0471:032e	Philips SPC 315 NC
sonixj	0471:0330	Philips SPC 710 NC
se401	047d:5001	Kensington67014
se401	047d:5002	Kensington6701(5/7)
se401	047d:5003	Kensington67016
spca501	0497:c001	Smile International
sunplus	04a5:3003	Benq DC 1300
sunplus	04a5:3008	Benq DC 1500
sunplus	04a5:300a	Benq DC 3410
spca500	04a5:300c	Benq DC 1016
benq	04a5:3035	Benq DC E300
vicam	04c1:009d	HomeConnect Webcam [vicam]
konica	04c8:0720	IntelYC 76
finepix	04cb:0104	Fujifilm FinePix 4800
finepix	04cb:0109	Fujifilm FinePix A202
finepix	04cb:010b	Fujifilm FinePix A203
finepix	04cb:010f	Fujifilm FinePix A204
finepix	04cb:0111	Fujifilm FinePix A205
finepix	04cb:0113	Fujifilm FinePix A210

continues on next page

Table 3 – continued from previous page

driver	vend:prod	Device
finepix	04cb:0115	Fujifilm FinePix A303
finepix	04cb:0117	Fujifilm FinePix A310
finepix	04cb:0119	Fujifilm FinePix F401
finepix	04cb:011b	Fujifilm FinePix F402
finepix	04cb:011d	Fujifilm FinePix F410
finepix	04cb:0121	Fujifilm FinePix F601
finepix	04cb:0123	Fujifilm FinePix F700
finepix	04cb:0125	Fujifilm FinePix M603
finepix	04cb:0127	Fujifilm FinePix S300
finepix	04cb:0129	Fujifilm FinePix S304
finepix	04cb:012b	Fujifilm FinePix S500
finepix	04cb:012d	Fujifilm FinePix S602
finepix	04cb:012f	Fujifilm FinePix S700
finepix	04cb:0131	Fujifilm FinePix unknown model
finepix	04cb:013b	Fujifilm FinePix unknown model
finepix	04cb:013d	Fujifilm FinePix unknown model
finepix	04cb:013f	Fujifilm FinePix F420
sunplus	04f1:1001	JVC GC A50
spca561	04fc:0561	Flexcam 100
spca1528	04fc:1528	Sunplus MD80 clone
sunplus	04fc:500c	Sunplus CA500C
sunplus	04fc:504a	Aiptek Mini PenCam 1.3
sunplus	04fc:504b	Maxell MaxPocket LE 1.3
sunplus	04fc:5330	Digitrex 2110
sunplus	04fc:5360	Sunplus Generic
spca500	04fc:7333	PalmPixDC85
sunplus	04fc:ffff	Pure DigitalDakota
nw80x	0502:d001	DVC V6
spca501	0506:00df	3Com HomeConnect Lite
sunplus	052b:1507	Megapixel 5 Pretec DC-1007
sunplus	052b:1513	Megapix V4
sunplus	052b:1803	MegaImage VI
nw80x	052b:d001	EZCam Pro p35u
tv8532	0545:808b	Veo Stingray
tv8532	0545:8333	Veo Stingray
sunplus	0546:3155	Polaroid PDC3070
sunplus	0546:3191	Polaroid Ion 80
sunplus	0546:3273	Polaroid PDC2030
touptek	0547:6801	TTUCMOS08000KPB, AS MU800
dtcs033	0547:7303	Anchor Chips, Inc
ov519	054c:0154	Sonny toy4
ov519	054c:0155	Sonny toy5
cpia1	0553:0002	CPIA CPiA (version1) based cameras
stv0680	0553:0202	STV0680 Camera
zc3xx	055f:c005	Mustek Wcam300A
spca500	055f:c200	Mustek Gsmart 300
sunplus	055f:c211	Kowa Bs888e Microcamera

continues on next page

Table 3 – continued from previous page

driver	vend:prod	Device
spca500	055f:c220	Gsmart Mini
sunplus	055f:c230	Mustek Digicam 330K
sunplus	055f:c232	Mustek MDC3500
sunplus	055f:c360	Mustek DV4000 Mpeg4
sunplus	055f:c420	Mustek gSmart Mini 2
sunplus	055f:c430	Mustek Gsmart LCD 2
sunplus	055f:c440	Mustek DV 3000
sunplus	055f:c520	Mustek gSmart Mini 3
sunplus	055f:c530	Mustek Gsmart LCD 3
sunplus	055f:c540	Gsmart D30
sunplus	055f:c630	Mustek MDC4000
sunplus	055f:c650	Mustek MDC5500Z
nw80x	055f:d001	Mustek Wcam 300 mini
zc3xx	055f:d003	Mustek WCam300A
zc3xx	055f:d004	Mustek WCam300 AN
conex	0572:0041	Creative Notebook cx11646
ov519	05a9:0511	Video Blaster WebCam 3/WebCam Plus, D-Link USB Digital Video Camera
ov519	05a9:0518	Creative WebCam
ov519	05a9:0519	OV519 Microphone
ov519	05a9:0530	OmniVision
ov534_9	05a9:1550	OmniVision VEHO Filmscanner
ov519	05a9:2800	OmniVision SuperCAM
ov519	05a9:4519	Webcam Classic
ov534_9	05a9:8065	OmniVision test kit ov538+ov9712
ov519	05a9:8519	OmniVision
ov519	05a9:a511	D-Link USB Digital Video Camera
ov519	05a9:a518	D-Link DSB-C310 Webcam
sunplus	05da:1018	Digital Dream Enigma 1.3
stk014	05e1:0893	Syntek DV4000
gl860	05e3:0503	Genesys Logic PC Camera
gl860	05e3:f191	Genesys Logic PC Camera
vicam	0602:1001	ViCam Webcam
spca561	060b:a001	Maxell Compact Pc PM3
zc3xx	0698:2003	CTX M730V built in
topro	06a2:0003	TP6800 PC Camera, CmoX CX0342 webcam
topro	06a2:6810	Creative Qmax
nw80x	06a5:0000	Typhoon Webcam 100 USB
nw80x	06a5:d001	Divio based webcams
nw80x	06a5:d800	Divio Chicony TwinkleCam, Trust SpaceCam
spca500	06bd:0404	Agfa CL20
spca500	06be:0800	Optimedia
nw80x	06be:d001	EZCam Pro p35u
sunplus	06d6:0031	Trust 610 LCD PowerC@m Zoom
sunplus	06d6:0041	Aashima Technology B.V.
spca506	06e1:a190	ADS Instant VCD
ov534	06f8:3002	Hercules Blog Webcam
ov534_9	06f8:3003	Hercules Dualpix HD Weblog

continues on next page

Table 3 – continued from previous page

driver	vend:prod	Device
sonixj	06f8:3004	Hercules Classic Silver
sonixj	06f8:3008	Hercules Deluxe Optical Glass
pac7302	06f8:3009	Hercules Classic Link
pac7302	06f8:301b	Hercules Link
nw80x	0728:d001	AVerMedia Camguard
spca508	0733:0110	ViewQuest VQ110
spca501	0733:0401	Intel Create and Share
spca501	0733:0402	ViewQuest M318B
spca505	0733:0430	Intel PC Camera Pro
sunplus	0733:1311	Digital Dream Epsilon 1.3
sunplus	0733:1314	Mercury 2.1MEG Deluxe Classic Cam
sunplus	0733:2211	Jenoptik jdc 21 LCD
sunplus	0733:2221	Mercury Digital Pro 3.1p
sunplus	0733:3261	Concord 3045 spca536a
sunplus	0733:3281	Cyberpix S550V
spca506	0734:043b	3DeMon USB Capture aka
cpia1	0813:0001	QX3 camera
ov519	0813:0002	Dual Mode USB Camera Plus
spca500	084d:0003	D-Link DSC-350
spca500	08ca:0103	Aiptek PocketDV
sunplus	08ca:0104	Aiptek PocketDVII 1.3
sunplus	08ca:0106	Aiptek Pocket DV3100+
mr97310a	08ca:0110	Trust Spyc@m 100
mr97310a	08ca:0111	Aiptek PenCam VGA+
sunplus	08ca:2008	Aiptek Mini PenCam 2 M
sunplus	08ca:2010	Aiptek PocketCam 3M
sunplus	08ca:2016	Aiptek PocketCam 2 Mega
sunplus	08ca:2018	Aiptek Pencam SD 2M
sunplus	08ca:2020	Aiptek Slim 3000F
sunplus	08ca:2022	Aiptek Slim 3200
sunplus	08ca:2024	Aiptek DV3500 Mpeg4
sunplus	08ca:2028	Aiptek PocketCam4M
sunplus	08ca:2040	Aiptek PocketDV4100M
sunplus	08ca:2042	Aiptek PocketDV5100
sunplus	08ca:2050	Medion MD 41437
sunplus	08ca:2060	Aiptek PocketDV5300
tv8532	0923:010f	ICM532 cams
mr97310a	093a:010e	All known CIF cams with this ID
mr97310a	093a:010f	All known VGA cams with this ID
mars	093a:050f	Mars-Semi Pc-Camera
pac207	093a:2460	Qtec Webcam 100
pac207	093a:2461	HP Webcam
pac207	093a:2463	Philips SPC 220 NC
pac207	093a:2464	Labtec Webcam 1200
pac207	093a:2468	Webcam WB-1400T
pac207	093a:2470	Genius GF112
pac207	093a:2471	Genius VideoCam ge111

continues on next page

Table 3 – continued from previous page

driver	vend:prod	Device
pac207	093a:2472	Genius VideoCam ge110
pac207	093a:2474	Genius iLook 111
pac207	093a:2476	Genius e-Messenger 112
pac7311	093a:2600	PAC7311 Typhoon
pac7311	093a:2601	Philips SPC 610 NC
pac7311	093a:2603	Philips SPC 500 NC
pac7311	093a:2608	Trust WB-3300p
pac7311	093a:260e	Gigaware VGA PC Camera, Trust WB-3350p, SIGMA cam 2350
pac7311	093a:260f	SnakeCam
pac7302	093a:2620	Apollo AC-905
pac7302	093a:2621	PAC731x
pac7302	093a:2622	Genius Eye 312
pac7302	093a:2623	Pixart Imaging, Inc.
pac7302	093a:2624	PAC7302
pac7302	093a:2625	Genius iSlim 310
pac7302	093a:2626	Labtec 2200
pac7302	093a:2627	Genius FaceCam 300
pac7302	093a:2628	Genius iLook 300
pac7302	093a:2629	Genius iSlim 300
pac7302	093a:262a	Webcam 300k
pac7302	093a:262c	Philips SPC 230 NC
jl2005bcd	0979:0227	Various brands, 19 known cameras supported
jeilinj	0979:0270	Sakar 57379
jeilinj	0979:0280	Sportscam DV15, Sakar 57379
zc3xx	0ac8:0301	Web Camera
zc3xx	0ac8:0302	Z-star Vimicro zc0302
vc032x	0ac8:0321	Vimicro generic vc0321
vc032x	0ac8:0323	Vimicro Vc0323
vc032x	0ac8:0328	A4Tech PK-130MG
zc3xx	0ac8:301b	Z-Star zc301b
zc3xx	0ac8:303b	Vimicro 0x303b
zc3xx	0ac8:305b	Z-star Vimicro zc0305b
zc3xx	0ac8:307b	PC Camera (ZS0211)
vc032x	0ac8:c001	Sony embedded vimicro
vc032x	0ac8:c002	Sony embedded vimicro
vc032x	0ac8:c301	Samsung Q1 Ultra Premium
spca508	0af9:0010	Hama USB Sightcam 100
spca508	0af9:0011	Hama USB Sightcam 100
ov519	0b62:0059	iBOT2 Webcam
sonixb	0c45:6001	Genius VideoCAM NB
sonixb	0c45:6005	Microdia Sweex Mini Webcam
sonixb	0c45:6007	Sonix sn9c101 + Tas5110D
sonixb	0c45:6009	spcaCam@120
sonixb	0c45:600d	spcaCam@120
sonixb	0c45:6011	Microdia PC Camera (SN9C102)
sonixb	0c45:6019	Generic Sonix OV7630
sonixb	0c45:6024	Generic Sonix Tas5130c

continues on next page

Table 3 – continued from previous page

driver	vend:prod	Device
sonixb	0c45:6025	Xcam Shanga
sonixb	0c45:6027	GeniusEye 310
sonixb	0c45:6028	Sonix Btc Pc380
sonixb	0c45:6029	spcaCam@150
sonixb	0c45:602a	Meade ETX-105EC Camera
sonixb	0c45:602c	Generic Sonix OV7630
sonixb	0c45:602d	LIC-200 LG
sonixb	0c45:602e	Genius VideoCam Messenger
sonixj	0c45:6040	Speed NVC 350K
sonixj	0c45:607c	Sonix sn9c102p Hv7131R
sonixb	0c45:6083	VideoCAM Look
sonixb	0c45:608c	VideoCAM Look
sonixb	0c45:608f	PC Camera (SN9C103 + OV7630)
sonixb	0c45:60a8	VideoCAM Look
sonixb	0c45:60aa	VideoCAM Look
sonixb	0c45:60af	VideoCAM Look
sonixb	0c45:60b0	Genius VideoCam Look
sonixj	0c45:60c0	Sangha Sn535
sonixj	0c45:60ce	USB-PC-Camera-168 (TALK-5067)
sonixj	0c45:60ec	SN9C105+MO4000
sonixj	0c45:60fb	Surfer NoName
sonixj	0c45:60fc	LG-LIC300
sonixj	0c45:60fe	Microdia Audio
sonixj	0c45:6100	PC Camera (SN9C128)
sonixj	0c45:6102	PC Camera (SN9C128)
sonixj	0c45:610a	PC Camera (SN9C128)
sonixj	0c45:610b	PC Camera (SN9C128)
sonixj	0c45:610c	PC Camera (SN9C128)
sonixj	0c45:610e	PC Camera (SN9C128)
sonixj	0c45:6128	Microdia/Sonix SNP325
sonixj	0c45:612a	Avant Camera
sonixj	0c45:612b	Speed-Link REFLECT2
sonixj	0c45:612c	Typhoon Rasy Cam 1.3MPix
sonixj	0c45:612e	PC Camera (SN9C110)
sonixj	0c45:6130	Sonix Pccam
sonixj	0c45:6138	Sn9c120 Mo4000
sonixj	0c45:613a	Microdia Sonix PC Camera
sonixj	0c45:613b	Surfer SN-206
sonixj	0c45:613c	Sonix Pccam168
sonixj	0c45:613e	PC Camera (SN9C120)
sonixj	0c45:6142	Hama PC-Webcam AC-150
sonixj	0c45:6143	Sonix Pccam168
sonixj	0c45:6148	Digitus DA-70811/ZSMC USB PC Camera ZS211/Microdia
sonixj	0c45:614a	Frontech E-Ccam (JIL-2225)
sn9c20x	0c45:6240	PC Camera (SN9C201 + MT9M001)
sn9c20x	0c45:6242	PC Camera (SN9C201 + MT9M111)
sn9c20x	0c45:6248	PC Camera (SN9C201 + OV9655)

continues on next page

Table 3 – continued from previous page

driver	vend:prod	Device
sn9c20x	0c45:624c	PC Camera (SN9C201 + MT9M112)
sn9c20x	0c45:624e	PC Camera (SN9C201 + SOI968)
sn9c20x	0c45:624f	PC Camera (SN9C201 + OV9650)
sn9c20x	0c45:6251	PC Camera (SN9C201 + OV9650)
sn9c20x	0c45:6253	PC Camera (SN9C201 + OV9650)
sn9c20x	0c45:6260	PC Camera (SN9C201 + OV7670)
sn9c20x	0c45:6270	PC Camera (SN9C201 + MT9V011/MT9V111/MT9V112)
sn9c20x	0c45:627b	PC Camera (SN9C201 + OV7660)
sn9c20x	0c45:627c	PC Camera (SN9C201 + HV7131R)
sn9c20x	0c45:627f	PC Camera (SN9C201 + OV9650)
sn9c20x	0c45:6280	PC Camera (SN9C202 + MT9M001)
sn9c20x	0c45:6282	PC Camera (SN9C202 + MT9M111)
sn9c20x	0c45:6288	PC Camera (SN9C202 + OV9655)
sn9c20x	0c45:628c	PC Camera (SN9C201 + MT9M112)
sn9c20x	0c45:628e	PC Camera (SN9C202 + SOI968)
sn9c20x	0c45:628f	PC Camera (SN9C202 + OV9650)
sn9c20x	0c45:62a0	PC Camera (SN9C202 + OV7670)
sn9c20x	0c45:62b0	PC Camera (SN9C202 + MT9V011/MT9V111/MT9V112)
sn9c20x	0c45:62b3	PC Camera (SN9C202 + OV9655)
sn9c20x	0c45:62bb	PC Camera (SN9C202 + OV7660)
sn9c20x	0c45:62bc	PC Camera (SN9C202 + HV7131R)
sn9c2028	0c45:8001	Wild Planet Digital Spy Camera
sn9c2028	0c45:8003	Sakar #11199, #6637x, #67480 keychain cams
sn9c2028	0c45:8008	Mini-Shotz ms-350
sn9c2028	0c45:800a	Vivitar Vivicam 3350B
sunplus	0d64:0303	Sunplus FashionCam DXG
ov519	0e96:c001	TRUST 380 USB2 SPACEC@M
etoms	102c:6151	Qcam Sangha CIF
etoms	102c:6251	Qcam xxxxxx VGA
ov519	1046:9967	W9967CF/W9968CF WebCam IC, Video Blaster WebCam Go
zc3xx	10fd:0128	Typhoon Webshot II USB 300k 0x0128
spca561	10fd:7e50	FlyCam Usb 100
zc3xx	10fd:804d	Typhoon Webshot II Webcam [zc0301]
zc3xx	10fd:8050	Typhoon Webshot II USB 300k
ov534	1415:2000	Sony HD Eye for PS3 (SLEH 00201)
pac207	145f:013a	Trust WB-1300N
pac7302	145f:013c	Trust
sn9c20x	145f:013d	Trust WB-3600R
vc032x	15b8:6001	HP 2.0 Megapixel
vc032x	15b8:6002	HP 2.0 Megapixel rz406aa
stk1135	174f:6a31	ASUSlaptop, MT9M112 sensor
spca501	1776:501c	Arowana 300K CMOS Camera
t613	17a1:0128	TASCORP JPEG Webcam, NGS Cyclops
vc032x	17ef:4802	Lenovo Vc0323+MI1310_SOC
pac7302	1ae7:2001	SpeedLinkSnappy Mic SL-6825-SBK
pac207	2001:f115	D-Link DSB-C120
sq905c	2770:9050	Disney pix micro (CIF)

continues on next page

Table 3 – continued from previous page

driver	vend:prod	Device
sq905c	2770:9051	Lego Bionicle
sq905c	2770:9052	Disney pix micro 2 (VGA)
sq905c	2770:905c	All 11 known cameras with this ID
sq905	2770:9120	All 24 known cameras with this ID
sq905c	2770:913d	All 4 known cameras with this ID
sq930x	2770:930b	Sweex Motion Tracking / I-Tec iCam Tracer
sq930x	2770:930c	Trust WB-3500T / NSG Robbie 2.0
spca500	2899:012c	Toptro Industrial
ov519	8020:ef04	ov519
spca508	8086:0110	Intel Easy PC Camera
spca500	8086:0630	Intel Pocket PC Camera
spca506	99fa:8988	Grandtec V.cap
sn9c20x	a168:0610	Dino-Lite Digital Microscope (SN9C201 + HV7131R)
sn9c20x	a168:0611	Dino-Lite Digital Microscope (SN9C201 + HV7131R)
sn9c20x	a168:0613	Dino-Lite Digital Microscope (SN9C201 + HV7131R)
sn9c20x	a168:0614	Dino-Lite Digital Microscope (SN9C201 + MT9M111)
sn9c20x	a168:0615	Dino-Lite Digital Microscope (SN9C201 + MT9M111)
sn9c20x	a168:0617	Dino-Lite Digital Microscope (SN9C201 + MT9M111)
sn9c20x	a168:0618	Dino-Lite Digital Microscope (SN9C201 + HV7131R)
spca561	abcd:cdee	Petcam

dvb-usb-dib0700 cards list

Card name	USB IDs
ASUS My Cinema U3000 Mini DVBT Tuner	0b05:171f
ASUS My Cinema U3100 Mini DVBT Tuner	0b05:173f
AVerMedia AVerTV DVB-T Express	07ca:b568
AVerMedia AVerTV DVB-T Volar	07ca:a807, 07ca:b808
Artec T14BR DVB-T	05d8:810f
Asus My Cinema-U3000Hybrid	0b05:1736
Compro Videomate U500	185b:1e78, 185b:1e80
DiBcom NIM7090 reference design	10b8:1bb2
DiBcom NIM8096MD reference design	10b8:1fa8
DiBcom NIM9090MD reference design	10b8:2384
DiBcom STK7070P reference design	10b8:1ebc
DiBcom STK7070PD reference design	10b8:1ebe
DiBcom STK7700D reference design	10b8:1ef0
DiBcom STK7700P reference design	10b8:1e14, 10b8:1e78
DiBcom STK7770P reference design	10b8:1e80

continues on next page

Table 4 – continued from previous page

Card name	USB IDs
DiBcom STK807xP reference design	10b8:1f90
DiBcom STK807xPVR reference design	10b8:1f98
DiBcom STK8096-PVR reference design	2013:1faa, 10b8:1faa
DiBcom STK8096GP reference design	10b8:1fa0
DiBcom STK9090M reference design	10b8:2383
DiBcom TFE7090PVR reference design	10b8:1bb4
DiBcom TFE7790P reference design	10b8:1e6e
DiBcom TFE8096P reference design	10b8:1f9C
Elgato EyeTV DTT	0fd9:0021
Elgato EyeTV DTT rev. 2	0fd9:003f
Elgato EyeTV Diversity	0fd9:0011
Elgato EyeTV Dtt Dlx PD378S	0fd9:0020
EvolutePC TVWay+	1e59:0002
Gigabyte U7000	1044:7001
Gigabyte U8000-RH	1044:7002
Hama DVB=T Hybrid USB Stick	147f:2758
Hauppauge ATSC MiniCard (B200)	2040:b200
Hauppauge ATSC MiniCard (B210)	2040:b210
Hauppauge Nova-T 500 Dual DVB-T	2040:9941, 2040:9950
Hauppauge Nova-T MyTV.t	2040:7080
Hauppauge Nova-T Stick	2040:7050, 2040:7060, 2040:7070
Hauppauge Nova-TD Stick (52009)	2040:5200
Hauppauge Nova-TD Stick/Elgato Eye-TV Diversity	2040:9580
Hauppauge Nova-TD-500 (84xxx)	2040:8400
Leadtek WinFast DTV Dongle H	0413:60f6
Leadtek Winfast DTV Dongle (STK7700P based)	0413:6f00, 0413:6f01
Medion CTX1921 DVB-T USB	1660:1921
Microsoft Xbox One Digital TV Tuner	045e:02d5
PCTV 2002e	2013:025c
PCTV 2002e SE	2013:025d
Pinnacle Expresscard 320cx	2304:022e
Pinnacle PCTV 2000e	2304:022c
Pinnacle PCTV 282e	2013:0248, 2304:0248
Pinnacle PCTV 340e HD Pro USB Stick	2304:023d
Pinnacle PCTV 72e	2304:0236
Pinnacle PCTV 73A	2304:0243
Pinnacle PCTV 73e	2304:0237
Pinnacle PCTV 73e SE	2013:0245, 2304:0245

continues on next page

Table 4 – continued from previous page

Card name	USB IDs
Pinnacle PCTV DVB-T Flash Stick	2304:0228
Pinnacle PCTV Dual DVB-T Diversity Stick	2304:0229
Pinnacle PCTV HD Pro USB Stick	2304:023a
Pinnacle PCTV HD USB Stick	2304:023b
Pinnacle PCTV Hybrid Stick Solo	2304:023e
Prolink Pixelview SBTVD	1554:5010
Sony PlayTV	1415:0003
TechniSat AirStar TeleStick 2	14f7:0004
Terratec Cinergy DT USB XS Diversity/ T5	0ccd:0081, 0ccd:10a1
Terratec Cinergy DT XS Diversity	0ccd:005a
Terratec Cinergy HT Express	0ccd:0060
Terratec Cinergy HT USB XE	0ccd:0058
Terratec Cinergy T Express	0ccd:0062
Terratec Cinergy T USB XXS (HD)/ T3	0ccd:0078, 0ccd:10a0, 0ccd:00ab
Uniwill STK7700P based (Hama and others)	1584:6003
YUAN High-Tech DiBcom STK7700D	1164:1e8c
YUAN High-Tech MC770	1164:0871
YUAN High-Tech STK7700D	1164:1efc
YUAN High-Tech STK7700PH	1164:1f08
Yuan EC372S	1164:1edc
Yuan PD378S	1164:2edc

dvb-usb-dibusb-mb cards list

Card name	USB IDs
AVerMedia AverTV DVBT USB1.1	14aa:0001, 14aa:0002
Artec T1 USB1.1 TVBOX with AN2135	05d8:8105, 05d8:8106
Artec T1 USB1.1 TVBOX with AN2235	05d8:8107, 05d8:8108
Artec T1 USB1.1 TVBOX with AN2235 (faulty USB IDs)	0547:2235
Artec T1 USB2.0	05d8:8109, 05d8:810a
Compro Videomate DVB-U2000 - DVB-T USB1.1 (please confirm to linux-dvb)	185b:d000, 145f:010c, 185b:d001
DiBcom USB1.1 DVB-T reference design (MOD3000)	10b8:0bb8, 10b8:0bb9
Grandtec USB1.1 DVB-T	5032:0fa0, 5032:0bb8, 5032:0fa1, 5032:0bb9
KWorld V-Stream XPERT DTV - DVB-T USB1.1	eb1a:17de, eb1a:17df
KWorld Xpert DVB-T USB2.0	eb2a:17de
KWorld/ADSTech Instant DVB-T USB2.0	06e1:a333, 06e1:a334
TwinhanDTV USB-Ter USB1.1 / Magic Box I / HAMA USB1.1 DVB-T device	13d3:3201, 1822:3201, 13d3:3202, 1822:3202
Unknown USB1.1 DVB-T device ???? please report the name to the author	1025:005e, 1025:005f
VideoWalker DVB-T USB	0458:701e, 0458:701f

dvb-usb-dibusb-mc cards list

Card name	USB IDs
Artec T1 USB2.0 TVBOX (please check the warm ID)	05d8:8109, 05d8:810a
Artec T14 - USB2.0 DVB-T	05d8:810b, 05d8:810c
DiBcom USB2.0 DVB-T reference design (MOD3000P)	10b8:0bc6, 10b8:0bc7
GRAND - USB2.0 DVB-T adapter	5032:0bc6, 5032:0bc7
Humax/Coex DVB-T USB Stick 2.0 High Speed	10b9:5000, 10b9:5001
LITE-ON USB2.0 DVB-T Tuner	04ca:f000, 04ca:f001
Leadtek - USB2.0 Winfast DTV dongle	0413:6025, 0413:6026
MSI Digivox Mini SL	eb1a:e360, eb1a:e361

dvb-usb-a800 cards list

Card name	USB IDs
AVerMedia AverTV DVB-T USB 2.0 (A800)	07ca:a800, 07ca:a801

dvb-usb-af9005 cards list

Card name	USB IDs
Afatech DVB-T USB1.1 stick	15a4:9020
Ansonic DVB-T USB1.1 stick	10b9:6000
TerraTec Cinergy T USB XE	0ccd:0055

dvb-usb-az6027 cards list

Card name	USB IDs
AZUREWAVE DVB-S/S2 USB2.0 (AZ6027)	13d3:3275
Elgato EyeTV Sat	0fd9:002a, 0fd9:0025, 0fd9:0036
TERRATEC S7	0ccd:10a4
TERRATEC S7 MKII	0ccd:10ac
Technisat SkyStar USB 2 HD CI	14f7:0001, 14f7:0002

dvb-usb-cinergyT2 cards list

Card name	USB IDs
TerraTec/qanu USB2.0 Highspeed DVB-T Receiver	0ccd:0x0038

dvb-usb-cxusb cards list

Card name	USB IDs
AVerMedia AVerTVHD Volar (A868R)	
Conexant DMB-TH Stick	
DViCO FusionHDTV DVB-T Dual Digital 2	
DViCO FusionHDTV DVB-T Dual Digital 4	
DViCO FusionHDTV DVB-T Dual Digital 4 (rev 2)	
DViCO FusionHDTV DVB-T Dual USB	
DViCO FusionHDTV DVB-T NANO2	
DViCO FusionHDTV DVB-T USB (LGZ201)	
DViCO FusionHDTV DVB-T USB (TH7579)	
DViCO FusionHDTV5 USB Gold	
DigitalNow DVB-T Dual USB	
Medion MD95700 (MDUSBTV-HYBRID)	
Mygica D689 DMB-TH	

dvb-usb-digitv cards list

Card name	USB IDs
Nebula Electronics uDigiTV DVB-T USB2.0)	0547:0201

dvb-usb-dtt200u cards list

Card name	USB IDs
WideView WT-220U PenType Receiver (Miglia)	18f3:0220
WideView WT-220U PenType Receiver (Typhoon/Freecom)	14aa:0222, 14aa:0220, 14aa:0221, 14aa:0225, 14aa:0226
WideView WT-220U PenType Receiver (based on ZL353)	14aa:022a, 14aa:022b
WideView/Yuan/Yakumo/Hama/Typhoon DVB-T USB2.0 (WT-200U)	14aa:0201, 14aa:0301

dvb-usb-dtv5100 cards list

Card name	USB IDs
AME DTV-5100 USB2.0 DVB-T	0x06be:0xa232

dvb-usb-dw2102 cards list

Card name	USB IDs
DVBWorld DVB-C 3101 USB2.0	04b4:3101
DVBWorld DVB-S 2101 USB2.0	04b4:0x2101
DVBWorld DVB-S 2102 USB2.0	04b4:2102
DVBWorld DW2104 USB2.0	04b4:2104
GOTVIEW Satellite HD	0x1FE1:5456
Geniatech T220 DVB-T/T2 USB2.0	0x1f4d:0xD220
SU3000HD DVB-S USB2.0	0x1f4d:0x3000
TeVii S482 (tuner 1)	0x9022:0xd483
TeVii S482 (tuner 2)	0x9022:0xd484
TeVii S630 USB	0x9022:d630
TeVii S650 USB2.0	0x9022:d650
TeVii S662	0x9022:d662
TechnoTrend TT-connect S2-4600	0b48:3011
TerraTec Cinergy S USB	0ccd:0064
Terratec Cinergy S2 PCIe Dual Port 1	153b:1181
Terratec Cinergy S2 PCIe Dual Port 2	153b:1182
Terratec Cinergy S2 USB BOX	0ccd:0x0105
Terratec Cinergy S2 USB HD	0ccd:00a8
Terratec Cinergy S2 USB HD Rev.2	0ccd:00b0
Terratec Cinergy S2 USB HD Rev.3	0ccd:0102
X3M TV SPC1400HD PCI	0x1f4d:0x3100

dvb-usb-gp8psk cards list

Card name	USB IDs
Genpix 8PSK-to-USB2 Rev.1 DVB-S receiver	09c0:0200, 09c0:0201
Genpix 8PSK-to-USB2 Rev.2 DVB-S receiver	09c0:0202
Genpix SkyWalker-1 DVB-S receiver	09c0:0203
Genpix SkyWalker-2 DVB-S receiver	09c0:0206

dvb-usb-m920x cards list

Card name	USB IDs
DTV-DVB UDTT7049	13d3:3219
Dposh DVB-T USB2.0	1498:9206, 1498:a090
LifeView TV Walker Twin DVB-T USB2.0	10fd:0514, 10fd:0513
MSI DIGI VOX mini II DVB-T USB2.0	10fd:1513
MSI Mega Sky 580 DVB-T USB2.0	0db0:5580
Pinnacle PCTV 310e	13d3:3211

dvb-usb-nova-t-usb2 cards list

Card name	USB IDs
Hauppauge WinTV-NOVA-T usb2	2040:9300, 2040:9301

dvb-usb-opera1 cards list

Card name	USB IDs
Opera1 DVB-S USB2.0	04b4:2830, 695c:3829

dvb-usb-pctv452e cards list

Card name	USB IDs
PCTV HDTV USB	2304:021f
Technotrend TT Connect S2-3600	0b48:3007
Technotrend TT Connect S2-3650-CI	0b48:300a

dvb-usb-technisat-usb2 cards list

Card name	USB IDs
Technisat SkyStar USB HD (DVB-S/S2)	14f7:0500

dvb-usb-ttusb2 cards list

Card name	USB IDs
Pinnacle 400e DVB-S USB2.0	2304:020f
Pinnacle 450e DVB-S USB2.0	2304:0222
Technotrend TT-connect CT-3650	0b48:300d
Technotrend TT-connect S-2400	0b48:3006
Technotrend TT-connect S-2400 (8kB EEPROM)	0b48:3009

dvb-usb-umt-010 cards list

Card name	USB IDs
Hanftek UMT-010 DVB-T USB2.0	15f4:0001, 15f4:0015

dvb-usb-vp702x cards list

Card name	USB IDs
TwinhanDTV StarBox DVB-S USB2.0 (VP7021)	13d3:3207

dvb-usb-vp7045 cards list

Card name	USB IDs
DigitalNow TinyUSB 2 DVB-t Receiver	13d3:3223, 13d3:3224
Twinhan USB2.0 DVB-T receiver (TwinhanDTV Alpha/MagicBox II)	13d3:3205, 13d3:3206

dvb-usb-af9015 cards list

Card name	USB IDs
AVerMedia A309	07ca:a309
AVerMedia AVerTV DVB-T Volar X	07ca:a815
Afatech AF9015 reference design	15a4:9015, 15a4:9016
AverMedia AVerTV Red HD+ (A850T)	07ca:850b
AverMedia AVerTV Volar Black HD (A850)	07ca:850a
AverMedia AVerTV Volar GPS 805 (A805)	07ca:a805

continues on next page

Table 5 – continued from previous page

Card name	USB IDs
AverMedia AVerTV Volar M (A815Mac)	07ca:815a
Conceptronic USB2.0 DVB-T CTVDIGRCU V3.0	1b80:e397
DigitalNow TinyTwin	13d3:3226
DigitalNow TinyTwin v2	1b80:e402
DigitalNow TinyTwin v3	1f4d:9016
Fujitsu-Siemens Slim Mobile USB DVB-T	07ca:8150
Genius TVGo DVB-T03	0458:4012
KWorld Digital MC-810	1b80:c810
KWorld PlusTV DVB-T PCI Pro Card (DVB-T PC160-T)	1b80:c161
KWorld PlusTV Dual DVB-T PCI (DVB-T PC160-2T)	1b80:c160
KWorld PlusTV Dual DVB-T Stick (DVB-T 399U)	1b80:e399, 1b80:e400
KWorld USB DVB-T Stick Mobile (UB383-T)	1b80:e383
KWorld USB DVB-T TV Stick II (VS-DVB-T 395U)	1b80:e396, 1b80:e39b, 1b80:e395, 1b80:e39a
Leadtek WinFast DTV Dongle Gold	0413:6029
Leadtek WinFast DTV2000DS	0413:6a04
MSI DIGIVOX Duo	1462:8801
MSI Digi VOX mini III	1462:8807
Pinnacle PCTV 71e	2304:022b
Sveon STV20 Tuner USB DVB-T HDTV	1b80:e39d
Sveon STV22 Dual USB DVB-T Tuner HDTV	1b80:e401
Telestar Starstick 2	10b9:8000
TerraTec Cinergy T Stick Dual RC	0ccd:0099
TerraTec Cinergy T Stick RC	0ccd:0097
TerraTec Cinergy T USB XE	0ccd:0069
TrekStor DVB-T USB Stick	15a4:901b
TwinHan AzureWave AD-TU700(704J)	13d3:3237
Xtensions XD-380	1ae7:0381

dvb-usb-af9035 cards list

Card name	USB IDs
AVerMedia AVerTV Volar HD/PRO (A835)	07ca:a835, 07ca:b835
AVerMedia HD Volar (A867)	07ca:1867, 07ca:a867, 07ca:0337
AVerMedia TD310 DVB-T2	07ca:1871
AVerMedia Twinstar (A825)	07ca:0825
Afatech AF9035 reference design	15a4:9035, 15a4:1000, 15a4:1001, 15a4:1002, 15a4:1003
Asus U3100Mini Plus	0b05:1779
Avermedia A835B(1835)	07ca:1835
Avermedia A835B(2835)	07ca:2835
Avermedia A835B(3835)	07ca:3835
Avermedia A835B(4835)	07ca:4835
Avermedia AverTV Volar HD 2 (TD110)	07ca:a110
Avermedia H335	07ca:0335
Digital Dual TV Receiver CTVDIG-DUAL_V2	1b80:e410
EVOLVEO XtraTV stick	1f4d:a115
Hauppauge WinTV-MiniStick 2	2040:f900
ITE 9135 Generic	048d:9135
ITE 9135(9005) Generic	048d:9005
ITE 9135(9006) Generic	048d:9006
ITE 9303 Generic	048d:9306
Kworld UB499-2T T09	1b80:e409
Leadtek WinFast DTV Dongle Dual	0413:6a05
Logilink VG0022A	1d19:0100
PCTV AndroiDTV (78e)	2013:025a
PCTV microStick (79e)	2013:0262
Sveon STV22 Dual DVB-T HDTV	1b80:e411
TerraTec Cinergy T Stick	0ccd:0093
TerraTec Cinergy T Stick (rev. 2)	0ccd:00aa
TerraTec Cinergy T Stick Dual RC (rev. 2)	0ccd:0099
TerraTec Cinergy TC2 Stick	0ccd:10b2
TerraTec T1	0ccd:10ae

dvb-usb-anysee cards list

Card name	USB IDs
Anysee	04b4:861f, 1c73:861f

dvb-usb-au6610 cards list

Card name	USB IDs
Sigmatek DVB-110	058f:6610

dvb-usb-az6007 cards list

Card name	USB IDs
Azurewave 6007	13d3:0ccd
Technisat CableStar Combo HD CI	14f7:0003
Terratec H7	0ccd:10b4, 0ccd:10a3

dvb-usb-ce6230 cards list

Card name	USB IDs
AVerMedia A310 USB 2.0 DVB-T tuner	07ca:a310
Intel CE9500 reference design	8086:9500

dvb-usb-dvbsky cards list

Card name	USB IDs
DVBSky S960/S860	0572:6831
DVBSky S960CI	0572:960c
DVBSky T330	0572:0320
DVBSky T680CI	0572:680c
MyGica Mini DVB-(T/T2/C) USB Stick T230	0572:c688
MyGica Mini DVB-(T/T2/C) USB Stick T230C	0572:c689
MyGica Mini DVB-(T/T2/C) USB Stick T230C Lite	0572:c699
MyGica Mini DVB-(T/T2/C) USB Stick T230C v2	0572:c68a
TechnoTrend TT-connect CT2-4650 CI	0b48:3012
TechnoTrend TT-connect CT2-4650 CI v1.1	0b48:3015
TechnoTrend TT-connect S2-4650 CI	0b48:3017
TechnoTrend TVStick CT2-4400	0b48:3014
Terratec Cinergy S2 Rev.4	0ccd:0105
Terratec H7 Rev.4	0ccd:10a5

dvb-usb-ec168 cards list

Card name	USB IDs
E3C EC168 reference design	18b4:1689, 18b4:fffa, 18b4:fffb, 18b4:1001, 18b4:1002

dvb-usb-gl861 cards list

Card name	USB IDs
774 Friio White ISDB-T USB2.0	7a69:0001
A-LINK DTU DVB-T USB2.0	05e3:f170
MSI Mega Sky 55801 DVB-T USB2.0	0db0:5581

dvb-usb-lmedm04 cards list

Card name	USB IDs
DM04_LME2510C_DVB-S	3344:1120
DM04_LME2510C_DVB-S RS2000	3344:22f0
DM04_LME2510_DVB-S	3344:1122

dvb-usb-mxl111sf cards list

Card name	USB IDs
HCW 117xxx	2040:b702
HCW 126xxx	2040:c602, 2040:c60a
Hauppauge 117xxx ATSC+	2040:b700, 2040:b703, 2040:b753, 2040:b763, 2040:b757, 2040:b767
Hauppauge 117xxx DVBT	2040:b704, 2040:b764
Hauppauge 126xxx	2040:c612, 2040:c61a
Hauppauge 126xxx ATSC	2040:c601, 2040:c609, 2040:b701
Hauppauge 126xxx ATSC+	2040:c600, 2040:c603, 2040:c60b, 2040:c653, 2040:c65b
Hauppauge 126xxx DVBT	2040:c604, 2040:c60c
Hauppauge 138xxx DVBT	2040:d854, 2040:d864, 2040:d8d4, 2040:d8e4
Hauppauge Mercury	2040:d853, 2040:d863, 2040:d8d3, 2040:d8e3, 2040:d8ff
Hauppauge WinTV-Aero-M	2040:c613, 2040:c61b

dvb-usb-rtl28xxu cards list

Card name	USB IDs
ASUS My Cinema-U3100Mini Plus V2	1b80:d3a8
Astrometa DVB-T2	15f4:0131
Compro VideoMate U620F	185b:0620
Compro VideoMate U650F	185b:0650
Crypto ReDi PC 50 A	1f4d:a803
Dexatek DK DVB-T Dongle	1d19:1101
Dexatek DK mini DVB-T Dongle	1d19:1102
DigitalNow Quad DVB-T Receiver	0413:6680
Freecom USB2.0 DVB-T	14aa:0160, 14aa:0161
G-Tek Electronics Group Lifeview LV5TDLX DVB-T	1f4d:b803
GIGABYTE U7300	1b80:d393
Genius TVGo DVB-T03	0458:707f
GoTView MasterHD 3	5654:ca42
Leadtek WinFast DTV Dongle mini	0413:6a03
Leadtek WinFast DTV2000DS Plus	0413:6f12

continues on next page

Table 7 – continued from previous page

Card name	USB IDs
Leadtek Winfast DTV Dongle Mini D	0413:6f0f
MSI DIGIVOX Micro HD	1d19:1104
MaxMedia HU394-T	1b80:d394
PROlectrix DV107669	1f4d:d803
Peak DVB-T USB	1b80:d395
Realtek RTL2831U reference design	0bda:2831
Realtek RTL2832U reference design	0bda:2832, 0bda:2838
Sveon STV20	1b80:d39d
Sveon STV21	1b80:d3b0
Sveon STV27	1b80:d3af
TURBO-X Pure TV Tuner DTT-2000	1b80:d3a4
TerraTec Cinergy T Stick Black	0ccd:00a9
TerraTec Cinergy T Stick RC (Rev. 3)	0ccd:00d3
TerraTec Cinergy T Stick+	0ccd:00d7
TerraTec NOXON DAB Stick	0ccd:00b3
TerraTec NOXON DAB Stick (rev 2)	0ccd:00e0
TerraTec NOXON DAB Stick (rev 3)	0ccd:00b4
Trekstor DVB-T Stick Terres 2.0	1f4d:C803

dvb-usb-zd1301 cards list

Card name	USB IDs
ZyDAS ZD1301 reference design	0ace:13a1

Other USB cards list

Driver	Card name	USB IDs
airspy	Airspy	1d50:60a1
dvb-as102	Abilis Systems DVB-Titan	1BA6:0001
dvb-as102	PCTV Systems picoStick (74e)	2013:0246
dvb-as102	Elgato EyeTV DTT Deluxe	0fd9:002c
dvb-as102	nBox DVB-T Dongle	0b89:0007
dvb-as102	Sky IT Digital Key (green led)	2137:0001
b2c2-flexcop-usb	Technisat/B2C2 FlexCop II/IIb/III Digital TV	0af7:0101
go7007	WIS GO7007 MPEG encoder	1943:a250, 093b:a00
hackrf	HackRF Software Decoder Radio	1d50:6089
hdpvr	Hauppauge HD PVR	2040:4900, 2040:490
msi2500	Mirics MSi3101 SDR Dongle	1df7:2500, 2040:d30
pvrusb2	Hauppauge WinTV-PVR USB2	2040:2900, 2040:295
pwc	Creative Webcam 5	041E:400C
pwc	Creative Webcam Pro Ex	041E:4011

Tab

Driver	Card name	USB IDs
pwc	Logitech QuickCam 3000 Pro	046D:08B0
pwc	Logitech QuickCam Notebook Pro	046D:08B1
pwc	Logitech QuickCam 4000 Pro	046D:08B2
pwc	Logitech QuickCam Zoom (old model)	046D:08B3
pwc	Logitech QuickCam Zoom (new model)	046D:08B4
pwc	Logitech QuickCam Orbit/Sphere	046D:08B5
pwc	Logitech/Cisco VT Camera	046D:08B6
pwc	Logitech ViewPort AV 100	046D:08B7
pwc	Logitech QuickCam	046D:08B8
pwc	Philips PCA645VC	0471:0302
pwc	Philips PCA646VC	0471:0303
pwc	Askey VC010 type 2	0471:0304
pwc	Philips PCVC675K (Vesta)	0471:0307
pwc	Philips PCVC680K (Vesta Pro)	0471:0308
pwc	Philips PCVC690K (Vesta Pro Scan)	0471:030C
pwc	Philips PCVC730K (ToUCam Fun), PCVC830 (ToUCam II)	0471:0310
pwc	Philips PCVC740K (ToUCam Pro), PCVC840 (ToUCam II)	0471:0311
pwc	Philips PCVC750K (ToUCam Pro Scan)	0471:0312
pwc	Philips PCVC720K/40 (ToUCam XS)	0471:0313
pwc	Philips SPC 900NC	0471:0329
pwc	Philips SPC 880NC	0471:032C
pwc	Sotec Afina Eye	04CC:8116
pwc	Samsung MPC-C10	055D:9000
pwc	Samsung MPC-C30	055D:9001
pwc	Samsung SNC-35E (Ver3.0)	055D:9002
pwc	Askey VC010 type 1	069A:0001
pwc	AME Co. Afina Eye	06BE:8116
pwc	Visionite VCS-UC300	0d81:1900
pwc	Visionite VCS-UM100	0d81:1910
s2255drv	Sensoray 2255	1943:2255, 1943:225
stk1160	STK1160 USB video capture dongle	05e1:0408
dvb-ttusb-budget	Technotrend/Hauppauge Nova-USB devices	0b48:1003, 0b48:100
dvb-ttusb_dec	Technotrend/Hauppauge MPEG decoder DEC3000-s	0b48:1006
dvb-ttusb_dec	Technotrend/Hauppauge MPEG decoder	0b48:1007
dvb-ttusb_dec	Technotrend/Hauppauge MPEG decoder DEC2000-t	0b48:1008
dvb-ttusb_dec	Technotrend/Hauppauge MPEG decoder DEC2540-t	0b48:1009
usbtv	Fushicai USBTV007 Audio-Video Grabber	1b71:3002, 1f71:330

59.6.2 PCI drivers

The PCI boards are identified by an identification called PCI ID. The PCI ID is actually composed by two parts:

- Vendor ID and device ID;
- Subsystem ID and Subsystem device ID;

The `lspci -nn` command allows identifying the vendor/device PCI IDs:

```
$ lspci -nn
...
00:0a.0 Multimedia controller [0480]: Philips Semiconductors SAA7131/SAA7133/
↪SAA7135 Video Broadcast Decoder [1131:7133] (rev d1)
00:0b.0 Multimedia controller [0480]: Brooktree Corporation Bt878 Audio␣
↪Capture [109e:0878] (rev 11)
01:00.0 Multimedia video controller [0400]: Conexant Systems, Inc. CX23887/8␣
↪PCIe Broadcast Audio and Video Decoder with 3D Comb [14f1:8880] (rev 0f)
02:01.0 Multimedia video controller [0400]: Internext Compression Inc iTVC15␣
↪(CX23415) Video Decoder [4444:0803] (rev 01)
02:02.0 Multimedia video controller [0400]: Conexant Systems, Inc. CX23418␣
↪Single-Chip MPEG-2 Encoder with Integrated Analog Video/Broadcast Audio␣
↪Decoder [14f1:5b7a]
02:03.0 Multimedia video controller [0400]: Brooktree Corporation Bt878 Video␣
↪Capture [109e:036e] (rev 11)
...
```

The subsystem IDs can be obtained using `lspci -vn`

```
$ lspci -vn
...
    00:0a.0 0480: 1131:7133 (rev d1)
            Subsystem: 1461:f01d
            Flags: bus master, medium devsel, latency 32, IRQ 209
            Memory at e2002000 (32-bit, non-prefetchable) [size=2K]
            Capabilities: [40] Power Management version 2
...
```

At the above example, the first card uses the `saa7134` driver, and has a vendor/device PCI ID equal to `1131:7133` and a PCI subsystem ID equal to `1461:f01d` (see *Saa7134 card list*).

Unfortunately, sometimes the same PCI subsystem ID is used by different products. So, several media drivers allow passing a `card=` parameter, in order to setup a card number that would match the correct settings for an specific board.

The current supported PCI/PCIe cards (not including staging drivers) are listed below[1].

Driver	Name
altera-ci	Altera FPGA based CI module
b2c2-flexcop-pci	Technisat/B2C2 Air/Sky/Cable2PC PCI
bt878	DVB/ATSC Support for bt878 based TV cards
bttv	BT8x8 Video For Linux
cobalt	Cisco Cobalt
cx18	Conexant cx23418 MPEG encoder
cx23885	Conexant cx23885 (2388x successor)
cx25821	Conexant cx25821
cx88xx	Conexant 2388x (bt878 successor)
ddbridge	Digital Devices bridge
dm1105	SDMC DM1105 based PCI cards

continues on next page

[1] some of the drivers have sub-drivers, not shown at this table

Table 9 – continued from previous page

Driver	Name
dt3155	DT3155 frame grabber
dvb-ttpci	AV7110 cards
earth-pt1	PT1 cards
earth-pt3	Earthsoft PT3 cards
hexium_gemini	Hexium Gemini frame grabber
hexium_orion	Hexium HV-PCI6 and Orion frame grabber
hopper	HOPPER based cards
ipu3-cio2	Intel ipu3-cio2 driver
ivtv	Conexant cx23416/cx23415 MPEG encoder/decoder
ivtvfb	Conexant cx23415 framebuffer
mantis	MANTIS based cards
mgb4	Digiteq Automotive MGB4 frame grabber
mxb	Siemens-Nixdorf 'Multimedia eXtension Board'
netup-unidvb	NetUP Universal DVB card
ngene	Micronas nGene
pluto2	Pluto2 cards
saa7134	Philips SAA7134
saa7164	NXP SAA7164
smipcie	SMI PCIe DVBSky cards
solo6x10	Bluecherry / Softlogic 6x10 capture cards (MPEG-4/H.264)
sta2x11_vip	STA2X11 VIP Video For Linux
tw5864	Techwell TW5864 video/audio grabber and encoder
tw686x	Intersil/Techwell TW686x
tw68	Techwell tw68x Video For Linux
zoran	Zoran-36057/36067 JPEG codec

Some of those drivers support multiple devices, as shown at the card lists below:

BTTV cards list

Card number	Card name	PCI subsystem IDs
0	*** UNKNOWN/GENERIC ***	
1	MIRO PCTV	
2	Hauppauge (bt848)	
3	STB, Gateway P/N 6000699 (bt848)	
4	Intel Create and Share PCI/ Smart Video Recorder III	
5	Diamond DTV2000	
6	AVerMedia TVPhone	
7	MATRIX-Vision MV-Delta	
8	Lifeview FlyVideo II (Bt848) LR26 / MAXI TV Video PCI2 LR26	
9	IMS/IXmicro TurboTV	

continues on next page

Table 10 – continued from previous page

Card number	Card name	PCI subsystem IDs
10	Hauppauge (bt878)	0070:13eb, 0070:3900, 2636:10b4
11	MIRO PCTV pro	
12	ADS Technologies Channel Surfer TV (bt848)	
13	AVerMedia TVCapture 98	1461:0002, 1461:0004, 1461:0300
14	Aimslab Video Highway Xtreme (VHX)	
15	Zoltrix TV-Max	a1a0:a0fc
16	Prolink Pixelview PlayTV (bt878)	
17	Leadtek WinView 601	
18	AVEC Intercapture	
19	Lifeview FlyVideo II EZ /FlyKit LR38 Bt848 (capture only)	
20	CEI Raffles Card	
21	Lifeview FlyVideo 98/ Lucky Star Image World ConferenceTV LR50	
22	Askey CPH050/ Phoebe Tv Master + FM	14ff:3002
23	Modular Technology MM201/MM202/MM205/MM210/MM215 PCTV, bt878	15c7:0101
24	Askey CPH05X/06X (bt878) [many vendors]	144f:3002, 144f:3005, 144f:5000, 14ff:3000
25	Terratec TerraTV+ Version 1.0 (Bt848)/ Terra TValue Version 1.0/ Vobis TV-Boostar	
26	Hauppauge WinCam newer (bt878)	
27	Lifeview FlyVideo 98/ MAXI TV Video PCI2 LR50	
28	Terratec TerraTV+ Version 1.1 (bt878)	153b:1127, 1852:1852
29	Imagenation PXC200	1295:200a
30	Lifeview FlyVideo 98 LR50	1f7f:1850
31	Formac iProTV, Formac ProTV I (bt848)	
32	Intel Create and Share PCI/ Smart Video Recorder III	
33	Terratec TerraTValue Version Bt878	153b:1117, 153b:1118, 153b:1119, 153b:111a, 153b:1134, 153b:5018
34	Leadtek WinFast 2000/ WinFast 2000 XP	107d:6606, 107d:6609, 6606:217d, f6ff:fff6
35	Lifeview FlyVideo 98 LR50 / Chronos Video Shuttle II	1851:1850, 1851:a050

continues on next page

Table 10 – continued from previous page

Card number	Card name	PCI subsystem IDs
36	Lifeview FlyVideo 98FM LR50 / Typhoon TView TV/FM Tuner	1852:1852
37	Prolink PixelView PlayTV pro	
38	Askey CPH06X TView99	144f:3000, 144f:a005, a04f:a0fc
39	Pinnacle PCTV Studio/Rave	11bd:0012, bd11:1200, bd11:ff00, 11bd:ff12
40	STB TV PCI FM, Gateway P/N 6000704 (bt878), 3Dfx VoodooTV 100	10b4:2636, 10b4:2645, 121a:3060
41	AVerMedia TVPhone 98	1461:0001, 1461:0003
42	ProVideo PV951	aa0c:146c
43	Little OnAir TV	
44	Sigma TVII-FM	
45	MATRIX-Vision MV-Delta 2	
46	Zoltrix Genie TV/FM	15b0:4000, 15b0:400a, 15b0:400d, 15b0:4010, 15b0:4016
47	Terratec TV/Radio+	153b:1123
48	Askey CPH03x/ Dynalink Magic TView	
49	IODATA GV-BCTV3/PCI	10fc:4020
50	Prolink PV-BT878P+4E / PixelView PlayTV PAK / Lenco MXTV-9578 CP	
51	Eagle Wireless Capricorn2 (bt878A)	
52	Pinnacle PCTV Studio Pro	
53	Typhoon TView RDS + FM Stereo / KNC1 TV Station RDS	
54	Lifeview FlyVideo 2000 /FlyVideo A2/ Lifetec LT 9415 TV [LR90]	
55	Askey CPH031/ BESTBUY Easy TV	
56	Lifeview FlyVideo 98FM LR50	a051:41a0
57	GrandTec 'Grand Video Capture' (Bt848)	4344:4142
58	Askey CPH060/ Phoebe TV Master Only (No FM)	
59	Askey CPH03x TV Capturer	
60	Modular Technology MM100PCTV	
61	AG Electronics GMV1	15cb:0101
62	Askey CPH061/ BESTBUY Easy TV (bt878)	
63	ATI TV-Wonder	1002:0001
64	ATI TV-Wonder VE	1002:0003
65	Lifeview FlyVideo 2000S LR90	
66	Terratec TValueRadio	153b:1135, 153b:ff3b
67	IODATA GV-BCTV4/PCI	10fc:4050

continues on next page

Table 10 – continued from previous page

Card number	Card name	PCI subsystem IDs
68	3Dfx VoodooTV FM (Euro)	10b4:2637
69	Active Imaging AIMMS	
70	Prolink Pixelview PV-BT878P+ (Rev.4C,8E)	
71	Lifeview FlyVideo 98EZ (capture only) LR51	1851:1851
72	Prolink Pixelview PV-BT878P+9B (PlayTV Pro rev.9B FM+NICAM)	1554:4011
73	Sensoray 311/611	6000:0311, 6000:0611
74	RemoteVision MX (RV605)	
75	Powercolor MTV878/ MTV878R/ MTV878F	
76	Canopus WinDVR PCI (COMPAQ Presario 3524JP, 5112JP)	0e11:0079
77	GrandTec Multi Capture Card (Bt878)	
78	Jetway TV/Capture JW-TV878-FBK, Kworld KW-TV878RF	0a01:17de
79	DSP Design TCVIDEO	
80	Hauppauge WinTV PVR	0070:4500
81	IODATA GV-BCTV5/PCI	10fc:4070, 10fc:d018
82	Osprey 100/150 (878)	0070:ff00
83	Osprey 100/150 (848)	
84	Osprey 101 (848)	
85	Osprey 101/151	
86	Osprey 101/151 w/ svid	
87	Osprey 200/201/250/251	
88	Osprey 200/250	0070:ff01
89	Osprey 210/220/230	
90	Osprey 500	0070:ff02
91	Osprey 540	0070:ff04
92	Osprey 2000	0070:ff03
93	IDS Eagle	
94	Pinnacle PCTV Sat	11bd:001c
95	Formac ProTV II (bt878)	
96	MachTV	
97	Euresys Picolo	
98	ProVideo PV150	aa00:1460, aa01:1461, aa02:1462, aa03:1463, aa04:1464, aa05:1465, aa06:1466, aa07:1467
99	AD-TVK503	
100	Hercules Smart TV Stereo	
101	Pace TV & Radio Card	

continues on next page

Table 10 – continued from previous page

Card number	Card name	PCI subsystem IDs
102	IVC-200	0000:a155, 0001:a155, 0002:a155, 0003:a155, 0100:a155, 0101:a155, 0102:a155, 0103:a155, 0800:a155, 0801:a155, 0802:a155, 0803:a155
103	Grand X-Guard / Trust 814PCI	0304:0102
104	Nebula Electronics DigiTV	0071:0101
105	ProVideo PV143	aa00:1430, aa00:1431, aa00:1432, aa00:1433, aa03:1433
106	PHYTEC VD-009-X1 VD-011 MiniDIN (bt878)	
107	PHYTEC VD-009-X1 VD-011 Combi (bt878)	
108	PHYTEC VD-009 MiniDIN (bt878)	
109	PHYTEC VD-009 Combi (bt878)	
110	IVC-100	ff00:a132
111	IVC-120G	ff00:a182, ff01:a182, ff02:a182, ff03:a182, ff04:a182, ff05:a182, ff06:a182, ff07:a182, ff08:a182, ff09:a182, ff0a:a182, ff0b:a182, ff0c:a182, ff0d:a182, ff0e:a182, ff0f:a182
112	pcHDTV HD-2000 TV	7063:2000
113	Twinhan DST + clones	11bd:0026, 1822:0001, 270f:fc00, 1822:0026
114	Winfast VC100	107d:6607
115	Teppro TEV-560/InterVision IV-560	
116	SIMUS GVC1100	aa6a:82b2
117	NGS NGSTV+	
118	LMLBT4	
119	Tekram M205 PRO	
120	Conceptronic CONTVFMi	

continues on next page

Table 10 – continued from previous page

Card number	Card name	PCI subsystem IDs
121	Euresys Picolo Tetra	1805:0105, 1805:0106, 1805:0107, 1805:0108
122	Spirit TV Tuner	
123	AVerMedia AVerTV DVB-T 771	1461:0771
124	AverMedia AverTV DVB-T 761	1461:0761
125	MATRIX Vision Sigma-SQ	
126	MATRIX Vision Sigma-SLC	
127	APAC Viewcomp 878(AMAX)	
128	DViCO FusionHDTV DVB-T Lite	18ac:db10, 18ac:db11
129	V-Gear MyVCD	
130	Super TV Tuner	
131	Tibet Systems 'Progress DVR' CS16	
132	Kodicom 4400R (master)	
133	Kodicom 4400R (slave)	
134	Adlink RTV24	
135	DViCO FusionHDTV 5 Lite	18ac:d500
136	Acorp Y878F	9511:1540
137	Conceptronic CTVFMi v2	036e:109e
138	Prolink Pixelview PV-BT878P+ (Rev.2E)	
139	Prolink PixelView PlayTV MPEG2 PV-M4900	
140	Osprey 440	0070:ff07
141	Asound Skyeye PCTV	
142	Sabrent TV-FM (bttv version)	
143	Hauppauge ImpactVCB (bt878)	0070:13eb
144	MagicTV	
145	SSAI Security Video Interface	4149:5353
146	SSAI Ultrasound Video Interface	414a:5353
147	VoodooTV 200 (USA)	121a:3000
148	DViCO FusionHDTV 2	dbc0:d200
149	Typhoon TV-Tuner PCI (50684)	
150	Geovision GV-600	008a:763c
151	Kozumi KTV-01C	
152	Encore ENL TV-FM-2	1000:1801
153	PHYTEC VD-012 (bt878)	
154	PHYTEC VD-012-X1 (bt878)	
155	PHYTEC VD-012-X2 (bt878)	
156	IVCE-8784	0000:f050, 0001:f050, 0002:f050, 0003:f050
157	Geovision GV-800(S) (master)	800a:763d

continues on next page

Table 10 – continued from previous page

Card number	Card name	PCI subsystem IDs
158	Geovision GV-800(S) (slave)	800b:763d, 800c:763d, 800d:763d
159	ProVideo PV183	1830:1540, 1831:1540, 1832:1540, 1833:1540, 1834:1540, 1835:1540, 1836:1540, 1837:1540
160	Tongwei Video Technology TD-3116	f200:3116
161	Aposonic W-DVR	0279:0228
162	Adlink MPG24	
163	Bt848 Capture 14MHz	
164	CyberVision CV06 (SV)	
165	Kworld V-Stream Xpert TV PVR878	
166	PCI-8604PW	

CX18 cards list

Those cards are supported by cx18 driver:

- Hauppauge HVR-1600 (ESMT memory)
- Hauppauge HVR-1600 (Samsung memory)
- Compro VideoMate H900
- Yuan MPC718 MiniPCI DVB-T/Analog
- Conexant Raptor PAL/SECAM
- Toshiba Qosmio DVB-T/Analog
- Leadtek WinFast PVR2100
- Leadtek WinFast DVR3100
- GoTView PCI DVD3 Hybrid
- Hauppauge HVR-1600 (s5h1411/tda18271)

cx23885 cards list

Card number	Card name	PCI subsystem IDs
0	UNKNOWN/GENERIC	0070:3400
1	Hauppauge WinTV-HVR1800lp	0070:7600
2	Hauppauge WinTV-HVR1800	0070:7800, 0070:7801, 0070:7809
3	Hauppauge WinTV-HVR1250	0070:7911
4	DViCO FusionHDTV5 Express	18ac:d500
5	Hauppauge WinTV-HVR1500Q	0070:7790, 0070:7797
6	Hauppauge WinTV-HVR1500	0070:7710, 0070:7717
7	Hauppauge WinTV-HVR1200	0070:71d1, 0070:71d3
8	Hauppauge WinTV-HVR1700	0070:8101
9	Hauppauge WinTV-HVR1400	0070:8010
10	DViCO FusionHDTV7 Dual Express	18ac:d618
11	DViCO FusionHDTV DVB-T Dual Express	18ac:db78
12	Leadtek Winfast PxDVR3200 H	107d:6681
13	Compro VideoMate E650F	185b:e800
14	TurboSight TBS 6920	6920:8888
15	TeVii S470	d470:9022
16	DVBWorld DVB-S2 2005	0001:2005
17	NetUP Dual DVB-S2 CI	1b55:2a2c
18	Hauppauge WinTV-HVR1270	0070:2211
19	Hauppauge WinTV-HVR1275	0070:2215, 0070:221d, 0070:22f2
20	Hauppauge WinTV-HVR1255	0070:2251, 0070:22f1
21	Hauppauge WinTV-HVR1210	0070:2291, 0070:2295, 0070:2299, 0070:229d, 0070:22f0, 0070:22f3, 0070:22f4, 0070:22f5
22	Mygica X8506 DMB-TH	14f1:8651
23	Magic-Pro ProHDTV Extreme 2	14f1:8657
24	Hauppauge WinTV-HVR1850	0070:8541
25	Compro VideoMate E800	1858:e800
26	Hauppauge WinTV-HVR1290	0070:8551
27	Mygica X8558 PRO DMB-TH	14f1:8578
28	LEADTEK WinFast PxTV1200	107d:6f22
29	GoTView X5 3D Hybrid	5654:2390

continues on next page

Table 11 – continued from previous page

Card number	Card name	PCI subsystem IDs
30	NetUP Dual DVB-T/C-CI RF	1b55:e2e4
31	Leadtek Winfast PxDVR3200 H XC4000	107d:6f39
32	MPX-885	
33	Mygica X8502/X8507 ISDB-T	14f1:8502
34	TerraTec Cinergy T PCIe Dual	153b:117e
35	TeVii S471	d471:9022
36	Hauppauge WinTV-HVR1255	0070:2259
37	Prof Revolution DVB-S2 8000	8000:3034
38	Hauppauge WinTV-HVR4400/HVR5500	0070:c108, 0070:c138, 0070:c1f8
39	AVerTV Hybrid Express Slim HC81R	1461:d939
40	TurboSight TBS 6981	6981:8888
41	TurboSight TBS 6980	6980:8888
42	Leadtek Winfast PxPVR2200	107d:6f21
43	Hauppauge ImpactVCB-e	0070:7133, 0070:7137
44	DViCO FusionHDTV DVB-T Dual Express2	18ac:db98
45	DVBSky T9580	4254:9580
46	DVBSky T980C	4254:980c
47	DVBSky S950C	4254:950c
48	Technotrend TT-budget CT2-4500 CI	13c2:3013
49	DVBSky S950	4254:0950
50	DVBSky S952	4254:0952
51	DVBSky T982	4254:0982
52	Hauppauge WinTV-HVR5525	0070:f038
53	Hauppauge WinTV Starburst	0070:c12a
54	ViewCast 260e	1576:0260
55	ViewCast 460e	1576:0460
56	Hauppauge WinTV-QuadHD-DVB	0070:6a28, 0070:6b28
57	Hauppauge WinTV-QuadHD-ATSC	0070:6a18, 0070:6b18
58	Hauppauge WinTV-HVR-1265(161111)	0070:2a18
59	Hauppauge WinTV-Starburst2	0070:f02a
60	Hauppauge WinTV-QuadHD-DVB(885)	
61	Hauppauge WinTV-QuadHD-ATSC(885)	
62	AVerMedia CE310B	1461:3100

CX88 cards list

Card number	Card name	PCI subsystem IDs
0	UNKNOWN/GENERIC	
1	Hauppauge WinTV 34xxx models	0070:3400, 0070:3401
2	GDI Black Gold	14c7:0106, 14c7:0107
3	PixelView	1554:4811
4	ATI TV Wonder Pro	1002:00f8, 1002:00f9
5	Leadtek Winfast 2000XP Expert	107d:6611, 107d:6613
6	AverTV Studio 303 (M126)	1461:000b
7	MSI TV-@nywhere Master	1462:8606
8	Leadtek Winfast DV2000	107d:6620, 107d:6621
9	Leadtek PVR 2000	107d:663b, 107d:663c, 107d:6632, 107d:6630, 107d:6638, 107d:6631, 107d:6637, 107d:663d
10	IODATA GV-VCP3/PCI	10fc:d003
11	Prolink PlayTV PVR	
12	ASUS PVR-416	1043:4823, 1461:c111
13	MSI TV-@nywhere	
14	KWorld/VStream XPert DVB-T	17de:08a6
15	DViCO FusionHDTV DVB-T1	18ac:db00
16	KWorld LTV883RF	
17	DViCO FusionHDTV 3 Gold-Q	18ac:d810, 18ac:d800
18	Hauppauge Nova-T DVB-T	0070:9002, 0070:9001, 0070:9000
19	Conexant DVB-T reference design	14f1:0187
20	Provideo PV259	1540:2580
21	DViCO FusionHDTV DVB-T Plus	18ac:db10, 18ac:db11
22	pcHDTV HD3000 HDTV	7063:3000
23	digitalnow DNTV Live! DVB-T	17de:a8a6
24	Hauppauge WinTV 28xxx (Roslyn) models	0070:2801
25	Digital-Logic MICROSPACE Entertainment Center (MEC)	14f1:0342
26	IODATA GV/BCTV7E	10fc:d035
27	PixelView PlayTV Ultra Pro (Stereo)	

continues on next page

Table 12 – continued from previous page

Card number	Card name	PCI subsystem IDs
28	DViCO FusionHDTV 3 Gold-T	18ac:d820
29	ADS Tech Instant TV DVB-T PCI	1421:0334
30	TerraTec Cinergy 1400 DVB-T	153b:1166
31	DViCO FusionHDTV 5 Gold	18ac:d500
32	AverMedia UltraTV Media Center PCI 550	1461:8011
33	Kworld V-Stream Xpert DVD	
34	ATI HDTV Wonder	1002:a101
35	WinFast DTV1000-T	107d:665f
36	AVerTV 303 (M126)	1461:000a
37	Hauppauge Nova-S-Plus DVB-S	0070:9201, 0070:9202
38	Hauppauge Nova-SE2 DVB-S	0070:9200
39	KWorld DVB-S 100	17de:08b2, 1421:0341
40	Hauppauge WinTV-HVR1100 DVB-T/Hybrid	0070:9400, 0070:9402
41	Hauppauge WinTV-HVR1100 DVB-T/Hybrid (Low Profile)	0070:9800, 0070:9802
42	digitalnow DNTV Live! DVB-T Pro	1822:0025, 1822:0019
43	KWorld/VStream XPert DVB-T with cx22702	17de:08a1, 12ab:2300
44	DViCO FusionHDTV DVB-T Dual Digital	18ac:db50, 18ac:db54
45	KWorld HardwareMpegTV XPert	17de:0840, 1421:0305
46	DViCO FusionHDTV DVB-T Hybrid	18ac:db40, 18ac:db44
47	pcHDTV HD5500 HDTV	7063:5500
48	Kworld MCE 200 Deluxe	17de:0841
49	PixelView PlayTV P7000	1554:4813
50	NPG Tech Real TV FM Top 10	14f1:0842
51	WinFast DTV2000 H	107d:665e
52	Geniatech DVB-S	14f1:0084
53	Hauppauge WinTV-HVR3000 TriMode Analog/DVB-S/DVB-T	0070:1404, 0070:1400, 0070:1401, 0070:1402
54	Norwood Micro TV Tuner	
55	Shenzhen Tungsten Ages Tech TE-DTV-250 / Swann OEM	c180:c980
56	Hauppauge WinTV-HVR1300 DVB-T/Hybrid MPEG Encoder	0070:9600, 0070:9601, 0070:9602
57	ADS Tech Instant Video PCI	1421:0390
58	Pinnacle PCTV HD 800i	11bd:0051

continues on next page

Table 12 – continued from previous page

Card number	Card name	PCI subsystem IDs
59	DViCO FusionHDTV 5 PCI nano	18ac:d530
60	Pinnacle Hybrid PCTV	12ab:1788
61	Leadtek TV2000 XP Global	107d:6f18, 107d:6618, 107d:6619
62	PowerColor RA330	14f1:ea3d
63	Geniatech X8000-MT DVBT	14f1:8852
64	DViCO FusionHDTV DVB-T PRO	18ac:db30
65	DViCO FusionHDTV 7 Gold	18ac:d610
66	Prolink Pixelview MPEG 8000GT	1554:4935
67	Kworld PlusTV HD PCI 120 (ATSC 120)	17de:08c1
68	Hauppauge WinTV-HVR4000 DVB-S/S2/T/Hybrid	0070:6900, 0070:6904, 0070:6902
69	Hauppauge WinTV-HVR4000(Lite) DVB-S/S2	0070:6905, 0070:6906
70	TeVii S460 DVB-S/S2	d460:9022
71	Omicom SS4 DVB-S/S2 PCI	A044:2011
72	TBS 8920 DVB-S/S2	8920:8888
73	TeVii S420 DVB-S	d420:9022
74	Prolink Pixelview Global Extreme	1554:4976
75	PROF 7300 DVB-S/S2	B033:3033
76	SATTRADE ST4200 DVB-S/S2	b200:4200
77	TBS 8910 DVB-S	8910:8888
78	Prof 6200 DVB-S	b022:3022
79	Terratec Cinergy HT PCI MKII	153b:1177
80	Hauppauge WinTV-IR Only	0070:9290
81	Leadtek WinFast DTV1800 Hybrid	107d:6654
82	WinFast DTV2000 H rev. J	107d:6f2b
83	Prof 7301 DVB-S/S2	b034:3034
84	Samsung SMT 7020 DVB-S	18ac:dc00, 18ac:dccd
85	Twinhan VP-1027 DVB-S	1822:0023
86	TeVii S464 DVB-S/S2	d464:9022
87	Leadtek WinFast DTV2000 H PLUS	107d:6f42
88	Leadtek WinFast DTV1800 H (XC4000)	107d:6f38
89	Leadtek TV2000 XP Global (SC4100)	107d:6f36
90	Leadtek TV2000 XP Global (XC4100)	107d:6f43
91	NotOnlyTV LV3H	

IVTV cards list

Card number	Card name	PCI subsystem IDs
0	Hauppauge WinTV PVR-250	IVTV16 104d:813d
1	Hauppauge WinTV PVR-350	IVTV16 104d:813d
2	Hauppauge WinTV PVR-150	IVTV16 104d:813d
3	AVerMedia M179	IVTV15 1461:a3cf, IVTV15 1461:a3ce
4	Yuan MPG600, Kuroutoshikou ITVC16-STVLP	IVTV16 12ab:fff3, IVTV16 12ab:ffff
5	YUAN MPG160, Kuroutoshikou ITVC15-STVLP, I/O Data GV-M2TV/PCI	IVTV15 10fc:40a0
6	Yuan PG600, Diamond PVR-550	IVTV16 ff92:0070, IVTV16 ffab:0600
7	Adaptec VideOh! AVC-2410	IVTV16 9005:0093
8	Adaptec VideOh! AVC-2010	IVTV16 9005:0092
9	Nagase Transgear 5000TV	IVTV16 1461:bfff
10	AOpen VA2000MAX-SNT6	IVTV16 0000:ff5f
11	Yuan MPG600GR, Kuroutoshikou CX23416GYC-STVLP	IVTV16 12ab:0600, IVTV16 fbab:0600, IVTV16 1154:0523
12	I/O Data GV-MVP/RX, GV-MVP/RX2W (dual tuner)	IVTV16 10fc:d01e, IVTV16 10fc:d038, IVTV16 10fc:d039
13	I/O Data GV-MVP/RX2E	IVTV16 10fc:d025
14	GotView PCI DVD	IVTV16 12ab:0600
15	GotView PCI DVD2 Deluxe	IVTV16 ffac:0600
16	Yuan MPC622	IVTV16 ff01:d998
17	Digital Cowboy DCT-MTVP1	IVTV16 1461:bfff

continues on next page

Table 13 – continued from previous page

Card number	Card name	PCI subsystem IDs
18	Yuan PG600-2, GotView PCI DVD Lite	IVTV16 ffab:0600, IVTV16 ffad:0600
19	Club3D ZAP-TV1x01	IVTV16 ffab:0600
20	AVerTV MCE 116 Plus	IVTV16 1461:c439
21	ASUS Falcon2	IVTV16 1043:4b66, IVTV16 1043:462e, IVTV16 1043:4b2e
22	AVerMedia PVR-150 Plus / AVerTV M113 Partsnic (Daewoo) Tuner	IVTV16 1461:c034, IVTV16 1461:c035
23	AVerMedia EZMaker PCI Deluxe	IVTV16 1461:c03f
24	AVerMedia M104	IVTV16 1461:c136
25	Buffalo PC-MV5L/PCI	IVTV16 1154:052b
26	AVerMedia UltraTV 1500 MCE / AVerTV M113 Philips Tuner	IVTV16 1461:c019, IVTV16 1461:c01b
27	Sony VAIO Giga Pocket (ENX Kikyou)	IVTV16 104d:813d
28	Hauppauge WinTV PVR-350 (V1)	IVTV16 104d:813d
29	Yuan MPG600GR, Kuroutoshikou CX23416GYC-STVLP (no GR)	IVTV16 104d:813d
30	Yuan MPG600GR, Kuroutoshikou CX23416GYC-STVLP (no GR/YCS)	IVTV16 104d:813d

SAA7134 cards list

Card number	Card name	PCI subsystem IDs
0	UNKNOWN/GENERIC	
1	Proteus Pro [philips reference design]	1131:2001, 1131:2001

continues on next page

Table 14 – continued from previous page

Card number	Card name	PCI subsystem IDs
2	LifeView FlyVIDEO3000	5168:0138, 4e42:0138
3	LifeView/Typhoon FlyVIDEO2000	5168:0138, 4e42:0138
4	EMPRESS	1131:6752
5	SKNet Monster TV	1131:4e85
6	Tevion MD 9717	
7	KNC One TV-Station RDS / Typhoon TV Tuner RDS	1131:fe01, 1894:fe01
8	Terratec Cinergy 400 TV	153b:1142
9	Medion 5044	
10	Kworld/KuroutoShikou SAA7130-TVPCI	
11	Terratec Cinergy 600 TV	153b:1143
12	Medion 7134	16be:0003, 16be:5000
13	Typhoon TV+Radio 90031	
14	ELSA EX-VISION 300TV	1048:226b
15	ELSA EX-VISION 500TV	1048:226a
16	ASUS TV-FM 7134	1043:4842, 1043:4830, 1043:4840
17	AOPEN VA1000 POWER	1131:7133
18	BMK MPEX No Tuner	
19	Compro VideoMate TV	185b:c100
20	Matrox CronosPlus	102B:48d0
21	10MOONS PCI TV CAPTURE CARD	1131:2001
22	AverMedia M156 / Medion 2819	1461:a70b
23	BMK MPEX Tuner	
24	KNC One TV-Station DVR	1894:a006
25	ASUS TV-FM 7133	1043:4843
26	Pinnacle PCTV Stereo (saa7134)	11bd:002b
27	Manli MuchTV M-TV002	
28	Manli MuchTV M-TV001	
29	Nagase Sangyo TransGear 3000TV	1461:050c
30	Elitegroup ECS TVP3XP FM1216 Tuner Card(PAL-BG,FM)	1019:4cb4
31	Elitegroup ECS TVP3XP FM1236 Tuner Card (NTSC,FM)	1019:4cb5
32	AVACS SmartTV	
33	AVerMedia DVD EZMaker	1461:10ff
34	Noval Prime TV 7133	
35	AverMedia AverTV Studio 305	1461:2115
36	UPMOST PURPLE TV	12ab:0800
37	Items MuchTV Plus / IT-005	
38	Terratec Cinergy 200 TV	153b:1152

continues on next page

Table 14 – continued from previous page

Card number	Card name	PCI subsystem IDs
39	LifeView FlyTV Platinum Mini	5168:0212, 4e42:0212, 5169:1502
40	Compro VideoMate TV PVR/FM	185b:c100
41	Compro VideoMate TV Gold+	185b:c100
42	Sabrent SBT-TVFM (saa7130)	
43	:Zolid Xpert TV7134	
44	Empire PCI TV-Radio LE	
45	Avermedia AVerTV Studio 307	1461:9715
46	AVerMedia Cardbus TV/Radio (E500)	1461:d6ee
47	Terratec Cinergy 400 mobile	153b:1162
48	Terratec Cinergy 600 TV MK3	153b:1158
49	Compro VideoMate Gold+ Pal	185b:c200
50	Pinnacle PCTV 300i DVB-T + PAL	11bd:002d
51	ProVideo PV952	1540:9524
52	AverMedia AverTV/305	1461:2108
53	ASUS TV-FM 7135	1043:4845
54	LifeView FlyTV Platinum FM / Gold	5168:0214, 5168:5214, 1489:0214, 5168:0304
55	LifeView FlyDVB-T DUO / MSI TV@nywhere Duo	5168:0306, 4E42:0306
56	Avermedia AVerTV 307	1461:a70a
57	Avermedia AVerTV GO 007 FM	1461:f31f
58	ADS Tech Instant TV (saa7135)	1421:0350, 1421:0351, 1421:0370, 1421:1370
59	Kworld/Tevion V-Stream Xpert TV PVR7134	
60	LifeView/Typhoon/Genius FlyDVB-T Duo Cardbus	5168:0502, 4e42:0502, 1489:0502
61	Philips TOUGH DVB-T reference design	1131:2004
62	Compro VideoMate TV Gold+II	
63	Kworld Xpert TV PVR7134	
64	FlyTV mini Asus Digimatrix	1043:0210
65	V-Stream Studio TV Terminator	
66	Yuan TUN-900 (saa7135)	
67	Beholder BeholdTV 409 FM	0000:4091
68	GoTView 7135 PCI	5456:7135
69	Philips EUROPA V3 reference design	1131:2004
70	Compro Videomate DVB-T300	185b:c900
71	Compro Videomate DVB-T200	185b:c901
72	RTD Embedded Technologies VFG7350	1435:7350

continues on next page

Table 14 – continued from previous page

Card number	Card name	PCI subsystem IDs
73	RTD Embedded Technologies VFG7330	1435:7330
74	LifeView FlyTV Platinum Mini2	14c0:1212
75	AVerMedia AVerTVHD MCE A180	1461:1044
76	SKNet MonsterTV Mobile	1131:4ee9
77	Pinnacle PCTV 40i/50i/110i (saa7133)	11bd:002e
78	ASUSTeK P7131 Dual	1043:4862
79	Sedna/MuchTV PC TV Cardbus TV/Radio (ITO25 Rev:2B)	
80	ASUS Digimatrix TV	1043:0210
81	Philips Tiger reference design	1131:2018
82	MSI TV@Anywhere plus	1462:6231, 1462:8624
83	Terratec Cinergy 250 PCI TV	153b:1160
84	LifeView FlyDVB Trio	5168:0319
85	AverTV DVB-T 777	1461:2c05, 1461:2c05
86	LifeView FlyDVB-T / Genius VideoWonder DVB-T	5168:0301, 1489:0301
87	ADS Instant TV Duo Cardbus PTV331	0331:1421
88	Tevion/KWorld DVB-T 220RF	17de:7201
89	ELSA EX-VISION 700TV	1048:226c
90	Kworld ATSC110/115	17de:7350, 17de:7352
91	AVerMedia A169 B	1461:7360
92	AVerMedia A169 B1	1461:6360
93	Medion 7134 Bridge #2	16be:0005
94	LifeView FlyDVB-T Hybrid Cardbus/MSI TV @nywhere A/D NB	5168:3306, 5168:3502, 5168:3307, 4e42:3502
95	LifeView FlyVIDEO3000 (NTSC)	5169:0138
96	Medion Md8800 Quadro	16be:0007, 16be:0008, 16be:000d
97	LifeView FlyDVB-S /Acorp TV134DS	5168:0300, 4e42:0300
98	Proteus Pro 2309	0919:2003
99	AVerMedia TV Hybrid A16AR	1461:2c00
100	Asus Europa2 OEM	1043:4860
101	Pinnacle PCTV 310i	11bd:002f
102	Avermedia AVerTV Studio 507	1461:9715
103	Compro Videomate DVB-T200A	

continues on next page

Table 14 – continued from previous page

Card number	Card name	PCI subsystem IDs
104	Hauppauge WinTV-HVR1110 DVB-T/Hybrid	0070:6700, 0070:6701, 0070:6702, 0070:6703, 0070:6704, 0070:6705
105	Terratec Cinergy HT PCMCIA	153b:1172
106	Encore ENLTV	1131:2342, 1131:2341, 3016:2344
107	Encore ENLTV-FM	1131:230f
108	Terratec Cinergy HT PCI	153b:1175
109	Philips Tiger - S Reference design	
110	Avermedia M102	1461:f31e
111	ASUS P7131 4871	1043:4871
112	ASUSTeK P7131 Hybrid	1043:4876
113	Elitegroup ECS TVP3XP FM1246 Tuner Card (PAL,FM)	1019:4cb6
114	KWorld DVB-T 210	17de:7250
115	Sabrent PCMCIA TV-PCB05	0919:2003
116	10MOONS TM300 TV Card	1131:2304
117	Avermedia Super 007	1461:f01d
118	Beholder BeholdTV 401	0000:4016
119	Beholder BeholdTV 403	0000:4036
120	Beholder BeholdTV 403 FM	0000:4037
121	Beholder BeholdTV 405	0000:4050
122	Beholder BeholdTV 405 FM	0000:4051
123	Beholder BeholdTV 407	0000:4070
124	Beholder BeholdTV 407 FM	0000:4071
125	Beholder BeholdTV 409	0000:4090
126	Beholder BeholdTV 505 FM	5ace:5050
127	Beholder BeholdTV 507 FM / BeholdTV 509 FM	5ace:5070, 5ace:5090
128	Beholder BeholdTV Columbus TV/FM	0000:5201
129	Beholder BeholdTV 607 FM	5ace:6070
130	Beholder BeholdTV M6	5ace:6190
131	Twinhan Hybrid DTV-DVB 3056 PCI	1822:0022
132	Genius TVGO AM11MCE	
133	NXP Snake DVB-S reference design	
134	Medion/Creatix CTX953 Hybrid	16be:0010
135	MSI TV@nywhere A/D v1.1	1462:8625
136	AVerMedia Cardbus TV/Radio (E506R)	1461:f436
137	AVerMedia Hybrid TV/Radio (A16D)	1461:f936
138	Avermedia M115	1461:a836
139	Compro VideoMate T750	185b:c900
140	Avermedia DVB-S Pro A700	1461:a7a1

continues on next page

Table 14 – continued from previous page

Card number	Card name	PCI subsystem IDs
141	Avermedia DVB-S Hybrid+FM A700	1461:a7a2
142	Beholder BeholdTV H6	5ace:6290
143	Beholder BeholdTV M63	5ace:6191
144	Beholder BeholdTV M6 Extra	5ace:6193
145	AVerMedia MiniPCI DVB-T Hybrid M103	1461:f636, 1461:f736
146	ASUSTeK P7131 Analog	
147	Asus Tiger 3in1	1043:4878
148	Encore ENLTV-FM v5.3	1a7f:2008
149	Avermedia PCI pure analog (M135A)	1461:f11d
150	Zogis Real Angel 220	
151	ADS Tech Instant HDTV	1421:0380
152	Asus Tiger Rev:1.00	1043:4857
153	Kworld Plus TV Analog Lite PCI	17de:7128
154	Avermedia AVerTV GO 007 FM Plus	1461:f31d
155	Hauppauge WinTV-HVR1150 ATSC/QAM-Hybrid	0070:6706, 0070:6708
156	Hauppauge WinTV-HVR1120 DVB-T/Hybrid	0070:6707, 0070:6709, 0070:670a
157	Avermedia AVerTV Studio 507UA	1461:a11b
158	AVerMedia Cardbus TV/Radio (E501R)	1461:b7e9
159	Beholder BeholdTV 505 RDS	0000:505B
160	Beholder BeholdTV 507 RDS	0000:5071
161	Beholder BeholdTV 507 RDS	0000:507B
162	Beholder BeholdTV 607 FM	5ace:6071
163	Beholder BeholdTV 609 FM	5ace:6090
164	Beholder BeholdTV 609 FM	5ace:6091
165	Beholder BeholdTV 607 RDS	5ace:6072
166	Beholder BeholdTV 607 RDS	5ace:6073
167	Beholder BeholdTV 609 RDS	5ace:6092
168	Beholder BeholdTV 609 RDS	5ace:6093
169	Compro VideoMate S350/S300	185b:c900
170	AverMedia AverTV Studio 505	1461:a115
171	Beholder BeholdTV X7	5ace:7595
172	RoverMedia TV Link Pro FM	19d1:0138
173	Zolid Hybrid TV Tuner PCI	1131:2004
174	Asus Europa Hybrid OEM	1043:4847
175	Leadtek Winfast DTV1000S	107d:6655
176	Beholder BeholdTV 505 RDS	0000:5051
177	Hawell HW-404M7	
178	Beholder BeholdTV H7	5ace:7190
179	Beholder BeholdTV A7	5ace:7090
180	Avermedia PCI M733A	1461:4155, 1461:4255
181	TechoTrend TT-budget T-3000	13c2:2804

continues on next page

Table 14 – continued from previous page

Card number	Card name	PCI subsystem IDs
182	Kworld PCI SBTVD/ISDB-T Full-Seg Hybrid	17de:b136
183	Compro VideoMate Vista M1F	185b:c900
184	Encore ENLTV-FM 3	1a7f:2108
185	MagicPro ProHDTV Pro2 DMB-TH/Hybrid	17de:d136
186	Beholder BeholdTV 501	5ace:5010
187	Beholder BeholdTV 503 FM	5ace:5030
188	Sensoray 811/911	6000:0811, 6000:0911
189	Kworld PC150-U	17de:a134
190	Asus My Cinema PS3-100	1043:48cd
191	Hawell HW-9004V1	
192	AverMedia AverTV Satellite Hybrid+FM A706	1461:2055
193	WIS Voyager or compatible	1905:7007
194	AverMedia AverTV/505	1461:a10a
195	Leadtek Winfast TV2100 FM	107d:6f3a
196	SnaZio* TVPVR PRO	1779:13cf

SAA7164 cards list

Card number	Card name	PCI subsystem IDs
0	Unknown	
1	Generic Rev2	
2	Generic Rev3	
3	Hauppauge WinTV-HVR2250	0070:8880, 0070:8810
4	Hauppauge WinTV-HVR2200	0070:8980
5	Hauppauge WinTV-HVR2200	0070:8900
6	Hauppauge WinTV-HVR2200	0070:8901
7	Hauppauge WinTV-HVR2250	0070:8891, 0070:8851
8	Hauppauge WinTV-HVR2250	0070:88A1
9	Hauppauge WinTV-HVR2200	0070:8940
10	Hauppauge WinTV-HVR2200	0070:8953
11	Hauppauge WinTV-HVR2255(proto)	0070:f111
12	Hauppauge WinTV-HVR2255	0070:f111
13	Hauppauge WinTV-HVR2205	0070:f123, 0070:f120

Zoran cards list

Card number	Card name	PCI subsystem IDs
0	DC10(old)	<any>
1	DC10(new)	<any>
2	DC10_PLUS	1031:7efe
3	DC30	<any>
4	DC30_PLUS	1031:d801
5	LML33	<any>
6	LML33R10	12f8:8a02
7	Buz	13ca:4231
8	6-Eyes	<any>

59.6.3 Platform drivers

There are several drivers that are focused on providing support for functionality that are already included at the main board, and don't use neither USB nor PCI bus. Those drivers are called platform drivers, and are very popular on embedded devices.

The current supported of platform drivers (not including staging drivers) are listed below

Driver	Name
am437x-vpfe	TI AM437x VPFE
aspeed-video	Aspeed AST2400 and AST2500
atmel-isc	ATMEL Image Sensor Controller (ISC)
atmel-isi	ATMEL Image Sensor Interface (ISI)
c8sectpfe	SDR platform devices
c8sectpfe	SDR platform devices
cafe_ccic	Marvell 88ALP01 (Cafe) CMOS Camera Controller
cdns-csi2rx	Cadence MIPI-CSI2 RX Controller
cdns-csi2tx	Cadence MIPI-CSI2 TX Controller
coda-vpu	Chips&Media Coda multi-standard codec IP
dm355_ccdc	TI DM355 CCDC video capture
dm644x_ccdc	TI DM6446 CCDC video capture
exynos-fimc-is	EXYNOS4x12 FIMC-IS (Imaging Subsystem)
exynos-fimc-lite	EXYNOS FIMC-LITE camera interface
exynos-gsc	Samsung Exynos G-Scaler
exy	Samsung S5P/EXYNOS4 SoC series Camera Subsystem
imx-pxp	i.MX Pixel Pipeline (PXP)
isdf	TI DM365 ISIF video capture
mmp_camera	Marvell Armada 610 integrated camera controller
mtk_jpeg	Mediatek JPEG Codec
mtk-mdp	Mediatek MDP
mtk-vcodec-dec	Mediatek Video Codec
mtk-vpu	Mediatek Video Processor Unit
mx2_emmaprp	MX2 eMMa-PrP

continues on next page

Table 15 – continued from previous page

Driver	Name
omap3-isp	OMAP 3 Camera
omap-vout	OMAP2/OMAP3 V4L2-Display
pxa_camera	PXA27x Quick Capture Interface
qcom-camss	Qualcomm V4L2 Camera Subsystem
rcar-csi2	R-Car MIPI CSI-2 Receiver
rcar_drif	Renesas Digital Radio Interface (DRIF)
rcar-fcp	Renesas Frame Compression Processor
rcar_fdp1	Renesas Fine Display Processor
rcar_jpu	Renesas JPEG Processing Unit
rcar-vin	R-Car Video Input (VIN)
renesas-ceu	Renesas Capture Engine Unit (CEU)
rockchip-rga	Rockchip Raster 2d Graphic Acceleration Unit
s3c-camif	Samsung S3C24XX/S3C64XX SoC Camera Interface
s5p-csis	S5P/EXYNOS MIPI-CSI2 receiver (MIPI-CSIS)
s5p-fimc	S5P/EXYNOS4 FIMC/CAMIF camera interface
s5p-g2d	Samsung S5P and EXYNOS4 G2D 2d graphics accelerator
s5p-jpeg	Samsung S5P/Exynos3250/Exynos4 JPEG codec
s5p-mfc	Samsung S5P MFC Video Codec
sh_veu	SuperH VEU mem2mem video processing
sh_vou	SuperH VOU video output
stm32-dcmi	STM32 Digital Camera Memory Interface (DCMI)
stm32-dma2d	STM32 Chrom-Art Accelerator Unit
sun4i-csi	Allwinner A10 CMOS Sensor Interface Support
sun6i-csi	Allwinner V3s Camera Sensor Interface
sun8i-di	Allwinner Deinterlace
sun8i-rotate	Allwinner DE2 rotation
ti-cal	TI Memory-to-memory multimedia devices
ti-csc	TI DVB platform devices
ti-vpe	TI VPE (Video Processing Engine)
venus-enc	Qualcomm Venus V4L2 encoder/decoder
via-camera	VIAFB camera controller
video-mux	Video Multiplexer
vpif_display	TI DaVinci VPIF V4L2-Display
vpif_capture	TI DaVinci VPIF video capture
vsp1	Renesas VSP1 Video Processing Engine
xilinx-tpg	Xilinx Video Test Pattern Generator
xilinx-video	Xilinx Video IP (EXPERIMENTAL)
xilinx-vtc	Xilinx Video Timing Controller

MMC/SDIO DVB adapters

Driver	Name
smssdio	Siano SMS1xxx based MDTV via SDIO interface

59.6.4 Radio drivers

There is also support for pure AM/FM radio, and even for some FM radio transmitters:

Driver	Name
si4713	Silicon Labs Si4713 FM Radio Transmitter
radio-aztech	Aztech/Packard Bell Radio
radio-cadet	ADS Cadet AM/FM Tuner
radio-gemtek	GemTek Radio card (or compatible)
radio-maxiradio	Guillemot MAXI Radio FM 2000 radio
radio-miropcm20	miroSOUND PCM20 radio
radio-aimslab	AIMSlab RadioTrack (aka RadioReveal)
radio-rtrack2	AIMSlab RadioTrack II
saa7706h	SAA7706H Car Radio DSP
radio-sf16fmi	SF16-FMI/SF16-FMP/SF16-FMD Radio
radio-sf16fmr2	SF16-FMR2/SF16-FMD2 Radio
radio-shark	Griffin radioSHARK USB radio receiver
shark2	Griffin radioSHARK2 USB radio receiver
radio-si470x-common	Silicon Labs Si470x FM Radio Receiver
radio-si476x	Silicon Laboratories Si476x I2C FM Radio
radio-tea5764	TEA5764 I2C FM radio
tef6862	TEF6862 Car Radio Enhanced Selectivity Tuner
radio-terratec	TerraTec ActiveRadio ISA Standalone
radio-timb	Enable the Timberdale radio driver
radio-trust	Trust FM radio card
radio-typhoon	Typhoon Radio (a.k.a. EcoRadio)
radio-wl1273	Texas Instruments WL1273 I2C FM Radio
fm_drv	ISA radio devices
fm_drv	ISA radio devices
radio-zoltrix	Zoltrix Radio
dsbr100	D-Link/GemTek USB FM radio
radio-keene	Keene FM Transmitter USB
radio-ma901	Masterkit MA901 USB FM radio
radio-mr800	AverMedia MR 800 USB FM radio
radio-raremono	Thanko's Raremono AM/FM/SW radio
radio-si470x-usb	Silicon Labs Si470x FM Radio Receiver support with USB
radio-usb-si4713	Silicon Labs Si4713 FM Radio Transmitter support with USB

59.6.5 I^{2}C drivers

The I^{2}C (Inter-Integrated Circuit) bus is a three-wires bus used internally at the media cards for communication between different chips. While the bus is not visible to the Linux Kernel, drivers need to send and receive commands via the bus. The Linux Kernel driver abstraction has support to implement different drivers for each component inside an I^{2}C bus, as if the bus were visible to the main system board.

One of the problems with I^{2}C devices is that sometimes the same device may work with different I^{2}C hardware. This is common, for example, on devices that comes with a tuner for North America market, and another one for Europe. Some drivers have a `tuner=` modprobe parameter to allow using a different tuner number in order to address such issue.

The current supported of I^{2}C drivers (not including staging drivers) are listed below.

Audio decoders, processors and mixers

Driver	Name
cs3308	Cirrus Logic CS3308 audio ADC
cs5345	Cirrus Logic CS5345 audio ADC
cs53l32a	Cirrus Logic CS53L32A audio ADC
msp3400	Micronas MSP34xx audio decoders
sony-btf-mpx	Sony BTF's internal MPX
tda1997x	NXP TDA1997x HDMI receiver
tda7432	Philips TDA7432 audio processor
tda9840	Philips TDA9840 audio processor
tea6415c	Philips TEA6415C audio processor
tea6420	Philips TEA6420 audio processor
tlv320aic23b	Texas Instruments TLV320AIC23B audio codec
tvaudio	Simple audio decoder chips
uda1342	Philips UDA1342 audio codec
vp27smpx	Panasonic VP27's internal MPX
wm8739	Wolfson Microelectronics WM8739 stereo audio ADC
wm8775	Wolfson Microelectronics WM8775 audio ADC with input mixer

Audio/Video compression chips

Driver	Name
saa6752hs	Philips SAA6752HS MPEG-2 Audio/Video Encoder

Camera sensor devices

Driver	Name
ccs	MIPI CCS compliant camera sensors (also SMIA++ and SMIA)
et8ek8	ET8EK8 camera sensor
hi556	Hynix Hi-556 sensor
hi846	Hynix Hi-846 sensor
imx208	Sony IMX208 sensor
imx214	Sony IMX214 sensor
imx219	Sony IMX219 sensor
imx258	Sony IMX258 sensor
imx274	Sony IMX274 sensor
imx290	Sony IMX290 sensor
imx319	Sony IMX319 sensor
imx334	Sony IMX334 sensor
imx355	Sony IMX355 sensor
imx412	Sony IMX412 sensor
mt9m001	mt9m001
mt9m111	mt9m111, mt9m112 and mt9m131
mt9p031	Aptina MT9P031
mt9t112	Aptina MT9T111/MT9T112
mt9v011	Micron mt9v011 sensor
mt9v032	Micron MT9V032 sensor
mt9v111	Aptina MT9V111 sensor
ov13858	OmniVision OV13858 sensor
ov13b10	OmniVision OV13B10 sensor
ov2640	OmniVision OV2640 sensor
ov2659	OmniVision OV2659 sensor
ov2680	OmniVision OV2680 sensor
ov2685	OmniVision OV2685 sensor
ov5640	OmniVision OV5640 sensor
ov5645	OmniVision OV5645 sensor
ov5647	OmniVision OV5647 sensor
ov5670	OmniVision OV5670 sensor
ov5675	OmniVision OV5675 sensor
ov5695	OmniVision OV5695 sensor
ov6650	OmniVision OV6650 sensor
ov7251	OmniVision OV7251 sensor
ov7640	OmniVision OV7640 sensor
ov7670	OmniVision OV7670 sensor
ov772x	OmniVision OV772x sensor
ov7740	OmniVision OV7740 sensor

continues on next page

Table 17 – continued from previous page

Driver	Name
ov8856	OmniVision OV8856 sensor
ov9640	OmniVision OV9640 sensor
ov9650	OmniVision OV9650/OV9652 sensor
rj54n1cb0c	Sharp RJ54N1CB0C sensor
s5c73m3	Samsung S5C73M3 sensor
s5k4ecgx	Samsung S5K4ECGX sensor
s5k5baf	Samsung S5K5BAF sensor
s5k6a3	Samsung S5K6A3 sensor

Flash devices

Driver	Name
adp1653	ADP1653 flash
lm3560	LM3560 dual flash driver
lm3646	LM3646 dual flash driver

IR I2C driver

Driver	Name
ir-kbd-i2c	I2C module for IR

Lens drivers

Driver	Name
ad5820	AD5820 lens voice coil
ak7375	AK7375 lens voice coil
dw9714	DW9714 lens voice coil
dw9768	DW9768 lens voice coil
dw9807-vcm	DW9807 lens voice coil

Miscellaneous helper chips

Driver	Name
video-i2c	I2C transport video
m52790	Mitsubishi M52790 A/V switch
st-mipid02	STMicroelectronics MIPID02 CSI-2 to PARALLEL bridge
ths7303	THS7303/53 Video Amplifier

RDS decoders

Driver	Name
saa6588	SAA6588 Radio Chip RDS decoder

SDR tuner chips

Driver	Name
max2175	Maxim 2175 RF to Bits tuner

Video and audio decoders

Driver	Name
cx25840	Conexant CX2584x audio/video decoders
saa717x	Philips SAA7171/3/4 audio/video decoders

Video decoders

Driver	Name
adv7180	Analog Devices ADV7180 decoder
adv7183	Analog Devices ADV7183 decoder
adv748x	Analog Devices ADV748x decoder
adv7604	Analog Devices ADV7604 decoder
adv7842	Analog Devices ADV7842 decoder
bt819	BT819A VideoStream decoder
bt856	BT856 VideoStream decoder
bt866	BT866 VideoStream decoder
ks0127	KS0127 video decoder
ml86v7667	OKI ML86V7667 video decoder
saa7110	Philips SAA7110 video decoder
saa7115	Philips SAA7111/3/4/5 video decoders
tc358743	Toshiba TC358743 decoder
tvp514x	Texas Instruments TVP514x video decoder
tvp5150	Texas Instruments TVP5150 video decoder
tvp7002	Texas Instruments TVP7002 video decoder
tw2804	Techwell TW2804 multiple video decoder
tw9903	Techwell TW9903 video decoder
tw9906	Techwell TW9906 video decoder
tw9910	Techwell TW9910 video decoder
vpx3220	vpx3220a, vpx3216b & vpx3214c video decoders

Video encoders

Driver	Name
adv7170	Analog Devices ADV7170 video encoder
adv7175	Analog Devices ADV7175 video encoder
adv7343	ADV7343 video encoder
adv7393	ADV7393 video encoder
adv7511-v4l2	Analog Devices ADV7511 encoder
ak881x	AK8813/AK8814 video encoders
saa7127	Philips SAA7127/9 digital video encoders
saa7185	Philips SAA7185 video encoder
ths8200	Texas Instruments THS8200 video encoder

Video improvement chips

Driver	Name
upd64031a	NEC Electronics uPD64031A Ghost Reduction
upd64083	NEC Electronics uPD64083 3-Dimensional Y/C separation

Tuner drivers

Driver	Name
e4000	Elonics E4000 silicon tuner
fc0011	Fitipower FC0011 silicon tuner
fc0012	Fitipower FC0012 silicon tuner
fc0013	Fitipower FC0013 silicon tuner
fc2580	FCI FC2580 silicon tuner
it913x	ITE Tech IT913x silicon tuner
m88rs6000t	Montage M88RS6000 internal tuner
max2165	Maxim MAX2165 silicon tuner
mc44s803	Freescale MC44S803 Low Power CMOS Broadband tuners
msi001	Mirics MSi001
mt2060	Microtune MT2060 silicon IF tuner
mt2063	Microtune MT2063 silicon IF tuner
mt20xx	Microtune 2032 / 2050 tuners
mt2131	Microtune MT2131 silicon tuner
mt2266	Microtune MT2266 silicon tuner
mxl301rf	MaxLinear MxL301RF tuner
mxl5005s	MaxLinear MSL5005S silicon tuner
mxl5007t	MaxLinear MxL5007T silicon tuner
qm1d1b0004	Sharp QM1D1B0004 tuner
qm1d1c0042	Sharp QM1D1C0042 tuner
qt1010	Quantek QT1010 silicon tuner
r820t	Rafael Micro R820T silicon tuner
si2157	Silicon Labs Si2157 silicon tuner

continues on next page

Table 18 – continued from previous page

Driver	Name
tuner-types	Simple tuner support
tda18212	NXP TDA18212 silicon tuner
tda18218	NXP TDA18218 silicon tuner
tda18250	NXP TDA18250 silicon tuner
tda18271	NXP TDA18271 silicon tuner
tda827x	Philips TDA827X silicon tuner
tda8290	TDA 8290/8295 + 8275(a)/18271 tuner combo
tda9887	TDA 9885/6/7 analog IF demodulator
tea5761	TEA 5761 radio tuner
tea5767	TEA 5767 radio tuner
tua9001	Infineon TUA9001 silicon tuner
xc2028	XCeive xc2028/xc3028 tuners
xc4000	Xceive XC4000 silicon tuner
xc5000	Xceive XC5000 silicon tuner

Tuner cards list

Tuner number	Card name
0	Temic PAL (4002 FH5)
1	Philips PAL_I (FI1246 and compatibles)
2	Philips NTSC (FI1236,FM1236 and compatibles)
3	Philips (SECAM+PAL_BG) (FI1216MF, FM1216MF, FR1216MF)
4	NoTuner
5	Philips PAL_BG (FI1216 and compatibles)
6	Temic NTSC (4032 FY5)
7	Temic PAL_I (4062 FY5)
8	Temic NTSC (4036 FY5)
9	Alps HSBH1
10	Alps TSBE1
11	Alps TSBB5
12	Alps TSBE5
13	Alps TSBC5
14	Temic PAL_BG (4006FH5)
15	Alps TSCH6
16	Temic PAL_DK (4016 FY5)
17	Philips NTSC_M (MK2)
18	Temic PAL_I (4066 FY5)
19	Temic PAL* auto (4006 FN5)
20	Temic PAL_BG (4009 FR5) or PAL_I (4069 FR5)
21	Temic NTSC (4039 FR5)
22	Temic PAL/SECAM multi (4046 FM5)
23	Philips PAL_DK (FI1256 and compatibles)
24	Philips PAL/SECAM multi (FQ1216ME)
25	LG PAL_I+FM (TAPC-I001D)
26	LG PAL_I (TAPC-I701D)

continues on next page

Table 19 – continued from previous page

Tuner number	Card name
27	LG NTSC+FM (TPI8NSR01F)
28	LG PAL_BG+FM (TPI8PSB01D)
29	LG PAL_BG (TPI8PSB11D)
30	Temic PAL* auto + FM (4009 FN5)
31	SHARP NTSC_JP (2U5JF5540)
32	Samsung PAL TCPM9091PD27
33	MT20xx universal
34	Temic PAL_BG (4106 FH5)
35	Temic PAL_DK/SECAM_L (4012 FY5)
36	Temic NTSC (4136 FY5)
37	LG PAL (newer TAPC series)
38	Philips PAL/SECAM multi (FM1216ME MK3)
39	LG NTSC (newer TAPC series)
40	HITACHI V7-J180AT
41	Philips PAL_MK (FI1216 MK)
42	Philips FCV1236D ATSC/NTSC dual in
43	Philips NTSC MK3 (FM1236MK3 or FM1236/F)
44	Philips 4 in 1 (ATI TV Wonder Pro/Conexant)
45	Microtune 4049 FM5
46	Panasonic VP27s/ENGE4324D
47	LG NTSC (TAPE series)
48	Tenna TNF 8831 BGFF)
49	Microtune 4042 FI5 ATSC/NTSC dual in
50	TCL 2002N
51	Philips PAL/SECAM_D (FM 1256 I-H3)
52	Thomson DTT 7610 (ATSC/NTSC)
53	Philips FQ1286
54	Philips/NXP TDA 8290/8295 + 8275/8275A/18271
55	TCL 2002MB
56	Philips PAL/SECAM multi (FQ1216AME MK4)
57	Philips FQ1236A MK4
58	Ymec TVision TVF-8531MF/8831MF/8731MF
59	Ymec TVision TVF-5533MF
60	Thomson DTT 761X (ATSC/NTSC)
61	Tena TNF9533-D/IF/TNF9533-B/DF
62	Philips TEA5767HN FM Radio
63	Philips FMD1216ME MK3 Hybrid Tuner
64	LG TDVS-H06xF
65	Ymec TVF66T5-B/DFF
66	LG TALN series
67	Philips TD1316 Hybrid Tuner
68	Philips TUV1236D ATSC/NTSC dual in
69	Tena TNF 5335 and similar models
70	Samsung TCPN 2121P30A
71	Xceive xc2028/xc3028 tuner
72	Thomson FE6600
73	Samsung TCPG 6121P30A

continues on next page

Table 19 – continued from previous page

Tuner number	Card name
75	Philips TEA5761 FM Radio
76	Xceive 5000 tuner
77	TCL tuner MF02GIP-5N-E
78	Philips FMD1216MEX MK3 Hybrid Tuner
79	Philips PAL/SECAM multi (FM1216 MK5)
80	Philips FQ1216LME MK3 PAL/SECAM w/active loopthrough
81	Partsnic (Daewoo) PTI-5NF05
82	Philips CU1216L
83	NXP TDA18271
84	Sony BTF-Pxn01Z
85	Philips FQ1236 MK5
86	Tena TNF5337 MFD
87	Xceive 4000 tuner
88	Xceive 5000C tuner
89	Sony BTF-PG472Z PAL/SECAM
90	Sony BTF-PK467Z NTSC-M-JP
91	Sony BTF-PB463Z NTSC-M

Frontend drivers

Note:

1) There is no guarantee that every frontend driver works out of the box with every card, because of different wiring.
2) The demodulator chips can be used with a variety of tuner/PLL chips, and not all combinations are supported. Often the demodulator and tuner/PLL chip are inside a metal box for shielding, and the whole metal box has its own part number.

Common Interface (EN50221) controller drivers

Driver	Name
cxd2099	Sony CXD2099AR Common Interface driver
sp2	CIMaX SP2

ATSC (North American/Korean Terrestrial/Cable DTV) frontends

Driver	Name
au8522_dig	Auvitek AU8522 based DTV demod
au8522_decoder	Auvitek AU8522 based ATV demod
bcm3510	Broadcom BCM3510
lg2160	LG Electronics LG216x based
lgdt3305	LG Electronics LGDT3304 and LGDT3305 based
lgdt3306a	LG Electronics LGDT3306A based
lgdt330x	LG Electronics LGDT3302/LGDT3303 based
nxt200x	NxtWave Communications NXT2002/NXT2004 based
or51132	Oren OR51132 based
or51211	Oren OR51211 based
s5h1409	Samsung S5H1409 based
s5h1411	Samsung S5H1411 based

DVB-C (cable) frontends

Driver	Name
stv0297	ST STV0297 based
tda10021	Philips TDA10021 based
tda10023	Philips TDA10023 based
ves1820	VLSI VES1820 based

DVB-S (satellite) frontends

Driver	Name
cx24110	Conexant CX24110 based
cx24116	Conexant CX24116 based
cx24117	Conexant CX24117 based
cx24120	Conexant CX24120 based
cx24123	Conexant CX24123 based
ds3000	Montage Technology DS3000 based
mb86a16	Fujitsu MB86A16 based
mt312	Zarlink VP310/MT312/ZL10313 based
s5h1420	Samsung S5H1420 based
si21xx	Silicon Labs SI21XX based
stb6000	ST STB6000 silicon tuner
stv0288	ST STV0288 based
stv0299	ST STV0299 based
stv0900	ST STV0900 based
stv6110	ST STV6110 silicon tuner
tda10071	NXP TDA10071
tda10086	Philips TDA10086 based
tda8083	Philips TDA8083 based
tda8261	Philips TDA8261 based
tda826x	Philips TDA826X silicon tuner
ts2020	Montage Technology TS2020 based tuners
tua6100	Infineon TUA6100 PLL
cx24113	Conexant CX24113/CX24128 tuner for DVB-S/DSS
itd1000	Integrant ITD1000 Zero IF tuner for DVB-S/DSS
ves1x93	VLSI VES1893 or VES1993 based
zl10036	Zarlink ZL10036 silicon tuner
zl10039	Zarlink ZL10039 silicon tuner

DVB-T (terrestrial) frontends

Driver	Name
af9013	Afatech AF9013 demodulator
cx22700	Conexant CX22700 based
cx22702	Conexant cx22702 demodulator (OFDM)
cxd2820r	Sony CXD2820R
cxd2841er	Sony CXD2841ER
cxd2880	Sony CXD2880 DVB-T2/T tuner + demodulator
dib3000mb	DiBcom 3000M-B
dib3000mc	DiBcom 3000P/M-C
dib7000m	DiBcom 7000MA/MB/PA/PB/MC
dib7000p	DiBcom 7000PC
dib9000	DiBcom 9000
drxd	Micronas DRXD driver
ec100	E3C EC100
l64781	LSI L64781
mt352	Zarlink MT352 based
nxt6000	NxtWave Communications NXT6000 based
rtl2830	Realtek RTL2830 DVB-T
rtl2832	Realtek RTL2832 DVB-T
rtl2832_sdr	Realtek RTL2832 SDR
s5h1432	Samsung s5h1432 demodulator (OFDM)
si2168	Silicon Labs Si2168
sp8870	Spase sp8870 based
sp887x	Spase sp887x based
stv0367	ST STV0367 based
tda10048	Philips TDA10048HN based
tda1004x	Philips TDA10045H/TDA10046H based
zd1301_demod	ZyDAS ZD1301
zl10353	Zarlink ZL10353 based

Digital terrestrial only tuners/PLL

Driver	Name
dvb-pll	Generic I2C PLL based tuners
dib0070	DiBcom DiB0070 silicon base-band tuner
dib0090	DiBcom DiB0090 silicon base-band tuner

ISDB-S (satellite) & ISDB-T (terrestrial) frontends

Driver	Name
mn88443x	Socionext MN88443x
tc90522	Toshiba TC90522

ISDB-T (terrestrial) frontends

Driver	Name
dib8000	DiBcom 8000MB/MC
mb86a20s	Fujitsu mb86a20s
s921	Sharp S921 frontend

Multistandard (cable + terrestrial) frontends

Driver	Name
drxk	Micronas DRXK based
mn88472	Panasonic MN88472
mn88473	Panasonic MN88473
si2165	Silicon Labs si2165 based
tda18271c2dd	NXP TDA18271C2 silicon tuner

Multistandard (satellite) frontends

Driver	Name
m88ds3103	Montage Technology M88DS3103
mxl5xx	MaxLinear MxL5xx based tuner-demodulators
stb0899	STB0899 based
stb6100	STB6100 based tuners
stv090x	STV0900/STV0903(A/B) based
stv0910	STV0910 based
stv6110x	STV6110/(A) based tuners
stv6111	STV6111 based tuners

SEC control devices for DVB-S

Driver	Name
a8293	Allegro A8293
af9033	Afatech AF9033 DVB-T demodulator
ascot2e	Sony Ascot2E tuner
atbm8830	AltoBeam ATBM8830/8831 DMB-TH demodulator
drx39xyj	Micronas DRX-J demodulator
helene	Sony HELENE Sat/Ter tuner (CXD2858ER)
horus3a	Sony Horus3A tuner
isl6405	ISL6405 SEC controller
isl6421	ISL6421 SEC controller
isl6423	ISL6423 SEC controller
ix2505v	Sharp IX2505V silicon tuner
lgs8gl5	Silicon Legend LGS-8GL5 demodulator (OFDM)
lgs8gxx	Legend Silicon LGS8913/LGS8GL5/LGS8GXX DMB-TH demodulator
lnbh25	LNBH25 SEC controller
lnbh29	LNBH29 SEC controller
lnbp21	LNBP21/LNBH24 SEC controllers
lnbp22	LNBP22 SEC controllers
m88rs2000	M88RS2000 DVB-S demodulator and tuner
tda665x	TDA665x tuner

Tools to develop new frontends

Driver	Name
dvb_dummy_fe	Dummy frontend driver

59.6.6 Firewire driver

The media subsystem also provides a firewire driver for digital TV:

Driver	Name
firedtv	FireDTV and FloppyDTV

59.6.7 Test drivers

In order to test userspace applications, there's a number of virtual drivers, with provide test functionality, simulating real hardware devices:

Driver	Name
vicodec	Virtual Codec Driver
vim2m	Virtual Memory-to-Memory Driver
vimc	Virtual Media Controller Driver (VIMC)
vivid	Virtual Video Test Driver

59.7 Video4Linux (V4L) driver-specific documentation

59.7.1 The bttv driver

Release notes for bttv

You'll need at least these config options for bttv:

```
./scripts/config -e PCI
./scripts/config -m I2C
./scripts/config -m INPUT
./scripts/config -m MEDIA_SUPPORT
./scripts/config -e MEDIA_PCI_SUPPORT
./scripts/config -e MEDIA_ANALOG_TV_SUPPORT
./scripts/config -e MEDIA_DIGITAL_TV_SUPPORT
./scripts/config -e MEDIA_RADIO_SUPPORT
./scripts/config -e RC_CORE
./scripts/config -m VIDEO_BT848
```

If your board has digital TV, you'll also need:

```
./scripts/config -m DVB_BT8XX
```

In this case, please see *How to get the bt8xx cards working* for additional notes.

Make bttv work with your card

If you have bttv compiled and installed, just booting the Kernel should be enough for it to try probing it. However, depending on the model, the Kernel may require additional information about the hardware, as the device may not be able to provide such info directly to the Kernel.

If it doesn't bttv likely could not autodetect your card and needs some insmod options. The most important insmod option for bttv is "card=n" to select the correct card type. If you get video but no sound you've very likely specified the wrong (or no) card type. A list of supported cards is in *BTTV cards list*.

If bttv takes very long to load (happens sometimes with the cheap cards which have no tuner), try adding this to your modules configuration file (usually, it is either `/etc/modules.conf` or some file at `/etc/modules-load.d/`, but the actual place depends on your distribution):

```
options i2c-algo-bit bit_test=1
```

Some cards may require an extra firmware file to work. For example, for the WinTV/PVR you need one firmware file from its driver CD, called: `hcwamc.rbf`. It is inside a self-extracting zip file called `pvr45xxx.exe`. Just placing it at the `/etc/firmware` directory should be enough for it to be autoload during the driver's probing mode (e. g. when the Kernel boots or when the driver is manually loaded via `modprobe` command).

If your card isn't listed in *BTTV cards list* or if you have trouble making audio work, please read *Still doesn't work?*.

Autodetecting cards

bttv uses the PCI Subsystem ID to autodetect the card type. lspci lists the Subsystem ID in the second line, looks like this:

```
00:0a.0 Multimedia video controller: Brooktree Corporation Bt878 (rev 02)
        Subsystem: Hauppauge computer works Inc. WinTV/GO
        Flags: bus master, medium devsel, latency 32, IRQ 5
        Memory at e2000000 (32-bit, prefetchable) [size=4K]
```

only bt878-based cards can have a subsystem ID (which does not mean that every card really has one). bt848 cards can't have a Subsystem ID and therefore can't be autodetected. There is a list with the ID's at *BTTV cards list* (in case you are interested or want to mail patches with updates).

Still doesn't work?

I do NOT have a lab with 30+ different grabber boards and a PAL/NTSC/SECAM test signal generator at home, so I often can't reproduce your problems. This makes debugging very difficult for me.

If you have some knowledge and spare time, please try to fix this yourself (patches very welcome of course...) You know: The linux slogan is ”Do it yourself”.

There is a mailing list at http://vger.kernel.org/vger-lists.html#linux-media

If you have trouble with some specific TV card, try to ask there instead of mailing me directly. The chance that someone with the same card listens there is much higher...

For problems with sound: There are a lot of different systems used for TV sound all over the world. And there are also different chips which decode the audio signal. Reports about sound problems (”stereo doesn't work”) are pretty useless unless you include some details about your hardware and the TV sound scheme used in your country (or at least the country you are living in).

Modprobe options

Note: The following argument list can be outdated, as we might add more options if ever needed. In case of doubt, please check with `modinfo <module>`.

This command prints various information about a kernel module, among them a complete and up-to-date list of insmod options.

---

bttv

The bt848/878 (grabber chip) driver

insmod args:

```
    card=n              card type, see CARDLIST for a list.
    tuner=n             tuner type, see CARDLIST for a list.
    radio=0/1   card supports radio
```

```
    pll=0/1/2   pll settings

                    0: don't use PLL
                    1: 28 MHz crystal installed
                    2: 35 MHz crystal installed

    triton1=0/1     for Triton1 (+others) compatibility
    vsfx=0/1    yet another chipset bug compatibility bit
                    see README.quirks for details on these two.

    bigendian=n Set the endianness of the gfx framebuffer.
                    Default is native endian.
    fieldnr=0/1 Count fields.  Some TV descrambling software
                    needs this, for others it only generates
                    50 useless IRQs/sec.  default is 0 (off).
    autoload=0/1        autoload helper modules (tuner, audio).
                    default is 1 (on).
    bttv_verbose=0/1/2  verbose level (at insmod time, while
                    looking at the hardware).  default is 1.
    bttv_debug=0/1      debug messages (for capture).
                    default is 0 (off).
    irq_debug=0/1       irq handler debug messages.
                    default is 0 (off).
    gbuffers=2-32       number of capture buffers for mmap'ed capture.
                    default is 4.
    gbufsize=   size of capture buffers. default and
                    maximum value is 0x208000 (~2MB)
    no_overlay=0        Enable overlay on broken hardware.  There
                    are some chipsets (SIS for example) which
                    are known to have problems with the PCI DMA
                    push used by bttv.  bttv will disable overlay
                    by default on this hardware to avoid crashes.
                    With this insmod option you can override this.
    no_overlay=1        Disable overlay. It should be used by broken
                    hardware that doesn't support PCI2PCI direct
                    transfers.
    automute=0/1        Automatically mutes the sound if there is
                    no TV signal, on by default.  You might try
                    to disable this if you have bad input signal
                    quality which leading to unwanted sound
                    dropouts.
    chroma_agc=0/1      AGC of chroma signal, off by default.
    adc_crush=0/1       Luminance ADC crush, on by default.
    i2c_udelay=     Allow reduce I2C speed. Default is 5 usecs
                    (meaning 66,67 Kbps). The default is the
                    maximum supported speed by kernel bitbang
                    algorithm. You may use lower numbers, if I2C
                    messages are lost (16 is known to work on
                    all supported cards).
```

```
    bttv_gpio=0/1
    gpiomask=
    audioall=
    audiomux=
                    See Sound-FAQ for a detailed description.

remap, card, radio and pll accept up to four comma-separated arguments
(for multiple boards).
```

tuner

The tuner driver. You need this unless you want to use only with a camera or the board doesn't provide analog TV tuning.

insmod args:

```
debug=1         print some debug info to the syslog
type=n          type of the tuner chip. n as follows:
                see CARDLIST for a complete list.
pal=[bdgil]     select PAL variant (used for some tuners
                only, important for the audio carrier).
```

tvaudio

Provide a single driver for all simple i2c audio control chips (tda/tea*).

insmod args:

```
tda8425  = 1    enable/disable the support for the
tda9840  = 1    various chips.
tda9850  = 1    The tea6300 can't be autodetected and is
tda9855  = 1    therefore off by default, if you have
tda9873  = 1    this one on your card (STB uses these)
tda9874a = 1    you have to enable it explicitly.
tea6300  = 0    The two tda985x chips use the same i2c
tea6420  = 1    address and can't be disturgished from
pic16c54 = 1    each other, you might have to disable
                the wrong one.
debug = 1       print debug messages
```

msp3400

The driver for the msp34xx sound processor chips. If you have a stereo card, you probably want to insmod this one.

insmod args:

```
debug=1/2       print some debug info to the syslog,
                2 is more verbose.
simple=1        Use the "short programming" method.  Newer
                msp34xx versions support this.  You need this
                for dbx stereo.  Default is on if supported by
                the chip.
once=1          Don't check the TV-stations Audio mode
                every few seconds, but only once after
                channel switches.
```

```
amsound=1       Audio carrier is AM/NICAM at 6.5 Mhz.  This
                should improve things for french people, the
                carrier autoscan seems to work with FM only...
```

If the box freezes hard with bttv

It might be a bttv driver bug. It also might be bad hardware. It also might be something else ...

Just mailing me ”bttv freezes” isn't going to help much. This README has a few hints how you can help to pin down the problem.

bttv bugs

If some version works and another doesn't it is likely to be a driver bug. It is very helpful if you can tell where exactly it broke (i.e. the last working and the first broken version).

With a hard freeze you probably doesn't find anything in the logfiles. The only way to capture any kernel messages is to hook up a serial console and let some terminal application log the messages. /me uses screen. See *Linux Serial Console* for details on setting up a serial console.

Read *Bug hunting* to learn how to get any useful information out of a register+stack dump printed by the kernel on protection faults (so-called ”kernel oops”).

If you run into some kind of deadlock, you can try to dump a call trace for each process using sysrq-t (see *Linux Magic System Request Key Hacks*). This way it is possible to figure where *exactly* some process in ”D” state is stuck.

I've seen reports that bttv 0.7.x crashes whereas 0.8.x works rock solid for some people. Thus probably a small buglet left somewhere in bttv 0.7.x. I have no idea where exactly, it works stable for me and a lot of other people. But in case you have problems with the 0.7.x versions you can give 0.8.x a try ...

hardware bugs

Some hardware can't deal with PCI-PCI transfers (i.e. grabber => vga). Sometimes problems show up with bttv just because of the high load on the PCI bus. The bt848/878 chips have a few workarounds for known incompatibilities, see README.quirks.

Some folks report that increasing the pci latency helps too, althrought I'm not sure whenever this really fixes the problems or only makes it less likely to happen. Both bttv and btaudio have a insmod option to set the PCI latency of the device.

Some mainboard have problems to deal correctly with multiple devices doing DMA at the same time. bttv + ide seems to cause this sometimes, if this is the case you likely see freezes only with video and hard disk access at the same time. Updating the IDE driver to get the latest and greatest workarounds for hardware bugs might fix these problems.

other

If you use some binary-only yunk (like nvidia module) try to reproduce the problem without.

IRQ sharing is known to cause problems in some cases. It works just fine in theory and many configurations. Neverless it might be worth a try to shuffle around the PCI cards to give bttv another IRQ or make it share the IRQ with some other piece of hardware. IRQ sharing with VGA cards seems to cause trouble sometimes. I've also seen funny effects with bttv sharing the IRQ with the ACPI bridge (and apci-enabled kernel).

Bttv quirks

Below is what the bt878 data book says about the PCI bug compatibility modes of the bt878 chip.

The triton1 insmod option sets the EN_TBFX bit in the control register. The vsfx insmod option does the same for EN_VSFX bit. If you have stability problems you can try if one of these options makes your box work solid.

drivers/pci/quirks.c knows about these issues, this way these bits are enabled automagically for known-buggy chipsets (look at the kernel messages, bttv tells you).

Normal PCI Mode

The PCI REQ signal is the logical-or of the incoming function requests. The inter-nal GNT[0:1] signals are gated asynchronously with GNT and demultiplexed by the audio request signal. Thus the arbiter defaults to the video function at power-up and parks there during no requests for bus access. This is desirable since the video will request the bus more often. However, the audio will have highest bus access priority. Thus the audio will have first access to the bus even when issuing a request after the video request but before the PCI external arbiter has granted access to the Bt879. Neither function can preempt the other once on the bus. The duration to empty the entire video PCI FIFO onto the PCI bus is very short compared to the bus access latency the audio PCI FIFO can tolerate.

430FX Compatibility Mode

When using the 430FX PCI, the following rules will ensure compatibility:

(1) Deassert REQ at the same time as asserting FRAME.

(2) Do not reassert REQ to request another bus transaction until after finish-ing the previous transaction.

Since the individual bus masters do not have direct control of REQ, a simple logical-or of video and audio requests would violate the rules. Thus, both the arbiter and the initiator contain 430FX compatibility mode logic. To enable 430FX mode, set the EN_TBFX bit as indicated in Device Control Register on page 104.

When EN_TBFX is enabled, the arbiter ensures that the two compatibility rules are satisfied. Before GNT is asserted by the PCI arbiter, this internal arbiter may still logical-or the two requests. However, once the GNT is issued, this arbiter must lock in its decision and now route only the granted request to the REQ pin. The arbiter decision lock happens regardless of the

state of FRAME because it does not know when FRAME will be asserted (typically - each initiator will assert FRAME on the cycle following GNT). When FRAME is asserted, it is the initiator s responsibility to remove its request at the same time. It is the arbiters responsibility to allow this request to flow through to REQ and not allow the other request to hold REQ asserted. The decision lock may be removed at the end of the transaction: for example, when the bus is idle (FRAME and IRDY). The arbiter decision may then continue asynchronously until GNT is again asserted.

Interfacing with Non-PCI 2.1 Compliant Core Logic

A small percentage of core logic devices may start a bus transaction during the same cycle that GNT is de-asserted. This is non PCI 2.1 compliant. To ensure compatibility when using PCs with these PCI controllers, the EN_VSFX bit must be enabled (refer to Device Control Register on page 104). When in this mode, the arbiter does not pass GNT to the internal functions unless REQ is asserted. This prevents a bus transaction from starting the same cycle as GNT is de-asserted. This also has the side effect of not being able to take advantage of bus parking, thus lowering arbitration performance. The Bt879 drivers must query for these non-compliant devices, and set the EN_VSFX bit only if required.

Other elements of the tvcards array

If you are trying to make a new card work you might find it useful to know what the other elements in the tvcards array are good for:

```
video_inputs    - # of video inputs the card has
audio_inputs    - historical cruft, not used any more.
tuner           - which input is the tuner
svhs            - which input is svhs (all others are labeled composite)
muxsel          - video mux, input->registervalue mapping
pll             - same as pll= insmod option
tuner_type      - same as tuner= insmod option
*_modulename    - hint whenever some card needs this or that audio
                module loaded to work properly.
has_radio       - whenever this TV card has a radio tuner.
no_msp34xx      - "1" disables loading of msp3400.o module
no_tda9875      - "1" disables loading of tda9875.o module
needs_tvaudio   - set to "1" to load tvaudio.o module
```

If some config item is specified both from the tvcards array and as insmod option, the insmod option takes precedence.

Cards

Note: For a more updated list, please check https://linuxtv.org/wiki/index.php/Hardware_Device_Information

Supported cards: Bt848/Bt848a/Bt849/Bt878/Bt879 cards

All cards with Bt848/Bt848a/Bt849/Bt878/Bt879 and normal Composite/S-VHS inputs are supported. Teletext and Intercast support (PAL only) for ALL cards via VBI sample decoding in software.

Some cards with additional multiplexing of inputs or other additional fancy chips are only partially supported (unless specifications by the card manufacturer are given). When a card is listed here it isn't necessarily fully supported.

All other cards only differ by additional components as tuners, sound decoders, EEPROMs, teletext decoders ...

MATRIX Vision

MV-Delta - Bt848A - 4 Composite inputs, 1 S-VHS input (shared with 4th composite) - EEPROM

http://www.matrix-vision.de/

This card has no tuner but supports all 4 composite (1 shared with an S-VHS input) of the Bt848A. Very nice card if you only have satellite TV but several tuners connected to the card via composite.

Many thanks to Matrix-Vision for giving us 2 cards for free which made Bt848a/Bt849 single crystal operation support possible!!!

Miro/Pinnacle PCTV

- Bt848 some (all??) come with 2 crystals for PAL/SECAM and NTSC
- PAL, SECAM or NTSC TV tuner (Philips or TEMIC)
- MSP34xx sound decoder on add on board decoder is supported but AFAIK does not yet work (other sound MUX setting in GPIO port needed??? somebody who fixed this???)
- 1 tuner, 1 composite and 1 S-VHS input
- tuner type is autodetected

http://www.miro.de/ http://www.miro.com/

Many thanks for the free card which made first NTSC support possible back in 1997!

Hauppauge Win/TV pci

There are many different versions of the Hauppauge cards with different tuners (TV+Radio ...), teletext decoders. Note that even cards with same model numbers have (depending on the revision) different chips on it.

- Bt848 (and others but always in 2 crystal operation???) newer cards have a Bt878
- PAL, SECAM, NTSC or tuner with or without Radio support

e.g.:

- PAL:
  - TDA5737: VHF, hyperband and UHF mixer/oscillator for TV and VCR 3-band tuners
  - TSA5522: 1.4 GHz I2C-bus controlled synthesizer, I2C 0xc2-0xc3
- NTSC:
  - TDA5731: VHF, hyperband and UHF mixer/oscillator for TV and VCR 3-band tuners
  - TSA5518: no datasheet available on Philips site
- Philips SAA5246 or SAA5284 (or no) Teletext decoder chip with buffer RAM (e.g. Winbond W24257AS-35: 32Kx8 CMOS static RAM) SAA5246 (I2C 0x22) is supported
- 256 bytes EEPROM: Microchip 24LC02B or Philips 8582E2Y with configuration information I2C address 0xa0 (24LC02B also responds to 0xa2-0xaf)
- 1 tuner, 1 composite and (depending on model) 1 S-VHS input
- 14052B: mux for selection of sound source
- sound decoder: TDA9800, MSP34xx (stereo cards)

Askey CPH-Series

Developed by TelSignal(?), OEMed by many vendors (Typhoon, Anubis, Dynalink)

- Card series: - CPH01x: BT848 capture only - CPH03x: BT848 - CPH05x: BT878 with FM - CPH06x: BT878 (w/o FM) - CPH07x: BT878 capture only
- TV standards: - CPH0x0: NTSC-M/M - CPH0x1: PAL-B/G - CPH0x2: PAL-I/I - CPH0x3: PAL-D/K - CPH0x4: SECAM-L/L - CPH0x5: SECAM-B/G - CPH0x6: SECAM-D/K - CPH0x7: PAL-N/N - CPH0x8: PAL-B/H - CPH0x9: PAL-M/M
- CPH03x was often sold as ”TV capturer”.

Identifying:

1) 878 cards can be identified by PCI Subsystem-ID: - 144f:3000 = CPH06x - 144F:3002 = CPH05x w/ FM - 144F:3005 = CPH06x_LC (w/o remote control)
2) The cards have a sticker with ”CPH”-model on the back.
3) These cards have a number printed on the PCB just above the tuner metal box: - ”80-CP2000300-x” = CPH03X - ”80-CP2000500-x” = CPH05X - ”80-CP2000600-x” = CPH06X / CPH06x_LC

Askey sells these cards as ”Magic TView series”, Brand ”MagicXpress”. Other OEM often call these ”Tview”, ”TView99” or else.

Lifeview Flyvideo Series:

The naming of these series differs in time and space.

Identifying:

1) Some models can be identified by PCI subsystem ID:
   - 1852:1852 = Flyvideo 98 FM
   - 1851:1850 = Flyvideo 98
   - 1851:1851 = Flyvideo 98 EZ (capture only)
2) There is a print on the PCB:
   - LR25 = Flyvideo (Zoran ZR36120, SAA7110A)
   - LR26 Rev.N = Flyvideo II (Bt848)
   - LR26 Rev.O = Flyvideo II (Bt878)
   - LR37 Rev.C = Flyvideo EZ (Capture only, ZR36120 + SAA7110)
   - LR38 Rev.A1= Flyvideo II EZ (Bt848 capture only)
   - LR50 Rev.Q = Flyvideo 98 (w/eeprom and PCI subsystem ID)
   - LR50 Rev.W = Flyvideo 98 (no eeprom)
   - LR51 Rev.E = Flyvideo 98 EZ (capture only)
   - LR90 = Flyvideo 2000 (Bt878)
   - LR90 Flyvideo 2000S (Bt878) w/Stereo TV (Package incl. LR91 daughterboard)
   - LR91 = Stereo daughter card for LR90
   - LR97 = Flyvideo DVBS
   - LR99 Rev.E = Low profile card for OEM integration (only internal audio!) bt878
   - LR136 = Flyvideo 2100/3100 (Low profile, SAA7130/SAA7134)
   - LR137 = Flyvideo DV2000/DV3000 (SAA7130/SAA7134 + IEEE1394)
   - LR138 Rev.C= Flyvideo 2000 (SAA7130)
   - LR138 Flyvideo 3000 (SAA7134) w/Stereo TV
     - These exist in variations w/FM and w/Remote sometimes denoted by suffixes ”FM” and ”R”.
3) You have a laptop (miniPCI card):
   - Product = FlyTV Platinum Mini
   - Model/Chip = LR212/saa7135

- Lifeview.com.tw states (Feb. 2002): ”The FlyVideo2000 and FlyVideo2000s product name have renamed to FlyVideo98.” Their Bt8x8 cards are listed as discontinued.
- Flyvideo 2000S was probably sold as Flyvideo 3000 in some countries(Europe?). The new Flyvideo 2000/3000 are SAA7130/SAA7134 based.

”Flyvideo II” had been the name for the 848 cards, nowadays (in Germany) this name is re-used for LR50 Rev.W.

The Lifeview website mentioned Flyvideo III at some time, but such a card has not yet been seen (perhaps it was the german name for LR90 [stereo]). These cards are sold by many OEMs too.

FlyVideo A2 (Elta 8680)= LR90 Rev.F (w/Remote, w/o FM, stereo TV by tda9821) {Germany}

Lifeview 3000 (Elta 8681) as sold by Plus(April 2002), Germany = LR138 w/ saa7134

lifeview config coding on gpio pins 0-9

- LR50 rev. Q (”PARTS: 7031505116), Tuner wurde als Nr. 5 erkannt, Eingänge SVideo, TV, Composite, Audio, Remote:
- CP9..1=100001001 (1: 0-Ohm-Widerstand gegen GND unbestückt; 0: bestückt)

Typhoon TV card series:

These can be CPH, Flyvideo, Pixelview or KNC1 series.

Typhoon is the brand of Anubis.

Model 50680 got re-used, some model no. had different contents over time.

Models:

- 50680 ”TV Tuner PCI Pal BG”(old,red package)=can be CPH03x(bt848) or CPH06x(bt878)
- 50680 ”TV Tuner Pal BG” (blue package)= Pixelview PV-BT878P+ (Rev 9B)
- 50681 ”TV Tuner PCI Pal I” (variant of 50680)
- 50682 ”TView TV/FM Tuner Pal BG” = Flyvideo 98FM (LR50 Rev.Q)

Note: The package has a picture of CPH05x (which would be a real TView)

- 50683 ”TV Tuner PCI SECAM” (variant of 50680)
- 50684 ”TV Tuner Pal BG” = Pixelview 878TV(Rev.3D)
- 50686 ”TV Tuner” = KNC1 TV Station
- 50687 ”TV Tuner stereo” = KNC1 TV Station pro
- 50688 ”TV Tuner RDS” (black package) = KNC1 TV Station RDS

- 50689 TV SAT DVB-S CARD CI PCI (SAA7146AH, SU1278?) = ”KNC1 TV Station DVB-S”
- 50692 ”TV/FM Tuner” (small PCB)
- 50694 TV TUNER CARD RDS (PHILIPS CHIPSET SAA7134HL)
- 50696 TV TUNER STEREO (PHILIPS CHIPSET SAA7134HL, MK3ME Tuner)
- 50804 PC-SAT TV/Audio Karte = Techni-PC-Sat (ZORAN 36120PQC, Tuner:Alps)
- 50866 TVIEW SAT RECEIVER+ADR
- 50868 ”TV/FM Tuner Pal I” (variant of 50682)
- 50999 ”TV/FM Tuner Secam” (variant of 50682)

Guillemot

Models:

- Maxi-TV PCI (ZR36120)
- Maxi TV Video 2 = LR50 Rev.Q (FI1216MF, PAL BG+SECAM)
- Maxi TV Video 3 = CPH064 (PAL BG + SECAM)

Mentor

Mentor TV card (”55-878TV-U1”) = Pixelview 878TV(Rev.3F) (w/FM w/Remote)

Prolink

- TV cards:
  - PixelView Play TV pro - (Model: PV-BT878P+ REV 8E)
  - PixelView Play TV pro - (Model: PV-BT878P+ REV 9D)
  - PixelView Play TV pro - (Model: PV-BT878P+ REV 4C / 8D / 10A)
  - PixelView Play TV - (Model: PV-BT848P+)
  - 878TV - (Model: PV-BT878TV)
- Multimedia TV packages (card + software pack):
  - PixelView Play TV Theater - (Model: PV-M4200) = PixelView Play TV pro + Software
  - PixelView Play TV PAK - (Model: PV-BT878P+ REV 4E)
  - PixelView Play TV/VCR - (Model: PV-M3200 REV 4C / 8D / 10A)
  - PixelView Studio PAK - (Model: M2200 REV 4C / 8D / 10A)
  - PixelView PowerStudio PAK - (Model: PV-M3600 REV 4E)
  - PixelView DigitalVCR PAK - (Model: PV-M2400 REV 4C / 8D / 10A)
  - PixelView PlayTV PAK II (TV/FM card + usb camera) PV-M3800

  - PixelView PlayTV XP PV-M4700,PV-M4700(w/FM)
  - PixelView PlayTV DVR PV-M4600 package contents:PixelView PlayTV pro, windvr & videoMail s/w
- Further Cards:
  - PV-BT878P+rev.9B (Play TV Pro, opt. w/FM w/NICAM)
  - PV-BT878P+rev.2F
  - PV-BT878P Rev.1D (bt878, capture only)
  - XCapture PV-CX881P (cx23881)
  - PlayTV HD PV-CX881PL+, PV-CX881PL+(w/FM) (cx23881)
  - DTV3000 PV-DTV3000P+ DVB-S CI = Twinhan VP-1030
  - DTV2000 DVB-S = Twinhan VP-1020
- Video Conferencing:
  - PixelView Meeting PAK - (Model: PV-BT878P)
  - PixelView Meeting PAK Lite - (Model: PV-BT878P)
  - PixelView Meeting PAK plus - (Model: PV-BT878P+rev 4C/8D/10A)
  - PixelView Capture - (Model: PV-BT848P)
  - PixelView PlayTV USB pro
  - Model No. PV-NT1004+, PV-NT1004+ (w/FM) = NT1004 USB decoder chip + SAA7113 video decoder chip

Dynalink

These are CPH series.

Phoebemicro

- TV Master = CPH030 or CPH060
- TV Master FM = CPH050

Genius/Kye

- Video Wonder/Genius Internet Video Kit = LR37 Rev.C
- Video Wonder Pro II (848 or 878) = LR26

Tekram

- VideoCap C205 (Bt848)
- VideoCap C210 (zr36120 +Philips)
- CaptureTV M200 (ISA)
- CaptureTV M205 (Bt848)

Lucky Star

- Image World Conference TV = LR50 Rev. Q

Leadtek

- WinView 601 (Bt848)
- WinView 610 (Zoran)
- WinFast2000
- WinFast2000 XP

Support for the Leadtek WinView 601 TV/FM

Author of this section: Jon Tombs <jon@gte.esi.us.es>

This card is basically the same as all the rest (Bt484A, Philips tuner), the main difference is that they have attached a programmable attenuator to 3 GPIO lines in order to give some volume control. They have also stuck an infra-red remote control decoded on the board, I will add support for this when I get time (it simple generates an interrupt for each key press, with the key code is placed in the GPIO port).

I don't yet have any application to test the radio support. The tuner frequency setting should work but it is possible that the audio multiplexer is wrong. If it doesn't work, send me email.

- No Thanks to Leadtek they refused to answer any questions about their hardware. The driver was written by visual inspection of the card. If you use this driver, send an email insult to them, and tell them you won't continue buying their hardware unless they support Linux.
- Little thanks to Princeton Technology Corp (http://www.princeton.com.tw) who make the audio attenuator. Their publicly available data-sheet available on their web site doesn't include the chip programming information! Hidden on their server are the full data-sheets, but don't ask how I found it.

To use the driver I use the following options, the tuner and pll settings might be different in your country. You can force it via modprobe parameters. For example:

```
modprobe bttv  tuner=1 pll=28 radio=1 card=17
```

Sets tuner type 1 (Philips PAL_I), PLL with a 28 MHz crystal, enables FM radio and selects bttv card ID 17 (Leadtek WinView 601).

KNC One

- TV-Station
- TV-Station SE (+Software Bundle)
- TV-Station pro (+TV stereo)
- TV-Station FM (+Radio)
- TV-Station RDS (+RDS)
- TV Station SAT (analog satellite)
- TV-Station DVB-S

Note: newer Cards have saa7134, but model name stayed the same?

Provideo

- PV951 or PV-951, now named PV-951T (also are sold as: Boeder TV-FM Video Capture Card, Titanmedia Supervision TV-2400, Provideo PV951 TF, 3DeMon PV951, MediaForte TV-Vision PV951, Yoko PV951, Vivanco Tuner Card PCI Art.-Nr.: 68404)
- Surveillance Series:
- PV-141
- PV-143
- PV-147
- PV-148 (capture only)
- PV-150
- PV-151
- TV-FM Tuner Series:
- PV-951TDV (tv tuner + 1394)
- PV-951T/TF
- PV-951PT/TF
- PV-956T/TF Low Profile
- PV-911

Highscreen

Models:

- TV Karte = LR50 Rev.S
- TV-Boostar = Terratec Terra TV+ Version 1.0 (Bt848, tda9821) ”ceb105.pcb”

Zoltrix

Models:

- Face to Face Capture (Bt848 capture only) (PCB ”VP-2848”)
- Face To Face TV MAX (Bt848) (PCB ”VP-8482 Rev1.3”)
- Genie TV (Bt878) (PCB ”VP-8790 Rev 2.1”)
- Genie Wonder Pro

AVerMedia

- AVer FunTV Lite (ISA, AV3001 chipset) ”M101.C”
- AVerTV
- AVerTV Stereo
- AVerTV Studio (w/FM)
- AVerMedia TV98 with Remote
- AVerMedia TV/FM98 Stereo
- AVerMedia TVCAM98
- TVCapture (Bt848)
- TVPhone (Bt848)
- TVCapture98 (=”AVerMedia TV98” in USA) (Bt878)
- TVPhone98 (Bt878, w/FM)

PCB	PCI-ID	Model-Name	Eeprom	Tuner	Sound	Country
M101.C	ISA !					
M108-B	Bt848		--	FR1236		US[2],[3]
M1A8-A	Bt848	AVer TV-Phone		FM1216	--	
M168-T	1461:0003	AVerTV Studio	48:17	FM1216	TDA9840T	D[1] w/FM w/Remote
M168-U	1461:0004	TVCapture98	40:11	FI1216	--	D w/Remote
M168II-B	1461:0003	Medion MD9592	48:16	FM1216	TDA9873H	D w/FM

[2] Sony NE41S soldered (stereo sound?)
[3] Daughterboard M118-A w/ pic 16c54 and 4 MHz quartz
[1] Daughterboard MB68-A with TDA9820T and TDA9840T

- US site has different drivers for (as of 09/2002):
  - EZ Capture/InterCam PCI (BT-848 chip)
  - EZ Capture/InterCam PCI (BT-878 chip)
  - TV-Phone (BT-848 chip)
  - TV98 (BT-848 chip)
  - TV98 With Remote (BT-848 chip)
  - TV98 (BT-878 chip)
  - TV98 With Remote (BT-878)
  - TV/FM98 (BT-878 chip)
  - AVerTV
  - AverTV Stereo
  - AVerTV Studio

DE hat diverse Treiber fuer diese Modelle (Stand 09/2002):

- TVPhone (848) mit Philips tuner FR12X6 (w/ FM radio)
- TVPhone (848) mit Philips tuner FM12X6 (w/ FM radio)
- TVCapture (848) w/Philips tuner FI12X6
- TVCapture (848) non-Philips tuner
- TVCapture98 (Bt878)
- TVPhone98 (Bt878)
- AVerTV und TVCapture98 w/VCR (Bt 878)
- AVerTVStudio und TVPhone98 w/VCR (Bt878)
- AVerTV GO Series (Kein SVideo Input)
- AVerTV98 (BT-878 chip)
- AVerTV98 mit Fernbedienung (BT-878 chip)
- AVerTV/FM98 (BT-878 chip)
- VDOmate (www.averm.com.cn) = M168U ?

Aimslab

Models:

- Video Highway or ”Video Highway TR200” (ISA)
- Video Highway Xtreme (aka ”VHX”) (Bt848, FM w/ TEA5757)

IXMicro (former: IMS=Integrated Micro Solutions)

Models:

- IXTV BT848 (=TurboTV)
- IXTV BT878
- IMS TurboTV (Bt848)

Lifetec/Medion/Tevion/Aldi

Models:

- LT9306/MD9306 = CPH061
- LT9415/MD9415 = LR90 Rev.F or Rev.G
- MD9592 = Avermedia TVphone98 (PCI_ID=1461:0003), PCB-Rev=M168II-B (w/TDA9873H)
- MD9717 = KNC One (Rev D4, saa7134, FM1216 MK2 tuner)
- MD5044 = KNC One (Rev D4, saa7134, FM1216ME MK3 tuner)

Modular Technologies (www.modulartech.com) UK

Models:

- MM100 PCTV (Bt848)
- MM201 PCTV (Bt878, Bt832) w/ Quartzsight camera
- MM202 PCTV (Bt878, Bt832, tda9874)
- MM205 PCTV (Bt878)
- MM210 PCTV (Bt878) (Galaxy TV, Galaxymedia ?)

Terratec

Models:

- Terra TV+ Version 1.0 (Bt848), ”ceb105.PCB” printed on the PCB, TDA9821
- Terra TV+ Version 1.1 (Bt878), ”LR74 Rev.E” printed on the PCB, TDA9821
- Terra TValueRadio, ”LR102 Rev.C” printed on the PCB
- Terra TV/Radio+ Version 1.0, ”80-CP2830100-0” TTTV3 printed on the PCB, ”CPH010-E83” on the back, SAA6588T, TDA9873H
- Terra TValue Version BT878, ”80-CP2830110-0 TTTV4” printed on the PCB, ”CPH011-D83” on back
- Terra TValue Version 1.0 ”ceb105.PCB” (really identical to Terra TV+ Version 1.0)
- Terra TValue New Revision ”LR102 Rec.C”

- Terra Active Radio Upgrade (tea5757h, saa6588t)
- LR74 is a newer PCB revision of ceb105 (both incl. connector for Active Radio Upgrade)
- Cinergy 400 (saa7134), ”E877 11(S)”, ”PM820092D” printed on PCB
- Cinergy 600 (saa7134)

Technisat

Models:

- Discos ADR PC-Karte ISA (no TV!)
- Discos ADR PC-Karte PCI (probably no TV?)
- Techni-PC-Sat (Sat. analog) Rev 1.2 (zr36120, vpx3220, stv0030, saa5246, BSJE3-494A)
- Mediafocus I (zr36120/zr36125, drp3510, Sat. analog + ADR Radio)
- Mediafocus II (saa7146, Sat. analog)
- SatADR Rev 2.1 (saa7146a, saa7113h, stv0056a, msp3400c, drp3510a, BSKE3-307A)
- SkyStar 1 DVB (AV7110) = Technotrend Premium
- SkyStar 2 DVB (B2C2) (=Sky2PC)

Siemens

Multimedia eXtension Board (MXB) (SAA7146, SAA7111)

Powercolor

Models:

- **MTV878**
  Package comes with different contents:
  a) pcb ”MTV878” (CARD=75)
  b) Pixelview Rev. 4_
- MTV878R w/Remote Control
- MTV878F w/Remote Control w/FM radio

Pinnacle

PCTV models:

- Mirovideo PCTV (Bt848)
- Mirovideo PCTV SE (Bt848)
- Mirovideo PCTV Pro (Bt848 + Daughterboard for TV Stereo and FM)
- Studio PCTV Rave (Bt848 Version = Mirovideo PCTV)
- Studio PCTV Rave (Bt878 package w/o infrared)
- Studio PCTV (Bt878)
- Studio PCTV Pro (Bt878 stereo w/ FM)
- Pinnacle PCTV (Bt878, MT2032)
- Pinnacle PCTV Pro (Bt878, MT2032)
- Pinncale PCTV Sat (bt878a, HM1821/1221) [”Conexant CX24110 with CX24108 tuner, aka HM1221/HM1811”]
- Pinnacle PCTV Sat XE

M(J)PEG capture and playback models:

- DC1+ (ISA)
- DC10 (zr36057, zr36060, saa7110, adv7176)
- DC10+ (zr36067, zr36060, saa7110, adv7176)
- DC20 (ql16x24b,zr36050, zr36016, saa7110, saa7187 ...)
- DC30 (zr36057, zr36050, zr36016, vpx3220, adv7176, ad1843, tea6415, miro FST97A1)
- DC30+ (zr36067, zr36050, zr36016, vpx3220, adv7176)
- DC50 (zr36067, zr36050, zr36016, saa7112, adv7176 (2 pcs.?), ad1843, miro FST97A1, Lattice ???)

Lenco

Models:

- MXR-9565 (=Technisat Mediafocus?)
- MXR-9571 (Bt848) (=CPH031?)
- MXR-9575
- MXR-9577 (Bt878) (=Prolink 878TV Rev.3x)
- MXTV-9578CP (Bt878) (= Prolink PV-BT878P+4E)

Iomega

Buz (zr36067, zr36060, saa7111, saa7185)

LML

LML33 (zr36067, zr36060, bt819, bt856)

Grandtec

Models:

- Grand Video Capture (Bt848)
- Multi Capture Card (Bt878)

Koutech

Models:

- KW-606 (Bt848)
- KW-607 (Bt848 capture only)
- KW-606RSF
- KW-607A (capture only)
- KW-608 (Zoran capture only)

IODATA (jp)

Models:

- GV-BCTV/PCI
- GV-BCTV2/PCI
- GV-BCTV3/PCI
- GV-BCTV4/PCI
- GV-VCP/PCI (capture only)
- GV-VCP2/PCI (capture only)

Canopus (jp)

WinDVR = Kworld "KW-TVL878RF"

www.sigmacom.co.kr

Sigma Cyber TV II

www.sasem.co.kr

Litte OnAir TV

hama

TV/Radio-Tuner Card, PCI (Model 44677) = CPH051

Sigma Designs

Hollywood plus (em8300, em9010, adv7175), (PCB "M340-10") MPEG DVD decoder

Formac

Models:

- iProTV (Card for iMac Mezzanine slot, Bt848+SCSI)
- ProTV (Bt848)
- ProTV II = ProTV Stereo (Bt878) ["stereo" means FM stereo, tv is still mono]

ATI

Models:

- TV-Wonder
- TV-Wonder VE

Diamond Multimedia

DTV2000 (Bt848, tda9875)

Aopen

- VA1000 Plus (w/ Stereo)
- VA1000 Lite
- VA1000 (=LR90)

Intel

Models:

- Smart Video Recorder (ISA full-length)
- Smart Video Recorder pro (ISA half-length)
- Smart Video Recorder III (Bt848)

STB

Models:

- STB Gateway 6000704 (bt878)
- STB Gateway 6000699 (bt848)
- STB Gateway 6000402 (bt848)
- STB TV130 PCI

Videologic

Models:

- Captivator Pro/TV (ISA?)
- Captivator PCI/VC (Bt848 bundled with camera) (capture only)

Technotrend

Models:

- TT-SAT PCI (PCB "Sat-PCI Rev.:1.3.1"; zr36125, vpx3225d, stc0056a, Tuner:BSKE6-155A
- **TT-DVB-Sat**
    - revisions 1.1, 1.3, 1.5, 1.6 and 2.1
    - This card is sold as OEM from:
        - Siemens DVB-s Card
        - Hauppauge WinTV DVB-S
        - Technisat SkyStar 1 DVB
        - Galaxis DVB Sat

- Now this card is called TT-PCline Premium Family
- TT-Budget (saa7146, bsru6-701a) This card is sold as OEM from:
  - Hauppauge WinTV Nova
  - Satelco Standard PCI (DVB-S)
- TT-DVB-C PCI

Teles

DVB-s (Rev. 2.2, BSRV2-301A, data only?)

Remote Vision

MX RV605 (Bt848 capture only)

Boeder

Models:

- PC ChatCam (Model 68252) (Bt848 capture only)
- Tv/Fm Capture Card (Model 68404) = PV951

Media-Surfer (esc-kathrein.de)

Models:

- Sat-Surfer (ISA)
- Sat-Surfer PCI = Techni-PC-Sat
- Cable-Surfer 1
- Cable-Surfer 2
- Cable-Surfer PCI (zr36120)
- Audio-Surfer (ISA Radio card)

Jetway (www.jetway.com.tw)

Models:

- JW-TV 878M
- JW-TV 878 = KWorld KW-TV878RF

Galaxis

Models:

- Galaxis DVB Card S CI
- Galaxis DVB Card C CI
- Galaxis DVB Card S
- Galaxis DVB Card C
- Galaxis plug.in S [neuer Name: Galaxis DVB Card S CI

Hauppauge

Models:

- many many WinTV models ...
- WinTV DVBs = Technotrend Premium 1.3
- WinTV NOVA = Technotrend Budget 1.1 "S-DVB DATA"
- WinTV NOVA-CI "SDVBACI"
- WinTV Nova USB (=Technotrend USB 1.0)
- WinTV-Nexus-s (=Technotrend Premium 2.1 or 2.2)
- WinTV PVR
- WinTV PVR 250
- WinTV PVR 450

US models

-990 WinTV-PVR-350 (249USD) (iTVC15 chipset + radio) -980 WinTV-PVR-250 (149USD) (iTVC15 chipset) -880 WinTV-PVR-PCI (199USD) (KFIR chipset + bt878) -881 WinTV-PVR-USB -190 WinTV-GO -191 WinTV-GO-FM -404 WinTV -401 WinTV-radio -495 WinTV-Theater -602 WinTV-USB -621 WinTV-USB-FM -600 USB-Live -698 WinTV-HD -697 WinTV-D -564 WinTV-Nexus-S

Deutsche Modelle:

-603 WinTV GO -719 WinTV Primio-FM -718 WinTV PCI-FM -497 WinTV Theater -569 WinTV USB -568 WinTV USB-FM -882 WinTV PVR -981 WinTV PVR 250 -891 WinTV-PVR-USB -541 WinTV Nova -488 WinTV Nova-Ci -564 WinTV-Nexus-s -727 WinTV-DVB-c -545 Common Interface -898 WinTV-Nova-USB

UK models:

-607 WinTV Go -693,793 WinTV Primio FM -647,747 WinTV PCI FM -498 WinTV Theater -883 WinTV PVR -893 WinTV PVR USB (Duplicate entry) -566 WinTV USB (UK) -573 WinTV USB FM -429 Impact VCB (bt848) -600 USB Live (Video-In 1x Comp, 1xSVHS) -542 WinTV Nova -717 WinTV DVB-S -909 Nova-t PCI -893 Nova-t USB (Duplicate entry) -802 MyTV -804 MyView -809 MyVideo -872 MyTV2Go FM -546 WinTV Nova-S CI -543 WinTV Nova -907 Nova-S USB -908 Nova-T USB -717 WinTV Nexus-S -157 DEC3000-s Standalone + USB

Spain:

-685 WinTV-Go -690 WinTV-PrimioFM -416 WinTV-PCI Nicam Estereo -677 WinTV-PCI-FM -699 WinTV-Theater -683 WinTV-USB -678 WinTV-USB-FM -983 WinTV-PVR-250 -883 WinTV-PVR-PCI -993 WinTV-PVR-350 -893 WinTV-PVR-USB -728 WinTV-DVB-C PCI -832 MyTV2Go -869 MyTV2Go-FM -805 MyVideo (USB)

Matrix-Vision

Models:

- MATRIX-Vision MV-Delta
- MATRIX-Vision MV-Delta 2
- MVsigma-SLC (Bt848)

Conceptronic (.net)

Models:

- TVCON FM, TV card w/ FM = CPH05x
- TVCON = CPH06x

BestData

Models:

- HCC100 = VCC100rev1 + camera
- VCC100 rev1 (bt848)
- VCC100 rev2 (bt878)

Gallant (www.gallantcom.com) www.minton.com.tw

Models:

- Intervision IV-510 (capture only bt8x8)
- Intervision IV-550 (bt8x8)
- Intervision IV-100 (zoran)
- Intervision IV-1000 (bt8x8)

Asonic (www.asonic.com.cn) (website down)

SkyEye tv 878

Hoontech

878TV/FM

Teppro (www.itcteppro.com.tw)

Models:

- ITC PCITV (Card Ver 1.0) ”Teppro TV1/TVFM1 Card”
- ITC PCITV (Card Ver 2.0)
- ITC PCITV (Card Ver 3.0) = ”PV-BT878P+ (REV.9D)”
- ITC PCITV (Card Ver 4.0)
- TEPPRO IV-550 (For BT848 Main Chip)
- ITC DSTTV (bt878, satellite)
- ITC VideoMaker (saa7146, StreamMachine sm2110, tvtuner) ”PV-SM2210P+ (REV:1C)”

Kworld (www.kworld.com.tw)

PC TV Station:

- KWORLD KW-TV878R TV (no radio)
- KWORLD KW-TV878RF TV (w/ radio)
- KWORLD KW-TVL878RF (low profile)
- KWORLD KW-TV713XRF (saa7134)

  MPEG TV Station (same cards as above plus WinDVR Software MPEG en/decoder)
- KWORLD KW-TV878R -Pro TV (no Radio)
- KWORLD KW-TV878RF-Pro TV (w/ Radio)
- KWORLD KW-TV878R -Ultra TV (no Radio)
- KWORLD KW-TV878RF-Ultra TV (w/ Radio)

JTT/ Justy Corp.(http://www.jtt.ne.jp/)

JTT-02 (JTT TV) ”TV watchmate pro” (bt848)

ADS www.adstech.com

Models:

- Channel Surfer TV (CHX-950)
- Channel Surfer TV+FM (CHX-960FM)

AVEC www.prochips.com

AVEC Intercapture (bt848, tea6320)

NoBrand

TV Excel = Australian Name for ”PV-BT878P+ 8E” or ”878TV Rev.3_”

Mach www.machspeed.com

Mach TV 878

Eline www.eline-net.com/

Models:

- Eline Vision TVMaster / TVMaster FM (ELV-TVM/ ELV-TVM-FM) = LR26 (bt878)
- Eline Vision TVMaster-2000 (ELV-TVM-2000, ELV-TVM-2000-FM)= LR138 (saa713x)

Spirit

- Spirit TV Tuner/Video Capture Card (bt848)

Boser www.boser.com.tw

Models:

- HS-878 Mini PCI Capture Add-on Card
- HS-879 Mini PCI 3D Audio and Capture Add-on Card (w/ ES1938 Solo-1)

Satelco www.citycom-gmbh.de, www.satelco.de

Models:

- TV-FM =KNC1 saa7134
- Standard PCI (DVB-S) = Technotrend Budget
- Standard PCI (DVB-S) w/ CI
- Satelco Highend PCI (DVB-S) = Technotrend Premium

Sensoray www.sensoray.com

Models:

- Sensoray 311 (PC/104 bus)
- Sensoray 611 (PCI)

CEI (Chartered Electronics Industries Pte Ltd [CEI] [FCC ID HBY])

Models:

- TV Tuner - HBY-33A-RAFFLES Brooktree Bt848KPF + Philips
- TV Tuner MG9910 - HBY33A-TVO CEI + Philips SAA7110 + OKI M548262 + ST STV8438CV
- Primetime TV (ISA)
  - acquired by Singapore Technologies
  - now operating as Chartered Semiconductor Manufacturing
  - Manufacturer of video cards is listed as:
    - Cogent Electronics Industries [CEI]

AITech

Models:

- Wavewatcher TV (ISA)
- AITech WaveWatcher TV-PCI = can be LR26 (Bt848) or LR50 (BT878)
- WaveWatcher TVR-202 TV/FM Radio Card (ISA)

MAXRON

Maxron MaxTV/FM Radio (KW-TV878-FNT) = Kworld or JW-TV878-FBK

www.ids-imaging.de

Models:

- Falcon Series (capture only)

In USA: http://www.theimagingsource.com/ - DFG/LC1

www.sknet-web.co.jp

SKnet Monster TV (saa7134)

A-Max www.amaxhk.com (Colormax, Amax, Napa)

APAC Viewcomp 878

Cybertainment

Models:

- CyberMail AV Video Email Kit w/ PCI Capture Card (capture only)
- CyberMail Xtreme

These are Flyvideo

VCR (http://www.vcrinc.com/)

Video Catcher 16

Twinhan

Models:

- DST Card/DST-IP (bt878, twinhan asic) VP-1020 - Sold as:
  - KWorld DVBS Satellite TV-Card
  - Powercolor DSTV Satellite Tuner Card
  - Prolink Pixelview DTV2000
  - Provideo PV-911 Digital Satellite TV Tuner Card With Common Interface ?
- DST-CI Card (DVB Satellite) VP-1030
- DCT Card (DVB cable)

MSI

Models:

- MSI TV@nywhere Tuner Card (MS-8876) (CX23881/883) Not Bt878 compatible.
- MS-8401 DVB-S

Focus www.focusinfo.com

InVideo PCI (bt878)

Sdisilk www.sdisilk.com/

Models:

- SDI Silk 100
- SDI Silk 200 SDI Input Card

www.euresys.com

PICOLO series

PMC/Pace

www.pacecom.co.uk website closed

Mercury www.kobian.com (UK and FR)

Models:

- LR50
- LR138RBG-Rx == LR138

TEC sound

TV-Mate = Zoltrix VP-8482

Though educated googling found: www.techmakers.com

(package and manuals don't have any other manufacturer info) TecSound

Lorenzen www.lorenzen.de

SL DVB-S PCI = Technotrend Budget PCI (su1278 or bsru version)

Origo (.uk) www.origo2000.com

PC TV Card = LR50

I/O Magic www.iomagic.com

PC PVR - Desktop TV Personal Video Recorder DR-PCTV100 = Pinnacle ROB2D-51009464 4.0 + Cyberlink PowerVCR II

Arowana

TV-Karte / Poso Power TV (?) = Zoltrix VP-8482 (?)

iTVC15 boards

kuroutoshikou.com ITVC15 yuan.com MPG160 PCI TV (Internal PCI MPEG2 encoder card plus TV-tuner)

Asus www.asuscom.com

Models:

- Asus TV Tuner Card 880 NTSC (low profile, cx23880)
- Asus TV (saa7134)

Hoontech

http://www.hoontech.de/

- HART Vision 848 (H-ART Vision 848)
- HART Vision 878 (H-Art Vision 878)

Chips used at bttv devices

- all boards:
  - Brooktree Bt848/848A/849/878/879: video capture chip
- Board specific
  - Miro PCTV:
    - Philips or Temic Tuner
  - Hauppauge Win/TV pci (version 405):
    - Microchip 24LC02B or Philips 8582E2Y:
      - 256 Byte EEPROM with configuration information
      - I2C 0xa0-0xa1, (24LC02B also responds to 0xa2-0xaf)
    - Philips SAA5246AGP/E: Videotext decoder chip, I2C 0x22-0x23
    - TDA9800: sound decoder
    - Winbond W24257AS-35: 32Kx8 CMOS static RAM (Videotext buffer mem)
    - 14052B: analog switch for selection of sound source
- PAL:
  - TDA5737: VHF, hyperband and UHF mixer/oscillator for TV and VCR 3-band tuners
  - TSA5522: 1.4 GHz I2C-bus controlled synthesizer, I2C 0xc2-0xc3
- NTSC:
  - TDA5731: VHF, hyperband and UHF mixer/oscillator for TV and VCR 3-band tuners
  - TSA5518: no datasheet available on Philips site
- STB TV pci:
  - ???
  - if you want better support for STB cards send me info! Look at the board! What chips are on it?

Specs

Philips http://www.Semiconductors.COM/pip/

Conexant http://www.conexant.com/

Micronas http://www.micronas.com/en/home/index.html

Thanks

Many thanks to:

- Markus Schroeder <schroedm@uni-duesseldorf.de> for information on the Bt848 and tuner programming and his control program xtvc.
- Martin Buck <martin-2.buck@student.uni-ulm.de> for his great Videotext package.
- Gerd Hoffmann for the MSP3400 support and the modular I2C, tuner, ... support.
- MATRIX Vision for giving us 2 cards for free, which made support of single crystal operation possible.
- MIRO for providing a free PCTV card and detailed information about the components on their cards. (E.g. how the tuner type is detected) Without their card I could not have debugged the NTSC mode.
- Hauppauge for telling how the sound input is selected and what components they do and will use on their radio cards. Also many thanks for faxing me the FM1216 data sheet.

Contributors

Michael Chu <mmchu@pobox.com>
AverMedia fix and more flexible card recognition

Alan Cox <alan@lxorguk.ukuu.org.uk>
Video4Linux interface and 2.1.x kernel adaptation

Chris Kleitsch
Hardware I2C

Gerd Hoffmann
Radio card (ITT sound processor)

bigfoot <bigfoot@net-way.net>

Ragnar Hojland Espinosa <ragnar@macula.net>
ConferenceTV card

- **many more (please mail me if you are missing in this list and would**
  like to be mentioned)

59.7.2 The cafe_ccic driver

Author: Jonathan Corbet <corbet@lwn.net>

Introduction

”cafe_ccic” is a driver for the Marvell 88ALP01 ”cafe” CMOS camera controller. This is the controller found in first-generation OLPC systems, and this driver was written with support from the OLPC project.

Current status: the core driver works. It can generate data in YUV422, RGB565, and RGB444 formats. (Anybody looking at the code will see RGB32 as well, but that is a debugging aid which will be removed shortly). VGA and QVGA modes work; CIF is there but the colors remain funky. Only the OV7670 sensor is known to work with this controller at this time.

To try it out: either of these commands will work:

```
$ mplayer tv:// -tv driver=v4l2:width=640:height=480 -nosound
$ mplayer tv:// -tv driver=v4l2:width=640:height=480:outfmt=bgr16 -nosound
```

The ”xawtv” utility also works; gqcam does not, for unknown reasons.

Load time options

There are a few load-time options, most of which can be changed after loading via sysfs as well:

- alloc_bufs_at_load: Normally, the driver will not allocate any DMA buffers until the time comes to transfer data. If this option is set, then worst-case-sized buffers will be allocated at module load time. This option nails down the memory for the life of the module, but perhaps decreases the chances of an allocation failure later on.
- dma_buf_size: The size of DMA buffers to allocate. Note that this option is only consulted for load-time allocation; when buffers are allocated at run time, they will be sized appropriately for the current camera settings.
- n_dma_bufs: The controller can cycle through either two or three DMA buffers. Normally, the driver tries to use three buffers; on faster systems, however, it will work well with only two.
- min_buffers: The minimum number of streaming I/O buffers that the driver will consent to work with. Default is one, but, on slower systems, better behavior with mplayer can be achieved by setting to a higher value (like six).
- max_buffers: The maximum number of streaming I/O buffers; default is ten. That number was carefully picked out of a hat and should not be assumed to actually mean much of anything.
- flip: If this boolean parameter is set, the sensor will be instructed to invert the video image. Whether it makes sense is determined by how your particular camera is mounted.

59.7.3 The cx88 driver

Author: Gerd Hoffmann

This is a v4l2 device driver for the cx2388x chip.

Current status

video

- Works.
- Overlay isn't supported.

audio

- Works. The TV standard detection is made by the driver, as the hardware has bugs to auto-detect.
- audio data dma (i.e. recording without loopback cable to the sound card) is supported via cx88-alsa.

vbi

- Works.

How to add support for new cards

The driver needs some config info for the TV cards. This stuff is in cx88-cards.c. If the driver doesn't work well you likely need a new entry for your card in that file. Check the kernel log (using dmesg) to see whenever the driver knows your card or not. There is a line like this one:

```
cx8800[0]: subsystem: 0070:3400, board: Hauppauge WinTV \
        34xxx models [card=1,autodetected]
```

If your card is listed as ”board: UNKNOWN/GENERIC” it is unknown to the driver. What to do then?

1) Try upgrading to the latest snapshot, maybe it has been added meanwhile.
2) You can try to create a new entry yourself, have a look at cx88-cards.c. If that worked, mail me your changes as unified diff (”diff -u”).
3) Or you can mail me the config information. We need at least the following information to add the card:
   - the PCI Subsystem ID (”0070:3400” from the line above, ”lspci -v” output is fine too).
   - the tuner type used by the card. You can try to find one by trial-and-error using the tuner=<n> insmod option. If you know which one the card has you can also have a look at the list in CARDLIST.tuner

59.7.4 The Samsung S5P/Exynos4 FIMC driver

Copyright © 2012 - 2013 Samsung Electronics Co., Ltd.

The FIMC (Fully Interactive Mobile Camera) device available in Samsung SoC Application Processors is an integrated camera host interface, color space converter, image resizer and rotator. It's also capable of capturing data from LCD controller (FIMD) through the SoC internal writeback data path. There are multiple FIMC instances in the SoCs (up to 4), having slightly different capabilities, like pixel alignment constraints, rotator availability, LCD writeback support, etc. The driver is located at drivers/media/platform/samsung/exynos4-is directory.

Supported SoCs

S5PC100 (mem-to-mem only), S5PV210, Exynos4210

Supported features

- camera parallel interface capture (ITU-R.BT601/565);
- camera serial interface capture (MIPI-CSI2);
- memory-to-memory processing (color space conversion, scaling, mirror and rotation);
- dynamic pipeline re-configuration at runtime (re-attachment of any FIMC instance to any parallel video input or any MIPI-CSI front-end);
- runtime PM and system wide suspend/resume

Not currently supported

- LCD writeback input
- per frame clock gating (mem-to-mem)

User space interfaces

Media device interface

The driver supports Media Controller API as defined at media_controller. The media device driver name is ”Samsung S5P FIMC”.

The purpose of this interface is to allow changing assignment of FIMC instances to the SoC peripheral camera input at runtime and optionally to control internal connections of the MIPI-CSIS device(s) to the FIMC entities.

The media device interface allows to configure the SoC for capturing image data from the sensor through more than one FIMC instance (e.g. for simultaneous viewfinder and still capture setup).

Reconfiguration is done by enabling/disabling media links created by the driver during initialization. The internal device topology can be easily discovered through media entity and links enumeration.

Memory-to-memory video node

V4L2 memory-to-memory interface at /dev/video? device node. This is standalone video device, it has no media pads. However please note the mem-to-mem and capture video node operation on same FIMC instance is not allowed. The driver detects such cases but the applications should prevent them to avoid an undefined behaviour.

Capture video node

The driver supports V4L2 Video Capture Interface as defined at devices.

At the capture and mem-to-mem video nodes only the multi-planar API is supported. For more details see: planar-apis.

Camera capture subdevs

Each FIMC instance exports a sub-device node (/dev/v4l-subdev?), a sub-device node is also created per each available and enabled at the platform level MIPI-CSI receiver device (currently up to two).

sysfs

In order to enable more precise camera pipeline control through the sub-device API the driver creates a sysfs entry associated with "s5p-fimc-md" platform device. The entry path is: /sys/platform/devices/s5p-fimc-md/subdev_conf_mode.

In typical use case there could be a following capture pipeline configuration: sensor subdev -> mipi-csi subdev -> fimc subdev -> video node

When we configure these devices through sub-device API at user space, the configuration flow must be from left to right, and the video node is configured as last one.

When we don't use sub-device user space API the whole configuration of all devices belonging to the pipeline is done at the video node driver. The sysfs entry allows to instruct the capture node driver not to configure the sub-devices (format, crop), to avoid resetting the subdevs' configuration when the last configuration steps at the video node is performed.

For full sub-device control support (subdevs configured at user space before starting streaming):

```
# echo "sub-dev" > /sys/platform/devices/s5p-fimc-md/subdev_conf_mode
```

For V4L2 video node control only (subdevs configured internally by the host driver):

```
# echo "vid-dev" > /sys/platform/devices/s5p-fimc-md/subdev_conf_mode
```

This is a default option.

5. Device mapping to video and subdev device nodes

There are associated two video device nodes with each device instance in hardware - video capture and mem-to-mem and additionally a subdev node for more precise FIMC capture subsystem control. In addition a separate v4l2 sub-device node is created per each MIPI-CSIS device.

How to find out which /dev/video? or /dev/v4l-subdev? is assigned to which device?

You can either grep through the kernel log to find relevant information, i.e.

```
# dmesg | grep -i fimc
```

(note that udev, if present, might still have rearranged the video nodes),

or retrieve the information from /dev/media? with help of the media-ctl tool:

```
# media-ctl -p
```

7. Build

If the driver is built as a loadable kernel module (CONFIG_VIDEO_SAMSUNG_S5P_FIMC=m) two modules are created (in addition to the core v4l2 modules): s5p-fimc.ko and optional s5p-csis.ko (MIPI-CSI receiver subdev).

59.7.5 i.MX Video Capture Driver

Introduction

The Freescale i.MX5/6 contains an Image Processing Unit (IPU), which handles the flow of image frames to and from capture devices and display devices.

For image capture, the IPU contains the following internal subunits:

- Image DMA Controller (IDMAC)
- Camera Serial Interface (CSI)
- Image Converter (IC)
- Sensor Multi-FIFO Controller (SMFC)
- Image Rotator (IRT)
- Video De-Interlacing or Combining Block (VDIC)

The IDMAC is the DMA controller for transfer of image frames to and from memory. Various dedicated DMA channels exist for both video capture and display paths. During transfer, the IDMAC is also capable of vertical image flip, 8x8 block transfer (see IRT description), pixel component re-ordering (for example UYVY to YUYV) within the same colorspace, and packed <--> planar conversion. The IDMAC can also perform a simple de-interlacing by interweaving even and odd lines during transfer (without motion compensation which requires the VDIC).

The CSI is the backend capture unit that interfaces directly with camera sensors over Parallel, BT.656/1120, and MIPI CSI-2 buses.

The IC handles color-space conversion, resizing (downscaling and upscaling), horizontal flip, and 90/270 degree rotation operations.

There are three independent ”tasks” within the IC that can carry out conversions concurrently: pre-process encoding, pre-process viewfinder, and post-processing. Within each task, conversions are split into three sections: downsizing section, main section (upsizing, flip, colorspace conversion, and graphics plane combining), and rotation section.

The IPU time-shares the IC task operations. The time-slice granularity is one burst of eight pixels in the downsizing section, one image line in the main processing section, one image frame in the rotation section.

The SMFC is composed of four independent FIFOs that each can transfer captured frames from sensors directly to memory concurrently via four IDMAC channels.

The IRT carries out 90 and 270 degree image rotation operations. The rotation operation is carried out on 8x8 pixel blocks at a time. This operation is supported by the IDMAC which handles the 8x8 block transfer along with block reordering, in coordination with vertical flip.

The VDIC handles the conversion of interlaced video to progressive, with support for different motion compensation modes (low, medium, and high motion). The deinterlaced output frames from the VDIC can be sent to the IC pre-process viewfinder task for further conversions. The VDIC also contains a Combiner that combines two image planes, with alpha blending and color keying.

In addition to the IPU internal subunits, there are also two units outside the IPU that are also involved in video capture on i.MX:

- MIPI CSI-2 Receiver for camera sensors with the MIPI CSI-2 bus interface. This is a Synopsys DesignWare core.
- Two video multiplexers for selecting among multiple sensor inputs to send to a CSI.

For more info, refer to the latest versions of the i.MX5/6 reference manuals[1] and[2].

Features

Some of the features of this driver include:

- Many different pipelines can be configured via media controller API, that correspond to the hardware video capture pipelines supported in the i.MX.
- Supports parallel, BT.565, and MIPI CSI-2 interfaces.
- Concurrent independent streams, by configuring pipelines to multiple video capture interfaces using independent entities.
- Scaling, color-space conversion, horizontal and vertical flip, and image rotation via IC task subdevs.
- Many pixel formats supported (RGB, packed and planar YUV, partial planar YUV).
- The VDIC subdev supports motion compensated de-interlacing, with three motion compensation modes: low, medium, and high motion. Pipelines are defined that allow sending frames to the VDIC subdev directly from the CSI. There is also support in the future for sending frames to the VDIC from memory buffers via a output/mem2mem devices.

[1] http://www.nxp.com/assets/documents/data/en/reference-manuals/IMX6DQRM.pdf
[2] http://www.nxp.com/assets/documents/data/en/reference-manuals/IMX6SDLRM.pdf

- Includes a Frame Interval Monitor (FIM) that can correct vertical sync problems with the ADV718x video decoders.

Topology

The following shows the media topologies for the i.MX6Q SabreSD and i.MX6Q SabreAuto. Refer to these diagrams in the entity descriptions in the next section.

The i.MX5/6 topologies can differ upstream from the IPUv3 CSI video multiplexers, but the internal IPUv3 topology downstream from there is common to all i.MX5/6 platforms. For example, the SabreSD, with the MIPI CSI-2 OV5640 sensor, requires the i.MX6 MIPI CSI-2 receiver. But the SabreAuto has only the ADV7180 decoder on a parallel bt.656 bus, and therefore does not require the MIPI CSI-2 receiver, so it is missing in its graph.

Fig. 1: Media pipeline graph on i.MX6Q SabreSD

Fig. 2: Media pipeline graph on i.MX6Q SabreAuto

Entities

imx6-mipi-csi2

This is the MIPI CSI-2 receiver entity. It has one sink pad to receive the MIPI CSI-2 stream (usually from a MIPI CSI-2 camera sensor). It has four source pads, corresponding to the four MIPI CSI-2 demuxed virtual channel outputs. Multiple source pads can be enabled to independently stream from multiple virtual channels.

This entity actually consists of two sub-blocks. One is the MIPI CSI-2 core. This is a Synopsys Designware MIPI CSI-2 core. The other sub-block is a "CSI-2 to IPU gasket". The gasket acts as a demultiplexer of the four virtual channels streams, providing four separate parallel buses containing each virtual channel that are routed to CSIs or video multiplexers as described below.

On i.MX6 solo/dual-lite, all four virtual channel buses are routed to two video multiplexers. Both CSI0 and CSI1 can receive any virtual channel, as selected by the video multiplexers.

On i.MX6 Quad, virtual channel 0 is routed to IPU1-CSI0 (after selected by a video mux), virtual channels 1 and 2 are hard-wired to IPU1-CSI1 and IPU2-CSI0, respectively, and virtual channel 3 is routed to IPU2-CSI1 (again selected by a video mux).

ipuX_csiY_mux

These are the video multiplexers. They have two or more sink pads to select from either camera sensors with a parallel interface, or from MIPI CSI-2 virtual channels from imx6-mipi-csi2 entity. They have a single source pad that routes to a CSI (ipuX_csiY entities).

On i.MX6 solo/dual-lite, there are two video mux entities. One sits in front of IPU1-CSI0 to select between a parallel sensor and any of the four MIPI CSI-2 virtual channels (a total of five sink pads). The other mux sits in front of IPU1-CSI1, and again has five sink pads to select between a parallel sensor and any of the four MIPI CSI-2 virtual channels.

On i.MX6 Quad, there are two video mux entities. One sits in front of IPU1-CSI0 to select between a parallel sensor and MIPI CSI-2 virtual channel 0 (two sink pads). The other mux sits in front of IPU2-CSI1 to select between a parallel sensor and MIPI CSI-2 virtual channel 3 (two sink pads).

ipuX_csiY

These are the CSI entities. They have a single sink pad receiving from either a video mux or from a MIPI CSI-2 virtual channel as described above.

This entity has two source pads. The first source pad can link directly to the ipuX_vdic entity or the ipuX_ic_prp entity, using hardware links that require no IDMAC memory buffer transfer.

When the direct source pad is routed to the ipuX_ic_prp entity, frames from the CSI can be processed by one or both of the IC pre-processing tasks.

When the direct source pad is routed to the ipuX_vdic entity, the VDIC will carry out motion-compensated de-interlace using "high motion" mode (see description of ipuX_vdic entity).

The second source pad sends video frames directly to memory buffers via the SMFC and an IDMAC channel, bypassing IC pre-processing. This source pad is routed to a capture device node, with a node name of the format "ipuX_csiY capture".

Note that since the IDMAC source pad makes use of an IDMAC channel, pixel reordering within the same colorspace can be carried out by the IDMAC channel. For example, if the CSI sink pad is receiving in UYVY order, the capture device linked to the IDMAC source pad can capture in YUYV order. Also, if the CSI sink pad is receiving a packed YUV format, the capture device can capture a planar YUV format such as YUV420.

The IDMAC channel at the IDMAC source pad also supports simple interweave without motion compensation, which is activated if the source pad's field type is sequential top-bottom or bottom-top, and the requested capture interface field type is set to interlaced (t-b, b-t, or unqualified interlaced). The capture interface will enforce the same field order as the source pad field order (interlaced-bt if source pad is seq-bt, interlaced-tb if source pad is seq-tb).

For events produced by ipuX_csiY, see ref:*imx_api_ipuX_csiY*.

Cropping in ipuX_csiY

The CSI supports cropping the incoming raw sensor frames. This is implemented in the ipuX_csiY entities at the sink pad, using the crop selection subdev API.

The CSI also supports fixed divide-by-two downscaling independently in width and height. This is implemented in the ipuX_csiY entities at the sink pad, using the compose selection subdev API.

The output rectangle at the ipuX_csiY source pad is the same as the compose rectangle at the sink pad. So the source pad rectangle cannot be negotiated, it must be set using the compose selection API at sink pad (if /2 downscale is desired, otherwise source pad rectangle is equal to incoming rectangle).

To give an example of crop and /2 downscale, this will crop a 1280x960 input frame to 640x480, and then /2 downscale in both dimensions to 320x240 (assumes ipu1_csi0 is linked to ipu1_csi0_mux):

```
media-ctl -V "'ipu1_csi0_mux':2[fmt:UYVY2X8/1280x960]"
media-ctl -V "'ipu1_csi0':0[crop:(0,0)/640x480]"
media-ctl -V "'ipu1_csi0':0[compose:(0,0)/320x240]"
```

Frame Skipping in ipuX_csiY

The CSI supports frame rate decimation, via frame skipping. Frame rate decimation is specified by setting the frame intervals at sink and source pads. The ipuX_csiY entity then applies the best frame skip setting to the CSI to achieve the desired frame rate at the source pad.

The following example reduces an assumed incoming 60 Hz frame rate by half at the IDMAC output source pad:

```
media-ctl -V "'ipu1_csi0':0[fmt:UYVY2X8/640x480@1/60]"
media-ctl -V "'ipu1_csi0':2[fmt:UYVY2X8/640x480@1/30]"
```

Frame Interval Monitor in ipuX_csiY

See ref:*imx_api_FIM*.

ipuX_vdic

The VDIC carries out motion compensated de-interlacing, with three motion compensation modes: low, medium, and high motion. The mode is specified with the menu control V4L2_CID_DEINTERLACING_MODE. The VDIC has two sink pads and a single source pad.

The direct sink pad receives from an ipuX_csiY direct pad. With this link the VDIC can only operate in high motion mode.

When the IDMAC sink pad is activated, it receives from an output or mem2mem device node. With this pipeline, the VDIC can also operate in low and medium modes, because these modes require receiving frames from memory buffers. Note that an output or mem2mem device is not implemented yet, so this sink pad currently has no links.

The source pad routes to the IC pre-processing entity ipuX_ic_prp.

ipuX_ic_prp

This is the IC pre-processing entity. It acts as a router, routing data from its sink pad to one or both of its source pads.

This entity has a single sink pad. The sink pad can receive from the ipuX_csiY direct pad, or from ipuX_vdic.

This entity has two source pads. One source pad routes to the pre-process encode task entity (ipuX_ic_prpenc), the other to the pre-process viewfinder task entity (ipuX_ic_prpvf). Both source pads can be activated at the same time if the sink pad is receiving from ipuX_csiY. Only the source pad to the pre-process viewfinder task entity can be activated if the sink pad is receiving from ipuX_vdic (frames from the VDIC can only be processed by the pre-process viewfinder task).

ipuX_ic_prpenc

This is the IC pre-processing encode entity. It has a single sink pad from ipuX_ic_prp, and a single source pad. The source pad is routed to a capture device node, with a node name of the format ”ipuX_ic_prpenc capture”.

This entity performs the IC pre-process encode task operations: color-space conversion, resizing (downscaling and upscaling), horizontal and vertical flip, and 90/270 degree rotation. Flip and rotation are provided via standard V4L2 controls.

Like the ipuX_csiY IDMAC source, this entity also supports simple de-interlace without motion compensation, and pixel reordering.

ipuX_ic_prpvf

This is the IC pre-processing viewfinder entity. It has a single sink pad from ipuX_ic_prp, and a single source pad. The source pad is routed to a capture device node, with a node name of the format "ipuX_ic_prpvf capture".

This entity is identical in operation to ipuX_ic_prpenc, with the same resizing and CSC operations and flip/rotation controls. It will receive and process de-interlaced frames from the ipuX_vdic if ipuX_ic_prp is receiving from ipuX_vdic.

Like the ipuX_csiY IDMAC source, this entity supports simple interweaving without motion compensation. However, note that if the ipuX_vdic is included in the pipeline (ipuX_ic_prp is receiving from ipuX_vdic), it's not possible to use interweave in ipuX_ic_prpvf, since the ipuX_vdic has already carried out de-interlacing (with motion compensation) and therefore the field type output from ipuX_vdic can only be none (progressive).

Capture Pipelines

The following describe the various use-cases supported by the pipelines.

The links shown do not include the backend sensor, video mux, or mipi csi-2 receiver links. This depends on the type of sensor interface (parallel or mipi csi-2). So these pipelines begin with:

sensor -> ipuX_csiY_mux -> ...

for parallel sensors, or:

sensor -> imx6-mipi-csi2 -> (ipuX_csiY_mux) -> ...

for mipi csi-2 sensors. The imx6-mipi-csi2 receiver may need to route to the video mux (ipuX_csiY_mux) before sending to the CSI, depending on the mipi csi-2 virtual channel, hence ipuX_csiY_mux is shown in parenthesis.

Unprocessed Video Capture:

Send frames directly from sensor to camera device interface node, with no conversions, via ipuX_csiY IDMAC source pad:

-> ipuX_csiY:2 -> ipuX_csiY capture

IC Direct Conversions:

This pipeline uses the preprocess encode entity to route frames directly from the CSI to the IC, to carry out scaling up to 1024x1024 resolution, CSC, flipping, and image rotation:

-> ipuX_csiY:1 -> 0:ipuX_ic_prp:1 -> 0:ipuX_ic_prpenc:1 -> ipuX_ic_prpenc capture

Motion Compensated De-interlace:

This pipeline routes frames from the CSI direct pad to the VDIC entity to support motion-compensated de-interlacing (high motion mode only), scaling up to 1024x1024, CSC, flip, and rotation:

-> ipuX_csiY:1 -> 0:ipuX_vdic:2 -> 0:ipuX_ic_prp:2 -> 0:ipuX_ic_prpvf:1 -> ipuX_ic_prpvf capture

Usage Notes

To aid in configuration and for backward compatibility with V4L2 applications that access controls only from video device nodes, the capture device interfaces inherit controls from the active entities in the current pipeline, so controls can be accessed either directly from the subdev or from the active capture device interface. For example, the FIM controls are available either from the ipuX_csiY subdevs or from the active capture device.

The following are specific usage notes for the Sabre* reference boards:

i.MX6Q SabreLite with OV5642 and OV5640

This platform requires the OmniVision OV5642 module with a parallel camera interface, and the OV5640 module with a MIPI CSI-2 interface. Both modules are available from Boundary Devices:

- https://boundarydevices.com/product/nit6x_5mp
- https://boundarydevices.com/product/nit6x_5mp_mipi

Note that if only one camera module is available, the other sensor node can be disabled in the device tree.

The OV5642 module is connected to the parallel bus input on the i.MX internal video mux to IPU1 CSI0. It's i2c bus connects to i2c bus 2.

The MIPI CSI-2 OV5640 module is connected to the i.MX internal MIPI CSI-2 receiver, and the four virtual channel outputs from the receiver are routed as follows: vc0 to the IPU1 CSI0 mux, vc1 directly to IPU1 CSI1, vc2 directly to IPU2 CSI0, and vc3 to the IPU2 CSI1 mux. The OV5640 is also connected to i2c bus 2 on the SabreLite, therefore the OV5642 and OV5640 must not share the same i2c slave address.

The following basic example configures unprocessed video capture pipelines for both sensors. The OV5642 is routed to ipu1_csi0, and the OV5640, transmitting on MIPI CSI-2 virtual channel 1 (which is imx6-mipi-csi2 pad 2), is routed to ipu1_csi1. Both sensors are configured to output 640x480, and the OV5642 outputs YUYV2X8, the OV5640 UYVY2X8:

```
# Setup links for OV5642
media-ctl -l "'ov5642 1-0042':0 -> 'ipu1_csi0_mux':1[1]"
media-ctl -l "'ipu1_csi0_mux':2 -> 'ipu1_csi0':0[1]"
media-ctl -l "'ipu1_csi0':2 -> 'ipu1_csi0 capture':0[1]"
# Setup links for OV5640
media-ctl -l "'ov5640 1-0040':0 -> 'imx6-mipi-csi2':0[1]"
media-ctl -l "'imx6-mipi-csi2':2 -> 'ipu1_csi1':0[1]"
media-ctl -l "'ipu1_csi1':2 -> 'ipu1_csi1 capture':0[1]"
```

```
# Configure pads for OV5642 pipeline
media-ctl -V "'ov5642 1-0042':0 [fmt:YUYV2X8/640x480 field:none]"
media-ctl -V "'ipu1_csi0_mux':2 [fmt:YUYV2X8/640x480 field:none]"
media-ctl -V "'ipu1_csi0':2 [fmt:AYUV32/640x480 field:none]"
# Configure pads for OV5640 pipeline
media-ctl -V "'ov5640 1-0040':0 [fmt:UYVY2X8/640x480 field:none]"
media-ctl -V "'imx6-mipi-csi2':2 [fmt:UYVY2X8/640x480 field:none]"
media-ctl -V "'ipu1_csi1':2 [fmt:AYUV32/640x480 field:none]"
```

Streaming can then begin independently on the capture device nodes ”ipu1_csi0 capture” and ”ipu1_csi1 capture”. The v4l2-ctl tool can be used to select any supported YUV pixelformat on the capture device nodes, including planar.

i.MX6Q SabreAuto with ADV7180 decoder

On the i.MX6Q SabreAuto, an on-board ADV7180 SD decoder is connected to the parallel bus input on the internal video mux to IPU1 CSI0.

The following example configures a pipeline to capture from the ADV7180 video decoder, assuming NTSC 720x480 input signals, using simple interweave (unconverted and without motion compensation). The adv7180 must output sequential or alternating fields (field type 'seq-bt' for NTSC, or 'alternate'):

```
# Setup links
media-ctl -l "'adv7180 3-0021':0 -> 'ipu1_csi0_mux':1[1]"
media-ctl -l "'ipu1_csi0_mux':2 -> 'ipu1_csi0':0[1]"
media-ctl -l "'ipu1_csi0':2 -> 'ipu1_csi0 capture':0[1]"
# Configure pads
media-ctl -V "'adv7180 3-0021':0 [fmt:UYVY2X8/720x480 field:seq-bt]"
media-ctl -V "'ipu1_csi0_mux':2 [fmt:UYVY2X8/720x480]"
media-ctl -V "'ipu1_csi0':2 [fmt:AYUV32/720x480]"
# Configure "ipu1_csi0 capture" interface (assumed at /dev/video4)
v4l2-ctl -d4 --set-fmt-video=field=interlaced_bt
```

Streaming can then begin on /dev/video4. The v4l2-ctl tool can also be used to select any supported YUV pixelformat on /dev/video4.

This example configures a pipeline to capture from the ADV7180 video decoder, assuming PAL 720x576 input signals, with Motion Compensated de-interlacing. The adv7180 must output sequential or alternating fields (field type 'seq-tb' for PAL, or 'alternate').

```
# Setup links
media-ctl -l "'adv7180 3-0021':0 -> 'ipu1_csi0_mux':1[1]"
media-ctl -l "'ipu1_csi0_mux':2 -> 'ipu1_csi0':0[1]"
media-ctl -l "'ipu1_csi0':1 -> 'ipu1_vdic':0[1]"
media-ctl -l "'ipu1_vdic':2 -> 'ipu1_ic_prp':0[1]"
media-ctl -l "'ipu1_ic_prp':2 -> 'ipu1_ic_prpvf':0[1]"
media-ctl -l "'ipu1_ic_prpvf':1 -> 'ipu1_ic_prpvf capture':0[1]"
# Configure pads
media-ctl -V "'adv7180 3-0021':0 [fmt:UYVY2X8/720x576 field:seq-tb]"
media-ctl -V "'ipu1_csi0_mux':2 [fmt:UYVY2X8/720x576]"
```

```
media-ctl -V "'ipu1_csi0':1 [fmt:AYUV32/720x576]"
media-ctl -V "'ipu1_vdic':2 [fmt:AYUV32/720x576 field:none]"
media-ctl -V "'ipu1_ic_prp':2 [fmt:AYUV32/720x576 field:none]"
media-ctl -V "'ipu1_ic_prpvf':1 [fmt:AYUV32/720x576 field:none]"
# Configure "ipu1_ic_prpvf capture" interface (assumed at /dev/video2)
v4l2-ctl -d2 --set-fmt-video=field=none
```

Streaming can then begin on /dev/video2. The v4l2-ctl tool can also be used to select any supported YUV pixelformat on /dev/video2.

This platform accepts Composite Video analog inputs to the ADV7180 on Ain1 (connector J42).

i.MX6DL SabreAuto with ADV7180 decoder

On the i.MX6DL SabreAuto, an on-board ADV7180 SD decoder is connected to the parallel bus input on the internal video mux to IPU1 CSI0.

The following example configures a pipeline to capture from the ADV7180 video decoder, assuming NTSC 720x480 input signals, using simple interweave (unconverted and without motion compensation). The adv7180 must output sequential or alternating fields (field type 'seq-bt' for NTSC, or 'alternate'):

```
# Setup links
media-ctl -l "'adv7180 4-0021':0 -> 'ipu1_csi0_mux':4[1]"
media-ctl -l "'ipu1_csi0_mux':5 -> 'ipu1_csi0':0[1]"
media-ctl -l "'ipu1_csi0':2 -> 'ipu1_csi0 capture':0[1]"
# Configure pads
media-ctl -V "'adv7180 4-0021':0 [fmt:UYVY2X8/720x480 field:seq-bt]"
media-ctl -V "'ipu1_csi0_mux':5 [fmt:UYVY2X8/720x480]"
media-ctl -V "'ipu1_csi0':2 [fmt:AYUV32/720x480]"
# Configure "ipu1_csi0 capture" interface (assumed at /dev/video0)
v4l2-ctl -d0 --set-fmt-video=field=interlaced_bt
```

Streaming can then begin on /dev/video0. The v4l2-ctl tool can also be used to select any supported YUV pixelformat on /dev/video0.

This example configures a pipeline to capture from the ADV7180 video decoder, assuming PAL 720x576 input signals, with Motion Compensated de-interlacing. The adv7180 must output sequential or alternating fields (field type 'seq-tb' for PAL, or 'alternate').

```
# Setup links
media-ctl -l "'adv7180 4-0021':0 -> 'ipu1_csi0_mux':4[1]"
media-ctl -l "'ipu1_csi0_mux':5 -> 'ipu1_csi0':0[1]"
media-ctl -l "'ipu1_csi0':1 -> 'ipu1_vdic':0[1]"
media-ctl -l "'ipu1_vdic':2 -> 'ipu1_ic_prp':0[1]"
media-ctl -l "'ipu1_ic_prp':2 -> 'ipu1_ic_prpvf':0[1]"
media-ctl -l "'ipu1_ic_prpvf':1 -> 'ipu1_ic_prpvf capture':0[1]"
# Configure pads
media-ctl -V "'adv7180 4-0021':0 [fmt:UYVY2X8/720x576 field:seq-tb]"
media-ctl -V "'ipu1_csi0_mux':5 [fmt:UYVY2X8/720x576]"
media-ctl -V "'ipu1_csi0':1 [fmt:AYUV32/720x576]"
media-ctl -V "'ipu1_vdic':2 [fmt:AYUV32/720x576 field:none]"
```

```
media-ctl -V "'ipu1_ic_prp':2 [fmt:AYUV32/720x576 field:none]"
media-ctl -V "'ipu1_ic_prpvf':1 [fmt:AYUV32/720x576 field:none]"
# Configure "ipu1_ic_prpvf capture" interface (assumed at /dev/video2)
v4l2-ctl -d2 --set-fmt-video=field=none
```

Streaming can then begin on /dev/video2. The v4l2-ctl tool can also be used to select any supported YUV pixelformat on /dev/video2.

This platform accepts Composite Video analog inputs to the ADV7180 on Ain1 (connector J42).

i.MX6Q SabreSD with MIPI CSI-2 OV5640

Similarly to i.MX6Q SabreLite, the i.MX6Q SabreSD supports a parallel interface OV5642 module on IPU1 CSI0, and a MIPI CSI-2 OV5640 module. The OV5642 connects to i2c bus 1 and the OV5640 to i2c bus 2.

The device tree for SabreSD includes OF graphs for both the parallel OV5642 and the MIPI CSI-2 OV5640, but as of this writing only the MIPI CSI-2 OV5640 has been tested, so the OV5642 node is currently disabled. The OV5640 module connects to MIPI connector J5. The NXP part number for the OV5640 module that connects to the SabreSD board is H120729.

The following example configures unprocessed video capture pipeline to capture from the OV5640, transmitting on MIPI CSI-2 virtual channel 0:

```
# Setup links
media-ctl -l "'ov5640 1-003c':0 -> 'imx6-mipi-csi2':0[1]"
media-ctl -l "'imx6-mipi-csi2':1 -> 'ipu1_csi0_mux':0[1]"
media-ctl -l "'ipu1_csi0_mux':2 -> 'ipu1_csi0':0[1]"
media-ctl -l "'ipu1_csi0':2 -> 'ipu1_csi0 capture':0[1]"
# Configure pads
media-ctl -V "'ov5640 1-003c':0 [fmt:UYVY2X8/640x480]"
media-ctl -V "'imx6-mipi-csi2':1 [fmt:UYVY2X8/640x480]"
media-ctl -V "'ipu1_csi0_mux':0 [fmt:UYVY2X8/640x480]"
media-ctl -V "'ipu1_csi0':0 [fmt:AYUV32/640x480]"
```

Streaming can then begin on ”ipu1_csi0 capture” node. The v4l2-ctl tool can be used to select any supported pixelformat on the capture device node.

To determine what is the /dev/video node correspondent to ”ipu1_csi0 capture”:

```
media-ctl -e "ipu1_csi0 capture"
/dev/video0
```

/dev/video0 is the streaming element in this case.

Starting the streaming via v4l2-ctl:

```
v4l2-ctl --stream-mmap -d /dev/video0
```

Starting the streaming via Gstreamer and sending the content to the display:

```
gst-launch-1.0 v4l2src device=/dev/video0 ! kmssink
```

The following example configures a direct conversion pipeline to capture from the OV5640, transmitting on MIPI CSI-2 virtual channel 0. It also shows colorspace conversion and scaling at IC output.

```
# Setup links
media-ctl -l "'ov5640 1-003c':0 -> 'imx6-mipi-csi2':0[1]"
media-ctl -l "'imx6-mipi-csi2':1 -> 'ipu1_csi0_mux':0[1]"
media-ctl -l "'ipu1_csi0_mux':2 -> 'ipu1_csi0':0[1]"
media-ctl -l "'ipu1_csi0':1 -> 'ipu1_ic_prp':0[1]"
media-ctl -l "'ipu1_ic_prp':1 -> 'ipu1_ic_prpenc':0[1]"
media-ctl -l "'ipu1_ic_prpenc':1 -> 'ipu1_ic_prpenc capture':0[1]"
# Configure pads
media-ctl -V "'ov5640 1-003c':0 [fmt:UYVY2X8/640x480]"
media-ctl -V "'imx6-mipi-csi2':1 [fmt:UYVY2X8/640x480]"
media-ctl -V "'ipu1_csi0_mux':2 [fmt:UYVY2X8/640x480]"
media-ctl -V "'ipu1_csi0':1 [fmt:AYUV32/640x480]"
media-ctl -V "'ipu1_ic_prp':1 [fmt:AYUV32/640x480]"
media-ctl -V "'ipu1_ic_prpenc':1 [fmt:ARGB8888_1X32/800x600]"
# Set a format at the capture interface
v4l2-ctl -d /dev/video1 --set-fmt-video=pixelformat=RGB3
```

Streaming can then begin on ”ipu1_ic_prpenc capture” node.

To determine what is the /dev/video node correspondent to ”ipu1_ic_prpenc capture”:

```
media-ctl -e "ipu1_ic_prpenc capture"
/dev/video1
```

/dev/video1 is the streaming element in this case.

Starting the streaming via v4l2-ctl:

```
v4l2-ctl --stream-mmap -d /dev/video1
```

Starting the streaming via Gstreamer and sending the content to the display:

```
gst-launch-1.0 v4l2src device=/dev/video1 ! kmssink
```

Known Issues

1. When using 90 or 270 degree rotation control at capture resolutions near the IC resizer limit of 1024x1024, and combined with planar pixel formats (YUV420, YUV422p), frame capture will often fail with no end-of-frame interrupts from the IDMAC channel. To work around this, use lower resolution and/or packed formats (YUYV, RGB3, etc.) when 90 or 270 rotations are needed.

File list

drivers/staging/media/imx/ include/media/imx.h include/linux/imx-media.h

References

Authors

- Steve Longerbeam <steve_longerbeam@mentor.com>
- Philipp Zabel <kernel@pengutronix.de>
- Russell King <linux@armlinux.org.uk>

Copyright (C) 2012-2017 Mentor Graphics Inc.

59.7.6 i.MX7 Video Capture Driver

Introduction

The i.MX7 contrary to the i.MX5/6 family does not contain an Image Processing Unit (IPU); because of that the capabilities to perform operations or manipulation of the capture frames are less feature rich.

For image capture the i.MX7 has three units: - CMOS Sensor Interface (CSI) - Video Multiplexer - MIPI CSI-2 Receiver

```
MIPI Camera Input ---> MIPI CSI-2 --- > |\
                                        | \
                                        |  \
                                        | M |
                                        | U | ------>  CSI ---> Capture
                                        | X |
                                        |  /
Parallel Camera Input ----------------> | /
                                        |/
```

For additional information, please refer to the latest versions of the i.MX7 reference manual[1].

Entities

imx-mipi-csi2

This is the MIPI CSI-2 receiver entity. It has one sink pad to receive the pixel data from MIPI CSI-2 camera sensor. It has one source pad, corresponding to the virtual channel 0. This module is compliant to previous version of Samsung D-phy, and supports two D-PHY Rx Data lanes.

[1] https://www.nxp.com/docs/en/reference-manual/IMX7SRM.pdf

csi-mux

This is the video multiplexer. It has two sink pads to select from either camera sensor with a parallel interface or from MIPI CSI-2 virtual channel 0. It has a single source pad that routes to the CSI.

csi

The CSI enables the chip to connect directly to external CMOS image sensor. CSI can interface directly with Parallel and MIPI CSI-2 buses. It has 256 x 64 FIFO to store received image pixel data and embedded DMA controllers to transfer data from the FIFO through AHB bus.

This entity has one sink pad that receives from the csi-mux entity and a single source pad that routes video frames directly to memory buffers. This pad is routed to a capture device node.

Usage Notes

To aid in configuration and for backward compatibility with V4L2 applications that access controls only from video device nodes, the capture device interfaces inherit controls from the active entities in the current pipeline, so controls can be accessed either directly from the subdev or from the active capture device interface. For example, the sensor controls are available either from the sensor subdevs or from the active capture device.

Warp7 with OV2680

On this platform an OV2680 MIPI CSI-2 module is connected to the internal MIPI CSI-2 receiver. The following example configures a video capture pipeline with an output of 800x600, and BGGR 10 bit bayer format:

```
# Setup links
media-ctl -l "'ov2680 1-0036':0 -> 'imx7-mipi-csis.0':0[1]"
media-ctl -l "'imx7-mipi-csis.0':1 -> 'csi-mux':1[1]"
media-ctl -l "'csi-mux':2 -> 'csi':0[1]"
media-ctl -l "'csi':1 -> 'csi capture':0[1]"

# Configure pads for pipeline
media-ctl -V "'ov2680 1-0036':0 [fmt:SBGGR10_1X10/800x600 field:none]"
media-ctl -V "'csi-mux':1 [fmt:SBGGR10_1X10/800x600 field:none]"
media-ctl -V "'csi-mux':2 [fmt:SBGGR10_1X10/800x600 field:none]"
media-ctl -V "'imx7-mipi-csis.0':0 [fmt:SBGGR10_1X10/800x600 field:none]"
media-ctl -V "'csi':0 [fmt:SBGGR10_1X10/800x600 field:none]"
```

After this streaming can start. The v4l2-ctl tool can be used to select any of the resolutions supported by the sensor.

```
# media-ctl -p
Media controller API version 5.2.0

Media device information
------------------------
```

```
driver          imx7-csi
model           imx-media
serial
bus info
hw revision     0x0
driver version  5.2.0

Device topology
- entity 1: csi (2 pads, 2 links)
            type V4L2 subdev subtype Unknown flags 0
            device node name /dev/v4l-subdev0
        pad0: Sink
                [fmt:SBGGR10_1X10/800x600 field:none colorspace:srgb xfer:srgb␣
↪ycbcr:601 quantization:full-range]
                <- "csi-mux":2 [ENABLED]
        pad1: Source
                [fmt:SBGGR10_1X10/800x600 field:none colorspace:srgb xfer:srgb␣
↪ycbcr:601 quantization:full-range]
                -> "csi capture":0 [ENABLED]

- entity 4: csi capture (1 pad, 1 link)
            type Node subtype V4L flags 0
            device node name /dev/video0
        pad0: Sink
                <- "csi":1 [ENABLED]

- entity 10: csi-mux (3 pads, 2 links)
             type V4L2 subdev subtype Unknown flags 0
             device node name /dev/v4l-subdev1
        pad0: Sink
                [fmt:Y8_1X8/1x1 field:none]
        pad1: Sink
               [fmt:SBGGR10_1X10/800x600 field:none]
                <- "imx7-mipi-csis.0":1 [ENABLED]
        pad2: Source
                [fmt:SBGGR10_1X10/800x600 field:none]
                -> "csi":0 [ENABLED]

- entity 14: imx7-mipi-csis.0 (2 pads, 2 links)
             type V4L2 subdev subtype Unknown flags 0
             device node name /dev/v4l-subdev2
        pad0: Sink
                [fmt:SBGGR10_1X10/800x600 field:none]
                <- "ov2680 1-0036":0 [ENABLED]
        pad1: Source
                [fmt:SBGGR10_1X10/800x600 field:none]
                -> "csi-mux":1 [ENABLED]

- entity 17: ov2680 1-0036 (1 pad, 1 link)
             type V4L2 subdev subtype Sensor flags 0
```

```
             device node name /dev/v4l-subdev3
        pad0: Source
                [fmt:SBGGR10_1X10/800x600@1/30 field:none colorspace:srgb]
                -> "imx7-mipi-csis.0":0 [ENABLED]
```

i.MX6ULL-EVK with OV5640

On this platform a parallel OV5640 sensor is connected to the CSI port. The following example configures a video capture pipeline with an output of 640x480 and UYVY8_2X8 format:

```
# Setup links
media-ctl -l "'ov5640 1-003c':0 -> 'csi':0[1]"
media-ctl -l "'csi':1 -> 'csi capture':0[1]"

# Configure pads for pipeline
media-ctl -v -V "'ov5640 1-003c':0 [fmt:UYVY8_2X8/640x480 field:none]"
```

After this streaming can start:

```
gst-launch-1.0 -v v4l2src device=/dev/video1 ! video/x-raw,format=UYVY,
↪width=640,height=480 ! v4l2convert ! fbdevsink
```

```
# media-ctl -p
Media controller API version 5.14.0

Media device information
------------------------
driver          imx7-csi
model           imx-media
serial
bus info
hw revision     0x0
driver version  5.14.0

Device topology
- entity 1: csi (2 pads, 2 links)
            type V4L2 subdev subtype Unknown flags 0
            device node name /dev/v4l-subdev0
        pad0: Sink
                [fmt:UYVY8_2X8/640x480 field:none colorspace:srgb xfer:srgb␣
↪ycbcr:601 quantization:full-range]
                <- "ov5640 1-003c":0 [ENABLED,IMMUTABLE]
        pad1: Source
                [fmt:UYVY8_2X8/640x480 field:none colorspace:srgb xfer:srgb␣
↪ycbcr:601 quantization:full-range]
                -> "csi capture":0 [ENABLED,IMMUTABLE]

- entity 4: csi capture (1 pad, 1 link)
            type Node subtype V4L flags 0
            device node name /dev/video1
```

```
        pad0: Sink
                <- "csi":1 [ENABLED,IMMUTABLE]

- entity 10: ov5640 1-003c (1 pad, 1 link)
             type V4L2 subdev subtype Sensor flags 0
             device node name /dev/v4l-subdev1
        pad0: Source
                [fmt:UYVY8_2X8/640x480@1/30 field:none colorspace:srgb␣
↪xfer:srgb ycbcr:601 quantization:full-range]
                -> "csi":0 [ENABLED,IMMUTABLE]
```

References

59.7.7 Intel Image Processing Unit 3 (IPU3) Imaging Unit (ImgU) driver

Copyright © 2018 Intel Corporation

Introduction

This file documents the Intel IPU3 (3rd generation Image Processing Unit) Imaging Unit drivers located under drivers/media/pci/intel/ipu3 (CIO2) as well as under drivers/staging/media/ipu3 (ImgU).

The Intel IPU3 found in certain Kaby Lake (as well as certain Sky Lake) platforms (U/Y processor lines) is made up of two parts namely the Imaging Unit (ImgU) and the CIO2 device (MIPI CSI2 receiver).

The CIO2 device receives the raw Bayer data from the sensors and outputs the frames in a format that is specific to the IPU3 (for consumption by the IPU3 ImgU). The CIO2 driver is available as drivers/media/pci/intel/ipu3/ipu3-cio2* and is enabled through the CONFIG_VIDEO_IPU3_CIO2 config option.

The Imaging Unit (ImgU) is responsible for processing images captured by the IPU3 CIO2 device. The ImgU driver sources can be found under drivers/staging/media/ipu3 directory. The driver is enabled through the CONFIG_VIDEO_IPU3_IMGU config option.

The two driver modules are named ipu3_csi2 and ipu3_imgu, respectively.

The drivers has been tested on Kaby Lake platforms (U/Y processor lines).

Both of the drivers implement V4L2, Media Controller and V4L2 sub-device interfaces. The IPU3 CIO2 driver supports camera sensors connected to the CIO2 MIPI CSI-2 interfaces through V4L2 sub-device sensor drivers.

CIO2

The CIO2 is represented as a single V4L2 subdev, which provides a V4L2 subdev interface to the user space. There is a video node for each CSI-2 receiver, with a single media controller interface for the entire device.

The CIO2 contains four independent capture channel, each with its own MIPI CSI-2 receiver and DMA engine. Each channel is modelled as a V4L2 sub-device exposed to userspace as a V4L2 sub-device node and has two pads:

Pad	Direction	Purpose
0	sink	MIPI CSI-2 input, connected to the sensor subdev
1	source	Raw video capture, connected to the V4L2 video interface

The V4L2 video interfaces model the DMA engines. They are exposed to userspace as V4L2 video device nodes.

Capturing frames in raw Bayer format

CIO2 MIPI CSI2 receiver is used to capture frames (in packed raw Bayer format) from the raw sensors connected to the CSI2 ports. The captured frames are used as input to the ImgU driver.

Image processing using IPU3 ImgU requires tools such as raw2pnm[2], and yavta[3] due to the following unique requirements and / or features specific to IPU3.

-- The IPU3 CSI2 receiver outputs the captured frames from the sensor in packed raw Bayer format that is specific to IPU3.

-- Multiple video nodes have to be operated simultaneously.

Let us take the example of ov5670 sensor connected to CSI2 port 0, for a 2592x1944 image capture.

Using the media controller APIs, the ov5670 sensor is configured to send frames in packed raw Bayer format to IPU3 CSI2 receiver.

```
# This example assumes /dev/media0 as the CIO2 media device
export MDEV=/dev/media0

# and that ov5670 sensor is connected to i2c bus 10 with address 0x36
export SDEV=$(media-ctl -d $MDEV -e "ov5670 10-0036")

# Establish the link for the media devices using media-ctl [#f3]_
media-ctl -d $MDEV -l "ov5670:0 -> ipu3-csi2 0:0[1]"
```

[2] https://github.com/intel/nvt
[3] http://git.ideasonboard.org/yavta.git

```
# Set the format for the media devices
media-ctl -d $MDEV -V "ov5670:0 [fmt:SGRBG10/2592x1944]"
media-ctl -d $MDEV -V "ipu3-csi2 0:0 [fmt:SGRBG10/2592x1944]"
media-ctl -d $MDEV -V "ipu3-csi2 0:1 [fmt:SGRBG10/2592x1944]"
```

Once the media pipeline is configured, desired sensor specific settings (such as exposure and gain settings) can be set, using the yavta tool.

e.g

```
yavta -w 0x009e0903 444 $SDEV
yavta -w 0x009e0913 1024 $SDEV
yavta -w 0x009e0911 2046 $SDEV
```

Once the desired sensor settings are set, frame captures can be done as below.

e.g

```
yavta --data-prefix -u -c10 -n5 -I -s2592x1944 --file=/tmp/frame-#.bin \
      -f IPU3_SGRBG10 $(media-ctl -d $MDEV -e "ipu3-cio2 0")
```

With the above command, 10 frames are captured at 2592x1944 resolution, with sGRBG10 format and output as IPU3_SGRBG10 format.

The captured frames are available as /tmp/frame-#.bin files.

ImgU

The ImgU is represented as two V4L2 subdevs, each of which provides a V4L2 subdev interface to the user space.

Each V4L2 subdev represents a pipe, which can support a maximum of 2 streams. This helps to support advanced camera features like Continuous View Finder (CVF) and Snapshot During Video(SDV).

The ImgU contains two independent pipes, each modelled as a V4L2 sub-device exposed to userspace as a V4L2 sub-device node.

Each pipe has two sink pads and three source pads for the following purpose:

Pad	Direction	Purpose
0	sink	Input raw video stream
1	sink	Processing parameters
2	source	Output processed video stream
3	source	Output viewfinder video stream
4	source	3A statistics

Each pad is connected to a corresponding V4L2 video interface, exposed to userspace as a V4L2 video device node.

Device operation

With ImgU, once the input video node (”ipu3-imgu 0/1”:0, in <entity>:<pad-number> format) is queued with buffer (in packed raw Bayer format), ImgU starts processing the buffer and produces the video output in YUV format and statistics output on respective output nodes. The driver is expected to have buffers ready for all of parameter, output and statistics nodes, when input video node is queued with buffer.

At a minimum, all of input, main output, 3A statistics and viewfinder video nodes should be enabled for IPU3 to start image processing.

Each ImgU V4L2 subdev has the following set of video nodes.

input, output and viewfinder video nodes

The frames (in packed raw Bayer format specific to the IPU3) received by the input video node is processed by the IPU3 Imaging Unit and are output to 2 video nodes, with each targeting a different purpose (main output and viewfinder output).

Details onand the Bayer format specific to the IPU3 can be found in v4l2-pix-fmt-ipu3-sbggr10.

The driver supports V4L2 Video Capture Interface as defined at devices.

Only the multi-planar API is supported. More details can be found at planar-apis.

Parameters video node

The parameters video node receives the ImgU algorithm parameters that are used to configure how the ImgU algorithms process the image.

Details on processing parameters specific to the IPU3 can be found in v4l2-meta-fmt-params.

3A statistics video node

3A statistics video node is used by the ImgU driver to output the 3A (auto focus, auto exposure and auto white balance) statistics for the frames that are being processed by the ImgU to user space applications. User space applications can use this statistics data to compute the desired algorithm parameters for the ImgU.

Configuring the Intel IPU3

The IPU3 ImgU pipelines can be configured using the Media Controller, defined at media_controller.

Running mode and firmware binary selection

ImgU works based on firmware, currently the ImgU firmware support run 2 pipes in time-sharing with single input frame data. Each pipe can run at certain mode - ”VIDEO” or ”STILL”, ”VIDEO” mode is commonly used for video frames capture, and ”STILL” is used for still frame capture. However, you can also select ”VIDEO” to capture still frames if you want to capture images with less system load and power. For ”STILL” mode, ImgU will try to use smaller BDS factor and output larger bayer frame for further YUV processing than ”VIDEO” mode to get high quality images. Besides, ”STILL” mode need XNR3 to do noise reduction, hence ”STILL” mode will need more power and memory bandwidth than ”VIDEO” mode. TNR will be enabled in ”VIDEO” mode and bypassed by ”STILL” mode. ImgU is running at ”VIDEO” mode by default, the user can use v4l2 control V4L2_CID_INTEL_IPU3_MODE (currently defined in drivers/staging/media/ipu3/include/uapi/intel-ipu3.h) to query and set the running mode. For user, there is no difference for buffer queueing between the ”VIDEO” and ”STILL” mode, mandatory input and main output node should be enabled and buffers need be queued, the statistics and the view-finder queues are optional.

The firmware binary will be selected according to current running mode, such log ”using binary if_to_osys_striped ” or ”using binary if_to_osys_primary_striped” could be observed if you enable the ImgU dynamic debug, the binary if_to_osys_striped is selected for ”VIDEO” and the binary ”if_to_osys_primary_striped” is selected for ”STILL”.

Processing the image in raw Bayer format

Configuring ImgU V4L2 subdev for image processing

The ImgU V4L2 subdevs have to be configured with media controller APIs to have all the video nodes setup correctly.

Let us take ”ipu3-imgu 0” subdev as an example.

```
media-ctl -d $MDEV -r
media-ctl -d $MDEV -l "ipu3-imgu 0 input":0 -> "ipu3-imgu 0":0[1]
media-ctl -d $MDEV -l "ipu3-imgu 0":2 -> "ipu3-imgu 0 output":0[1]
media-ctl -d $MDEV -l "ipu3-imgu 0":3 -> "ipu3-imgu 0 viewfinder":0[1]
media-ctl -d $MDEV -l "ipu3-imgu 0":4 -> "ipu3-imgu 0 3a stat":0[1]
```

Also the pipe mode of the corresponding V4L2 subdev should be set as desired (e.g 0 for video mode or 1 for still mode) through the control id 0x009819a1 as below.

```
yavta -w "0x009819A1 1" /dev/v4l-subdev7
```

Certain hardware blocks in ImgU pipeline can change the frame resolution by cropping or scaling, these hardware blocks include Input Feeder(IF), Bayer Down Scaler (BDS) and Geometric Distortion Correction (GDC). There is also a block which can change the frame resolution - YUV Scaler, it is only applicable to the secondary output.

RAW Bayer frames go through these ImgU pipeline hardware blocks and the final processed image output to the DDR memory.

Input Feeder

Fig. 3: IPU3 resolution change hardware blocks

Input Feeder gets the Bayer frame data from the sensor, it can enable cropping of lines and columns from the frame and then store pixels into device's internal pixel buffer which are ready to readout by following blocks.

Bayer Down Scaler

Bayer Down Scaler is capable of performing image scaling in Bayer domain, the downscale factor can be configured from 1X to 1/4X in each axis with configuration steps of 0.03125 (1/32).

Geometric Distortion Correction

Geometric Distortion Correction is used to perform correction of distortions and image filtering. It needs some extra filter and envelope padding pixels to work, so the input resolution of GDC should be larger than the output resolution.

YUV Scaler

YUV Scaler which similar with BDS, but it is mainly do image down scaling in YUV domain, it can support up to 1/12X down scaling, but it can not be applied to the main output.

The ImgU V4L2 subdev has to be configured with the supported resolutions in all the above hardware blocks, for a given input resolution. For a given supported resolution for an input frame, the Input Feeder, Bayer Down Scaler and GDC blocks should be configured with the supported resolutions as each hardware block has its own alignment requirement.

You must configure the output resolution of the hardware blocks smartly to meet the hardware requirement along with keeping the maximum field of view. The intermediate resolutions can be generated by specific tool -

https://github.com/intel/intel-ipu3-pipecfg

This tool can be used to generate intermediate resolutions. More information can be obtained by looking at the following IPU3 ImgU configuration table.

https://chromium.googlesource.com/chromiumos/overlays/board-overlays/+/master

Under baseboard-poppy/media-libs/cros-camera-hal-configs-poppy/files/gcss directory, graph_settings_ov5670.xml can be used as an example.

The following steps prepare the ImgU pipeline for the image processing.

1. The ImgU V4L2 subdev data format should be set by using the VIDIOC_SUBDEV_S_FMT on pad 0, using the GDC width and height obtained above.

2. The ImgU V4L2 subdev cropping should be set by using the VIDIOC_SUBDEV_S_SELECTION on pad 0, with V4L2_SEL_TGT_CROP as the target, using the input feeder height and width.

3. The ImgU V4L2 subdev composing should be set by using the VIDIOC_SUBDEV_S_SELECTION on pad 0, with V4L2_SEL_TGT_COMPOSE as the target, using the BDS height and width.

For the ov5670 example, for an input frame with a resolution of 2592x1944 (which is input to the ImgU subdev pad 0), the corresponding resolutions for input feeder, BDS and GDC are 2592x1944, 2592x1944 and 2560x1920 respectively.

Once this is done, the received raw Bayer frames can be input to the ImgU V4L2 subdev as below, using the open source application v4l2nPage 2291, 2.

For an image captured with 2592x1944[5] resolution, with desired output resolution as 2560x1920 and viewfinder resolution as 2560x1920, the following v4l2n command can be used. This helps process the raw Bayer frames and produces the desired results for the main output image and the viewfinder output, in NV12 format.

```
v4l2n --pipe=4 --load=/tmp/frame-#.bin --open=/dev/video4
      --fmt=type:VIDEO_OUTPUT_MPLANE,width=2592,height=1944,
↪pixelformat=0X47337069 \
      --reqbufs=type:VIDEO_OUTPUT_MPLANE,count:1 --pipe=1 \
      --output=/tmp/frames.out --open=/dev/video5 \
      --fmt=type:VIDEO_CAPTURE_MPLANE,width=2560,height=1920,pixelformat=NV12 \
      --reqbufs=type:VIDEO_CAPTURE_MPLANE,count:1 --pipe=2 \
      --output=/tmp/frames.vf --open=/dev/video6 \
      --fmt=type:VIDEO_CAPTURE_MPLANE,width=2560,height=1920,pixelformat=NV12 \
      --reqbufs=type:VIDEO_CAPTURE_MPLANE,count:1 --pipe=3 --open=/dev/video7 \
      --output=/tmp/frames.3A --fmt=type:META_CAPTURE,? \
      --reqbufs=count:1,type:META_CAPTURE --pipe=1,2,3,4 --stream=5
```

You can also use yavtaPage 2291, 3 command to do same thing as above:

```
yavta --data-prefix -Bcapture-mplane -c10 -n5 -I -s2592x1944 \
      --file=frame-#.out-f NV12 /dev/video5 & \
yavta --data-prefix -Bcapture-mplane -c10 -n5 -I -s2592x1944 \
      --file=frame-#.vf -f NV12 /dev/video6 & \
yavta --data-prefix -Bmeta-capture -c10 -n5 -I \
      --file=frame-#.3a /dev/video7 & \
yavta --data-prefix -Boutput-mplane -c10 -n5 -I -s2592x1944 \
      --file=/tmp/frame-in.cio2 -f IPU3_SGRBG10 /dev/video4
```

where /dev/video4, /dev/video5, /dev/video6 and /dev/video7 devices point to input, output, viewfinder and 3A statistics video nodes respectively.

[5] ImgU limitation requires an additional 16x16 for all input resolutions

Converting the raw Bayer image into YUV domain

The processed images after the above step, can be converted to YUV domain as below.

Main output frames

```
raw2pnm -x2560 -y1920 -fNV12 /tmp/frames.out /tmp/frames.out.ppm
```

where 2560x1920 is output resolution, NV12 is the video format, followed by input frame and output PNM file.

Viewfinder output frames

```
raw2pnm -x2560 -y1920 -fNV12 /tmp/frames.vf /tmp/frames.vf.ppm
```

where 2560x1920 is output resolution, NV12 is the video format, followed by input frame and output PNM file.

Example user space code for IPU3

User space code that configures and uses IPU3 is available here.

https://chromium.googlesource.com/chromiumos/platform/arc-camera/+/master/

The source can be located under hal/intel directory.

Overview of IPU3 pipeline

IPU3 pipeline has a number of image processing stages, each of which takes a set of parameters as input. The major stages of pipelines are shown here:

The table below presents a description of the above algorithms.

Fig. 4: IPU3 ImgU Pipeline Diagram

Name	Description
Optical Black Correction	Optical Black Correction block subtracts a pre-defined value from the respective pixel values to obtain better image quality. Defined in struct ipu3_uapi_obgrid_param.
Linearization	This algo block uses linearization parameters to address non-linearity sensor effects. The Lookup table table is defined in struct ipu3_uapi_isp_lin_vmem_params.
SHD	Lens shading correction is used to correct spatial non-uniformity of the pixel response due to optical lens shading. This is done by applying a different gain for each pixel. The gain, black level etc are configured in struct ipu3_uapi_shd_config_static.
BNR	Bayer noise reduction block removes image noise by applying a bilateral filter. See struct ipu3_uapi_bnr_static_config for details.
ANR	Advanced Noise Reduction is a block based algorithm that performs noise reduction in the Bayer domain. The convolution matrix etc can be found in struct ipu3_uapi_anr_config.
DM	Demosaicing converts raw sensor data in Bayer format into RGB (Red, Green, Blue) presentation. Then add outputs of estimation of Y channel for following stream processing by Firmware. The struct is defined as struct ipu3_uapi_dm_config.
Color Correction	Color Correction algo transforms sensor specific color space to the standard "sRGB" color space. This is done by applying 3x3 matrix defined in struct ipu3_uapi_ccm_mat_config.
Gamma correction	Gamma correction struct ipu3_uapi_gamma_config is a basic non-linear tone mapping correction that is applied per pixel for each pixel component.
CSC	Color space conversion transforms each pixel from the RGB primary presentation to YUV (Y: brightness, UV: Luminance) presentation. This is done by applying a 3x3 matrix defined in struct ipu3_uapi_csc_mat_config
CDS	Chroma down sampling After the CSC is performed, the Chroma Down Sampling is applied for a UV plane down sampling by a factor of 2 in each direction for YUV 4:2:0 using a 4x2 configurable filter struct ipu3_uapi_cds_params.
CHNR	Chroma noise reduction This block processes only the chrominance pixels and performs noise reduction by cleaning the high frequency noise. See struct struct ipu3_uapi_yuvp1_chnr_config.
TCC	Total color correction as defined in struct struct ipu3_uapi_yuvp2_tcc_static_config.
XNR3	eXtreme Noise Reduction V3 is the third revision of noise reduction algorithm used to improve image quality. This removes the low frequency noise in the captured image. Two related structs are being defined, struct ipu3_uapi_isp_xnr3_params for ISP data memory and struct ipu3_uapi_isp_xnr3_vmem_params for vector memory.
TNR	Temporal Noise Reduction block compares successive frames in time to remove anomalies / noise in pixel values. struct ipu3_uapi_isp_tnr3_vmem_params and struct ipu3_uapi_isp_tnr3_params are defined for ISP vector and data memory respectively.

Other often encountered acronyms not listed in above table:

ACC
Accelerator cluster

AWB_FR
Auto white balance filter response statistics

BDS
Bayer downscaler parameters

CCM
Color correction matrix coefficients

IEFd
Image enhancement filter directed

Obgrid
Optical black level compensation

OSYS
Output system configuration

ROI
Region of interest

YDS
Y down sampling

YTM
Y-tone mapping

A few stages of the pipeline will be executed by firmware running on the ISP processor, while many others will use a set of fixed hardware blocks also called accelerator cluster (ACC) to crunch pixel data and produce statistics.

ACC parameters of individual algorithms, as defined by struct ipu3_uapi_acc_param, can be chosen to be applied by the user space through struct struct ipu3_uapi_flags embedded in struct ipu3_uapi_params structure. For parameters that are configured as not enabled by the user space, the corresponding structs are ignored by the driver, in which case the existing configuration of the algorithm will be preserved.

References

59.7.8 The ivtv driver

Author: Hans Verkuil <hverkuil@xs4all.nl>

This is a v4l2 device driver for the Conexant cx23415/6 MPEG encoder/decoder. The cx23415 can do both encoding and decoding, the cx23416 can only do MPEG encoding. Currently the only card featuring full decoding support is the Hauppauge PVR-350.

Note:

1) This driver requires the latest encoder firmware (version 2.06.039, size 376836 bytes). Get the firmware from here:

   https://linuxtv.org/downloads/firmware/#conexant

2) 'normal' TV applications do not work with this driver, you need an application that can handle MPEG input such as mplayer, xine, MythTV, etc.

The primary goal of the IVTV project is to provide a ”clean room” Linux Open Source driver implementation for video capture cards based on the iCompression iTVC15 or Conexant CX23415/CX23416 MPEG Codec.

Features

- Hardware mpeg2 capture of broadcast video (and sound) via the tuner or S-Video/Composite and audio line-in.
- Hardware mpeg2 capture of FM radio where hardware support exists
- Supports NTSC, PAL, SECAM with stereo sound
- Supports SAP and bilingual transmissions.
- Supports raw VBI (closed captions and teletext).
- Supports sliced VBI (closed captions and teletext) and is able to insert this into the captured MPEG stream.
- Supports raw YUV and PCM input.

Additional features for the PVR-350 (CX23415 based)

- Provides hardware mpeg2 playback
- Provides comprehensive OSD (On Screen Display: ie. graphics overlaying the video signal)
- Provides a framebuffer (allowing X applications to appear on the video device)
- Supports raw YUV output.

IMPORTANT: In case of problems first read this page:
https://help.ubuntu.com/community/Install_IVTV_Troubleshooting

See also

https://linuxtv.org

IRC

irc://irc.freenode.net/#v4l

Devices

A maximum of 12 ivtv boards are allowed at the moment.

Cards that don't have a video output capability (i.e. non PVR350 cards) lack the vbi8, vbi16, video16 and video48 devices. They also do not support the framebuffer device /dev/fbx for OSD.

The radio0 device may or may not be present, depending on whether the card has a radio tuner or not.

Here is a list of the base v4l devices:

```
crw-rw----    1 root     video     81,   0 Jun 19 22:22 /dev/video0
crw-rw----    1 root     video     81,  16 Jun 19 22:22 /dev/video16
crw-rw----    1 root     video     81,  24 Jun 19 22:22 /dev/video24
crw-rw----    1 root     video     81,  32 Jun 19 22:22 /dev/video32
crw-rw----    1 root     video     81,  48 Jun 19 22:22 /dev/video48
crw-rw----    1 root     video     81,  64 Jun 19 22:22 /dev/radio0
crw-rw----    1 root     video     81, 224 Jun 19 22:22 /dev/vbi0
crw-rw----    1 root     video     81, 228 Jun 19 22:22 /dev/vbi8
crw-rw----    1 root     video     81, 232 Jun 19 22:22 /dev/vbi16
```

Base devices

For every extra card you have the numbers increased by one. For example, /dev/video0 is listed as the 'base' encoding capture device so we have:

- /dev/video0 is the encoding capture device for the first card (card 0)
- /dev/video1 is the encoding capture device for the second card (card 1)
- /dev/video2 is the encoding capture device for the third card (card 2)

Note that if the first card doesn't have a feature (eg no decoder, so no video16, the second card will still use video17. The simple rule is 'add the card number to the base device number'. If you have other capture cards (e.g. WinTV PCI) that are detected first, then you have to tell the ivtv module about it so that it will start counting at 1 (or 2, or whatever). Otherwise the device numbers can get confusing. The ivtv 'ivtv_first_minor' module option can be used for that.

- /dev/video0

  The encoding capture device(s).

  Read-only.

  Reading from this device gets you the MPEG1/2 program stream. Example:

  ```
  cat /dev/video0 > my.mpg (you need to hit ctrl-c to exit)
  ```

- /dev/video16

  The decoder output device(s)

  Write-only. Only present if the MPEG decoder (i.e. CX23415) exists.

  An mpeg2 stream sent to this device will appear on the selected video display, audio will appear on the line-out/audio out. It is only available for cards that support video out. Example:

```
cat my.mpg >/dev/video16
```

- /dev/video24

  The raw audio capture device(s).

  Read-only

  The raw audio PCM stereo stream from the currently selected tuner or audio line-in. Reading from this device results in a raw (signed 16 bit Little Endian, 48000 Hz, stereo pcm) capture. This device only captures audio. This should be replaced by an ALSA device in the future. Note that there is no corresponding raw audio output device, this is not supported in the decoder firmware.

- /dev/video32

  The raw video capture device(s)

  Read-only

  The raw YUV video output from the current video input. The YUV format is a 16x16 linear tiled NV12 format (V4L2_PIX_FMT_NV12_16L16)

  Note that the YUV and PCM streams are not synchronized, so they are of limited use.

- /dev/video48

  The raw video display device(s)

  Write-only. Only present if the MPEG decoder (i.e. CX23415) exists.

  Writes a YUV stream to the decoder of the card.

- /dev/radio0

  The radio tuner device(s)

  Cannot be read or written.

  Used to enable the radio tuner and tune to a frequency. You cannot read or write audio streams with this device. Once you use this device to tune the radio, use /dev/video24 to read the raw pcm stream or /dev/video0 to get an mpeg2 stream with black video.

- /dev/vbi0

  The 'vertical blank interval' (Teletext, CC, WSS etc) capture device(s)

  Read-only

  Captures the raw (or sliced) video data sent during the Vertical Blank Interval. This data is used to encode teletext, closed captions, VPS, widescreen signalling, electronic program guide information, and other services.

- /dev/vbi8

  Processed vbi feedback device(s)

  Read-only. Only present if the MPEG decoder (i.e. CX23415) exists.

  The sliced VBI data embedded in an MPEG stream is reproduced on this device. So while playing back a recording on /dev/video16, you can read the embedded VBI data from /dev/vbi8.

- /dev/vbi16

  The vbi 'display' device(s)

  Write-only. Only present if the MPEG decoder (i.e. CX23415) exists.

  Can be used to send sliced VBI data to the video-out connector.

59.7.9 mgb4 sysfs interface

The mgb4 driver provides a sysfs interface, that is used to configure video stream related parameters (some of them must be set properly before the v4l2 device can be opened) and obtain the video device/stream status.

There are two types of parameters - global / PCI card related, found under `/sys/class/video4linux/videoX/device` and module specific found under `/sys/class/video4linux/videoX`.

Global (PCI card) parameters

module_type (R):
Module type.

0 - No module present
1 - FPDL3
2 - GMSL

module_version (R):
Module version number. Zero in case of a missing module.

fw_type (R):
Firmware type.

1 - FPDL3
2 - GMSL

fw_version (R):
Firmware version number.

serial_number (R):
Card serial number. The format is:

```
PRODUCT-REVISION-SERIES-SERIAL
```

where each component is a 8b number.

Common FPDL3/GMSL input parameters

input_id (R):
Input number ID, zero based.

oldi_lane_width (RW):
Number of deserializer output lanes.

0 - single
1 - dual (default)

color_mapping (RW):
Mapping of the incoming bits in the signal to the colour bits of the pixels.

0 - OLDI/JEIDA
1 - SPWG/VESA (default)

link_status (R):
Video link status. If the link is locked, chips are properly connected and communicating at the same speed and protocol. The link can be locked without an active video stream.

A value of 0 is equivalent to the V4L2_IN_ST_NO_SYNC flag of the V4L2 VIDIOC_ENUMINPUT status bits.

0 - unlocked
1 - locked

stream_status (R):
Video stream status. A stream is detected if the link is locked, the input pixel clock is running and the DE signal is moving.

A value of 0 is equivalent to the V4L2_IN_ST_NO_SIGNAL flag of the V4L2 VIDIOC_ENUMINPUT status bits.

0 - not detected
1 - detected

video_width (R):
Video stream width. This is the actual width as detected by the HW.

The value is identical to what VIDIOC_QUERY_DV_TIMINGS returns in the width field of the v4l2_bt_timings struct.

video_height (R):
Video stream height. This is the actual height as detected by the HW.

The value is identical to what VIDIOC_QUERY_DV_TIMINGS returns in the height field of the v4l2_bt_timings struct.

vsync_status (R):
The type of VSYNC pulses as detected by the video format detector.

The value is equivalent to the flags returned by VIDIOC_QUERY_DV_TIMINGS in the polarities field of the v4l2_bt_timings struct.

0 - active low
1 - active high
2 - not available

hsync_status (R):
The type of HSYNC pulses as detected by the video format detector.

The value is equivalent to the flags returned by VIDIOC_QUERY_DV_TIMINGS in the polarities field of the v4l2_bt_timings struct.

0 - active low
1 - active high
2 - not available

vsync_gap_length (RW):
If the incoming video signal does not contain synchronization VSYNC and HSYNC pulses, these must be generated internally in the FPGA to achieve the correct frame ordering. This value indicates, how many ”empty” pixels (pixels with deasserted Data Enable signal) are necessary to generate the internal VSYNC pulse.

hsync_gap_length (RW):
If the incoming video signal does not contain synchronization VSYNC and HSYNC pulses, these must be generated internally in the FPGA to achieve the correct frame ordering. This value indicates, how many ”empty” pixels (pixels with deasserted Data Enable signal) are necessary to generate the internal HSYNC pulse. The value must be greater than 1 and smaller than vsync_gap_length.

pclk_frequency (R):
Input pixel clock frequency in kHz.

The value is identical to what VIDIOC_QUERY_DV_TIMINGS returns in the pixelclock field of the v4l2_bt_timings struct.

Note: The frequency_range parameter must be set properly first to get a valid frequency here.

hsync_width (R):
Width of the HSYNC signal in PCLK clock ticks.

The value is identical to what VIDIOC_QUERY_DV_TIMINGS returns in the hsync field of the v4l2_bt_timings struct.

vsync_width (R):
Width of the VSYNC signal in PCLK clock ticks.

The value is identical to what VIDIOC_QUERY_DV_TIMINGS returns in the vsync field of the v4l2_bt_timings struct.

hback_porch (R):
Number of PCLK pulses between deassertion of the HSYNC signal and the first valid pixel in the video line (marked by DE=1).

The value is identical to what VIDIOC_QUERY_DV_TIMINGS returns in the hbackporch field of the v4l2_bt_timings struct.

hfront_porch (R):
Number of PCLK pulses between the end of the last valid pixel in the video line (marked by DE=1) and assertion of the HSYNC signal.

The value is identical to what VIDIOC_QUERY_DV_TIMINGS returns in the hfrontporch field of the v4l2_bt_timings struct.

vback_porch (R):
Number of video lines between deassertion of the VSYNC signal and the video line with the first valid pixel (marked by DE=1).

The value is identical to what VIDIOC_QUERY_DV_TIMINGS returns in the vbackporch field of the v4l2_bt_timings struct.

vfront_porch (R):
Number of video lines between the end of the last valid pixel line (marked by DE=1) and assertion of the VSYNC signal.

The value is identical to what VIDIOC_QUERY_DV_TIMINGS returns in the vfrontporch field of the v4l2_bt_timings struct.

frequency_range (RW)
PLL frequency range of the OLDI input clock generator. The PLL frequency is derived from the Pixel Clock Frequency (PCLK) and is equal to PCLK if oldi_lane_width is set to ”single” and PCLK/2 if oldi_lane_width is set to ”dual”.

0 - PLL < 50MHz (default)
1 - PLL >= 50MHz

Note: This parameter can not be changed while the input v4l2 device is open.

Common FPDL3/GMSL output parameters

output_id (R):
Output number ID, zero based.

video_source (RW):
Output video source. If set to 0 or 1, the source is the corresponding card input and the v4l2 output devices are disabled. If set to 2 or 3, the source is the corresponding v4l2 video output device. The default is the corresponding v4l2 output, i.e. 2 for OUT1 and 3 for OUT2.

0 - input 0
1 - input 1
2 - v4l2 output 0
3 - v4l2 output 1

Note: This parameter can not be changed while ANY of the input/output v4l2 devices is open.

display_width (RW):
Display width. There is no autodetection of the connected display, so the proper value must be set before the start of streaming. The default width is 1280.

Note: This parameter can not be changed while the output v4l2 device is open.

display_height (RW):
Display height. There is no autodetection of the connected display, so the proper value must be set before the start of streaming. The default height is 640.

Note: This parameter can not be changed while the output v4l2 device is open.

frame_rate (RW):
Output video frame rate in frames per second. The default frame rate is 60Hz.

hsync_polarity (RW):
HSYNC signal polarity.

0 - active low (default)
1 - active high

vsync_polarity (RW):
VSYNC signal polarity.

0 - active low (default)
1 - active high

de_polarity (RW):
DE signal polarity.

0 - active low
1 - active high (default)

pclk_frequency (RW):
Output pixel clock frequency. Allowed values are between 25000-190000(kHz) and there is a non-linear stepping between two consecutive allowed frequencies. The driver finds the nearest allowed frequency to the given value and sets it. When reading this property, you get the exact frequency set by the driver. The default frequency is 70000kHz.

Note: This parameter can not be changed while the output v4l2 device is open.

hsync_width (RW):
Width of the HSYNC signal in pixels. The default value is 16.

vsync_width (RW):
Width of the VSYNC signal in video lines. The default value is 2.

hback_porch (RW):
Number of PCLK pulses between deassertion of the HSYNC signal and the first valid pixel in the video line (marked by DE=1). The default value is 32.

hfront_porch (RW):
Number of PCLK pulses between the end of the last valid pixel in the video line (marked by DE=1) and assertion of the HSYNC signal. The default value is 32.

vback_porch (RW):
Number of video lines between deassertion of the VSYNC signal and the video line with the first valid pixel (marked by DE=1). The default value is 2.

vfront_porch (RW):
Number of video lines between the end of the last valid pixel line (marked by DE=1) and assertion of the VSYNC signal. The default value is 2.

FPDL3 specific input parameters

fpdl3_input_width (RW):
Number of deserializer input lines.

0 - auto (default)
1 - single
2 - dual

FPDL3 specific output parameters

fpdl3_output_width (RW):
Number of serializer output lines.

0 - auto (default)
1 - single
2 - dual

GMSL specific input parameters

gmsl_mode (RW):
GMSL speed mode.

0 - 12Gb/s (default)
1 - 6Gb/s
2 - 3Gb/s
3 - 1.5Gb/s

gmsl_stream_id (RW):
The GMSL multi-stream contains up to four video streams. This parameter selects which stream is captured by the video input. The value is the zero-based index of the stream. The default stream id is 0.

Note: This parameter can not be changed while the input v4l2 device is open.

gmsl_fec (RW):
GMSL Forward Error Correction (FEC).

0 - disabled
1 - enabled (default)

59.7.10 mgb4 mtd partitions

The mgb4 driver creates a MTD device with two partitions:

- mgb4-fw.X - FPGA firmware.
- mgb4-data.X - Factory settings, e.g. card serial number.

The *mgb4-fw* partition is writable and is used for FW updates, *mgb4-data* is read-only. The *X* attached to the partition name represents the card number. Depending on the CONFIG_MTD_PARTITIONED_MASTER kernel configuration, you may also have a third partition named *mgb4-flash* available in the system. This partition represents the whole, unpartitioned, card's FLASH memory and one should not fiddle with it...

59.7.11 mgb4 iio (triggers)

The mgb4 driver creates an Industrial I/O (IIO) device that provides trigger and signal level status capability. The following scan elements are available:

activity:
The trigger levels and pending status.

bit 1 - trigger 1 pending
bit 2 - trigger 2 pending

bit 5 - trigger 1 level
bit 6 - trigger 2 level

timestamp:
The trigger event timestamp.

The iio device can operate either in "raw" mode where you can fetch the signal levels (activity bits 5 and 6) using sysfs access or in triggered buffer mode. In the triggered buffer mode you can follow the signal level changes (activity bits 1 and 2) using the iio device in /dev. If you enable the timestamps, you will also get the exact trigger event time that can be matched to a video frame (every mgb4 video frame has a timestamp with the same clock source).

Note: although the activity sample always contains all the status bits, it makes no sense to get the pending bits in raw mode or the level bits in the triggered buffer mode - the values do not represent valid data in such case.

59.7.12 OMAP 3 Image Signal Processor (ISP) driver

Copyright © 2010 Nokia Corporation

Copyright © 2009 Texas Instruments, Inc.

Contacts: Laurent Pinchart <laurent.pinchart@ideasonboard.com>, Sakari Ailus <sakari.ailus@iki.fi>, David Cohen <dacohen@gmail.com>

Introduction

This file documents the Texas Instruments OMAP 3 Image Signal Processor (ISP) driver located under drivers/media/platform/ti/omap3isp. The original driver was written by Texas Instruments but since that it has been rewritten (twice) at Nokia.

The driver has been successfully used on the following versions of OMAP 3:

- 3430
- 3530
- 3630

The driver implements V4L2, Media controller and v4l2_subdev interfaces. Sensor, lens and flash drivers using the v4l2_subdev interface in the kernel are supported.

Split to subdevs

The OMAP 3 ISP is split into V4L2 subdevs, each of the blocks inside the ISP having one subdev to represent it. Each of the subdevs provide a V4L2 subdev interface to userspace.

- OMAP3 ISP CCP2
- OMAP3 ISP CSI2a
- OMAP3 ISP CCDC
- OMAP3 ISP preview

- OMAP3 ISP resizer
- OMAP3 ISP AEWB
- OMAP3 ISP AF
- OMAP3 ISP histogram

Each possible link in the ISP is modelled by a link in the Media controller interface. For an example program see[1].

Controlling the OMAP 3 ISP

In general, the settings given to the OMAP 3 ISP take effect at the beginning of the following frame. This is done when the module becomes idle during the vertical blanking period on the sensor. In memory-to-memory operation the pipe is run one frame at a time. Applying the settings is done between the frames.

All the blocks in the ISP, excluding the CSI-2 and possibly the CCP2 receiver, insist on receiving complete frames. Sensors must thus never send the ISP partial frames.

Autoidle does have issues with some ISP blocks on the 3430, at least. Autoidle is only enabled on 3630 when the omap3isp module parameter autoidle is non-zero.

Technical reference manuals (TRMs) and other documentation

OMAP 3430 TRM: <URL:http://focus.ti.com/pdfs/wtbu/OMAP34xx_ES3.1.x_PUBLIC_TRM_vZM.zip> Referenced 2011-03-05.

OMAP 35xx TRM: <URL:http://www.ti.com/litv/pdf/spruf98o> Referenced 2011-03-05.

OMAP 3630 TRM: <URL:http://focus.ti.com/pdfs/wtbu/OMAP36xx_ES1.x_PUBLIC_TRM_vQ.zip> Referenced 2011-03-05.

DM 3730 TRM: <URL:http://www.ti.com/litv/pdf/sprugn4h> Referenced 2011-03-06.

References

59.7.13 OMAP4 ISS Driver

Author: Sergio Aguirre <sergio.a.aguirre@gmail.com>

Copyright (C) 2012, Texas Instruments

[1] http://git.ideasonboard.org/?p=media-ctl.git;a=summary

Introduction

The OMAP44XX family of chips contains the Imaging SubSystem (a.k.a. ISS), Which contains several components that can be categorized in 3 big groups:

- Interfaces (2 Interfaces: CSI2-A & CSI2-B/CCP2)
- ISP (Image Signal Processor)
- SIMCOP (Still Image Coprocessor)

For more information, please look in[1] for latest version of: "OMAP4430 Multimedia Device Silicon Revision 2.x"

As of Revision AB, the ISS is described in detail in section 8.

This driver is supporting **only** the CSI2-A/B interfaces for now.

It makes use of the Media Controller framework[2], and inherited most of the code from OMAP3 ISP driver (found under drivers/media/platform/ti/omap3isp/*), except that it doesn't need an IOMMU now for ISS buffers memory mapping.

Supports usage of MMAP buffers only (for now).

Tested platforms

- OMAP4430SDP, w/ ES2.1 GP & SEVM4430-CAM-V1-0 (Contains IMX060 & OV5640, in which only the last one is supported, outputting YUV422 frames).
- TI Blaze MDP, w/ OMAP4430 ES2.2 EMU (Contains 1 IMX060 & 2 OV5650 sensors, in which only the OV5650 are supported, outputting RAW10 frames).
- PandaBoard, Rev. A2, w/ OMAP4430 ES2.1 GP & OV adapter board, tested with following sensors: * OV5640 * OV5650
- Tested on mainline kernel:

  http://git.kernel.org/?p=linux/kernel/git/torvalds/linux.git;a=summary

  Tag: v3.3 (commit c16fa4f2ad19908a47c63d8fa436a1178438c7e7)

File list

drivers/staging/media/omap4iss/ include/linux/platform_data/media/omap4iss.h

[1] http://focus.ti.com/general/docs/wtbu/wtbudocumentcenter.tsp?navigationId=12037&templateId=6123#62
[2] http://lwn.net/Articles/420485/

References

59.7.14 Philips webcams (pwc driver)

This file contains some additional information for the Philips and OEM webcams. E-mail: webcam@smcc.demon.nl Last updated: 2004-01-19 Site: http://www.smcc.demon.nl/webcam/

As of this moment, the following cameras are supported:

- Philips PCA645
- Philips PCA646
- Philips PCVC675
- Philips PCVC680
- Philips PCVC690
- Philips PCVC720/40
- Philips PCVC730
- Philips PCVC740
- Philips PCVC750
- Askey VC010
- Creative Labs Webcam 5
- Creative Labs Webcam Pro Ex
- Logitech QuickCam 3000 Pro
- Logitech QuickCam 4000 Pro
- Logitech QuickCam Notebook Pro
- Logitech QuickCam Zoom
- Logitech QuickCam Orbit
- Logitech QuickCam Sphere
- Samsung MPC-C10
- Samsung MPC-C30
- Sotec Afina Eye
- AME CU-001
- Visionite VCS-UM100
- Visionite VCS-UC300

The main webpage for the Philips driver is at the address above. It contains a lot of extra information, a FAQ, and the binary plugin 'PWCX'. This plugin contains decompression routines that allow you to use higher image sizes and framerates; in addition the webcam uses less bandwidth on the USB bus (handy if you want to run more than 1 camera simultaneously). These routines fall under a NDA, and may therefore not be distributed as source; however, its use is completely optional.

You can build this code either into your kernel, or as a module. I recommend the latter, since it makes troubleshooting a lot easier. The built-in microphone is supported through the USB Audio class.

When you load the module you can set some default settings for the camera; some programs depend on a particular image-size or -format and don't know how to set it properly in the driver. The options are:

size
Can be one of 'sqcif', 'qsif', 'qcif', 'sif', 'cif' or 'vga', for an image size of resp. 128x96, 160x120, 176x144, 320x240, 352x288 and 640x480 (of course, only for those cameras that support these resolutions).

fps
Specifies the desired framerate. Is an integer in the range of 4-30.

fbufs
This parameter specifies the number of internal buffers to use for storing frames from the cam. This will help if the process that reads images from the cam is a bit slow or momentarily busy. However, on slow machines it only introduces lag, so choose carefully. The default is 3, which is reasonable. You can set it between 2 and 5.

mbufs
This is an integer between 1 and 10. It will tell the module the number of buffers to reserve for mmap(), VIDIOCCGMBUF, VIDIOCMCAPTURE and friends. The default is 2, which is adequate for most applications (double buffering).

Should you experience a lot of 'Dumping frame...' messages during grabbing with a tool that uses mmap(), you might want to increase if. However, it doesn't really buffer images, it just gives you a bit more slack when your program is behind. But you need a multi-threaded or forked program to really take advantage of these buffers.

The absolute maximum is 10, but don't set it too high! Every buffer takes up 460 KB of RAM, so unless you have a lot of memory setting this to something more than 4 is an absolute waste. This memory is only allocated during open(), so nothing is wasted when the camera is not in use.

power_save
When power_save is enabled (set to 1), the module will try to shut down the cam on close() and re-activate on open(). This will save power and turn off the LED. Not all cameras support this though (the 645 and 646 don't have power saving at all), and some models don't work either (they will shut down, but never wake up). Consider this experimental. By default this option is disabled.

compression (only useful with the plugin)
With this option you can control the compression factor that the camera uses to squeeze the image through the USB bus. You can set the parameter between 0 and 3:

```
0 = prefer uncompressed images; if the requested mode is not available
    in an uncompressed format, the driver will silently switch to low
    compression.
1 = low compression.
2 = medium compression.
3 = high compression.
```

High compression takes less bandwidth of course, but it could also introduce some unwanted artefacts. The default is 2, medium compression. See the FAQ on the website for an overview of which modes require compression.

The compression parameter does not apply to the 645 and 646 cameras and OEM models derived from those (only a few). Most cams honour this parameter.

leds

This settings takes 2 integers, that define the on/off time for the LED (in milliseconds). One of the interesting things that you can do with this is let the LED blink while the camera is in use. This:

```
leds=500,500
```

will blink the LED once every second. But with:

```
leds=0,0
```

the LED never goes on, making it suitable for silent surveillance.

By default the camera's LED is on solid while in use, and turned off when the camera is not used anymore.

This parameter works only with the ToUCam range of cameras (720, 730, 740, 750) and OEMs. For other cameras this command is silently ignored, and the LED cannot be controlled.

Finally: this parameters does not take effect UNTIL the first time you open the camera device. Until then, the LED remains on.

dev_hint

A long standing problem with USB devices is their dynamic nature: you never know what device a camera gets assigned; it depends on module load order, the hub configuration, the order in which devices are plugged in, and the phase of the moon (i.e. it can be random). With this option you can give the driver a hint as to what video device node (/dev/videoX) it should use with a specific camera. This is also handy if you have two cameras of the same model.

A camera is specified by its type (the number from the camera model, like PCA645, PCVC750VC, etc) and optionally the serial number (visible in /sys/kernel/debug/usb/devices). A hint consists of a string with the following format:

```
[type[.serialnumber]:]node
```

The square brackets mean that both the type and the serialnumber are optional, but a serialnumber cannot be specified without a type (which would be rather pointless). The serialnumber is separated from the type by a '.'; the node number by a ':'.

This somewhat cryptic syntax is best explained by a few examples:

```
dev_hint=3,5              The first detected cam gets assigned
                          /dev/video3, the second /dev/video5. Any
                          other cameras will get the first free
                          available slot (see below).

dev_hint=645:1,680:2      The PCA645 camera will get /dev/video1,
```

```
                            and a PCVC680 /dev/video2.

dev_hint=645.0123:3,645.4567:0     The PCA645 camera with serialnumber
                                   0123 goes to /dev/video3, the same
                                   camera model with the 4567 serial
                                   gets /dev/video0.

dev_hint=750:1,4,5,6       The PCVC750 camera will get /dev/video1, the
                           next 3 Philips cams will use /dev/video4
                           through /dev/video6.
```

Some points worth knowing:

- Serialnumbers are case sensitive and must be written full, including leading zeroes (it's treated as a string).
- If a device node is already occupied, registration will fail and the webcam is not available.
- You can have up to 64 video devices; be sure to make enough device nodes in /dev if you want to spread the numbers. After /dev/video9 comes /dev/video10 (not /dev/videoA).
- If a camera does not match any dev_hint, it will simply get assigned the first available device node, just as it used to be.

trace

In order to better detect problems, it is now possible to turn on a 'trace' of some of the calls the module makes; it logs all items in your kernel log at debug level.

The trace variable is a bitmask; each bit represents a certain feature. If you want to trace something, look up the bit value(s) in the table below, add the values together and supply that to the trace variable.

Value (dec)	Value (hex)	Description	Default
1	0x1	Module initialization; this will log messages while loading and unloading the module	On
2	0x2	probe() and disconnect() traces	On
4	0x4	Trace open() and close() calls	Off
8	0x8	read(), mmap() and associated ioctl() calls	Off
16	0x10	Memory allocation of buffers, etc.	Off
32	0x20	Showing underflow, overflow and Dumping frame messages	On
64	0x40	Show viewport and image sizes	Off
128	0x80	PWCX debugging	Off

For example, to trace the open() & read() functions, sum 8 + 4 = 12, so you would supply trace=12 during insmod or modprobe. If you want to turn the initialization and probing tracing off, set trace=0. The default value for trace is 35 (0x23).

Example:

```
# modprobe pwc size=cif fps=15 power_save=1
```

The fbufs, mbufs and trace parameters are global and apply to all connected cameras. Each camera has its own set of buffers.

size and fps only specify defaults when you open() the device; this is to accommodate some tools that don't set the size. You can change these settings after open() with the Video4Linux ioctl() calls. The default of defaults is QCIF size at 10 fps.

The compression parameter is semiglobal; it sets the initial compression preference for all camera's, but this parameter can be set per camera with the VIDIOCPWCSCQUAL ioctl() call.

All parameters are optional.

59.7.15 Qualcomm Camera Subsystem driver

Introduction

This file documents the Qualcomm Camera Subsystem driver located under drivers/media/platform/qcom/camss.

The current version of the driver supports the Camera Subsystem found on Qualcomm MSM8916/APQ8016 and MSM8996/APQ8096 processors.

The driver implements V4L2, Media controller and V4L2 subdev interfaces. Camera sensor using V4L2 subdev interface in the kernel is supported.

The driver is implemented using as a reference the Qualcomm Camera Subsystem driver for Android as found in Code Linaro[1][2].

Qualcomm Camera Subsystem hardware

The Camera Subsystem hardware found on 8x16 / 8x96 processors and supported by the driver consists of:

- 2 / 3 CSIPHY modules. They handle the Physical layer of the CSI2 receivers. A separate camera sensor can be connected to each of the CSIPHY module;
- 2 / 4 CSID (CSI Decoder) modules. They handle the Protocol and Application layer of the CSI2 receivers. A CSID can decode data stream from any of the CSIPHY. Each CSID also contains a TG (Test Generator) block which can generate artificial input data for test purposes;
- ISPIF (ISP Interface) module. Handles the routing of the data streams from the CSIDs to the inputs of the VFE;
- 1 / 2 VFE (Video Front End) module(s). Contain a pipeline of image processing hardware blocks. The VFE has different input interfaces. The PIX (Pixel) input interface feeds the input data to the image processing pipeline. The image processing pipeline contains also a scale and crop module at the end. Three RDI (Raw Dump Interface) input interfaces bypass the image processing pipeline. The VFE also contains the AXI bus interface which writes the output data to memory.

[1] https://git.codelinaro.org/clo/la/kernel/msm-3.10/
[2] https://git.codelinaro.org/clo/la/kernel/msm-3.18/

Supported functionality

The current version of the driver supports:

- Input from camera sensor via CSIPHY;
- Generation of test input data by the TG in CSID;
- RDI interface of VFE
  - Raw dump of the input data to memory.

    Supported formats:

    * YUYV/UYVY/YVYU/VYUY (packed YUV 4:2:2 - V4L2_PIX_FMT_YUYV / V4L2_PIX_FMT_UYVY / V4L2_PIX_FMT_YVYU / V4L2_PIX_FMT_VYUY);
    * MIPI RAW8 (8bit Bayer RAW - V4L2_PIX_FMT_SRGGB8 / V4L2_PIX_FMT_SGRBG8 / V4L2_PIX_FMT_SGBRG8 / V4L2_PIX_FMT_SBGGR8);
    * MIPI RAW10 (10bit packed Bayer RAW - V4L2_PIX_FMT_SBGGR10P / V4L2_PIX_FMT_SGBRG10P / V4L2_PIX_FMT_SGRBG10P / V4L2_PIX_FMT_SRGGB10P / V4L2_PIX_FMT_Y10P);
    * MIPI RAW12 (12bit packed Bayer RAW - V4L2_PIX_FMT_SRGGB12P / V4L2_PIX_FMT_SGBRG12P / V4L2_PIX_FMT_SGRBG12P / V4L2_PIX_FMT_SRGGB12P).
    * (8x96 only) MIPI RAW14 (14bit packed Bayer RAW - V4L2_PIX_FMT_SRGGB14P / V4L2_PIX_FMT_SGBRG14P / V4L2_PIX_FMT_SGRBG14P / V4L2_PIX_FMT_SRGGB14P).
  - (8x96 only) Format conversion of the input data.

    Supported input formats:

    * MIPI RAW10 (10bit packed Bayer RAW - V4L2_PIX_FMT_SBGGR10P / V4L2_PIX_FMT_Y10P).

    Supported output formats:

    * Plain16 RAW10 (10bit unpacked Bayer RAW - V4L2_PIX_FMT_SBGGR10 / V4L2_PIX_FMT_Y10).
- PIX interface of VFE
  - Format conversion of the input data.

    Supported input formats:

    * YUYV/UYVY/YVYU/VYUY (packed YUV 4:2:2 - V4L2_PIX_FMT_YUYV / V4L2_PIX_FMT_UYVY / V4L2_PIX_FMT_YVYU / V4L2_PIX_FMT_VYUY).

    Supported output formats:

    * NV12/NV21 (two plane YUV 4:2:0 - V4L2_PIX_FMT_NV12 / V4L2_PIX_FMT_NV21);
    * NV16/NV61 (two plane YUV 4:2:2 - V4L2_PIX_FMT_NV16 / V4L2_PIX_FMT_NV61).
    * (8x96 only) YUYV/UYVY/YVYU/VYUY (packed YUV 4:2:2 - V4L2_PIX_FMT_YUYV / V4L2_PIX_FMT_UYVY / V4L2_PIX_FMT_YVYU / V4L2_PIX_FMT_VYUY).

  - Scaling support. Configuration of the VFE Encoder Scale module for downscalling with ratio up to 16x.
  - Cropping support. Configuration of the VFE Encoder Crop module.
- Concurrent and independent usage of two (8x96: three) data inputs - could be camera sensors and/or TG.

Driver Architecture and Design

The driver implements the V4L2 subdev interface. With the goal to model the hardware links between the modules and to expose a clean, logical and usable interface, the driver is split into V4L2 sub-devices as follows (8x16 / 8x96):

- 2 / 3 CSIPHY sub-devices - each CSIPHY is represented by a single sub-device;
- 2 / 4 CSID sub-devices - each CSID is represented by a single sub-device;
- 2 / 4 ISPIF sub-devices - ISPIF is represented by a number of sub-devices equal to the number of CSID sub-devices;
- 4 / 8 VFE sub-devices - VFE is represented by a number of sub-devices equal to the number of the input interfaces (3 RDI and 1 PIX for each VFE).

The considerations to split the driver in this particular way are as follows:

- representing CSIPHY and CSID modules by a separate sub-device for each module allows to model the hardware links between these modules;
- representing VFE by a separate sub-devices for each input interface allows to use the input interfaces concurrently and independently as this is supported by the hardware;
- representing ISPIF by a number of sub-devices equal to the number of CSID sub-devices allows to create linear media controller pipelines when using two cameras simultaneously. This avoids branches in the pipelines which otherwise will require a) userspace and b) media framework (e.g. power on/off operations) to make assumptions about the data flow from a sink pad to a source pad on a single media entity.

Each VFE sub-device is linked to a separate video device node.

The media controller pipeline graph is as follows (with connected two / three OV5645 camera sensors):

Implementation

Runtime configuration of the hardware (updating settings while streaming) is not required to implement the currently supported functionality. The complete configuration on each hardware module is applied on STREAMON ioctl based on the current active media links, formats and controls set.

The output size of the scaler module in the VFE is configured with the actual compose selection rectangle on the sink pad of the 'msm_vfe0_pix' entity.

The crop output area of the crop module in the VFE is configured with the actual crop selection rectangle on the source pad of the 'msm_vfe0_pix' entity.

Fig. 5: Media pipeline graph 8x16

Fig. 6: Media pipeline graph 8x96

Documentation

APQ8016 Specification: https://developer.qualcomm.com/download/sd410/snapdragon-410-processor-device-specification.pdf Referenced 2016-11-24.

APQ8096 Specification: https://developer.qualcomm.com/download/sd820e/qualcomm-snapdragon-820e-processor-apq8096sge-device-specification.pdf Referenced 2018-06-22.

References

59.7.16 Renesas R-Car Fine Display Processor (FDP1) Driver

The R-Car FDP1 driver implements driver-specific controls as follows.

`V4L2_CID_DEINTERLACING_MODE (menu)`
The video deinterlacing mode (such as Bob, Weave, ...). The R-Car FDP1 driver implements the following modes.

"Progressive" (0)	The input image video stream is progressive (not interlaced). No deinterlacing is performed. Apart from (optional) format and encoding conversion output frames are identical to the input frames.
"Adaptive 2D/3D" (1)	Motion adaptive version of 2D and 3D deinterlacing. Use 3D deinterlacing in the presence of fast motion and 2D deinterlacing with diagonal interpolation otherwise.
"Fixed 2D" (2)	The current field is scaled vertically by averaging adjacent lines to recover missing lines. This method is also known as blending or Line Averaging (LAV).
"Fixed 3D" (3)	The previous and next fields are averaged to recover lines missing from the current field. This method is also known as Field Averaging (FAV).
"Previous field" (4)	The current field is weaved with the previous field, i.e. the previous field is used to fill missing lines from the current field. This method is also known as weave deinterlacing.
"Next field" (5)	The current field is weaved with the next field, i.e. the next field is used to fill missing lines from the current field. This method is also known as weave deinterlacing.

59.7.17 Rockchip Image Signal Processor (rkisp1)

Introduction

This file documents the driver for the Rockchip ISP1 that is part of RK3288 and RK3399 SoCs. The driver is located under drivers/media/platform/rockchip/ rkisp1 and uses the Media-Controller API.

Revisions

There exist multiple smaller revisions to this ISP that got introduced in later SoCs. Revisions can be found in the enum `rkisp1_cif_isp_version` in the UAPI and the revision of the ISP inside the running SoC can be read in the field hw_revision of struct media_device_info as returned by ioctl MEDIA_IOC_DEVICE_INFO.

Versions in use are:

- RKISP1_V10: used at least in rk3288 and rk3399
- RKISP1_V11: declared in the original vendor code, but not used
- RKISP1_V12: used at least in rk3326 and px30
- RKISP1_V13: used at least in rk1808

Topology

The driver has 4 video devices:

- rkisp1_mainpath: capture device for retrieving images, usually in higher resolution.
- rkisp1_selfpath: capture device for retrieving images.
- rkisp1_stats: a metadata capture device that sends statistics.
- rkisp1_params: a metadata output device that receives parameters configurations from userspace.

The driver has 3 subdevices:

- rkisp1_resizer_mainpath: used to resize and downsample frames for the mainpath capture device.
- rkisp1_resizer_selfpath: used to resize and downsample frames for the selfpath capture device.
- rkisp1_isp: is connected to the sensor and is responsible for all the isp operations.

rkisp1_mainpath, rkisp1_selfpath - Frames Capture Video Nodes

Those are the *mainpath* and *selfpath* capture devices to capture frames. Those entities are the DMA engines that write the frames to memory. The selfpath video device can capture YUV/RGB formats. Its input is YUV encoded stream and it is able to convert it to RGB. The selfpath is not able to capture bayer formats. The mainpath can capture both bayer and YUV formats but it is not able to capture RGB formats. Both capture videos support the `V4L2_CAP_IO_MC` capability.

rkisp1_resizer_mainpath, rkisp1_resizer_selfpath - Resizers Subdevices Nodes

Those are resizer entities for the mainpath and the selfpath. Those entities can scale the frames up and down and also change the YUV sampling (for example YUV4:2:2 -> YUV4:2:0). They also have cropping capability on the sink pad. The resizers entities can only operate on YUV:4:2:2 format (MEDIA_BUS_FMT_YUYV8_2X8). The mainpath capture device supports capturing video in bayer formats. In that case the resizer of the mainpath is set to 'bypass' mode - it just forward the frame without operating on it.

rkisp1_isp - Image Signal Processing Subdevice Node

This is the isp entity. It is connected to the sensor on sink pad 0 and receives the frames using the CSI-2 protocol. It is responsible of configuring the CSI-2 protocol. It has a cropping capability on sink pad 0 that is connected to the sensor and on source pad 2 connected to the resizer entities. Cropping on sink pad 0 defines the image region from the sensor. Cropping on source pad 2 defines the region for the Image Stabilizer (IS).

rkisp1_stats - Statistics Video Node

The statistics video node outputs the 3A (auto focus, auto exposure and auto white balance) statistics, and also histogram statistics for the frames that are being processed by the rkisp1 to userspace applications. Using these data, applications can implement algorithms and re-parameterize the driver through the rkisp_params node to improve image quality during a video stream. The buffer format is defined by struct `rkisp1_stat_buffer`, and userspace should set V4L2_META_FMT_RK_ISP1_STAT_3A as the dataformat.

rkisp1_params - Parameters Video Node

The rkisp1_params video node receives a set of parameters from userspace to be applied to the hardware during a video stream, allowing userspace to dynamically modify values such as black level, cross talk corrections and others.

The buffer format is defined by struct `rkisp1_params_cfg`, and userspace should set V4L2_META_FMT_RK_ISP1_PARAMS as the dataformat.

Capturing Video Frames Example

In the following example, the sensor connected to pad 0 of 'rkisp1_isp' is imx219.

The following commands can be used to capture video from the selfpath video node with dimension 900x800 planar format YUV 4:2:2. It uses all cropping capabilities possible, (see explanation right below)

```
# set the links
"media-ctl" "-d" "platform:rkisp1" "-r"
"media-ctl" "-d" "platform:rkisp1" "-l" "'imx219 4-0010':0 -> 'rkisp1_isp':0␣
↪[1]"
"media-ctl" "-d" "platform:rkisp1" "-l" "'rkisp1_isp':2 -> 'rkisp1_resizer_
↪selfpath':0 [1]"
"media-ctl" "-d" "platform:rkisp1" "-l" "'rkisp1_isp':2 -> 'rkisp1_resizer_
↪mainpath':0 [0]"

# set format for imx219 4-0010:0
"media-ctl" "-d" "platform:rkisp1" "--set-v4l2" '"imx219 4-0010":0␣
↪[fmt:SRGGB10_1X10/1640x1232]'

# set format for rkisp1_isp pads:
"media-ctl" "-d" "platform:rkisp1" "--set-v4l2" '"rkisp1_isp":0 [fmt:SRGGB10_
↪1X10/1640x1232 crop: (0,0)/1600x1200]'
"media-ctl" "-d" "platform:rkisp1" "--set-v4l2" '"rkisp1_isp":2 [fmt:YUYV8_2X8/
↪1600x1200 crop: (0,0)/1500x1100]'

# set format for rkisp1_resizer_selfpath pads:
"media-ctl" "-d" "platform:rkisp1" "--set-v4l2" '"rkisp1_resizer_selfpath":0␣
↪[fmt:YUYV8_2X8/1500x1100 crop: (300,400)/1400x1000]'
"media-ctl" "-d" "platform:rkisp1" "--set-v4l2" '"rkisp1_resizer_selfpath":1␣
↪[fmt:YUYV8_2X8/900x800]'

# set format for rkisp1_selfpath:
"v4l2-ctl" "-z" "platform:rkisp1" "-d" "rkisp1_selfpath" "-v" "width=900,
↪height=800,"
"v4l2-ctl" "-z" "platform:rkisp1" "-d" "rkisp1_selfpath" "-v" "pixelformat=422P
↪"

# start streaming:
v4l2-ctl "-z" "platform:rkisp1" "-d" "rkisp1_selfpath" "--stream-mmap" "--
↪stream-count" "10"
```

In the above example the sensor is configured to bayer format: *SRGGB10_1X10/1640x1232*. The rkisp1_isp:0 pad should be configured to the same mbus format and dimensions as the sensor, otherwise streaming will fail with 'EPIPE' error. So it is also configured to *SRGGB10_1X10/1640x1232*. In addition, the rkisp1_isp:0 pad is configured to cropping *(0,0)/1600x1200*.

The cropping dimensions are automatically propagated to be the format of the isp source pad *rkisp1_isp:2*. Another cropping operation is configured on the isp source pad: *(0,0)/1500x1100*.

The resizer's sink pad *rkisp1_resizer_selfpath* should be configured to format

YUYV8_2X8/1500x1100 in order to match the format on the other side of the link. In addition a cropping *(300,400)/1400x1000* is configured on it.

The source pad of the resizer, *rkisp1_resizer_selfpath:1* is configured to format *YUYV8_2X8/900x800*. That means that the resizer first crop a window of *(300,400)/1400x100* from the received frame and then scales this window to dimension *900x800*.

Note that the above example does not uses the stats-params control loop. Therefore the capture frames will not go through the 3A algorithms and probably won't have a good quality, and can even look dark and greenish.

Configuring Quantization

The driver supports limited and full range quantization on YUV formats, where limited is the default. To switch between one or the other, userspace should use the Colorspace Conversion API (CSC) for subdevices on source pad 2 of the isp (*rkisp1_isp:2*). The quantization configured on this pad is the quantization of the captured video frames on the main-path and selfpath video nodes. Note that the resizer and capture entities will always report `V4L2_QUANTIZATION_DEFAULT` even if the quantization is configured to full range on *rkisp1_isp:2*. So in order to get the configured quantization, application should get it from pad *rkisp1_isp:2*.

59.7.18 The saa7134 driver

Author Gerd Hoffmann

This is a v4l2/oss device driver for saa7130/33/34/35 based capture / TV boards.

Status

Almost everything is working. video, sound, tuner, radio, mpeg ts, ...

As with bttv, card-specific tweaks are needed. Check CARDLIST for a list of known TV cards and saa7134-cards.c for the drivers card configuration info.

Build

Once you pick up a Kernel source, you should configure, build, install and boot the new kernel. You'll need at least these config options:

```
./scripts/config -e PCI
./scripts/config -e INPUT
./scripts/config -m I2C
./scripts/config -m MEDIA_SUPPORT
./scripts/config -e MEDIA_PCI_SUPPORT
./scripts/config -e MEDIA_ANALOG_TV_SUPPORT
./scripts/config -e MEDIA_DIGITAL_TV_SUPPORT
./scripts/config -e MEDIA_RADIO_SUPPORT
./scripts/config -e RC_CORE
./scripts/config -e MEDIA_SUBDRV_AUTOSELECT
./scripts/config -m VIDEO_SAA7134
```

```
./scripts/config -e SAA7134_ALSA
./scripts/config -e VIDEO_SAA7134_RC
./scripts/config -e VIDEO_SAA7134_DVB
./scripts/config -e VIDEO_SAA7134_GO7007
```

To build and install, you should run:

```
make && make modules_install && make install
```

Once the new Kernel is booted, saa7134 driver should be loaded automatically.

Depending on the card you might have to pass `card=<nr>` as insmod option. If so, please check *SAA7134 cards list* for valid choices.

Once you have your card type number, you can pass a modules configuration via a file (usually, it is either `/etc/modules.conf` or some file at `/etc/modules-load.d/`, but the actual place depends on your distribution), with this content:

```
options saa7134 card=13 # Assuming that your card type is #13
```

Changes / Fixes

Please mail to linux-media AT vger.kernel.org unified diffs against the linux media git tree:

> https://git.linuxtv.org/media_tree.git/

This is done by committing a patch at a clone of the git tree and submitting the patch using `git send-email`. Don't forget to describe at the lots what it changes / which problem it fixes / whatever it is good for ...

Known Problems

- The tuner for the flyvideos isn't detected automatically and the default might not work for you depending on which version you have. There is a `tuner=` insmod option to override the driver's default.

Credits

andrew.stevens@philips.com + werner.leeb@philips.com for providing saa7134 hardware specs and sample board.

59.7.19 The Silicon Labs Si470x FM Radio Receivers driver

Copyright © 2009 Tobias Lorenz <tobias.lorenz@gmx.net>

Information from Silicon Labs

Silicon Laboratories is the manufacturer of the radio ICs, that nowadays are the most often used radio receivers in cell phones. Usually they are connected with I2C. But SiLabs also provides a reference design, which integrates this IC, together with a small microcontroller C8051F321, to form a USB radio. Part of this reference design is also a radio application in binary and source code. The software also contains an automatic firmware upgrade to the most current version. Information on these can be downloaded here: http://www.silabs.com/usbradio

Supported ICs

The following ICs have a very similar register set, so that they are or will be supported somewhen by the driver:

- Si4700: FM radio receiver
- Si4701: FM radio receiver, RDS Support
- Si4702: FM radio receiver
- Si4703: FM radio receiver, RDS Support
- Si4704: FM radio receiver, no external antenna required
- Si4705: FM radio receiver, no external antenna required, RDS support, Dig I/O
- **Si4706: Enhanced FM RDS/TMC radio receiver, no external antenna required, RDS** Support
- Si4707: Dedicated weather band radio receiver with SAME decoder, RDS Support
- Si4708: Smallest FM receivers
- Si4709: Smallest FM receivers, RDS Support

More information on these can be downloaded here: http://www.silabs.com/products/mcu/Pages/USBFMRadioRD.aspx

Supported USB devices

Currently the following USB radios (vendor:product) with the Silicon Labs si470x chips are known to work:

- 10c4:818a: Silicon Labs USB FM Radio Reference Design
- 06e1:a155: ADS/Tech FM Radio Receiver (formerly Instant FM Music) (RDX-155-EF)
- 1b80:d700: KWorld USB FM Radio SnapMusic Mobile 700 (FM700)
- 10c5:819a: Sanei Electric, Inc. FM USB Radio (sold as DealExtreme.com PCear)

Software

Testing is usually done with most application under Debian/testing:

- fmtools - Utility for managing FM tuner cards
- gnomeradio - FM-radio tuner for the GNOME desktop
- gradio - GTK FM radio tuner
- kradio - Comfortable Radio Application for KDE
- radio - ncurses-based radio application
- mplayer - The Ultimate Movie Player For Linux
- v4l2-ctl - Collection of command line video4linux utilities

For example, you can use:

```
v4l2-ctl -d /dev/radio0 --set-ctrl=volume=10,mute=0 --set-freq=95.21 --all
```

There is also a library libv4l, which can be used. It's going to have a function for frequency seeking, either by using hardware functionality as in radio-si470x or by implementing a function as we currently have in every of the mentioned programs. Somewhen the radio programs should make use of libv4l.

For processing RDS information, there is a project ongoing at: http://rdsd.berlios.de/

There is currently no project for making TMC sentences human readable.

Audio Listing

USB Audio is provided by the ALSA snd_usb_audio module. It is recommended to also select SND_USB_AUDIO, as this is required to get sound from the radio. For listing you have to redirect the sound, for example using one of the following commands. Please adjust the audio devices to your needs (/dev/dsp* and hw:x,x).

If you just want to test audio (very poor quality):

```
cat /dev/dsp1 > /dev/dsp
```

If you use sox + OSS try:

```
sox -2 --endian little -r 96000 -t oss /dev/dsp1 -t oss /dev/dsp
```

or using sox + alsa:

```
sox --endian little -c 2 -S -r 96000 -t alsa hw:1 -t alsa -r 96000 hw:0
```

If you use arts try:

```
arecord -D hw:1,0 -r96000 -c2 -f S16_LE | artsdsp aplay -B -
```

If you use mplayer try:

```
mplayer -radio adevice=hw=1.0:arate=96000 \
        -rawaudio rate=96000 \
        radio://<frequency>/capture
```

Module Parameters

After loading the module, you still have access to some of them in the sysfs mount under /sys/module/radio_si470x/parameters. The contents of read-only files (0444) are not updated, even if space, band and de are changed using private video controls. The others are runtime changeable.

Errors

Increase tune_timeout, if you often get -EIO errors.

When timed out or band limit is reached, hw_freq_seek returns -EAGAIN.

If you get any errors from snd_usb_audio, please report them to the ALSA people.

Open Issues

V4L minor device allocation and parameter setting is not perfect. A solution is currently under discussion.

There is an USB interface for downloading/uploading new firmware images. Support for it can be implemented using the request_firmware interface.

There is a RDS interrupt mode. The driver is already using the same interface for polling RDS information, but is currently not using the interrupt mode.

There is a LED interface, which can be used to override the LED control programmed in the firmware. This can be made available using the LED support functions in the kernel.

Other useful information and links

http://www.silabs.com/usbradio

59.7.20 The Silicon Labs Si4713 FM Radio Transmitter Driver

Copyright © 2009 Nokia Corporation

Contact: Eduardo Valentin <eduardo.valentin@nokia.com>

Information about the Device

This chip is a Silicon Labs product. It is a I2C device, currently on 0x63 address. Basically, it has transmission and signal noise level measurement features.

The Si4713 integrates transmit functions for FM broadcast stereo transmission. The chip also allows integrated receive power scanning to identify low signal power FM channels.

The chip is programmed using commands and responses. There are also several properties which can change the behavior of this chip.

Users must comply with local regulations on radio frequency (RF) transmission.

Device driver description

There are two modules to handle this device. One is a I2C device driver and the other is a platform driver.

The I2C device driver exports a v4l2-subdev interface to the kernel. All properties can also be accessed by v4l2 extended controls interface, by using the v4l2-subdev calls (g_ext_ctrls, s_ext_ctrls).

The platform device driver exports a v4l2 radio device interface to user land. So, it uses the I2C device driver as a sub device in order to send the user commands to the actual device. Basically it is a wrapper to the I2C device driver.

Applications can use v4l2 radio API to specify frequency of operation, mute state, etc. But mostly of its properties will be present in the extended controls.

When the v4l2 mute property is set to 1 (true), the driver will turn the chip off.

Properties description

The properties can be accessed using v4l2 extended controls. Here is an output from v4l2-ctl util:

```
/ # v4l2-ctl -d /dev/radio0 --all -L
Driver Info:
        Driver name   : radio-si4713
        Card type     : Silicon Labs Si4713 Modulator
        Bus info      :
        Driver version: 0
        Capabilities  : 0x00080800
                RDS Output
                Modulator
Audio output: 0 (FM Modulator Audio Out)
Frequency: 1408000 (88.000000 MHz)
Video Standard = 0x00000000
Modulator:
        Name                 : FM Modulator
        Capabilities         : 62.5 Hz stereo rds
        Frequency range      : 76.0 MHz - 108.0 MHz
        Subchannel modulation: stereo+rds
```

```
User Controls

                         mute (bool) : default=1 value=0

FM Radio Modulator Controls

        rds_signal_deviation (int)  : min=0 max=90000 step=10 default=200␣
↪value=200 flags=slider
                rds_program_id (int)  : min=0 max=65535 step=1 default=0␣
↪value=0
        rds_program_type (int)  : min=0 max=31 step=1 default=0 value=0
                rds_ps_name (str)  : min=0 max=96 step=8 value='si4713  '
                rds_radio_text (str)  : min=0 max=384 step=32 value=''
audio_limiter_feature_enabled (bool) : default=1 value=1
audio_limiter_release_time (int)  : min=250 max=102390 step=50 default=5010␣
↪value=5010 flags=slider
        audio_limiter_deviation (int)  : min=0 max=90000 step=10 default=66250␣
↪value=66250 flags=slider
audio_compression_feature_enabl (bool) : default=1 value=1
        audio_compression_gain (int)  : min=0 max=20 step=1 default=15␣
↪value=15 flags=slider
audio_compression_threshold (int)  : min=-40 max=0 step=1 default=-40 value=-
↪40 flags=slider
audio_compression_attack_time (int)  : min=0 max=5000 step=500 default=0␣
↪value=0 flags=slider
audio_compression_release_time (int)  : min=100000 max=1000000 step=100000␣
↪default=1000000 value=1000000 flags=slider
pilot_tone_feature_enabled (bool) : default=1 value=1
        pilot_tone_deviation (int)  : min=0 max=90000 step=10 default=6750␣
↪value=6750 flags=slider
        pilot_tone_frequency (int)  : min=0 max=19000 step=1 default=19000␣
↪value=19000 flags=slider
        pre_emphasis_settings (menu) : min=0 max=2 default=1 value=1
        tune_power_level (int)  : min=0 max=120 step=1 default=88 value=88␣
↪flags=slider
        tune_antenna_capacitor (int)  : min=0 max=191 step=1 default=0␣
↪value=110 flags=slider
```

Here is a summary of them:

- Pilot is an audible tone sent by the device.
- pilot_frequency - Configures the frequency of the stereo pilot tone.
- pilot_deviation - Configures pilot tone frequency deviation level.
- pilot_enabled - Enables or disables the pilot tone feature.
- The si4713 device is capable of applying audio compression to the transmitted signal.
- acomp_enabled - Enables or disables the audio dynamic range control feature.
- acomp_gain - Sets the gain for audio dynamic range control.

- acomp_threshold - Sets the threshold level for audio dynamic range control.
- acomp_attack_time - Sets the attack time for audio dynamic range control.
- acomp_release_time - Sets the release time for audio dynamic range control.
- Limiter setups audio deviation limiter feature. Once a over deviation occurs, it is possible to adjust the front-end gain of the audio input and always prevent over deviation.
- limiter_enabled - Enables or disables the limiter feature.
- limiter_deviation - Configures audio frequency deviation level.
- limiter_release_time - Sets the limiter release time.
- Tuning power
- power_level - Sets the output power level for signal transmission. antenna_capacitor - This selects the value of antenna tuning capacitor manually or automatically if set to zero.
- RDS related
- rds_ps_name - Sets the RDS ps name field for transmission.
- rds_radio_text - Sets the RDS radio text for transmission.
- rds_pi - Sets the RDS PI field for transmission.
- rds_pty - Sets the RDS PTY field for transmission.
- Region related
- preemphasis - sets the preemphasis to be applied for transmission.

RNL

This device also has an interface to measure received noise level. To do that, you should ioctl the device node. Here is an code of example:

```
int main (int argc, char *argv[])
{
        struct si4713_rnl rnl;
        int fd = open("/dev/radio0", O_RDWR);
        int rval;

        if (argc < 2)
                return -EINVAL;

        if (fd < 0)
                return fd;

        sscanf(argv[1], "%d", &rnl.frequency);

        rval = ioctl(fd, SI4713_IOC_MEASURE_RNL, &rnl);
        if (rval < 0)
                return rval;

        printf("received noise level: %d\n", rnl.rnl);
```

```
        close(fd);
}
```

The struct si4713_rnl and SI4713_IOC_MEASURE_RNL are defined under include/linux/platform_data/media/si4713.h.

Stereo/Mono and RDS subchannels

The device can also be configured using the available sub channels for transmission. To do that use S/G_MODULATOR ioctl and configure txsubchans properly. Refer to the V4L2 API specification for proper use of this ioctl.

Testing

Testing is usually done with v4l2-ctl utility for managing FM tuner cards. The tool can be found in v4l-dvb repository under v4l2-apps/util directory.

Example for setting rds ps name:

```
# v4l2-ctl -d /dev/radio0 --set-ctrl=rds_ps_name="Dummy"
```

59.7.21 The SI476x Driver

Copyright © 2013 Andrey Smirnov <andrew.smirnov@gmail.com>

TODO for the driver

- According to the SiLabs' datasheet it is possible to update the firmware of the radio chip in the run-time, thus bringing it to the most recent version. Unfortunately I couldn't find any mentioning of the said firmware update for the old chips that I tested the driver against, so for chips like that the driver only exposes the old functionality.

Parameters exposed over debugfs

SI476x allow user to get multiple characteristics that can be very useful for EoL testing/RF performance estimation, parameters that have very little to do with V4L2 subsystem. Such parameters are exposed via debugfs and can be accessed via regular file I/O operations.

The drivers exposes following files:

- /sys/kernel/debug/<device-name>/acf This file contains ACF(Automatically Controlled Features) status information. The contents of the file is binary data of the following layout:

Offset	Name	Description
0x00	blend_int	Flag, set when stereo separation has crossed below the blend threshold
0x01	hblend_int	Flag, set when HiBlend cutoff frequency is lower than threshold
0x02	hicut_int	Flag, set when HiCut cutoff frequency is lower than threshold
0x03	chbw_int	Flag, set when channel filter bandwidth is less than threshold
0x04	soft-mute_int	Flag indicating that softmute attenuation has increased above softmute threshold
0x05	smute	0 - Audio is not soft muted 1 - Audio is soft muted
0x06	smattn	Soft mute attenuation level in dB
0x07	chbw	Channel filter bandwidth in kHz
0x08	hicut	HiCut cutoff frequency in units of 100Hz
0x09	hiblend	HiBlend cutoff frequency in units of 100 Hz
0x10	pilot	0 - Stereo pilot is not present 1 - Stereo pilot is present
0x11	stblend	Stereo blend in %

- /sys/kernel/debug/<device-name>/rds_blckcnt This file contains statistics about RDS receptions. It's binary data has the following layout:

Offset	Name	Description
0x00	expected	Number of expected RDS blocks
0x02	received	Number of received RDS blocks
0x04	uncor-rectable	Number of uncorrectable RDS blocks

- /sys/kernel/debug/<device-name>/agc This file contains information about parameters pertaining to AGC(Automatic Gain Control)

  The layout is:

Offset	Name	Description
0x00	mxhi	0 - FM Mixer PD high threshold is not tripped 1 - FM Mixer PD high threshold is tripped
0x01	mxlo	ditto for FM Mixer PD low
0x02	lnahi	ditto for FM LNA PD high
0x03	lnalo	ditto for FM LNA PD low
0x04	fmagc1	FMAGC1 attenuator resistance (see datasheet for more detail)
0x05	fmagc2	ditto for FMAGC2
0x06	pgagain	PGA gain in dB
0x07	fmwblang	FM/WB LNA Gain in dB

- /sys/kernel/debug/<device-name>/rsq This file contains information about parameters pertaining to RSQ(Received Signal Quality)

  The layout is:

Offset	Name	Description
0x00	multhint	0 - multipath value has not crossed the Multipath high threshold 1 - multipath value has crossed the Multipath high threshold
0x01	multlint	ditto for Multipath low threshold
0x02	snrhint	0 - received signal's SNR has not crossed high threshold 1 - received signal's SNR has crossed high threshold
0x03	snrlint	ditto for low threshold
0x04	rssihint	ditto for RSSI high threshold
0x05	rssilint	ditto for RSSI low threshold
0x06	bltf	Flag indicating if seek command reached/wrapped seek band limit
0x07	snr_ready	Indicates that SNR metrics is ready
0x08	rssiready	ditto for RSSI metrics
0x09	injside	0 - Low-side injection is being used 1 - High-side injection is used
0x10	afcrl	Flag indicating if AFC rails
0x11	valid	Flag indicating if channel is valid
0x12	readfreq	Current tuned frequency
0x14	freqoff	Signed frequency offset in units of 2ppm
0x15	rssi	Signed value of RSSI in dBuV
0x16	snr	Signed RF SNR in dB
0x17	issi	Signed Image Strength Signal indicator
0x18	lassi	Signed Low side adjacent Channel Strength indicator
0x19	hassi	ditto for High side
0x20	mult	Multipath indicator
0x21	dev	Frequency deviation
0x24	assi	Adjacent channel SSI
0x25	usn	Ultrasonic noise indicator
0x26	pilotdev	Pilot deviation in units of 100 Hz
0x27	rdsdev	ditto for RDS
0x28	assidev	ditto for ASSI
0x29	strongdev	Frequency deviation
0x30	rdspi	RDS PI code

- /sys/kernel/debug/<device-name>/rsq_primary This file contains information about parameters pertaining to RSQ(Received Signal Quality) for primary tuner only. Layout is as the one above.

59.7.22 Starfive Camera Subsystem driver

Introduction

This file documents the driver for the Starfive Camera Subsystem found on Starfive JH7110 SoC. The driver is located under drivers/staging/media/starfive/ camss.

The driver implements V4L2, Media controller and v4l2_subdev interfaces. Camera sensor using V4L2 subdev interface in the kernel is supported.

The driver has been successfully used on the Gstreamer 1.18.5 with v4l2src plugin.

Starfive Camera Subsystem hardware

The Starfive Camera Subsystem hardware consists of:

```
                   |\         +---------------+      +-----------+
+----------+       |  \       |               |      |           |
|          |       |   |      |               |      |           |
|   MIPI   |------>|   |----->|      ISP      |----->|           |
|          |       |   |      |               |      |           |
+----------+       |   |      |               |      |  Memory   |
                   |MUX|      +---------------+      | Interface |
+----------+       |   |                             |           |
|          |       |   |---------------------------->|           |
| Parallel |------>|   |                             |           |
|          |       |   |                             |           |
+----------+       |  /                              |           |
                   |/                                +-----------+
```

- MIPI: The MIPI interface, receiving data from a MIPI CSI-2 camera sensor.
- Parallel: The parallel interface, receiving data from a parallel sensor.
- ISP: The ISP, processing raw Bayer data from an image sensor and producing YUV frames.

Topology

The media controller pipeline graph is as follows:

The driver has 2 video devices:

- capture_raw: The capture device, capturing image data directly from a sensor.
- capture_yuv: The capture device, capturing YUV frame data processed by the ISP module

The driver has 3 subdevices:

- stf_isp: is responsible for all the isp operations, outputs YUV frames.
- cdns_csi2rx: a CSI-2 bridge supporting up to 4 CSI lanes in input, and 4 different pixel streams in output.
- imx219: an image sensor, image data is sent through MIPI CSI-2.

59.7.23 The Virtual Media Controller Driver (vimc)

The vimc driver emulates complex video hardware using the V4L2 API and the Media API. It has a capture device and three subdevices: sensor, debayer and scaler.

imx219 6-0010
/dev/v4l-subdev1
0
0
cdns_csi2rx.19800000.csi-bridge
1
2
3
4
0
stf_isp
/dev/v4l-subdev0
1
capture_raw
/dev/video0
capture_yuv
/dev/video1

Topology

The topology is hardcoded, although you could modify it in vimc-core and recompile the driver to achieve your own topology. This is the default topology:

Fig. 7: Media pipeline graph on vimc

Configuring the topology

Each subdevice will come with its default configuration (pixelformat, height, width, ...). One needs to configure the topology in order to match the configuration on each linked subdevice to stream frames through the pipeline. If the configuration doesn't match, the stream will fail. The `v4l-utils` package is a bundle of user-space applications, that comes with `media-ctl` and `v4l2-ctl` that can be used to configure the vimc configuration. This sequence of commands fits for the default topology:

```
media-ctl -d platform:vimc -V '"Sensor A":0[fmt:SBGGR8_1X8/640x480]'
media-ctl -d platform:vimc -V '"Debayer A":0[fmt:SBGGR8_1X8/640x480]'
media-ctl -d platform:vimc -V '"Scaler":0[fmt:RGB888_1X24/640x480]'
media-ctl -d platform:vimc -V '"Scaler":0[crop:(100,50)/400x150]'
media-ctl -d platform:vimc -V '"Scaler":1[fmt:RGB888_1X24/300x700]'
v4l2-ctl -z platform:vimc -d "RGB/YUV Capture" -v width=300,height=700
v4l2-ctl -z platform:vimc -d "Raw Capture 0" -v pixelformat=BA81
```

Subdevices

Subdevices define the behavior of an entity in the topology. Depending on the subdevice, the entity can have multiple pads of type source or sink.

vimc-sensor:

Generates images in several formats using video test pattern generator. Exposes:

- 1 Pad source

vimc-lens:
Ancillary lens for a sensor. Supports auto focus control. Linked to a vimc-sensor using an ancillary link. The lens supports FOCUS_ABSOLUTE control.

```
media-ctl -p
...
- entity 28: Lens A (0 pad, 0 link)
                type V4L2 subdev subtype Lens flags 0
                device node name /dev/v4l-subdev6
- entity 29: Lens B (0 pad, 0 link)
                type V4L2 subdev subtype Lens flags 0
                device node name /dev/v4l-subdev7
v4l2-ctl -d /dev/v4l-subdev7 -C focus_absolute
focus_absolute: 0
```

vimc-debayer:
Transforms images in bayer format into a non-bayer format. Exposes:

- 1 Pad sink
- 1 Pad source

vimc-scaler:
Re-size the image to meet the source pad resolution. E.g.: if the sync pad is configured to 360x480 and the source to 1280x720, the image will be stretched to fit the source resolution. Works for any resolution within the vimc limitations (even shrinking the image if necessary). Exposes:

- 1 Pad sink
- 1 Pad source

vimc-capture:
Exposes node /dev/videoX to allow userspace to capture the stream. Exposes:

- 1 Pad sink
- 1 Pad source

Module options

Vimc has a module parameter to configure the driver.

- `allocator=<unsigned int>`

  memory allocator selection, default is 0. It specifies the way buffers will be allocated.

  - 0: vmalloc
  - 1: dma-contig

59.7.24 The Virtual Stateless Decoder Driver (visl)

A virtual stateless decoder device for stateless uAPI development purposes.

This tool's objective is to help the development and testing of userspace applications that use the V4L2 stateless API to decode media.

A userspace implementation can use visl to run a decoding loop even when no hardware is available or when the kernel uAPI for the codec has not been upstreamed yet. This can reveal bugs at an early stage.

This driver can also trace the contents of the V4L2 controls submitted to it. It can also dump the contents of the vb2 buffers through a debugfs interface. This is in many ways similar to the tracing infrastructure available for other popular encode/decode APIs out there and can help develop a userspace application by using another (working) one as a reference.

Note: No actual decoding of video frames is performed by visl. The V4L2 test pattern generator is used to write various debug information to the capture buffers instead.

Module parameters

- visl_debug: Activates debug info, printing various debug messages through dprintk. Also controls whether per-frame debug info is shown. Defaults to off. Note that enabling this feature can result in slow performance through serial.
- visl_transtime_ms: Simulated process time in milliseconds. Slowing down the decoding speed can be useful for debugging.
- visl_dprintk_frame_start, visl_dprintk_frame_nframes: Dictates a range of frames where dprintk is activated. This only controls the dprintk tracing on a per-frame basis. Note that printing a lot of data can be slow through serial.
- keep_bitstream_buffers: Controls whether bitstream (i.e. OUTPUT) buffers are kept after a decoding session. Defaults to false so as to reduce the amount of clutter. keep_bitstream_buffers == false works well when live debugging the client program with GDB.
- bitstream_trace_frame_start, bitstream_trace_nframes: Similar to visl_dprintk_frame_start, visl_dprintk_nframes, but controls the dumping of buffer data through debugfs instead.

What is the default use case for this driver?

This driver can be used as a way to compare different userspace implementations. This assumes that a working client is run against visl and that the ftrace and OUTPUT buffer data is subsequently used to debug a work-in-progress implementation.

Information on reference frames, their timestamps, the status of the OUTPUT and CAPTURE queues and more can be read directly from the CAPTURE buffers.

Supported codecs

The following codecs are supported:

- FWHT
- MPEG2
- VP8
- VP9
- H.264
- HEVC
- AV1

visl trace events

The trace events are defined on a per-codec basis, e.g.:

```
$ ls /sys/kernel/tracing/events/ | grep visl
visl_av1_controls
visl_fwht_controls
visl_h264_controls
visl_hevc_controls
visl_mpeg2_controls
visl_vp8_controls
visl_vp9_controls
```

For example, in order to dump HEVC SPS data:

```
$ echo 1 >  /sys/kernel/tracing/events/visl_hevc_controls/v4l2_ctrl_hevc_sps/
↪enable
```

The SPS data will be dumped to the trace buffer, i.e.:

```
$ cat /sys/kernel/tracing/trace
video_parameter_set_id 0
seq_parameter_set_id 0
pic_width_in_luma_samples 1920
pic_height_in_luma_samples 1080
bit_depth_luma_minus8 0
bit_depth_chroma_minus8 0
log2_max_pic_order_cnt_lsb_minus4 4
sps_max_dec_pic_buffering_minus1 6
sps_max_num_reorder_pics 2
sps_max_latency_increase_plus1 0
log2_min_luma_coding_block_size_minus3 0
log2_diff_max_min_luma_coding_block_size 3
log2_min_luma_transform_block_size_minus2 0
log2_diff_max_min_luma_transform_block_size 3
max_transform_hierarchy_depth_inter 2
max_transform_hierarchy_depth_intra 2
```

```
pcm_sample_bit_depth_luma_minus1 0
pcm_sample_bit_depth_chroma_minus1 0
log2_min_pcm_luma_coding_block_size_minus3 0
log2_diff_max_min_pcm_luma_coding_block_size 0
num_short_term_ref_pic_sets 0
num_long_term_ref_pics_sps 0
chroma_format_idc 1
sps_max_sub_layers_minus1 0
flags AMP_ENABLED|SAMPLE_ADAPTIVE_OFFSET|TEMPORAL_MVP_ENABLED|STRONG_INTRA_
↪SMOOTHING_ENABLED
```

Dumping OUTPUT buffer data through debugfs

If the **VISL_DEBUGFS** Kconfig is enabled, visl will populate **/sys/kernel/debug/visl/bitstream** with OUTPUT buffer data according to the values of bitstream_trace_frame_start and bitstream_trace_nframes. This can highlight errors as broken clients may fail to fill the buffers properly.

A single file is created for each processed OUTPUT buffer. Its name contains an integer that denotes the buffer sequence, i.e.:

```
snprintf(name, 32, "bitstream%d", run->src->sequence);
```

Dumping the values is simply a matter of reading from the file, i.e.:

For the buffer with sequence == 0:

```
$ xxd /sys/kernel/debug/visl/bitstream/bitstream0
00000000: 2601 af04 d088 bc25 a173 0e41 a4f2 3274  &......%.s.A..2t
00000010: c668 cb28 e775 b4ac f53a ba60 f8fd 3aa1  .h.(.u...:.`..:.
00000020: 46b4 bcfc 506c e227 2372 e5f5 d7ea 579f  F...Pl.'#r....W.
00000030: 6371 5eb5 0eb8 23b5 ca6a 5de5 983a 19e4  cq^...#..j]..:..
00000040: e8c3 4320 b4ba a226 cbc1 4138 3a12 32d6  ..C ...&..A8:.2.
00000050: fef3 247b 3523 4e90 9682 ac8e eb0c a389  ..${5#N........
00000060: ddd0 6cfc 0187 0e20 7aae b15b 1812 3d33  ..l.... z..[..=3
00000070: e1c5 f425 a83a 00b7 4f18 8127 3c4c aefb  ...%.:..O..'<L..
```

For the buffer with sequence == 1:

```
$ xxd /sys/kernel/debug/visl/bitstream/bitstream1
00000000: 0201 d021 49e1 0c40 aa11 1449 14a6 01dc  ...!I..@...I....
00000010: 7023 889a c8cd 2cd0 13b4 dab0 e8ca 21fe  p#....,.......!.
00000020: c4c8 ab4c 486e 4e2f b0df 96cc c74e 8dde  ...LHnN/.....N..
00000030: 8ce7 ee36 d880 4095 4d64 30a0 ff4f 0c5e  ...6..@.Md0..O.^
00000040: f16b a6a1 d806 ca2a 0ece a673 7bea 1f37  .k.....*...s{..7
00000050: 370f 5bb9 1dc4 ba21 6434 bc53 0173 cba0  7.[....!d4.S.s..
00000060: dfe6 bc99 01ea b6e0 346b 92b5 c8de 9f5d  ........4k.....]
00000070: e7cc 3484 1769 fef2 a693 a945 2c8b 31da  ..4..i.....E,.1.
```

And so on.

By default, the files are removed during STREAMOFF. This is to reduce the amount of clutter.

59.7.25 The Virtual Video Test Driver (vivid)

This driver emulates video4linux hardware of various types: video capture, video output, vbi capture and output, metadata capture and output, radio receivers and transmitters, touch capture and a software defined radio receiver. In addition a simple framebuffer device is available for testing capture and output overlays.

Up to 64 vivid instances can be created, each with up to 16 inputs and 16 outputs.

Each input can be a webcam, TV capture device, S-Video capture device or an HDMI capture device. Each output can be an S-Video output device or an HDMI output device.

These inputs and outputs act exactly as a real hardware device would behave. This allows you to use this driver as a test input for application development, since you can test the various features without requiring special hardware.

This document describes the features implemented by this driver:

- Support for read()/write(), MMAP, USERPTR and DMABUF streaming I/O.
- A large list of test patterns and variations thereof
- Working brightness, contrast, saturation and hue controls
- Support for the alpha color component
- Full colorspace support, including limited/full RGB range
- All possible control types are present
- Support for various pixel aspect ratios and video aspect ratios
- Error injection to test what happens if errors occur
- Supports crop/compose/scale in any combination for both input and output
- Can emulate up to 4K resolutions
- All Field settings are supported for testing interlaced capturing
- Supports all standard YUV and RGB formats, including two multiplanar YUV formats
- Raw and Sliced VBI capture and output support
- Radio receiver and transmitter support, including RDS support
- Software defined radio (SDR) support
- Capture and output overlay support
- Metadata capture and output support
- Touch capture support

These features will be described in more detail below.

Configuring the driver

By default the driver will create a single instance that has a video capture device with webcam, TV, S-Video and HDMI inputs, a video output device with S-Video and HDMI outputs, one vbi capture device, one vbi output device, one radio receiver device, one radio transmitter device and one SDR device.

The number of instances, devices, video inputs and outputs and their types are all configurable using the following module options:

- n_devs:

  number of driver instances to create. By default set to 1. Up to 64 instances can be created.

- node_types:

  which devices should each driver instance create. An array of hexadecimal values, one for each instance. The default is 0x1d3d. Each value is a bitmask with the following meaning:

  - bit 0: Video Capture node
  - bit 2-3: VBI Capture node: 0 = none, 1 = raw vbi, 2 = sliced vbi, 3 = both
  - bit 4: Radio Receiver node
  - bit 5: Software Defined Radio Receiver node
  - bit 8: Video Output node
  - bit 10-11: VBI Output node: 0 = none, 1 = raw vbi, 2 = sliced vbi, 3 = both
  - bit 12: Radio Transmitter node
  - bit 16: Framebuffer for testing overlays
  - bit 17: Metadata Capture node
  - bit 18: Metadata Output node
  - bit 19: Touch Capture node

  So to create four instances, the first two with just one video capture device, the second two with just one video output device you would pass these module options to vivid:

  ```
  n_devs=4 node_types=0x1,0x1,0x100,0x100
  ```

- num_inputs:

  the number of inputs, one for each instance. By default 4 inputs are created for each video capture device. At most 16 inputs can be created, and there must be at least one.

- input_types:

  the input types for each instance, the default is 0xe4. This defines what the type of each input is when the inputs are created for each driver instance. This is a hexadecimal value with up to 16 pairs of bits, each pair gives the type and bits

0-1 map to input 0, bits 2-3 map to input 1, 30-31 map to input 15. Each pair of bits has the following meaning:

- 00: this is a webcam input
- 01: this is a TV tuner input
- 10: this is an S-Video input
- 11: this is an HDMI input

So to create a video capture device with 8 inputs where input 0 is a TV tuner, inputs 1-3 are S-Video inputs and inputs 4-7 are HDMI inputs you would use the following module options:

```
num_inputs=8 input_types=0xffa9
```

- num_outputs:

  the number of outputs, one for each instance. By default 2 outputs are created for each video output device. At most 16 outputs can be created, and there must be at least one.

- output_types:

  the output types for each instance, the default is 0x02. This defines what the type of each output is when the outputs are created for each driver instance. This is a hexadecimal value with up to 16 bits, each bit gives the type and bit 0 maps to output 0, bit 1 maps to output 1, bit 15 maps to output 15. The meaning of each bit is as follows:

  - 0: this is an S-Video output
  - 1: this is an HDMI output

  So to create a video output device with 8 outputs where outputs 0-3 are S-Video outputs and outputs 4-7 are HDMI outputs you would use the following module options:

```
num_outputs=8 output_types=0xf0
```

- vid_cap_nr:

  give the desired videoX start number for each video capture device. The default is -1 which will just take the first free number. This allows you to map capture video nodes to specific videoX device nodes. Example:

```
n_devs=4 vid_cap_nr=2,4,6,8
```

  This will attempt to assign /dev/video2 for the video capture device of the first vivid instance, video4 for the next up to video8 for the last instance. If it can't succeed, then it will just take the next free number.

- vid_out_nr:

  give the desired videoX start number for each video output device. The default is -1 which will just take the first free number.

- vbi_cap_nr:

give the desired vbiX start number for each vbi capture device. The default is -1 which will just take the first free number.

- vbi_out_nr:

  give the desired vbiX start number for each vbi output device. The default is -1 which will just take the first free number.

- radio_rx_nr:

  give the desired radioX start number for each radio receiver device. The default is -1 which will just take the first free number.

- radio_tx_nr:

  give the desired radioX start number for each radio transmitter device. The default is -1 which will just take the first free number.

- sdr_cap_nr:

  give the desired swradioX start number for each SDR capture device. The default is -1 which will just take the first free number.

- meta_cap_nr:

  give the desired videoX start number for each metadata capture device. The default is -1 which will just take the first free number.

- meta_out_nr:

  give the desired videoX start number for each metadata output device. The default is -1 which will just take the first free number.

- touch_cap_nr:

  give the desired v4l-touchX start number for each touch capture device. The default is -1 which will just take the first free number.

- ccs_cap_mode:

  specify the allowed video capture crop/compose/scaling combination for each driver instance. Video capture devices can have any combination of cropping, composing and scaling capabilities and this will tell the vivid driver which of those is should emulate. By default the user can select this through controls.

  The value is either -1 (controlled by the user) or a set of three bits, each enabling (1) or disabling (0) one of the features:

  - bit 0:

    Enable crop support. Cropping will take only part of the incoming picture.

  - bit 1:

    Enable compose support. Composing will copy the incoming picture into a larger buffer.

  - bit 2:

Enable scaling support. Scaling can scale the incoming picture. The scaler of the vivid driver can enlarge up or down to four times the original size. The scaler is very simple and low-quality. Simplicity and speed were key, not quality.

Note that this value is ignored by webcam inputs: those enumerate discrete framesizes and that is incompatible with cropping, composing or scaling.

- ccs_out_mode:

  specify the allowed video output crop/compose/scaling combination for each driver instance. Video output devices can have any combination of cropping, composing and scaling capabilities and this will tell the vivid driver which of those is should emulate. By default the user can select this through controls.

  The value is either -1 (controlled by the user) or a set of three bits, each enabling (1) or disabling (0) one of the features:

  - bit 0:

    Enable crop support. Cropping will take only part of the outgoing buffer.

  - bit 1:

    Enable compose support. Composing will copy the incoming buffer into a larger picture frame.

  - bit 2:

    Enable scaling support. Scaling can scale the incoming buffer. The scaler of the vivid driver can enlarge up or down to four times the original size. The scaler is very simple and low-quality. Simplicity and speed were key, not quality.

- multiplanar:

  select whether each device instance supports multi-planar formats, and thus the V4L2 multi-planar API. By default device instances are single-planar.

  This module option can override that for each instance. Values are:

  - 1: this is a single-planar instance.
  - 2: this is a multi-planar instance.

- vivid_debug:

  enable driver debugging info

- no_error_inj:

  if set disable the error injecting controls. This option is needed in order to run a tool like v4l2-compliance. Tools like that exercise all controls including a control like 'Disconnect' which emulates a USB disconnect, making the device inaccessible and so all tests that v4l2-compliance is doing will fail afterwards.

  There may be other situations as well where you want to disable the error injection support of vivid. When this option is set, then the controls that select crop, compose and scale behavior are also removed. Unless overridden by ccs_cap_mode and/or ccs_out_mode the will default to enabling crop, compose and scaling.

- allocators:

  memory allocator selection, default is 0. It specifies the way buffers will be allocated.

  - 0: vmalloc
  - 1: dma-contig

- cache_hints:

  specifies if the device should set queues' user-space cache and memory consistency hint capability (V4L2_BUF_CAP_SUPPORTS_MMAP_CACHE_HINTS). The hints are valid only when using MMAP streaming I/O. Default is 0.

  - 0: forbid hints
  - 1: allow hints

Taken together, all these module options allow you to precisely customize the driver behavior and test your application with all sorts of permutations. It is also very suitable to emulate hardware that is not yet available, e.g. when developing software for a new upcoming device.

Video Capture

This is probably the most frequently used feature. The video capture device can be configured by using the module options num_inputs, input_types and ccs_cap_mode (see section 1 for more detailed information), but by default four inputs are configured: a webcam, a TV tuner, an S-Video and an HDMI input, one input for each input type. Those are described in more detail below.

Special attention has been given to the rate at which new frames become available. The jitter will be around 1 jiffie (that depends on the HZ configuration of your kernel, so usually 1/100, 1/250 or 1/1000 of a second), but the long-term behavior is exactly following the framerate. So a framerate of 59.94 Hz is really different from 60 Hz. If the framerate exceeds your kernel's HZ value, then you will get dropped frames, but the frame/field sequence counting will keep track of that so the sequence count will skip whenever frames are dropped.

Webcam Input

The webcam input supports three framesizes: 320x180, 640x360 and 1280x720. It supports frames per second settings of 10, 15, 25, 30, 50 and 60 fps. Which ones are available depends on the chosen framesize: the larger the framesize, the lower the maximum frames per second.

The initially selected colorspace when you switch to the webcam input will be sRGB.

TV and S-Video Inputs

The only difference between the TV and S-Video input is that the TV has a tuner. Otherwise they behave identically.

These inputs support audio inputs as well: one TV and one Line-In. They both support all TV standards. If the standard is queried, then the Vivid controls 'Standard Signal Mode' and 'Standard' determine what the result will be.

These inputs support all combinations of the field setting. Special care has been taken to faithfully reproduce how fields are handled for the different TV standards. This is particularly noticeable when generating a horizontally moving image so the temporal effect of using interlaced formats becomes clearly visible. For 50 Hz standards the top field is the oldest and the bottom field is the newest in time. For 60 Hz standards that is reversed: the bottom field is the oldest and the top field is the newest in time.

When you start capturing in V4L2_FIELD_ALTERNATE mode the first buffer will contain the top field for 50 Hz standards and the bottom field for 60 Hz standards. This is what capture hardware does as well.

Finally, for PAL/SECAM standards the first half of the top line contains noise. This simulates the Wide Screen Signal that is commonly placed there.

The initially selected colorspace when you switch to the TV or S-Video input will be SMPTE-170M.

The pixel aspect ratio will depend on the TV standard. The video aspect ratio can be selected through the 'Standard Aspect Ratio' Vivid control. Choices are '4x3', '16x9' which will give letterboxed widescreen video and '16x9 Anamorphic' which will give full screen squashed anamorphic widescreen video that will need to be scaled accordingly.

The TV 'tuner' supports a frequency range of 44-958 MHz. Channels are available every 6 MHz, starting from 49.25 MHz. For each channel the generated image will be in color for the +/- 0.25 MHz around it, and in grayscale for +/- 1 MHz around the channel. Beyond that it is just noise. The VIDIOC_G_TUNER ioctl will return 100% signal strength for +/- 0.25 MHz and 50% for +/- 1 MHz. It will also return correct afc values to show whether the frequency is too low or too high.

The audio subchannels that are returned are MONO for the +/- 1 MHz range around a valid channel frequency. When the frequency is within +/- 0.25 MHz of the channel it will return either MONO, STEREO, either MONO | SAP (for NTSC) or LANG1 | LANG2 (for others), or STEREO | SAP.

Which one is returned depends on the chosen channel, each next valid channel will cycle through the possible audio subchannel combinations. This allows you to test the various combinations by just switching channels..

Finally, for these inputs the v4l2_timecode struct is filled in the dequeued v4l2_buffer struct.

HDMI Input

The HDMI inputs supports all CEA-861 and DMT timings, both progressive and interlaced, for pixelclock frequencies between 25 and 600 MHz. The field mode for interlaced formats is always V4L2_FIELD_ALTERNATE. For HDMI the field order is always top field first, and when you start capturing an interlaced format you will receive the top field first.

The initially selected colorspace when you switch to the HDMI input or select an HDMI timing is based on the format resolution: for resolutions less than or equal to 720x576 the colorspace is set to SMPTE-170M, for others it is set to REC-709 (CEA-861 timings) or sRGB (VESA DMT timings).

The pixel aspect ratio will depend on the HDMI timing: for 720x480 is it set as for the NTSC TV standard, for 720x576 it is set as for the PAL TV standard, and for all others a 1:1 pixel aspect ratio is returned.

The video aspect ratio can be selected through the 'DV Timings Aspect Ratio' Vivid control. Choices are 'Source Width x Height' (just use the same ratio as the chosen format), '4x3' or '16x9', either of which can result in pillarboxed or letterboxed video.

For HDMI inputs it is possible to set the EDID. By default a simple EDID is provided. You can only set the EDID for HDMI inputs. Internally, however, the EDID is shared between all HDMI inputs.

No interpretation is done of the EDID data with the exception of the physical address. See the CEC section for more details.

There is a maximum of 15 HDMI inputs (if there are more, then they will be reduced to 15) since that's the limitation of the EDID physical address.

Video Output

The video output device can be configured by using the module options num_outputs, output_types and ccs_out_mode (see section 1 for more detailed information), but by default two outputs are configured: an S-Video and an HDMI input, one output for each output type. Those are described in more detail below.

Like with video capture the framerate is also exact in the long term.

S-Video Output

This output supports audio outputs as well: ”Line-Out 1” and ”Line-Out 2”. The S-Video output supports all TV standards.

This output supports all combinations of the field setting.

The initially selected colorspace when you switch to the TV or S-Video input will be SMPTE-170M.

HDMI Output

The HDMI output supports all CEA-861 and DMT timings, both progressive and interlaced, for pixelclock frequencies between 25 and 600 MHz. The field mode for interlaced formats is always V4L2_FIELD_ALTERNATE.

The initially selected colorspace when you switch to the HDMI output or select an HDMI timing is based on the format resolution: for resolutions less than or equal to 720x576 the colorspace is set to SMPTE-170M, for others it is set to REC-709 (CEA-861 timings) or sRGB (VESA DMT timings).

The pixel aspect ratio will depend on the HDMI timing: for 720x480 is it set as for the NTSC TV standard, for 720x576 it is set as for the PAL TV standard, and for all others a 1:1 pixel aspect ratio is returned.

An HDMI output has a valid EDID which can be obtained through VIDIOC_G_EDID.

There is a maximum of 15 HDMI outputs (if there are more, then they will be reduced to 15) since that's the limitation of the EDID physical address. See also the CEC section for more details.

VBI Capture

There are three types of VBI capture devices: those that only support raw (undecoded) VBI, those that only support sliced (decoded) VBI and those that support both. This is determined by the node_types module option. In all cases the driver will generate valid VBI data: for 60 Hz standards it will generate Closed Caption and XDS data. The closed caption stream will alternate between ”Hello world!” and ”Closed captions test” every second. The XDS stream will give the current time once a minute. For 50 Hz standards it will generate the Wide Screen Signal which is based on the actual Video Aspect Ratio control setting and teletext pages 100-159, one page per frame.

The VBI device will only work for the S-Video and TV inputs, it will give back an error if the current input is a webcam or HDMI.

VBI Output

There are three types of VBI output devices: those that only support raw (undecoded) VBI, those that only support sliced (decoded) VBI and those that support both. This is determined by the node_types module option.

The sliced VBI output supports the Wide Screen Signal and the teletext signal for 50 Hz standards and Closed Captioning + XDS for 60 Hz standards.

The VBI device will only work for the S-Video output, it will give back an error if the current output is HDMI.

Radio Receiver

The radio receiver emulates an FM/AM/SW receiver. The FM band also supports RDS. The frequency ranges are:

- FM: 64 MHz - 108 MHz
- AM: 520 kHz - 1710 kHz
- SW: 2300 kHz - 26.1 MHz

Valid channels are emulated every 1 MHz for FM and every 100 kHz for AM and SW. The signal strength decreases the further the frequency is from the valid frequency until it becomes 0% at +/- 50 kHz (FM) or 5 kHz (AM/SW) from the ideal frequency. The initial frequency when the driver is loaded is set to 95 MHz.

The FM receiver supports RDS as well, both using 'Block I/O' and 'Controls' modes. In the 'Controls' mode the RDS information is stored in read-only controls. These controls are updated every time the frequency is changed, or when the tuner status is requested. The Block I/O method uses the read() interface to pass the RDS blocks on to the application for decoding.

The RDS signal is 'detected' for +/- 12.5 kHz around the channel frequency, and the further the frequency is away from the valid frequency the more RDS errors are randomly introduced into the block I/O stream, up to 50% of all blocks if you are +/- 12.5 kHz from the channel frequency. All four errors can occur in equal proportions: blocks marked 'CORRECTED', blocks marked 'ERROR', blocks marked 'INVALID' and dropped blocks.

The generated RDS stream contains all the standard fields contained in a 0B group, and also radio text and the current time.

The receiver supports HW frequency seek, either in Bounded mode, Wrap Around mode or both, which is configurable with the ”Radio HW Seek Mode” control.

Radio Transmitter

The radio transmitter emulates an FM/AM/SW transmitter. The FM band also supports RDS. The frequency ranges are:

- FM: 64 MHz - 108 MHz
- AM: 520 kHz - 1710 kHz
- SW: 2300 kHz - 26.1 MHz

The initial frequency when the driver is loaded is 95.5 MHz.

The FM transmitter supports RDS as well, both using 'Block I/O' and 'Controls' modes. In the 'Controls' mode the transmitted RDS information is configured using controls, and in 'Block I/O' mode the blocks are passed to the driver using write().

Software Defined Radio Receiver

The SDR receiver has three frequency bands for the ADC tuner:

- 300 kHz
- 900 kHz - 2800 kHz
- 3200 kHz

The RF tuner supports 50 MHz - 2000 MHz.

The generated data contains the In-phase and Quadrature components of a 1 kHz tone that has an amplitude of sqrt(2).

Metadata Capture

The Metadata capture generates UVC format metadata. The PTS and SCR are transmitted based on the values set in vivid controls.

The Metadata device will only work for the Webcam input, it will give back an error for all other inputs.

Metadata Output

The Metadata output can be used to set brightness, contrast, saturation and hue.

The Metadata device will only work for the Webcam output, it will give back an error for all other outputs.

Touch Capture

The Touch capture generates touch patterns simulating single tap, double tap, triple tap, move from left to right, zoom in, zoom out, palm press (simulating a large area being pressed on a touchpad), and simulating 16 simultaneous touch points.

Controls

Different devices support different controls. The sections below will describe each control and which devices support them.

User Controls - Test Controls

The Button, Boolean, Integer 32 Bits, Integer 64 Bits, Menu, String, Bitmask and Integer Menu are controls that represent all possible control types. The Menu control and the Integer Menu control both have 'holes' in their menu list, meaning that one or more menu items return EINVAL when VIDIOC_QUERYMENU is called. Both menu controls also have a non-zero minimum control value. These features allow you to check if your application can handle such things correctly. These controls are supported for every device type.

User Controls - Video Capture

The following controls are specific to video capture.

The Brightness, Contrast, Saturation and Hue controls actually work and are standard. There is one special feature with the Brightness control: each video input has its own brightness value, so changing input will restore the brightness for that input. In addition, each video input uses a different brightness range (minimum and maximum control values). Switching inputs will cause a control event to be sent with the V4L2_EVENT_CTRL_CH_RANGE flag set. This allows you to test controls that can change their range.

The 'Gain, Automatic' and Gain controls can be used to test volatile controls: if 'Gain, Automatic' is set, then the Gain control is volatile and changes constantly. If 'Gain, Automatic' is cleared, then the Gain control is a normal control.

The 'Horizontal Flip' and 'Vertical Flip' controls can be used to flip the image. These combine with the 'Sensor Flipped Horizontally/Vertically' Vivid controls.

The 'Alpha Component' control can be used to set the alpha component for formats containing an alpha channel.

User Controls - Audio

The following controls are specific to video capture and output and radio receivers and transmitters.

The 'Volume' and 'Mute' audio controls are typical for such devices to control the volume and mute the audio. They don't actually do anything in the vivid driver.

Vivid Controls

These vivid custom controls control the image generation, error injection, etc.

Test Pattern Controls

The Test Pattern Controls are all specific to video capture.

- Test Pattern:

  selects which test pattern to use. Use the CSC Colorbar for testing colorspace conversions: the colors used in that test pattern map to valid colors in all colorspaces. The colorspace conversion is disabled for the other test patterns.

- OSD Text Mode:

  selects whether the text superimposed on the test pattern should be shown, and if so, whether only counters should be displayed or the full text.

- Horizontal Movement:

  selects whether the test pattern should move to the left or right and at what speed.

- Vertical Movement:

  does the same for the vertical direction.

- Show Border:

  show a two-pixel wide border at the edge of the actual image, excluding letter or pillarboxing.

- Show Square:

  show a square in the middle of the image. If the image is displayed with the correct pixel and image aspect ratio corrections, then the width and height of the square on the monitor should be the same.

- Insert SAV Code in Image:

  adds a SAV (Start of Active Video) code to the image. This can be used to check if such codes in the image are inadvertently interpreted instead of being ignored.

- Insert EAV Code in Image:

  does the same for the EAV (End of Active Video) code.

- Insert Video Guard Band

  adds 4 columns of pixels with the HDMI Video Guard Band code at the left hand side of the image. This only works with 3 or 4 byte RGB pixel formats. The RGB pixel value 0xab/0x55/0xab turns out to be equivalent to the HDMI Video Guard Band code that precedes each active video line (see section 5.2.2.1 in the HDMI 1.3 Specification). To test if a video receiver has correct HDMI Video Guard Band processing, enable this control and then move the image to the left hand side of the screen. That will result in video lines that start with multiple pixels that have the same value as the Video Guard Band that precedes them. Receivers that will just keep skipping Video Guard Band values will now fail and either loose sync or these video lines will shift.

Capture Feature Selection Controls

These controls are all specific to video capture.

- Sensor Flipped Horizontally:

  the image is flipped horizontally and the V4L2_IN_ST_HFLIP input status flag is set. This emulates the case where a sensor is for example mounted upside down.

- Sensor Flipped Vertically:

  the image is flipped vertically and the V4L2_IN_ST_VFLIP input status flag is set. This emulates the case where a sensor is for example mounted upside down.

- Standard Aspect Ratio:

  selects if the image aspect ratio as used for the TV or S-Video input should be 4x3, 16x9 or anamorphic widescreen. This may introduce letterboxing.

- DV Timings Aspect Ratio:

  selects if the image aspect ratio as used for the HDMI input should be the same as the source width and height ratio, or if it should be 4x3 or 16x9. This may introduce letter or pillarboxing.

- Timestamp Source:

selects when the timestamp for each buffer is taken.

- Colorspace:

  selects which colorspace should be used when generating the image. This only applies if the CSC Colorbar test pattern is selected, otherwise the test pattern will go through unconverted. This behavior is also what you want, since a 75% Colorbar should really have 75% signal intensity and should not be affected by colorspace conversions.

  Changing the colorspace will result in the V4L2_EVENT_SOURCE_CHANGE to be sent since it emulates a detected colorspace change.

- Transfer Function:

  selects which colorspace transfer function should be used when generating an image. This only applies if the CSC Colorbar test pattern is selected, otherwise the test pattern will go through unconverted. This behavior is also what you want, since a 75% Colorbar should really have 75% signal intensity and should not be affected by colorspace conversions.

  Changing the transfer function will result in the V4L2_EVENT_SOURCE_CHANGE to be sent since it emulates a detected colorspace change.

- Y'CbCr Encoding:

  selects which Y'CbCr encoding should be used when generating a Y'CbCr image. This only applies if the format is set to a Y'CbCr format as opposed to an RGB format.

  Changing the Y'CbCr encoding will result in the V4L2_EVENT_SOURCE_CHANGE to be sent since it emulates a detected colorspace change.

- Quantization:

  selects which quantization should be used for the RGB or Y'CbCr encoding when generating the test pattern.

  Changing the quantization will result in the V4L2_EVENT_SOURCE_CHANGE to be sent since it emulates a detected colorspace change.

- Limited RGB Range (16-235):

  selects if the RGB range of the HDMI source should be limited or full range. This combines with the Digital Video 'Rx RGB Quantization Range' control and can be used to test what happens if a source provides you with the wrong quantization range information. See the description of that control for more details.

- Apply Alpha To Red Only:

  apply the alpha channel as set by the 'Alpha Component' user control to the red color of the test pattern only.

- Enable Capture Cropping:

  enables crop support. This control is only present if the ccs_cap_mode module option is set to the default value of -1 and if the no_error_inj module option is set to 0 (the default).

- Enable Capture Composing:

enables composing support. This control is only present if the ccs_cap_mode module option is set to the default value of -1 and if the no_error_inj module option is set to 0 (the default).

- Enable Capture Scaler:

  enables support for a scaler (maximum 4 times upscaling and downscaling). This control is only present if the ccs_cap_mode module option is set to the default value of -1 and if the no_error_inj module option is set to 0 (the default).

- Maximum EDID Blocks:

  determines how many EDID blocks the driver supports. Note that the vivid driver does not actually interpret new EDID data, it just stores it. It allows for up to 256 EDID blocks which is the maximum supported by the standard.

- Fill Percentage of Frame:

  can be used to draw only the top X percent of the image. Since each frame has to be drawn by the driver, this demands a lot of the CPU. For large resolutions this becomes problematic. By drawing only part of the image this CPU load can be reduced.

Output Feature Selection Controls

These controls are all specific to video output.

- Enable Output Cropping:

  enables crop support. This control is only present if the ccs_out_mode module option is set to the default value of -1 and if the no_error_inj module option is set to 0 (the default).

- Enable Output Composing:

  enables composing support. This control is only present if the ccs_out_mode module option is set to the default value of -1 and if the no_error_inj module option is set to 0 (the default).

- Enable Output Scaler:

  enables support for a scaler (maximum 4 times upscaling and downscaling). This control is only present if the ccs_out_mode module option is set to the default value of -1 and if the no_error_inj module option is set to 0 (the default).

Error Injection Controls

The following two controls are only valid for video and vbi capture.

- Standard Signal Mode:

  selects the behavior of VIDIOC_QUERYSTD: what should it return?

  Changing this control will result in the V4L2_EVENT_SOURCE_CHANGE to be sent since it emulates a changed input condition (e.g. a cable was plugged in or out).

- Standard:

  selects the standard that VIDIOC_QUERYSTD should return if the previous control is set to ”Selected Standard”.

  Changing this control will result in the V4L2_EVENT_SOURCE_CHANGE to be sent since it emulates a changed input standard.

The following two controls are only valid for video capture.

- DV Timings Signal Mode:

  selects the behavior of VIDIOC_QUERY_DV_TIMINGS: what should it return?

  Changing this control will result in the V4L2_EVENT_SOURCE_CHANGE to be sent since it emulates a changed input condition (e.g. a cable was plugged in or out).

- DV Timings:

  selects the timings the VIDIOC_QUERY_DV_TIMINGS should return if the previous control is set to ”Selected DV Timings”.

  Changing this control will result in the V4L2_EVENT_SOURCE_CHANGE to be sent since it emulates changed input timings.

The following controls are only present if the no_error_inj module option is set to 0 (the default). These controls are valid for video and vbi capture and output streams and for the SDR capture device except for the Disconnect control which is valid for all devices.

- Wrap Sequence Number:

  test what happens when you wrap the sequence number in struct v4l2_buffer around.

- Wrap Timestamp:

  test what happens when you wrap the timestamp in struct v4l2_buffer around.

- Percentage of Dropped Buffers:

  sets the percentage of buffers that are never returned by the driver (i.e., they are dropped).

- Disconnect:

  emulates a USB disconnect. The device will act as if it has been disconnected. Only after all open filehandles to the device node have been closed will the device become 'connected' again.

- Inject V4L2_BUF_FLAG_ERROR:

  when pressed, the next frame returned by the driver will have the error flag set (i.e. the frame is marked corrupt).

- Inject VIDIOC_REQBUFS Error:

  when pressed, the next REQBUFS or CREATE_BUFS ioctl call will fail with an error. To be precise: the videobuf2 queue_setup() op will return -EINVAL.

- Inject VIDIOC_QBUF Error:

when pressed, the next VIDIOC_QBUF or VIDIOC_PREPARE_BUFFER ioctl call will fail with an error. To be precise: the videobuf2 buf_prepare() op will return -EINVAL.

- Inject VIDIOC_STREAMON Error:

  when pressed, the next VIDIOC_STREAMON ioctl call will fail with an error. To be precise: the videobuf2 start_streaming() op will return -EINVAL.

- Inject Fatal Streaming Error:

  when pressed, the streaming core will be marked as having suffered a fatal error, the only way to recover from that is to stop streaming. To be precise: the videobuf2 vb2_queue_error() function is called.

VBI Raw Capture Controls

- Interlaced VBI Format:

  if set, then the raw VBI data will be interlaced instead of providing it grouped by field.

Digital Video Controls

- Rx RGB Quantization Range:

  sets the RGB quantization detection of the HDMI input. This combines with the Vivid 'Limited RGB Range (16-235)' control and can be used to test what happens if a source provides you with the wrong quantization range information. This can be tested by selecting an HDMI input, setting this control to Full or Limited range and selecting the opposite in the 'Limited RGB Range (16-235)' control. The effect is easy to see if the 'Gray Ramp' test pattern is selected.

- Tx RGB Quantization Range:

  sets the RGB quantization detection of the HDMI output. It is currently not used for anything in vivid, but most HDMI transmitters would typically have this control.

- Transmit Mode:

  sets the transmit mode of the HDMI output to HDMI or DVI-D. This affects the reported colorspace since DVI_D outputs will always use sRGB.

- Display Present:

  sets the presence of a ”display” on the HDMI output. This affects the tx_edid_present, tx_hotplug and tx_rxsense controls.

FM Radio Receiver Controls

- RDS Reception:

  set if the RDS receiver should be enabled.

- RDS Program Type:
- RDS PS Name:
- RDS Radio Text:
- RDS Traffic Announcement:
- RDS Traffic Program:
- RDS Music:

  these are all read-only controls. If RDS Rx I/O Mode is set to "Block I/O", then they are inactive as well. If RDS Rx I/O Mode is set to "Controls", then these controls report the received RDS data.

Note: The vivid implementation of this is pretty basic: they are only updated when you set a new frequency or when you get the tuner status (VIDIOC_G_TUNER).

- Radio HW Seek Mode:

  can be one of "Bounded", "Wrap Around" or "Both". This determines if VIDIOC_S_HW_FREQ_SEEK will be bounded by the frequency range or wrap-around or if it is selectable by the user.

- Radio Programmable HW Seek:

  if set, then the user can provide the lower and upper bound of the HW Seek. Otherwise the frequency range boundaries will be used.

- Generate RBDS Instead of RDS:

  if set, then generate RBDS (the US variant of RDS) data instead of RDS (European-style RDS). This affects only the PICODE and PTY codes.

- RDS Rx I/O Mode:

  this can be "Block I/O" where the RDS blocks have to be read() by the application, or "Controls" where the RDS data is provided by the RDS controls mentioned above.

FM Radio Modulator Controls

- RDS Program ID:
- RDS Program Type:
- RDS PS Name:
- RDS Radio Text:
- RDS Stereo:

- RDS Artificial Head:
- RDS Compressed:
- RDS Dynamic PTY:
- RDS Traffic Announcement:
- RDS Traffic Program:
- RDS Music:

  these are all controls that set the RDS data that is transmitted by the FM modulator.
- RDS Tx I/O Mode:

  this can be ”Block I/O” where the application has to use write() to pass the RDS blocks to the driver, or ”Controls” where the RDS data is Provided by the RDS controls mentioned above.

Metadata Capture Controls

- Generate PTS

  if set, then the generated metadata stream contains Presentation timestamp.
- Generate SCR

  if set, then the generated metadata stream contains Source Clock information.

Video, VBI and RDS Looping

The vivid driver supports looping of video output to video input, VBI output to VBI input and RDS output to RDS input. For video/VBI looping this emulates as if a cable was hooked up between the output and input connector. So video and VBI looping is only supported between S-Video and HDMI inputs and outputs. VBI is only valid for S-Video as it makes no sense for HDMI.

Since radio is wireless this looping always happens if the radio receiver frequency is close to the radio transmitter frequency. In that case the radio transmitter will 'override' the emulated radio stations.

Looping is currently supported only between devices created by the same vivid driver instance.

Video and Sliced VBI looping

The way to enable video/VBI looping is currently fairly crude. A 'Loop Video' control is available in the ”Vivid” control class of the video capture and VBI capture devices. When checked the video looping will be enabled. Once enabled any video S-Video or HDMI input will show a static test pattern until the video output has started. At that time the video output will be looped to the video input provided that:

- the input type matches the output type. So the HDMI input cannot receive video from the S-Video output.

- the video resolution of the video input must match that of the video output. So it is not possible to loop a 50 Hz (720x576) S-Video output to a 60 Hz (720x480) S-Video input, or a 720p60 HDMI output to a 1080p30 input.
- the pixel formats must be identical on both sides. Otherwise the driver would have to do pixel format conversion as well, and that's taking things too far.
- the field settings must be identical on both sides. Same reason as above: requiring the driver to convert from one field format to another complicated matters too much. This also prohibits capturing with 'Field Top' or 'Field Bottom' when the output video is set to 'Field Alternate'. This combination, while legal, became too complicated to support. Both sides have to be 'Field Alternate' for this to work. Also note that for this specific case the sequence and field counting in struct v4l2_buffer on the capture side may not be 100% accurate.
- field settings V4L2_FIELD_SEQ_TB/BT are not supported. While it is possible to implement this, it would mean a lot of work to get this right. Since these field values are rarely used the decision was made not to implement this for now.
- on the input side the ”Standard Signal Mode” for the S-Video input or the ”DV Timings Signal Mode” for the HDMI input should be configured so that a valid signal is passed to the video input.

The framerates do not have to match, although this might change in the future.

By default you will see the OSD text superimposed on top of the looped video. This can be turned off by changing the ”OSD Text Mode” control of the video capture device.

For VBI looping to work all of the above must be valid and in addition the vbi output must be configured for sliced VBI. The VBI capture side can be configured for either raw or sliced VBI. Note that at the moment only CC/XDS (60 Hz formats) and WSS (50 Hz formats) VBI data is looped. Teletext VBI data is not looped.

Radio & RDS Looping

As mentioned in section 6 the radio receiver emulates stations are regular frequency intervals. Depending on the frequency of the radio receiver a signal strength value is calculated (this is returned by VIDIOC_G_TUNER). However, it will also look at the frequency set by the radio transmitter and if that results in a higher signal strength than the settings of the radio transmitter will be used as if it was a valid station. This also includes the RDS data (if any) that the transmitter 'transmits'. This is received faithfully on the receiver side. Note that when the driver is loaded the frequencies of the radio receiver and transmitter are not identical, so initially no looping takes place.

Cropping, Composing, Scaling

This driver supports cropping, composing and scaling in any combination. Normally which features are supported can be selected through the Vivid controls, but it is also possible to hardcode it when the module is loaded through the ccs_cap_mode and ccs_out_mode module options. See section 1 on the details of these module options.

This allows you to test your application for all these variations.

Note that the webcam input never supports cropping, composing or scaling. That only applies to the TV/S-Video/HDMI inputs and outputs. The reason is that webcams, including this virtual implementation, normally use VIDIOC_ENUM_FRAMESIZES to list a set of discrete framesizes that it supports. And that does not combine with cropping, composing or scaling. This is primarily a limitation of the V4L2 API which is carefully reproduced here.

The minimum and maximum resolutions that the scaler can achieve are 16x16 and (4096 * 4) x (2160 x 4), but it can only scale up or down by a factor of 4 or less. So for a source resolution of 1280x720 the minimum the scaler can do is 320x180 and the maximum is 5120x2880. You can play around with this using the qv4l2 test tool and you will see these dependencies.

This driver also supports larger 'bytesperline' settings, something that VIDIOC_S_FMT allows but that few drivers implement.

The scaler is a simple scaler that uses the Coarse Bresenham algorithm. It's designed for speed and simplicity, not quality.

If the combination of crop, compose and scaling allows it, then it is possible to change crop and compose rectangles on the fly.

Formats

The driver supports all the regular packed and planar 4:4:4, 4:2:2 and 4:2:0 YUYV formats, 8, 16, 24 and 32 RGB packed formats and various multiplanar formats.

The alpha component can be set through the 'Alpha Component' User control for those formats that support it. If the 'Apply Alpha To Red Only' control is set, then the alpha component is only used for the color red and set to 0 otherwise.

The driver has to be configured to support the multiplanar formats. By default the driver instances are single-planar. This can be changed by setting the multiplanar module option, see section 1 for more details on that option.

If the driver instance is using the multiplanar formats/API, then the first single planar format (YUYV) and the multiplanar NV16M and NV61M formats the will have a plane that has a non-zero data_offset of 128 bytes. It is rare for data_offset to be non-zero, so this is a useful feature for testing applications.

Video output will also honor any data_offset that the application set.

Capture Overlay

Note: capture overlay support is implemented primarily to test the existing V4L2 capture overlay API. In practice few if any GPUs support such overlays anymore, and neither are they generally needed anymore since modern hardware is so much more capable. By setting flag 0x10000 in the node_types module option the vivid driver will create a simple framebuffer device that can be used for testing this API. Whether this API should be used for new drivers is questionable.

This driver has support for a destructive capture overlay with bitmap clipping and list clipping (up to 16 rectangles) capabilities. Overlays are not supported for multiplanar formats. It also honors the struct v4l2_window field setting: if it is set to FIELD_TOP or FIELD_BOTTOM and the capture setting is FIELD_ALTERNATE, then only the top or bottom fields will be copied to the overlay.

The overlay only works if you are also capturing at that same time. This is a vivid limitation since it copies from a buffer to the overlay instead of filling the overlay directly. And if you are not capturing, then no buffers are available to fill.

In addition, the pixelformat of the capture format and that of the framebuffer must be the same for the overlay to work. Otherwise VIDIOC_OVERLAY will return an error.

In order to really see what it going on you will need to create two vivid instances: the first with a framebuffer enabled. You configure the capture overlay of the second instance to use the framebuffer of the first, then you start capturing in the second instance. For the first instance you setup the output overlay for the video output, turn on video looping and capture to see the blended framebuffer overlay that's being written to by the second instance. This setup would require the following commands:

```
$ sudo modprobe vivid n_devs=2 node_types=0x10101,0x1
$ v4l2-ctl -d1 --find-fb
/dev/fb1 is the framebuffer associated with base address 0x12800000
$ sudo v4l2-ctl -d2 --set-fbuf fb=1
$ v4l2-ctl -d1 --set-fbuf fb=1
$ v4l2-ctl -d0 --set-fmt-video=pixelformat='AR15'
$ v4l2-ctl -d1 --set-fmt-video-out=pixelformat='AR15'
$ v4l2-ctl -d2 --set-fmt-video=pixelformat='AR15'
$ v4l2-ctl -d0 -i2
$ v4l2-ctl -d2 -i2
$ v4l2-ctl -d2 -c horizontal_movement=4
$ v4l2-ctl -d1 --overlay=1
$ v4l2-ctl -d0 -c loop_video=1
$ v4l2-ctl -d2 --stream-mmap --overlay=1
```

And from another console:

```
$ v4l2-ctl -d1 --stream-out-mmap
```

And yet another console:

```
$ qv4l2
```

and start streaming.

As you can see, this is not for the faint of heart...

Output Overlay

Note: output overlays are primarily implemented in order to test the existing V4L2 output overlay API. Whether this API should be used for new drivers is questionable.

This driver has support for an output overlay and is capable of:

- bitmap clipping,
- list clipping (up to 16 rectangles)
- chromakey
- source chromakey
- global alpha
- local alpha
- local inverse alpha

Output overlays are not supported for multiplanar formats. In addition, the pixelformat of the capture format and that of the framebuffer must be the same for the overlay to work. Otherwise VIDIOC_OVERLAY will return an error.

Output overlays only work if the driver has been configured to create a framebuffer by setting flag 0x10000 in the node_types module option. The created framebuffer has a size of 720x576 and supports ARGB 1:5:5:5 and RGB 5:6:5.

In order to see the effects of the various clipping, chromakeying or alpha processing capabilities you need to turn on video looping and see the results on the capture side. The use of the clipping, chromakeying or alpha processing capabilities will slow down the video loop considerably as a lot of checks have to be done per pixel.

CEC (Consumer Electronics Control)

If there are HDMI inputs then a CEC adapter will be created that has the same number of input ports. This is the equivalent of e.g. a TV that has that number of inputs. Each HDMI output will also create a CEC adapter that is hooked up to the corresponding input port, or (if there are more outputs than inputs) is not hooked up at all. In other words, this is the equivalent of hooking up each output device to an input port of the TV. Any remaining output devices remain unconnected.

The EDID that each output reads reports a unique CEC physical address that is based on the physical address of the EDID of the input. So if the EDID of the receiver has physical address A.B.0.0, then each output will see an EDID containing physical address A.B.C.0 where C is 1 to the number of inputs. If there are more outputs than inputs then the remaining outputs have a CEC adapter that is disabled and reports an invalid physical address.

Some Future Improvements

Just as a reminder and in no particular order:

- Add a virtual alsa driver to test audio
- Add virtual sub-devices and media controller support
- Some support for testing compressed video
- Add support to loop raw VBI output to raw VBI input
- Add support to loop teletext sliced VBI output to VBI input
- Fix sequence/field numbering when looping of video with alternate fields
- Add support for V4L2_CID_BG_COLOR for video outputs
- Add ARGB888 overlay support: better testing of the alpha channel
- Improve pixel aspect support in the tpg code by passing a real v4l2_fract
- Use per-queue locks and/or per-device locks to improve throughput
- Add support to loop from a specific output to a specific input across vivid instances
- The SDR radio should use the same 'frequencies' for stations as the normal radio receiver, and give back noise if the frequency doesn't match up with a station frequency
- Make a thread for the RDS generation, that would help in particular for the ”Controls” RDS Rx I/O Mode as the read-only RDS controls could be updated in real-time.
- Changing the EDID should cause hotplug detect emulation to happen.

59.8 Digital TV driver-specific documentation

59.8.1 Avermedia DVB-T on BT878 Release Notes

February 14th 2006

Note: Several other Avermedia devices are supported. For a more broader and updated content about that, please check:

https://linuxtv.org/wiki/index.php/AVerMedia

The Avermedia DVB-T

The Avermedia DVB-T is a budget PCI DVB card. It has 3 inputs:

- RF Tuner Input
- Composite Video Input (RCA Jack)
- SVIDEO Input (Mini-DIN)

The RF Tuner Input is the input to the tuner module of the card. The Tuner is otherwise known as the ”Frontend” . The Frontend of the Avermedia DVB-T is a Microtune 7202D. A timely post to the linux-dvb mailing list ascertained that the Microtune 7202D is supported by the sp887x driver which is found in the dvb-hw CVS module.

The DVB-T card is based around the BT878 chip which is a very common multimedia bridge and often found on Analogue TV cards. There is no on-board MPEG2 decoder, which means that all MPEG2 decoding must be done in software, or if you have one, on an MPEG2 hardware decoding card or chipset.

Getting the card going

At this stage, it has not been able to ascertain the functionality of the remaining device nodes in respect of the Avermedia DVBT. However, full functionality in respect of tuning, receiving and supplying the MPEG2 data stream is possible with the currently available versions of the driver. It may be possible that additional functionality is available from the card (i.e. viewing the additional analogue inputs that the card presents), but this has not been tested yet. If I get around to this, I'll update the document with whatever I find.

To power up the card, load the following modules in the following order:

- modprobe bttv (normally loaded automatically)
- modprobe dvb-bt8xx (or place dvb-bt8xx in /etc/modules)

Insertion of these modules into the running kernel will activate the appropriate DVB device nodes. It is then possible to start accessing the card with utilities such as scan, tzap, dvbstream etc.

The frontend module sp887x.o, requires an external firmware. Please use the command ”get_dvb_firmware sp887x” to download it. Then copy it to /usr/lib/hotplug/firmware or /lib/firmware/ (depending on configuration of firmware hotplug).

Known Limitations

At present I can say with confidence that the frontend tunes via /dev/dvb/adapter{x}/frontend0 and supplies an MPEG2 stream via /dev/dvb/adapter{x}/dvr0. I have not tested the functionality of any other part of the card yet. I will do so over time and update this document.

There are some limitations in the i2c layer due to a returned error message inconsistency. Although this generates errors in dmesg and the system logs, it does not appear to affect the ability of the frontend to function correctly.

Further Update

dvbstream and VideoLAN Client on windows works a treat with DVB, in fact this is currently serving as my main way of viewing DVB-T at the moment. Additionally, VLC is happily decoding HDTV signals, although the PC is dropping the odd frame here and there - I assume due to processing capability - as all the decoding is being done under windows in software.

Many thanks to Nigel Pearson for the updates to this document since the recent revision of the driver.

59.8.2 How to get the bt8xx cards working

Authors:
Richard Walker, Jamie Honan, Michael Hunold, Manu Abraham, Uwe Bugla, Michael Krufky

General information

This class of cards has a bt878a as the PCI interface, and require the bttv driver for accessing the i2c bus and the gpio pins of the bt8xx chipset.

Please see *BTTV cards list* for a complete list of Cards based on the Conexant Bt8xx PCI bridge supported by the Linux Kernel.

In order to be able to compile the kernel, some config options should be enabled:

```
./scripts/config -e PCI
./scripts/config -e INPUT
./scripts/config -m I2C
./scripts/config -m MEDIA_SUPPORT
./scripts/config -e MEDIA_PCI_SUPPORT
./scripts/config -e MEDIA_ANALOG_TV_SUPPORT
./scripts/config -e MEDIA_DIGITAL_TV_SUPPORT
./scripts/config -e MEDIA_RADIO_SUPPORT
./scripts/config -e RC_CORE
./scripts/config -m VIDEO_BT848
./scripts/config -m DVB_BT8XX
```

If you want to automatically support all possible variants of the Bt8xx cards, you should also do:

```
./scripts/config -e MEDIA_SUBDRV_AUTOSELECT
```

Note: Please use the following options with care as deselection of drivers which are in fact necessary may result in DVB devices that cannot be tuned due to lack of driver support.

If your goal is to just support an specific board, you may, instead, disable MEDIA_SUBDRV_AUTOSELECT and manually select the frontend drivers required by your board. With that, you can save some RAM.

You can do that by calling make xconfig/qconfig/menuconfig and look at the options on those menu options (only enabled if `Autoselect ancillary drivers` is disabled:

1) `Device drivers` => `Multimedia support` => `Customize TV tuners`
2) `Device drivers` => `Multimedia support` => `Customize DVB frontends`

Then, on each of the above menu, please select your card-specific frontend and tuner modules.

Loading Modules

Regular case: If the bttv driver detects a bt8xx-based DVB card, all frontend and backend modules will be loaded automatically.

Exceptions are:

- Old TV cards without EEPROMs, sharing a common PCI subsystem ID;
- Old TwinHan DST cards or clones with or without CA slot and not containing an Eeprom.

In the following cases overriding the PCI type detection for bttv and for dvb-bt8xx drivers by passing modprobe parameters may be necessary.

Running TwinHan and Clones

As shown at *BTTV cards list*, TwinHan and clones use `card=113` modprobe parameter. So, in order to properly detect it for devices without EEPROM, you should use:

```
$ modprobe bttv card=113
$ modprobe dst
```

Useful parameters for verbosity level and debugging the dst module:

```
verbose=0:              messages are disabled
        1:              only error messages are displayed
        2:              notifications are displayed
        3:              other useful messages are displayed
        4:              debug setting
dst_addons=0:           card is a free to air (FTA) card only
        0x20:   card has a conditional access slot for scrambled channels
dst_algo=0:             (default) Software tuning algorithm
         1:             Hardware tuning algorithm
```

The autodetected values are determined by the cards' ”response string”.

In your logs see f. ex.: dst_get_device_id: Recognize [DSTMCI].

For bug reports please send in a complete log with verbose=4 activated. Please also see *Digital TV Conditional Access Interface*.

Running multiple cards

See *BTTV cards list* for a complete list of Card ID. Some examples:

Brand name	ID
Pinnacle PCTV Sat	94
Nebula Electronics Digi TV	104
pcHDTV HD-2000 TV	112
Twinhan DST and clones	113
Avermedia AverTV DVB-T 77:	123
Avermedia AverTV DVB-T 761	124
DViCO FusionHDTV DVB-T Lite	128
DViCO FusionHDTV 5 Lite	135

Note: When you have multiple cards, the order of the card ID should match the order where they're detected by the system. Please notice that removing/inserting other PCI cards may change the detection order.

Example:

```
$ modprobe bttv card=113 card=135
```

In case of further problems please subscribe and send questions to the mailing list: linux-media@vger.kernel.org.

Probing the cards with broken PCI subsystem ID

There are some TwinHan cards whose EEPROM has become corrupted for some reason. The cards do not have a correct PCI subsystem ID. Still, it is possible to force probing the cards with:

```
$ echo 109e 0878 $subvendor $subdevice > \
        /sys/bus/pci/drivers/bt878/new_id
```

The two numbers there are:

```
109e: PCI_VENDOR_ID_BROOKTREE
0878: PCI_DEVICE_ID_BROOKTREE_878
```

59.8.3 Firmware files for lmedm04 cards

To extract firmware for the DM04/QQBOX you need to copy the following file(s) to this directory.

For DM04+/QQBOX LME2510C (Sharp 7395 Tuner)

The Sharp 7395 driver can be found in windows/system32/drivers

US2A0D.sys (dated 17 Mar 2009)

and run:

```
scripts/get_dvb_firmware lme2510c_s7395
```

will produce dvb-usb-lme2510c-s7395.fw

An alternative but older firmware can be found on the driver disk DVB-S_EN_3.5A in BDADriver/driver

LMEBDA_DVBS7395C.sys (dated 18 Jan 2008)

and run:

```
./get_dvb_firmware lme2510c_s7395_old
```

will produce dvb-usb-lme2510c-s7395.fw

The LG firmware can be found on the driver disk DM04+_5.1A[LG] in BDADriver/driver

For DM04 LME2510 (LG Tuner)

LMEBDA_DVBS.sys (dated 13 Nov 2007)

and run:

```
./get_dvb_firmware lme2510_lg
```

will produce dvb-usb-lme2510-lg.fw

Other LG firmware can be extracted manually from US280D.sys only found in windows/system32/drivers

dd if=US280D.sys ibs=1 skip=42360 count=3924 of=dvb-usb-lme2510-lg.fw

For DM04 LME2510C (LG Tuner)

```
dd if=US280D.sys ibs=1 skip=35200 count=3850 of=dvb-usb-lme2510c-lg.fw
```

The Sharp 0194 tuner driver can be found in windows/system32/drivers

US290D.sys (dated 09 Apr 2009)

For LME2510

```
dd if=US290D.sys ibs=1 skip=36856 count=3976 of=dvb-usb-lme2510-s0194.fw
```

For LME2510C

```
dd if=US290D.sys ibs=1 skip=33152 count=3697 of=dvb-usb-lme2510c-s0194.fw
```

The m88rs2000 tuner driver can be found in windows/system32/drivers

US2B0D.sys (dated 29 Jun 2010)

```
dd if=US2B0D.sys ibs=1 skip=34432 count=3871 of=dvb-usb-lme2510c-rs2000.fw
```

We need to modify id of rs2000 firmware or it will warm boot id 3344:1120.

```
echo -ne \\xF0\\x22 | dd conv=notrunc bs=1 count=2 seek=266 of=dvb-usb-
↪lme2510c-rs2000.fw
```

Copy the firmware file(s) to /lib/firmware

59.8.4 Opera firmware

Author: Marco Gittler <g.marco@freenet.de>

To extract the firmware for the Opera DVB-S1 USB-Box you need to copy the files:

2830SCap2.sys 2830SLoad2.sys

from the windriver disk into this directory.

Then run:

```
scripts/get_dvb_firmware opera1
```

and after that you have 2 files:

dvb-usb-opera-01.fw dvb-usb-opera1-fpga-01.fw

in here.

Copy them into /lib/firmware/ .

After that the driver can load the firmware (if you have enabled firmware loading in kernel config and have hotplug running).

59.8.5 How to set up the Technisat/B2C2 Flexcop devices

Note: This documentation is outdated.

Author: Uwe Bugla <uwe.bugla@gmx.de> August 2009

Find out what device you have

Important Notice: The driver does NOT support Technisat USB 2 devices!

First start your linux box with a shipped kernel:

```
lspci -vvv for a PCI device (lsusb -vvv for an USB device) will show you for␣
↪example:
02:0b.0 Network controller: Techsan Electronics Co Ltd B2C2 FlexCopII DVB chip␣
↪/
Technisat SkyStar2 DVB card (rev 02)

dmesg | grep frontend may show you for example:
DVB: registering frontend 0 (Conexant CX24123/CX24109)...
```

Kernel compilation:

If the Flexcop / Technisat is the only DVB / TV / Radio device in your box get rid of unnecessary modules and check this one:

`Multimedia support => Customise analog and hybrid tuner modules to build`

In this directory uncheck every driver which is activated there (except `Simple tuner support` for ATSC 3rd generation only -> see case 9 please).

Then please activate:

- Main module part:

  `Multimedia support => DVB/ATSC adapters => Technisat/B2C2 FlexcopII(b) and FlexCopIII adapters`

  1) => `Technisat/B2C2 Air/Sky/Cable2PC PCI` (PCI card) or
  2) => `Technisat/B2C2 Air/Sky/Cable2PC USB` (USB 1.1 adapter) and for troubleshooting purposes:
  3) => `Enable debug for the B2C2 FlexCop drivers`

- Frontend / Tuner / Demodulator module part:

  `Multimedia support => DVB/ATSC adapters`
  => `Customise the frontend modules to build` `Customise DVB frontends =>`

  - SkyStar DVB-S Revision 2.3:

    1) => `Zarlink VP310/MT312/ZL10313 based`
    2) => `Generic I2C PLL based tuners`

- SkyStar DVB-S Revision 2.6:
  1) => `ST STV0299 based`
  2) => `Generic I2C PLL based tuners`
- SkyStar DVB-S Revision 2.7:
  1) => `Samsung S5H1420 based`
  2) => `Integrant ITD1000 Zero IF tuner for DVB-S/DSS`
  3) => `ISL6421 SEC controller`
- SkyStar DVB-S Revision 2.8:
  1) => `Conexant CX24123 based`
  2) => `Conexant CX24113/CX24128 tuner for DVB-S/DSS`
  3) => `ISL6421 SEC controller`
- AirStar DVB-T card:
  1) => `Zarlink MT352 based`
  2) => `Generic I2C PLL based tuners`
- CableStar DVB-C card:
  1) => `ST STV0297 based`
  2) => `Generic I2C PLL based tuners`
- AirStar ATSC card 1st generation:
  1) => `Broadcom BCM3510`
- AirStar ATSC card 2nd generation:
  1) => `NxtWave Communications NXT2002/NXT2004 based`
  2) => `Generic I2C PLL based tuners`
- AirStar ATSC card 3rd generation:
  1) => `LG Electronics LGDT3302/LGDT3303 based`
  2) `Multimedia support => Customise analog and hybrid tuner modules to build => Simple tuner support`

59.8.6 TechnoTrend/Hauppauge DEC USB Driver

Driver Status

Supported:

- DEC2000-t
- DEC2450-t
- DEC3000-s
- Video Streaming

- Audio Streaming
- Section Filters
- Channel Zapping
- Hotplug firmware loader

To Do:

- Tuner status information
- DVB network interface
- Streaming video PC->DEC
- Conax support for 2450-t

Getting the Firmware

To download the firmware, use the following commands:

```
scripts/get_dvb_firmware dec2000t
scripts/get_dvb_firmware dec2540t
scripts/get_dvb_firmware dec3000s
```

Hotplug Firmware Loading

Since 2.6 kernels, the firmware is loaded at the point that the driver module is loaded.

Copy the three files downloaded above into the /usr/lib/hotplug/firmware or /lib/firmware directory (depending on configuration of firmware hotplug).

Copyright © 1999-2020 : LinuxTV Developers

```
This documentation is free software; you can redistribute it and/or modify it
under the terms of the GNU General Public License as published by the Free
Software Foundation; either version 2 of the License, or (at your option) any
later version.

This program is distributed in the hope that it will be useful, but WITHOUT
ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or
FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for
more details.

For more details see the file COPYING in the source distribution of Linux.
```

CHAPTER

SIXTY

MEMORY MANAGEMENT

Linux memory management subsystem is responsible, as the name implies, for managing the memory in the system. This includes implementation of virtual memory and demand paging, memory allocation both for kernel internal structures and user space programs, mapping of files into processes address space and many other cool things.

Linux memory management is a complex system with many configurable settings. Most of these settings are available via `/proc` filesystem and can be quired and adjusted using `sysctl`. These APIs are described in *Documentation for /proc/sys/vm/* and in man 5 proc.

Linux memory management has its own jargon and if you are not yet familiar with it, consider reading *Concepts overview*.

Here we document in detail how to interact with various mechanisms in the Linux memory management.

60.1 Concepts overview

The memory management in Linux is a complex system that evolved over the years and included more and more functionality to support a variety of systems from MMU-less microcontrollers to supercomputers. The memory management for systems without an MMU is called `nommu` and it definitely deserves a dedicated document, which hopefully will be eventually written. Yet, although some of the concepts are the same, here we assume that an MMU is available and a CPU can translate a virtual address to a physical address.

- *Virtual Memory Primer*
- *Huge Pages*
- *Zones*
- *Nodes*
- *Page cache*
- *Anonymous Memory*
- *Reclaim*
- *Compaction*
- *OOM killer*

60.1.1 Virtual Memory Primer

The physical memory in a computer system is a limited resource and even for systems that support memory hotplug there is a hard limit on the amount of memory that can be installed. The physical memory is not necessarily contiguous; it might be accessible as a set of distinct address ranges. Besides, different CPU architectures, and even different implementations of the same architecture have different views of how these address ranges are defined.

All this makes dealing directly with physical memory quite complex and to avoid this complexity a concept of virtual memory was developed.

The virtual memory abstracts the details of physical memory from the application software, allows to keep only needed information in the physical memory (demand paging) and provides a mechanism for the protection and controlled sharing of data between processes.

With virtual memory, each and every memory access uses a virtual address. When the CPU decodes an instruction that reads (or writes) from (or to) the system memory, it translates the *virtual* address encoded in that instruction to a *physical* address that the memory controller can understand.

The physical system memory is divided into page frames, or pages. The size of each page is architecture specific. Some architectures allow selection of the page size from several supported values; this selection is performed at the kernel build time by setting an appropriate kernel configuration option.

Each physical memory page can be mapped as one or more virtual pages. These mappings are described by page tables that allow translation from a virtual address used by programs to the physical memory address. The page tables are organized hierarchically.

The tables at the lowest level of the hierarchy contain physical addresses of actual pages used by the software. The tables at higher levels contain physical addresses of the pages belonging to the lower levels. The pointer to the top level page table resides in a register. When the CPU performs the address translation, it uses this register to access the top level page table. The high bits of the virtual address are used to index an entry in the top level page table. That entry is then used to access the next level in the hierarchy with the next bits of the virtual address as the index to that level page table. The lowest bits in the virtual address define the offset inside the actual page.

60.1.2 Huge Pages

The address translation requires several memory accesses and memory accesses are slow relatively to CPU speed. To avoid spending precious processor cycles on the address translation, CPUs maintain a cache of such translations called Translation Lookaside Buffer (or TLB). Usually TLB is pretty scarce resource and applications with large memory working set will experience performance hit because of TLB misses.

Many modern CPU architectures allow mapping of the memory pages directly by the higher levels in the page table. For instance, on x86, it is possible to map 2M and even 1G pages using entries in the second and the third level page tables. In Linux such pages are called *huge*. Usage of huge pages significantly reduces pressure on TLB, improves TLB hit-rate and thus improves overall system performance.

There are two mechanisms in Linux that enable mapping of the physical memory with the huge pages. The first one is *HugeTLB filesystem*, or hugetlbfs. It is a pseudo filesystem that uses

RAM as its backing store. For the files created in this filesystem the data resides in the memory and mapped using huge pages. The hugetlbfs is described at *HugeTLB Pages*.

Another, more recent, mechanism that enables use of the huge pages is called *Transparent HugePages*, or THP. Unlike the hugetlbfs that requires users and/or system administrators to configure what parts of the system memory should and can be mapped by the huge pages, THP manages such mappings transparently to the user and hence the name. See *Transparent Hugepage Support* for more details about THP.

60.1.3 Zones

Often hardware poses restrictions on how different physical memory ranges can be accessed. In some cases, devices cannot perform DMA to all the addressable memory. In other cases, the size of the physical memory exceeds the maximal addressable size of virtual memory and special actions are required to access portions of the memory. Linux groups memory pages into *zones* according to their possible usage. For example, ZONE_DMA will contain memory that can be used by devices for DMA, ZONE_HIGHMEM will contain memory that is not permanently mapped into kernel's address space and ZONE_NORMAL will contain normally addressed pages.

The actual layout of the memory zones is hardware dependent as not all architectures define all zones, and requirements for DMA are different for different platforms.

60.1.4 Nodes

Many multi-processor machines are NUMA - Non-Uniform Memory Access - systems. In such systems the memory is arranged into banks that have different access latency depending on the ”distance” from the processor. Each bank is referred to as a *node* and for each node Linux constructs an independent memory management subsystem. A node has its own set of zones, lists of free and used pages and various statistics counters. You can find more details about NUMA in Documentation/mm/numa.rst` and in *NUMA Memory Policy*.

60.1.5 Page cache

The physical memory is volatile and the common case for getting data into the memory is to read it from files. Whenever a file is read, the data is put into the *page cache* to avoid expensive disk access on the subsequent reads. Similarly, when one writes to a file, the data is placed in the page cache and eventually gets into the backing storage device. The written pages are marked as *dirty* and when Linux decides to reuse them for other purposes, it makes sure to synchronize the file contents on the device with the updated data.

60.1.6 Anonymous Memory

The *anonymous memory* or *anonymous mappings* represent memory that is not backed by a filesystem. Such mappings are implicitly created for program's stack and heap or by explicit calls to mmap(2) system call. Usually, the anonymous mappings only define virtual memory areas that the program is allowed to access. The read accesses will result in creation of a page table entry that references a special physical page filled with zeroes. When the program performs a write, a regular physical page will be allocated to hold the written data. The page will be marked dirty and if the kernel decides to repurpose it, the dirty page will be swapped out.

60.1.7 Reclaim

Throughout the system lifetime, a physical page can be used for storing different types of data. It can be kernel internal data structures, DMA'able buffers for device drivers use, data read from a filesystem, memory allocated by user space processes etc.

Depending on the page usage it is treated differently by the Linux memory management. The pages that can be freed at any time, either because they cache the data available elsewhere, for instance, on a hard disk, or because they can be swapped out, again, to the hard disk, are called *reclaimable*. The most notable categories of the reclaimable pages are page cache and anonymous memory.

In most cases, the pages holding internal kernel data and used as DMA buffers cannot be repurposed, and they remain pinned until freed by their user. Such pages are called *unreclaimable*. However, in certain circumstances, even pages occupied with kernel data structures can be reclaimed. For instance, in-memory caches of filesystem metadata can be re-read from the storage device and therefore it is possible to discard them from the main memory when system is under memory pressure.

The process of freeing the reclaimable physical memory pages and repurposing them is called (surprise!) *reclaim*. Linux can reclaim pages either asynchronously or synchronously, depending on the state of the system. When the system is not loaded, most of the memory is free and allocation requests will be satisfied immediately from the free pages supply. As the load increases, the amount of the free pages goes down and when it reaches a certain threshold (low watermark), an allocation request will awaken the `kswapd` daemon. It will asynchronously scan memory pages and either just free them if the data they contain is available elsewhere, or evict to the backing storage device (remember those dirty pages?). As memory usage increases even more and reaches another threshold - min watermark - an allocation will trigger *direct reclaim*. In this case allocation is stalled until enough memory pages are reclaimed to satisfy the request.

60.1.8 Compaction

As the system runs, tasks allocate and free the memory and it becomes fragmented. Although with virtual memory it is possible to present scattered physical pages as virtually contiguous range, sometimes it is necessary to allocate large physically contiguous memory areas. Such need may arise, for instance, when a device driver requires a large buffer for DMA, or when THP allocates a huge page. Memory *compaction* addresses the fragmentation issue. This mechanism moves occupied pages from the lower part of a memory zone to free pages in the upper part of the zone. When a compaction scan is finished free pages are grouped together at the beginning of the zone and allocations of large physically contiguous areas become possible.

Like reclaim, the compaction may happen asynchronously in the `kcompactd` daemon or synchronously as a result of a memory allocation request.

60.1.9 OOM killer

It is possible that on a loaded machine memory will be exhausted and the kernel will be unable to reclaim enough memory to continue to operate. In order to save the rest of the system, it invokes the *OOM killer*.

The *OOM killer* selects a task to sacrifice for the sake of the overall system health. The selected task is killed in a hope that after it exits enough memory will be freed to continue normal operation.

60.2 CMA Debugfs Interface

The CMA debugfs interface is useful to retrieve basic information out of the different CMA areas and to test allocation/release in each of the areas.

Each CMA area represents a directory under <debugfs>/cma/, represented by its CMA name like below:

<debugfs>/cma/<cma_name>

The structure of the files created under that directory is as follows:

- [RO] base_pfn: The base PFN (Page Frame Number) of the zone.
- [RO] count: Amount of memory in the CMA area.
- [RO] order_per_bit: Order of pages represented by one bit.
- [RO] bitmap: The bitmap of page states in the zone.
- [WO] alloc: Allocate N pages from that CMA area. For example:

```
echo 5 > <debugfs>/cma/<cma_name>/alloc
```

would try to allocate 5 pages from the 'cma_name' area.

- [WO] free: Free N pages from that CMA area, similar to the above.

60.3 DAMON: Data Access MONitor

DAMON allows light-weight data access monitoring. Using DAMON, users can analyze the memory access patterns of their systems and optimize those.

60.3.1 Getting Started

This document briefly describes how you can use DAMON by demonstrating its default user space tool. Please note that this document describes only a part of its features for brevity. Please refer to the usage doc of the tool for more details.

Prerequisites

Kernel

You should first ensure your system is running on a kernel built with `CONFIG_DAMON_*=y`.

User Space Tool

For the demonstration, we will use the default user space tool for DAMON, called DAMON Operator (DAMO). It is available at https://github.com/awslabs/damo. The examples below assume that `damo` is on your `$PATH`. It's not mandatory, though.

Because DAMO is using the sysfs interface (refer to *Detailed Usages* for the detail) of DAMON, you should ensure sysfs is mounted.

Recording Data Access Patterns

The commands below record the memory access patterns of a program and save the monitoring results to a file.

```
$ git clone https://github.com/sjp38/masim
$ cd masim; make; ./masim ./configs/zigzag.cfg &
$ sudo damo record -o damon.data $(pidof masim)
```

The first two lines of the commands download an artificial memory access generator program and run it in the background. The generator will repeatedly access two 100 MiB sized memory regions one by one. You can substitute this with your real workload. The last line asks `damo` to record the access pattern in the `damon.data` file.

Visualizing Recorded Patterns

You can visualize the pattern in a heatmap, showing which memory region (x-axis) got accessed when (y-axis) and how frequently (number).:

```
$ sudo damo report heats --heatmap stdout
22222222222222222222222222222222222222111111111111111111111111111111111111111100
44444444444444444444444444444444444444344444444444444444444444444444444444443200
44444444444444444444444444444444444444334444444444444444444444444444444444444200
33333333333333333333333333333333333333445555555555555555555555555555555555555200
33333333333333333333333333333333333333444444444444444444444444444444444444444200
22222222222222222222222222222222222222335555555555555555555555555555555555555200
00000000000000000000000000000000000000288888888888888888888888888888888888888400
00000000000000000000000000000000000000288888888888888888888888888888888888888400
33333333333333333333333333333333333333355555555555555555555555555555555555555200
88888888888888888888888888888888888888600000000000000000000000000000000000000000
88888888888888888888888888888888888888600000000000000000000000000000000000000000
33333333333333333333333333333333333333444444444444444444444444444444444444443200
00000000000000000000000000000000000000288888888888888888888888888888888888888400
[...]
# access_frequency:  0  1  2  3  4  5  6  7  8  9
# x-axis: space (139728247021568-139728453431248: 196.848 MiB)
# y-axis: time (15256597248362-15326899978162: 1 m 10.303 s)
# resolution: 80x40 (2.461 MiB and 1.758 s for each character)
```

You can also visualize the distribution of the working set size, sorted by the size.:

```
$ sudo damo report wss --range 0 101 10
# <percentile> <wss>
# target_id     18446632103789443072
# avr:  107.708 MiB
  0             0 B |                                                          ␣
 ↪ |
 10      95.328 MiB |****************************                              ␣
 ↪ |
 20      95.332 MiB |****************************                              ␣
 ↪ |
 30      95.340 MiB |****************************                              ␣
 ↪ |
 40      95.387 MiB |****************************                              ␣
 ↪ |
 50      95.387 MiB |****************************                              ␣
 ↪ |
 60      95.398 MiB |****************************                              ␣
 ↪ |
 70      95.398 MiB |****************************                              ␣
 ↪ |
 80      95.504 MiB |****************************                              ␣
 ↪ |
 90     190.703 MiB |************************************************************␣
 ↪ |
100     196.875 MiB␣
 ↪|************************************************************|
```

Using `--sortby` option with the above command, you can show how the working set size has chronologically changed.:

```
$ sudo damo report wss --range 0 101 10 --sortby time
# <percentile> <wss>
# target_id     18446632103789443072
# avr:  107.708 MiB
  0       3.051 MiB |                                                        ␣
↪ |
 10     190.703 MiB␣
↪|*************************************************************|
 20      95.336 MiB |******************************                          ␣
↪ |
 30      95.328 MiB |******************************                          ␣
↪ |
 40      95.387 MiB |******************************                          ␣
↪ |
 50      95.332 MiB |******************************                          ␣
↪ |
 60      95.320 MiB |******************************                          ␣
↪ |
 70      95.398 MiB |******************************                          ␣
↪ |
 80      95.398 MiB |******************************                          ␣
↪ |
 90      95.340 MiB |******************************                          ␣
↪ |
100      95.398 MiB |******************************                          ␣
↪ |
```

Data Access Pattern Aware Memory Management

Below command makes every memory region of size >=4K that has not accessed for >=60 seconds in your workload to be swapped out.

```
$ sudo damo schemes --damos_access_rate 0 0 --damos_sz_region 4K max \
                    --damos_age 60s max --damos_action pageout \
                    <pid of your workload>
```

60.3.2 Detailed Usages

DAMON provides below interfaces for different users.

- *DAMON user space tool.* This is for privileged people such as system administrators who want a just-working human-friendly interface. Using this, users can use the DAMON's major features in a human-friendly way. It may not be highly tuned for special cases, though. For more detail, please refer to its usage document.
- *sysfs interface.* *This* is for privileged user space programmers who want more optimized use of DAMON. Using this, users can use DAMON's major features by reading from and writing to special sysfs files. Therefore, you can write and use your personalized DAMON sysfs wrapper programs that reads/writes the sysfs files instead of you. The DAMON user space tool is one example of such programs.
- *Kernel Space Programming Interface.* This is for kernel space programmers. Using this, users can utilize every feature of DAMON most flexibly and efficiently by writing kernel space DAMON application programs for you. You can even extend DAMON for various address spaces. For detail, please refer to the interface document.
- *debugfs interface. (DEPRECATED!)* *This* is almost identical to *sysfs interface*. This is deprecated, so users should move to the *sysfs interface*. If you depend on this and cannot move, please report your usecase to damon@lists.linux.dev and linux-mm@kvack.org.

sysfs Interface

DAMON sysfs interface is built when `CONFIG_DAMON_SYSFS` is defined. It creates multiple directories and files under its sysfs directory, `<sysfs>/kernel/mm/damon/`. You can control DAMON by writing to and reading from the files under the directory.

For a short example, users can monitor the virtual address space of a given workload as below.

```
# cd /sys/kernel/mm/damon/admin/
# echo 1 > kdamonds/nr_kdamonds && echo 1 > kdamonds/0/contexts/nr_contexts
# echo vaddr > kdamonds/0/contexts/0/operations
# echo 1 > kdamonds/0/contexts/0/targets/nr_targets
# echo $(pidof <workload>) > kdamonds/0/contexts/0/targets/0/pid_target
# echo on > kdamonds/0/state
```

Files Hierarchy

The files hierarchy of DAMON sysfs interface is shown below. In the below figure, parents-children relations are represented with indentations, each directory is having / suffix, and files in each directory are separated by comma (",").

```
/sys/kernel/mm/damon/admin
| kdamonds/nr_kdamonds
| | 0/state,pid
| | | contexts/nr_contexts
| | | | 0/avail_operations,operations
| | | | | monitoring_attrs/
| | | | | | intervals/sample_us,aggr_us,update_us
```

```
| | | | | | nr_regions/min,max
| | | | | targets/nr_targets
| | | | | | 0/pid_target
| | | | | | | regions/nr_regions
| | | | | | | | 0/start,end
| | | | | | | | ...
| | | | | | ...
| | | | | schemes/nr_schemes
| | | | | | 0/action,apply_interval_us
| | | | | | | access_pattern/
| | | | | | | | sz/min,max
| | | | | | | | nr_accesses/min,max
| | | | | | | | age/min,max
| | | | | | | quotas/ms,bytes,reset_interval_ms
| | | | | | | | weights/sz_permil,nr_accesses_permil,age_permil
| | | | | | | | goals/nr_goals
| | | | | | | | | 0/target_value,current_value
| | | | | | | watermarks/metric,interval_us,high,mid,low
| | | | | | | filters/nr_filters
| | | | | | | | 0/type,matching,memcg_id
| | | | | | | stats/nr_tried,sz_tried,nr_applied,sz_applied,qt_exceeds
| | | | | | | tried_regions/total_bytes
| | | | | | | | 0/start,end,nr_accesses,age
| | | | | | | | ...
| | | | | | ...
| | | | ...
| | ...
```

Root

The root of the DAMON sysfs interface is `<sysfs>/kernel/mm/damon/`, and it has one directory named `admin`. The directory contains the files for privileged user space programs' control of DAMON. User space tools or daemons having the root permission could use this directory.

kdamonds/

Under the `admin` directory, one directory, `kdamonds`, which has files for controlling the kdamonds (refer to design for more details) exists. In the beginning, this directory has only one file, `nr_kdamonds`. Writing a number (`N`) to the file creates the number of child directories named `0` to `N-1`. Each directory represents each kdamond.

kdamonds/<N>/

In each kdamond directory, two files (`state` and `pid`) and one directory (`contexts`) exist.

Reading `state` returns `on` if the kdamond is currently running, or `off` if it is not running.

Users can write below commands for the kdamond to the `state` file.

- `on`: Start running.
- `off`: Stop running.
- `commit`: Read the user inputs in the sysfs files except `state` file again.
- `commit_schemes_quota_goals`: Read the DAMON-based operation schemes' *quota goals*.
- `update_schemes_stats`: Update the contents of stats files for each DAMON-based operation scheme of the kdamond. For details of the stats, please refer to *stats section*.
- `update_schemes_tried_regions`: Update the DAMON-based operation scheme action tried regions directory for each DAMON-based operation scheme of the kdamond. For details of the DAMON-based operation scheme action tried regions directory, please refer to *tried_regions section*.
- `update_schemes_tried_bytes`: Update only `.../tried_regions/total_bytes` files.
- `clear_schemes_tried_regions`: Clear the DAMON-based operating scheme action tried regions directory for each DAMON-based operation scheme of the kdamond.

If the state is `on`, reading `pid` shows the pid of the kdamond thread.

`contexts` directory contains files for controlling the monitoring contexts that this kdamond will execute.

kdamonds/<N>/contexts/

In the beginning, this directory has only one file, `nr_contexts`. Writing a number (`N`) to the file creates the number of child directories named as `0` to `N-1`. Each directory represents each monitoring context (refer to design for more details). At the moment, only one context per kdamond is supported, so only `0` or `1` can be written to the file.

contexts/<N>/

In each context directory, two files (`avail_operations` and `operations`) and three directories (`monitoring_attrs`, `targets`, and `schemes`) exist.

DAMON supports multiple types of monitoring operations, including those for virtual address space and the physical address space. You can get the list of available monitoring operations set on the currently running kernel by reading `avail_operations` file. Based on the kernel configuration, the file will list some or all of below keywords.

- vaddr: Monitor virtual address spaces of specific processes
- fvaddr: Monitor fixed virtual address ranges
- paddr: Monitor the physical address space of the system

Please refer to *regions sysfs directory* for detailed differences between the operations sets in terms of the monitoring target regions.

You can set and get what type of monitoring operations DAMON will use for the context by writing one of the keywords listed in `avail_operations` file and reading from the `operations` file.

contexts/<N>/monitoring_attrs/

Files for specifying attributes of the monitoring including required quality and efficiency of the monitoring are in `monitoring_attrs` directory. Specifically, two directories, `intervals` and `nr_regions` exist in this directory.

Under `intervals` directory, three files for DAMON's sampling interval (`sample_us`), aggregation interval (`aggr_us`), and update interval (`update_us`) exist. You can set and get the values in micro-seconds by writing to and reading from the files.

Under `nr_regions` directory, two files for the lower-bound and upper-bound of DAMON's monitoring regions (`min` and `max`, respectively), which controls the monitoring overhead, exist. You can set and get the values by writing to and rading from the files.

For more details about the intervals and monitoring regions range, please refer to the Design document (/mm/damon/design).

contexts/<N>/targets/

In the beginning, this directory has only one file, `nr_targets`. Writing a number (`N`) to the file creates the number of child directories named `0` to `N-1`. Each directory represents each monitoring target.

targets/<N>/

In each target directory, one file (`pid_target`) and one directory (`regions`) exist.

If you wrote `vaddr` to the `contexts/<N>/operations`, each target should be a process. You can specify the process to DAMON by writing the pid of the process to the `pid_target` file.

targets/<N>/regions

When `vaddr` monitoring operations set is being used (`vaddr` is written to the `contexts/<N>/operations` file), DAMON automatically sets and updates the monitoring target regions so that entire memory mappings of target processes can be covered. However, users could want to set the initial monitoring region to specific address ranges.

In contrast, DAMON do not automatically sets and updates the monitoring target regions when `fvaddr` or `paddr` monitoring operations sets are being used (`fvaddr` or `paddr` have written to the `contexts/<N>/operations`). Therefore, users should set the monitoring target regions by themselves in the cases.

For such cases, users can explicitly set the initial monitoring target regions as they want, by writing proper values to the files under this directory.

In the beginning, this directory has only one file, `nr_regions`. Writing a number (`N`) to the file creates the number of child directories named `0` to `N-1`. Each directory represents each initial monitoring target region.

regions/<N>/

In each region directory, you will find two files (`start` and `end`). You can set and get the start and end addresses of the initial monitoring target region by writing to and reading from the files, respectively.

Each region should not overlap with others. `end` of directory `N` should be equal or smaller than `start` of directory `N+1`.

contexts/<N>/schemes/

The directory for DAMON-based Operation Schemes (DAMOS). Users can get and set the schemes by reading from and writing to files under this directory.

In the beginning, this directory has only one file, `nr_schemes`. Writing a number (`N`) to the file creates the number of child directories named `0` to `N-1`. Each directory represents each DAMON-based operation scheme.

schemes/<N>/

In each scheme directory, five directories (`access_pattern`, `quotas`, `watermarks`, `filters`, `stats`, and `tried_regions`) and two files (`action` and `apply_interval`) exist.

The `action` file is for setting and getting the scheme's action. The keywords that can be written to and read from the file and their meaning are as below.

Note that support of each action depends on the running DAMON operations set *implementation*.

- `willneed`: Call `madvise()` for the region with `MADV_WILLNEED`. Supported by `vaddr` and `fvaddr` operations set.
- `cold`: Call `madvise()` for the region with `MADV_COLD`. Supported by `vaddr` and `fvaddr` operations set.
- `pageout`: Call `madvise()` for the region with `MADV_PAGEOUT`. Supported by `vaddr`, `fvaddr` and `paddr` operations set.
- `hugepage`: Call `madvise()` for the region with `MADV_HUGEPAGE`. Supported by `vaddr` and `fvaddr` operations set.
- `nohugepage`: Call `madvise()` for the region with `MADV_NOHUGEPAGE`. Supported by `vaddr` and `fvaddr` operations set.
- `lru_prio`: Prioritize the region on its LRU lists. Supported by `paddr` operations set.
- `lru_deprio`: Deprioritize the region on its LRU lists. Supported by `paddr` operations set.
- `stat`: Do nothing but count the statistics. Supported by all operations sets.

The `apply_interval_us` file is for setting and getting the scheme's apply_interval in microseconds.

schemes/<N>/access_pattern/

The directory for the target access pattern of the given DAMON-based operation scheme.

Under the `access_pattern` directory, three directories (`sz`, `nr_accesses`, and `age`) each having two files (`min` and `max`) exist. You can set and get the access pattern for the given scheme by writing to and reading from the `min` and `max` files under `sz`, `nr_accesses`, and `age` directories, respectively. Note that the `min` and the `max` form a closed interval.

schemes/<N>/quotas/

The directory for the quotas of the given DAMON-based operation scheme.

Under `quotas` directory, three files (`ms`, `bytes`, `reset_interval_ms`) and two directores (`weights` and `goals`) exist.

You can set the `time quota` in milliseconds, `size quota` in bytes, and `reset interval` in milliseconds by writing the values to the three files, respectively. Then, DAMON tries to use only up to `time quota` milliseconds for applying the `action` to memory regions of the `access_pattern`, and to apply the action to only up to `bytes` bytes of memory regions within the `reset_interval_ms`. Setting both `ms` and `bytes` zero disables the quota limits.

Under `weights` directory, three files (`sz_permil`, `nr_accesses_permil`, and `age_permil`) exist. You can set the prioritization weights for size, access frequency, and age in per-thousand unit by writing the values to the three files under the `weights` directory.

schemes/<N>/quotas/goals/

The directory for the automatic quota tuning goals of the given DAMON-based operation scheme.

In the beginning, this directory has only one file, `nr_goals`. Writing a number (`N`) to the file creates the number of child directories named `0` to `N-1`. Each directory represents each goal and current achievement. Among the multiple feedback, the best one is used.

Each goal directory contains two files, namely `target_value` and `current_value`. Users can set and get any number to those files to set the feedback. User space main workload's latency or throughput, system metrics like free memory ratio or memory pressure stall time (PSI) could be example metrics for the values. Note that users should write `commit_schemes_quota_goals` to the `state` file of the *kdamond directory* to pass the feedback to DAMON.

schemes/<N>/watermarks/

The directory for the watermarks of the given DAMON-based operation scheme.

Under the watermarks directory, five files (`metric`, `interval_us`, `high`, `mid`, and `low`) for setting the metric, the time interval between check of the metric, and the three watermarks exist. You can set and get the five values by writing to the files, respectively.

Keywords and meanings of those that can be written to the `metric` file are as below.

- none: Ignore the watermarks
- free_mem_rate: System's free memory rate (per thousand)

The `interval` should written in microseconds unit.

schemes/<N>/filters/

The directory for the filters of the given DAMON-based operation scheme.

In the beginning, this directory has only one file, `nr_filters`. Writing a number (`N`) to the file creates the number of child directories named `0` to `N-1`. Each directory represents each filter. The filters are evaluated in the numeric order.

Each filter directory contains six files, namely `type`, `matcing`, `memcg_path`, `addr_start`, `addr_end`, and `target_idx`. To `type` file, you can write one of four special keywords: `anon` for anonymous pages, `memcg` for specific memory cgroup, `addr` for specific address range (an open-ended interval), or `target` for specific DAMON monitoring target filtering. In case of the memory cgroup filtering, you can specify the memory cgroup of the interest by writing the path of the memory cgroup from the cgroups mount point to `memcg_path` file. In case of the address range filtering, you can specify the start and end address of the range to `addr_start` and `addr_end` files, respectively. For the DAMON monitoring target filtering, you can specify the index of the target between the list of the DAMON context's monitoring targets list to `target_idx` file. You can write `Y` or `N` to `matching` file to filter out pages that does or does not match to the type, respectively. Then, the scheme's action will not be applied to the pages that specified to be filtered out.

For example, below restricts a DAMOS action to be applied to only non-anonymous pages of all memory cgroups except `/having_care_already`.:

```
# echo 2 > nr_filters
# # filter out anonymous pages
echo anon > 0/type
echo Y > 0/matching
# # further filter out all cgroups except one at '/having_care_already'
echo memcg > 1/type
echo /having_care_already > 1/memcg_path
echo N > 1/matching
```

Note that `anon` and `memcg` filters are currently supported only when `paddr` *implementation* is being used.

Also, memory regions that are filtered out by `addr` or `target` filters are not counted as the scheme has tried to those, while regions that filtered out by other type filters are counted as the scheme has tried to. The difference is applied to *stats* and *tried regions*.

schemes/<N>/stats/

DAMON counts the total number and bytes of regions that each scheme is tried to be applied, the two numbers for the regions that each scheme is successfully applied, and the total number of the quota limit exceeds. This statistics can be used for online analysis or tuning of the schemes.

The statistics can be retrieved by reading the files under `stats` directory (`nr_tried`, `sz_tried`, `nr_applied`, `sz_applied`, and `qt_exceeds`), respectively. The files are not updated in real time, so you should ask DAMON sysfs interface to update the content of the files for the stats by writing a special keyword, `update_schemes_stats` to the relevant `kdamonds/<N>/state` file.

schemes/<N>/tried_regions/

This directory initially has one file, `total_bytes`.

When a special keyword, `update_schemes_tried_regions`, is written to the relevant `kdamonds/<N>/state` file, DAMON updates the `total_bytes` file so that reading it returns the total size of the scheme tried regions, and creates directories named integer starting from `0` under this directory. Each directory contains files exposing detailed information about each of the memory region that the corresponding scheme's `action` has tried to be applied under this directory, during next apply interval of the corresponding scheme. The information includes address range, `nr_accesses`, and `age` of the region.

Writing `update_schemes_tried_bytes` to the relevant `kdamonds/<N>/state` file will only update the `total_bytes` file, and will not create the subdirectories.

The directories will be removed when another special keyword, `clear_schemes_tried_regions`, is written to the relevant `kdamonds/<N>/state` file.

The expected usage of this directory is investigations of schemes' behaviors, and query-like efficient data access monitoring results retrievals. For the latter use case, in particular, users can set the `action` as `stat` and set the `access pattern` as their interested pattern that they want to query.

tried_regions/<N>/

In each region directory, you will find four files (`start`, `end`, `nr_accesses`, and `age`). Reading the files will show the start and end addresses, `nr_accesses`, and `age` of the region that corresponding DAMON-based operation scheme `action` has tried to be applied.

Example

Below commands applies a scheme saying "If a memory region of size in [4KiB, 8KiB] is showing accesses per aggregate interval in [0, 5] for aggregate interval in [10, 20], page out the region. For the paging out, use only up to 10ms per second, and also don't page out more than 1GiB per second. Under the limitation, page out memory regions having longer age first. Also, check the free memory rate of the system every 5 seconds, start the monitoring and paging out when the free memory rate becomes lower than 50%, but stop it if the free memory rate becomes larger than 60%, or lower than 30%".

```
# cd <sysfs>/kernel/mm/damon/admin
# # populate directories
# echo 1 > kdamonds/nr_kdamonds; echo 1 > kdamonds/0/contexts/nr_contexts;
# echo 1 > kdamonds/0/contexts/0/schemes/nr_schemes
# cd kdamonds/0/contexts/0/schemes/0
# # set the basic access pattern and the action
# echo 4096 > access_pattern/sz/min
# echo 8192 > access_pattern/sz/max
# echo 0 > access_pattern/nr_accesses/min
# echo 5 > access_pattern/nr_accesses/max
# echo 10 > access_pattern/age/min
# echo 20 > access_pattern/age/max
# echo pageout > action
# # set quotas
# echo 10 > quotas/ms
# echo $((1024*1024*1024)) > quotas/bytes
# echo 1000 > quotas/reset_interval_ms
# # set watermark
# echo free_mem_rate > watermarks/metric
# echo 5000000 > watermarks/interval_us
# echo 600 > watermarks/high
# echo 500 > watermarks/mid
# echo 300 > watermarks/low
```

Please note that it's highly recommended to use user space tools like damo rather than manually reading and writing the files as above. Above is only for an example.

Tracepoints for Monitoring Results

Users can get the monitoring results via the *tried_regions*. The interface is useful for getting a snapshot, but it could be inefficient for fully recording all the monitoring results. For the purpose, two trace points, namely `damon:damon_aggregated` and `damon:damos_before_apply`, are provided. `damon:damon_aggregated` provides the whole monitoring results, while `damon:damos_before_apply` provides the monitoring results for regions that each DAMON-based Operation Scheme (DAMOS) is gonna be applied. Hence, `damon:damos_before_apply` is more useful for recording internal behavior of DAMOS, or DAMOS target access pattern based query-like efficient monitoring results recording.

While the monitoring is turned on, you could record the tracepoint events and show results using tracepoint supporting tools like `perf`. For example:

```
# echo on > monitor_on
# perf record -e damon:damon_aggregated &
# sleep 5
# kill 9 $(pidof perf)
# echo off > monitor_on
# perf script
kdamond.0 46568 [027] 79357.842179: damon:damon_aggregated: target_id=0 nr_
↪regions=11 122509119488-135708762112: 0 864
[...]
```

Each line of the perf script output represents each monitoring region. The first five fields are as usual other tracepoint outputs. The sixth field (`target_id=X`) shows the ide of the monitoring target of the region. The seventh field (`nr_regions=X`) shows the total number of monitoring regions for the target. The eighth field (`X-Y:`) shows the start (`X`) and end (`Y`) addresses of the region in bytes. The ninth field (`X`) shows the `nr_accesses` of the region (refer to design for more details of the counter). Finally the tenth field (`X`) shows the `age` of the region (refer to design for more details of the counter).

If the event was `damon:damos_beofre_apply`, the `perf script` output would be somewhat like below:

```
kdamond.0 47293 [000] 80801.060214: damon:damos_before_apply: ctx_idx=0 scheme_
↪idx=0 target_idx=0 nr_regions=11 121932607488-135128711168: 0 136
[...]
```

Each line of the output represents each monitoring region that each DAMON-based Operation Scheme was about to be applied at the traced time. The first five fields are as usual. It shows the index of the DAMON context (`ctx_idx=X`) of the scheme in the list of the contexts of the context's kdamond, the index of the scheme (`scheme_idx=X`) in the list of the schemes of the context, in addition to the output of `damon_aggregated` tracepoint.

debugfs Interface (DEPRECATED!)

Note: THIS IS DEPRECATED!

DAMON debugfs interface is deprecated, so users should move to the *sysfs interface*. If you depend on this and cannot move, please report your usecase to damon@lists.linux.dev and linux-mm@kvack.org.

DAMON exports eight files, `attrs`, `target_ids`, `init_regions`, `schemes`, `monitor_on`, `kdamond_pid`, `mk_contexts` and `rm_contexts` under its debugfs directory, `<debugfs>/damon/`.

Attributes

Users can get and set the `sampling interval`, `aggregation interval`, `update interval`, and min/max number of monitoring target regions by reading from and writing to the `attrs` file. To know about the monitoring attributes in detail, please refer to the /mm/damon/design. For example, below commands set those values to 5 ms, 100 ms, 1,000 ms, 10 and 1000, and then check it again:

```
# cd <debugfs>/damon
# echo 5000 100000 1000000 10 1000 > attrs
# cat attrs
5000 100000 1000000 10 1000
```

Target IDs

Some types of address spaces supports multiple monitoring target. For example, the virtual memory address spaces monitoring can have multiple processes as the monitoring targets. Users can set the targets by writing relevant id values of the targets to, and get the ids of the current targets by reading from the `target_ids` file. In case of the virtual address spaces monitoring, the values should be pids of the monitoring target processes. For example, below commands set processes having pids 42 and 4242 as the monitoring targets and check it again:

```
# cd <debugfs>/damon
# echo 42 4242 > target_ids
# cat target_ids
42 4242
```

Users can also monitor the physical memory address space of the system by writing a special keyword, "`paddr\n`" to the file. Because physical address space monitoring doesn't support multiple targets, reading the file will show a fake value, 42, as below:

```
# cd <debugfs>/damon
# echo paddr > target_ids
# cat target_ids
42
```

Note that setting the target ids doesn't start the monitoring.

Initial Monitoring Target Regions

In case of the virtual address space monitoring, DAMON automatically sets and updates the monitoring target regions so that entire memory mappings of target processes can be covered. However, users can want to limit the monitoring region to specific address ranges, such as the heap, the stack, or specific file-mapped area. Or, some users can know the initial access pattern of their workloads and therefore want to set optimal initial regions for the 'adaptive regions adjustment'.

In contrast, DAMON do not automatically sets and updates the monitoring target regions in case of physical memory monitoring. Therefore, users should set the monitoring target regions by themselves.

In such cases, users can explicitly set the initial monitoring target regions as they want, by writing proper values to the `init_regions` file. The input should be a sequence of three integers separated by white spaces that represent one region in below form.:

```
<target idx> <start address> <end address>
```

The `target idx` should be the index of the target in `target_ids` file, starting from `0`, and the regions should be passed in address order. For example, below commands will set a couple of address ranges, `1-100` and `100-200` as the initial monitoring target region of pid 42, which is the first one (index `0`) in `target_ids`, and another couple of address ranges, `20-40` and `50-100` as that of pid 4242, which is the second one (index `1`) in `target_ids`.:

```
# cd <debugfs>/damon
# cat target_ids
```

```
42 4242
# echo "0   1       100 \
        0   100     200 \
        1   20      40  \
        1   50      100" > init_regions
```

Note that this sets the initial monitoring target regions only. In case of virtual memory monitoring, DAMON will automatically updates the boundary of the regions after one `update interval`. Therefore, users should set the `update interval` large enough in this case, if they don't want the update.

Schemes

Users can get and set the DAMON-based operation schemes by reading from and writing to `schemes` debugfs file. Reading the file also shows the statistics of each scheme. To the file, each of the schemes should be represented in each line in below form:

```
<target access pattern> <action> <quota> <watermarks>
```

You can disable schemes by simply writing an empty string to the file.

Target Access Pattern

The target access pattern of the scheme. The `<target access pattern>` is constructed with three ranges in below form:

```
min-size max-size min-acc max-acc min-age max-age
```

Specifically, bytes for the size of regions (`min-size` and `max-size`), number of monitored accesses per aggregate interval for access frequency (`min-acc` and `max-acc`), number of aggregate intervals for the age of regions (`min-age` and `max-age`) are specified. Note that the ranges are closed interval.

Action

The `<action>` is a predefined integer for memory management actions. The supported numbers and their meanings are as below.

- 0: Call `madvise()` for the region with `MADV_WILLNEED`. Ignored if `target` is `paddr`.
- 1: Call `madvise()` for the region with `MADV_COLD`. Ignored if `target` is `paddr`.
- 2: Call `madvise()` for the region with `MADV_PAGEOUT`.
- 3: Call `madvise()` for the region with `MADV_HUGEPAGE`. Ignored if `target` is `paddr`.
- 4: Call `madvise()` for the region with `MADV_NOHUGEPAGE`. Ignored if `target` is `paddr`.
- 5: Do nothing but count the statistics

Quota

Users can set the quotas of the given scheme via the `<quota>` in below form:

```
<ms> <sz> <reset interval> <priority weights>
```

This makes DAMON to try to use only up to `<ms>` milliseconds for applying the action to memory regions of the `target access pattern` within the `<reset interval>` milliseconds, and to apply the action to only up to `<sz>` bytes of memory regions within the `<reset interval>`. Setting both `<ms>` and `<sz>` zero disables the quota limits.

For the prioritization, users can set the weights for the three properties in `<priority weights>` in below form:

```
<size weight> <access frequency weight> <age weight>
```

Watermarks

Users can specify watermarks of the given scheme via `<watermarks>` in below form:

```
<metric> <check interval> <high mark> <middle mark> <low mark>
```

`<metric>` is a predefined integer for the metric to be checked. The supported numbers and their meanings are as below.

- 0: Ignore the watermarks
- 1: System's free memory rate (per thousand)

The value of the metric is checked every `<check interval>` microseconds.

If the value is higher than `<high mark>` or lower than `<low mark>`, the scheme is deactivated. If the value is lower than `<mid mark>`, the scheme is activated.

Statistics

It also counts the total number and bytes of regions that each scheme is tried to be applied, the two numbers for the regions that each scheme is successfully applied, and the total number of the quota limit exceeds. This statistics can be used for online analysis or tuning of the schemes.

The statistics can be shown by reading the `schemes` file. Reading the file will show each scheme you entered in each line, and the five numbers for the statistics will be added at the end of each line.

Example

Below commands applies a scheme saying "If a memory region of size in [4KiB, 8KiB] is showing accesses per aggregate interval in [0, 5] for aggregate interval in [10, 20], page out the region. For the paging out, use only up to 10ms per second, and also don't page out more than 1GiB per second. Under the limitation, page out memory regions having longer age first. Also, check the free memory rate of the system every 5 seconds, start the monitoring and paging out when the free memory rate becomes lower than 50%, but stop it if the free memory rate becomes larger than 60%, or lower than 30%".:

```
# cd <debugfs>/damon
# scheme="4096 8192  0 5    10 20    2"  # target access pattern and action
# scheme+=" 10 $((1024*1024*1024)) 1000" # quotas
# scheme+=" 0 0 100"                     # prioritization weights
# scheme+=" 1 5000000 600 500 300"       # watermarks
# echo "$scheme" > schemes
```

Turning On/Off

Setting the files as described above doesn't incur effect unless you explicitly start the monitoring. You can start, stop, and check the current status of the monitoring by writing to and reading from the `monitor_on` file. Writing `on` to the file starts the monitoring of the targets with the attributes. Writing `off` to the file stops those. DAMON also stops if every target process is terminated. Below example commands turn on, off, and check the status of DAMON:

```
# cd <debugfs>/damon
# echo on > monitor_on
# echo off > monitor_on
# cat monitor_on
off
```

Please note that you cannot write to the above-mentioned debugfs files while the monitoring is turned on. If you write to the files while DAMON is running, an error code such as `-EBUSY` will be returned.

Monitoring Thread PID

DAMON does requested monitoring with a kernel thread called `kdamond`. You can get the pid of the thread by reading the `kdamond_pid` file. When the monitoring is turned off, reading the file returns `none`.

```
# cd <debugfs>/damon
# cat monitor_on
off
# cat kdamond_pid
none
# echo on > monitor_on
# cat kdamond_pid
18594
```

Using Multiple Monitoring Threads

One kdamond thread is created for each monitoring context. You can create and remove monitoring contexts for multiple kdamond required use case using the mk_contexts and rm_contexts files.

Writing the name of the new context to the mk_contexts file creates a directory of the name on the DAMON debugfs directory. The directory will have DAMON debugfs files for the context.

```
# cd <debugfs>/damon
# ls foo
# ls: cannot access 'foo': No such file or directory
# echo foo > mk_contexts
# ls foo
# attrs  init_regions  kdamond_pid  schemes  target_ids
```

If the context is not needed anymore, you can remove it and the corresponding directory by putting the name of the context to the rm_contexts file.

```
# echo foo > rm_contexts
# ls foo
# ls: cannot access 'foo': No such file or directory
```

Note that mk_contexts, rm_contexts, and monitor_on files are in the root directory only.

60.3.3 DAMON-based Reclamation

DAMON-based Reclamation (DAMON_RECLAIM) is a static kernel module that aimed to be used for proactive and lightweight reclamation under light memory pressure. It doesn't aim to replace the LRU-list based page_granularity reclamation, but to be selectively used for different level of memory pressure and requirements.

Where Proactive Reclamation is Required?

On general memory over-committed systems, proactively reclaiming cold pages helps saving memory and reducing latency spikes that incurred by the direct reclaim of the process or CPU consumption of kswapd, while incurring only minimal performance degradation[1][2] .

Free Pages Reporting[3] based memory over-commit virtualization systems are good example of the cases. In such systems, the guest VMs reports their free memory to host, and the host reallocates the reported memory to other guests. As a result, the memory of the systems are fully utilized. However, the guests could be not so memory-frugal, mainly because some kernel subsystems and user-space applications are designed to use as much memory as available. Then, guests could report only small amount of memory as free to host, results in memory utilization drop of the systems. Running the proactive reclamation in guests could mitigate this problem.

[1] https://research.google/pubs/pub48551/
[2] https://lwn.net/Articles/787611/
[3] https://www.kernel.org/doc/html/latest/mm/free_page_reporting.html

How It Works?

DAMON_RECLAIM finds memory regions that didn't accessed for specific time duration and page out. To avoid it consuming too much CPU for the paging out operation, a speed limit can be configured. Under the speed limit, it pages out memory regions that didn't accessed longer time first. System administrators can also configure under what situation this scheme should automatically activated and deactivated with three memory pressure watermarks.

Interface: Module Parameters

To use this feature, you should first ensure your system is running on a kernel that is built with `CONFIG_DAMON_RECLAIM=y`.

To let sysadmins enable or disable it and tune for the given system, DAMON_RECLAIM utilizes module parameters. That is, you can put `damon_reclaim.<parameter>=<value>` on the kernel boot command line or write proper values to `/sys/module/damon_reclaim/parameters/<parameter>` files.

Below are the description of each parameter.

enabled

Enable or disable DAMON_RECLAIM.

You can enable DAMON_RCLAIM by setting the value of this parameter as `Y`. Setting it as `N` disables DAMON_RECLAIM. Note that DAMON_RECLAIM could do no real monitoring and reclamation due to the watermarks-based activation condition. Refer to below descriptions for the watermarks parameter for this.

commit_inputs

Make DAMON_RECLAIM reads the input parameters again, except `enabled`.

Input parameters that updated while DAMON_RECLAIM is running are not applied by default. Once this parameter is set as `Y`, DAMON_RECLAIM reads values of parametrs except `enabled` again. Once the re-reading is done, this parameter is set as `N`. If invalid parameters are found while the re-reading, DAMON_RECLAIM will be disabled.

min_age

Time threshold for cold memory regions identification in microseconds.

If a memory region is not accessed for this or longer time, DAMON_RECLAIM identifies the region as cold, and reclaims it.

120 seconds by default.

quota_ms

Limit of time for the reclamation in milliseconds.

DAMON_RECLAIM tries to use only up to this time within a time window (quota_reset_interval_ms) for trying reclamation of cold pages. This can be used for limiting CPU consumption of DAMON_RECLAIM. If the value is zero, the limit is disabled.

10 ms by default.

quota_sz

Limit of size of memory for the reclamation in bytes.

DAMON_RECLAIM charges amount of memory which it tried to reclaim within a time window (quota_reset_interval_ms) and makes no more than this limit is tried. This can be used for limiting consumption of CPU and IO. If this value is zero, the limit is disabled.

128 MiB by default.

quota_reset_interval_ms

The time/size quota charge reset interval in milliseconds.

The charget reset interval for the quota of time (quota_ms) and size (quota_sz). That is, DAMON_RECLAIM does not try reclamation for more than quota_ms milliseconds or quota_sz bytes within quota_reset_interval_ms milliseconds.

1 second by default.

wmarks_interval

Minimal time to wait before checking the watermarks, when DAMON_RECLAIM is enabled but inactive due to its watermarks rule.

wmarks_high

Free memory rate (per thousand) for the high watermark.

If free memory of the system in bytes per thousand bytes is higher than this, DAMON_RECLAIM becomes inactive, so it does nothing but only periodically checks the watermarks.

wmarks_mid

Free memory rate (per thousand) for the middle watermark.

If free memory of the system in bytes per thousand bytes is between this and the low watermark, DAMON_RECLAIM becomes active, so starts the monitoring and the reclaiming.

wmarks_low

Free memory rate (per thousand) for the low watermark.

If free memory of the system in bytes per thousand bytes is lower than this, DAMON_RECLAIM becomes inactive, so it does nothing but periodically checks the watermarks. In the case, the system falls back to the LRU-list based page granularity reclamation logic.

sample_interval

Sampling interval for the monitoring in microseconds.

The sampling interval of DAMON for the cold memory monitoring. Please refer to the DAMON documentation (*Detailed Usages*) for more detail.

aggr_interval

Aggregation interval for the monitoring in microseconds.

The aggregation interval of DAMON for the cold memory monitoring. Please refer to the DAMON documentation (*Detailed Usages*) for more detail.

min_nr_regions

Minimum number of monitoring regions.

The minimal number of monitoring regions of DAMON for the cold memory monitoring. This can be used to set lower-bound of the monitoring quality. But, setting this too high could result in increased monitoring overhead. Please refer to the DAMON documentation (*Detailed Usages*) for more detail.

max_nr_regions

Maximum number of monitoring regions.

The maximum number of monitoring regions of DAMON for the cold memory monitoring. This can be used to set upper-bound of the monitoring overhead. However, setting this too low could result in bad monitoring quality. Please refer to the DAMON documentation (*Detailed Usages*) for more detail.

monitor_region_start

Start of target memory region in physical address.

The start physical address of memory region that DAMON_RECLAIM will do work against. That is, DAMON_RECLAIM will find cold memory regions in this region and reclaims. By default, biggest System RAM is used as the region.

monitor_region_end

End of target memory region in physical address.

The end physical address of memory region that DAMON_RECLAIM will do work against. That is, DAMON_RECLAIM will find cold memory regions in this region and reclaims. By default, biggest System RAM is used as the region.

skip_anon

Skip anonymous pages reclamation.

If this parameter is set as `Y`, DAMON_RECLAIM does not reclaim anonymous pages. By default, `N`.

kdamond_pid

PID of the DAMON thread.

If DAMON_RECLAIM is enabled, this becomes the PID of the worker thread. Else, -1.

nr_reclaim_tried_regions

Number of memory regions that tried to be reclaimed by DAMON_RECLAIM.

bytes_reclaim_tried_regions

Total bytes of memory regions that tried to be reclaimed by DAMON_RECLAIM.

nr_reclaimed_regions

Number of memory regions that successfully be reclaimed by DAMON_RECLAIM.

bytes_reclaimed_regions

Total bytes of memory regions that successfully be reclaimed by DAMON_RECLAIM.

nr_quota_exceeds

Number of times that the time/space quota limits have exceeded.

Example

Below runtime example commands make DAMON_RECLAIM to find memory regions that not accessed for 30 seconds or more and pages out. The reclamation is limited to be done only up to 1 GiB per second to avoid DAMON_RECLAIM consuming too much CPU time for the paging out operation. It also asks DAMON_RECLAIM to do nothing if the system's free memory rate is more than 50%, but start the real works if it becomes lower than 40%. If DAMON_RECLAIM doesn't make progress and therefore the free memory rate becomes lower than 20%, it asks DAMON_RECLAIM to do nothing again, so that we can fall back to the LRU-list based page granularity reclamation.

```
# cd /sys/module/damon_reclaim/parameters
# echo 30000000 > min_age
# echo $((1 * 1024 * 1024 * 1024)) > quota_sz
# echo 1000 > quota_reset_interval_ms
# echo 500 > wmarks_high
# echo 400 > wmarks_mid
# echo 200 > wmarks_low
# echo Y > enabled
```

60.3.4 DAMON-based LRU-lists Sorting

DAMON-based LRU-lists Sorting (DAMON_LRU_SORT) is a static kernel module that aimed to be used for proactive and lightweight data access pattern based (de)prioritization of pages on their LRU-lists for making LRU-lists a more trusworthy data access pattern source.

Where Proactive LRU-lists Sorting is Required?

As page-granularity access checking overhead could be significant on huge systems, LRU lists are normally not proactively sorted but partially and reactively sorted for special events including specific user requests, system calls and memory pressure. As a result, LRU lists are sometimes not so perfectly prepared to be used as a trustworthy access pattern source for some situations including reclamation target pages selection under sudden memory pressure.

Because DAMON can identify access patterns of best-effort accuracy while inducing only user-specified range of overhead, proactively running DAMON_LRU_SORT could be helpful for making LRU lists more trustworthy access pattern source with low and controlled overhead.

How It Works?

DAMON_LRU_SORT finds hot pages (pages of memory regions that showing access rates that higher than a user-specified threshold) and cold pages (pages of memory regions that showing no access for a time that longer than a user-specified threshold) using DAMON, and prioritizes hot pages while deprioritizing cold pages on their LRU-lists. To avoid it consuming too much CPU for the prioritizations, a CPU time usage limit can be configured. Under the limit, it prioritizes and deprioritizes more hot and cold pages first, respectively. System administrators can also configure under what situation this scheme should automatically activated and deactivated with three memory pressure watermarks.

Its default parameters for hotness/coldness thresholds and CPU quota limit are conservatively chosen. That is, the module under its default parameters could be widely used without harm for common situations while providing a level of benefits for systems having clear hot/cold access patterns under memory pressure while consuming only a limited small portion of CPU time.

Interface: Module Parameters

To use this feature, you should first ensure your system is running on a kernel that is built with `CONFIG_DAMON_LRU_SORT=y`.

To let sysadmins enable or disable it and tune for the given system, DAMON_LRU_SORT utilizes module parameters. That is, you can put `damon_lru_sort.<parameter>=<value>` on the kernel boot command line or write proper values to `/sys/module/damon_lru_sort/parameters/<parameter>` files.

Below are the description of each parameter.

enabled

Enable or disable DAMON_LRU_SORT.

You can enable DAMON_LRU_SORT by setting the value of this parameter as `Y`. Setting it as `N` disables DAMON_LRU_SORT. Note that DAMON_LRU_SORT could do no real monitoring and LRU-lists sorting due to the watermarks-based activation condition. Refer to below descriptions for the watermarks parameter for this.

commit_inputs

Make DAMON_LRU_SORT reads the input parameters again, except `enabled`.

Input parameters that updated while DAMON_LRU_SORT is running are not applied by default. Once this parameter is set as `Y`, DAMON_LRU_SORT reads values of parametrs except `enabled` again. Once the re-reading is done, this parameter is set as `N`. If invalid parameters are found while the re-reading, DAMON_LRU_SORT will be disabled.

hot_thres_access_freq

Access frequency threshold for hot memory regions identification in permil.

If a memory region is accessed in frequency of this or higher, DAMON_LRU_SORT identifies the region as hot, and mark it as accessed on the LRU list, so that it could not be reclaimed under memory pressure. 50% by default.

cold_min_age

Time threshold for cold memory regions identification in microseconds.

If a memory region is not accessed for this or longer time, DAMON_LRU_SORT identifies the region as cold, and mark it as unaccessed on the LRU list, so that it could be reclaimed first under memory pressure. 120 seconds by default.

quota_ms

Limit of time for trying the LRU lists sorting in milliseconds.

DAMON_LRU_SORT tries to use only up to this time within a time window (quota_reset_interval_ms) for trying LRU lists sorting. This can be used for limiting CPU consumption of DAMON_LRU_SORT. If the value is zero, the limit is disabled.

10 ms by default.

quota_reset_interval_ms

The time quota charge reset interval in milliseconds.

The charge reset interval for the quota of time (quota_ms). That is, DAMON_LRU_SORT does not try LRU-lists sorting for more than quota_ms milliseconds or quota_sz bytes within quota_reset_interval_ms milliseconds.

1 second by default.

wmarks_interval

The watermarks check time interval in microseconds.

Minimal time to wait before checking the watermarks, when DAMON_LRU_SORT is enabled but inactive due to its watermarks rule. 5 seconds by default.

wmarks_high

Free memory rate (per thousand) for the high watermark.

If free memory of the system in bytes per thousand bytes is higher than this, DAMON_LRU_SORT becomes inactive, so it does nothing but periodically checks the watermarks. 200 (20%) by default.

wmarks_mid

Free memory rate (per thousand) for the middle watermark.

If free memory of the system in bytes per thousand bytes is between this and the low watermark, DAMON_LRU_SORT becomes active, so starts the monitoring and the LRU-lists sorting. 150 (15%) by default.

wmarks_low

Free memory rate (per thousand) for the low watermark.

If free memory of the system in bytes per thousand bytes is lower than this, DAMON_LRU_SORT becomes inactive, so it does nothing but periodically checks the watermarks. 50 (5%) by default.

sample_interval

Sampling interval for the monitoring in microseconds.

The sampling interval of DAMON for the cold memory monitoring. Please refer to the DAMON documentation (*Detailed Usages*) for more detail. 5ms by default.

aggr_interval

Aggregation interval for the monitoring in microseconds.

The aggregation interval of DAMON for the cold memory monitoring. Please refer to the DAMON documentation (*Detailed Usages*) for more detail. 100ms by default.

min_nr_regions

Minimum number of monitoring regions.

The minimal number of monitoring regions of DAMON for the cold memory monitoring. This can be used to set lower-bound of the monitoring quality. But, setting this too high could result in increased monitoring overhead. Please refer to the DAMON documentation (*Detailed Usages*) for more detail. 10 by default.

max_nr_regions

Maximum number of monitoring regions.

The maximum number of monitoring regions of DAMON for the cold memory monitoring. This can be used to set upper-bound of the monitoring overhead. However, setting this too low could result in bad monitoring quality. Please refer to the DAMON documentation (*Detailed Usages*) for more detail. 1000 by defaults.

monitor_region_start

Start of target memory region in physical address.

The start physical address of memory region that DAMON_LRU_SORT will do work against. By default, biggest System RAM is used as the region.

monitor_region_end

End of target memory region in physical address.

The end physical address of memory region that DAMON_LRU_SORT will do work against. By default, biggest System RAM is used as the region.

kdamond_pid

PID of the DAMON thread.

If DAMON_LRU_SORT is enabled, this becomes the PID of the worker thread. Else, -1.

nr_lru_sort_tried_hot_regions

Number of hot memory regions that tried to be LRU-sorted.

bytes_lru_sort_tried_hot_regions

Total bytes of hot memory regions that tried to be LRU-sorted.

nr_lru_sorted_hot_regions

Number of hot memory regions that successfully be LRU-sorted.

bytes_lru_sorted_hot_regions

Total bytes of hot memory regions that successfully be LRU-sorted.

nr_hot_quota_exceeds

Number of times that the time quota limit for hot regions have exceeded.

nr_lru_sort_tried_cold_regions

Number of cold memory regions that tried to be LRU-sorted.

bytes_lru_sort_tried_cold_regions

Total bytes of cold memory regions that tried to be LRU-sorted.

nr_lru_sorted_cold_regions

Number of cold memory regions that successfully be LRU-sorted.

bytes_lru_sorted_cold_regions

Total bytes of cold memory regions that successfully be LRU-sorted.

nr_cold_quota_exceeds

Number of times that the time quota limit for cold regions have exceeded.

Example

Below runtime example commands make DAMON_LRU_SORT to find memory regions having >=50% access frequency and LRU-prioritize while LRU-deprioritizing memory regions that not accessed for 120 seconds. The prioritization and deprioritization is limited to be done using only up to 1% CPU time to avoid DAMON_LRU_SORT consuming too much CPU time for the (de)prioritization. It also asks DAMON_LRU_SORT to do nothing if the system's free memory rate is more than 50%, but start the real works if it becomes lower than 40%. If DAMON_RECLAIM doesn't make progress and therefore the free memory rate becomes lower than 20%, it asks DAMON_LRU_SORT to do nothing again, so that we can fall back to the LRU-list based page granularity reclamation.

```
# cd /sys/module/damon_lru_sort/parameters
# echo 500 > hot_thres_access_freq
# echo 120000000 > cold_min_age
# echo 10 > quota_ms
# echo 1000 > quota_reset_interval_ms
```

```
# echo 500 > wmarks_high
# echo 400 > wmarks_mid
# echo 200 > wmarks_low
# echo Y > enabled
```

60.4 HugeTLB Pages

60.4.1 Overview

The intent of this file is to give a brief summary of hugetlbpage support in the Linux kernel. This support is built on top of multiple page size support that is provided by most modern architectures. For example, x86 CPUs normally support 4K and 2M (1G if architecturally supported) page sizes, ia64 architecture supports multiple page sizes 4K, 8K, 64K, 256K, 1M, 4M, 16M, 256M and ppc64 supports 4K and 16M. A TLB is a cache of virtual-to-physical translations. Typically this is a very scarce resource on processor. Operating systems try to make best use of limited number of TLB resources. This optimization is more critical now as bigger and bigger physical memories (several GBs) are more readily available.

Users can use the huge page support in Linux kernel by either using the mmap system call or standard SYSV shared memory system calls (shmget, shmat).

First the Linux kernel needs to be built with the CONFIG_HUGETLBFS (present under "File systems") and CONFIG_HUGETLB_PAGE (selected automatically when CONFIG_HUGETLBFS is selected) configuration options.

The `/proc/meminfo` file provides information about the total number of persistent hugetlb pages in the kernel's huge page pool. It also displays default huge page size and information about the number of free, reserved and surplus huge pages in the pool of huge pages of default size. The huge page size is needed for generating the proper alignment and size of the arguments to system calls that map huge page regions.

The output of `cat /proc/meminfo` will include lines like:

```
HugePages_Total: uuu
HugePages_Free:  vvv
HugePages_Rsvd:  www
HugePages_Surp:  xxx
Hugepagesize:    yyy kB
Hugetlb:         zzz kB
```

where:

HugePages_Total
is the size of the pool of huge pages.

HugePages_Free
is the number of huge pages in the pool that are not yet allocated.

HugePages_Rsvd
is short for "reserved," and is the number of huge pages for which a commitment to allocate from the pool has been made, but no allocation has yet been made. Reserved huge pages

guarantee that an application will be able to allocate a huge page from the pool of huge pages at fault time.

HugePages_Surp
is short for "surplus," and is the number of huge pages in the pool above the value in `/proc/sys/vm/nr_hugepages`. The maximum number of surplus huge pages is controlled by `/proc/sys/vm/nr_overcommit_hugepages`. Note: When the feature of freeing unused vmemmap pages associated with each hugetlb page is enabled, the number of surplus huge pages may be temporarily larger than the maximum number of surplus huge pages when the system is under memory pressure.

Hugepagesize
is the default hugepage size (in kB).

Hugetlb
is the total amount of memory (in kB), consumed by huge pages of all sizes. If huge pages of different sizes are in use, this number will exceed HugePages_Total * Hugepagesize. To get more detailed information, please, refer to `/sys/kernel/mm/hugepages` (described below).

`/proc/filesystems` should also show a filesystem of type "hugetlbfs" configured in the kernel.

`/proc/sys/vm/nr_hugepages` indicates the current number of "persistent" huge pages in the kernel's huge page pool. "Persistent" huge pages will be returned to the huge page pool when freed by a task. A user with root privileges can dynamically allocate more or free some persistent huge pages by increasing or decreasing the value of `nr_hugepages`.

Note: When the feature of freeing unused vmemmap pages associated with each hugetlb page is enabled, we can fail to free the huge pages triggered by the user when the system is under memory pressure. Please try again later.

Pages that are used as huge pages are reserved inside the kernel and cannot be used for other purposes. Huge pages cannot be swapped out under memory pressure.

Once a number of huge pages have been pre-allocated to the kernel huge page pool, a user with appropriate privilege can use either the mmap system call or shared memory system calls to use the huge pages. See the discussion of *Using Huge Pages*, below.

The administrator can allocate persistent huge pages on the kernel boot command line by specifying the "hugepages=N" parameter, where 'N' = the number of huge pages requested. This is the most reliable method of allocating huge pages as memory has not yet become fragmented.

Some platforms support multiple huge page sizes. To allocate huge pages of a specific size, one must precede the huge pages boot command parameters with a huge page size selection parameter "hugepagesz=<size>". <size> must be specified in bytes with optional scale suffix [kKmMgG]. The default huge page size may be selected with the "default_hugepagesz=<size>" boot parameter.

Hugetlb boot command line parameter semantics

hugepagesz
Specify a huge page size. Used in conjunction with hugepages parameter to preallocate a number of huge pages of the specified size. Hence, hugepagesz and hugepages are typically specified in pairs such as:

```
hugepagesz=2M hugepages=512
```

hugepagesz can only be specified once on the command line for a specific huge page size. Valid huge page sizes are architecture dependent.

hugepages
Specify the number of huge pages to preallocate. This typically follows a valid hugepagesz or default_hugepagesz parameter. However, if hugepages is the first or only hugetlb command line parameter it implicitly specifies the number of huge pages of default size to allocate. If the number of huge pages of default size is implicitly specified, it can not be overwritten by a hugepagesz,hugepages parameter pair for the default size. This parameter also has a node format. The node format specifies the number of huge pages to allocate on specific nodes.

For example, on an architecture with 2M default huge page size:

```
hugepages=256 hugepagesz=2M hugepages=512
```

will result in 256 2M huge pages being allocated and a warning message indicating that the hugepages=512 parameter is ignored. If a hugepages parameter is preceded by an invalid hugepagesz parameter, it will be ignored.

Node format example:

```
hugepagesz=2M hugepages=0:1,1:2
```

It will allocate 1 2M hugepage on node0 and 2 2M hugepages on node1. If the node number is invalid, the parameter will be ignored.

default_hugepagesz
Specify the default huge page size. This parameter can only be specified once on the command line. default_hugepagesz can optionally be followed by the hugepages parameter to preallocate a specific number of huge pages of default size. The number of default sized huge pages to preallocate can also be implicitly specified as mentioned in the hugepages section above. Therefore, on an architecture with 2M default huge page size:

```
hugepages=256
default_hugepagesz=2M hugepages=256
hugepages=256 default_hugepagesz=2M
```

will all result in 256 2M huge pages being allocated. Valid default huge page size is architecture dependent.

hugetlb_free_vmemmap
When CONFIG_HUGETLB_PAGE_OPTIMIZE_VMEMMAP is set, this enables HugeTLB Vmemmap Optimization (HVO).

When multiple huge page sizes are supported, `/proc/sys/vm/nr_hugepages` indicates the current number of pre-allocated huge pages of the default size. Thus, one can use the following command to dynamically allocate/deallocate default sized persistent huge pages:

```
echo 20 > /proc/sys/vm/nr_hugepages
```

This command will try to adjust the number of default sized huge pages in the huge page pool to 20, allocating or freeing huge pages, as required.

On a NUMA platform, the kernel will attempt to distribute the huge page pool over all the set of allowed nodes specified by the NUMA memory policy of the task that modifies `nr_hugepages`.

The default for the allowed nodes--when the task has default memory policy--is all on-line nodes with memory. Allowed nodes with insufficient available, contiguous memory for a huge page will be silently skipped when allocating persistent huge pages. See the *discussion below* of the interaction of task memory policy, cpusets and per node attributes with the allocation and freeing of persistent huge pages.

The success or failure of huge page allocation depends on the amount of physically contiguous memory that is present in system at the time of the allocation attempt. If the kernel is unable to allocate huge pages from some nodes in a NUMA system, it will attempt to make up the difference by allocating extra pages on other nodes with sufficient available contiguous memory, if any.

System administrators may want to put this command in one of the local rc init files. This will enable the kernel to allocate huge pages early in the boot process when the possibility of getting physical contiguous pages is still very high. Administrators can verify the number of huge pages actually allocated by checking the sysctl or meminfo. To check the per node distribution of huge pages in a NUMA system, use:

```
cat /sys/devices/system/node/node*/meminfo | fgrep Huge
```

`/proc/sys/vm/nr_overcommit_hugepages` specifies how large the pool of huge pages can grow, if more huge pages than `/proc/sys/vm/nr_hugepages` are requested by applications. Writing any non-zero value into this file indicates that the hugetlb subsystem is allowed to try to obtain that number of ”surplus” huge pages from the kernel's normal page pool, when the persistent huge page pool is exhausted. As these surplus huge pages become unused, they are freed back to the kernel's normal page pool.

When increasing the huge page pool size via `nr_hugepages`, any existing surplus pages will first be promoted to persistent huge pages. Then, additional huge pages will be allocated, if necessary and if possible, to fulfill the new persistent huge page pool size.

The administrator may shrink the pool of persistent huge pages for the default huge page size by setting the `nr_hugepages` sysctl to a smaller value. The kernel will attempt to balance the freeing of huge pages across all nodes in the memory policy of the task modifying `nr_hugepages`. Any free huge pages on the selected nodes will be freed back to the kernel's normal page pool.

Caveat: Shrinking the persistent huge page pool via `nr_hugepages` such that it becomes less than the number of huge pages in use will convert the balance of the in-use huge pages to surplus huge pages. This will occur even if the number of surplus pages would exceed the overcommit value. As long as this condition holds--that is, until `nr_hugepages+nr_overcommit_hugepages` is increased sufficiently, or the surplus huge pages go out of use and are freed-- no more surplus huge pages will be allowed to be allocated.

With support for multiple huge page pools at run-time available, much of the huge page userspace interface in `/proc/sys/vm` has been duplicated in sysfs. The `/proc` interfaces discussed above have been retained for backwards compatibility. The root huge page control directory in sysfs is:

```
/sys/kernel/mm/hugepages
```

For each huge page size supported by the running kernel, a subdirectory will exist, of the form:

```
hugepages-${size}kB
```

Inside each of these directories, the set of files contained in /proc will exist. In addition, two additional interfaces for demoting huge pages may exist:

```
demote
demote_size
nr_hugepages
nr_hugepages_mempolicy
nr_overcommit_hugepages
free_hugepages
resv_hugepages
surplus_hugepages
```

The demote interfaces provide the ability to split a huge page into smaller huge pages. For example, the x86 architecture supports both 1GB and 2MB huge pages sizes. A 1GB huge page can be split into 512 2MB huge pages. Demote interfaces are not available for the smallest huge page size. The demote interfaces are:

demote_size
is the size of demoted pages. When a page is demoted a corresponding number of huge pages of demote_size will be created. By default, demote_size is set to the next smaller huge page size. If there are multiple smaller huge page sizes, demote_size can be set to any of these smaller sizes. Only huge page sizes less than the current huge pages size are allowed.

demote
is used to demote a number of huge pages. A user with root privileges can write to this file. It may not be possible to demote the requested number of huge pages. To determine how many pages were actually demoted, compare the value of nr_hugepages before and after writing to the demote interface. demote is a write only interface.

The interfaces which are the same as in /proc (all except demote and demote_size) function as described above for the default huge page-sized case.

60.4.2 Interaction of Task Memory Policy with Huge Page Allocation/Freeing

Whether huge pages are allocated and freed via the /proc interface or the /sysfs interface using the nr_hugepages_mempolicy attribute, the NUMA nodes from which huge pages are allocated or freed are controlled by the NUMA memory policy of the task that modifies the nr_hugepages_mempolicy sysctl or attribute. When the nr_hugepages attribute is used, mempolicy is ignored.

The recommended method to allocate or free huge pages to/from the kernel huge page pool, using the nr_hugepages example above, is:

```
numactl --interleave <node-list> echo 20 \
                            >/proc/sys/vm/nr_hugepages_mempolicy
```

or, more succinctly:

```
numactl -m <node-list> echo 20 >/proc/sys/vm/nr_hugepages_mempolicy
```

This will allocate or free abs(20 - nr_hugepages) to or from the nodes specified in <node-list>, depending on whether number of persistent huge pages is initially less than or greater

than 20, respectively. No huge pages will be allocated nor freed on any node not included in the specified <node-list>.

When adjusting the persistent hugepage count via `nr_hugepages_mempolicy`, any memory policy mode--bind, preferred, local or interleave--may be used. The resulting effect on persistent huge page allocation is as follows:

1. Regardless of mempolicy mode [see *NUMA Memory Policy*], persistent huge pages will be distributed across the node or nodes specified in the mempolicy as if "interleave" had been specified. However, if a node in the policy does not contain sufficient contiguous memory for a huge page, the allocation will not "fallback" to the nearest neighbor node with sufficient contiguous memory. To do this would cause undesirable imbalance in the distribution of the huge page pool, or possibly, allocation of persistent huge pages on nodes not allowed by the task's memory policy.
2. One or more nodes may be specified with the bind or interleave policy. If more than one node is specified with the preferred policy, only the lowest numeric id will be used. Local policy will select the node where the task is running at the time the nodes_allowed mask is constructed. For local policy to be deterministic, the task must be bound to a cpu or cpus in a single node. Otherwise, the task could be migrated to some other node at any time after launch and the resulting node will be indeterminate. Thus, local policy is not very useful for this purpose. Any of the other mempolicy modes may be used to specify a single node.
3. The nodes allowed mask will be derived from any non-default task mempolicy, whether this policy was set explicitly by the task itself or one of its ancestors, such as numactl. This means that if the task is invoked from a shell with non-default policy, that policy will be used. One can specify a node list of "all" with numactl --interleave or --membind [-m] to achieve interleaving over all nodes in the system or cpuset.
4. Any task mempolicy specified--e.g., using numactl--will be constrained by the resource limits of any cpuset in which the task runs. Thus, there will be no way for a task with non-default policy running in a cpuset with a subset of the system nodes to allocate huge pages outside the cpuset without first moving to a cpuset that contains all of the desired nodes.
5. Boot-time huge page allocation attempts to distribute the requested number of huge pages over all on-lines nodes with memory.

60.4.3 Per Node Hugepages Attributes

A subset of the contents of the root huge page control directory in sysfs, described above, will be replicated under each the system device of each NUMA node with memory in:

```
/sys/devices/system/node/node[0-9]*/hugepages/
```

Under this directory, the subdirectory for each supported huge page size contains the following attribute files:

```
nr_hugepages
free_hugepages
surplus_hugepages
```

The free_' and surplus_' attribute files are read-only. They return the number of free and surplus [overcommitted] huge pages, respectively, on the parent node.

The `nr_hugepages` attribute returns the total number of huge pages on the specified node. When this attribute is written, the number of persistent huge pages on the parent node will be adjusted to the specified value, if sufficient resources exist, regardless of the task's mempolicy or cpuset constraints.

Note that the number of overcommit and reserve pages remain global quantities, as we don't know until fault time, when the faulting task's mempolicy is applied, from which node the huge page allocation will be attempted.

60.4.4 Using Huge Pages

If the user applications are going to request huge pages using mmap system call, then it is required that system administrator mount a file system of type hugetlbfs:

```
mount -t hugetlbfs \
      -o uid=<value>,gid=<value>,mode=<value>,pagesize=<value>,size=<value>,\
      min_size=<value>,nr_inodes=<value> none /mnt/huge
```

This command mounts a (pseudo) filesystem of type hugetlbfs on the directory `/mnt/huge`. Any file created on `/mnt/huge` uses huge pages.

The `uid` and `gid` options sets the owner and group of the root of the file system. By default the `uid` and `gid` of the current process are taken.

The `mode` option sets the mode of root of file system to value & 01777. This value is given in octal. By default the value 0755 is picked.

If the platform supports multiple huge page sizes, the `pagesize` option can be used to specify the huge page size and associated pool. `pagesize` is specified in bytes. If `pagesize` is not specified the platform's default huge page size and associated pool will be used.

The `size` option sets the maximum value of memory (huge pages) allowed for that filesystem (`/mnt/huge`). The `size` option can be specified in bytes, or as a percentage of the specified huge page pool (`nr_hugepages`). The size is rounded down to HPAGE_SIZE boundary.

The `min_size` option sets the minimum value of memory (huge pages) allowed for the filesystem. `min_size` can be specified in the same way as `size`, either bytes or a percentage of the huge page pool. At mount time, the number of huge pages specified by `min_size` are reserved for use by the filesystem. If there are not enough free huge pages available, the mount will fail. As huge pages are allocated to the filesystem and freed, the reserve count is adjusted so that the sum of allocated and reserved huge pages is always at least `min_size`.

The option `nr_inodes` sets the maximum number of inodes that `/mnt/huge` can use.

If the `size`, `min_size` or `nr_inodes` option is not provided on command line then no limits are set.

For `pagesize`, `size`, `min_size` and `nr_inodes` options, you can use [G|g]/[M|m]/[K|k] to represent giga/mega/kilo. For example, size=2K has the same meaning as size=2048.

While read system calls are supported on files that reside on hugetlb file systems, write system calls are not.

Regular chown, chgrp, and chmod commands (with right permissions) could be used to change the file attributes on hugetlbfs.

Also, it is important to note that no such mount command is required if applications are going to use only shmat/shmget system calls or mmap with MAP_HUGETLB. For an example of how to use mmap with MAP_HUGETLB see *map_hugetlb* below.

Users who wish to use hugetlb memory via shared memory segment should be members of a supplementary group and system admin needs to configure that gid into `/proc/sys/vm/hugetlb_shm_group`. It is possible for same or different applications to use any combination of mmaps and shm* calls, though the mount of filesystem will be required for using mmap calls without MAP_HUGETLB.

Syscalls that operate on memory backed by hugetlb pages only have their lengths aligned to the native page size of the processor; they will normally fail with errno set to EINVAL or exclude hugetlb pages that extend beyond the length if not hugepage aligned. For example, munmap(2) will fail if memory is backed by a hugetlb page and the length is smaller than the hugepage size.

60.4.5 Examples

`map_hugetlb`
: see tools/testing/selftests/mm/map_hugetlb.c

`hugepage-shm`
: see tools/testing/selftests/mm/hugepage-shm.c

`hugepage-mmap`
: see tools/testing/selftests/mm/hugepage-mmap.c

The libhugetlbfs library provides a wide range of userspace tools to help with huge page usability, environment setup, and control.

60.5 Idle Page Tracking

60.5.1 Motivation

The idle page tracking feature allows to track which memory pages are being accessed by a workload and which are idle. This information can be useful for estimating the workload's working set size, which, in turn, can be taken into account when configuring the workload parameters, setting memory cgroup limits, or deciding where to place the workload within a compute cluster.

It is enabled by CONFIG_IDLE_PAGE_TRACKING=y.

60.5.2 User API

The idle page tracking API is located at `/sys/kernel/mm/page_idle`. Currently, it consists of the only read-write file, `/sys/kernel/mm/page_idle/bitmap`.

The file implements a bitmap where each bit corresponds to a memory page. The bitmap is represented by an array of 8-byte integers, and the page at PFN #i is mapped to bit #i%64 of array element #i/64, byte order is native. When a bit is set, the corresponding page is idle.

A page is considered idle if it has not been accessed since it was marked idle (for more details on what "accessed" actually means see the *Implementation Details* section). To mark a page idle one has to set the bit corresponding to the page by writing to the file. A value written to the file is OR-ed with the current bitmap value.

Only accesses to user memory pages are tracked. These are pages mapped to a process address space, page cache and buffer pages, swap cache pages. For other page types (e.g. SLAB pages) an attempt to mark a page idle is silently ignored, and hence such pages are never reported idle.

For huge pages the idle flag is set only on the head page, so one has to read `/proc/kpageflags` in order to correctly count idle huge pages.

Reading from or writing to `/sys/kernel/mm/page_idle/bitmap` will return -EINVAL if you are not starting the read/write on an 8-byte boundary, or if the size of the read/write is not a multiple of 8 bytes. Writing to this file beyond max PFN will return -ENXIO.

That said, in order to estimate the amount of pages that are not used by a workload one should:

1. Mark all the workload's pages as idle by setting corresponding bits in `/sys/kernel/mm/page_idle/bitmap`. The pages can be found by reading `/proc/pid/pagemap` if the workload is represented by a process, or by filtering out alien pages using `/proc/kpagecgroup` in case the workload is placed in a memory cgroup.
2. Wait until the workload accesses its working set.
3. Read `/sys/kernel/mm/page_idle/bitmap` and count the number of bits set. If one wants to ignore certain types of pages, e.g. mlocked pages since they are not reclaimable, he or she can filter them out using `/proc/kpageflags`.

The page-types tool in the tools/mm directory can be used to assist in this. If the tool is run initially with the appropriate option, it will mark all the queried pages as idle. Subsequent runs of the tool can then show which pages have their idle flag cleared in the interim.

See *Examining Process Page Tables* for more information about `/proc/pid/pagemap`, `/proc/kpageflags`, and `/proc/kpagecgroup`.

60.5.3 Implementation Details

The kernel internally keeps track of accesses to user memory pages in order to reclaim unreferenced pages first on memory shortage conditions. A page is considered referenced if it has been recently accessed via a process address space, in which case one or more PTEs it is mapped to will have the Accessed bit set, or marked accessed explicitly by the kernel (see mark_page_accessed()). The latter happens when:

- a userspace process reads or writes a page using a system call (e.g. read(2) or write(2))

- a page that is used for storing filesystem buffers is read or written, because a process needs filesystem metadata stored in it (e.g. lists a directory tree)
- a page is accessed by a device driver using get_user_pages()

When a dirty page is written to swap or disk as a result of memory reclaim or exceeding the dirty memory limit, it is not marked referenced.

The idle memory tracking feature adds a new page flag, the Idle flag. This flag is set manually, by writing to `/sys/kernel/mm/page_idle/bitmap` (see the *User API* section), and cleared automatically whenever a page is referenced as defined above.

When a page is marked idle, the Accessed bit must be cleared in all PTEs it is mapped to, otherwise we will not be able to detect accesses to the page coming from a process address space. To avoid interference with the reclaimer, which, as noted above, uses the Accessed bit to promote actively referenced pages, one more page flag is introduced, the Young flag. When the PTE Accessed bit is cleared as a result of setting or updating a page's Idle flag, the Young flag is set on the page. The reclaimer treats the Young flag as an extra PTE Accessed bit and therefore will consider such a page as referenced.

Since the idle memory tracking feature is based on the memory reclaimer logic, it only works with pages that are on an LRU list, other pages are silently ignored. That means it will ignore a user memory page if it is isolated, but since there are usually not many of them, it should not affect the overall result noticeably. In order not to stall scanning of the idle page bitmap, locked pages may be skipped too.

60.6 Kernel Samepage Merging

60.6.1 Overview

KSM is a memory-saving de-duplication feature, enabled by CONFIG_KSM=y, added to the Linux kernel in 2.6.32. See `mm/ksm.c` for its implementation, and http://lwn.net/Articles/306704/ and https://lwn.net/Articles/330589/

KSM was originally developed for use with KVM (where it was known as Kernel Shared Memory), to fit more virtual machines into physical memory, by sharing the data common between them. But it can be useful to any application which generates many instances of the same data.

The KSM daemon ksmd periodically scans those areas of user memory which have been registered with it, looking for pages of identical content which can be replaced by a single write-protected page (which is automatically copied if a process later wants to update its content). The amount of pages that KSM daemon scans in a single pass and the time between the passes are configured using *sysfs interface*

KSM only merges anonymous (private) pages, never pagecache (file) pages. KSM's merged pages were originally locked into kernel memory, but can now be swapped out just like other user pages (but sharing is broken when they are swapped back in: ksmd must rediscover their identity and merge again).

60.6.2 Controlling KSM with madvise

KSM only operates on those areas of address space which an application has advised to be likely candidates for merging, by using the madvise(2) system call:

```
int madvise(addr, length, MADV_MERGEABLE)
```

The app may call

```
int madvise(addr, length, MADV_UNMERGEABLE)
```

to cancel that advice and restore unshared pages: whereupon KSM unmerges whatever it merged in that range. Note: this unmerging call may suddenly require more memory than is available - possibly failing with EAGAIN, but more probably arousing the Out-Of-Memory killer.

If KSM is not configured into the running kernel, madvise MADV_MERGEABLE and MADV_UNMERGEABLE simply fail with EINVAL. If the running kernel was built with CONFIG_KSM=y, those calls will normally succeed: even if the KSM daemon is not currently running, MADV_MERGEABLE still registers the range for whenever the KSM daemon is started; even if the range cannot contain any pages which KSM could actually merge; even if MADV_UNMERGEABLE is applied to a range which was never MADV_MERGEABLE.

If a region of memory must be split into at least one new MADV_MERGEABLE or MADV_UNMERGEABLE region, the madvise may return ENOMEM if the process will exceed `vm.max_map_count` (see *Documentation for /proc/sys/vm/*).

Like other madvise calls, they are intended for use on mapped areas of the user address space: they will report ENOMEM if the specified range includes unmapped gaps (though working on the intervening mapped areas), and might fail with EAGAIN if not enough memory for internal structures.

Applications should be considerate in their use of MADV_MERGEABLE, restricting its use to areas likely to benefit. KSM's scans may use a lot of processing power: some installations will disable KSM for that reason.

60.6.3 KSM daemon sysfs interface

The KSM daemon is controlled by sysfs files in `/sys/kernel/mm/ksm/`, readable by all but writable only by root:

pages_to_scan
: how many pages to scan before ksmd goes to sleep e.g. `echo 100 > /sys/kernel/mm/ksm/pages_to_scan`.

  The pages_to_scan value cannot be changed if `advisor_mode` has been set to scan-time.

  Default: 100 (chosen for demonstration purposes)

sleep_millisecs
: how many milliseconds ksmd should sleep before next scan e.g. `echo 20 > /sys/kernel/mm/ksm/sleep_millisecs`

  Default: 20 (chosen for demonstration purposes)

merge_across_nodes
specifies if pages from different NUMA nodes can be merged. When set to 0, ksm merges only pages which physically reside in the memory area of same NUMA node. That brings lower latency to access of shared pages. Systems with more nodes, at significant NUMA distances, are likely to benefit from the lower latency of setting 0. Smaller systems, which need to minimize memory usage, are likely to benefit from the greater sharing of setting 1 (default). You may wish to compare how your system performs under each setting, before deciding on which to use. `merge_across_nodes` setting can be changed only when there are no ksm shared pages in the system: set run 2 to unmerge pages first, then to 1 after changing `merge_across_nodes`, to remerge according to the new setting.

Default: 1 (merging across nodes as in earlier releases)

run

- set to 0 to stop ksmd from running but keep merged pages,
- set to 1 to run ksmd e.g. `echo 1 > /sys/kernel/mm/ksm/run`,
- set to 2 to stop ksmd and unmerge all pages currently merged, but leave mergeable areas registered for next run.

Default: 0 (must be changed to 1 to activate KSM, except if CONFIG_SYSFS is disabled)

use_zero_pages
specifies whether empty pages (i.e. allocated pages that only contain zeroes) should be treated specially. When set to 1, empty pages are merged with the kernel zero page(s) instead of with each other as it would happen normally. This can improve the performance on architectures with coloured zero pages, depending on the workload. Care should be taken when enabling this setting, as it can potentially degrade the performance of KSM for some workloads, for example if the checksums of pages candidate for merging match the checksum of an empty page. This setting can be changed at any time, it is only effective for pages merged after the change.

Default: 0 (normal KSM behaviour as in earlier releases)

max_page_sharing
Maximum sharing allowed for each KSM page. This enforces a deduplication limit to avoid high latency for virtual memory operations that involve traversal of the virtual mappings that share the KSM page. The minimum value is 2 as a newly created KSM page will have at least two sharers. The higher this value the faster KSM will merge the memory and the higher the deduplication factor will be, but the slower the worst case virtual mappings traversal could be for any given KSM page. Slowing down this traversal means there will be higher latency for certain virtual memory operations happening during swapping, compaction, NUMA balancing and page migration, in turn decreasing responsiveness for the caller of those virtual memory operations. The scheduler latency of other tasks not involved with the VM operations doing the virtual mappings traversal is not affected by this parameter as these traversals are always schedule friendly themselves.

stable_node_chains_prune_millisecs
specifies how frequently KSM checks the metadata of the pages that hit the deduplication limit for stale information. Smaller milllisecs values will free up the KSM metadata with lower latency, but they will make ksmd use more CPU during the scan. It's a noop if not a single KSM page hit the `max_page_sharing` yet.

smart_scan
Historically KSM checked every candidate page for each scan. It did not take into account

historic information. When smart scan is enabled, pages that have previously not been de-duplicated get skipped. How often these pages are skipped depends on how often de-duplication has already been tried and failed. By default this optimization is enabled. The `pages_skipped` metric shows how effective the setting is.

advisor_mode
The `advisor_mode` selects the current advisor. Two modes are supported: none and scan-time. The default is none. By setting `advisor_mode` to scan-time, the scan time advisor is enabled. The section about `advisor` explains in detail how the scan time advisor works.

adivsor_max_cpu
specifies the upper limit of the cpu percent usage of the ksmd background thread. The default is 70.

advisor_target_scan_time
specifies the target scan time in seconds to scan all the candidate pages. The default value is 200 seconds.

advisor_min_pages_to_scan
specifies the lower limit of the `pages_to_scan` parameter of the scan time advisor. The default is 500.

adivsor_max_pages_to_scan
specifies the upper limit of the `pages_to_scan` parameter of the scan time advisor. The default is 30000.

The effectiveness of KSM and MADV_MERGEABLE is shown in `/sys/kernel/mm/ksm/`:

general_profit
how effective is KSM. The calculation is explained below.

pages_scanned
how many pages are being scanned for ksm

pages_shared
how many shared pages are being used

pages_sharing
how many more sites are sharing them i.e. how much saved

pages_unshared
how many pages unique but repeatedly checked for merging

pages_volatile
how many pages changing too fast to be placed in a tree

pages_skipped
how many pages did the ”smart” page scanning algorithm skip

full_scans
how many times all mergeable areas have been scanned

stable_node_chains
the number of KSM pages that hit the `max_page_sharing` limit

stable_node_dups
number of duplicated KSM pages

ksm_zero_pages
how many zero pages that are still mapped into processes were mapped by KSM when deduplicating.

When `use_zero_pages` is/was enabled, the sum of `pages_sharing + ksm_zero_pages` represents the actual number of pages saved by KSM. if `use_zero_pages` has never been enabled, `ksm_zero_pages` is 0.

A high ratio of `pages_sharing` to `pages_shared` indicates good sharing, but a high ratio of `pages_unshared` to `pages_sharing` indicates wasted effort. `pages_volatile` embraces several different kinds of activity, but a high proportion there would also indicate poor use of madvise MADV_MERGEABLE.

The maximum possible `pages_sharing/pages_shared` ratio is limited by the `max_page_sharing` tunable. To increase the ratio `max_page_sharing` must be increased accordingly.

60.6.4 Monitoring KSM profit

KSM can save memory by merging identical pages, but also can consume additional memory, because it needs to generate a number of rmap_items to save each scanned page's brief rmap information. Some of these pages may be merged, but some may not be abled to be merged after being checked several times, which are unprofitable memory consumed.

1) How to determine whether KSM save memory or consume memory in system-wide range? Here is a simple approximate calculation for reference:

```
general_profit =~ ksm_saved_pages * sizeof(page) - (all_rmap_items) *
                  sizeof(rmap_item);
```

   where ksm_saved_pages equals to the sum of `pages_sharing + ksm_zero_pages` of the system, and all_rmap_items can be easily obtained by summing `pages_sharing`, `pages_shared`, `pages_unshared` and `pages_volatile`.

2) The KSM profit inner a single process can be similarly obtained by the following approximate calculation:

```
process_profit =~ ksm_saved_pages * sizeof(page) -
                  ksm_rmap_items * sizeof(rmap_item).
```

   where ksm_saved_pages equals to the sum of `ksm_merging_pages` and `ksm_zero_pages`, both of which are shown under the directory `/proc/<pid>/ksm_stat`, and ksm_rmap_items is also shown in `/proc/<pid>/ksm_stat`. The process profit is also shown in `/proc/<pid>/ksm_stat` as ksm_process_profit.

From the perspective of application, a high ratio of `ksm_rmap_items` to `ksm_merging_pages` means a bad madvise-applied policy, so developers or administrators have to rethink how to change madvise policy. Giving an example for reference, a page's size is usually 4K, and the rmap_item's size is separately 32B on 32-bit CPU architecture and 64B on 64-bit CPU architecture. so if the `ksm_rmap_items/ksm_merging_pages` ratio exceeds 64 on 64-bit CPU or exceeds 128 on 32-bit CPU, then the app's madvise policy should be dropped, because the ksm profit is approximately zero or negative.

60.6.5 Monitoring KSM events

There are some counters in /proc/vmstat that may be used to monitor KSM events. KSM might help save memory, it's a tradeoff by may suffering delay on KSM COW or on swapping in copy. Those events could help users evaluate whether or how to use KSM. For example, if cow_ksm increases too fast, user may decrease the range of madvise(, , MADV_MERGEABLE).

cow_ksm
is incremented every time a KSM page triggers copy on write (COW) when users try to write to a KSM page, we have to make a copy.

ksm_swpin_copy
is incremented every time a KSM page is copied when swapping in note that KSM page might be copied when swapping in because do_swap_page() cannot do all the locking needed to reconstitute a cross-anon_vma KSM page.

60.6.6 Advisor

The number of candidate pages for KSM is dynamic. It can be often observed that during the startup of an application more candidate pages need to be processed. Without an advisor the `pages_to_scan` parameter needs to be sized for the maximum number of candidate pages. The scan time advisor can changes the `pages_to_scan` parameter based on demand.

The advisor can be enabled, so KSM can automatically adapt to changes in the number of candidate pages to scan. Two advisors are implemented: none and scan-time. With none, no advisor is enabled. The default is none.

The scan time advisor changes the `pages_to_scan` parameter based on the observed scan times. The possible values for the `pages_to_scan` parameter is limited by the `advisor_max_cpu` parameter. In addition there is also the `advisor_target_scan_time` parameter. This parameter sets the target time to scan all the KSM candidate pages. The parameter `advisor_target_scan_time` decides how aggressive the scan time advisor scans candidate pages. Lower values make the scan time advisor to scan more aggresively. This is the most important parameter for the configuration of the scan time advisor.

The initial value and the maximum value can be changed with `advisor_min_pages_to_scan` and `advisor_max_pages_to_scan`. The default values are sufficient for most workloads and use cases.

The `pages_to_scan` parameter is re-calculated after a scan has been completed.

-- Izik Eidus, Hugh Dickins, 17 Nov 2009

60.7 Memory Hot(Un)Plug

This document describes generic Linux support for memory hot(un)plug with a focus on System RAM, including ZONE_MOVABLE support.

- *Introduction*
  - *Memory Hot(Un)Plug Granularity*

  - *Phases of Memory Hotplug*
  - *Phases of Memory Hotunplug*
- *Memory Hotplug Notifications*
  - *ACPI Notifications*
  - *Manual Probing*
- *Onlining and Offlining Memory Blocks*
  - *Onlining Memory Blocks Manually*
  - *Onlining Memory Blocks Automatically*
  - *Offlining Memory Blocks*
  - *Observing the State of Memory Blocks*
- *Configuring Memory Hot(Un)Plug*
  - *Memory Hot(Un)Plug Configuration via Sysfs*
  - *Memory Block Configuration via Sysfs*
  - *Command Line Parameters*
  - *Module Parameters*
- *ZONE_MOVABLE*
  - *Zone Imbalances*
  - *ZONE_MOVABLE Sizing Considerations*
  - *Memory Offlining and ZONE_MOVABLE*

60.7.1 Introduction

Memory hot(un)plug allows for increasing and decreasing the size of physical memory available to a machine at runtime. In the simplest case, it consists of physically plugging or unplugging a DIMM at runtime, coordinated with the operating system.

Memory hot(un)plug is used for various purposes:

- The physical memory available to a machine can be adjusted at runtime, up- or downgrading the memory capacity. This dynamic memory resizing, sometimes referred to as "capacity on demand", is frequently used with virtual machines and logical partitions.
- Replacing hardware, such as DIMMs or whole NUMA nodes, without downtime. One example is replacing failing memory modules.
- Reducing energy consumption either by physically unplugging memory modules or by logically unplugging (parts of) memory modules from Linux.

Further, the basic memory hot(un)plug infrastructure in Linux is nowadays also used to expose persistent memory, other performance-differentiated memory and reserved memory regions as ordinary system RAM to Linux.

Linux only supports memory hot(un)plug on selected 64 bit architectures, such as x86_64, arm64, ppc64 and s390x.

Memory Hot(Un)Plug Granularity

Memory hot(un)plug in Linux uses the SPARSEMEM memory model, which divides the physical memory address space into chunks of the same size: memory sections. The size of a memory section is architecture dependent. For example, x86_64 uses 128 MiB and ppc64 uses 16 MiB.

Memory sections are combined into chunks referred to as "memory blocks". The size of a memory block is architecture dependent and corresponds to the smallest granularity that can be hot(un)plugged. The default size of a memory block is the same as memory section size, unless an architecture specifies otherwise.

All memory blocks have the same size.

Phases of Memory Hotplug

Memory hotplug consists of two phases:

(1) Adding the memory to Linux

(2) Onlining memory blocks

In the first phase, metadata, such as the memory map ("memmap") and page tables for the direct mapping, is allocated and initialized, and memory blocks are created; the latter also creates sysfs files for managing newly created memory blocks.

In the second phase, added memory is exposed to the page allocator. After this phase, the memory is visible in memory statistics, such as free and total memory, of the system.

Phases of Memory Hotunplug

Memory hotunplug consists of two phases:

(1) Offlining memory blocks

(2) Removing the memory from Linux

In the first phase, memory is "hidden" from the page allocator again, for example, by migrating busy memory to other memory locations and removing all relevant free pages from the page allocator After this phase, the memory is no longer visible in memory statistics of the system.

In the second phase, the memory blocks are removed and metadata is freed.

60.7.2 Memory Hotplug Notifications

There are various ways how Linux is notified about memory hotplug events such that it can start adding hotplugged memory. This description is limited to systems that support ACPI; mechanisms specific to other firmware interfaces or virtual machines are not described.

ACPI Notifications

Platforms that support ACPI, such as x86_64, can support memory hotplug notifications via ACPI.

In general, a firmware supporting memory hotplug defines a memory class object HID "PNP0C80". When notified about hotplug of a new memory device, the ACPI driver will hotplug the memory to Linux.

If the firmware supports hotplug of NUMA nodes, it defines an object _HID "ACPI0004", "PNP0A05", or "PNP0A06". When notified about an hotplug event, all assigned memory devices are added to Linux by the ACPI driver.

Similarly, Linux can be notified about requests to hotunplug a memory device or a NUMA node via ACPI. The ACPI driver will try offlining all relevant memory blocks, and, if successful, hotunplug the memory from Linux.

Manual Probing

On some architectures, the firmware may not be able to notify the operating system about a memory hotplug event. Instead, the memory has to be manually probed from user space.

The probe interface is located at:

```
/sys/devices/system/memory/probe
```

Only complete memory blocks can be probed. Individual memory blocks are probed by providing the physical start address of the memory block:

```
% echo addr > /sys/devices/system/memory/probe
```

Which results in a memory block for the range [addr, addr + memory_block_size) being created.

Note: Using the probe interface is discouraged as it is easy to crash the kernel, because Linux cannot validate user input; this interface might be removed in the future.

60.7.3 Onlining and Offlining Memory Blocks

After a memory block has been created, Linux has to be instructed to actually make use of that memory: the memory block has to be "online".

Before a memory block can be removed, Linux has to stop using any memory part of the memory block: the memory block has to be "offlined".

The Linux kernel can be configured to automatically online added memory blocks and drivers automatically trigger offlining of memory blocks when trying hotunplug of memory. Memory blocks can only be removed once offlining succeeded and drivers may trigger offlining of memory blocks when attempting hotunplug of memory.

Onlining Memory Blocks Manually

If auto-onlining of memory blocks isn't enabled, user-space has to manually trigger onlining of memory blocks. Often, udev rules are used to automate this task in user space.

Onlining of a memory block can be triggered via:

```
% echo online > /sys/devices/system/memory/memoryXXX/state
```

Or alternatively:

```
% echo 1 > /sys/devices/system/memory/memoryXXX/online
```

The kernel will select the target zone automatically, depending on the configured `online_policy`.

One can explicitly request to associate an offline memory block with ZONE_MOVABLE by:

```
% echo online_movable > /sys/devices/system/memory/memoryXXX/state
```

Or one can explicitly request a kernel zone (usually ZONE_NORMAL) by:

```
% echo online_kernel > /sys/devices/system/memory/memoryXXX/state
```

In any case, if onlining succeeds, the state of the memory block is changed to be ”online”. If it fails, the state of the memory block will remain unchanged and the above commands will fail.

Onlining Memory Blocks Automatically

The kernel can be configured to try auto-onlining of newly added memory blocks. If this feature is disabled, the memory blocks will stay offline until explicitly onlined from user space.

The configured auto-online behavior can be observed via:

```
% cat /sys/devices/system/memory/auto_online_blocks
```

Auto-onlining can be enabled by writing `online`, `online_kernel` or `online_movable` to that file, like:

```
% echo online > /sys/devices/system/memory/auto_online_blocks
```

Similarly to manual onlining, with `online` the kernel will select the target zone automatically, depending on the configured `online_policy`.

Modifying the auto-online behavior will only affect all subsequently added memory blocks only.

Note: In corner cases, auto-onlining can fail. The kernel won't retry. Note that auto-onlining is not expected to fail in default configurations.

Note: DLPAR on ppc64 ignores the `offline` setting and will still online added memory blocks; if onlining fails, memory blocks are removed again.

Offlining Memory Blocks

In the current implementation, Linux's memory offlining will try migrating all movable pages off the affected memory block. As most kernel allocations, such as page tables, are unmovable, page migration can fail and, therefore, inhibit memory offlining from succeeding.

Having the memory provided by memory block managed by ZONE_MOVABLE significantly increases memory offlining reliability; still, memory offlining can fail in some corner cases.

Further, memory offlining might retry for a long time (or even forever), until aborted by the user.

Offlining of a memory block can be triggered via:

```
% echo offline > /sys/devices/system/memory/memoryXXX/state
```

Or alternatively:

```
% echo 0 > /sys/devices/system/memory/memoryXXX/online
```

If offlining succeeds, the state of the memory block is changed to be "offline". If it fails, the state of the memory block will remain unchanged and the above commands will fail, for example, via:

```
bash: echo: write error: Device or resource busy
```

or via:

```
bash: echo: write error: Invalid argument
```

Observing the State of Memory Blocks

The state (online/offline/going-offline) of a memory block can be observed either via:

```
% cat /sys/devices/system/memory/memoryXXX/state
```

Or alternatively (1/0) via:

```
% cat /sys/devices/system/memory/memoryXXX/online
```

For an online memory block, the managing zone can be observed via:

```
% cat /sys/devices/system/memory/memoryXXX/valid_zones
```

60.7.4 Configuring Memory Hot(Un)Plug

There are various ways how system administrators can configure memory hot(un)plug and interact with memory blocks, especially, to online them.

Memory Hot(Un)Plug Configuration via Sysfs

Some memory hot(un)plug properties can be configured or inspected via sysfs in:

```
/sys/devices/system/memory/
```

The following files are currently defined:

`auto_online_blocks`	read-write: set or get the default state of new memory blocks; configure auto-onlining. The default value depends on the CONFIG_MEMORY_HOTPLUG_DEFAULT_ONLINE kernel configuration option. See the `state` property of memory blocks for details.
`block_size_bytes`	read-only: the size in bytes of a memory block.
`probe`	write-only: add (probe) selected memory blocks manually from user space by supplying the physical start address. Availability depends on the CONFIG_ARCH_MEMORY_PROBE kernel configuration option.
`uevent`	read-write: generic udev file for device subsystems.
`crash_hotplug`	read-only: when changes to the system memory map occur due to hot un/plug of memory, this file contains '1' if the kernel updates the kdump capture kernel memory map itself (via elfcorehdr), or '0' if userspace must update the kdump capture kernel memory map. Availability depends on the CONFIG_MEMORY_HOTPLUG kernel configuration option.

Note: When the CONFIG_MEMORY_FAILURE kernel configuration option is enabled, two additional files `hard_offline_page` and `soft_offline_page` are available to trigger hwpoisoning of pages, for example, for testing purposes. Note that this functionality is not really related to memory hot(un)plug or actual offlining of memory blocks.

Memory Block Configuration via Sysfs

Each memory block is represented as a memory block device that can be onlined or offlined. All memory blocks have their device information located in sysfs. Each present memory block is listed under `/sys/devices/system/memory` as:

```
/sys/devices/system/memory/memoryXXX
```

where XXX is the memory block id; the number of digits is variable.

A present memory block indicates that some memory in the range is present; however, a memory block might span memory holes. A memory block spanning memory holes cannot be offlined.

For example, assume 1 GiB memory block size. A device for a memory starting at 0x100000000 is `/sys/devices/system/memory/memory4`:

```
(0x100000000 / 1Gib = 4)
```

This device covers address range [0x100000000 ... 0x140000000)

The following files are currently defined:

online	read-write: simplified interface to trigger onlining / offlining and to observe the state of a memory block. When onlining, the zone is selected automatically.
phys_device	read-only: legacy interface only ever used on s390x to expose the covered storage increment.
phys_index	read-only: the memory block id (XXX).
removable	read-only: legacy interface that indicated whether a memory block was likely to be offlineable or not. Nowadays, the kernel return 1 if and only if it supports memory offlining.
state	read-write: advanced interface to trigger onlining / offlining and to observe the state of a memory block. When writing, online, offline, online_kernel and online_movable are supported. online_movable specifies onlining to ZONE_MOVABLE. online_kernel specifies onlining to the default kernel zone for the memory block, such as ZONE_NORMAL. online let's the kernel select the zone automatically. When reading, online, offline and going-offline may be returned.
uevent	read-write: generic uevent file for devices.
valid_zones	read-only: when a block is online, shows the zone it belongs to; when a block is offline, shows what zone will manage it when the block will be onlined. For online memory blocks, DMA, DMA32, Normal, Movable and none may be returned. none indicates that memory provided by a memory block is managed by multiple zones or spans multiple nodes; such memory blocks cannot be offlined. Movable indicates ZONE_MOVABLE. Other values indicate a kernel zone. For offline memory blocks, the first column shows the zone the kernel would select when onlining the memory block right now without further specifying a zone. Availability depends on the CONFIG_MEMORY_HOTREMOVE kernel configuration option.

Note: If the CONFIG_NUMA kernel configuration option is enabled, the memoryXXX/ directories can also be accessed via symbolic links located in the /sys/devices/system/node/node* directories.

For example:

```
/sys/devices/system/node/node0/memory9 -> ../../memory/memory9
```

A backlink will also be created:

```
/sys/devices/system/memory/memory9/node0 -> ../../node/node0
```

Command Line Parameters

Some command line parameters affect memory hot(un)plug handling. The following command line parameters are relevant:

`memhp_default_state`	configure auto-onlining by essentially setting `/sys/devices/system/memory/auto_online_blocks`.
`movable_node`	configure automatic zone selection in the kernel when using the `contig-zones` online policy. When set, the kernel will default to ZONE_MOVABLE when onlining a memory block, unless other zones can be kept contiguous.

See *The kernel's command-line parameters* for a more generic description of these command line parameters.

Module Parameters

Instead of additional command line parameters or sysfs files, the `memory_hotplug` subsystem now provides a dedicated namespace for module parameters. Module parameters can be set via the command line by predicating them with `memory_hotplug.` such as:

```
memory_hotplug.memmap_on_memory=1
```

and they can be observed (and some even modified at runtime) via:

```
/sys/module/memory_hotplug/parameters/
```

The following module parameters are currently defined:

`memmap_on_memory`	read-write: Allocate memory for the memmap from the added memory block itself. Even if enabled, actual support depends on various other system properties and should only be regarded as a hint whether the behavior would be desired. While allocating the memmap from the memory block itself makes memory hotplug less likely to fail and keeps the memmap on the same NUMA node in any case, it can fragment physical memory in a way that huge pages in bigger granularity cannot be formed on hotplugged memory. With value "force" it could result in memory wastage due to memmap size limitations. For example, if the memmap for a memory block requires 1 MiB, but the pageblock size is 2 MiB, 1 MiB of hotplugged memory will be wasted. Note that there are still cases where the feature cannot be enforced: for example, if the memmap is smaller than a single page, or if the architecture does not support the forced mode in all configurations.
`online_policy`	read-write: Set the basic policy used for automatic zone selection when onlining memory blocks without specifying a target zone. `contig-zones` has been the kernel default before this parameter was added. After an online policy was configured and memory was online, the policy should not be changed anymore. When set to `contig-zones`, the kernel will try keeping zones contiguous. If a memory block intersects multiple zones or no zone, the behavior depends on the `movable_node` kernel command line parameter: default to ZONE_MOVABLE if set, default to the applicable kernel zone (usually ZONE_NORMAL) if not set. When set to `auto-movable`, the kernel will try onlining memory blocks to ZONE_MOVABLE if possible according to the configuration and memory device details. With this policy, one can avoid zone imbalances when eventually hotplugging a lot of memory later and still wanting to be able to hotunplug as much as possible reliably, very desirable in virtualized environments. This policy ignores the `movable_node` kernel command line parameter and isn't really applicable in environments that require it (e.g., bare metal with hotunpluggable nodes) where hotplugged memory might be exposed via the firmware-provided memory map early during boot to the system instead of getting detected, added and onlined later during boot (such as done by virtio-mem or by some hypervisors implementing emulated DIMMs). As one example, a hotplugged DIMM will be onlined either completely to ZONE_MOVABLE or completely to ZONE_NORMAL, not a mixture. As another example, as many memory blocks belonging to a virtio-mem device will be onlined to ZONE_MOVABLE as possible, special-casing units of memory blocks that can only get hotunplugged together. *This policy does not protect from setups that are problematic with ZONE_MOVABLE and does not change the zone of memory blocks dynamically after they were onlined.*
`auto_movable_ratio`	read-write: Set the maximum MOVABLE:KERNEL memory ratio in % for the `auto-movable` online policy. Whether the ratio applies only for the system across all NUMA nodes or also per NUMA nodes depends on the `auto_movable_numa_aware` configuration. All accounting is based on present memory pages in the zones combined with accounting per memory device. Memory dedi-

60.7.5 ZONE_MOVABLE

ZONE_MOVABLE is an important mechanism for more reliable memory offlining. Further, having system RAM managed by ZONE_MOVABLE instead of one of the kernel zones can increase the number of possible transparent huge pages and dynamically allocated huge pages.

Most kernel allocations are unmovable. Important examples include the memory map (usually 1/64ths of memory), page tables, and kmalloc(). Such allocations can only be served from the kernel zones.

Most user space pages, such as anonymous memory, and page cache pages are movable. Such allocations can be served from ZONE_MOVABLE and the kernel zones.

Only movable allocations are served from ZONE_MOVABLE, resulting in unmovable allocations being limited to the kernel zones. Without ZONE_MOVABLE, there is absolutely no guarantee whether a memory block can be offlined successfully.

Zone Imbalances

Having too much system RAM managed by ZONE_MOVABLE is called a zone imbalance, which can harm the system or degrade performance. As one example, the kernel might crash because it runs out of free memory for unmovable allocations, although there is still plenty of free memory left in ZONE_MOVABLE.

Usually, MOVABLE:KERNEL ratios of up to 3:1 or even 4:1 are fine. Ratios of 63:1 are definitely impossible due to the overhead for the memory map.

Actual safe zone ratios depend on the workload. Extreme cases, like excessive long-term pinning of pages, might not be able to deal with ZONE_MOVABLE at all.

Note: CMA memory part of a kernel zone essentially behaves like memory in ZONE_MOVABLE and similar considerations apply, especially when combining CMA with ZONE_MOVABLE.

ZONE_MOVABLE Sizing Considerations

We usually expect that a large portion of available system RAM will actually be consumed by user space, either directly or indirectly via the page cache. In the normal case, ZONE_MOVABLE can be used when allocating such pages just fine.

With that in mind, it makes sense that we can have a big portion of system RAM managed by ZONE_MOVABLE. However, there are some things to consider when using ZONE_MOVABLE, especially when fine-tuning zone ratios:

- Having a lot of offline memory blocks. Even offline memory blocks consume memory for metadata and page tables in the direct map; having a lot of offline memory blocks is not a typical case, though.
- Memory ballooning without balloon compaction is incompatible with ZONE_MOVABLE. Only some implementations, such as virtio-balloon and pseries CMM, fully support balloon compaction.

Further, the CONFIG_BALLOON_COMPACTION kernel configuration option might be disabled. In that case, balloon inflation will only perform unmovable allocations and silently create a zone imbalance, usually triggered by inflation requests from the hypervisor.

- Gigantic pages are unmovable, resulting in user space consuming a lot of unmovable memory.
- Huge pages are unmovable when an architectures does not support huge page migration, resulting in a similar issue as with gigantic pages.
- Page tables are unmovable. Excessive swapping, mapping extremely large files or ZONE_DEVICE memory can be problematic, although only really relevant in corner cases. When we manage a lot of user space memory that has been swapped out or is served from a file/persistent memory/... we still need a lot of page tables to manage that memory once user space accessed that memory.
- In certain DAX configurations the memory map for the device memory will be allocated from the kernel zones.
- KASAN can have a significant memory overhead, for example, consuming 1/8th of the total system memory size as (unmovable) tracking metadata.
- Long-term pinning of pages. Techniques that rely on long-term pinnings (especially, RDMA and vfio/mdev) are fundamentally problematic with ZONE_MOVABLE, and therefore, memory offlining. Pinned pages cannot reside on ZONE_MOVABLE as that would turn these pages unmovable. Therefore, they have to be migrated off that zone while pinning. Pinning a page can fail even if there is plenty of free memory in ZONE_MOVABLE.

  In addition, using ZONE_MOVABLE might make page pinning more expensive, because of the page migration overhead.

By default, all the memory configured at boot time is managed by the kernel zones and ZONE_MOVABLE is not used.

To enable ZONE_MOVABLE to include the memory present at boot and to control the ratio between movable and kernel zones there are two command line options: `kernelcore=` and `movablecore=`. See *The kernel's command-line parameters* for their description.

Memory Offlining and ZONE_MOVABLE

Even with ZONE_MOVABLE, there are some corner cases where offlining a memory block might fail:

- Memory blocks with memory holes; this applies to memory blocks present during boot and can apply to memory blocks hotplugged via the XEN balloon and the Hyper-V balloon.
- Mixed NUMA nodes and mixed zones within a single memory block prevent memory offlining; this applies to memory blocks present during boot only.
- Special memory blocks prevented by the system from getting offlined. Examples include any memory available during boot on arm64 or memory blocks spanning the crashkernel area on s390x; this usually applies to memory blocks present during boot only.
- Memory blocks overlapping with CMA areas cannot be offlined, this applies to memory blocks present during boot only.
- Concurrent activity that operates on the same physical memory area, such as allocating gigantic pages, can result in temporary offlining failures.

- Out of memory when dissolving huge pages, especially when HugeTLB Vmemmap Optimization (HVO) is enabled.

  Offlining code may be able to migrate huge page contents, but may not be able to dissolve the source huge page because it fails allocating (unmovable) pages for the vmemmap, because the system might not have free memory in the kernel zones left.

  Users that depend on memory offlining to succeed for movable zones should carefully consider whether the memory savings gained from this feature are worth the risk of possibly not being able to offline memory in certain situations.

Further, when running into out of memory situations while migrating pages, or when still encountering permanently unmovable pages within ZONE_MOVABLE (-> BUG), memory offlining will keep retrying until it eventually succeeds.

When offlining is triggered from user space, the offlining context can be terminated by sending a signal. A timeout based offlining can easily be implemented via:

```
% timeout $TIMEOUT offline_block | failure_handling
```

60.8 Multi-Gen LRU

The multi-gen LRU is an alternative LRU implementation that optimizes page reclaim and improves performance under memory pressure. Page reclaim decides the kernel's caching policy and ability to overcommit memory. It directly impacts the kswapd CPU usage and RAM efficiency.

60.8.1 Quick start

Build the kernel with the following configurations.

- `CONFIG_LRU_GEN=y`
- `CONFIG_LRU_GEN_ENABLED=y`

All set!

60.8.2 Runtime options

`/sys/kernel/mm/lru_gen/` contains stable ABIs described in the following subsections.

Kill switch

`enabled` accepts different values to enable or disable the following components. Its default value depends on `CONFIG_LRU_GEN_ENABLED`. All the components should be enabled unless some of them have unforeseen side effects. Writing to `enabled` has no effect when a component is not supported by the hardware, and valid values will be accepted even when the main switch is off.

Values	Components
0x0001	The main switch for the multi-gen LRU.
0x0002	Clearing the accessed bit in leaf page table entries in large batches, when MMU sets it (e.g., on x86). This behavior can theoretically worsen lock contention (mmap_lock). If it is disabled, the multi-gen LRU will suffer a minor performance degradation for workloads that contiguously map hot pages, whose accessed bits can be otherwise cleared by fewer larger batches.
0x0004	Clearing the accessed bit in non-leaf page table entries as well, when MMU sets it (e.g., on x86). This behavior was not verified on x86 varieties other than Intel and AMD. If it is disabled, the multi-gen LRU will suffer a negligible performance degradation.
[yYnN]	Apply to all the components above.

E.g.,

```
echo y >/sys/kernel/mm/lru_gen/enabled
cat /sys/kernel/mm/lru_gen/enabled
0x0007
echo 5 >/sys/kernel/mm/lru_gen/enabled
cat /sys/kernel/mm/lru_gen/enabled
0x0005
```

Thrashing prevention

Personal computers are more sensitive to thrashing because it can cause janks (lags when rendering UI) and negatively impact user experience. The multi-gen LRU offers thrashing prevention to the majority of laptop and desktop users who do not have `oomd`.

Users can write `N` to `min_ttl_ms` to prevent the working set of `N` milliseconds from getting evicted. The OOM killer is triggered if this working set cannot be kept in memory. In other words, this option works as an adjustable pressure relief valve, and when open, it terminates applications that are hopefully not being used.

Based on the average human detectable lag (~100ms), `N=1000` usually eliminates intolerable janks due to thrashing. Larger values like `N=3000` make janks less noticeable at the risk of premature OOM kills.

The default value `0` means disabled.

60.8.3 Experimental features

`/sys/kernel/debug/lru_gen` accepts commands described in the following subsections. Multiple command lines are supported, so does concatenation with delimiters `,` and `;`.

`/sys/kernel/debug/lru_gen_full` provides additional stats for debugging. `CONFIG_LRU_GEN_STATS=y` keeps historical stats from evicted generations in this file.

Working set estimation

Working set estimation measures how much memory an application needs in a given time interval, and it is usually done with little impact on the performance of the application. E.g., data centers want to optimize job scheduling (bin packing) to improve memory utilizations. When a new job comes in, the job scheduler needs to find out whether each server it manages can allocate a certain amount of memory for this new job before it can pick a candidate. To do so, the job scheduler needs to estimate the working sets of the existing jobs.

When it is read, `lru_gen` returns a histogram of numbers of pages accessed over different time intervals for each memcg and node. `MAX_NR_GENS` decides the number of bins for each histogram. The histograms are noncumulative.

```
memcg  memcg_id  memcg_path
   node  node_id
       min_gen_nr  age_in_ms  nr_anon_pages  nr_file_pages
       ...
       max_gen_nr  age_in_ms  nr_anon_pages  nr_file_pages
```

Each bin contains an estimated number of pages that have been accessed within `age_in_ms`. E.g., `min_gen_nr` contains the coldest pages and `max_gen_nr` contains the hottest pages, since `age_in_ms` of the former is the largest and that of the latter is the smallest.

Users can write the following command to `lru_gen` to create a new generation `max_gen_nr+1`:

```
+ memcg_id node_id max_gen_nr [can_swap [force_scan]]
```

`can_swap` defaults to the swap setting and, if it is set to `1`, it forces the scan of anon pages when swap is off, and vice versa. `force_scan` defaults to `1` and, if it is set to `0`, it employs heuristics to reduce the overhead, which is likely to reduce the coverage as well.

A typical use case is that a job scheduler runs this command at a certain time interval to create new generations, and it ranks the servers it manages based on the sizes of their cold pages defined by this time interval.

Proactive reclaim

Proactive reclaim induces page reclaim when there is no memory pressure. It usually targets cold pages only. E.g., when a new job comes in, the job scheduler wants to proactively reclaim cold pages on the server it selected, to improve the chance of successfully landing this new job.

Users can write the following command to `lru_gen` to evict generations less than or equal to `min_gen_nr`.

```
- memcg_id node_id min_gen_nr [swappiness [nr_to_reclaim]]
```

`min_gen_nr` should be less than `max_gen_nr-1`, since `max_gen_nr` and `max_gen_nr-1` are not fully aged (equivalent to the active list) and therefore cannot be evicted. `swappiness` overrides the default value in `/proc/sys/vm/swappiness`. `nr_to_reclaim` limits the number of pages to evict.

A typical use case is that a job scheduler runs this command before it tries to land a new job on a server. If it fails to materialize enough cold pages because of the overestimation, it retries on the next server according to the ranking result obtained from the working set estimation step. This less forceful approach limits the impacts on the existing jobs.

60.9 No-MMU memory mapping support

The kernel has limited support for memory mapping under no-MMU conditions, such as are used in uClinux environments. From the userspace point of view, memory mapping is made use of in conjunction with the mmap() system call, the shmat() call and the execve() system call. From the kernel's point of view, execve() mapping is actually performed by the binfmt drivers, which call back into the mmap() routines to do the actual work.

Memory mapping behaviour also involves the way fork(), vfork(), clone() and ptrace() work. Under uClinux there is no fork(), and clone() must be supplied the CLONE_VM flag.

The behaviour is similar between the MMU and no-MMU cases, but not identical; and it's also much more restricted in the latter case:

(1) Anonymous mapping, MAP_PRIVATE

In the MMU case: VM regions backed by arbitrary pages; copy-on-write across fork.

In the no-MMU case: VM regions backed by arbitrary contiguous runs of pages.

(2) Anonymous mapping, MAP_SHARED

These behave very much like private mappings, except that they're shared across fork() or clone() without CLONE_VM in the MMU case. Since the no-MMU case doesn't support these, behaviour is identical to MAP_PRIVATE there.

(3) File, MAP_PRIVATE, PROT_READ / PROT_EXEC, !PROT_WRITE

In the MMU case: VM regions backed by pages read from file; changes to the underlying file are reflected in the mapping; copied across fork.

In the no-MMU case:

- If one exists, the kernel will re-use an existing mapping to the same segment of the same file if that has compatible permissions, even if this was created by another process.
- If possible, the file mapping will be directly on the backing device if the backing device has the NOMMU_MAP_DIRECT capability and appropriate mapping protection capabilities. Ramfs, romfs, cramfs and mtd might all permit this.
- If the backing device can't or won't permit direct sharing, but does have the NOMMU_MAP_COPY capability, then a copy of the appropriate bit of the file will be read into a contiguous bit of memory and any extraneous space beyond the EOF will be cleared
- Writes to the file do not affect the mapping; writes to the mapping are visible in other processes (no MMU protection), but should not happen.

(4) File, MAP_PRIVATE, PROT_READ / PROT_EXEC, PROT_WRITE

In the MMU case: like the non-PROT_WRITE case, except that the pages in question get copied before the write actually happens. From that point on writes to the file underneath that page no longer get reflected into the mapping's backing pages. The page is then backed by swap instead.

In the no-MMU case: works much like the non-PROT_WRITE case, except that a copy is always taken and never shared.

(5) Regular file / blockdev, MAP_SHARED, PROT_READ / PROT_EXEC / PROT_WRITE

In the MMU case: VM regions backed by pages read from file; changes to pages written back to file; writes to file reflected into pages backing mapping; shared across fork.

In the no-MMU case: not supported.

(6) Memory backed regular file, MAP_SHARED, PROT_READ / PROT_EXEC / PROT_WRITE

In the MMU case: As for ordinary regular files.

In the no-MMU case: The filesystem providing the memory-backed file (such as ramfs or tmpfs) may choose to honour an open, truncate, mmap sequence by providing a contiguous sequence of pages to map. In that case, a shared-writable memory mapping will be possible. It will work as for the MMU case. If the filesystem does not provide any such support, then the mapping request will be denied.

(7) Memory backed blockdev, MAP_SHARED, PROT_READ / PROT_EXEC / PROT_WRITE

In the MMU case: As for ordinary regular files.

In the no-MMU case: As for memory backed regular files, but the blockdev must be able to provide a contiguous run of pages without truncate being called. The ramdisk driver could do this if it allocated all its memory as a contiguous array upfront.

(8) Memory backed chardev, MAP_SHARED, PROT_READ / PROT_EXEC / PROT_WRITE

In the MMU case: As for ordinary regular files.

In the no-MMU case: The character device driver may choose to honour the mmap() by providing direct access to the underlying device if it provides memory or quasi-memory that can be accessed directly. Examples of such are frame buffers and flash devices. If the driver does not provide any such support, then the mapping request will be denied.

60.9.1 Further notes on no-MMU MMAP

(1) A request for a private mapping of a file may return a buffer that is not page-aligned. This is because XIP may take place, and the data may not be paged aligned in the backing store.

(2) A request for an anonymous mapping will always be page aligned. If possible the size of the request should be a power of two otherwise some of the space may be wasted as the kernel must allocate a power-of-2 granule but will only discard the excess if appropriately configured as this has an effect on fragmentation.

(3) The memory allocated by a request for an anonymous mapping will normally be cleared by the kernel before being returned in accordance with the Linux man pages (ver 2.22 or later).

In the MMU case this can be achieved with reasonable performance as regions are backed by virtual pages, with the contents only being mapped to cleared physical pages when a write happens on that specific page (prior to which, the pages are effectively mapped to

the global zero page from which reads can take place). This spreads out the time it takes to initialize the contents of a page - depending on the write-usage of the mapping.

In the no-MMU case, however, anonymous mappings are backed by physical pages, and the entire map is cleared at allocation time. This can cause significant delays during a userspace malloc() as the C library does an anonymous mapping and the kernel then does a memset for the entire map.

However, for memory that isn't required to be precleared - such as that returned by malloc() - mmap() can take a MAP_UNINITIALIZED flag to indicate to the kernel that it shouldn't bother clearing the memory before returning it. Note that CONFIG_MMAP_ALLOW_UNINITIALIZED must be enabled to permit this, otherwise the flag will be ignored.

uClibc uses this to speed up malloc(), and the ELF-FDPIC binfmt uses this to allocate the brk and stack region.

(4) A list of all the private copy and anonymous mappings on the system is visible through /proc/maps in no-MMU mode.

(5) A list of all the mappings in use by a process is visible through /proc/<pid>/maps in no-MMU mode.

(6) Supplying MAP_FIXED or a requesting a particular mapping address will result in an error.

(7) Files mapped privately usually have to have a read method provided by the driver or filesystem so that the contents can be read into the memory allocated if mmap() chooses not to map the backing device directly. An error will result if they don't. This is most likely to be encountered with character device files, pipes, fifos and sockets.

60.9.2 Interprocess shared memory

Both SYSV IPC SHM shared memory and POSIX shared memory is supported in NOMMU mode. The former through the usual mechanism, the latter through files created on ramfs or tmpfs mounts.

60.9.3 Futexes

Futexes are supported in NOMMU mode if the arch supports them. An error will be given if an address passed to the futex system call lies outside the mappings made by a process or if the mapping in which the address lies does not support futexes (such as an I/O chardev mapping).

60.9.4 No-MMU mremap

The mremap() function is partially supported. It may change the size of a mapping, and may move it[1] if MREMAP_MAYMOVE is specified and if the new size of the mapping exceeds the size of the slab object currently occupied by the memory to which the mapping refers, or if a smaller slab object could be used.

MREMAP_FIXED is not supported, though it is ignored if there's no change of address and the object does not need to be moved.

[1] Not currently supported.

Shared mappings may not be moved. Shareable mappings may not be moved either, even if they are not currently shared.

The mremap() function must be given an exact match for base address and size of a previously mapped object. It may not be used to create holes in existing mappings, move parts of existing mappings or resize parts of mappings. It must act on a complete mapping.

60.9.5 Providing shareable character device support

To provide shareable character device support, a driver must provide a file->f_op->get_unmapped_area() operation. The mmap() routines will call this to get a proposed address for the mapping. This may return an error if it doesn't wish to honour the mapping because it's too long, at a weird offset, under some unsupported combination of flags or whatever.

The driver should also provide backing device information with capabilities set to indicate the permitted types of mapping on such devices. The default is assumed to be readable and writable, not executable, and only shareable directly (can't be copied).

The file->f_op->mmap() operation will be called to actually inaugurate the mapping. It can be rejected at that point. Returning the ENOSYS error will cause the mapping to be copied instead if NOMMU_MAP_COPY is specified.

The vm_ops->close() routine will be invoked when the last mapping on a chardev is removed. An existing mapping will be shared, partially or not, if possible without notifying the driver.

It is permitted also for the file->f_op->get_unmapped_area() operation to return -ENOSYS. This will be taken to mean that this operation just doesn't want to handle it, despite the fact it's got an operation. For instance, it might try directing the call to a secondary driver which turns out not to implement it. Such is the case for the framebuffer driver which attempts to direct the call to the device-specific driver. Under such circumstances, the mapping request will be rejected if NOMMU_MAP_COPY is not specified, and a copy mapped otherwise.

Important: Some types of device may present a different appearance to anyone looking at them in certain modes. Flash chips can be like this; for instance if they're in programming or erase mode, you might see the status reflected in the mapping, instead of the data.

In such a case, care must be taken lest userspace see a shared or a private mapping showing such information when the driver is busy controlling the device. Remember especially: private executable mappings may still be mapped directly off the device under some circumstances!

60.9.6 Providing shareable memory-backed file support

Provision of shared mappings on memory backed files is similar to the provision of support for shared mapped character devices. The main difference is that the filesystem providing the service will probably allocate a contiguous collection of pages and permit mappings to be made on that.

It is recommended that a truncate operation applied to such a file that increases the file size, if that file is empty, be taken as a request to gather enough pages to honour a mapping. This is required to support POSIX shared memory.

Memory backed devices are indicated by the mapping's backing device info having the memory_backed flag set.

60.9.7 Providing shareable block device support

Provision of shared mappings on block device files is exactly the same as for character devices. If there isn't a real device underneath, then the driver should allocate sufficient contiguous memory to honour any supported mapping.

60.9.8 Adjusting page trimming behaviour

NOMMU mmap automatically rounds up to the nearest power-of-2 number of pages when performing an allocation. This can have adverse effects on memory fragmentation, and as such, is left configurable. The default behaviour is to aggressively trim allocations and discard any excess pages back in to the page allocator. In order to retain finer-grained control over fragmentation, this behaviour can either be disabled completely, or bumped up to a higher page watermark where trimming begins.

Page trimming behaviour is configurable via the sysctl `vm.nr_trim_pages`.

60.10 NUMA Memory Policy

60.10.1 What is NUMA Memory Policy?

In the Linux kernel, ”memory policy” determines from which node the kernel will allocate memory in a NUMA system or in an emulated NUMA system. Linux has supported platforms with Non-Uniform Memory Access architectures since 2.4.?. The current memory policy support was added to Linux 2.6 around May 2004. This document attempts to describe the concepts and APIs of the 2.6 memory policy support.

Memory policies should not be confused with cpusets (`Documentation/admin-guide/cgroup-v1/cpusets.rst`) which is an administrative mechanism for restricting the nodes from which memory may be allocated by a set of processes. Memory policies are a programming interface that a NUMA-aware application can take advantage of. When both cpusets and policies are applied to a task, the restrictions of the cpuset takes priority. See *Memory Policies and cpusets* below for more details.

60.10.2 Memory Policy Concepts

Scope of Memory Policies

The Linux kernel supports _scopes_ of memory policy, described here from most general to most specific:

System Default Policy
this policy is ”hard coded” into the kernel. It is the policy that governs all page allocations that aren't controlled by one of the more specific policy scopes discussed below. When the system is ”up and running”, the system default policy will use ”local allocation” described

below. However, during boot up, the system default policy will be set to interleave allocations across all nodes with "sufficient" memory, so as not to overload the initial boot node with boot-time allocations.

Task/Process Policy

this is an optional, per-task policy. When defined for a specific task, this policy controls all page allocations made by or on behalf of the task that aren't controlled by a more specific scope. If a task does not define a task policy, then all page allocations that would have been controlled by the task policy "fall back" to the System Default Policy.

The task policy applies to the entire address space of a task. Thus, it is inheritable, and indeed is inherited, across both fork() [clone() w/o the CLONE_VM flag] and exec*(). This allows a parent task to establish the task policy for a child task exec()'d from an executable image that has no awareness of memory policy. See the *Memory Policy APIs* section, below, for an overview of the system call that a task may use to set/change its task/process policy.

In a multi-threaded task, task policies apply only to the thread [Linux kernel task] that installs the policy and any threads subsequently created by that thread. Any sibling threads existing at the time a new task policy is installed retain their current policy.

A task policy applies only to pages allocated after the policy is installed. Any pages already faulted in by the task when the task changes its task policy remain where they were allocated based on the policy at the time they were allocated.

VMA Policy

A "VMA" or "Virtual Memory Area" refers to a range of a task's virtual address space. A task may define a specific policy for a range of its virtual address space. See the *Memory Policy APIs* section, below, for an overview of the mbind() system call used to set a VMA policy.

A VMA policy will govern the allocation of pages that back this region of the address space. Any regions of the task's address space that don't have an explicit VMA policy will fall back to the task policy, which may itself fall back to the System Default Policy.

VMA policies have a few complicating details:

- VMA policy applies ONLY to anonymous pages. These include pages allocated for anonymous segments, such as the task stack and heap, and any regions of the address space mmap()ed with the MAP_ANONYMOUS flag. If a VMA policy is applied to a file mapping, it will be ignored if the mapping used the MAP_SHARED flag. If the file mapping used the MAP_PRIVATE flag, the VMA policy will only be applied when an anonymous page is allocated on an attempt to write to the mapping-- i.e., at Copy-On-Write.
- VMA policies are shared between all tasks that share a virtual address space--a.k.a. threads--independent of when the policy is installed; and they are inherited across fork(). However, because VMA policies refer to a specific region of a task's address space, and because the address space is discarded and recreated on exec*(), VMA policies are NOT inheritable across exec(). Thus, only NUMA-aware applications may use VMA policies.
- A task may install a new VMA policy on a sub-range of a previously mmap()ed region. When this happens, Linux splits the existing virtual memory area into 2 or 3 VMAs, each with its own policy.
- By default, VMA policy applies only to pages allocated after the policy is installed. Any pages already faulted into the VMA range remain where they were allocated based on

the policy at the time they were allocated. However, since 2.6.16, Linux supports page migration via the mbind() system call, so that page contents can be moved to match a newly installed policy.

Shared Policy

Conceptually, shared policies apply to ”memory objects” mapped shared into one or more tasks' distinct address spaces. An application installs shared policies the same way as VMA policies--using the mbind() system call specifying a range of virtual addresses that map the shared object. However, unlike VMA policies, which can be considered to be an attribute of a range of a task's address space, shared policies apply directly to the shared object. Thus, all tasks that attach to the object share the policy, and all pages allocated for the shared object, by any task, will obey the shared policy.

As of 2.6.22, only shared memory segments, created by shmget() or mmap(MAP_ANONYMOUS|MAP_SHARED), support shared policy. When shared policy support was added to Linux, the associated data structures were added to hugetlbfs shmem segments. At the time, hugetlbfs did not support allocation at fault time--a.k.a lazy allocation--so hugetlbfs shmem segments were never ”hooked up” to the shared policy support. Although hugetlbfs segments now support lazy allocation, their support for shared policy has not been completed.

As mentioned above in *VMA policies* section, allocations of page cache pages for regular files mmap()ed with MAP_SHARED ignore any VMA policy installed on the virtual address range backed by the shared file mapping. Rather, shared page cache pages, including pages backing private mappings that have not yet been written by the task, follow task policy, if any, else System Default Policy.

The shared policy infrastructure supports different policies on subset ranges of the shared object. However, Linux still splits the VMA of the task that installs the policy for each range of distinct policy. Thus, different tasks that attach to a shared memory segment can have different VMA configurations mapping that one shared object. This can be seen by examining the /proc/<pid>/numa_maps of tasks sharing a shared memory region, when one task has installed shared policy on one or more ranges of the region.

Components of Memory Policies

A NUMA memory policy consists of a ”mode”, optional mode flags, and an optional set of nodes. The mode determines the behavior of the policy, the optional mode flags determine the behavior of the mode, and the optional set of nodes can be viewed as the arguments to the policy behavior.

Internally, memory policies are implemented by a reference counted structure, struct mempolicy. Details of this structure will be discussed in context, below, as required to explain the behavior.

NUMA memory policy supports the following 4 behavioral modes:

Default Mode--MPOL_DEFAULT

This mode is only used in the memory policy APIs. Internally, MPOL_DEFAULT is converted to the NULL memory policy in all policy scopes. Any existing non-default policy will simply be removed when MPOL_DEFAULT is specified. As a result, MPOL_DEFAULT means ”fall back to the next most specific policy scope.”

For example, a NULL or default task policy will fall back to the system default policy. A NULL or default vma policy will fall back to the task policy.

When specified in one of the memory policy APIs, the Default mode does not use the optional set of nodes.

It is an error for the set of nodes specified for this policy to be non-empty.

MPOL_BIND
This mode specifies that memory must come from the set of nodes specified by the policy. Memory will be allocated from the node in the set with sufficient free memory that is closest to the node where the allocation takes place.

MPOL_PREFERRED
This mode specifies that the allocation should be attempted from the single node specified in the policy. If that allocation fails, the kernel will search other nodes, in order of increasing distance from the preferred node based on information provided by the platform firmware.

Internally, the Preferred policy uses a single node--the preferred_node member of struct mempolicy. When the internal mode flag MPOL_F_LOCAL is set, the preferred_node is ignored and the policy is interpreted as local allocation. "Local" allocation policy can be viewed as a Preferred policy that starts at the node containing the cpu where the allocation takes place.

It is possible for the user to specify that local allocation is always preferred by passing an empty nodemask with this mode. If an empty nodemask is passed, the policy cannot use the MPOL_F_STATIC_NODES or MPOL_F_RELATIVE_NODES flags described below.

MPOL_INTERLEAVED
This mode specifies that page allocations be interleaved, on a page granularity, across the nodes specified in the policy. This mode also behaves slightly differently, based on the context where it is used:

For allocation of anonymous pages and shared memory pages, Interleave mode indexes the set of nodes specified by the policy using the page offset of the faulting address into the segment [VMA] containing the address modulo the number of nodes specified by the policy. It then attempts to allocate a page, starting at the selected node, as if the node had been specified by a Preferred policy or had been selected by a local allocation. That is, allocation will follow the per node zonelist.

For allocation of page cache pages, Interleave mode indexes the set of nodes specified by the policy using a node counter maintained per task. This counter wraps around to the lowest specified node after it reaches the highest specified node. This will tend to spread the pages out over the nodes specified by the policy based on the order in which they are allocated, rather than based on any page offset into an address range or file. During system boot up, the temporary interleaved system default policy works in this mode.

MPOL_PREFERRED_MANY
This mode specifies that the allocation should be preferably satisfied from the nodemask specified in the policy. If there is a memory pressure on all nodes in the nodemask, the allocation can fall back to all existing numa nodes. This is effectively MPOL_PREFERRED allowed for a mask rather than a single node.

NUMA memory policy supports the following optional mode flags:

MPOL_F_STATIC_NODES
This flag specifies that the nodemask passed by the user should not be remapped if the task or VMA's set of allowed nodes changes after the memory policy has been defined.

Without this flag, any time a mempolicy is rebound because of a change in the set of allowed nodes, the preferred nodemask (Preferred Many), preferred node (Preferred) or nodemask (Bind, Interleave) is remapped to the new set of allowed nodes. This may result in nodes being used that were previously undesired.

With this flag, if the user-specified nodes overlap with the nodes allowed by the task's cpuset, then the memory policy is applied to their intersection. If the two sets of nodes do not overlap, the Default policy is used.

For example, consider a task that is attached to a cpuset with mems 1-3 that sets an Interleave policy over the same set. If the cpuset's mems change to 3-5, the Interleave will now occur over nodes 3, 4, and 5. With this flag, however, since only node 3 is allowed from the user's nodemask, the ”interleave” only occurs over that node. If no nodes from the user's nodemask are now allowed, the Default behavior is used.

MPOL_F_STATIC_NODES cannot be combined with the MPOL_F_RELATIVE_NODES flag. It also cannot be used for MPOL_PREFERRED policies that were created with an empty nodemask (local allocation).

MPOL_F_RELATIVE_NODES

This flag specifies that the nodemask passed by the user will be mapped relative to the set of the task or VMA's set of allowed nodes. The kernel stores the user-passed nodemask, and if the allowed nodes changes, then that original nodemask will be remapped relative to the new set of allowed nodes.

Without this flag (and without MPOL_F_STATIC_NODES), anytime a mempolicy is rebound because of a change in the set of allowed nodes, the node (Preferred) or nodemask (Bind, Interleave) is remapped to the new set of allowed nodes. That remap may not preserve the relative nature of the user's passed nodemask to its set of allowed nodes upon successive rebinds: a nodemask of 1,3,5 may be remapped to 7-9 and then to 1-3 if the set of allowed nodes is restored to its original state.

With this flag, the remap is done so that the node numbers from the user's passed nodemask are relative to the set of allowed nodes. In other words, if nodes 0, 2, and 4 are set in the user's nodemask, the policy will be effected over the first (and in the Bind or Interleave case, the third and fifth) nodes in the set of allowed nodes. The nodemask passed by the user represents nodes relative to task or VMA's set of allowed nodes.

If the user's nodemask includes nodes that are outside the range of the new set of allowed nodes (for example, node 5 is set in the user's nodemask when the set of allowed nodes is only 0-3), then the remap wraps around to the beginning of the nodemask and, if not already set, sets the node in the mempolicy nodemask.

For example, consider a task that is attached to a cpuset with mems 2-5 that sets an Interleave policy over the same set with MPOL_F_RELATIVE_NODES. If the cpuset's mems change to 3-7, the interleave now occurs over nodes 3,5-7. If the cpuset's mems then change to 0,2-3,5, then the interleave occurs over nodes 0,2-3,5.

Thanks to the consistent remapping, applications preparing nodemasks to specify memory policies using this flag should disregard their current, actual cpuset imposed memory placement and prepare the nodemask as if they were always located on memory nodes 0 to N-1, where N is the number of memory nodes the policy is intended to manage. Let the kernel then remap to the set of memory nodes allowed by the task's cpuset, as that may change over time.

MPOL_F_RELATIVE_NODES cannot be combined with the MPOL_F_STATIC_NODES flag.

It also cannot be used for MPOL_PREFERRED policies that were created with an empty nodemask (local allocation).

60.10.3 Memory Policy Reference Counting

To resolve use/free races, struct mempolicy contains an atomic reference count field. Internal interfaces, mpol_get()/mpol_put() increment and decrement this reference count, respectively. mpol_put() will only free the structure back to the mempolicy kmem cache when the reference count goes to zero.

When a new memory policy is allocated, its reference count is initialized to '1', representing the reference held by the task that is installing the new policy. When a pointer to a memory policy structure is stored in another structure, another reference is added, as the task's reference will be dropped on completion of the policy installation.

During run-time "usage" of the policy, we attempt to minimize atomic operations on the reference count, as this can lead to cache lines bouncing between cpus and NUMA nodes. "Usage" here means one of the following:

1) querying of the policy, either by the task itself [using the get_mempolicy() API discussed below] or by another task using the /proc/<pid>/numa_maps interface.

2) examination of the policy to determine the policy mode and associated node or node lists, if any, for page allocation. This is considered a "hot path". Note that for MPOL_BIND, the "usage" extends across the entire allocation process, which may sleep during page reclamation, because the BIND policy nodemask is used, by reference, to filter ineligible nodes.

We can avoid taking an extra reference during the usages listed above as follows:

1) we never need to get/free the system default policy as this is never changed nor freed, once the system is up and running.

2) for querying the policy, we do not need to take an extra reference on the target task's task policy nor vma policies because we always acquire the task's mm's mmap_lock for read during the query. The set_mempolicy() and mbind() APIs [see below] always acquire the mmap_lock for write when installing or replacing task or vma policies. Thus, there is no possibility of a task or thread freeing a policy while another task or thread is querying it.

3) Page allocation usage of task or vma policy occurs in the fault path where we hold them mmap_lock for read. Again, because replacing the task or vma policy requires that the mmap_lock be held for write, the policy can't be freed out from under us while we're using it for page allocation.

4) Shared policies require special consideration. One task can replace a shared memory policy while another task, with a distinct mmap_lock, is querying or allocating a page based on the policy. To resolve this potential race, the shared policy infrastructure adds an extra reference to the shared policy during lookup while holding a spin lock on the shared policy management structure. This requires that we drop this extra reference when we're finished "using" the policy. We must drop the extra reference on shared policies in the same query/allocation paths used for non-shared policies. For this reason, shared policies are marked as such, and the extra reference is dropped "conditionally"--i.e., only for shared policies.

   Because of this extra reference counting, and because we must lookup shared policies in a tree structure under spinlock, shared policies are more expensive to use in the page al-

location path. This is especially true for shared policies on shared memory regions shared by tasks running on different NUMA nodes. This extra overhead can be avoided by always falling back to task or system default policy for shared memory regions, or by prefaulting the entire shared memory region into memory and locking it down. However, this might not be appropriate for all applications.

60.10.4 Memory Policy APIs

Linux supports 4 system calls for controlling memory policy. These APIS always affect only the calling task, the calling task's address space, or some shared object mapped into the calling task's address space.

Note: the headers that define these APIs and the parameter data types for user space applications reside in a package that is not part of the Linux kernel. The kernel system call interfaces, with the 'sys_' prefix, are defined in <linux/syscalls.h>; the mode and flag definitions are defined in <linux/mempolicy.h>.

Set [Task] Memory Policy:

```
long set_mempolicy(int mode, const unsigned long *nmask,
                                 unsigned long maxnode);
```

Set's the calling task's ”task/process memory policy” to mode specified by the 'mode' argument and the set of nodes defined by 'nmask'. 'nmask' points to a bit mask of node ids containing at least 'maxnode' ids. Optional mode flags may be passed by combining the 'mode' argument with the flag (for example: MPOL_INTERLEAVE | MPOL_F_STATIC_NODES).

See the set_mempolicy(2) man page for more details

Get [Task] Memory Policy or Related Information:

```
long get_mempolicy(int *mode,
                   const unsigned long *nmask, unsigned long maxnode,
                   void *addr, int flags);
```

Queries the ”task/process memory policy” of the calling task, or the policy or location of a specified virtual address, depending on the 'flags' argument.

See the get_mempolicy(2) man page for more details

Install VMA/Shared Policy for a Range of Task's Address Space:

```
long mbind(void *start, unsigned long len, int mode,
           const unsigned long *nmask, unsigned long maxnode,
           unsigned flags);
```

mbind() installs the policy specified by (mode, nmask, maxnodes) as a VMA policy for the range of the calling task's address space specified by the 'start' and 'len' arguments. Additional actions may be requested via the 'flags' argument.

See the mbind(2) man page for more details.

Set home node for a Range of Task's Address Spacec:

```
long sys_set_mempolicy_home_node(unsigned long start, unsigned long len,
                                 unsigned long home_node,
                                 unsigned long flags);
```

sys_set_mempolicy_home_node set the home node for a VMA policy present in the task's address range. The system call updates the home node only for the existing mempolicy range. Other address ranges are ignored. A home node is the NUMA node closest to which page allocation will come from. Specifying the home node override the default allocation policy to allocate memory close to the local node for an executing CPU.

60.10.5 Memory Policy Command Line Interface

Although not strictly part of the Linux implementation of memory policy, a command line tool, numactl(8), exists that allows one to:

- set the task policy for a specified program via set_mempolicy(2), fork(2) and exec(2)
- set the shared policy for a shared memory segment via mbind(2)

The numactl(8) tool is packaged with the run-time version of the library containing the memory policy system call wrappers. Some distributions package the headers and compile-time libraries in a separate development package.

60.10.6 Memory Policies and cpusets

Memory policies work within cpusets as described above. For memory policies that require a node or set of nodes, the nodes are restricted to the set of nodes whose memories are allowed by the cpuset constraints. If the nodemask specified for the policy contains nodes that are not allowed by the cpuset and MPOL_F_RELATIVE_NODES is not used, the intersection of the set of nodes specified for the policy and the set of nodes with memory is used. If the result is the empty set, the policy is considered invalid and cannot be installed. If MPOL_F_RELATIVE_NODES is used, the policy's nodes are mapped onto and folded into the task's set of allowed nodes as previously described.

The interaction of memory policies and cpusets can be problematic when tasks in two cpusets share access to a memory region, such as shared memory segments created by shmget() of mmap() with the MAP_ANONYMOUS and MAP_SHARED flags, and any of the tasks install shared policy on the region, only nodes whose memories are allowed in both cpusets may be used in the policies. Obtaining this information requires ”stepping outside” the memory policy APIs to use the cpuset information and requires that one know in what cpusets other task might be attaching to the shared region. Furthermore, if the cpusets' allowed memory sets are disjoint, ”local” allocation is the only valid policy.

60.11 NUMA Memory Performance

60.11.1 NUMA Locality

Some platforms may have multiple types of memory attached to a compute node. These disparate memory ranges may share some characteristics, such as CPU cache coherence, but may have different performance. For example, different media types and buses affect bandwidth and latency.

A system supports such heterogeneous memory by grouping each memory type under different domains, or "nodes", based on locality and performance characteristics. Some memory may share the same node as a CPU, and others are provided as memory only nodes. While memory only nodes do not provide CPUs, they may still be local to one or more compute nodes relative to other nodes. The following diagram shows one such example of two compute nodes with local memory and a memory only node for each of compute node:

```
+-----------------+     +-----------------+
| Compute Node 0  +-----+ Compute Node 1  |
| Local Node0 Mem |     | Local Node1 Mem |
+--------+--------+     +--------+--------+
         |                       |
+--------+--------+     +--------+--------+
| Slower Node2 Mem |     | Slower Node3 Mem |
+-----------------+     +--------+--------+
```

A "memory initiator" is a node containing one or more devices such as CPUs or separate memory I/O devices that can initiate memory requests. A "memory target" is a node containing one or more physical address ranges accessible from one or more memory initiators.

When multiple memory initiators exist, they may not all have the same performance when accessing a given memory target. Each initiator-target pair may be organized into different ranked access classes to represent this relationship. The highest performing initiator to a given target is considered to be one of that target's local initiators, and given the highest access class, 0. Any given target may have one or more local initiators, and any given initiator may have multiple local memory targets.

To aid applications matching memory targets with their initiators, the kernel provides symlinks to each other. The following example lists the relationship for the access class "0" memory initiators and targets:

```
# symlinks -v /sys/devices/system/node/nodeX/access0/targets/
relative: /sys/devices/system/node/nodeX/access0/targets/nodeY -> ../../nodeY

# symlinks -v /sys/devices/system/node/nodeY/access0/initiators/
relative: /sys/devices/system/node/nodeY/access0/initiators/nodeX -> ../../
↪nodeX
```

A memory initiator may have multiple memory targets in the same access class. The target memory's initiators in a given class indicate the nodes' access characteristics share the same performance relative to other linked initiator nodes. Each target within an initiator's access class, though, do not necessarily perform the same as each other.

The access class "1" is used to allow differentiation between initiators that are CPUs and hence

suitable for generic task scheduling, and IO initiators such as GPUs and NICs. Unlike access class 0, only nodes containing CPUs are considered.

60.11.2 NUMA Performance

Applications may wish to consider which node they want their memory to be allocated from based on the node's performance characteristics. If the system provides these attributes, the kernel exports them under the node sysfs hierarchy by appending the attributes directory under the memory node's access class 0 initiators as follows:

```
/sys/devices/system/node/nodeY/access0/initiators/
```

These attributes apply only when accessed from nodes that have the are linked under the this access's initiators.

The performance characteristics the kernel provides for the local initiators are exported are as follows:

```
# tree -P "read*|write*" /sys/devices/system/node/nodeY/access0/initiators/
/sys/devices/system/node/nodeY/access0/initiators/
|-- read_bandwidth
|-- read_latency
|-- write_bandwidth
`-- write_latency
```

The bandwidth attributes are provided in MiB/second.

The latency attributes are provided in nanoseconds.

The values reported here correspond to the rated latency and bandwidth for the platform.

Access class 1 takes the same form but only includes values for CPU to memory activity.

60.11.3 NUMA Cache

System memory may be constructed in a hierarchy of elements with various performance characteristics in order to provide large address space of slower performing memory cached by a smaller higher performing memory. The system physical addresses memory initiators are aware of are provided by the last memory level in the hierarchy. The system meanwhile uses higher performing memory to transparently cache access to progressively slower levels.

The term ”far memory” is used to denote the last level memory in the hierarchy. Each increasing cache level provides higher performing initiator access, and the term ”near memory” represents the fastest cache provided by the system.

This numbering is different than CPU caches where the cache level (ex: L1, L2, L3) uses the CPU-side view where each increased level is lower performing. In contrast, the memory cache level is centric to the last level memory, so the higher numbered cache level corresponds to memory nearer to the CPU, and further from far memory.

The memory-side caches are not directly addressable by software. When software accesses a system address, the system will return it from the near memory cache if it is present. If it is not present, the system accesses the next level of memory until there is either a hit in that cache level, or it reaches far memory.

An application does not need to know about caching attributes in order to use the system. Software may optionally query the memory cache attributes in order to maximize the performance out of such a setup. If the system provides a way for the kernel to discover this information, for example with ACPI HMAT (Heterogeneous Memory Attribute Table), the kernel will append these attributes to the NUMA node memory target.

When the kernel first registers a memory cache with a node, the kernel will create the following directory:

```
/sys/devices/system/node/nodeX/memory_side_cache/
```

If that directory is not present, the system either does not provide a memory-side cache, or that information is not accessible to the kernel.

The attributes for each level of cache is provided under its cache level index:

```
/sys/devices/system/node/nodeX/memory_side_cache/indexA/
/sys/devices/system/node/nodeX/memory_side_cache/indexB/
/sys/devices/system/node/nodeX/memory_side_cache/indexC/
```

Each cache level's directory provides its attributes. For example, the following shows a single cache level and the attributes available for software to query:

```
# tree /sys/devices/system/node/node0/memory_side_cache/
/sys/devices/system/node/node0/memory_side_cache/
|-- index1
|   |-- indexing
|   |-- line_size
|   |-- size
|   `-- write_policy
```

The ”indexing” will be 0 if it is a direct-mapped cache, and non-zero for any other indexed based, multi-way associativity.

The ”line_size” is the number of bytes accessed from the next cache level on a miss.

The ”size” is the number of bytes provided by this cache level.

The ”write_policy” will be 0 for write-back, and non-zero for write-through caching.

60.11.4 See Also

[1] https://www.uefi.org/sites/default/files/resources/ACPI_6_2.pdf - Section 5.2.27

60.12 Examining Process Page Tables

pagemap is a new (as of 2.6.25) set of interfaces in the kernel that allow userspace programs to examine the page tables and related information by reading files in `/proc`.

There are four components to pagemap:

- `/proc/pid/pagemap`. This file lets a userspace process find out which physical frame each virtual page is mapped to. It contains one 64-bit value for each virtual page, containing the following data (from `fs/proc/task_mmu.c`, above pagemap_read):
  - Bits 0-54 page frame number (PFN) if present
  - Bits 0-4 swap type if swapped
  - Bits 5-54 swap offset if swapped
  - Bit 55 pte is soft-dirty (see *Soft-Dirty PTEs*)
  - Bit 56 page exclusively mapped (since 4.2)
  - Bit 57 pte is uffd-wp write-protected (since 5.13) (see *Userfaultfd*)
  - Bits 58-60 zero
  - Bit 61 page is file-page or shared-anon (since 3.5)
  - Bit 62 page swapped
  - Bit 63 page present

  Since Linux 4.0 only users with the CAP_SYS_ADMIN capability can get PFNs. In 4.0 and 4.1 opens by unprivileged fail with -EPERM. Starting from 4.2 the PFN field is zeroed if the user does not have CAP_SYS_ADMIN. Reason: information about PFNs helps in exploiting Rowhammer vulnerability.

  If the page is not present but in swap, then the PFN contains an encoding of the swap file number and the page's offset into the swap. Unmapped pages return a null PFN. This allows determining precisely which pages are mapped (or in swap) and comparing mapped pages between processes.

  Efficient users of this interface will use `/proc/pid/maps` to determine which areas of memory are actually mapped and llseek to skip over unmapped regions.

- `/proc/kpagecount`. This file contains a 64-bit count of the number of times each page is mapped, indexed by PFN.

The page-types tool in the tools/mm directory can be used to query the number of times a page is mapped.

- `/proc/kpageflags`. This file contains a 64-bit set of flags for each page, indexed by PFN.

  The flags are (from `fs/proc/page.c`, above kpageflags_read):

  0. LOCKED

1. ERROR
2. REFERENCED
3. UPTODATE
4. DIRTY
5. LRU
6. ACTIVE
7. SLAB
8. WRITEBACK
9. RECLAIM
10. BUDDY
11. MMAP
12. ANON
13. SWAPCACHE
14. SWAPBACKED
15. COMPOUND_HEAD
16. COMPOUND_TAIL
17. HUGE
18. UNEVICTABLE
19. HWPOISON
20. NOPAGE
21. KSM
22. THP
23. OFFLINE
24. ZERO_PAGE
25. IDLE
26. PGTABLE

- `/proc/kpagecgroup`. This file contains a 64-bit inode number of the memory cgroup each page is charged to, indexed by PFN. Only available when CONFIG_MEMCG is set.

60.12.1 Short descriptions to the page flags

0 - LOCKED
The page is being locked for exclusive access, e.g. by undergoing read/write IO.

7 - SLAB
The page is managed by the SLAB/SLUB kernel memory allocator. When compound page is used, either will only set this flag on the head page.

10 - BUDDY
A free memory block managed by the buddy system allocator. The buddy system organizes free memory in blocks of various orders. An order N block has 2^N physically contiguous pages, with the BUDDY flag set for and _only_ for the first page.

15 - COMPOUND_HEAD
A compound page with order N consists of 2^N physically contiguous pages. A compound page with order 2 takes the form of "HTTT", where H donates its head page and T donates its tail page(s). The major consumers of compound pages are hugeTLB pages (*HugeTLB Pages*), the SLUB etc. memory allocators and various device drivers. However in this interface, only huge/giga pages are made visible to end users.

16 - COMPOUND_TAIL
A compound page tail (see description above).

17 - HUGE
This is an integral part of a HugeTLB page.

19 - HWPOISON
Hardware detected memory corruption on this page: don't touch the data!

20 - NOPAGE
No page frame exists at the requested address.

21 - KSM
Identical memory pages dynamically shared between one or more processes.

22 - THP
Contiguous pages which construct transparent hugepages.

23 - OFFLINE
The page is logically offline.

24 - ZERO_PAGE
Zero page for pfn_zero or huge_zero page.

25 - IDLE
The page has not been accessed since it was marked idle (see *Idle Page Tracking*). Note that this flag may be stale in case the page was accessed via a PTE. To make sure the flag is up-to-date one has to read `/sys/kernel/mm/page_idle/bitmap` first.

26 - PGTABLE
The page is in use as a page table.

IO related page flags

1 - ERROR
IO error occurred.

3 - UPTODATE
The page has up-to-date data. ie. for file backed page: (in-memory data revision >= on-disk one)

4 - DIRTY
The page has been written to, hence contains new data. i.e. for file backed page: (in-memory data revision > on-disk one)

8 - WRITEBACK
The page is being synced to disk.

LRU related page flags

5 - LRU
The page is in one of the LRU lists.

6 - ACTIVE
The page is in the active LRU list.

18 - UNEVICTABLE
The page is in the unevictable (non-)LRU list It is somehow pinned and not a candidate for LRU page reclaims, e.g. ramfs pages, shmctl(SHM_LOCK) and mlock() memory segments.

2 - REFERENCED
The page has been referenced since last LRU list enqueue/requeue.

9 - RECLAIM
The page will be reclaimed soon after its pageout IO completed.

11 - MMAP
A memory mapped page.

12 - ANON
A memory mapped page that is not part of a file.

13 - SWAPCACHE
The page is mapped to swap space, i.e. has an associated swap entry.

14 - SWAPBACKED
The page is backed by swap/RAM.

The page-types tool in the tools/mm directory can be used to query the above flags.

60.12.2 Using pagemap to do something useful

The general procedure for using pagemap to find out about a process' memory usage goes like this:

1. Read `/proc/pid/maps` to determine which parts of the memory space are mapped to what.
2. Select the maps you are interested in -- all of them, or a particular library, or the stack or the heap, etc.
3. Open `/proc/pid/pagemap` and seek to the pages you would like to examine.
4. Read a u64 for each page from pagemap.
5. Open `/proc/kpagecount` and/or `/proc/kpageflags`. For each PFN you just read, seek to that entry in the file, and read the data you want.

For example, to find the ”unique set size” (USS), which is the amount of memory that a process is using that is not shared with any other process, you can go through every map in the process, find the PFNs, look those up in kpagecount, and tally up the number of pages that are only referenced once.

60.12.3 Exceptions for Shared Memory

Page table entries for shared pages are cleared when the pages are zapped or swapped out. This makes swapped out pages indistinguishable from never-allocated ones.

In kernel space, the swap location can still be retrieved from the page cache. However, values stored only on the normal PTE get lost irretrievably when the page is swapped out (i.e. SOFT_DIRTY).

In user space, whether the page is present, swapped or none can be deduced with the help of lseek and/or mincore system calls.

lseek() can differentiate between accessed pages (present or swapped out) and holes (none/non-allocated) by specifying the SEEK_DATA flag on the file where the pages are backed. For anonymous shared pages, the file can be found in `/proc/pid/map_files/`.

mincore() can differentiate between pages in memory (present, including swap cache) and out of memory (swapped out or none/non-allocated).

60.12.4 Other notes

Reading from any of the files will return -EINVAL if you are not starting the read on an 8-byte boundary (e.g., if you sought an odd number of bytes into the file), or if the size of the read is not a multiple of 8 bytes.

Before Linux 3.11 pagemap bits 55-60 were used for ”page-shift” (which is always 12 at most architectures). Since Linux 3.11 their meaning changes after first clear of soft-dirty bits. Since Linux 4.2 they are used for flags unconditionally.

60.12.5 Pagemap Scan IOCTL

The PAGEMAP_SCAN IOCTL on the pagemap file can be used to get or optionally clear the info about page table entries. The following operations are supported in this IOCTL:

- Scan the address range and get the memory ranges matching the provided criteria. This is performed when the output buffer is specified.
- Write-protect the pages. The PM_SCAN_WP_MATCHING is used to write-protect the pages of interest. The PM_SCAN_CHECK_WPASYNC aborts the operation if non-Async Write Protected pages are found. The PM_SCAN_WP_MATCHING can be used with or without PM_SCAN_CHECK_WPASYNC.
- Both of those operations can be combined into one atomic operation where we can get and write protect the pages as well.

Following flags about pages are currently supported:

- PAGE_IS_WPALLOWED - Page has async-write-protection enabled
- PAGE_IS_WRITTEN - Page has been written to from the time it was write protected
- PAGE_IS_FILE - Page is file backed
- PAGE_IS_PRESENT - Page is present in the memory
- PAGE_IS_SWAPPED - Page is in swapped
- PAGE_IS_PFNZERO - Page has zero PFN
- PAGE_IS_HUGE - Page is THP or Hugetlb backed
- PAGE_IS_SOFT_DIRTY - Page is soft-dirty

The struct pm_scan_arg is used as the argument of the IOCTL.

1. The size of the struct pm_scan_arg must be specified in the size field. This field will be helpful in recognizing the structure if extensions are done later.
2. The flags can be specified in the flags field. The PM_SCAN_WP_MATCHING and PM_SCAN_CHECK_WPASYNC are the only added flags at this time. The get operation is optionally performed depending upon if the output buffer is provided or not.
3. The range is specified through start and end.
4. The walk can abort before visiting the complete range such as the user buffer can get full etc. The walk ending address is specified in``end_walk``.
5. The output buffer of struct page_region array and size is specified in vec and vec_len.
6. The optional maximum requested pages are specified in the max_pages.
7. The masks are specified in category_mask, category_anyof_mask, category_inverted and return_mask.

Find pages which have been written and WP them as well:

```
struct pm_scan_arg arg = {
.size = sizeof(arg),
.flags = PM_SCAN_CHECK_WPASYNC | PM_SCAN_CHECK_WPASYNC,
..
.category_mask = PAGE_IS_WRITTEN,
```

```
.return_mask = PAGE_IS_WRITTEN,
};
```

Find pages which have been written, are file backed, not swapped and either present or huge:

```
struct pm_scan_arg arg = {
.size = sizeof(arg),
.flags = 0,
..
.category_mask = PAGE_IS_WRITTEN | PAGE_IS_SWAPPED,
.category_inverted = PAGE_IS_SWAPPED,
.category_anyof_mask = PAGE_IS_PRESENT | PAGE_IS_HUGE,
.return_mask = PAGE_IS_WRITTEN | PAGE_IS_SWAPPED |
               PAGE_IS_PRESENT | PAGE_IS_HUGE,
};
```

The `PAGE_IS_WRITTEN` flag can be considered as a better-performing alternative of soft-dirty flag. It doesn't get affected by VMA merging of the kernel and hence the user can find the true soft-dirty pages in case of normal pages. (There may still be extra dirty pages reported for THP or Hugetlb pages.)

”PAGE_IS_WRITTEN” category is used with uffd write protect-enabled ranges to implement memory dirty tracking in userspace:

1. The userfaultfd file descriptor is created with `userfaultfd` syscall.
2. The `UFFD_FEATURE_WP_UNPOPULATED` and `UFFD_FEATURE_WP_ASYNC` features are set by `UFFDIO_API` IOCTL.
3. The memory range is registered with `UFFDIO_REGISTER_MODE_WP` mode through `UFFDIO_REGISTER` IOCTL.
4. Then any part of the registered memory or the whole memory region must be write protected using `PAGEMAP_SCAN` IOCTL with flag `PM_SCAN_WP_MATCHING` or the `UFFDIO_WRITEPROTECT` IOCTL can be used. Both of these perform the same operation. The former is better in terms of performance.
5. Now the `PAGEMAP_SCAN` IOCTL can be used to either just find pages which have been written to since they were last marked and/or optionally write protect the pages as well.

60.13 Shrinker Debugfs Interface

Shrinker debugfs interface provides a visibility into the kernel memory shrinkers subsystem and allows to get information about individual shrinkers and interact with them.

For each shrinker registered in the system a directory in **<debugfs>/shrinker/** is created. The directory's name is composed from the shrinker's name and an unique id: e.g. *kfree_rcu-0* or *sb-xfs:vda1-36*.

Each shrinker directory contains **count** and **scan** files, which allow to trigger *count_objects()* and *scan_objects()* callbacks for each memcg and numa node (if applicable).

60.13.1 Usage:

1. *List registered shrinkers*

```
$ cd /sys/kernel/debug/shrinker/
$ ls
dquota-cache-16     sb-devpts-28     sb-proc-47       sb-tmpfs-42
mm-shadow-18        sb-devtmpfs-5    sb-proc-48       sb-tmpfs-43
mm-zspool:zram0-34  sb-hugetlbfs-17  sb-pstore-31     sb-tmpfs-44
rcu-kfree-0         sb-hugetlbfs-33  sb-rootfs-2      sb-tmpfs-49
sb-aio-20           sb-iomem-12      sb-securityfs-6  sb-tracefs-13
sb-anon_inodefs-15  sb-mqueue-21     sb-selinuxfs-22  sb-xfs:vda1-36
sb-bdev-3           sb-nsfs-4        sb-sockfs-8      sb-zsmalloc-19
sb-bpf-32           sb-pipefs-14     sb-sysfs-26      thp-deferred_
 ↪split-10
sb-btrfs:vda2-24    sb-proc-25       sb-tmpfs-1       thp-zero-9
sb-cgroup2-30       sb-proc-39       sb-tmpfs-27      xfs-buf:vda1-37
sb-configfs-23      sb-proc-41       sb-tmpfs-29      xfs-inodegc:vda1-
 ↪38
sb-dax-11           sb-proc-45       sb-tmpfs-35
sb-debugfs-7        sb-proc-46       sb-tmpfs-40
```

2. *Get information about a specific shrinker*

```
$ cd sb-btrfs\:vda2-24/
$ ls
count            scan
```

3. *Count objects*

   Each line in the output has the following format:

```
<cgroup inode id> <nr of objects on node 0> <nr of objects on node 1> .
 ↪..
<cgroup inode id> <nr of objects on node 0> <nr of objects on node 1> .
 ↪..
...
```

   If there are no objects on all numa nodes, a line is omitted. If there are no objects at all, the output might be empty.

   If the shrinker is not memcg-aware or CONFIG_MEMCG is off, 0 is printed as cgroup inode id. If the shrinker is not numa-aware, 0's are printed for all nodes except the first one.

```
$ cat count
1 224 2
21 98 0
55 818 10
2367 2 0
2401 30 0
225 13 0
599 35 0
```

```
939 124 0
1041 3 0
1075 1 0
1109 1 0
1279 60 0
1313 7 0
1347 39 0
1381 3 0
1449 14 0
1483 63 0
1517 53 0
1551 6 0
1585 1 0
1619 6 0
1653 40 0
1687 11 0
1721 8 0
1755 4 0
1789 52 0
1823 888 0
1857 1 0
1925 2 0
1959 32 0
2027 22 0
2061 9 0
2469 799 0
2537 861 0
2639 1 0
2707 70 0
2775 4 0
2877 84 0
293 1 0
735 8 0
```

4. *Scan objects*

   The expected input format:

```
<cgroup inode id> <numa id> <number of objects to scan>
```

   For a non-memcg-aware shrinker or on a system with no memory cgrups **0** should be passed as cgroup id.

```
$ cd /sys/kernel/debug/shrinker/
$ cd sb-btrfs\:vda2-24/

$ cat count | head -n 5
1 212 0
21 97 0
55 802 5
2367 2 0
```

```
225 13 0

$ echo "55 0 200" > scan

$ cat count | head -n 5
1 212 0
21 96 0
55 752 5
2367 2 0
225 13 0
```

60.14 Soft-Dirty PTEs

The soft-dirty is a bit on a PTE which helps to track which pages a task writes to. In order to do this tracking one should

1. Clear soft-dirty bits from the task's PTEs.

   This is done by writing ”4” into the `/proc/PID/clear_refs` file of the task in question.

2. Wait some time.

3. Read soft-dirty bits from the PTEs.

   This is done by reading from the `/proc/PID/pagemap`. The bit 55 of the 64-bit qword is the soft-dirty one. If set, the respective PTE was written to since step 1.

Internally, to do this tracking, the writable bit is cleared from PTEs when the soft-dirty bit is cleared. So, after this, when the task tries to modify a page at some virtual address the #PF occurs and the kernel sets the soft-dirty bit on the respective PTE.

Note, that although all the task's address space is marked as r/o after the soft-dirty bits clear, the #PF-s that occur after that are processed fast. This is so, since the pages are still mapped to physical memory, and thus all the kernel does is finds this fact out and puts both writable and soft-dirty bits on the PTE.

While in most cases tracking memory changes by #PF-s is more than enough there is still a scenario when we can lose soft dirty bits -- a task unmaps a previously mapped memory region and then maps a new one at exactly the same place. When unmap is called, the kernel internally clears PTE values including soft dirty bits. To notify user space application about such memory region renewal the kernel always marks new memory regions (and expanded regions) as soft dirty.

This feature is actively used by the checkpoint-restore project. You can find more details about it on http://criu.org

-- Pavel Emelyanov, Apr 9, 2013

60.15 Automatically bind swap device to numa node

If the system has more than one swap device and swap device has the node information, we can make use of this information to decide which swap device to use in get_swap_pages() to get better performance.

60.15.1 How to use this feature

Swap device has priority and that decides the order of it to be used. To make use of automatically binding, there is no need to manipulate priority settings for swap devices. e.g. on a 2 node machine, assume 2 swap devices swapA and swapB, with swapA attached to node 0 and swapB attached to node 1, are going to be swapped on. Simply swapping them on by doing:

```
# swapon /dev/swapA
# swapon /dev/swapB
```

Then node 0 will use the two swap devices in the order of swapA then swapB and node 1 will use the two swap devices in the order of swapB then swapA. Note that the order of them being swapped on doesn't matter.

A more complex example on a 4 node machine. Assume 6 swap devices are going to be swapped on: swapA and swapB are attached to node 0, swapC is attached to node 1, swapD and swapE are attached to node 2 and swapF is attached to node3. The way to swap them on is the same as above:

```
# swapon /dev/swapA
# swapon /dev/swapB
# swapon /dev/swapC
# swapon /dev/swapD
# swapon /dev/swapE
# swapon /dev/swapF
```

Then node 0 will use them in the order of:

```
swapA/swapB -> swapC -> swapD -> swapE -> swapF
```

swapA and swapB will be used in a round robin mode before any other swap device.

node 1 will use them in the order of:

```
swapC -> swapA -> swapB -> swapD -> swapE -> swapF
```

node 2 will use them in the order of:

```
swapD/swapE -> swapA -> swapB -> swapC -> swapF
```

Similaly, swapD and swapE will be used in a round robin mode before any other swap devices.

node 3 will use them in the order of:

```
swapF -> swapA -> swapB -> swapC -> swapD -> swapE
```

60.15.2 Implementation details

The current code uses a priority based list, swap_avail_list, to decide which swap device to use and if multiple swap devices share the same priority, they are used round robin. This change here replaces the single global swap_avail_list with a per-numa-node list, i.e. for each numa node, it sees its own priority based list of available swap devices. Swap device's priority can be promoted on its matching node's swap_avail_list.

The current swap device's priority is set as: user can set a >=0 value, or the system will pick one starting from -1 then downwards. The priority value in the swap_avail_list is the negated value of the swap device's due to plist being sorted from low to high. The new policy doesn't change the semantics for priority >=0 cases, the previous starting from -1 then downwards now becomes starting from -2 then downwards and -1 is reserved as the promoted value. So if multiple swap devices are attached to the same node, they will all be promoted to priority -1 on that node's plist and will be used round robin before any other swap devices.

60.16 Transparent Hugepage Support

60.16.1 Objective

Performance critical computing applications dealing with large memory working sets are already running on top of libhugetlbfs and in turn hugetlbfs. Transparent HugePage Support (THP) is an alternative mean of using huge pages for the backing of virtual memory with huge pages that supports the automatic promotion and demotion of page sizes and without the shortcomings of hugetlbfs.

Currently THP only works for anonymous memory mappings and tmpfs/shmem. But in the future it can expand to other filesystems.

Note: in the examples below we presume that the basic page size is 4K and the huge page size is 2M, although the actual numbers may vary depending on the CPU architecture.

The reason applications are running faster is because of two factors. The first factor is almost completely irrelevant and it's not of significant interest because it'll also have the downside of requiring larger clear-page copy-page in page faults which is a potentially negative effect. The first factor consists in taking a single page fault for each 2M virtual region touched by userland (so reducing the enter/exit kernel frequency by a 512 times factor). This only matters the first time the memory is accessed for the lifetime of a memory mapping. The second long lasting and much more important factor will affect all subsequent accesses to the memory for the whole runtime of the application. The second factor consist of two components:

1) the TLB miss will run faster (especially with virtualization using nested pagetables but almost always also on bare metal without virtualization)

2) a single TLB entry will be mapping a much larger amount of virtual memory in turn reducing the number of TLB misses. With virtualization and nested pagetables the TLB can be mapped of larger size only if both KVM and the Linux guest are using hugepages but a significant speedup already happens if only one of the two is using hugepages just because of the fact the TLB miss is going to run faster.

Modern kernels support ”multi-size THP” (mTHP), which introduces the ability to allocate memory in blocks that are bigger than a base page but smaller than traditional PMD-size (as described above), in increments of a power-of-2 number of pages. mTHP can back anonymous memory (for example 16K, 32K, 64K, etc). These THPs continue to be PTE-mapped, but in many cases can still provide similar benefits to those outlined above: Page faults are significantly reduced (by a factor of e.g. 4, 8, 16, etc), but latency spikes are much less prominent because the size of each page isn't as huge as the PMD-sized variant and there is less memory to clear in each page fault. Some architectures also employ TLB compression mechanisms to squeeze more entries in when a set of PTEs are virtually and physically contiguous and approporiately aligned. In this case, TLB misses will occur less often.

THP can be enabled system wide or restricted to certain tasks or even memory ranges inside task's address space. Unless THP is completely disabled, there is `khugepaged` daemon that scans memory and collapses sequences of basic pages into PMD-sized huge pages.

The THP behaviour is controlled via *sysfs* interface and using madvise(2) and prctl(2) system calls.

Transparent Hugepage Support maximizes the usefulness of free memory if compared to the reservation approach of hugetlbfs by allowing all unused memory to be used as cache or other movable (or even unmovable entities). It doesn't require reservation to prevent hugepage allocation failures to be noticeable from userland. It allows paging and all other advanced VM features to be available on the hugepages. It requires no modifications for applications to take advantage of it.

Applications however can be further optimized to take advantage of this feature, like for example they've been optimized before to avoid a flood of mmap system calls for every malloc(4k). Optimizing userland is by far not mandatory and khugepaged already can take care of long lived page allocations even for hugepage unaware applications that deals with large amounts of memory.

In certain cases when hugepages are enabled system wide, application may end up allocating more memory resources. An application may mmap a large region but only touch 1 byte of it, in that case a 2M page might be allocated instead of a 4k page for no good. This is why it's possible to disable hugepages system-wide and to only have them inside MADV_HUGEPAGE madvise regions.

Embedded systems should enable hugepages only inside madvise regions to eliminate any risk of wasting any precious byte of memory and to only run faster.

Applications that gets a lot of benefit from hugepages and that don't risk to lose memory by using hugepages, should use madvise(MADV_HUGEPAGE) on their critical mmapped regions.

60.16.2 sysfs

Global THP controls

Transparent Hugepage Support for anonymous memory can be entirely disabled (mostly for debugging purposes) or only enabled inside MADV_HUGEPAGE regions (to avoid the risk of consuming more memory resources) or enabled system wide. This can be achieved per-supported-THP-size with one of:

```
echo always >/sys/kernel/mm/transparent_hugepage/hugepages-<size>kB/enabled
echo madvise >/sys/kernel/mm/transparent_hugepage/hugepages-<size>kB/enabled
echo never >/sys/kernel/mm/transparent_hugepage/hugepages-<size>kB/enabled
```

where <size> is the hugepage size being addressed, the available sizes for which vary by system.

For example:

```
echo always >/sys/kernel/mm/transparent_hugepage/hugepages-2048kB/enabled
```

Alternatively it is possible to specify that a given hugepage size will inherit the top-level "enabled" value:

```
echo inherit >/sys/kernel/mm/transparent_hugepage/hugepages-<size>kB/enabled
```

For example:

```
echo inherit >/sys/kernel/mm/transparent_hugepage/hugepages-2048kB/enabled
```

The top-level setting (for use with "inherit") can be set by issuing one of the following commands:

```
echo always >/sys/kernel/mm/transparent_hugepage/enabled
echo madvise >/sys/kernel/mm/transparent_hugepage/enabled
echo never >/sys/kernel/mm/transparent_hugepage/enabled
```

By default, PMD-sized hugepages have enabled="inherit" and all other hugepage sizes have enabled="never". If enabling multiple hugepage sizes, the kernel will select the most appropriate enabled size for a given allocation.

It's also possible to limit defrag efforts in the VM to generate anonymous hugepages in case they're not immediately free to madvise regions or to never try to defrag memory and simply fallback to regular pages unless hugepages are immediately available. Clearly if we spend CPU time to defrag memory, we would expect to gain even more by the fact we use hugepages later instead of regular pages. This isn't always guaranteed, but it may be more likely in case the allocation is for a MADV_HUGEPAGE region.

```
echo always >/sys/kernel/mm/transparent_hugepage/defrag
echo defer >/sys/kernel/mm/transparent_hugepage/defrag
echo defer+madvise >/sys/kernel/mm/transparent_hugepage/defrag
echo madvise >/sys/kernel/mm/transparent_hugepage/defrag
echo never >/sys/kernel/mm/transparent_hugepage/defrag
```

always
: means that an application requesting THP will stall on allocation failure and directly reclaim pages and compact memory in an effort to allocate a THP immediately. This may be desirable for virtual machines that benefit heavily from THP use and are willing to delay the VM start to utilise them.

defer
: means that an application will wake kswapd in the background to reclaim pages and wake kcompactd to compact memory so that THP is available in the near future. It's the responsibility of khugepaged to then install the THP pages later.

defer+madvise
will enter direct reclaim and compaction like `always`, but only for regions that have used madvise(MADV_HUGEPAGE); all other regions will wake kswapd in the background to reclaim pages and wake kcompactd to compact memory so that THP is available in the near future.

madvise
will enter direct reclaim like `always` but only for regions that are have used madvise(MADV_HUGEPAGE). This is the default behaviour.

never
should be self-explanatory.

By default kernel tries to use huge, PMD-mappable zero page on read page fault to anonymous mapping. It's possible to disable huge zero page by writing 0 or enable it back by writing 1:

```
echo 0 >/sys/kernel/mm/transparent_hugepage/use_zero_page
echo 1 >/sys/kernel/mm/transparent_hugepage/use_zero_page
```

Some userspace (such as a test program, or an optimized memory allocation library) may want to know the size (in bytes) of a PMD-mappable transparent hugepage:

```
cat /sys/kernel/mm/transparent_hugepage/hpage_pmd_size
```

khugepaged will be automatically started when one or more hugepage sizes are enabled (either by directly setting ”always” or ”madvise”, or by setting ”inherit” while the top-level enabled is set to ”always” or ”madvise”), and it'll be automatically shutdown when the last hugepage size is disabled (either by directly setting ”never”, or by setting ”inherit” while the top-level enabled is set to ”never”).

Khugepaged controls

Note: khugepaged currently only searches for opportunities to collapse to PMD-sized THP and no attempt is made to collapse to other THP sizes.

khugepaged runs usually at low frequency so while one may not want to invoke defrag algorithms synchronously during the page faults, it should be worth invoking defrag at least in khugepaged. However it's also possible to disable defrag in khugepaged by writing 0 or enable defrag in khugepaged by writing 1:

```
echo 0 >/sys/kernel/mm/transparent_hugepage/khugepaged/defrag
echo 1 >/sys/kernel/mm/transparent_hugepage/khugepaged/defrag
```

You can also control how many pages khugepaged should scan at each pass:

```
/sys/kernel/mm/transparent_hugepage/khugepaged/pages_to_scan
```

and how many milliseconds to wait in khugepaged between each pass (you can set this to 0 to run khugepaged at 100% utilization of one core):

```
/sys/kernel/mm/transparent_hugepage/khugepaged/scan_sleep_millisecs
```

and how many milliseconds to wait in khugepaged if there's an hugepage allocation failure to throttle the next allocation attempt:

```
/sys/kernel/mm/transparent_hugepage/khugepaged/alloc_sleep_millisecs
```

The khugepaged progress can be seen in the number of pages collapsed (note that this counter may not be an exact count of the number of pages collapsed, since "collapsed" could mean multiple things: (1) A PTE mapping being replaced by a PMD mapping, or (2) All 4K physical pages replaced by one 2M hugepage. Each may happen independently, or together, depending on the type of memory and the failures that occur. As such, this value should be interpreted roughly as a sign of progress, and counters in /proc/vmstat consulted for more accurate accounting):

```
/sys/kernel/mm/transparent_hugepage/khugepaged/pages_collapsed
```

for each pass:

```
/sys/kernel/mm/transparent_hugepage/khugepaged/full_scans
```

`max_ptes_none` specifies how many extra small pages (that are not already mapped) can be allocated when collapsing a group of small pages into one large page:

```
/sys/kernel/mm/transparent_hugepage/khugepaged/max_ptes_none
```

A higher value leads to use additional memory for programs. A lower value leads to gain less thp performance. Value of max_ptes_none can waste cpu time very little, you can ignore it.

`max_ptes_swap` specifies how many pages can be brought in from swap when collapsing a group of pages into a transparent huge page:

```
/sys/kernel/mm/transparent_hugepage/khugepaged/max_ptes_swap
```

A higher value can cause excessive swap IO and waste memory. A lower value can prevent THPs from being collapsed, resulting fewer pages being collapsed into THPs, and lower memory access performance.

`max_ptes_shared` specifies how many pages can be shared across multiple processes. Exceeding the number would block the collapse:

```
/sys/kernel/mm/transparent_hugepage/khugepaged/max_ptes_shared
```

A higher value may increase memory footprint for some workloads.

60.16.3 Boot parameter

You can change the sysfs boot time defaults of Transparent Hugepage Support by passing the parameter `transparent_hugepage=always` or `transparent_hugepage=madvise` or `transparent_hugepage=never` to the kernel command line.

60.16.4 Hugepages in tmpfs/shmem

You can control hugepage allocation policy in tmpfs with mount option `huge=`. It can have following values:

always
Attempt to allocate huge pages every time we need a new page;

never
Do not allocate huge pages;

within_size
Only allocate huge page if it will be fully within i_size. Also respect fadvise()/madvise() hints;

advise
Only allocate huge pages if requested with fadvise()/madvise();

The default policy is `never`.

`mount -o remount,huge= /mountpoint` works fine after mount: remounting `huge=never` will not attempt to break up huge pages at all, just stop more from being allocated.

There's also sysfs knob to control hugepage allocation policy for internal shmem mount: /sys/kernel/mm/transparent_hugepage/shmem_enabled. The mount is used for SysV SHM, memfds, shared anonymous mmaps (of /dev/zero or MAP_ANONYMOUS), GPU drivers' DRM objects, Ashmem.

In addition to policies listed above, shmem_enabled allows two further values:

deny
For use in emergencies, to force the huge option off from all mounts;

force
Force the huge option on for all - very useful for testing;

60.16.5 Need of application restart

The transparent_hugepage/enabled and transparent_hugepage/hugepages-<size>kB/enabled values and tmpfs mount option only affect future behavior. So to make them effective you need to restart any application that could have been using hugepages. This also applies to the regions registered in khugepaged.

60.16.6 Monitoring usage

Note: Currently the below counters only record events relating to PMD-sized THP. Events relating to other THP sizes are not included.

The number of PMD-sized anonymous transparent huge pages currently used by the system is available by reading the AnonHugePages field in `/proc/meminfo`. To identify what applications are using PMD-sized anonymous transparent huge pages, it is necessary to read `/proc/PID/smaps` and count the AnonHugePages fields for each mapping. (Note that AnonHugePages only

applies to traditional PMD-sized THP for historical reasons and should have been called AnonHugePmdMapped).

The number of file transparent huge pages mapped to userspace is available by reading ShmemPmdMapped and ShmemHugePages fields in `/proc/meminfo`. To identify what applications are mapping file transparent huge pages, it is necessary to read `/proc/PID/smaps` and count the FileHugeMapped fields for each mapping.

Note that reading the smaps file is expensive and reading it frequently will incur overhead.

There are a number of counters in `/proc/vmstat` that may be used to monitor how successfully the system is providing huge pages for use.

thp_fault_alloc
is incremented every time a huge page is successfully allocated to handle a page fault.

thp_collapse_alloc
is incremented by khugepaged when it has found a range of pages to collapse into one huge page and has successfully allocated a new huge page to store the data.

thp_fault_fallback
is incremented if a page fault fails to allocate a huge page and instead falls back to using small pages.

thp_fault_fallback_charge
is incremented if a page fault fails to charge a huge page and instead falls back to using small pages even though the allocation was successful.

thp_collapse_alloc_failed
is incremented if khugepaged found a range of pages that should be collapsed into one huge page but failed the allocation.

thp_file_alloc
is incremented every time a file huge page is successfully allocated.

thp_file_fallback
is incremented if a file huge page is attempted to be allocated but fails and instead falls back to using small pages.

thp_file_fallback_charge
is incremented if a file huge page cannot be charged and instead falls back to using small pages even though the allocation was successful.

thp_file_mapped
is incremented every time a file huge page is mapped into user address space.

thp_split_page
is incremented every time a huge page is split into base pages. This can happen for a variety of reasons but a common reason is that a huge page is old and is being reclaimed. This action implies splitting all PMD the page mapped with.

thp_split_page_failed
is incremented if kernel fails to split huge page. This can happen if the page was pinned by somebody.

thp_deferred_split_page
is incremented when a huge page is put onto split queue. This happens when a huge page is partially unmapped and splitting it would free up some memory. Pages on split queue are going to be split under memory pressure.

thp_split_pmd
is incremented every time a PMD split into table of PTEs. This can happen, for instance, when application calls mprotect() or munmap() on part of huge page. It doesn't split huge page, only page table entry.

thp_zero_page_alloc
is incremented every time a huge zero page used for thp is successfully allocated. Note, it doesn't count every map of the huge zero page, only its allocation.

thp_zero_page_alloc_failed
is incremented if kernel fails to allocate huge zero page and falls back to using small pages.

thp_swpout
is incremented every time a huge page is swapout in one piece without splitting.

thp_swpout_fallback
is incremented if a huge page has to be split before swapout. Usually because failed to allocate some continuous swap space for the huge page.

As the system ages, allocating huge pages may be expensive as the system uses memory compaction to copy data around memory to free a huge page for use. There are some counters in `/proc/vmstat` to help monitor this overhead.

compact_stall
is incremented every time a process stalls to run memory compaction so that a huge page is free for use.

compact_success
is incremented if the system compacted memory and freed a huge page for use.

compact_fail
is incremented if the system tries to compact memory but failed.

It is possible to establish how long the stalls were using the function tracer to record how long was spent in __alloc_pages() and using the mm_page_alloc tracepoint to identify which allocations were for huge pages.

60.16.7 Optimizing the applications

To be guaranteed that the kernel will map a THP immediately in any memory region, the mmap region has to be hugepage naturally aligned. posix_memalign() can provide that guarantee.

60.16.8 Hugetlbfs

You can use hugetlbfs on a kernel that has transparent hugepage support enabled just fine as always. No difference can be noted in hugetlbfs other than there will be less overall fragmentation. All usual features belonging to hugetlbfs are preserved and unaffected. libhugetlbfs will also work fine as usual.

60.17 Userfaultfd

60.17.1 Objective

Userfaults allow the implementation of on-demand paging from userland and more generally they allow userland to take control of various memory page faults, something otherwise only the kernel code could do.

For example userfaults allows a proper and more optimal implementation of the `PROT_NONE+SIGSEGV` trick.

60.17.2 Design

Userspace creates a new userfaultfd, initializes it, and registers one or more regions of virtual memory with it. Then, any page faults which occur within the region(s) result in a message being delivered to the userfaultfd, notifying userspace of the fault.

The `userfaultfd` (aside from registering and unregistering virtual memory ranges) provides two primary functionalities:

1) `read/POLLIN` protocol to notify a userland thread of the faults happening

2) various `UFFDIO_*` ioctls that can manage the virtual memory regions registered in the `userfaultfd` that allows userland to efficiently resolve the userfaults it receives via 1) or to manage the virtual memory in the background

The real advantage of userfaults if compared to regular virtual memory management of mremap/mprotect is that the userfaults in all their operations never involve heavyweight structures like vmas (in fact the `userfaultfd` runtime load never takes the mmap_lock for writing). Vmas are not suitable for page- (or hugepage) granular fault tracking when dealing with virtual address spaces that could span Terabytes. Too many vmas would be needed for that.

The `userfaultfd`, once created, can also be passed using unix domain sockets to a manager process, so the same manager process could handle the userfaults of a multitude of different processes without them being aware about what is going on (well of course unless they later try to use the `userfaultfd` themselves on the same region the manager is already tracking, which is a corner case that would currently return `-EBUSY`).

60.17.3 API

Creating a userfaultfd

There are two ways to create a new userfaultfd, each of which provide ways to restrict access to this functionality (since historically userfaultfds which handle kernel page faults have been a useful tool for exploiting the kernel).

The first way, supported since userfaultfd was introduced, is the userfaultfd(2) syscall. Access to this is controlled in several ways:

- Any user can always create a userfaultfd which traps userspace page faults only. Such a userfaultfd can be created using the userfaultfd(2) syscall with the flag UFFD_USER_MODE_ONLY.

- In order to also trap kernel page faults for the address space, either the process needs the CAP_SYS_PTRACE capability, or the system must have vm.unprivileged_userfaultfd set to 1. By default, vm.unprivileged_userfaultfd is set to 0.

The second way, added to the kernel more recently, is by opening /dev/userfaultfd and issuing a USERFAULTFD_IOC_NEW ioctl to it. This method yields equivalent userfaultfds to the userfaultfd(2) syscall.

Unlike userfaultfd(2), access to /dev/userfaultfd is controlled via normal filesystem permissions (user/group/mode), which gives fine grained access to userfaultfd specifically, without also granting other unrelated privileges at the same time (as e.g. granting CAP_SYS_PTRACE would do). Users who have access to /dev/userfaultfd can always create userfaultfds that trap kernel page faults; vm.unprivileged_userfaultfd is not considered.

Initializing a userfaultfd

When first opened the `userfaultfd` must be enabled invoking the `UFFDIO_API` ioctl specifying a `uffdio_api.api` value set to `UFFD_API` (or a later API version) which will specify the `read/POLLIN` protocol userland intends to speak on the `UFFD` and the `uffdio_api.features` userland requires. The `UFFDIO_API` ioctl if successful (i.e. if the requested `uffdio_api.api` is spoken also by the running kernel and the requested features are going to be enabled) will return into `uffdio_api.features` and `uffdio_api.ioctls` two 64bit bitmasks of respectively all the available features of the read(2) protocol and the generic ioctl available.

The `uffdio_api.features` bitmask returned by the `UFFDIO_API` ioctl defines what memory types are supported by the `userfaultfd` and what events, except page fault notifications, may be generated:

- The `UFFD_FEATURE_EVENT_*` flags indicate that various other events other than page faults are supported. These events are described in more detail below in the *Non-cooperative userfaultfd* section.
- `UFFD_FEATURE_MISSING_HUGETLBFS` and `UFFD_FEATURE_MISSING_SHMEM` indicate that the kernel supports `UFFDIO_REGISTER_MODE_MISSING` registrations for hugetlbfs and shared memory (covering all shmem APIs, i.e. tmpfs, `IPCSHM`, `/dev/zero`, `MAP_SHARED`, `memfd_create`, etc) virtual memory areas, respectively.
- `UFFD_FEATURE_MINOR_HUGETLBFS` indicates that the kernel supports `UFFDIO_REGISTER_MODE_MINOR` registration for hugetlbfs virtual memory areas. `UFFD_FEATURE_MINOR_SHMEM` is the analogous feature indicating support for shmem virtual memory areas.
- `UFFD_FEATURE_MOVE` indicates that the kernel supports moving an existing page contents from userspace.

The userland application should set the feature flags it intends to use when invoking the `UFFDIO_API` ioctl, to request that those features be enabled if supported.

Once the `userfaultfd` API has been enabled the `UFFDIO_REGISTER` ioctl should be invoked (if present in the returned `uffdio_api.ioctls` bitmask) to register a memory range in the `userfaultfd` by setting the uffdio_register structure accordingly. The `uffdio_register.mode` bitmask will specify to the kernel which kind of faults to track for the range. The `UFFDIO_REGISTER` ioctl will return the `uffdio_register.ioctls` bitmask of ioctls that are suitable to resolve userfaults on the range registered. Not all ioctls will necessarily be supported

for all memory types (e.g. anonymous memory vs. shmem vs. hugetlbfs), or all types of intercepted faults.

Userland can use the `uffdio_register.ioctls` to manage the virtual address space in the background (to add or potentially also remove memory from the `userfaultfd` registered range). This means a userfault could be triggering just before userland maps in the background the user-faulted page.

Resolving Userfaults

There are three basic ways to resolve userfaults:

- `UFFDIO_COPY` atomically copies some existing page contents from userspace.
- `UFFDIO_ZEROPAGE` atomically zeros the new page.
- `UFFDIO_CONTINUE` maps an existing, previously-populated page.

These operations are atomic in the sense that they guarantee nothing can see a half-populated page, since readers will keep userfaulting until the operation has finished.

By default, these wake up userfaults blocked on the range in question. They support a `UFFDIO_*_MODE_DONTWAKE` `mode` flag, which indicates that waking will be done separately at some later time.

Which ioctl to choose depends on the kind of page fault, and what we'd like to do to resolve it:

- For `UFFDIO_REGISTER_MODE_MISSING` faults, the fault needs to be resolved by either providing a new page (`UFFDIO_COPY`), or mapping the zero page (`UFFDIO_ZEROPAGE`). By default, the kernel would map the zero page for a missing fault. With userfaultfd, userspace can decide what content to provide before the faulting thread continues.
- For `UFFDIO_REGISTER_MODE_MINOR` faults, there is an existing page (in the page cache). Userspace has the option of modifying the page's contents before resolving the fault. Once the contents are correct (modified or not), userspace asks the kernel to map the page and let the faulting thread continue with `UFFDIO_CONTINUE`.

Notes:

- You can tell which kind of fault occurred by examining `pagefault.flags` within the `uffd_msg`, checking for the `UFFD_PAGEFAULT_FLAG_*` flags.
- None of the page-delivering ioctls default to the range that you registered with. You must fill in all fields for the appropriate ioctl struct including the range.
- You get the address of the access that triggered the missing page event out of a struct uffd_msg that you read in the thread from the uffd. You can supply as many pages as you want with these IOCTLs. Keep in mind that unless you used DONTWAKE then the first of any of those IOCTLs wakes up the faulting thread.
- Be sure to test for all errors including (`pollfd[0].revents & POLLERR`). This can happen, e.g. when ranges supplied were incorrect.

Write Protect Notifications

This is equivalent to (but faster than) using mprotect and a SIGSEGV signal handler.

Firstly you need to register a range with `UFFDIO_REGISTER_MODE_WP`. Instead of using mprotect(2) you use `ioctl(uffd, UFFDIO_WRITEPROTECT, struct *uffdio_writeprotect)` while `mode = UFFDIO_WRITEPROTECT_MODE_WP` in the struct passed in. The range does not default to and does not have to be identical to the range you registered with. You can write protect as many ranges as you like (inside the registered range). Then, in the thread reading from uffd the struct will have `msg.arg.pagefault.flags & UFFD_PAGEFAULT_FLAG_WP` set. Now you send `ioctl(uffd, UFFDIO_WRITEPROTECT, struct *uffdio_writeprotect)` again while `pagefault.mode` does not have `UFFDIO_WRITEPROTECT_MODE_WP` set. This wakes up the thread which will continue to run with writes. This allows you to do the bookkeeping about the write in the uffd reading thread before the ioctl.

If you registered with both `UFFDIO_REGISTER_MODE_MISSING` and `UFFDIO_REGISTER_MODE_WP` then you need to think about the sequence in which you supply a page and undo write protect. Note that there is a difference between writes into a WP area and into a !WP area. The former will have `UFFD_PAGEFAULT_FLAG_WP` set, the latter `UFFD_PAGEFAULT_FLAG_WRITE`. The latter did not fail on protection but you still need to supply a page when `UFFDIO_REGISTER_MODE_MISSING` was used.

Userfaultfd write-protect mode currently behave differently on none ptes (when e.g. page is missing) over different types of memories.

For anonymous memory, `ioctl(UFFDIO_WRITEPROTECT)` will ignore none ptes (e.g. when pages are missing and not populated). For file-backed memories like shmem and hugetlbfs, none ptes will be write protected just like a present pte. In other words, there will be a userfaultfd write fault message generated when writing to a missing page on file typed memories, as long as the page range was write-protected before. Such a message will not be generated on anonymous memories by default.

If the application wants to be able to write protect none ptes on anonymous memory, one can pre-populate the memory with e.g. MADV_POPULATE_READ. On newer kernels, one can also detect the feature UFFD_FEATURE_WP_UNPOPULATED and set the feature bit in advance to make sure none ptes will also be write protected even upon anonymous memory.

When using `UFFDIO_REGISTER_MODE_WP` in combination with either `UFFDIO_REGISTER_MODE_MISSING` or `UFFDIO_REGISTER_MODE_MINOR`, when resolving missing / minor faults with `UFFDIO_COPY` or `UFFDIO_CONTINUE` respectively, it may be desirable for the new page / mapping to be write-protected (so future writes will also result in a WP fault). These ioctls support a mode flag (`UFFDIO_COPY_MODE_WP` or `UFFDIO_CONTINUE_MODE_WP` respectively) to configure the mapping this way.

If the userfaultfd context has `UFFD_FEATURE_WP_ASYNC` feature bit set, any vma registered with write-protection will work in async mode rather than the default sync mode.

In async mode, there will be no message generated when a write operation happens, meanwhile the write-protection will be resolved automatically by the kernel. It can be seen as a more accurate version of soft-dirty tracking and it can be different in a few ways:

- The dirty result will not be affected by vma changes (e.g. vma merging) because the dirty is only tracked by the pte.
- It supports range operations by default, so one can enable tracking on any range of memory as long as page aligned.

- Dirty information will not get lost if the pte was zapped due to various reasons (e.g. during split of a shmem transparent huge page).
- Due to a reverted meaning of soft-dirty (page clean when uffd-wp bit set; dirty when uffd-wp bit cleared), it has different semantics on some of the memory operations. For example: `MADV_DONTNEED` on anonymous (or `MADV_REMOVE` on a file mapping) will be treated as dirtying of memory by dropping uffd-wp bit during the procedure.

The user app can collect the "written/dirty" status by looking up the uffd-wp bit for the pages being interested in /proc/pagemap.

The page will not be under track of uffd-wp async mode until the page is explicitly write-protected by `ioctl(UFFDIO_WRITEPROTECT)` with the mode flag `UFFDIO_WRITEPROTECT_MODE_WP` set. Trying to resolve a page fault that was tracked by async mode userfaultfd-wp is invalid.

When userfaultfd-wp async mode is used alone, it can be applied to all kinds of memory.

Memory Poisioning Emulation

In response to a fault (either missing or minor), an action userspace can take to "resolve" it is to issue a `UFFDIO_POISON`. This will cause any future faulters to either get a SIGBUS, or in KVM's case the guest will receive an MCE as if there were hardware memory poisoning.

This is used to emulate hardware memory poisoning. Imagine a VM running on a machine which experiences a real hardware memory error. Later, we live migrate the VM to another physical machine. Since we want the migration to be transparent to the guest, we want that same address range to act as if it was still poisoned, even though it's on a new physical host which ostensibly doesn't have a memory error in the exact same spot.

60.17.4 QEMU/KVM

QEMU/KVM is using the `userfaultfd` syscall to implement postcopy live migration. Postcopy live migration is one form of memory externalization consisting of a virtual machine running with part or all of its memory residing on a different node in the cloud. The `userfaultfd` abstraction is generic enough that not a single line of KVM kernel code had to be modified in order to add postcopy live migration to QEMU.

Guest async page faults, `FOLL_NOWAIT` and all other `GUP*` features work just fine in combination with userfaults. Userfaults trigger async page faults in the guest scheduler so those guest processes that aren't waiting for userfaults (i.e. network bound) can keep running in the guest vcpus.

It is generally beneficial to run one pass of precopy live migration just before starting postcopy live migration, in order to avoid generating userfaults for readonly guest regions.

The implementation of postcopy live migration currently uses one single bidirectional socket but in the future two different sockets will be used (to reduce the latency of the userfaults to the minimum possible without having to decrease `/proc/sys/net/ipv4/tcp_wmem`).

The QEMU in the source node writes all pages that it knows are missing in the destination node, into the socket, and the migration thread of the QEMU running in the destination node runs `UFFDIO_COPY|ZEROPAGE` ioctls on the `userfaultfd` in order to map the received pages into the guest (`UFFDIO_ZEROCOPY` is used if the source page was a zero page).

A different postcopy thread in the destination node listens with poll() to the `userfaultfd` in parallel. When a `POLLIN` event is generated after a userfault triggers, the postcopy thread read() from the `userfaultfd` and receives the fault address (or `-EAGAIN` in case the userfault was already resolved and waken by a `UFFDIO_COPY|ZEROPAGE` run by the parallel QEMU migration thread).

After the QEMU postcopy thread (running in the destination node) gets the userfault address it writes the information about the missing page into the socket. The QEMU source node receives the information and roughly "seeks" to that page address and continues sending all remaining missing pages from that new page offset. Soon after that (just the time to flush the tcp_wmem queue through the network) the migration thread in the QEMU running in the destination node will receive the page that triggered the userfault and it'll map it as usual with the `UFFDIO_COPY|ZEROPAGE` (without actually knowing if it was spontaneously sent by the source or if it was an urgent page requested through a userfault).

By the time the userfaults start, the QEMU in the destination node doesn't need to keep any per-page state bitmap relative to the live migration around and a single per-page bitmap has to be maintained in the QEMU running in the source node to know which pages are still missing in the destination node. The bitmap in the source node is checked to find which missing pages to send in round robin and we seek over it when receiving incoming userfaults. After sending each page of course the bitmap is updated accordingly. It's also useful to avoid sending the same page twice (in case the userfault is read by the postcopy thread just before `UFFDIO_COPY|ZEROPAGE` runs in the migration thread).

60.17.5 Non-cooperative userfaultfd

When the `userfaultfd` is monitored by an external manager, the manager must be able to track changes in the process virtual memory layout. Userfaultfd can notify the manager about such changes using the same read(2) protocol as for the page fault notifications. The manager has to explicitly enable these events by setting appropriate bits in `uffdio_api.features` passed to `UFFDIO_API` ioctl:

`UFFD_FEATURE_EVENT_FORK`
: enable `userfaultfd` hooks for fork(). When this feature is enabled, the `userfaultfd` context of the parent process is duplicated into the newly created process. The manager receives `UFFD_EVENT_FORK` with file descriptor of the new `userfaultfd` context in the `uffd_msg.fork`.

`UFFD_FEATURE_EVENT_REMAP`
: enable notifications about mremap() calls. When the non-cooperative process moves a virtual memory area to a different location, the manager will receive `UFFD_EVENT_REMAP`. The `uffd_msg.remap` will contain the old and new addresses of the area and its original length.

`UFFD_FEATURE_EVENT_REMOVE`
: enable notifications about madvise(MADV_REMOVE) and madvise(MADV_DONTNEED) calls. The event `UFFD_EVENT_REMOVE` will be generated upon these calls to madvise(). The `uffd_msg.remove` will contain start and end addresses of the removed area.

`UFFD_FEATURE_EVENT_UNMAP`
: enable notifications about memory unmapping. The manager will get `UFFD_EVENT_UNMAP` with `uffd_msg.remove` containing start and end addresses of the unmapped area.

Although the `UFFD_FEATURE_EVENT_REMOVE` and `UFFD_FEATURE_EVENT_UNMAP` are pretty similar,

they quite differ in the action expected from the `userfaultfd` manager. In the former case, the virtual memory is removed, but the area is not, the area remains monitored by the `userfaultfd`, and if a page fault occurs in that area it will be delivered to the manager. The proper resolution for such page fault is to zeromap the faulting address. However, in the latter case, when an area is unmapped, either explicitly (with munmap() system call), or implicitly (e.g. during mremap()), the area is removed and in turn the `userfaultfd` context for such area disappears too and the manager will not get further userland page faults from the removed area. Still, the notification is required in order to prevent manager from using `UFFDIO_COPY` on the unmapped area.

Unlike userland page faults which have to be synchronous and require explicit or implicit wakeup, all the events are delivered asynchronously and the non-cooperative process resumes execution as soon as manager executes read(). The `userfaultfd` manager should carefully synchronize calls to `UFFDIO_COPY` with the events processing. To aid the synchronization, the `UFFDIO_COPY` ioctl will return `-ENOSPC` when the monitored process exits at the time of `UFFDIO_COPY`, and `-ENOENT`, when the non-cooperative process has changed its virtual memory layout simultaneously with outstanding `UFFDIO_COPY` operation.

The current asynchronous model of the event delivery is optimal for single threaded non-cooperative `userfaultfd` manager implementations. A synchronous event delivery model can be added later as a new `userfaultfd` feature to facilitate multithreading enhancements of the non cooperative manager, for example to allow `UFFDIO_COPY` ioctls to run in parallel to the event reception. Single threaded implementations should continue to use the current async event delivery model instead.

60.18 zswap

60.18.1 Overview

Zswap is a lightweight compressed cache for swap pages. It takes pages that are in the process of being swapped out and attempts to compress them into a dynamically allocated RAM-based memory pool. zswap basically trades CPU cycles for potentially reduced swap I/O. This trade-off can also result in a significant performance improvement if reads from the compressed cache are faster than reads from a swap device.

Some potential benefits:

- Desktop/laptop users with limited RAM capacities can mitigate the performance impact of swapping.
- Overcommitted guests that share a common I/O resource can dramatically reduce their swap I/O pressure, avoiding heavy handed I/O throttling by the hypervisor. This allows more work to get done with less impact to the guest workload and guests sharing the I/O subsystem
- Users with SSDs as swap devices can extend the life of the device by drastically reducing life-shortening writes.

Zswap evicts pages from compressed cache on an LRU basis to the backing swap device when the compressed pool reaches its size limit. This requirement had been identified in prior community discussions.

Whether Zswap is enabled at the boot time depends on whether the `CONFIG_ZSWAP_DEFAULT_ON` Kconfig option is enabled or not. This setting can then be overridden by providing the kernel

command line `zswap.enabled=` option, for example `zswap.enabled=0`. Zswap can also be enabled and disabled at runtime using the sysfs interface. An example command to enable zswap at runtime, assuming sysfs is mounted at `/sys`, is:

```
echo 1 > /sys/module/zswap/parameters/enabled
```

When zswap is disabled at runtime it will stop storing pages that are being swapped out. However, it will _not_ immediately write out or fault back into memory all of the pages stored in the compressed pool. The pages stored in zswap will remain in the compressed pool until they are either invalidated or faulted back into memory. In order to force all pages out of the compressed pool, a swapoff on the swap device(s) will fault back into memory all swapped out pages, including those in the compressed pool.

60.18.2 Design

Zswap receives pages for compression from the swap subsystem and is able to evict pages from its own compressed pool on an LRU basis and write them back to the backing swap device in the case that the compressed pool is full.

Zswap makes use of zpool for the managing the compressed memory pool. Each allocation in zpool is not directly accessible by address. Rather, a handle is returned by the allocation routine and that handle must be mapped before being accessed. The compressed memory pool grows on demand and shrinks as compressed pages are freed. The pool is not preallocated. By default, a zpool of type selected in `CONFIG_ZSWAP_ZPOOL_DEFAULT` Kconfig option is created, but it can be overridden at boot time by setting the `zpool` attribute, e.g. `zswap.zpool=zbud`. It can also be changed at runtime using the sysfs `zpool` attribute, e.g.:

```
echo zbud > /sys/module/zswap/parameters/zpool
```

The zbud type zpool allocates exactly 1 page to store 2 compressed pages, which means the compression ratio will always be 2:1 or worse (because of half-full zbud pages). The zsmalloc type zpool has a more complex compressed page storage method, and it can achieve greater storage densities.

When a swap page is passed from swapout to zswap, zswap maintains a mapping of the swap entry, a combination of the swap type and swap offset, to the zpool handle that references that compressed swap page. This mapping is achieved with a red-black tree per swap type. The swap offset is the search key for the tree nodes.

During a page fault on a PTE that is a swap entry, the swapin code calls the zswap load function to decompress the page into the page allocated by the page fault handler.

Once there are no PTEs referencing a swap page stored in zswap (i.e. the count in the swap_map goes to 0) the swap code calls the zswap invalidate function to free the compressed entry.

Zswap seeks to be simple in its policies. Sysfs attributes allow for one user controlled policy:

- max_pool_percent - The maximum percentage of memory that the compressed pool can occupy.

The default compressor is selected in `CONFIG_ZSWAP_COMPRESSOR_DEFAULT` Kconfig option, but it can be overridden at boot time by setting the `compressor` attribute, e.g. `zswap.compressor=lzo`. It can also be changed at runtime using the sysfs "compressor" attribute, e.g.:

```
echo lzo > /sys/module/zswap/parameters/compressor
```

When the zpool and/or compressor parameter is changed at runtime, any existing compressed pages are not modified; they are left in their own zpool. When a request is made for a page in an old zpool, it is uncompressed using its original compressor. Once all pages are removed from an old zpool, the zpool and its compressor are freed.

Some of the pages in zswap are same-value filled pages (i.e. contents of the page have same value or repetitive pattern). These pages include zero-filled pages and they are handled differently. During store operation, a page is checked if it is a same-value filled page before compressing it. If true, the compressed length of the page is set to zero and the pattern or same-filled value is stored.

Same-value filled pages identification feature is enabled by default and can be disabled at boot time by setting the `same_filled_pages_enabled` attribute to 0, e.g. `zswap.same_filled_pages_enabled=0`. It can also be enabled and disabled at runtime using the sysfs `same_filled_pages_enabled` attribute, e.g.:

```
echo 1 > /sys/module/zswap/parameters/same_filled_pages_enabled
```

When zswap same-filled page identification is disabled at runtime, it will stop checking for the same-value filled pages during store operation. In other words, every page will be then considered non-same-value filled. However, the existing pages which are marked as same-value filled pages remain stored unchanged in zswap until they are either loaded or invalidated.

In some circumstances it might be advantageous to make use of just the zswap ability to efficiently store same-filled pages without enabling the whole compressed page storage. In this case the handling of non-same-value pages by zswap (enabled by default) can be disabled by setting the `non_same_filled_pages_enabled` attribute to 0, e.g. `zswap.non_same_filled_pages_enabled=0`. It can also be enabled and disabled at runtime using the sysfs `non_same_filled_pages_enabled` attribute, e.g.:

```
echo 1 > /sys/module/zswap/parameters/non_same_filled_pages_enabled
```

Disabling both `zswap.same_filled_pages_enabled` and `zswap.non_same_filled_pages_enabled` effectively disables accepting any new pages by zswap.

To prevent zswap from shrinking pool when zswap is full and there's a high pressure on swap (this will result in flipping pages in and out zswap pool without any real benefit but with a performance drop for the system), a special parameter has been introduced to implement a sort of hysteresis to refuse taking pages into zswap pool until it has sufficient space if the limit has been hit. To set the threshold at which zswap would start accepting pages again after it became full, use the sysfs `accept_threshold_percent` attribute, e. g.:

```
echo 80 > /sys/module/zswap/parameters/accept_threshold_percent
```

Setting this parameter to 100 will disable the hysteresis.

Some users cannot tolerate the swapping that comes with zswap store failures and zswap writebacks. Swapping can be disabled entirely (without disabling zswap itself) on a cgroup-basis as follows:

echo 0 > /sys/fs/cgroup/<cgroup-name>/memory.zswap.writeback

Note that if the store failures are recurring (for e.g if the pages are incompressible), users can observe reclaim inefficiency after disabling writeback (because the same pages might be rejected again and again).

When there is a sizable amount of cold memory residing in the zswap pool, it can be advantageous to proactively write these cold pages to swap and reclaim the memory for other use cases. By default, the zswap shrinker is disabled. User can enable it as follows:

```
echo Y > /sys/module/zswap/parameters/shrinker_enabled
```

This can be enabled at the boot time if `CONFIG_ZSWAP_SHRINKER_DEFAULT_ON` is selected.

A debugfs interface is provided for various statistic about pool size, number of pages stored, same-value filled pages and various counters for the reasons pages are rejected.

CHAPTER

SIXTYONE

KERNEL MODULE SIGNING FACILITY

61.1 Overview

The kernel module signing facility cryptographically signs modules during installation and then checks the signature upon loading the module. This allows increased kernel security by disallowing the loading of unsigned modules or modules signed with an invalid key. Module signing increases security by making it harder to load a malicious module into the kernel. The module signature checking is done by the kernel so that it is not necessary to have trusted userspace bits.

This facility uses X.509 ITU-T standard certificates to encode the public keys involved. The signatures are not themselves encoded in any industrial standard type. The built-in facility currently only supports the RSA & NIST P-384 ECDSA public key signing standard (though it is pluggable and permits others to be used). The possible hash algorithms that can be used are SHA-2 and SHA-3 of sizes 256, 384, and 512 (the algorithm is selected by data in the signature).

61.2 Configuring module signing

The module signing facility is enabled by going to the *Enable Loadable Module Support* section of the kernel configuration and turning on:

```
CONFIG_MODULE_SIG       "Module signature verification"
```

This has a number of options available:

(1) *Require modules to be validly signed* (`CONFIG_MODULE_SIG_FORCE`)

This specifies how the kernel should deal with a module that has a signature for which the key is not known or a module that is unsigned.

If this is off (ie. ”permissive”), then modules for which the key is not available and modules that are unsigned are permitted, but the kernel will be marked as being tainted, and the concerned modules will be marked as tainted, shown with the character 'E'.

If this is on (ie. ”restrictive”), only modules that have a valid signature that can be verified by a public key in the kernel's possession will be loaded. All other modules will generate an error.

Irrespective of the setting here, if the module has a signature block that cannot be parsed, it will be rejected out of hand.

(2) *Automatically sign all modules* (CONFIG_MODULE_SIG_ALL)

If this is on then modules will be automatically signed during the modules_install phase of a build. If this is off, then the modules must be signed manually using:

```
scripts/sign-file
```

(3) *Which hash algorithm should modules be signed with?*

This presents a choice of which hash algorithm the installation phase will sign the modules with:

CONFIG_MODULE_SIG_SHA256	*Sign modules with SHA-256*
CONFIG_MODULE_SIG_SHA384	*Sign modules with SHA-384*
CONFIG_MODULE_SIG_SHA512	*Sign modules with SHA-512*
CONFIG_MODULE_SIG_SHA3_256	*Sign modules with SHA3-256*
CONFIG_MODULE_SIG_SHA3_384	*Sign modules with SHA3-384*
CONFIG_MODULE_SIG_SHA3_512	*Sign modules with SHA3-512*

The algorithm selected here will also be built into the kernel (rather than being a module) so that modules signed with that algorithm can have their signatures checked without causing a dependency loop.

(4) *File name or PKCS#11 URI of module signing key* (CONFIG_MODULE_SIG_KEY)

Setting this option to something other than its default of `certs/signing_key.pem` will disable the autogeneration of signing keys and allow the kernel modules to be signed with a key of your choosing. The string provided should identify a file containing both a private key and its corresponding X.509 certificate in PEM form, or — on systems where the OpenSSL ENGINE_pkcs11 is functional — a PKCS#11 URI as defined by RFC7512. In the latter case, the PKCS#11 URI should reference both a certificate and a private key.

If the PEM file containing the private key is encrypted, or if the PKCS#11 token requires a PIN, this can be provided at build time by means of the `KBUILD_SIGN_PIN` variable.

(5) *Additional X.509 keys for default system keyring* (CONFIG_SYSTEM_TRUSTED_KEYS)

This option can be set to the filename of a PEM-encoded file containing additional certificates which will be included in the system keyring by default.

Note that enabling module signing adds a dependency on the OpenSSL devel packages to the kernel build processes for the tool that does the signing.

61.3 Generating signing keys

Cryptographic keypairs are required to generate and check signatures. A private key is used to generate a signature and the corresponding public key is used to check it. The private key is only needed during the build, after which it can be deleted or stored securely. The public key gets built into the kernel so that it can be used to check the signatures as the modules are loaded.

Under normal conditions, when `CONFIG_MODULE_SIG_KEY` is unchanged from its default, the kernel build will automatically generate a new keypair using openssl if one does not exist in the file:

```
certs/signing_key.pem
```

during the building of vmlinux (the public part of the key needs to be built into vmlinux) using parameters in the:

```
certs/x509.genkey
```

file (which is also generated if it does not already exist).

One can select between RSA (`MODULE_SIG_KEY_TYPE_RSA`) and ECDSA (`MODULE_SIG_KEY_TYPE_ECDSA`) to generate either RSA 4k or NIST P-384 keypair.

It is strongly recommended that you provide your own x509.genkey file.

Most notably, in the x509.genkey file, the req_distinguished_name section should be altered from the default:

```
[ req_distinguished_name ]
#O = Unspecified company
CN = Build time autogenerated kernel key
#emailAddress = unspecified.user@unspecified.company
```

The generated RSA key size can also be set with:

```
[ req ]
default_bits = 4096
```

It is also possible to manually generate the key private/public files using the x509.genkey key generation configuration file in the root node of the Linux kernel sources tree and the openssl command. The following is an example to generate the public/private key files:

```
openssl req -new -nodes -utf8 -sha256 -days 36500 -batch -x509 \
   -config x509.genkey -outform PEM -out kernel_key.pem \
   -keyout kernel_key.pem
```

The full pathname for the resulting kernel_key.pem file can then be specified in the `CONFIG_MODULE_SIG_KEY` option, and the certificate and key therein will be used instead of an autogenerated keypair.

61.4 Public keys in the kernel

The kernel contains a ring of public keys that can be viewed by root. They're in a keyring called ".builtin_trusted_keys" that can be seen by:

```
[root@deneb ~]# cat /proc/keys
...
223c7853 I------     1 perm 1f030000     0     0 keyring   .builtin_trusted_
↪keys: 1
302d2d52 I------     1 perm 1f010000     0     0 asymmetri Fedora kernel␣
↪signing key: d69a84e6bce3d216b979e9505b3e3ef9a7118079: X509.RSA a7118079 []
...
```

Beyond the public key generated specifically for module signing, additional trusted certificates can be provided in a PEM-encoded file referenced by the `CONFIG_SYSTEM_TRUSTED_KEYS` configuration option.

Further, the architecture code may take public keys from a hardware store and add those in also (e.g. from the UEFI key database).

Finally, it is possible to add additional public keys by doing:

```
keyctl padd asymmetric "" [.builtin_trusted_keys-ID] <[key-file]
```

e.g.:

```
keyctl padd asymmetric "" 0x223c7853 <my_public_key.x509
```

Note, however, that the kernel will only permit keys to be added to `.builtin_trusted_keys` **if** the new key's X.509 wrapper is validly signed by a key that is already resident in the `.builtin_trusted_keys` at the time the key was added.

61.5 Manually signing modules

To manually sign a module, use the scripts/sign-file tool available in the Linux kernel source tree. The script requires 4 arguments:

1. The hash algorithm (e.g., sha256)
2. The private key filename or PKCS#11 URI
3. The public key filename
4. The kernel module to be signed

The following is an example to sign a kernel module:

```
scripts/sign-file sha512 kernel-signkey.priv \
        kernel-signkey.x509 module.ko
```

The hash algorithm used does not have to match the one configured, but if it doesn't, you should make sure that hash algorithm is either built into the kernel or can be loaded without requiring itself.

If the private key requires a passphrase or PIN, it can be provided in the $KBUILD_SIGN_PIN environment variable.

61.6 Signed modules and stripping

A signed module has a digital signature simply appended at the end. The string `~Module signature appended~.` at the end of the module's file confirms that a signature is present but it does not confirm that the signature is valid!

Signed modules are BRITTLE as the signature is outside of the defined ELF container. Thus they MAY NOT be stripped once the signature is computed and attached. Note the entire module is the signed payload, including any and all debug information present at the time of signing.

61.7 Loading signed modules

Modules are loaded with insmod, modprobe, `init_module()` or `finit_module()`, exactly as for unsigned modules as no processing is done in userspace. The signature checking is all done within the kernel.

61.8 Non-valid signatures and unsigned modules

If `CONFIG_MODULE_SIG_FORCE` is enabled or module.sig_enforce=1 is supplied on the kernel command line, the kernel will only load validly signed modules for which it has a public key. Otherwise, it will also load modules that are unsigned. Any module for which the kernel has a key, but which proves to have a signature mismatch will not be permitted to load.

Any module that has an unparsable signature will be rejected.

61.9 Administering/protecting the private key

Since the private key is used to sign modules, viruses and malware could use the private key to sign modules and compromise the operating system. The private key must be either destroyed or moved to a secure location and not kept in the root node of the kernel source tree.

If you use the same private key to sign modules for multiple kernel configurations, you must ensure that the module version information is sufficient to prevent loading a module into a different kernel. Either set `CONFIG_MODVERSIONS=y` or ensure that each configuration has a different kernel release string by changing `EXTRAVERSION` or `CONFIG_LOCALVERSION`.

CHAPTER

SIXTYTWO

MONO(TM) BINARY KERNEL SUPPORT FOR LINUX

To configure Linux to automatically execute Mono-based .NET binaries (in the form of .exe files) without the need to use the mono CLR wrapper, you can use the BINFMT_MISC kernel support.

This will allow you to execute Mono-based .NET binaries just like any other program after you have done the following:

1) You MUST FIRST install the Mono CLR support, either by downloading a binary package, a source tarball or by installing from Git. Binary packages for several distributions can be found at:

   https://www.mono-project.com/download/

   Instructions for compiling Mono can be found at:

   https://www.mono-project.com/docs/compiling-mono/linux/

   Once the Mono CLR support has been installed, just check that `/usr/bin/mono` (which could be located elsewhere, for example `/usr/local/bin/mono`) is working.

2) You have to compile BINFMT_MISC either as a module or into the kernel (`CONFIG_BINFMT_MISC`) and set it up properly. If you choose to compile it as a module, you will have to insert it manually with modprobe/insmod, as kmod cannot be easily supported with binfmt_misc. Read the file `binfmt_misc.txt` in this directory to know more about the configuration process.

3) Add the following entries to `/etc/rc.local` or similar script to be run at system startup:

```
# Insert BINFMT_MISC module into the kernel
if [ ! -e /proc/sys/fs/binfmt_misc/register ]; then
    /sbin/modprobe binfmt_misc
    # Some distributions, like Fedora Core, perform
    # the following command automatically when the
    # binfmt_misc module is loaded into the kernel
    # or during normal boot up (systemd-based systems).
    # Thus, it is possible that the following line
    # is not needed at all.
    mount -t binfmt_misc none /proc/sys/fs/binfmt_misc
fi

# Register support for .NET CLR binaries
if [ -e /proc/sys/fs/binfmt_misc/register ]; then
    # Replace /usr/bin/mono with the correct pathname to
    # the Mono CLR runtime (usually /usr/local/bin/mono
```

```
    # when compiling from sources or CVS).
    echo ':CLR:M::MZ::/usr/bin/mono:' > /proc/sys/fs/binfmt_misc/register
else
    echo "No binfmt_misc support"
    exit 1
fi
```

4) Check that `.exe` binaries can be ran without the need of a wrapper script, simply by launching the `.exe` file directly from a command prompt, for example:

```
/usr/bin/xsd.exe
```

Note: If this fails with a permission denied error, check that the `.exe` file has execute permissions.

CHAPTER

SIXTYTHREE

NAMESPACES

63.1 Namespaces compatibility list

This document contains the information about the problems user may have when creating tasks living in different namespaces.

Here's the summary. This matrix shows the known problems, that occur when tasks share some namespace (the columns) while living in different other namespaces (the rows):

•	UTS	IPC	VFS	PID	User	Net
UTS	X					
IPC		X	1			
VFS			X			
PID		1	1	X		
User		2	2		X	
Net						X

1. Both the IPC and the PID namespaces provide IDs to address object inside the kernel. E.g. semaphore with IPCID or process group with pid.

   In both cases, tasks shouldn't try exposing this ID to some other task living in a different namespace via a shared filesystem or IPC shmem/message. The fact is that this ID is only valid within the namespace it was obtained in and may refer to some other object in another namespace.

2. Intentionally, two equal user IDs in different user namespaces should not be equal from the VFS point of view. In other words, user 10 in one user namespace shouldn't have the same access permissions to files, belonging to user 10 in another namespace.

   The same is true for the IPC namespaces being shared - two users from different user namespaces should not access the same IPC objects even having equal UIDs.

   But currently this is not so.

63.2 Namespaces research control

There are a lot of kinds of objects in the kernel that don't have individual limits or that have limits that are ineffective when a set of processes is allowed to switch user ids. With user namespaces enabled in a kernel for people who don't trust their users or their users programs to play nice this problems becomes more acute.

Therefore it is recommended that memory control groups be enabled in kernels that enable user namespaces, and it is further recommended that userspace configure memory control groups to limit how much memory user's they don't trust to play nice can use.

Memory control groups can be configured by installing the libcgroup package present on most distros editing /etc/cgrules.conf, /etc/cgconfig.conf and setting up libpam-cgroup.

CHAPTER

SIXTYFOUR

NUMA POLICY HIT/MISS STATISTICS

/sys/devices/system/node/node*/numastat

All units are pages. Hugepages have separate counters.

The numa_hit, numa_miss and numa_foreign counters reflect how well processes are able to allocate memory from nodes they prefer. If they succeed, numa_hit is incremented on the preferred node, otherwise numa_foreign is incremented on the preferred node and numa_miss on the node where allocation succeeded.

Usually preferred node is the one local to the CPU where the process executes, but restrictions such as mempolicies can change that, so there are also two counters based on CPU local node. local_node is similar to numa_hit and is incremented on allocation from a node by CPU on the same node. other_node is similar to numa_miss and is incremented on the node where allocation succeeds from a CPU from a different node. Note there is no counter analogical to numa_foreign.

In more detail:

numa_hit	A process wanted to allocate memory from this node, and succeeded.
numa_miss	A process wanted to allocate memory from another node, but ended up with memory from this node.
numa_foreign	A process wanted to allocate on this node, but ended up with memory from another node.
local_node	A process ran on this node's CPU, and got memory from this node.
other_node	A process ran on a different node's CPU and got memory from this node.
interleave_hit	Interleaving wanted to allocate from this node and succeeded.

For easier reading you can use the numastat utility from the numactl package (http://oss.sgi.com/projects/libnuma/). Note that it only works well right now on machines with a small number of CPUs.

Note that on systems with memoryless nodes (where a node has CPUs but no memory) the numa_hit, numa_miss and numa_foreign statistics can be skewed heavily. In the current kernel implementation, if a process prefers a memoryless node (i.e. because it is running on one of its local CPU), the implementation actually treats one of the nearest nodes with memory as the preferred node. As a result, such allocation will not increase the numa_foreign counter on the memoryless node, and will skew the numa_hit, numa_miss and numa_foreign statistics of the nearest node.

CHAPTER

SIXTYFIVE

PARPORT

The `parport` code provides parallel-port support under Linux. This includes the ability to share one port between multiple device drivers.

You can pass parameters to the `parport` code to override its automatic detection of your hardware. This is particularly useful if you want to use IRQs, since in general these can't be autoprobed successfully. By default IRQs are not used even if they **can** be probed. This is because there are a lot of people using the same IRQ for their parallel port and a sound card or network card.

The `parport` code is split into two parts: generic (which deals with port-sharing) and architecture-dependent (which deals with actually using the port).

65.1 Parport as modules

If you load the *parport`* code as a module, say:

```
# insmod parport
```

to load the generic `parport` code. You then must load the architecture-dependent code with (for example):

```
# insmod parport_pc io=0x3bc,0x378,0x278 irq=none,7,auto
```

to tell the `parport` code that you want three PC-style ports, one at 0x3bc with no IRQ, one at 0x378 using IRQ 7, and one at 0x278 with an auto-detected IRQ. Currently, PC-style (`parport_pc`), Sun `bpp`, Amiga, Atari, and MFC3 hardware is supported.

PCI parallel I/O card support comes from `parport_pc`. Base I/O addresses should not be specified for supported PCI cards since they are automatically detected.

65.1.1 modprobe

If you use modprobe , you will find it useful to add lines as below to a configuration file in /etc/modprobe.d/ directory:

```
alias parport_lowlevel parport_pc
options parport_pc io=0x378,0x278 irq=7,auto
```

modprobe will load `parport_pc` (with the options `io=0x378,0x278 irq=7,auto`) whenever a parallel port device driver (such as `lp`) is loaded.

Note that these are example lines only! You shouldn't in general need to specify any options to `parport_pc` in order to be able to use a parallel port.

65.1.2 Parport probe [optional]

In 2.2 kernels there was a module called `parport_probe`, which was used for collecting IEEE 1284 device ID information. This has now been enhanced and now lives with the IEEE 1284 support. When a parallel port is detected, the devices that are connected to it are analysed, and information is logged like this:

```
parport0: Printer, BJC-210 (Canon)
```

The probe information is available from files in `/proc/sys/dev/parport/`.

65.2 Parport linked into the kernel statically

If you compile the `parport` code into the kernel, then you can use kernel boot parameters to get the same effect. Add something like the following to your LILO command line:

```
parport=0x3bc parport=0x378,7 parport=0x278,auto,nofifo
```

You can have many `parport=...` statements, one for each port you want to add. Adding `parport=0` to the kernel command-line will disable parport support entirely. Adding `parport=auto` to the kernel command-line will make `parport` use any IRQ lines or DMA channels that it auto-detects.

65.3 Files in /proc

If you have configured the `/proc` filesystem into your kernel, you will see a new directory entry: `/proc/sys/dev/parport`. In there will be a directory entry for each parallel port for which parport is configured. In each of those directories are a collection of files describing that parallel port.

The `/proc/sys/dev/parport` directory tree looks like:

```
parport
|-- default
|   |-- spintime
```

```
|   `-- timeslice
|-- parport0
|   |-- autoprobe
|   |-- autoprobe0
|   |-- autoprobe1
|   |-- autoprobe2
|   |-- autoprobe3
|   |-- devices
|   |   |-- active
|   |   `-- lp
|   |       `-- timeslice
|   |-- base-addr
|   |-- irq
|   |-- dma
|   |-- modes
|   `-- spintime
`-- parport1
|-- autoprobe
|-- autoprobe0
|-- autoprobe1
|-- autoprobe2
|-- autoprobe3
|-- devices
|   |-- active
|   `-- ppa
|       `-- timeslice
|-- base-addr
|-- irq
|-- dma
|-- modes
`-- spintime
```

File	Contents
`devices/active`	A list of the device drivers using that port. A ”+” will appear by the name of the device currently using the port (it might not appear against any). The string ”none” means that there are no device drivers using that port.
`base-addr`	Parallel port's base address, or addresses if the port has more than one in which case they are separated with tabs. These values might not have any sensible meaning for some ports.
`irq`	Parallel port's IRQ, or -1 if none is being used.
`dma`	Parallel port's DMA channel, or -1 if none is being used.
`modes`	Parallel port's hardware modes, comma-separated, meaning: • **PCSPP** PC-style SPP registers are available. • **TRISTATE** Port is bidirectional. • **COMPAT** Hardware acceleration for printers is available and will be used. • **EPP** Hardware acceleration for EPP protocol is available and will be used. • **ECP** Hardware acceleration for ECP protocol is available and will be used. • **DMA** DMA is available and will be used. Note that the current implementation will only take advantage of COMPAT and ECP modes if it has an IRQ line to use.
`autoprobe`	Any IEEE-1284 device ID information that has been acquired from the (non-IEEE 1284.3) device.
`autoprobe[0-3]`	IEEE 1284 device ID information retrieved from daisy-chain devices that conform to IEEE 1284.3.
`spintime`	The number of microseconds to busy-loop while waiting for the peripheral to respond. You might find that adjusting this improves performance, depending on your peripherals. This is a port-wide setting, i.e. it applies to all devices on a particular port.
`timeslice`	The number of milliseconds that a device driver is allowed to keep a port claimed for. This is advisory, and driver can ignore it if it must.
`default/*`	The defaults for spintime and timeslice. When a new port is registered, it picks up the default spintime. When a new device is registered, it picks up the default timeslice.

65.4 Device drivers

Once the parport code is initialised, you can attach device drivers to specific ports. Normally this happens automatically; if the lp driver is loaded it will create one lp device for each port found. You can override this, though, by using parameters either when you load the lp driver:

```
# insmod lp parport=0,2
```

or on the LILO command line:

```
lp=parport0 lp=parport2
```

Both the above examples would inform lp that you want `/dev/lp0` to be the first parallel port, and /dev/lp1 to be the **third** parallel port, with no lp device associated with the second port (parport1). Note that this is different to the way older kernels worked; there used to be a static association between the I/O port address and the device name, so `/dev/lp0` was always the port at 0x3bc. This is no longer the case - if you only have one port, it will default to being `/dev/lp0`, regardless of base address.

Also:

- If you selected the IEEE 1284 support at compile time, you can say `lp=auto` on the kernel command line, and lp will create devices only for those ports that seem to have printers attached.
- If you give PLIP the `timid` parameter, either with `plip=timid` on the command line, or with `insmod plip timid=1` when using modules, it will avoid any ports that seem to be in use by other devices.
- IRQ autoprobing works only for a few port types at the moment.

65.5 Reporting printer problems with parport

If you are having problems printing, please go through these steps to try to narrow down where the problem area is.

When reporting problems with parport, really you need to give all of the messages that `parport_pc` spits out when it initialises. There are several code paths:

- polling
- interrupt-driven, protocol in software
- interrupt-driven, protocol in hardware using PIO
- interrupt-driven, protocol in hardware using DMA

The kernel messages that `parport_pc` logs give an indication of which code path is being used. (They could be a lot better actually..)

For normal printer protocol, having IEEE 1284 modes enabled or not should not make a difference.

To turn off the 'protocol in hardware' code paths, disable `CONFIG_PARPORT_PC_FIFO`. Note that when they are enabled they are not necessarily **used**; it depends on whether the hardware is available, enabled by the BIOS, and detected by the driver.

So, to start with, disable `CONFIG_PARPORT_PC_FIFO`, and load `parport_pc` with `irq=none`. See if printing works then. It really should, because this is the simplest code path.

If that works fine, try with `io=0x378 irq=7` (adjust for your hardware), to make it use interrupt-driven in-software protocol.

If **that** works fine, then one of the hardware modes isn't working right. Enable `CONFIG_FIFO` (no, it isn't a module option, and yes, it should be), set the port to ECP mode in the BIOS and note the DMA channel, and try with:

```
io=0x378 irq=7 dma=none (for PIO)
io=0x378 irq=7 dma=3 (for DMA)
```

---

philb@gnu.org tim@cyberelk.net

CHAPTER

SIXTYSIX

PERF EVENTS AND TOOL SECURITY

66.1 Overview

Usage of Performance Counters for Linux (perf_events)[1] ,[2] ,[3] can impose a considerable risk of leaking sensitive data accessed by monitored processes. The data leakage is possible both in scenarios of direct usage of perf_events system call APIPage 2503, 2 and over data files generated by Perf tool user mode utility (Perf)[3] ,[4] . The risk depends on the nature of data that perf_events performance monitoring units (PMU)[2] and Perf collect and expose for performance analysis. Collected system and performance data may be split into several categories:

1. System hardware and software configuration data, for example: a CPU model and its cache configuration, an amount of available memory and its topology, used kernel and Perf versions, performance monitoring setup including experiment time, events configuration, Perf command line parameters, etc.
2. User and kernel module paths and their load addresses with sizes, process and thread names with their PIDs and TIDs, timestamps for captured hardware and software events.
3. Content of kernel software counters (e.g., for context switches, page faults, CPU migrations), architectural hardware performance counters (PMC)[8] and machine specific registers (MSR)[9] that provide execution metrics for various monitored parts of the system (e.g., memory controller (IMC), interconnect (QPI/UPI) or peripheral (PCIe) uncore counters) without direct attribution to any execution context state.
4. Content of architectural execution context registers (e.g., RIP, RSP, RBP on x86_64), process user and kernel space memory addresses and data, content of various architectural MSRs that capture data from this category.

Data that belong to the fourth category can potentially contain sensitive process data. If PMUs in some monitoring modes capture values of execution context registers or data from process memory then access to such monitoring modes requires to be ordered and secured properly. So, perf_events performance monitoring and observability operations are the subject for security access control management[5] .

[1] https://lwn.net/Articles/337493/
[2] http://man7.org/linux/man-pages/man2/perf_event_open.2.html
[3] http://web.eece.maine.edu/~vweaver/projects/perf_events/
[4] https://perf.wiki.kernel.org/index.php/Main_Page
[8] https://en.wikipedia.org/wiki/Hardware_performance_counter
[9] https://en.wikipedia.org/wiki/Model-specific_register
[5] https://www.kernel.org/doc/html/latest/security/credentials.html

66.2 perf_events access control

To perform security checks, the Linux implementation splits processes into two categories[6] : a) privileged processes (whose effective user ID is 0, referred to as superuser or root), and b) unprivileged processes (whose effective UID is nonzero). Privileged processes bypass all kernel security permission checks so perf_events performance monitoring is fully available to privileged processes without access, scope and resource restrictions.

Unprivileged processes are subject to a full security permission check based on the process's credentials[5] (usually: effective UID, effective GID, and supplementary group list).

Linux divides the privileges traditionally associated with superuser into distinct units, known as capabilities[6] , which can be independently enabled and disabled on per-thread basis for processes and files of unprivileged users.

Unprivileged processes with enabled CAP_PERFMON capability are treated as privileged processes with respect to perf_events performance monitoring and observability operations, thus, bypass *scope* permissions checks in the kernel. CAP_PERFMON implements the principle of least privilege[13] (POSIX 1003.1e: 2.2.2.39) for performance monitoring and observability operations in the kernel and provides a secure approach to performance monitoring and observability in the system.

For backward compatibility reasons the access to perf_events monitoring and observability operations is also open for CAP_SYS_ADMIN privileged processes but CAP_SYS_ADMIN usage for secure monitoring and observability use cases is discouraged with respect to the CAP_PERFMON capability. If system audit records[14] for a process using perf_events system call API contain denial records of acquiring both CAP_PERFMON and CAP_SYS_ADMIN capabilities then providing the process with CAP_PERFMON capability singly is recommended as the preferred secure approach to resolve double access denial logging related to usage of performance monitoring and observability.

Prior Linux v5.9 unprivileged processes using perf_events system call are also subject for PTRACE_MODE_READ_REALCREDS ptrace access mode check[7] , whose outcome determines whether monitoring is permitted. So unprivileged processes provided with CAP_SYS_PTRACE capability are effectively permitted to pass the check. Starting from Linux v5.9 CAP_SYS_PTRACE capability is not required and CAP_PERFMON is enough to be provided for processes to make performance monitoring and observability operations.

Other capabilities being granted to unprivileged processes can effectively enable capturing of additional data required for later performance analysis of monitored processes or a system. For example, CAP_SYSLOG capability permits reading kernel space memory addresses from /proc/kallsyms file.

[6] http://man7.org/linux/man-pages/man7/capabilities.7.html
[13] https://sites.google.com/site/fullycapable
[14] http://man7.org/linux/man-pages/man8/auditd.8.html
[7] http://man7.org/linux/man-pages/man2/ptrace.2.html

66.3 Privileged Perf users groups

Mechanisms of capabilities, privileged capability-dumb files[6], file system ACLs[10] and sudo[15] utility can be used to create dedicated groups of privileged Perf users who are permitted to execute performance monitoring and observability without limits. The following steps can be taken to create such groups of privileged Perf users.

1. Create perf_users group of privileged Perf users, assign perf_users group to Perf tool executable and limit access to the executable for other users in the system who are not in the perf_users group:

```
# groupadd perf_users
# ls -alhF
-rwxr-xr-x  2 root root  11M Oct 19 15:12 perf
# chgrp perf_users perf
# ls -alhF
-rwxr-xr-x  2 root perf_users  11M Oct 19 15:12 perf
# chmod o-rwx perf
# ls -alhF
-rwxr-x---  2 root perf_users  11M Oct 19 15:12 perf
```

2. Assign the required capabilities to the Perf tool executable file and enable members of perf_users group with monitoring and observability privileges[Page 2504, 6] :

```
# setcap "cap_perfmon,cap_sys_ptrace,cap_syslog=ep" perf
# setcap -v "cap_perfmon,cap_sys_ptrace,cap_syslog=ep" perf
perf: OK
# getcap perf
perf = cap_sys_ptrace,cap_syslog,cap_perfmon+ep
```

If the libcap[16] installed doesn't yet support ”cap_perfmon”, use ”38” instead, i.e.:

```
# setcap "38,cap_ipc_lock,cap_sys_ptrace,cap_syslog=ep" perf
```

Note that you may need to have 'cap_ipc_lock' in the mix for tools such as 'perf top', alternatively use 'perf top -m N', to reduce the memory that it uses for the perf ring buffer, see the memory allocation section below.

Using a libcap without support for CAP_PERFMON will make cap_get_flag(caps, 38, CAP_EFFECTIVE, &val) fail, which will lead the default event to be 'cycles:u', so as a workaround explicitly ask for the 'cycles' event, i.e.:

```
# perf top -e cycles
```

To get kernel and user samples with a perf binary with just CAP_PERFMON.

As a result, members of perf_users group are capable of conducting performance monitoring and observability by using functionality of the configured Perf tool executable that, when executes, passes perf_events subsystem scope checks.

[10] http://man7.org/linux/man-pages/man5/acl.5.html
[15] https://man7.org/linux/man-pages/man8/sudo.8.html
[16] https://git.kernel.org/pub/scm/libs/libcap/libcap.git/

In case Perf tool executable can't be assigned required capabilities (e.g. file system is mounted with nosuid option or extended attributes are not supported by the file system) then creation of the capabilities privileged environment, naturally shell, is possible. The shell provides inherent processes with CAP_PERFMON and other required capabilities so that performance monitoring and observability operations are available in the environment without limits. Access to the environment can be open via sudo utility for members of perf_users group only. In order to create such environment:

1. Create shell script that uses capsh utility[16] to assign CAP_PERFMON and other required capabilities into ambient capability set of the shell process, lock the process security bits after enabling SECBIT_NO_SETUID_FIXUP, SECBIT_NOROOT and SECBIT_NO_CAP_AMBIENT_RAISE bits and then change the process identity to sudo caller of the script who should essentially be a member of perf_users group:

```
# ls -alh /usr/local/bin/perf.shell
-rwxr-xr-x. 1 root root 83 Oct 13 23:57 /usr/local/bin/perf.shell
# cat /usr/local/bin/perf.shell
exec /usr/sbin/capsh --iab=^cap_perfmon --secbits=239 --user=$SUDO_USER -- -l
```

2. Extend sudo policy at /etc/sudoers file with a rule for perf_users group:

```
# grep perf_users /etc/sudoers
%perf_users    ALL=/usr/local/bin/perf.shell
```

3. Check that members of perf_users group have access to the privileged shell and have CAP_PERFMON and other required capabilities enabled in permitted, effective and ambient capability sets of an inherent process:

```
$ id
uid=1003(capsh_test) gid=1004(capsh_test) groups=1004(capsh_test),1000(perf_
↪users) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
$ sudo perf.shell
[sudo] password for capsh_test:
$ grep Cap /proc/self/status
CapInh:        0000004000000000
CapPrm:        0000004000000000
CapEff:        0000004000000000
CapBnd:        000000ffffffffff
CapAmb:        0000004000000000
$ capsh --decode=0000004000000000
0x0000004000000000=cap_perfmon
```

As a result, members of perf_users group have access to the privileged environment where they can use tools employing performance monitoring APIs governed by CAP_PERFMON Linux capability.

This specific access control management is only available to superuser or root running processes with CAP_SETPCAP, CAP_SETFCAP[Page 2504, 6] capabilities.

66.4 Unprivileged users

perf_events *scope* and *access* control for unprivileged processes is governed by perf_event_paranoid[Page 2503, 2] setting:

-1:
Impose no *scope* and *access* restrictions on using perf_events performance monitoring. Per-user per-cpu perf_event_mlock_kb[Page 2503, 2] locking limit is ignored when allocating memory buffers for storing performance data. This is the least secure mode since allowed monitored *scope* is maximized and no perf_events specific limits are imposed on *resources* allocated for performance monitoring.

>=0:
scope includes per-process and system wide performance monitoring but excludes raw tracepoints and ftrace function tracepoints monitoring. CPU and system events happened when executing either in user or in kernel space can be monitored and captured for later analysis. Per-user per-cpu perf_event_mlock_kb locking limit is imposed but ignored for unprivileged processes with CAP_IPC_LOCK[Page 2504, 6] capability.

>=1:
scope includes per-process performance monitoring only and excludes system wide performance monitoring. CPU and system events happened when executing either in user or in kernel space can be monitored and captured for later analysis. Per-user per-cpu perf_event_mlock_kb locking limit is imposed but ignored for unprivileged processes with CAP_IPC_LOCK capability.

>=2:
scope includes per-process performance monitoring only. CPU and system events happened when executing in user space only can be monitored and captured for later analysis. Per-user per-cpu perf_event_mlock_kb locking limit is imposed but ignored for unprivileged processes with CAP_IPC_LOCK capability.

66.5 Resource control

66.5.1 Open file descriptors

The perf_events system call API[Page 2503, 2] allocates file descriptors for every configured PMU event. Open file descriptors are a per-process accountable resource governed by the RLIMIT_NOFILE[11] limit (ulimit -n), which is usually derived from the login shell process. When configuring Perf collection for a long list of events on a large server system, this limit can be easily hit preventing required monitoring configuration. RLIMIT_NOFILE limit can be increased on per-user basis modifying content of the limits.conf file[12] . Ordinarily, a Perf sampling session (perf record) requires an amount of open perf_event file descriptors that is not less than the number of monitored events multiplied by the number of monitored CPUs.

[11] http://man7.org/linux/man-pages/man2/getrlimit.2.html
[12] http://man7.org/linux/man-pages/man5/limits.conf.5.html

66.5.2 Memory allocation

The amount of memory available to user processes for capturing performance monitoring data is governed by the perf_event_mlock_kb[Page 2503, 2] setting. This perf_event specific resource setting defines overall per-cpu limits of memory allowed for mapping by the user processes to execute performance monitoring. The setting essentially extends the RLIMIT_MEMLOCK[Page 2507, 11] limit, but only for memory regions mapped specifically for capturing monitored performance events and related data.

For example, if a machine has eight cores and perf_event_mlock_kb limit is set to 516 KiB, then a user process is provided with 516 KiB * 8 = 4128 KiB of memory above the RLIMIT_MEMLOCK limit (ulimit -l) for perf_event mmap buffers. In particular, this means that, if the user wants to start two or more performance monitoring processes, the user is required to manually distribute the available 4128 KiB between the monitoring processes, for example, using the --mmap-pages Perf record mode option. Otherwise, the first started performance monitoring process allocates all available 4128 KiB and the other processes will fail to proceed due to the lack of memory.

RLIMIT_MEMLOCK and perf_event_mlock_kb resource constraints are ignored for processes with the CAP_IPC_LOCK capability. Thus, perf_events/Perf privileged users can be provided with memory above the constraints for perf_events/Perf performance monitoring purpose by providing the Perf executable with CAP_IPC_LOCK capability.

66.6 Bibliography

CHAPTER

SIXTYSEVEN

POWER MANAGEMENT

67.1 Power Management Strategies

Copyright
© 2017 Intel Corporation

Author
Rafael J. Wysocki <rafael.j.wysocki@intel.com>

The Linux kernel supports two major high-level power management strategies.

One of them is based on using global low-power states of the whole system in which user space code cannot be executed and the overall system activity is significantly reduced, referred to as *sleep states*. The kernel puts the system into one of these states when requested by user space and the system stays in it until a special signal is received from one of designated devices, triggering a transition to the `working state` in which user space code can run. Because sleep states are global and the whole system is affected by the state changes, this strategy is referred to as the *system-wide power management*.

The other strategy, referred to as the *working-state power management*, is based on adjusting the power states of individual hardware components of the system, as needed, in the working state. In consequence, if this strategy is in use, the working state of the system usually does not correspond to any particular physical configuration of it, but can be treated as a metastate covering a range of different power states of the system in which the individual components of it can be either `active` (in use) or `inactive` (idle). If they are active, they have to be in power states allowing them to process data and to be accessed by software. In turn, if they are inactive, ideally, they should be in low-power states in which they may not be accessible.

If all of the system components are active, the system as a whole is regarded as ”runtime active” and that situation typically corresponds to the maximum power draw (or maximum energy usage) of it. If all of them are inactive, the system as a whole is regarded as ”runtime idle” which may be very close to a sleep state from the physical system configuration and power draw perspective, but then it takes much less time and effort to start executing user space code than for the same system in a sleep state. However, transitions from sleep states back to the working state can only be started by a limited set of devices, so typically the system can spend much more time in a sleep state than it can be runtime idle in one go. For this reason, systems usually use less energy in sleep states than when they are runtime idle most of the time.

Moreover, the two power management strategies address different usage scenarios. Namely, if the user indicates that the system will not be in use going forward, for example by closing its lid (if the system is a laptop), it probably should go into a sleep state at that point. On the other hand, if the user simply goes away from the laptop keyboard, it probably should stay in the

working state and use the working-state power management in case it becomes idle, because the user may come back to it at any time and then may want the system to be immediately accessible.

67.2 System-Wide Power Management

67.2.1 System Sleep States

Copyright
© 2017 Intel Corporation

Author
Rafael J. Wysocki <rafael.j.wysocki@intel.com>

Sleep states are global low-power states of the entire system in which user space code cannot be executed and the overall system activity is significantly reduced.

Sleep States That Can Be Supported

Depending on its configuration and the capabilities of the platform it runs on, the Linux kernel can support up to four system sleep states, including hibernation and up to three variants of system suspend. The sleep states that can be supported by the kernel are listed below.

Suspend-to-Idle

This is a generic, pure software, light-weight variant of system suspend (also referred to as S2I or S2Idle). It allows more energy to be saved relative to runtime idle by freezing user space, suspending the timekeeping and putting all I/O devices into low-power states (possibly lower-power than available in the working state), such that the processors can spend time in their deepest idle states while the system is suspended.

The system is woken up from this state by in-band interrupts, so theoretically any devices that can cause interrupts to be generated in the working state can also be set up as wakeup devices for S2Idle.

This state can be used on platforms without support for *standby* or *suspend-to-RAM*, or it can be used in addition to any of the deeper system suspend variants to provide reduced resume latency. It is always supported if the `CONFIG_SUSPEND` kernel configuration option is set.

Standby

This state, if supported, offers moderate, but real, energy savings, while providing a relatively straightforward transition back to the working state. No operating state is lost (the system core logic retains power), so the system can go back to where it left off easily enough.

In addition to freezing user space, suspending the timekeeping and putting all I/O devices into low-power states, which is done for *suspend-to-idle* too, nonboot CPUs are taken offline and all low-level system functions are suspended during transitions into this state. For this reason, it should allow more energy to be saved relative to *suspend-to-idle*, but the resume latency will generally be greater than for that state.

The set of devices that can wake up the system from this state usually is reduced relative to *suspend-to-idle* and it may be necessary to rely on the platform for setting up the wakeup functionality as appropriate.

This state is supported if the `CONFIG_SUSPEND` kernel configuration option is set and the support for it is registered by the platform with the core system suspend subsystem. On ACPI-based systems this state is mapped to the S1 system state defined by ACPI.

Suspend-to-RAM

This state (also referred to as STR or S2RAM), if supported, offers significant energy savings as everything in the system is put into a low-power state, except for memory, which should be placed into the self-refresh mode to retain its contents. All of the steps carried out when entering *standby* are also carried out during transitions to S2RAM. Additional operations may take place depending on the platform capabilities. In particular, on ACPI-based systems the kernel passes control to the platform firmware (BIOS) as the last step during S2RAM transitions and that usually results in powering down some more low-level components that are not directly controlled by the kernel.

The state of devices and CPUs is saved and held in memory. All devices are suspended and put into low-power states. In many cases, all peripheral buses lose power when entering S2RAM, so devices must be able to handle the transition back to the ”on” state.

On ACPI-based systems S2RAM requires some minimal boot-strapping code in the platform firmware to resume the system from it. This may be the case on other platforms too.

The set of devices that can wake up the system from S2RAM usually is reduced relative to *suspend-to-idle* and *standby* and it may be necessary to rely on the platform for setting up the wakeup functionality as appropriate.

S2RAM is supported if the `CONFIG_SUSPEND` kernel configuration option is set and the support for it is registered by the platform with the core system suspend subsystem. On ACPI-based systems it is mapped to the S3 system state defined by ACPI.

Hibernation

This state (also referred to as Suspend-to-Disk or STD) offers the greatest energy savings and can be used even in the absence of low-level platform support for system suspend. However, it requires some low-level code for resuming the system to be present for the underlying CPU architecture.

Hibernation is significantly different from any of the system suspend variants. It takes three system state changes to put it into hibernation and two system state changes to resume it.

First, when hibernation is triggered, the kernel stops all system activity and creates a snapshot image of memory to be written into persistent storage. Next, the system goes into a state in which the snapshot image can be saved, the image is written out and finally the system goes into the target low-power state in which power is cut from almost all of its hardware components, including memory, except for a limited set of wakeup devices.

Once the snapshot image has been written out, the system may either enter a special low-power state (like ACPI S4), or it may simply power down itself. Powering down means minimum power

draw and it allows this mechanism to work on any system. However, entering a special low-power state may allow additional means of system wakeup to be used (e.g. pressing a key on the keyboard or opening a laptop lid).

After wakeup, control goes to the platform firmware that runs a boot loader which boots a fresh instance of the kernel (control may also go directly to the boot loader, depending on the system configuration, but anyway it causes a fresh instance of the kernel to be booted). That new instance of the kernel (referred to as the `restore kernel`) looks for a hibernation image in persistent storage and if one is found, it is loaded into memory. Next, all activity in the system is stopped and the restore kernel overwrites itself with the image contents and jumps into a special trampoline area in the original kernel stored in the image (referred to as the `image kernel`), which is where the special architecture-specific low-level code is needed. Finally, the image kernel restores the system to the pre-hibernation state and allows user space to run again.

Hibernation is supported if the `CONFIG_HIBERNATION` kernel configuration option is set. However, this option can only be set if support for the given CPU architecture includes the low-level code for system resume.

Basic `sysfs` Interfaces for System Suspend and Hibernation

The power management subsystem provides userspace with a unified `sysfs` interface for system sleep regardless of the underlying system architecture or platform. That interface is located in the `/sys/power/` directory (assuming that `sysfs` is mounted at `/sys`) and it consists of the following attributes (files):

`state`
This file contains a list of strings representing sleep states supported by the kernel. Writing one of these strings into it causes the kernel to start a transition of the system into the sleep state represented by that string.

In particular, the ”disk”, ”freeze” and ”standby” strings represent the *hibernation*, *suspend-to-idle* and *standby* sleep states, respectively. The ”mem” string is interpreted in accordance with the contents of the `mem_sleep` file described below.

If the kernel does not support any system sleep states, this file is not present.

`mem_sleep`
This file contains a list of strings representing supported system suspend variants and allows user space to select the variant to be associated with the ”mem” string in the `state` file described above.

The strings that may be present in this file are ”s2idle”, ”shallow” and ”deep”. The ”s2idle” string always represents *suspend-to-idle* and, by convention, ”shallow” and ”deep” represent *standby* and *suspend-to-RAM*, respectively.

Writing one of the listed strings into this file causes the system suspend variant represented by it to be associated with the ”mem” string in the `state` file. The string representing the suspend variant currently associated with the ”mem” string in the `state` file is shown in square brackets.

If the kernel does not support system suspend, this file is not present.

`disk`
This file controls the operating mode of hibernation (Suspend-to-Disk). Specifically, it tells the kernel what to do after creating a hibernation image.

Reading from it returns a list of supported options encoded as:

platform
Put the system into a special low-power state (e.g. ACPI S4) to make additional wakeup options available and possibly allow the platform firmware to take a simplified initialization path after wakeup.

It is only available if the platform provides a special mechanism to put the system to sleep after creating a hibernation image (platforms with ACPI do that as a rule, for example).

shutdown
Power off the system.

reboot
Reboot the system (useful for diagnostics mostly).

suspend
Hybrid system suspend. Put the system into the suspend sleep state selected through the `mem_sleep` file described above. If the system is successfully woken up from that state, discard the hibernation image and continue. Otherwise, use the image to restore the previous state of the system.

It is available if system suspend is supported.

test_resume
Diagnostic operation. Load the image as though the system had just woken up from hibernation and the currently running kernel instance was a restore kernel and follow up with full system resume.

Writing one of the strings listed above into this file causes the option represented by it to be selected.

The currently selected option is shown in square brackets, which means that the operation represented by it will be carried out after creating and saving the image when hibernation is triggered by writing `disk` to `/sys/power/state`.

If the kernel does not support hibernation, this file is not present.

image_size
This file controls the size of hibernation images.

It can be written a string representing a non-negative integer that will be used as a best-effort upper limit of the image size, in bytes. The hibernation core will do its best to ensure that the image size will not exceed that number, but if that turns out to be impossible to achieve, a hibernation image will still be created and its size will be as small as possible. In particular, writing '0' to this file causes the size of hibernation images to be minimum.

Reading from it returns the current image size limit, which is set to around 2/5 of the available RAM size by default.

pm_trace
This file controls the "PM trace" mechanism saving the last suspend or resume event point in the RTC memory across reboots. It helps to debug hard lockups or reboots due to device driver failures that occur during system suspend or resume (which is more common) more effectively.

If it contains "1", the fingerprint of each suspend/resume event point in turn will be stored in the RTC memory (overwriting the actual RTC information), so it will survive a system

crash if one occurs right after storing it and it can be used later to identify the driver that caused the crash to happen.

It contains ”0” by default, which may be changed to ”1” by writing a string representing a nonzero integer into it.

According to the above, there are two ways to make the system go into the *suspend-to-idle* state. The first one is to write ”freeze” directly to `/sys/power/state`. The second one is to write ”s2idle” to `/sys/power/mem_sleep` and then to write ”mem” to `/sys/power/state`. Likewise, there are two ways to make the system go into the *standby* state (the strings to write to the control files in that case are ”standby” or ”shallow” and ”mem”, respectively) if that state is supported by the platform. However, there is only one way to make the system go into the *suspend-to-RAM* state (write ”deep” into `/sys/power/mem_sleep` and ”mem” into `/sys/power/state`).

The default suspend variant (ie. the one to be used without writing anything into `/sys/power/mem_sleep`) is either ”deep” (on the majority of systems supporting *suspend-to-RAM*) or ”s2idle”, but it can be overridden by the value of the `mem_sleep_default` parameter in the kernel command line. On some systems with ACPI, depending on the information in the ACPI tables, the default may be ”s2idle” even if *suspend-to-RAM* is supported in principle.

67.2.2 System Suspend Code Flows

Copyright
© 2020 Intel Corporation

Author
Rafael J. Wysocki <rafael.j.wysocki@intel.com>

At least one global system-wide transition needs to be carried out for the system to get from the working state into one of the supported *sleep states*. Hibernation requires more than one transition to occur for this purpose, but the other sleep states, commonly referred to as *system-wide suspend* (or simply *system suspend*) states, need only one.

For those sleep states, the transition from the working state of the system into the target sleep state is referred to as *system suspend* too (in the majority of cases, whether this means a transition or a sleep state of the system should be clear from the context) and the transition back from the sleep state into the working state is referred to as *system resume*.

The kernel code flows associated with the suspend and resume transitions for different sleep states of the system are quite similar, but there are some significant differences between the *suspend-to-idle* code flows and the code flows related to the *suspend-to-RAM* and *standby* sleep states.

The *suspend-to-RAM* and *standby* sleep states cannot be implemented without platform support and the difference between them boils down to the platform-specific actions carried out by the suspend and resume hooks that need to be provided by the platform driver to make them available. Apart from that, the suspend and resume code flows for these sleep states are mostly identical, so they both together will be referred to as *platform-dependent suspend* states in what follows.

Suspend-to-idle Suspend Code Flow

The following steps are taken in order to transition the system from the working state to the *suspend-to-idle* sleep state:

1. Invoking system-wide suspend notifiers.

   Kernel subsystems can register callbacks to be invoked when the suspend transition is about to occur and when the resume transition has finished.

   That allows them to prepare for the change of the system state and to clean up after getting back to the working state.

2. Freezing tasks.

   Tasks are frozen primarily in order to avoid unchecked hardware accesses from user space through MMIO regions or I/O registers exposed directly to it and to prevent user space from entering the kernel while the next step of the transition is in progress (which might have been problematic for various reasons).

   All user space tasks are intercepted as though they were sent a signal and put into uninterruptible sleep until the end of the subsequent system resume transition.

   The kernel threads that choose to be frozen during system suspend for specific reasons are frozen subsequently, but they are not intercepted. Instead, they are expected to periodically check whether or not they need to be frozen and to put themselves into uninterruptible sleep if so. [Note, however, that kernel threads can use locking and other concurrency controls available in kernel space to synchronize themselves with system suspend and resume, which can be much more precise than the freezing, so the latter is not a recommended option for kernel threads.]

3. Suspending devices and reconfiguring IRQs.

   Devices are suspended in four phases called *prepare*, *suspend*, *late suspend* and *noirq suspend* (see driverapi_pm_devices for more information on what exactly happens in each phase).

   Every device is visited in each phase, but typically it is not physically accessed in more than two of them.

   The runtime PM API is disabled for every device during the *late* suspend phase and high-level (”action”) interrupt handlers are prevented from being invoked before the *noirq* suspend phase.

   Interrupts are still handled after that, but they are only acknowledged to interrupt controllers without performing any device-specific actions that would be triggered in the working state of the system (those actions are deferred till the subsequent system resume transition as described *below*).

   IRQs associated with system wakeup devices are ”armed” so that the resume transition of the system is started when one of them signals an event.

4. Freezing the scheduler tick and suspending timekeeping.

   When all devices have been suspended, CPUs enter the idle loop and are put into the deepest available idle state. While doing that, each of them ”freezes” its own scheduler tick so that the timer events associated with the tick do not occur until the CPU is woken up by another interrupt source.

The last CPU to enter the idle state also stops the timekeeping which (among other things) prevents high resolution timers from triggering going forward until the first CPU that is woken up restarts the timekeeping. That allows the CPUs to stay in the deep idle state relatively long in one go.

From this point on, the CPUs can only be woken up by non-timer hardware interrupts. If that happens, they go back to the idle state unless the interrupt that woke up one of them comes from an IRQ that has been armed for system wakeup, in which case the system resume transition is started.

Suspend-to-idle Resume Code Flow

The following steps are taken in order to transition the system from the *suspend-to-idle* sleep state into the working state:

1. Resuming timekeeping and unfreezing the scheduler tick.

   When one of the CPUs is woken up (by a non-timer hardware interrupt), it leaves the idle state entered in the last step of the preceding suspend transition, restarts the timekeeping (unless it has been restarted already by another CPU that woke up earlier) and the scheduler tick on that CPU is unfrozen.

   If the interrupt that has woken up the CPU was armed for system wakeup, the system resume transition begins.

2. Resuming devices and restoring the working-state configuration of IRQs.

   Devices are resumed in four phases called *noirq resume*, *early resume*, *resume* and *complete* (see driverapi_pm_devices for more information on what exactly happens in each phase).

   Every device is visited in each phase, but typically it is not physically accessed in more than two of them.

   The working-state configuration of IRQs is restored after the *noirq* resume phase and the runtime PM API is re-enabled for every device whose driver supports it during the *early* resume phase.

3. Thawing tasks.

   Tasks frozen in step 2 of the preceding *suspend* transition are ”thawed”, which means that they are woken up from the uninterruptible sleep that they went into at that time and user space tasks are allowed to exit the kernel.

4. Invoking system-wide resume notifiers.

   This is analogous to step 1 of the *suspend* transition and the same set of callbacks is invoked at this point, but a different ”notification type” parameter value is passed to them.

Platform-dependent Suspend Code Flow

The following steps are taken in order to transition the system from the working state to platform-dependent suspend state:

1. Invoking system-wide suspend notifiers.

   This step is the same as step 1 of the suspend-to-idle suspend transition described *above*.

2. Freezing tasks.

   This step is the same as step 2 of the suspend-to-idle suspend transition described *above*.

3. Suspending devices and reconfiguring IRQs.

   This step is analogous to step 3 of the suspend-to-idle suspend transition described *above*, but the arming of IRQs for system wakeup generally does not have any effect on the platform.

   There are platforms that can go into a very deep low-power state internally when all CPUs in them are in sufficiently deep idle states and all I/O devices have been put into low-power states. On those platforms, suspend-to-idle can reduce system power very effectively.

   On the other platforms, however, low-level components (like interrupt controllers) need to be turned off in a platform-specific way (implemented in the hooks provided by the platform driver) to achieve comparable power reduction.

   That usually prevents in-band hardware interrupts from waking up the system, which must be done in a special platform-dependent way. Then, the configuration of system wakeup sources usually starts when system wakeup devices are suspended and is finalized by the platform suspend hooks later on.

4. Disabling non-boot CPUs.

   On some platforms the suspend hooks mentioned above must run in a one-CPU configuration of the system (in particular, the hardware cannot be accessed by any code running in parallel with the platform suspend hooks that may, and often do, trap into the platform firmware in order to finalize the suspend transition).

   For this reason, the CPU offline/online (CPU hotplug) framework is used to take all of the CPUs in the system, except for one (the boot CPU), offline (typically, the CPUs that have been taken offline go into deep idle states).

   This means that all tasks are migrated away from those CPUs and all IRQs are rerouted to the only CPU that remains online.

5. Suspending core system components.

   This prepares the core system components for (possibly) losing power going forward and suspends the timekeeping.

6. Platform-specific power removal.

   This is expected to remove power from all of the system components except for the memory controller and RAM (in order to preserve the contents of the latter) and some devices designated for system wakeup.

   In many cases control is passed to the platform firmware which is expected to finalize the suspend transition as needed.

Platform-dependent Resume Code Flow

The following steps are taken in order to transition the system from a platform-dependent suspend state into the working state:

1. Platform-specific system wakeup.

   The platform is woken up by a signal from one of the designated system wakeup devices (which need not be an in-band hardware interrupt) and control is passed back to the kernel (the working configuration of the platform may need to be restored by the platform firmware before the kernel gets control again).

2. Resuming core system components.

   The suspend-time configuration of the core system components is restored and the timekeeping is resumed.

3. Re-enabling non-boot CPUs.

   The CPUs disabled in step 4 of the preceding suspend transition are taken back online and their suspend-time configuration is restored.

4. Resuming devices and restoring the working-state configuration of IRQs.

   This step is the same as step 2 of the suspend-to-idle suspend transition described *above*.

5. Thawing tasks.

   This step is the same as step 3 of the suspend-to-idle suspend transition described *above*.

6. Invoking system-wide resume notifiers.

   This step is the same as step 4 of the suspend-to-idle suspend transition described *above*.

67.3 Working-State Power Management

67.3.1 CPU Idle Time Management

Copyright
© 2018 Intel Corporation

Author
Rafael J. Wysocki <rafael.j.wysocki@intel.com>

Concepts

Modern processors are generally able to enter states in which the execution of a program is suspended and instructions belonging to it are not fetched from memory or executed. Those states are the *idle* states of the processor.

Since part of the processor hardware is not used in idle states, entering them generally allows power drawn by the processor to be reduced and, in consequence, it is an opportunity to save energy.

CPU idle time management is an energy-efficiency feature concerned about using the idle states of processors for this purpose.

Logical CPUs

CPU idle time management operates on CPUs as seen by the *CPU scheduler* (that is the part of the kernel responsible for the distribution of computational work in the system). In its view, CPUs are *logical* units. That is, they need not be separate physical entities and may just be interfaces appearing to software as individual single-core processors. In other words, a CPU is an entity which appears to be fetching instructions that belong to one sequence (program) from memory and executing them, but it need not work this way physically. Generally, three different cases can be consider here.

First, if the whole processor can only follow one sequence of instructions (one program) at a time, it is a CPU. In that case, if the hardware is asked to enter an idle state, that applies to the processor as a whole.

Second, if the processor is multi-core, each core in it is able to follow at least one program at a time. The cores need not be entirely independent of each other (for example, they may share caches), but still most of the time they work physically in parallel with each other, so if each of them executes only one program, those programs run mostly independently of each other at the same time. The entire cores are CPUs in that case and if the hardware is asked to enter an idle state, that applies to the core that asked for it in the first place, but it also may apply to a larger unit (say a ”package” or a ”cluster”) that the core belongs to (in fact, it may apply to an entire hierarchy of larger units containing the core). Namely, if all of the cores in the larger unit except for one have been put into idle states at the ”core level” and the remaining core asks the processor to enter an idle state, that may trigger it to put the whole larger unit into an idle state which also will affect the other cores in that unit.

Finally, each core in a multi-core processor may be able to follow more than one program in the same time frame (that is, each core may be able to fetch instructions from multiple locations in memory and execute them in the same time frame, but not necessarily entirely in parallel with each other). In that case the cores present themselves to software as ”bundles” each consisting of multiple individual single-core ”processors”, referred to as *hardware threads* (or hyper-threads specifically on Intel hardware), that each can follow one sequence of instructions. Then, the hardware threads are CPUs from the CPU idle time management perspective and if the processor is asked to enter an idle state by one of them, the hardware thread (or CPU) that asked for it is stopped, but nothing more happens, unless all of the other hardware threads within the same core also have asked the processor to enter an idle state. In that situation, the core may be put into an idle state individually or a larger unit containing it may be put into an idle state as a whole (if the other cores within the larger unit are in idle states already).

Idle CPUs

Logical CPUs, simply referred to as ”CPUs” in what follows, are regarded as *idle* by the Linux kernel when there are no tasks to run on them except for the special ”idle” task.

Tasks are the CPU scheduler's representation of work. Each task consists of a sequence of instructions to execute, or code, data to be manipulated while running that code, and some context information that needs to be loaded into the processor every time the task's code is run by a CPU. The CPU scheduler distributes work by assigning tasks to run to the CPUs present in the system.

Tasks can be in various states. In particular, they are *runnable* if there are no specific conditions preventing their code from being run by a CPU as long as there is a CPU available for that (for example, they are not waiting for any events to occur or similar). When a task becomes

runnable, the CPU scheduler assigns it to one of the available CPUs to run and if there are no more runnable tasks assigned to it, the CPU will load the given task's context and run its code (from the instruction following the last one executed so far, possibly by another CPU). [If there are multiple runnable tasks assigned to one CPU simultaneously, they will be subject to prioritization and time sharing in order to allow them to make some progress over time.]

The special ”idle” task becomes runnable if there are no other runnable tasks assigned to the given CPU and the CPU is then regarded as idle. In other words, in Linux idle CPUs run the code of the ”idle” task called *the idle loop*. That code may cause the processor to be put into one of its idle states, if they are supported, in order to save energy, but if the processor does not support any idle states, or there is not enough time to spend in an idle state before the next wakeup event, or there are strict latency constraints preventing any of the available idle states from being used, the CPU will simply execute more or less useless instructions in a loop until it is assigned a new task to run.

The Idle Loop

The idle loop code takes two major steps in every iteration of it. First, it calls into a code module referred to as the *governor* that belongs to the CPU idle time management subsystem called `CPUIdle` to select an idle state for the CPU to ask the hardware to enter. Second, it invokes another code module from the `CPUIdle` subsystem, called the *driver*, to actually ask the processor hardware to enter the idle state selected by the governor.

The role of the governor is to find an idle state most suitable for the conditions at hand. For this purpose, idle states that the hardware can be asked to enter by logical CPUs are represented in an abstract way independent of the platform or the processor architecture and organized in a one-dimensional (linear) array. That array has to be prepared and supplied by the `CPUIdle` driver matching the platform the kernel is running on at the initialization time. This allows `CPUIdle` governors to be independent of the underlying hardware and to work with any platforms that the Linux kernel can run on.

Each idle state present in that array is characterized by two parameters to be taken into account by the governor, the *target residency* and the (worst-case) *exit latency*. The target residency is the minimum time the hardware must spend in the given state, including the time needed to enter it (which may be substantial), in order to save more energy than it would save by entering one of the shallower idle states instead. [The ”depth” of an idle state roughly corresponds to the power drawn by the processor in that state.] The exit latency, in turn, is the maximum time it will take a CPU asking the processor hardware to enter an idle state to start executing the first instruction after a wakeup from that state. Note that in general the exit latency also must cover the time needed to enter the given state in case the wakeup occurs when the hardware is entering it and it must be entered completely to be exited in an ordered manner.

There are two types of information that can influence the governor's decisions. First of all, the governor knows the time until the closest timer event. That time is known exactly, because the kernel programs timers and it knows exactly when they will trigger, and it is the maximum time the hardware that the given CPU depends on can spend in an idle state, including the time necessary to enter and exit it. However, the CPU may be woken up by a non-timer event at any time (in particular, before the closest timer triggers) and it generally is not known when that may happen. The governor can only see how much time the CPU actually was idle after it has been woken up (that time will be referred to as the *idle duration* from now on) and it can use that information somehow along with the time until the closest timer to estimate the idle duration in future. How the governor uses that information depends on what algorithm

is implemented by it and that is the primary reason for having more than one governor in the `CPUIdle` subsystem.

There are four `CPUIdle` governors available, `menu`, *TEO*, `ladder` and `haltpoll`. Which of them is used by default depends on the configuration of the kernel and in particular on whether or not the scheduler tick can be *stopped by the idle loop*. Available governors can be read from the `available_governors`, and the governor can be changed at runtime. The name of the `CPUIdle` governor currently used by the kernel can be read from the `current_governor_ro` or `current_governor` file under `/sys/devices/system/cpu/cpuidle/` in `sysfs`.

Which `CPUIdle` driver is used, on the other hand, usually depends on the platform the kernel is running on, but there are platforms with more than one matching driver. For example, there are two drivers that can work with the majority of Intel platforms, `intel_idle` and `acpi_idle`, one with hardcoded idle states information and the other able to read that information from the system's ACPI tables, respectively. Still, even in those cases, the driver chosen at the system initialization time cannot be replaced later, so the decision on which one of them to use has to be made early (on Intel platforms the `acpi_idle` driver will be used if `intel_idle` is disabled for some reason or if it does not recognize the processor). The name of the `CPUIdle` driver currently used by the kernel can be read from the `current_driver` file under `/sys/devices/system/cpu/cpuidle/` in `sysfs`.

Idle CPUs and The Scheduler Tick

The scheduler tick is a timer that triggers periodically in order to implement the time sharing strategy of the CPU scheduler. Of course, if there are multiple runnable tasks assigned to one CPU at the same time, the only way to allow them to make reasonable progress in a given time frame is to make them share the available CPU time. Namely, in rough approximation, each task is given a slice of the CPU time to run its code, subject to the scheduling class, prioritization and so on and when that time slice is used up, the CPU should be switched over to running (the code of) another task. The currently running task may not want to give the CPU away voluntarily, however, and the scheduler tick is there to make the switch happen regardless. That is not the only role of the tick, but it is the primary reason for using it.

The scheduler tick is problematic from the CPU idle time management perspective, because it triggers periodically and relatively often (depending on the kernel configuration, the length of the tick period is between 1 ms and 10 ms). Thus, if the tick is allowed to trigger on idle CPUs, it will not make sense for them to ask the hardware to enter idle states with target residencies above the tick period length. Moreover, in that case the idle duration of any CPU will never exceed the tick period length and the energy used for entering and exiting idle states due to the tick wakeups on idle CPUs will be wasted.

Fortunately, it is not really necessary to allow the tick to trigger on idle CPUs, because (by definition) they have no tasks to run except for the special ”idle” one. In other words, from the CPU scheduler perspective, the only user of the CPU time on them is the idle loop. Since the time of an idle CPU need not be shared between multiple runnable tasks, the primary reason for using the tick goes away if the given CPU is idle. Consequently, it is possible to stop the scheduler tick entirely on idle CPUs in principle, even though that may not always be worth the effort.

Whether or not it makes sense to stop the scheduler tick in the idle loop depends on what is expected by the governor. First, if there is another (non-tick) timer due to trigger within the tick range, stopping the tick clearly would be a waste of time, even though the timer hardware may not need to be reprogrammed in that case. Second, if the governor is expecting a non-timer

wakeup within the tick range, stopping the tick is not necessary and it may even be harmful. Namely, in that case the governor will select an idle state with the target residency within the time until the expected wakeup, so that state is going to be relatively shallow. The governor really cannot select a deep idle state then, as that would contradict its own expectation of a wakeup in short order. Now, if the wakeup really occurs shortly, stopping the tick would be a waste of time and in this case the timer hardware would need to be reprogrammed, which is expensive. On the other hand, if the tick is stopped and the wakeup does not occur any time soon, the hardware may spend indefinite amount of time in the shallow idle state selected by the governor, which will be a waste of energy. Hence, if the governor is expecting a wakeup of any kind within the tick range, it is better to allow the tick trigger. Otherwise, however, the governor will select a relatively deep idle state, so the tick should be stopped so that it does not wake up the CPU too early.

In any case, the governor knows what it is expecting and the decision on whether or not to stop the scheduler tick belongs to it. Still, if the tick has been stopped already (in one of the previous iterations of the loop), it is better to leave it as is and the governor needs to take that into account.

The kernel can be configured to disable stopping the scheduler tick in the idle loop altogether. That can be done through the build-time configuration of it (by unsetting the `CONFIG_NO_HZ_IDLE` configuration option) or by passing `nohz=off` to it in the command line. In both cases, as the stopping of the scheduler tick is disabled, the governor's decisions regarding it are simply ignored by the idle loop code and the tick is never stopped.

The systems that run kernels configured to allow the scheduler tick to be stopped on idle CPUs are referred to as *tickless* systems and they are generally regarded as more energy-efficient than the systems running kernels in which the tick cannot be stopped. If the given system is tickless, it will use the `menu` governor by default and if it is not tickless, the default `CPUIdle` governor on it will be `ladder`.

The `menu` Governor

The `menu` governor is the default `CPUIdle` governor for tickless systems. It is quite complex, but the basic principle of its design is straightforward. Namely, when invoked to select an idle state for a CPU (i.e. an idle state that the CPU will ask the processor hardware to enter), it attempts to predict the idle duration and uses the predicted value for idle state selection.

It first obtains the time until the closest timer event with the assumption that the scheduler tick will be stopped. That time, referred to as the *sleep length* in what follows, is the upper bound on the time before the next CPU wakeup. It is used to determine the sleep length range, which in turn is needed to get the sleep length correction factor.

The `menu` governor maintains two arrays of sleep length correction factors. One of them is used when tasks previously running on the given CPU are waiting for some I/O operations to complete and the other one is used when that is not the case. Each array contains several correction factor values that correspond to different sleep length ranges organized so that each range represented in the array is approximately 10 times wider than the previous one.

The correction factor for the given sleep length range (determined before selecting the idle state for the CPU) is updated after the CPU has been woken up and the closer the sleep length is to the observed idle duration, the closer to 1 the correction factor becomes (it must fall between 0 and 1 inclusive). The sleep length is multiplied by the correction factor for the range that it falls into to obtain the first approximation of the predicted idle duration.

Next, the governor uses a simple pattern recognition algorithm to refine its idle duration prediction. Namely, it saves the last 8 observed idle duration values and, when predicting the idle duration next time, it computes the average and variance of them. If the variance is small (smaller than 400 square milliseconds) or it is small relative to the average (the average is greater that 6 times the standard deviation), the average is regarded as the ”typical interval” value. Otherwise, the longest of the saved observed idle duration values is discarded and the computation is repeated for the remaining ones. Again, if the variance of them is small (in the above sense), the average is taken as the ”typical interval” value and so on, until either the ”typical interval” is determined or too many data points are disregarded, in which case the ”typical interval” is assumed to equal ”infinity” (the maximum unsigned integer value). The ”typical interval” computed this way is compared with the sleep length multiplied by the correction factor and the minimum of the two is taken as the predicted idle duration.

Then, the governor computes an extra latency limit to help ”interactive” workloads. It uses the observation that if the exit latency of the selected idle state is comparable with the predicted idle duration, the total time spent in that state probably will be very short and the amount of energy to save by entering it will be relatively small, so likely it is better to avoid the overhead related to entering that state and exiting it. Thus selecting a shallower state is likely to be a better option then. The first approximation of the extra latency limit is the predicted idle duration itself which additionally is divided by a value depending on the number of tasks that previously ran on the given CPU and now they are waiting for I/O operations to complete. The result of that division is compared with the latency limit coming from the power management quality of service, or *PM QoS*, framework and the minimum of the two is taken as the limit for the idle states' exit latency.

Now, the governor is ready to walk the list of idle states and choose one of them. For this purpose, it compares the target residency of each state with the predicted idle duration and the exit latency of it with the computed latency limit. It selects the state with the target residency closest to the predicted idle duration, but still below it, and exit latency that does not exceed the limit.

In the final step the governor may still need to refine the idle state selection if it has not decided to *stop the scheduler tick*. That happens if the idle duration predicted by it is less than the tick period and the tick has not been stopped already (in a previous iteration of the idle loop). Then, the sleep length used in the previous computations may not reflect the real time until the closest timer event and if it really is greater than that time, the governor may need to select a shallower state with a suitable target residency.

The Timer Events Oriented (TEO) Governor

The timer events oriented (TEO) governor is an alternative `CPUIdle` governor for tickless systems. It follows the same basic strategy as the `menu` *one*: it always tries to find the deepest idle state suitable for the given conditions. However, it applies a different approach to that problem.

The idea of this governor is based on the observation that on many systems timer events are two or more orders of magnitude more frequent than any other interrupts, so they are likely to be the most significant cause of CPU wakeups from idle states. Moreover, information about what happened in the (relatively recent) past can be used to estimate whether or not the deepest idle state with target residency within the (known) time till the closest timer event, referred to as the sleep length, is likely to be suitable for the upcoming CPU idle period and, if not, then which of the shallower idle states to choose instead of it.

Of course, non-timer wakeup sources are more important in some use cases which can be covered by taking a few most recent idle time intervals of the CPU into account. However, even in that context it is not necessary to consider idle duration values greater than the sleep length, because the closest timer will ultimately wake up the CPU anyway unless it is woken up earlier.

Thus this governor estimates whether or not the prospective idle duration of a CPU is likely to be significantly shorter than the sleep length and selects an idle state for it accordingly.

The computations carried out by this governor are based on using bins whose boundaries are aligned with the target residency parameter values of the CPU idle states provided by the `CPUIdle` driver in the ascending order. That is, the first bin spans from 0 up to, but not including, the target residency of the second idle state (idle state 1), the second bin spans from the target residency of idle state 1 up to, but not including, the target residency of idle state 2, the third bin spans from the target residency of idle state 2 up to, but not including, the target residency of idle state 3 and so on. The last bin spans from the target residency of the deepest idle state supplied by the driver to infinity.

Two metrics called ”hits” and ”intercepts” are associated with each bin. They are updated every time before selecting an idle state for the given CPU in accordance with what happened last time.

The ”hits” metric reflects the relative frequency of situations in which the sleep length and the idle duration measured after CPU wakeup fall into the same bin (that is, the CPU appears to wake up ”on time” relative to the sleep length). In turn, the ”intercepts” metric reflects the relative frequency of situations in which the measured idle duration is so much shorter than the sleep length that the bin it falls into corresponds to an idle state shallower than the one whose bin is fallen into by the sleep length (these situations are referred to as ”intercepts” below).

In addition to the metrics described above, the governor counts recent intercepts (that is, intercepts that have occurred during the last `NR_RECENT` invocations of it for the given CPU) for each bin.

In order to select an idle state for a CPU, the governor takes the following steps (modulo the possible latency constraint that must be taken into account too):

1. Find the deepest CPU idle state whose target residency does not exceed the current sleep length (the candidate idle state) and compute 3 sums as follows:
   - The sum of the ”hits” and ”intercepts” metrics for the candidate state and all of the deeper idle states (it represents the cases in which the CPU was idle long enough to avoid being intercepted if the sleep length had been equal to the current one).
   - The sum of the ”intercepts” metrics for all of the idle states shallower than the candidate one (it represents the cases in which the CPU was not idle long enough to avoid being intercepted if the sleep length had been equal to the current one).
   - The sum of the numbers of recent intercepts for all of the idle states shallower than the candidate one.
2. If the second sum is greater than the first one or the third sum is greater than `NR_RECENT` / 2, the CPU is likely to wake up early, so look for an alternative idle state to select.
   - Traverse the idle states shallower than the candidate one in the descending order.
   - For each of them compute the sum of the ”intercepts” metrics and the sum of the numbers of recent intercepts over all of the idle states between it and the candidate one (including the former and excluding the latter).

- If each of these sums that needs to be taken into account (because the check related to it has indicated that the CPU is likely to wake up early) is greater than a half of the corresponding sum computed in step 1 (which means that the target residency of the state in question had not exceeded the idle duration in over a half of the relevant cases), select the given idle state instead of the candidate one.

3. By default, select the candidate state.

Util-awareness mechanism:

The idea behind the util-awareness extension is that there are two distinct scenarios for the CPU which should result in two different approaches to idle state selection - utilized and not utilized.

In this case, 'utilized' means that the average runqueue util of the CPU is above a certain threshold.

When the CPU is utilized while going into idle, more likely than not it will be woken up to do more work soon and so a shallower idle state should be selected to minimise latency and maximise performance. When the CPU is not being utilized, the usual metrics-based approach to selecting the deepest available idle state should be preferred to take advantage of the power saving.

In order to achieve this, the governor uses a utilization threshold. The threshold is computed per-CPU as a percentage of the CPU's capacity by bit shifting the capacity value. Based on testing, the shift of 6 (~1.56%) seems to be getting the best results.

Before selecting the next idle state, the governor compares the current CPU util to the precomputed util threshold. If it's below, it defaults to the TEO metrics mechanism. If it's above, the closest shallower idle state will be selected instead, as long as is not a polling state.

Representation of Idle States

For the CPU idle time management purposes all of the physical idle states supported by the processor have to be represented as a one-dimensional array of `struct cpuidle_state` objects each allowing an individual (logical) CPU to ask the processor hardware to enter an idle state of certain properties. If there is a hierarchy of units in the processor, one `struct cpuidle_state` object can cover a combination of idle states supported by the units at different levels of the hierarchy. In that case, the *target residency and exit latency parameters of it*, must reflect the properties of the idle state at the deepest level (i.e. the idle state of the unit containing all of the other units).

For example, take a processor with two cores in a larger unit referred to as a ”module” and suppose that asking the hardware to enter a specific idle state (say ”X”) at the ”core” level by one core will trigger the module to try to enter a specific idle state of its own (say ”MX”) if the other core is in idle state ”X” already. In other words, asking for idle state ”X” at the ”core” level gives the hardware a license to go as deep as to idle state ”MX” at the ”module” level, but there is no guarantee that this is going to happen (the core asking for idle state ”X” may just end up in that state by itself instead). Then, the target residency of the `struct cpuidle_state` object representing idle state ”X” must reflect the minimum time to spend in idle state ”MX” of the module (including the time needed to enter it), because that is the minimum time the CPU needs to be idle to save any energy in case the hardware enters that state. Analogously, the exit latency parameter of that object must cover the exit time of idle state ”MX” of the module (and usually its entry time too), because that is the maximum delay between a wakeup signal and the time the CPU will start to execute the first new instruction (assuming that both cores in the

module will always be ready to execute instructions as soon as the module becomes operational as a whole).

There are processors without direct coordination between different levels of the hierarchy of units inside them, however. In those cases asking for an idle state at the "core" level does not automatically affect the "module" level, for example, in any way and the `CPUIdle` driver is responsible for the entire handling of the hierarchy. Then, the definition of the idle state objects is entirely up to the driver, but still the physical properties of the idle state that the processor hardware finally goes into must always follow the parameters used by the governor for idle state selection (for instance, the actual exit latency of that idle state must not exceed the exit latency parameter of the idle state object selected by the governor).

In addition to the target residency and exit latency idle state parameters discussed above, the objects representing idle states each contain a few other parameters describing the idle state and a pointer to the function to run in order to ask the hardware to enter that state. Also, for each `struct cpuidle_state` object, there is a corresponding `struct cpuidle_state_usage` one containing usage statistics of the given idle state. That information is exposed by the kernel via `sysfs`.

For each CPU in the system, there is a `/sys/devices/system/cpu/cpu<N>/cpuidle/` directory in `sysfs`, where the number `<N>` is assigned to the given CPU at the initialization time. That directory contains a set of subdirectories called `state0`, `state1` and so on, up to the number of idle state objects defined for the given CPU minus one. Each of these directories corresponds to one idle state object and the larger the number in its name, the deeper the (effective) idle state represented by it. Each of them contains a number of files (attributes) representing the properties of the idle state object corresponding to it, as follows:

above
: Total number of times this idle state had been asked for, but the observed idle duration was certainly too short to match its target residency.

below
: Total number of times this idle state had been asked for, but certainly a deeper idle state would have been a better match for the observed idle duration.

desc
: Description of the idle state.

disable
: Whether or not this idle state is disabled.

default_status
: The default status of this state, "enabled" or "disabled".

latency
: Exit latency of the idle state in microseconds.

name
: Name of the idle state.

power
: Power drawn by hardware in this idle state in milliwatts (if specified, 0 otherwise).

residency
: Target residency of the idle state in microseconds.

time

Total time spent in this idle state by the given CPU (as measured by the kernel) in microseconds.

usage
Total number of times the hardware has been asked by the given CPU to enter this idle state.

rejected
Total number of times a request to enter this idle state on the given CPU was rejected.

The `desc` and `name` files both contain strings. The difference between them is that the name is expected to be more concise, while the description may be longer and it may contain white space or special characters. The other files listed above contain integer numbers.

The `disable` attribute is the only writeable one. If it contains 1, the given idle state is disabled for this particular CPU, which means that the governor will never select it for this particular CPU and the `CPUIdle` driver will never ask the hardware to enter it for that CPU as a result. However, disabling an idle state for one CPU does not prevent it from being asked for by the other CPUs, so it must be disabled for all of them in order to never be asked for by any of them. [Note that, due to the way the `ladder` governor is implemented, disabling an idle state prevents that governor from selecting any idle states deeper than the disabled one too.]

If the `disable` attribute contains 0, the given idle state is enabled for this particular CPU, but it still may be disabled for some or all of the other CPUs in the system at the same time. Writing 1 to it causes the idle state to be disabled for this particular CPU and writing 0 to it allows the governor to take it into consideration for the given CPU and the driver to ask for it, unless that state was disabled globally in the driver (in which case it cannot be used at all).

The `power` attribute is not defined very well, especially for idle state objects representing combinations of idle states at different levels of the hierarchy of units in the processor, and it generally is hard to obtain idle state power numbers for complex hardware, so `power` often contains 0 (not available) and if it contains a nonzero number, that number may not be very accurate and it should not be relied on for anything meaningful.

The number in the `time` file generally may be greater than the total time really spent by the given CPU in the given idle state, because it is measured by the kernel and it may not cover the cases in which the hardware refused to enter this idle state and entered a shallower one instead of it (or even it did not enter any idle state at all). The kernel can only measure the time span between asking the hardware to enter an idle state and the subsequent wakeup of the CPU and it cannot say what really happened in the meantime at the hardware level. Moreover, if the idle state object in question represents a combination of idle states at different levels of the hierarchy of units in the processor, the kernel can never say how deep the hardware went down the hierarchy in any particular case. For these reasons, the only reliable way to find out how much time has been spent by the hardware in different idle states supported by it is to use idle state residency counters in the hardware, if available.

Generally, an interrupt received when trying to enter an idle state causes the idle state entry request to be rejected, in which case the `CPUIdle` driver may return an error code to indicate that this was the case. The `usage` and `rejected` files report the number of times the given idle state was entered successfully or rejected, respectively.

Power Management Quality of Service for CPUs

The power management quality of service (PM QoS) framework in the Linux kernel allows kernel code and user space processes to set constraints on various energy-efficiency features of the kernel to prevent performance from dropping below a required level.

CPU idle time management can be affected by PM QoS in two ways, through the global CPU latency limit and through the resume latency constraints for individual CPUs. Kernel code (e.g. device drivers) can set both of them with the help of special internal interfaces provided by the PM QoS framework. User space can modify the former by opening the `cpu_dma_latency` special device file under `/dev/` and writing a binary value (interpreted as a signed 32-bit integer) to it. In turn, the resume latency constraint for a CPU can be modified from user space by writing a string (representing a signed 32-bit integer) to the `power/pm_qos_resume_latency_us` file under `/sys/devices/system/cpu/cpu<N>/` in `sysfs`, where the CPU number `<N>` is allocated at the system initialization time. Negative values will be rejected in both cases and, also in both cases, the written integer number will be interpreted as a requested PM QoS constraint in microseconds.

The requested value is not automatically applied as a new constraint, however, as it may be less restrictive (greater in this particular case) than another constraint previously requested by someone else. For this reason, the PM QoS framework maintains a list of requests that have been made so far for the global CPU latency limit and for each individual CPU, aggregates them and applies the effective (minimum in this particular case) value as the new constraint.

In fact, opening the `cpu_dma_latency` special device file causes a new PM QoS request to be created and added to a global priority list of CPU latency limit requests and the file descriptor coming from the ”open” operation represents that request. If that file descriptor is then used for writing, the number written to it will be associated with the PM QoS request represented by it as a new requested limit value. Next, the priority list mechanism will be used to determine the new effective value of the entire list of requests and that effective value will be set as a new CPU latency limit. Thus requesting a new limit value will only change the real limit if the effective ”list” value is affected by it, which is the case if it is the minimum of the requested values in the list.

The process holding a file descriptor obtained by opening the `cpu_dma_latency` special device file controls the PM QoS request associated with that file descriptor, but it controls this particular PM QoS request only.

Closing the `cpu_dma_latency` special device file or, more precisely, the file descriptor obtained while opening it, causes the PM QoS request associated with that file descriptor to be removed from the global priority list of CPU latency limit requests and destroyed. If that happens, the priority list mechanism will be used again, to determine the new effective value for the whole list and that value will become the new limit.

In turn, for each CPU there is one resume latency PM QoS request associated with the `power/pm_qos_resume_latency_us` file under `/sys/devices/system/cpu/cpu<N>/` in `sysfs` and writing to it causes this single PM QoS request to be updated regardless of which user space process does that. In other words, this PM QoS request is shared by the entire user space, so access to the file associated with it needs to be arbitrated to avoid confusion. [Arguably, the only legitimate use of this mechanism in practice is to pin a process to the CPU in question and let it use the `sysfs` interface to control the resume latency constraint for it.] It is still only a request, however. It is an entry in a priority list used to determine the effective value to be set as the resume latency constraint for the CPU in question every time the list of requests is updated this way or another (there may be other requests coming from kernel code in that list).

CPU idle time governors are expected to regard the minimum of the global (effective) CPU latency limit and the effective resume latency constraint for the given CPU as the upper limit for the exit latency of the idle states that they are allowed to select for that CPU. They should never select any idle states with exit latency beyond that limit.

Idle States Control Via Kernel Command Line

In addition to the `sysfs` interface allowing individual idle states to be *disabled for individual CPUs*, there are kernel command line parameters affecting CPU idle time management.

The `cpuidle.off=1` kernel command line option can be used to disable the CPU idle time management entirely. It does not prevent the idle loop from running on idle CPUs, but it prevents the CPU idle time governors and drivers from being invoked. If it is added to the kernel command line, the idle loop will ask the hardware to enter idle states on idle CPUs via the CPU architecture support code that is expected to provide a default mechanism for this purpose. That default mechanism usually is the least common denominator for all of the processors implementing the architecture (i.e. CPU instruction set) in question, however, so it is rather crude and not very energy-efficient. For this reason, it is not recommended for production use.

The `cpuidle.governor=` kernel command line switch allows the `CPUIdle` governor to use to be specified. It has to be appended with a string matching the name of an available governor (e.g. `cpuidle.governor=menu`) and that governor will be used instead of the default one. It is possible to force the `menu` governor to be used on the systems that use the `ladder` governor by default this way, for example.

The other kernel command line parameters controlling CPU idle time management described below are only relevant for the *x86* architecture and references to `intel_idle` affect Intel processors only.

The *x86* architecture support code recognizes three kernel command line options related to CPU idle time management: `idle=poll`, `idle=halt`, and `idle=nomwait`. The first two of them disable the `acpi_idle` and `intel_idle` drivers altogether, which effectively causes the entire `CPUIdle` subsystem to be disabled and makes the idle loop invoke the architecture support code to deal with idle CPUs. How it does that depends on which of the two parameters is added to the kernel command line. In the `idle=halt` case, the architecture support code will use the `HLT` instruction of the CPUs (which, as a rule, suspends the execution of the program and causes the hardware to attempt to enter the shallowest available idle state) for this purpose, and if `idle=poll` is used, idle CPUs will execute a more or less "lightweight" sequence of instructions in a tight loop. [Note that using `idle=poll` is somewhat drastic in many cases, as preventing idle CPUs from saving almost any energy at all may not be the only effect of it. For example, on Intel hardware it effectively prevents CPUs from using P-states (see *CPU Performance Scaling*) that require any number of CPUs in a package to be idle, so it very well may hurt single-thread computations performance as well as energy-efficiency. Thus using it for performance reasons may not be a good idea at all.]

The `idle=nomwait` option prevents the use of `MWAIT` instruction of the CPU to enter idle states. When this option is used, the `acpi_idle` driver will use the `HLT` instruction instead of `MWAIT`. On systems running Intel processors, this option disables the `intel_idle` driver and forces the use of the `acpi_idle` driver instead. Note that in either case, `acpi_idle` driver will function only if all the information needed by it is in the system's ACPI tables.

In addition to the architecture-level kernel command line options affecting CPU idle time management, there are parameters affecting individual `CPUIdle` drivers that can be passed to them

via the kernel command line. Specifically, the `intel_idle.max_cstate=<n>` and `processor.max_cstate=<n>` parameters, where `<n>` is an idle state index also used in the name of the given state's directory in `sysfs` (see *Representation of Idle States*), causes the `intel_idle` and `acpi_idle` drivers, respectively, to discard all of the idle states deeper than idle state `<n>`. In that case, they will never ask for any of those idle states or expose them to the governor. [The behavior of the two drivers is different for `<n>` equal to `0`. Adding `intel_idle.max_cstate=0` to the kernel command line disables the `intel_idle` driver and allows `acpi_idle` to be used, whereas `processor.max_cstate=0` is equivalent to `processor.max_cstate=1`. Also, the `acpi_idle` driver is part of the `processor` kernel module that can be loaded separately and `max_cstate=<n>` can be passed to it as a module parameter when it is loaded.]

67.3.2 `intel_idle` CPU Idle Time Management Driver

Copyright
© 2020 Intel Corporation

Author
Rafael J. Wysocki <rafael.j.wysocki@intel.com>

General Information

`intel_idle` is a part of the *CPU idle time management subsystem* in the Linux kernel (`CPUIdle`). It is the default CPU idle time management driver for the Nehalem and later generations of Intel processors, but the level of support for a particular processor model in it depends on whether or not it recognizes that processor model and may also depend on information coming from the platform firmware. [To understand `intel_idle` it is necessary to know how `CPUIdle` works in general, so this is the time to get familiar with *CPU Idle Time Management* if you have not done that yet.]

`intel_idle` uses the `MWAIT` instruction to inform the processor that the logical CPU executing it is idle and so it may be possible to put some of the processor's functional blocks into low-power states. That instruction takes two arguments (passed in the `EAX` and `ECX` registers of the target CPU), the first of which, referred to as a *hint*, can be used by the processor to determine what can be done (for details refer to Intel Software Developer's Manual[1]). Accordingly, `intel_idle` refuses to work with processors in which the support for the `MWAIT` instruction has been disabled (for example, via the platform firmware configuration menu) or which do not support that instruction at all.

`intel_idle` is not modular, so it cannot be unloaded, which means that the only way to pass early-configuration-time parameters to it is via the kernel command line.

[1] *Intel® 64 and IA-32 Architectures Software Developer's Manual Volume 2B*, https://www.intel.com/content/www/us/en/architecture-and-technology/64-ia-32-architectures-software-developer-vol-2b-manual.html

Enumeration of Idle States

Each MWAIT hint value is interpreted by the processor as a license to reconfigure itself in a certain way in order to save energy. The processor configurations (with reduced power draw) resulting from that are referred to as C-states (in the ACPI terminology) or idle states. The list of meaningful MWAIT hint values and idle states (i.e. low-power configurations of the processor) corresponding to them depends on the processor model and it may also depend on the configuration of the platform.

In order to create a list of available idle states required by the CPUIdle subsystem (see *Representation of Idle States* in *CPU Idle Time Management*), intel_idle can use two sources of information: static tables of idle states for different processor models included in the driver itself and the ACPI tables of the system. The former are always used if the processor model at hand is recognized by intel_idle and the latter are used if that is required for the given processor model (which is the case for all server processor models recognized by intel_idle) or if the processor model is not recognized. [There is a module parameter that can be used to make the driver use the ACPI tables with any processor model recognized by it; see *below*.]

If the ACPI tables are going to be used for building the list of available idle states, intel_idle first looks for a _CST object under one of the ACPI objects corresponding to the CPUs in the system (refer to the ACPI specification[2] for the description of _CST and its output package). Because the CPUIdle subsystem expects that the list of idle states supplied by the driver will be suitable for all of the CPUs handled by it and intel_idle is registered as the CPUIdle driver for all of the CPUs in the system, the driver looks for the first _CST object returning at least one valid idle state description and such that all of the idle states included in its return package are of the FFH (Functional Fixed Hardware) type, which means that the MWAIT instruction is expected to be used to tell the processor that it can enter one of them. The return package of that _CST is then assumed to be applicable to all of the other CPUs in the system and the idle state descriptions extracted from it are stored in a preliminary list of idle states coming from the ACPI tables. [This step is skipped if intel_idle is configured to ignore the ACPI tables; see *below*.]

Next, the first (index 0) entry in the list of available idle states is initialized to represent a ”polling idle state” (a pseudo-idle state in which the target CPU continuously fetches and executes instructions), and the subsequent (real) idle state entries are populated as follows.

If the processor model at hand is recognized by intel_idle, there is a (static) table of idle state descriptions for it in the driver. In that case, the ”internal” table is the primary source of information on idle states and the information from it is copied to the final list of available idle states. If using the ACPI tables for the enumeration of idle states is not required (depending on the processor model), all of the listed idle state are enabled by default (so all of them will be taken into consideration by CPUIdle governors during CPU idle state selection). Otherwise, some of the listed idle states may not be enabled by default if there are no matching entries in the preliminary list of idle states coming from the ACPI tables. In that case user space still can enable them later (on a per-CPU basis) with the help of the disable idle state attribute in sysfs (see *Representation of Idle States* in *CPU Idle Time Management*). This basically means that the idle states ”known” to the driver may not be enabled by default if they have not been exposed by the platform firmware (through the ACPI tables).

If the given processor model is not recognized by intel_idle, but it supports MWAIT, the preliminary list of idle states coming from the ACPI tables is used for building the final list that will be supplied to the CPUIdle core during driver registration. For each idle state in that list, the

[2] *Advanced Configuration and Power Interface (ACPI) Specification*, https://uefi.org/specifications

description, MWAIT hint and exit latency are copied to the corresponding entry in the final list of idle states. The name of the idle state represented by it (to be returned by the name idle state attribute in sysfs) is "CX_ACPI", where X is the index of that idle state in the final list (note that the minimum value of X is 1, because 0 is reserved for the "polling" state), and its target residency is based on the exit latency value. Specifically, for C1-type idle states the exit latency value is also used as the target residency (for compatibility with the majority of the "internal" tables of idle states for various processor models recognized by intel_idle) and for the other idle state types (C2 and C3) the target residency value is 3 times the exit latency (again, that is because it reflects the target residency to exit latency ratio in the majority of cases for the processor models recognized by intel_idle). All of the idle states in the final list are enabled by default in this case.

Initialization

The initialization of intel_idle starts with checking if the kernel command line options forbid the use of the MWAIT instruction. If that is the case, an error code is returned right away.

The next step is to check whether or not the processor model is known to the driver, which determines the idle states enumeration method (see *above*), and whether or not the processor supports MWAIT (the initialization fails if that is not the case). Then, the MWAIT support in the processor is enumerated through CPUID and the driver initialization fails if the level of support is not as expected (for example, if the total number of MWAIT substates returned is 0).

Next, if the driver is not configured to ignore the ACPI tables (see *below*), the idle states information provided by the platform firmware is extracted from them.

Then, CPUIdle device objects are allocated for all CPUs and the list of available idle states is created as explained *above*.

Finally, intel_idle is registered with the help of cpuidle_register_driver() as the CPUIdle driver for all CPUs in the system and a CPU online callback for configuring individual CPUs is registered via cpuhp_setup_state(), which (among other things) causes the callback routine to be invoked for all of the CPUs present in the system at that time (each CPU executes its own instance of the callback routine). That routine registers a CPUIdle device for the CPU running it (which enables the CPUIdle subsystem to operate that CPU) and optionally performs some CPU-specific initialization actions that may be required for the given processor model.

Kernel Command Line Options and Module Parameters

The *x86* architecture support code recognizes three kernel command line options related to CPU idle time management: idle=poll, idle=halt, and idle=nomwait. If any of them is present in the kernel command line, the MWAIT instruction is not allowed to be used, so the initialization of intel_idle will fail.

Apart from that there are five module parameters recognized by intel_idle itself that can be set via the kernel command line (they cannot be updated via sysfs, so that is the only way to change their values).

The max_cstate parameter value is the maximum idle state index in the list of idle states supplied to the CPUIdle core during the registration of the driver. It is also the maximum number of regular (non-polling) idle states that can be used by intel_idle, so the enumeration of idle states is terminated after finding that number of usable idle states (the other idle states that potentially might have been used if max_cstate had been greater are not taken into consideration

at all). Setting max_cstate can prevent intel_idle from exposing idle states that are regarded as "too deep" for some reason to the CPUIdle core, but it does so by making them effectively invisible until the system is shut down and started again which may not always be desirable. In practice, it is only really necessary to do that if the idle states in question cannot be enabled during system startup, because in the working state of the system the CPU power management quality of service (PM QoS) feature can be used to prevent CPUIdle from touching those idle states even if they have been enumerated (see *Power Management Quality of Service for CPUs* in *CPU Idle Time Management*). Setting max_cstate to 0 causes the intel_idle initialization to fail.

The no_acpi and use_acpi module parameters (recognized by intel_idle if the kernel has been configured with ACPI support) can be set to make the driver ignore the system's ACPI tables entirely or use them for all of the recognized processor models, respectively (they both are unset by default and use_acpi has no effect if no_acpi is set).

The value of the states_off module parameter (0 by default) represents a list of idle states to be disabled by default in the form of a bitmask.

Namely, the positions of the bits that are set in the states_off value are the indices of idle states to be disabled by default (as reflected by the names of the corresponding idle state directories in sysfs, state0, state1 ... state<i> ..., where <i> is the index of the given idle state; see *Representation of Idle States* in *CPU Idle Time Management*).

For example, if states_off is equal to 3, the driver will disable idle states 0 and 1 by default, and if it is equal to 8, idle state 3 will be disabled by default and so on (bit positions beyond the maximum idle state index are ignored).

The idle states disabled this way can be enabled (on a per-CPU basis) from user space via sysfs.

The ibrs_off module parameter is a boolean flag (defaults to false). If set, it is used to control if IBRS (Indirect Branch Restricted Speculation) should be turned off when the CPU enters an idle state. This flag does not affect CPUs that use Enhanced IBRS which can remain on with little performance impact.

For some CPUs, IBRS will be selected as mitigation for Spectre v2 and Retbleed security vulnerabilities by default. Leaving the IBRS mode on while idling may have a performance impact on its sibling CPU. The IBRS mode will be turned off by default when the CPU enters into a deep idle state, but not in some shallower ones. Setting the ibrs_off module parameter will force the IBRS mode to off when the CPU is in any one of the available idle states. This may help performance of a sibling CPU at the expense of a slightly higher wakeup latency for the idle CPU.

Core and Package Levels of Idle States

Typically, in a processor supporting the MWAIT instruction there are (at least) two levels of idle states (or C-states). One level, referred to as "core C-states", covers individual cores in the processor, whereas the other level, referred to as "package C-states", covers the entire processor package and it may also involve other components of the system (GPUs, memory controllers, I/O hubs etc.).

Some of the MWAIT hint values allow the processor to use core C-states only (most importantly, that is the case for the MWAIT hint value corresponding to the C1 idle state), but the majority of them give it a license to put the target core (i.e. the core containing the logical CPU executing MWAIT with the given hint value) into a specific core C-state and then (if possible) to enter a specific package C-state at the deeper level. For example, the MWAIT hint value representing

the C3 idle state allows the processor to put the target core into the low-power state referred to as "core C3" (or CC3), which happens if all of the logical CPUs (SMT siblings) in that core have executed MWAIT with the C3 hint value (or with a hint value representing a deeper idle state), and in addition to that (in the majority of cases) it gives the processor a license to put the entire package (possibly including some non-CPU components such as a GPU or a memory controller) into the low-power state referred to as "package C3" (or PC3), which happens if all of the cores have gone into the CC3 state and (possibly) some additional conditions are satisfied (for instance, if the GPU is covered by PC3, it may be required to be in a certain GPU-specific low-power state for PC3 to be reachable).

As a rule, there is no simple way to make the processor use core C-states only if the conditions for entering the corresponding package C-states are met, so the logical CPU executing MWAIT with a hint value that is not core-level only (like for C1) must always assume that this may cause the processor to enter a package C-state. [That is why the exit latency and target residency values corresponding to the majority of MWAIT hint values in the "internal" tables of idle states in intel_idle reflect the properties of package C-states.] If using package C-states is not desirable at all, either *PM QoS* or the max_cstate module parameter of intel_idle described *above* must be used to restrict the range of permissible idle states to the ones with core-level only MWAIT hint values (like C1).

References

67.3.3 CPU Performance Scaling

Copyright
© 2017 Intel Corporation

Author
Rafael J. Wysocki <rafael.j.wysocki@intel.com>

The Concept of CPU Performance Scaling

The majority of modern processors are capable of operating in a number of different clock frequency and voltage configurations, often referred to as Operating Performance Points or P-states (in ACPI terminology). As a rule, the higher the clock frequency and the higher the voltage, the more instructions can be retired by the CPU over a unit of time, but also the higher the clock frequency and the higher the voltage, the more energy is consumed over a unit of time (or the more power is drawn) by the CPU in the given P-state. Therefore there is a natural tradeoff between the CPU capacity (the number of instructions that can be executed over a unit of time) and the power drawn by the CPU.

In some situations it is desirable or even necessary to run the program as fast as possible and then there is no reason to use any P-states different from the highest one (i.e. the highest-performance frequency/voltage configuration available). In some other cases, however, it may not be necessary to execute instructions so quickly and maintaining the highest available CPU capacity for a relatively long time without utilizing it entirely may be regarded as wasteful. It also may not be physically possible to maintain maximum CPU capacity for too long for thermal or power supply capacity reasons or similar. To cover those cases, there are hardware interfaces allowing CPUs to be switched between different frequency/voltage configurations or (in the ACPI terminology) to be put into different P-states.

Typically, they are used along with algorithms to estimate the required CPU capacity, so as to decide which P-states to put the CPUs into. Of course, since the utilization of the system generally changes over time, that has to be done repeatedly on a regular basis. The activity by which this happens is referred to as CPU performance scaling or CPU frequency scaling (because it involves adjusting the CPU clock frequency).

CPU Performance Scaling in Linux

The Linux kernel supports CPU performance scaling by means of the `CPUFreq` (CPU Frequency scaling) subsystem that consists of three layers of code: the core, scaling governors and scaling drivers.

The `CPUFreq` core provides the common code infrastructure and user space interfaces for all platforms that support CPU performance scaling. It defines the basic framework in which the other components operate.

Scaling governors implement algorithms to estimate the required CPU capacity. As a rule, each governor implements one, possibly parametrized, scaling algorithm.

Scaling drivers talk to the hardware. They provide scaling governors with information on the available P-states (or P-state ranges in some cases) and access platform-specific hardware interfaces to change CPU P-states as requested by scaling governors.

In principle, all available scaling governors can be used with every scaling driver. That design is based on the observation that the information used by performance scaling algorithms for P-state selection can be represented in a platform-independent form in the majority of cases, so it should be possible to use the same performance scaling algorithm implemented in exactly the same way regardless of which scaling driver is used. Consequently, the same set of scaling governors should be suitable for every supported platform.

However, that observation may not hold for performance scaling algorithms based on information provided by the hardware itself, for example through feedback registers, as that information is typically specific to the hardware interface it comes from and may not be easily represented in an abstract, platform-independent way. For this reason, `CPUFreq` allows scaling drivers to bypass the governor layer and implement their own performance scaling algorithms. That is done by the *intel_pstate* scaling driver.

`CPUFreq` Policy Objects

In some cases the hardware interface for P-state control is shared by multiple CPUs. That is, for example, the same register (or set of registers) is used to control the P-state of multiple CPUs at the same time and writing to it affects all of those CPUs simultaneously.

Sets of CPUs sharing hardware P-state control interfaces are represented by `CPUFreq` as struct cpufreq_policy objects. For consistency, struct cpufreq_policy is also used when there is only one CPU in the given set.

The `CPUFreq` core maintains a pointer to a struct cpufreq_policy object for every CPU in the system, including CPUs that are currently offline. If multiple CPUs share the same hardware P-state control interface, all of the pointers corresponding to them point to the same struct cpufreq_policy object.

`CPUFreq` uses struct cpufreq_policy as its basic data type and the design of its user space interface is based on the policy concept.

CPU Initialization

First of all, a scaling driver has to be registered for `CPUFreq` to work. It is only possible to register one scaling driver at a time, so the scaling driver is expected to be able to handle all CPUs in the system.

The scaling driver may be registered before or after CPU registration. If CPUs are registered earlier, the driver core invokes the `CPUFreq` core to take a note of all of the already registered CPUs during the registration of the scaling driver. In turn, if any CPUs are registered after the registration of the scaling driver, the `CPUFreq` core will be invoked to take note of them at their registration time.

In any case, the `CPUFreq` core is invoked to take note of any logical CPU it has not seen so far as soon as it is ready to handle that CPU. [Note that the logical CPU may be a physical single-core processor, or a single core in a multicore processor, or a hardware thread in a physical processor or processor core. In what follows ”CPU” always means ”logical CPU” unless explicitly stated otherwise and the word ”processor” is used to refer to the physical part possibly including multiple logical CPUs.]

Once invoked, the `CPUFreq` core checks if the policy pointer is already set for the given CPU and if so, it skips the policy object creation. Otherwise, a new policy object is created and initialized, which involves the creation of a new policy directory in `sysfs`, and the policy pointer corresponding to the given CPU is set to the new policy object's address in memory.

Next, the scaling driver's `->init()` callback is invoked with the policy pointer of the new CPU passed to it as the argument. That callback is expected to initialize the performance scaling hardware interface for the given CPU (or, more precisely, for the set of CPUs sharing the hardware interface it belongs to, represented by its policy object) and, if the policy object it has been called for is new, to set parameters of the policy, like the minimum and maximum frequencies supported by the hardware, the table of available frequencies (if the set of supported P-states is not a continuous range), and the mask of CPUs that belong to the same policy (including both online and offline CPUs). That mask is then used by the core to populate the policy pointers for all of the CPUs in it.

The next major initialization step for a new policy object is to attach a scaling governor to it (to begin with, that is the default scaling governor determined by the kernel command line or configuration, but it may be changed later via `sysfs`). First, a pointer to the new policy object is passed to the governor's `->init()` callback which is expected to initialize all of the data structures necessary to handle the given policy and, possibly, to add a governor `sysfs` interface to it. Next, the governor is started by invoking its `->start()` callback.

That callback is expected to register per-CPU utilization update callbacks for all of the online CPUs belonging to the given policy with the CPU scheduler. The utilization update callbacks will be invoked by the CPU scheduler on important events, like task enqueue and dequeue, on every iteration of the scheduler tick or generally whenever the CPU utilization may change (from the scheduler's perspective). They are expected to carry out computations needed to determine the P-state to use for the given policy going forward and to invoke the scaling driver to make changes to the hardware in accordance with the P-state selection. The scaling driver may be invoked directly from scheduler context or asynchronously, via a kernel thread or workqueue, depending on the configuration and capabilities of the scaling driver and the governor.

Similar steps are taken for policy objects that are not new, but were ”inactive” previously, meaning that all of the CPUs belonging to them were offline. The only practical difference in that case is that the `CPUFreq` core will attempt to use the scaling governor previously used with the policy that became ”inactive” (and is re-initialized now) instead of the default governor.

In turn, if a previously offline CPU is being brought back online, but some other CPUs sharing the policy object with it are online already, there is no need to re-initialize the policy object at all. In that case, it only is necessary to restart the scaling governor so that it can take the new online CPU into account. That is achieved by invoking the governor's `->stop` and `->start()` callbacks, in this order, for the entire policy.

As mentioned before, the *intel_pstate* scaling driver bypasses the scaling governor layer of `CPUFreq` and provides its own P-state selection algorithms. Consequently, if *intel_pstate* is used, scaling governors are not attached to new policy objects. Instead, the driver's `->setpolicy()` callback is invoked to register per-CPU utilization update callbacks for each policy. These callbacks are invoked by the CPU scheduler in the same way as for scaling governors, but in the *intel_pstate* case they both determine the P-state to use and change the hardware configuration accordingly in one go from scheduler context.

The policy objects created during CPU initialization and other data structures associated with them are torn down when the scaling driver is unregistered (which happens when the kernel module containing it is unloaded, for example) or when the last CPU belonging to the given policy in unregistered.

Policy Interface in `sysfs`

During the initialization of the kernel, the `CPUFreq` core creates a `sysfs` directory (kobject) called `cpufreq` under `/sys/devices/system/cpu/`.

That directory contains a `policyX` subdirectory (where `X` represents an integer number) for every policy object maintained by the `CPUFreq` core. Each `policyX` directory is pointed to by `cpufreq` symbolic links under `/sys/devices/system/cpu/cpuY/` (where `Y` represents an integer that may be different from the one represented by `X`) for all of the CPUs associated with (or belonging to) the given policy. The `policyX` directories in `/sys/devices/system/cpu/cpufreq` each contain policy-specific attributes (files) to control `CPUFreq` behavior for the corresponding policy objects (that is, for all of the CPUs associated with them).

Some of those attributes are generic. They are created by the `CPUFreq` core and their behavior generally does not depend on what scaling driver is in use and what scaling governor is attached to the given policy. Some scaling drivers also add driver-specific attributes to the policy directories in `sysfs` to control policy-specific aspects of driver behavior.

The generic attributes under `/sys/devices/system/cpu/cpufreq/policyX/` are the following:

`affected_cpus`
: List of online CPUs belonging to this policy (i.e. sharing the hardware performance scaling interface represented by the `policyX` policy object).

`bios_limit`
: If the platform firmware (BIOS) tells the OS to apply an upper limit to CPU frequencies, that limit will be reported through this attribute (if present).

  The existence of the limit may be a result of some (often unintentional) BIOS settings, restrictions coming from a service processor or another BIOS/HW-based mechanisms.

  This does not cover ACPI thermal limitations which can be discovered through a generic thermal driver.

  This attribute is not present if the scaling driver in use does not support it.

cpuinfo_cur_freq
Current frequency of the CPUs belonging to this policy as obtained from the hardware (in KHz).

This is expected to be the frequency the hardware actually runs at. If that frequency cannot be determined, this attribute should not be present.

cpuinfo_max_freq
Maximum possible operating frequency the CPUs belonging to this policy can run at (in kHz).

cpuinfo_min_freq
Minimum possible operating frequency the CPUs belonging to this policy can run at (in kHz).

cpuinfo_transition_latency
The time it takes to switch the CPUs belonging to this policy from one P-state to another, in nanoseconds.

If unknown or if known to be so high that the scaling driver does not work with the *ondemand* governor, -1 (`CPUFREQ_ETERNAL`) will be returned by reads from this attribute.

related_cpus
List of all (online and offline) CPUs belonging to this policy.

scaling_available_governors
List of `CPUFreq` scaling governors present in the kernel that can be attached to this policy or (if the *intel_pstate* scaling driver is in use) list of scaling algorithms provided by the driver that can be applied to this policy.

[Note that some governors are modular and it may be necessary to load a kernel module for the governor held by it to become available and be listed by this attribute.]

scaling_cur_freq
Current frequency of all of the CPUs belonging to this policy (in kHz).

In the majority of cases, this is the frequency of the last P-state requested by the scaling driver from the hardware using the scaling interface provided by it, which may or may not reflect the frequency the CPU is actually running at (due to hardware design and other limitations).

Some architectures (e.g. `x86`) may attempt to provide information more precisely reflecting the current CPU frequency through this attribute, but that still may not be the exact current CPU frequency as seen by the hardware at the moment.

scaling_driver
The scaling driver currently in use.

scaling_governor
The scaling governor currently attached to this policy or (if the *intel_pstate* scaling driver is in use) the scaling algorithm provided by the driver that is currently applied to this policy.

This attribute is read-write and writing to it will cause a new scaling governor to be attached to this policy or a new scaling algorithm provided by the scaling driver to be applied to it (in the *intel_pstate* case), as indicated by the string written to this attribute (which must be one of the names listed by the `scaling_available_governors` attribute described above).

scaling_max_freq
Maximum frequency the CPUs belonging to this policy are allowed to be running at (in kHz).

This attribute is read-write and writing a string representing an integer to it will cause a new limit to be set (it must not be lower than the value of the `scaling_min_freq` attribute).

scaling_min_freq
Minimum frequency the CPUs belonging to this policy are allowed to be running at (in kHz).

This attribute is read-write and writing a string representing a non-negative integer to it will cause a new limit to be set (it must not be higher than the value of the `scaling_max_freq` attribute).

scaling_setspeed
This attribute is functional only if the *userspace* scaling governor is attached to the given policy.

It returns the last frequency requested by the governor (in kHz) or can be written to in order to set a new frequency for the policy.

Generic Scaling Governors

`CPUFreq` provides generic scaling governors that can be used with all scaling drivers. As stated before, each of them implements a single, possibly parametrized, performance scaling algorithm.

Scaling governors are attached to policy objects and different policy objects can be handled by different scaling governors at the same time (although that may lead to suboptimal results in some cases).

The scaling governor for a given policy object can be changed at any time with the help of the `scaling_governor` policy attribute in `sysfs`.

Some governors expose `sysfs` attributes to control or fine-tune the scaling algorithms implemented by them. Those attributes, referred to as governor tunables, can be either global (system-wide) or per-policy, depending on the scaling driver in use. If the driver requires governor tunables to be per-policy, they are located in a subdirectory of each policy directory. Otherwise, they are located in a subdirectory under `/sys/devices/system/cpu/cpufreq/`. In either case the name of the subdirectory containing the governor tunables is the name of the governor providing them.

`performance`

When attached to a policy object, this governor causes the highest frequency, within the `scaling_max_freq` policy limit, to be requested for that policy.

The request is made once at that time the governor for the policy is set to `performance` and whenever the `scaling_max_freq` or `scaling_min_freq` policy limits change after that.

powersave

When attached to a policy object, this governor causes the lowest frequency, within the `scaling_min_freq` policy limit, to be requested for that policy.

The request is made once at that time the governor for the policy is set to `powersave` and whenever the `scaling_max_freq` or `scaling_min_freq` policy limits change after that.

userspace

This governor does not do anything by itself. Instead, it allows user space to set the CPU frequency for the policy it is attached to by writing to the `scaling_setspeed` attribute of that policy.

schedutil

This governor uses CPU utilization data available from the CPU scheduler. It generally is regarded as a part of the CPU scheduler, so it can access the scheduler's internal data structures directly.

It runs entirely in scheduler context, although in some cases it may need to invoke the scaling driver asynchronously when it decides that the CPU frequency should be changed for a given policy (that depends on whether or not the driver is capable of changing the CPU frequency from scheduler context).

The actions of this governor for a particular CPU depend on the scheduling class invoking its utilization update callback for that CPU. If it is invoked by the RT or deadline scheduling classes, the governor will increase the frequency to the allowed maximum (that is, the `scaling_max_freq` policy limit). In turn, if it is invoked by the CFS scheduling class, the governor will use the Per-Entity Load Tracking (PELT) metric for the root control group of the given CPU as the CPU utilization estimate (see the *Per-entity load tracking* LWN.net article[1] for a description of the PELT mechanism). Then, the new CPU frequency to apply is computed in accordance with the formula

f = 1.25 * `f_0` * `util` / `max`

where `util` is the PELT number, `max` is the theoretical maximum of `util`, and `f_0` is either the maximum possible CPU frequency for the given policy (if the PELT number is frequency-invariant), or the current CPU frequency (otherwise).

This governor also employs a mechanism allowing it to temporarily bump up the CPU frequency for tasks that have been waiting on I/O most recently, called "IO-wait boosting". That happens when the `SCHED_CPUFREQ_IOWAIT` flag is passed by the scheduler to the governor callback which causes the frequency to go up to the allowed maximum immediately and then draw back to the value returned by the above formula over time.

This governor exposes only one tunable:

`rate_limit_us`
: Minimum time (in microseconds) that has to pass between two consecutive runs of governor computations (default: 1000 times the scaling driver's transition latency).

[1] Jonathan Corbet, *Per-entity load tracking*, https://lwn.net/Articles/531853/

The purpose of this tunable is to reduce the scheduler context overhead of the governor which might be excessive without it.

This governor generally is regarded as a replacement for the older *ondemand* and *conservative* governors (described below), as it is simpler and more tightly integrated with the CPU scheduler, its overhead in terms of CPU context switches and similar is less significant, and it uses the scheduler's own CPU utilization metric, so in principle its decisions should not contradict the decisions made by the other parts of the scheduler.

ondemand

This governor uses CPU load as a CPU frequency selection metric.

In order to estimate the current CPU load, it measures the time elapsed between consecutive invocations of its worker routine and computes the fraction of that time in which the given CPU was not idle. The ratio of the non-idle (active) time to the total CPU time is taken as an estimate of the load.

If this governor is attached to a policy shared by multiple CPUs, the load is estimated for all of them and the greatest result is taken as the load estimate for the entire policy.

The worker routine of this governor has to run in process context, so it is invoked asynchronously (via a workqueue) and CPU P-states are updated from there if necessary. As a result, the scheduler context overhead from this governor is minimum, but it causes additional CPU context switches to happen relatively often and the CPU P-state updates triggered by it can be relatively irregular. Also, it affects its own CPU load metric by running code that reduces the CPU idle time (even though the CPU idle time is only reduced very slightly by it).

It generally selects CPU frequencies proportional to the estimated load, so that the value of the `cpuinfo_max_freq` policy attribute corresponds to the load of 1 (or 100%), and the value of the `cpuinfo_min_freq` policy attribute corresponds to the load of 0, unless when the load exceeds a (configurable) speedup threshold, in which case it will go straight for the highest frequency it is allowed to use (the `scaling_max_freq` policy limit).

This governor exposes the following tunables:

sampling_rate
This is how often the governor's worker routine should run, in microseconds.

Typically, it is set to values of the order of 10000 (10 ms). Its default value is equal to the value of `cpuinfo_transition_latency` for each policy this governor is attached to (but since the unit here is greater by 1000, this means that the time represented by `sampling_rate` is 1000 times greater than the transition latency by default).

If this tunable is per-policy, the following shell command sets the time represented by it to be 750 times as high as the transition latency:

```
# echo `$(($(cat cpuinfo_transition_latency) * 750 / 1000)) > ondemand/
↪sampling_rate
```

up_threshold
If the estimated CPU load is above this value (in percent), the governor will set the frequency to the maximum value allowed for the policy. Otherwise, the selected frequency will be proportional to the estimated CPU load.

ignore_nice_load

If set to 1 (default 0), it will cause the CPU load estimation code to treat the CPU time spent on executing tasks with "nice" levels greater than 0 as CPU idle time.

This may be useful if there are tasks in the system that should not be taken into account when deciding what frequency to run the CPUs at. Then, to make that happen it is sufficient to increase the "nice" level of those tasks above 0 and set this attribute to 1.

sampling_down_factor

Temporary multiplier, between 1 (default) and 100 inclusive, to apply to the `sampling_rate` value if the CPU load goes above `up_threshold`.

This causes the next execution of the governor's worker routine (after setting the frequency to the allowed maximum) to be delayed, so the frequency stays at the maximum level for a longer time.

Frequency fluctuations in some bursty workloads may be avoided this way at the cost of additional energy spent on maintaining the maximum CPU capacity.

powersave_bias

Reduction factor to apply to the original frequency target of the governor (including the maximum value used when the `up_threshold` value is exceeded by the estimated CPU load) or sensitivity threshold for the AMD frequency sensitivity powersave bias driver (`drivers/cpufreq/amd_freq_sensitivity.c`), between 0 and 1000 inclusive.

If the AMD frequency sensitivity powersave bias driver is not loaded, the effective frequency to apply is given by

f * (1 - `powersave_bias` / 1000)

where f is the governor's original frequency target. The default value of this attribute is 0 in that case.

If the AMD frequency sensitivity powersave bias driver is loaded, the value of this attribute is 400 by default and it is used in a different way.

On Family 16h (and later) AMD processors there is a mechanism to get a measured workload sensitivity, between 0 and 100% inclusive, from the hardware. That value can be used to estimate how the performance of the workload running on a CPU will change in response to frequency changes.

The performance of a workload with the sensitivity of 0 (memory-bound or IO-bound) is not expected to increase at all as a result of increasing the CPU frequency, whereas workloads with the sensitivity of 100% (CPU-bound) are expected to perform much better if the CPU frequency is increased.

If the workload sensitivity is less than the threshold represented by the `powersave_bias` value, the sensitivity powersave bias driver will cause the governor to select a frequency lower than its original target, so as to avoid over-provisioning workloads that will not benefit from running at higher CPU frequencies.

conservative

This governor uses CPU load as a CPU frequency selection metric.

It estimates the CPU load in the same way as the *ondemand* governor described above, but the CPU frequency selection algorithm implemented by it is different.

Namely, it avoids changing the frequency significantly over short time intervals which may not be suitable for systems with limited power supply capacity (e.g. battery-powered). To achieve that, it changes the frequency in relatively small steps, one step at a time, up or down - depending on whether or not a (configurable) threshold has been exceeded by the estimated CPU load.

This governor exposes the following tunables:

freq_step
: Frequency step in percent of the maximum frequency the governor is allowed to set (the `scaling_max_freq` policy limit), between 0 and 100 (5 by default).

  This is how much the frequency is allowed to change in one go. Setting it to 0 will cause the default frequency step (5 percent) to be used and setting it to 100 effectively causes the governor to periodically switch the frequency between the `scaling_min_freq` and `scaling_max_freq` policy limits.

down_threshold
: Threshold value (in percent, 20 by default) used to determine the frequency change direction.

  If the estimated CPU load is greater than this value, the frequency will go up (by `freq_step`). If the load is less than this value (and the `sampling_down_factor` mechanism is not in effect), the frequency will go down. Otherwise, the frequency will not be changed.

sampling_down_factor
: Frequency decrease deferral factor, between 1 (default) and 10 inclusive.

  It effectively causes the frequency to go down `sampling_down_factor` times slower than it ramps up.

Frequency Boost Support

Background

Some processors support a mechanism to raise the operating frequency of some cores in a multicore package temporarily (and above the sustainable frequency threshold for the whole package) under certain conditions, for example if the whole chip is not fully utilized and below its intended thermal or power budget.

Different names are used by different vendors to refer to this functionality. For Intel processors it is referred to as ”Turbo Boost”, AMD calls it ”Turbo-Core” or (in technical documentation) ”Core Performance Boost” and so on. As a rule, it also is implemented differently by different vendors. The simple term ”frequency boost” is used here for brevity to refer to all of those implementations.

The frequency boost mechanism may be either hardware-based or software-based. If it is hardware-based (e.g. on x86), the decision to trigger the boosting is made by the hardware

(although in general it requires the hardware to be put into a special state in which it can control the CPU frequency within certain limits). If it is software-based (e.g. on ARM), the scaling driver decides whether or not to trigger boosting and when to do that.

The `boost` File in `sysfs`

This file is located under `/sys/devices/system/cpu/cpufreq/` and controls the "boost" setting for the whole system. It is not present if the underlying scaling driver does not support the frequency boost mechanism (or supports it, but provides a driver-specific interface for controlling it, like *intel_pstate*).

If the value in this file is 1, the frequency boost mechanism is enabled. This means that either the hardware can be put into states in which it is able to trigger boosting (in the hardware-based case), or the software is allowed to trigger boosting (in the software-based case). It does not mean that boosting is actually in use at the moment on any CPUs in the system. It only means a permission to use the frequency boost mechanism (which still may never be used for other reasons).

If the value in this file is 0, the frequency boost mechanism is disabled and cannot be used at all.

The only values that can be written to this file are 0 and 1.

Rationale for Boost Control Knob

The frequency boost mechanism is generally intended to help to achieve optimum CPU performance on time scales below software resolution (e.g. below the scheduler tick interval) and it is demonstrably suitable for many workloads, but it may lead to problems in certain situations.

For this reason, many systems make it possible to disable the frequency boost mechanism in the platform firmware (BIOS) setup, but that requires the system to be restarted for the setting to be adjusted as desired, which may not be practical at least in some cases. For example:

1. Boosting means overclocking the processor, although under controlled conditions. Generally, the processor's energy consumption increases as a result of increasing its frequency and voltage, even temporarily. That may not be desirable on systems that switch to power sources of limited capacity, such as batteries, so the ability to disable the boost mechanism while the system is running may help there (but that depends on the workload too).
2. In some situations deterministic behavior is more important than performance or energy consumption (or both) and the ability to disable boosting while the system is running may be useful then.
3. To examine the impact of the frequency boost mechanism itself, it is useful to be able to run tests with and without boosting, preferably without restarting the system in the meantime.
4. Reproducible results are important when running benchmarks. Since the boosting functionality depends on the load of the whole package, single-thread performance may vary because of it which may lead to unreproducible results sometimes. That can be avoided by disabling the frequency boost mechanism before running benchmarks sensitive to that issue.

Legacy AMD cpb Knob

The AMD powernow-k8 scaling driver supports a sysfs knob very similar to the global boost one. It is used for disabling/enabling the "Core Performance Boost" feature of some AMD processors.

If present, that knob is located in every CPUFreq policy directory in sysfs (/sys/devices/system/cpu/cpufreq/policyX/) and is called cpb, which indicates a more fine grained control interface. The actual implementation, however, works on the system-wide basis and setting that knob for one policy causes the same value of it to be set for all of the other policies at the same time.

That knob is still supported on AMD processors that support its underlying hardware feature, but it may be configured out of the kernel (via the CONFIG_X86_ACPI_CPUFREQ_CPB configuration option) and the global boost knob is present regardless. Thus it is always possible use the boost knob instead of the cpb one which is highly recommended, as that is more consistent with what all of the other systems do (and the cpb knob may not be supported any more in the future).

The cpb knob is never present for any processors without the underlying hardware feature (e.g. all Intel ones), even if the CONFIG_X86_ACPI_CPUFREQ_CPB configuration option is set.

References

67.3.4 intel_pstate CPU Performance Scaling Driver

Copyright
© 2017 Intel Corporation

Author
Rafael J. Wysocki <rafael.j.wysocki@intel.com>

General Information

intel_pstate is a part of the *CPU performance scaling subsystem* in the Linux kernel (CPUFreq). It is a scaling driver for the Sandy Bridge and later generations of Intel processors. Note, however, that some of those processors may not be supported. [To understand intel_pstate it is necessary to know how CPUFreq works in general, so this is the time to read *CPU Performance Scaling* if you have not done that yet.]

For the processors supported by intel_pstate, the P-state concept is broader than just an operating frequency or an operating performance point (see the LinuxCon Europe 2015 presentation by Kristen Accardi[1] for more information about that). For this reason, the representation of P-states used by intel_pstate internally follows the hardware specification (for details refer to Intel Software Developer's Manual[2]). However, the CPUFreq core uses frequencies for identifying operating performance points of CPUs and frequencies are involved in the user space interface exposed by it, so intel_pstate maps its internal representation of P-states to frequencies too (fortunately, that mapping is unambiguous). At the same time, it would not be

[1] Kristen Accardi, *Balancing Power and Performance in the Linux Kernel*, https://events.static.linuxfound.org/sites/events/files/slides/LinuxConEurope_2015.pdf

[2] *Intel® 64 and IA-32 Architectures Software Developer's Manual Volume 3: System Programming Guide*, https://www.intel.com/content/www/us/en/architecture-and-technology/64-ia-32-architectures-software-developer-system-programming-manual-325384.html

practical for intel_pstate to supply the CPUFreq core with a table of available frequencies due to the possible size of it, so the driver does not do that. Some functionality of the core is limited by that.

Since the hardware P-state selection interface used by intel_pstate is available at the logical CPU level, the driver always works with individual CPUs. Consequently, if intel_pstate is in use, every CPUFreq policy object corresponds to one logical CPU and CPUFreq policies are effectively equivalent to CPUs. In particular, this means that they become "inactive" every time the corresponding CPU is taken offline and need to be re-initialized when it goes back online.

intel_pstate is not modular, so it cannot be unloaded, which means that the only way to pass early-configuration-time parameters to it is via the kernel command line. However, its configuration can be adjusted via sysfs to a great extent. In some configurations it even is possible to unregister it via sysfs which allows another CPUFreq scaling driver to be loaded and registered (see *below*).

Operation Modes

intel_pstate can operate in two different modes, active or passive. In the active mode, it uses its own internal performance scaling governor algorithm or allows the hardware to do performance scaling by itself, while in the passive mode it responds to requests made by a generic CPUFreq governor implementing a certain performance scaling algorithm. Which of them will be in effect depends on what kernel command line options are used and on the capabilities of the processor.

Active Mode

This is the default operation mode of intel_pstate for processors with hardware-managed P-states (HWP) support. If it works in this mode, the scaling_driver policy attribute in sysfs for all CPUFreq policies contains the string "intel_pstate".

In this mode the driver bypasses the scaling governors layer of CPUFreq and provides its own scaling algorithms for P-state selection. Those algorithms can be applied to CPUFreq policies in the same way as generic scaling governors (that is, through the scaling_governor policy attribute in sysfs). [Note that different P-state selection algorithms may be chosen for different policies, but that is not recommended.]

They are not generic scaling governors, but their names are the same as the names of some of those governors. Moreover, confusingly enough, they generally do not work in the same way as the generic governors they share the names with. For example, the powersave P-state selection algorithm provided by intel_pstate is not a counterpart of the generic powersave governor (roughly, it corresponds to the schedutil and ondemand governors).

There are two P-state selection algorithms provided by intel_pstate in the active mode: powersave and performance. The way they both operate depends on whether or not the hardware-managed P-states (HWP) feature has been enabled in the processor and possibly on the processor model.

Which of the P-state selection algorithms is used by default depends on the CONFIG_CPU_FREQ_DEFAULT_GOV_PERFORMANCE kernel configuration option. Namely, if that option is set, the performance algorithm will be used by default, and the other one will be used by default if it is not set.

Active Mode With HWP

If the processor supports the HWP feature, it will be enabled during the processor initialization and cannot be disabled after that. It is possible to avoid enabling it by passing the `intel_pstate=no_hwp` argument to the kernel in the command line.

If the HWP feature has been enabled, `intel_pstate` relies on the processor to select P-states by itself, but still it can give hints to the processor's internal P-state selection logic. What those hints are depends on which P-state selection algorithm has been applied to the given policy (or to the CPU it corresponds to).

Even though the P-state selection is carried out by the processor automatically, `intel_pstate` registers utilization update callbacks with the CPU scheduler in this mode. However, they are not used for running a P-state selection algorithm, but for periodic updates of the current CPU frequency information to be made available from the `scaling_cur_freq` policy attribute in `sysfs`.

HWP + `performance`

In this configuration `intel_pstate` will write 0 to the processor's Energy-Performance Preference (EPP) knob (if supported) or its Energy-Performance Bias (EPB) knob (otherwise), which means that the processor's internal P-state selection logic is expected to focus entirely on performance.

This will override the EPP/EPB setting coming from the `sysfs` interface (see *Energy vs Performance Hints* below). Moreover, any attempts to change the EPP/EPB to a value different from 0 ("performance") via `sysfs` in this configuration will be rejected.

Also, in this configuration the range of P-states available to the processor's internal P-state selection logic is always restricted to the upper boundary (that is, the maximum P-state that the driver is allowed to use).

HWP + `powersave`

In this configuration `intel_pstate` will set the processor's Energy-Performance Preference (EPP) knob (if supported) or its Energy-Performance Bias (EPB) knob (otherwise) to whatever value it was previously set to via `sysfs` (or whatever default value it was set to by the platform firmware). This usually causes the processor's internal P-state selection logic to be less performance-focused.

Active Mode Without HWP

This operation mode is optional for processors that do not support the HWP feature or when the `intel_pstate=no_hwp` argument is passed to the kernel in the command line. The active mode is used in those cases if the `intel_pstate=active` argument is passed to the kernel in the command line. In this mode `intel_pstate` may refuse to work with processors that are not recognized by it. [Note that `intel_pstate` will never refuse to work with any processor with the HWP feature enabled.]

In this mode `intel_pstate` registers utilization update callbacks with the CPU scheduler in order to run a P-state selection algorithm, either `powersave` or `performance`, depending on the

scaling_governor policy setting in sysfs. The current CPU frequency information to be made available from the scaling_cur_freq policy attribute in sysfs is periodically updated by those utilization update callbacks too.

performance

Without HWP, this P-state selection algorithm is always the same regardless of the processor model and platform configuration.

It selects the maximum P-state it is allowed to use, subject to limits set via sysfs, every time the driver configuration for the given CPU is updated (e.g. via sysfs).

This is the default P-state selection algorithm if the CONFIG_CPU_FREQ_DEFAULT_GOV_PERFORMANCE kernel configuration option is set.

powersave

Without HWP, this P-state selection algorithm is similar to the algorithm implemented by the generic schedutil scaling governor except that the utilization metric used by it is based on numbers coming from feedback registers of the CPU. It generally selects P-states proportional to the current CPU utilization.

This algorithm is run by the driver's utilization update callback for the given CPU when it is invoked by the CPU scheduler, but not more often than every 10 ms. Like in the performance case, the hardware configuration is not touched if the new P-state turns out to be the same as the current one.

This is the default P-state selection algorithm if the CONFIG_CPU_FREQ_DEFAULT_GOV_PERFORMANCE kernel configuration option is not set.

Passive Mode

This is the default operation mode of intel_pstate for processors without hardware-managed P-states (HWP) support. It is always used if the intel_pstate=passive argument is passed to the kernel in the command line regardless of whether or not the given processor supports HWP. [Note that the intel_pstate=no_hwp setting causes the driver to start in the passive mode if it is not combined with intel_pstate=active.] Like in the active mode without HWP support, in this mode intel_pstate may refuse to work with processors that are not recognized by it if HWP is prevented from being enabled through the kernel command line.

If the driver works in this mode, the scaling_driver policy attribute in sysfs for all CPUFreq policies contains the string ”intel_cpufreq”. Then, the driver behaves like a regular CPUFreq scaling driver. That is, it is invoked by generic scaling governors when necessary to talk to the hardware in order to change the P-state of a CPU (in particular, the schedutil governor can invoke it directly from scheduler context).

While in this mode, intel_pstate can be used with all of the (generic) scaling governors listed by the scaling_available_governors policy attribute in sysfs (and the P-state selection algorithms described above are not used). Then, it is responsible for the configuration of policy objects corresponding to CPUs and provides the CPUFreq core (and the scaling governors attached to the policy objects) with accurate information on the maximum and minimum

operating frequencies supported by the hardware (including the so-called ”turbo” frequency ranges). In other words, in the passive mode the entire range of available P-states is exposed by `intel_pstate` to the `CPUFreq` core. However, in this mode the driver does not register utilization update callbacks with the CPU scheduler and the `scaling_cur_freq` information comes from the `CPUFreq` core (and is the last frequency selected by the current scaling governor for the given policy).

Turbo P-states Support

In the majority of cases, the entire range of P-states available to `intel_pstate` can be divided into two sub-ranges that correspond to different types of processor behavior, above and below a boundary that will be referred to as the ”turbo threshold” in what follows.

The P-states above the turbo threshold are referred to as ”turbo P-states” and the whole sub-range of P-states they belong to is referred to as the ”turbo range”. These names are related to the Turbo Boost technology allowing a multicore processor to opportunistically increase the P-state of one or more cores if there is enough power to do that and if that is not going to cause the thermal envelope of the processor package to be exceeded.

Specifically, if software sets the P-state of a CPU core within the turbo range (that is, above the turbo threshold), the processor is permitted to take over performance scaling control for that core and put it into turbo P-states of its choice going forward. However, that permission is interpreted differently by different processor generations. Namely, the Sandy Bridge generation of processors will never use any P-states above the last one set by software for the given core, even if it is within the turbo range, whereas all of the later processor generations will take it as a license to use any P-states from the turbo range, even above the one set by software. In other words, on those processors setting any P-state from the turbo range will enable the processor to put the given core into all turbo P-states up to and including the maximum supported one as it sees fit.

One important property of turbo P-states is that they are not sustainable. More precisely, there is no guarantee that any CPUs will be able to stay in any of those states indefinitely, because the power distribution within the processor package may change over time or the thermal envelope it was designed for might be exceeded if a turbo P-state was used for too long.

In turn, the P-states below the turbo threshold generally are sustainable. In fact, if one of them is set by software, the processor is not expected to change it to a lower one unless in a thermal stress or a power limit violation situation (a higher P-state may still be used if it is set for another CPU in the same package at the same time, for example).

Some processors allow multiple cores to be in turbo P-states at the same time, but the maximum P-state that can be set for them generally depends on the number of cores running concurrently. The maximum turbo P-state that can be set for 3 cores at the same time usually is lower than the analogous maximum P-state for 2 cores, which in turn usually is lower than the maximum turbo P-state that can be set for 1 core. The one-core maximum turbo P-state is thus the maximum supported one overall.

The maximum supported turbo P-state, the turbo threshold (the maximum supported non-turbo P-state) and the minimum supported P-state are specific to the processor model and can be determined by reading the processor's model-specific registers (MSRs). Moreover, some processors support the Configurable TDP (Thermal Design Power) feature and, when that feature is enabled, the turbo threshold effectively becomes a configurable value that can be set by the platform firmware.

Unlike _PSS objects in the ACPI tables, intel_pstate always exposes the entire range of available P-states, including the whole turbo range, to the CPUFreq core and (in the passive mode) to generic scaling governors. This generally causes turbo P-states to be set more often when intel_pstate is used relative to ACPI-based CPU performance scaling (see *below* for more information).

Moreover, since intel_pstate always knows what the real turbo threshold is (even if the Configurable TDP feature is enabled in the processor), its no_turbo attribute in sysfs (described *below*) should work as expected in all cases (that is, if set to disable turbo P-states, it always should prevent intel_pstate from using them).

Processor Support

To handle a given processor intel_pstate requires a number of different pieces of information on it to be known, including:

- The minimum supported P-state.
- The maximum supported *non-turbo P-state*.
- Whether or not turbo P-states are supported at all.
- The maximum supported *one-core turbo P-state* (if turbo P-states are supported).
- The scaling formula to translate the driver's internal representation of P-states into frequencies and the other way around.

Generally, ways to obtain that information are specific to the processor model or family. Although it often is possible to obtain all of it from the processor itself (using model-specific registers), there are cases in which hardware manuals need to be consulted to get to it too.

For this reason, there is a list of supported processors in intel_pstate and the driver initialization will fail if the detected processor is not in that list, unless it supports the HWP feature. [The interface to obtain all of the information listed above is the same for all of the processors supporting the HWP feature, which is why intel_pstate works with all of them.]

User Space Interface in sysfs

Global Attributes

intel_pstate exposes several global attributes (files) in sysfs to control its functionality at the system level. They are located in the /sys/devices/system/cpu/intel_pstate/ directory and affect all CPUs.

Some of them are not present if the intel_pstate=per_cpu_perf_limits argument is passed to the kernel in the command line.

max_perf_pct

Maximum P-state the driver is allowed to set in percent of the maximum supported performance level (the highest supported *turbo P-state*).

This attribute will not be exposed if the intel_pstate=per_cpu_perf_limits argument is present in the kernel command line.

min_perf_pct
Minimum P-state the driver is allowed to set in percent of the maximum supported performance level (the highest supported *turbo P-state*).

This attribute will not be exposed if the `intel_pstate=per_cpu_perf_limits` argument is present in the kernel command line.

num_pstates
Number of P-states supported by the processor (between 0 and 255 inclusive) including both turbo and non-turbo P-states (see *Turbo P-states Support*).

This attribute is present only if the value exposed by it is the same for all of the CPUs in the system.

The value of this attribute is not affected by the `no_turbo` setting described *below*.

This attribute is read-only.

turbo_pct
Ratio of the *turbo range* size to the size of the entire range of supported P-states, in percent.

This attribute is present only if the value exposed by it is the same for all of the CPUs in the system.

This attribute is read-only.

no_turbo
If set (equal to 1), the driver is not allowed to set any turbo P-states (see *Turbo P-states Support*). If unset (equal to 0, which is the default), turbo P-states can be set by the driver. [Note that `intel_pstate` does not support the general `boost` attribute (supported by some other scaling drivers) which is replaced by this one.]

This attribute does not affect the maximum supported frequency value supplied to the `CPUFreq` core and exposed via the policy interface, but it affects the maximum possible value of per-policy P-state limits (see *Interpretation of Policy Attributes* below for details).

hwp_dynamic_boost
This attribute is only present if `intel_pstate` works in the *active mode with the HWP feature enabled* in the processor. If set (equal to 1), it causes the minimum P-state limit to be increased dynamically for a short time whenever a task previously waiting on I/O is selected to run on a given logical CPU (the purpose of this mechanism is to improve performance).

This setting has no effect on logical CPUs whose minimum P-state limit is directly set to the highest non-turbo P-state or above it.

status
Operation mode of the driver: "active", "passive" or "off".

"active"
The driver is functional and in the *active mode*.

"passive"
The driver is functional and in the *passive mode*.

"off"
The driver is not functional (it is not registered as a scaling driver with the `CPUFreq` core).

This attribute can be written to in order to change the driver's operation mode or to unregister it. The string written to it must be one of the possible values of it and, if successful, the write will cause the driver to switch over to the operation mode represented by that string - or to be unregistered in the "off" case. [Actually, switching over from the active mode to the passive mode or the other way around causes the driver to be unregistered and registered again with a different set of callbacks, so all of its settings (the global as well as the per-policy ones) are then reset to their default values, possibly depending on the target operation mode.]

energy_efficiency
This attribute is only present on platforms with CPUs matching the Kaby Lake or Coffee Lake desktop CPU model. By default, energy-efficiency optimizations are disabled on these CPU models if HWP is enabled. Enabling energy-efficiency optimizations may limit maximum operating frequency with or without the HWP feature. With HWP enabled, the optimizations are done only in the turbo frequency range. Without it, they are done in the entire available frequency range. Setting this attribute to "1" enables the energy-efficiency optimizations and setting to "0" disables them.

Interpretation of Policy Attributes

The interpretation of some `CPUFreq` policy attributes described in *CPU Performance Scaling* is special with `intel_pstate` as the current scaling driver and it generally depends on the driver's *operation mode*.

First of all, the values of the `cpuinfo_max_freq`, `cpuinfo_min_freq` and `scaling_cur_freq` attributes are produced by applying a processor-specific multiplier to the internal P-state representation used by `intel_pstate`. Also, the values of the `scaling_max_freq` and `scaling_min_freq` attributes are capped by the frequency corresponding to the maximum P-state that the driver is allowed to set.

If the `no_turbo` *global attribute* is set, the driver is not allowed to use turbo P-states, so the maximum value of `scaling_max_freq` and `scaling_min_freq` is limited to the maximum non-turbo P-state frequency. Accordingly, setting `no_turbo` causes `scaling_max_freq` and `scaling_min_freq` to go down to that value if they were above it before. However, the old values of `scaling_max_freq` and `scaling_min_freq` will be restored after unsetting `no_turbo`, unless these attributes have been written to after `no_turbo` was set.

If `no_turbo` is not set, the maximum possible value of `scaling_max_freq` and `scaling_min_freq` corresponds to the maximum supported turbo P-state, which also is the value of `cpuinfo_max_freq` in either case.

Next, the following policy attributes have special meaning if `intel_pstate` works in the *active mode*:

scaling_available_governors
List of P-state selection algorithms provided by `intel_pstate`.

scaling_governor
P-state selection algorithm provided by `intel_pstate` currently in use with the given policy.

scaling_cur_freq
Frequency of the average P-state of the CPU represented by the given policy for the time

interval between the last two invocations of the driver's utilization update callback by the CPU scheduler for that CPU.

One more policy attribute is present if the HWP feature is enabled in the processor:

base_frequency
Shows the base frequency of the CPU. Any frequency above this will be in the turbo frequency range.

The meaning of these attributes in the *passive mode* is the same as for other scaling drivers.

Additionally, the value of the `scaling_driver` attribute for `intel_pstate` depends on the operation mode of the driver. Namely, it is either "intel_pstate" (in the *active mode*) or "intel_cpufreq" (in the *passive mode*).

Coordination of P-State Limits

`intel_pstate` allows P-state limits to be set in two ways: with the help of the `max_perf_pct` and `min_perf_pct` *global attributes* or via the `scaling_max_freq` and `scaling_min_freq` `CPUFreq` policy attributes. The coordination between those limits is based on the following rules, regardless of the current operation mode of the driver:

1. All CPUs are affected by the global limits (that is, none of them can be requested to run faster than the global maximum and none of them can be requested to run slower than the global minimum).
2. Each individual CPU is affected by its own per-policy limits (that is, it cannot be requested to run faster than its own per-policy maximum and it cannot be requested to run slower than its own per-policy minimum). The effective performance depends on whether the platform supports per core P-states, hyper-threading is enabled and on current performance requests from other CPUs. When platform doesn't support per core P-states, the effective performance can be more than the policy limits set on a CPU, if other CPUs are requesting higher performance at that moment. Even with per core P-states support, when hyper-threading is enabled, if the sibling CPU is requesting higher performance, the other siblings will get higher performance than their policy limits.
3. The global and per-policy limits can be set independently.

In the *active mode with the HWP feature enabled*, the resulting effective values are written into hardware registers whenever the limits change in order to request its internal P-state selection logic to always set P-states within these limits. Otherwise, the limits are taken into account by scaling governors (in the *passive mode*) and by the driver every time before setting a new P-state for a CPU.

Additionally, if the `intel_pstate=per_cpu_perf_limits` command line argument is passed to the kernel, `max_perf_pct` and `min_perf_pct` are not exposed at all and the only way to set the limits is by using the policy attributes.

Energy vs Performance Hints

If the hardware-managed P-states (HWP) is enabled in the processor, additional attributes, intended to allow user space to help `intel_pstate` to adjust the processor's internal P-state selection logic by focusing it on performance or on energy-efficiency, or somewhere between the two extremes, are present in every `CPUFreq` policy directory in `sysfs`. They are :

`energy_performance_preference`
: Current value of the energy vs performance hint for the given policy (or the CPU represented by it).

  The hint can be changed by writing to this attribute.

`energy_performance_available_preferences`
: List of strings that can be written to the `energy_performance_preference` attribute.

  They represent different energy vs performance hints and should be self-explanatory, except that `default` represents whatever hint value was set by the platform firmware.

Strings written to the `energy_performance_preference` attribute are internally translated to integer values written to the processor's Energy-Performance Preference (EPP) knob (if supported) or its Energy-Performance Bias (EPB) knob. It is also possible to write a positive integer value between 0 to 255, if the EPP feature is present. If the EPP feature is not present, writing integer value to this attribute is not supported. In this case, user can use the ”/sys/devices/system/cpu/cpu*/power/energy_perf_bias” interface.

[Note that tasks may by migrated from one CPU to another by the scheduler's load-balancing algorithm and if different energy vs performance hints are set for those CPUs, that may lead to undesirable outcomes. To avoid such issues it is better to set the same energy vs performance hint for all CPUs or to pin every task potentially sensitive to them to a specific CPU.]

`intel_pstate` vs `acpi-cpufreq`

On the majority of systems supported by `intel_pstate`, the ACPI tables provided by the platform firmware contain `_PSS` objects returning information that can be used for CPU performance scaling (refer to the ACPI specification[3] for details on the `_PSS` objects and the format of the information returned by them).

The information returned by the ACPI `_PSS` objects is used by the `acpi-cpufreq` scaling driver. On systems supported by `intel_pstate` the `acpi-cpufreq` driver uses the same hardware CPU performance scaling interface, but the set of P-states it can use is limited by the `_PSS` output.

On those systems each `_PSS` object returns a list of P-states supported by the corresponding CPU which basically is a subset of the P-states range that can be used by `intel_pstate` on the same system, with one exception: the whole *turbo range* is represented by one item in it (the topmost one). By convention, the frequency returned by `_PSS` for that item is greater by 1 MHz than the frequency of the highest non-turbo P-state listed by it, but the corresponding P-state representation (following the hardware specification) returned for it matches the maximum supported turbo P-state (or is the special value 255 meaning essentially ”go as high as you can get”).

[3] *Advanced Configuration and Power Interface Specification*, https://uefi.org/sites/default/files/resources/ACPI_6_3_final_Jan30.pdf

The list of P-states returned by `_PSS` is reflected by the table of available frequencies supplied by `acpi-cpufreq` to the `CPUFreq` core and scaling governors and the minimum and maximum supported frequencies reported by it come from that list as well. In particular, given the special representation of the turbo range described above, this means that the maximum supported frequency reported by `acpi-cpufreq` is higher by 1 MHz than the frequency of the highest supported non-turbo P-state listed by `_PSS` which, of course, affects decisions made by the scaling governors, except for `powersave` and `performance`.

For example, if a given governor attempts to select a frequency proportional to estimated CPU load and maps the load of 100% to the maximum supported frequency (possibly multiplied by a constant), then it will tend to choose P-states below the turbo threshold if `acpi-cpufreq` is used as the scaling driver, because in that case the turbo range corresponds to a small fraction of the frequency band it can use (1 MHz vs 1 GHz or more). In consequence, it will only go to the turbo range for the highest loads and the other loads above 50% that might benefit from running at turbo frequencies will be given non-turbo P-states instead.

One more issue related to that may appear on systems supporting the *Configurable TDP feature* allowing the platform firmware to set the turbo threshold. Namely, if that is not coordinated with the lists of P-states returned by `_PSS` properly, there may be more than one item corresponding to a turbo P-state in those lists and there may be a problem with avoiding the turbo range (if desirable or necessary). Usually, to avoid using turbo P-states overall, `acpi-cpufreq` simply avoids using the topmost state listed by `_PSS`, but that is not sufficient when there are other turbo P-states in the list returned by it.

Apart from the above, `acpi-cpufreq` works like `intel_pstate` in the *passive mode*, except that the number of P-states it can set is limited to the ones listed by the ACPI `_PSS` objects.

Kernel Command Line Options for `intel_pstate`

Several kernel command line options can be used to pass early-configuration-time parameters to `intel_pstate` in order to enforce specific behavior of it. All of them have to be prepended with the `intel_pstate=` prefix.

`disable`
: Do not register `intel_pstate` as the scaling driver even if the processor is supported by it.

`active`
: Register `intel_pstate` in the *active mode* to start with.

`passive`
: Register `intel_pstate` in the *passive mode* to start with.

`force`
: Register `intel_pstate` as the scaling driver instead of `acpi-cpufreq` even if the latter is preferred on the given system.

  This may prevent some platform features (such as thermal controls and power capping) that rely on the availability of ACPI P-states information from functioning as expected, so it should be used with caution.

  This option does not work with processors that are not supported by `intel_pstate` and on platforms where the `pcc-cpufreq` scaling driver is used instead of `acpi-cpufreq`.

no_hwp
Do not enable the hardware-managed P-states (HWP) feature even if it is supported by the processor.

hwp_only
Register `intel_pstate` as the scaling driver only if the hardware-managed P-states (HWP) feature is supported by the processor.

support_acpi_ppc
Take ACPI `_PPC` performance limits into account.

If the preferred power management profile in the FADT (Fixed ACPI Description Table) is set to "Enterprise Server" or "Performance Server", the ACPI `_PPC` limits are taken into account by default and this option has no effect.

per_cpu_perf_limits
Use per-logical-CPU P-State limits (see *Coordination of P-state Limits* for details).

Diagnostics and Tuning

Trace Events

There are two static trace events that can be used for `intel_pstate` diagnostics. One of them is the `cpu_frequency` trace event generally used by `CPUFreq`, and the other one is the `pstate_sample` trace event specific to `intel_pstate`. Both of them are triggered by `intel_pstate` only if it works in the *active mode*.

The following sequence of shell commands can be used to enable them and see their output (if the kernel is generally configured to support event tracing):

```
# cd /sys/kernel/tracing/
# echo 1 > events/power/pstate_sample/enable
# echo 1 > events/power/cpu_frequency/enable
# cat trace
gnome-terminal--4510  [001] ..s.  1177.680733: pstate_sample: core_busy=107␣
↪scaled=94 from=26 to=26 mperf=1143818 aperf=1230607 tsc=29838618 freq=2474476
cat-5235  [002] ..s.  1177.681723: cpu_frequency: state=2900000 cpu_id=2
```

If `intel_pstate` works in the *passive mode*, the `cpu_frequency` trace event will be triggered either by the `schedutil` scaling governor (for the policies it is attached to), or by the `CPUFreq` core (for the policies with other scaling governors).

ftrace

The `ftrace` interface can be used for low-level diagnostics of `intel_pstate`. For example, to check how often the function to set a P-state is called, the `ftrace` filter can be set to `intel_pstate_set_pstate()`:

```
# cd /sys/kernel/tracing/
# cat available_filter_functions | grep -i pstate
intel_pstate_set_pstate
intel_pstate_cpu_init
```

```
...
# echo intel_pstate_set_pstate > set_ftrace_filter
# echo function > current_tracer
# cat trace | head -15
# tracer: function
#
# entries-in-buffer/entries-written: 80/80   #P:4
#
#                              _-----=> irqs-off
#                             / _----=> need-resched
#                            | / _---=> hardirq/softirq
#                            || / _--=> preempt-depth
#                            ||| /     delay
#           TASK-PID   CPU#  ||||    TIMESTAMP  FUNCTION
#              | |       |   ||||       |         |
            Xorg-3129  [000] ..s.  2537.644844: intel_pstate_set_pstate <-
↪intel_pstate_timer_func
 gnome-terminal--4510  [002] ..s.  2537.649844: intel_pstate_set_pstate <-
↪intel_pstate_timer_func
     gnome-shell-3409  [001] ..s.  2537.650850: intel_pstate_set_pstate <-
↪intel_pstate_timer_func
          <idle>-0     [000] ..s.  2537.654843: intel_pstate_set_pstate <-
↪intel_pstate_timer_func
```

References

67.3.5 amd-pstate CPU Performance Scaling Driver

Copyright
© 2021 Advanced Micro Devices, Inc.

Author
Huang Rui <ray.huang@amd.com>

Introduction

`amd-pstate` is the AMD CPU performance scaling driver that introduces a new CPU frequency control mechanism on modern AMD APU and CPU series in Linux kernel. The new mechanism is based on Collaborative Processor Performance Control (CPPC) which provides finer grain frequency management than legacy ACPI hardware P-States. Current AMD CPU/APU platforms are using the ACPI P-states driver to manage CPU frequency and clocks with switching only in 3 P-states. CPPC replaces the ACPI P-states controls and allows a flexible, low-latency interface for the Linux kernel to directly communicate the performance hints to hardware.

`amd-pstate` leverages the Linux kernel governors such as `schedutil`, `ondemand`, etc. to manage the performance hints which are provided by CPPC hardware functionality that internally follows the hardware specification (for details refer to AMD64 Architecture Programmer's Manual Volume 2: System Programming[1]). Currently, `amd-pstate` supports basic frequency control

[1] AMD64 Architecture Programmer's Manual Volume 2: System Programming, https://www.amd.com/system/

function according to kernel governors on some of the Zen2 and Zen3 processors, and we will implement more AMD specific functions in future after we verify them on the hardware and SBIOS.

AMD CPPC Overview

Collaborative Processor Performance Control (CPPC) interface enumerates a continuous, abstract, and unit-less performance value in a scale that is not tied to a specific performance state / frequency. This is an ACPI standard[2] which software can specify application performance goals and hints as a relative target to the infrastructure limits. AMD processors provide the low latency register model (MSR) instead of an AML code interpreter for performance adjustments. `amd-pstate` will initialize a `struct cpufreq_driver` instance, `amd_pstate_driver`, with the callbacks to manage each performance update behavior.

```
Highest Perf ------>+-----------------------+                          +--------
 ↪---------------+
                    |                       |                          |       ␣
 ↪               |
                    |                       |                          |       ␣
 ↪               |
                    |                       |          Max Perf  ---->|       ␣
 ↪               |
                    |                       |                          |       ␣
 ↪               |
                    |                       |                          |       ␣
 ↪               |
Nominal Perf ------>+-----------------------+                          +--------
 ↪---------------+
                    |                       |                          |       ␣
 ↪               |
                    |                       |                          |       ␣
 ↪               |
                    |                       |                          |       ␣
 ↪               |
                    |                       |                          |       ␣
 ↪               |
                    |                       |                          |       ␣
 ↪               |
                    |                       |                          |       ␣
 ↪               |
                    |                       |      Desired Perf  ---->|       ␣
 ↪               |
                    |                       |                          |       ␣
 ↪               |
                    |                       |                          |       ␣
 ↪               |
                    |                       |                          |       ␣
 ↪               |
```

files/TechDocs/24593.pdf

[2] Advanced Configuration and Power Interface Specification, https://uefi.org/sites/default/files/resources/ACPI_Spec_6_4_Jan22.pdf

```
                    |                        |                          |       ␣
↪               |
                    |                        |                          |       ␣
↪               |
                    |                        |                          |       ␣
↪               |
                    |                        |                          |       ␣
↪               |
                    |                        |                          |       ␣
↪               |
                    |                        |                          |       ␣
↪               |
Lowest non-         |                        |                          |       ␣
↪               |
linear perf ------>+------------------------+                          +--------
↪---------------+
                    |                        |                          |       ␣
↪               |
                    |                        |       Lowest perf  ---->|       ␣
↪               |
                    |                        |                          |       ␣
↪               |
Lowest perf ------>+------------------------+                          +--------
↪---------------+
                    |                        |                          |       ␣
↪               |
                    |                        |                          |       ␣
↪               |
                    |                        |                          |       ␣
↪               |
        0   ------>+------------------------+                          +--------
↪---------------+

                                    AMD P-States Performance Scale
```

AMD CPPC Performance Capability

Highest Performance (RO)

This is the absolute maximum performance an individual processor may reach, assuming ideal conditions. This performance level may not be sustainable for long durations and may only be achievable if other platform components are in a specific state; for example, it may require other processors to be in an idle state. This would be equivalent to the highest frequencies supported by the processor.

Nominal (Guaranteed) Performance (RO)

This is the maximum sustained performance level of the processor, assuming ideal operating conditions. In the absence of an external constraint (power, thermal, etc.), this is the performance level the processor is expected to be able to maintain continuously. All cores/processors are expected to be able to sustain their nominal performance state simultaneously.

Lowest non-linear Performance (RO)

This is the lowest performance level at which nonlinear power savings are achieved, for example, due to the combined effects of voltage and frequency scaling. Above this threshold, lower performance levels should be generally more energy efficient than higher performance levels. This register effectively conveys the most efficient performance level to `amd-pstate`.

Lowest Performance (RO)

This is the absolute lowest performance level of the processor. Selecting a performance level lower than the lowest nonlinear performance level may cause an efficiency penalty but should reduce the instantaneous power consumption of the processor.

AMD CPPC Performance Control

`amd-pstate` passes performance goals through these registers. The register drives the behavior of the desired performance target.

Minimum requested performance (RW)

`amd-pstate` specifies the minimum allowed performance level.

Maximum requested performance (RW)

`amd-pstate` specifies a limit the maximum performance that is expected to be supplied by the hardware.

Desired performance target (RW)

`amd-pstate` specifies a desired target in the CPPC performance scale as a relative number. This can be expressed as percentage of nominal performance (infrastructure max). Below the nominal sustained performance level, desired performance expresses the average performance level of the processor subject to hardware. Above the nominal performance level, the processor must provide at least nominal performance requested and go higher if current operating conditions allow.

Energy Performance Preference (EPP) (RW)

This attribute provides a hint to the hardware if software wants to bias toward performance (0x0) or energy efficiency (0xff).

Key Governors Support

`amd-pstate` can be used with all the (generic) scaling governors listed by the `scaling_available_governors` policy attribute in `sysfs`. Then, it is responsible for the configuration of policy objects corresponding to CPUs and provides the `CPUFreq` core (and the scaling governors attached to the policy objects) with accurate information on the maximum and minimum operating frequencies supported by the hardware. Users can check the `scaling_cur_freq` information comes from the `CPUFreq` core.

`amd-pstate` mainly supports `schedutil` and `ondemand` for dynamic frequency control. It is to fine tune the processor configuration on `amd-pstate` to the `schedutil` with CPU CFS scheduler. `amd-pstate` registers the adjust_perf callback to implement performance update behavior similar to CPPC. It is initialized by `sugov_start` and then populates the CPU's update_util_data pointer to assign `sugov_update_single_perf` as the utilization update callback function in the CPU scheduler. The CPU scheduler will call `cpufreq_update_util` and assigns the target performance according to the `struct sugov_cpu` that the utilization update belongs to. Then, `amd-pstate` updates the desired performance according to the CPU scheduler assigned.

Processor Support

The `amd-pstate` initialization will fail if the `_CPC` entry in the ACPI SBIOS does not exist in the detected processor. It uses `acpi_cpc_valid` to check the existence of `_CPC`. All Zen based processors support the legacy ACPI hardware P-States function, so when `amd-pstate` fails initialization, the kernel will fall back to initialize the `acpi-cpufreq` driver.

There are two types of hardware implementations for `amd-pstate`: one is *Full MSR Support* and another is *Shared Memory Support*. It can use the `X86_FEATURE_CPPC` feature flag to indicate the different types. (For details, refer to the Processor Programming Reference (PPR) for AMD Family 19h Model 51h, Revision A1 Processors[3].) `amd-pstate` is to register different `static_call` instances for different hardware implementations.

Currently, some of the Zen2 and Zen3 processors support `amd-pstate`. In the future, it will be supported on more and more AMD processors.

[3] Processor Programming Reference (PPR) for AMD Family 19h Model 51h, Revision A1 Processors https://www.amd.com/system/files/TechDocs/56569-A1-PUB.zip

Full MSR Support

Some new Zen3 processors such as Cezanne provide the MSR registers directly while the X86_FEATURE_CPPC CPU feature flag is set. amd-pstate can handle the MSR register to implement the fast switch function in CPUFreq that can reduce the latency of frequency control in interrupt context. The functions with a pstate_xxx prefix represent the operations on MSR registers.

Shared Memory Support

If the X86_FEATURE_CPPC CPU feature flag is not set, the processor supports the shared memory solution. In this case, amd-pstate uses the cppc_acpi helper methods to implement the callback functions that are defined on static_call. The functions with the cppc_xxx prefix represent the operations of ACPI CPPC helpers for the shared memory solution.

AMD P-States and ACPI hardware P-States always can be supported in one processor. But AMD P-States has the higher priority and if it is enabled with MSR_AMD_CPPC_ENABLE or cppc_set_enable, it will respond to the request from AMD P-States.

User Space Interface in sysfs - Per-policy control

amd-pstate exposes several global attributes (files) in sysfs to control its functionality at the system level. They are located in the /sys/devices/system/cpu/cpufreq/policyX/ directory and affect all CPUs.

```
root@hr-test1:/home/ray# ls /sys/devices/system/cpu/cpufreq/policy0/*amd*
/sys/devices/system/cpu/cpufreq/policy0/amd_pstate_highest_perf
/sys/devices/system/cpu/cpufreq/policy0/amd_pstate_lowest_nonlinear_freq
/sys/devices/system/cpu/cpufreq/policy0/amd_pstate_max_freq
```

amd_pstate_highest_perf / amd_pstate_max_freq

Maximum CPPC performance and CPU frequency that the driver is allowed to set, in percent of the maximum supported CPPC performance level (the highest performance supported in *AMD CPPC Performance Capability*). In some ASICs, the highest CPPC performance is not the one in the _CPC table, so we need to expose it to sysfs. If boost is not active, but still supported, this maximum frequency will be larger than the one in cpuinfo. This attribute is read-only.

amd_pstate_lowest_nonlinear_freq

The lowest non-linear CPPC CPU frequency that the driver is allowed to set, in percent of the maximum supported CPPC performance level. (Please see the lowest non-linear performance in *AMD CPPC Performance Capability*.) This attribute is read-only.

energy_performance_available_preferences

A list of all the supported EPP preferences that could be used for energy_performance_preference on this system. These profiles represent different hints that are provided to the low-level firmware about the user's desired energy vs efficiency tradeoff. default represents the epp value is set by platform firmware. This attribute is read-only.

energy_performance_preference

The current energy performance preference can be read from this attribute. and user can change current preference according to energy or performance needs Please get all support profiles list from `energy_performance_available_preferences` attribute, all the profiles are integer values defined between 0 to 255 when EPP feature is enabled by platform firmware, if EPP feature is disabled, driver will ignore the written value This attribute is read-write.

Other performance and frequency values can be read back from `/sys/devices/system/cpu/cpuX/acpi_cppc/`, see *CPPC*.

amd-pstate vs acpi-cpufreq

On the majority of AMD platforms supported by `acpi-cpufreq`, the ACPI tables provided by the platform firmware are used for CPU performance scaling, but only provide 3 P-states on AMD processors. However, on modern AMD APU and CPU series, hardware provides the Collaborative Processor Performance Control according to the ACPI protocol and customizes this for AMD platforms. That is, fine-grained and continuous frequency ranges instead of the legacy hardware P-states. `amd-pstate` is the kernel module which supports the new AMD P-States mechanism on most of the future AMD platforms. The AMD P-States mechanism is the more performance and energy efficiency frequency management method on AMD processors.

AMD Pstate Driver Operation Modes

`amd_pstate` CPPC has 3 operation modes: autonomous (active) mode, non-autonomous (passive) mode and guided autonomous (guided) mode. Active/passive/guided mode can be chosen by different kernel parameters.

- In autonomous mode, platform ignores the desired performance level request and takes into account only the values set to the minimum, maximum and energy performance preference registers.
- In non-autonomous mode, platform gets desired performance level from OS directly through Desired Performance Register.
- In guided-autonomous mode, platform sets operating performance level autonomously according to the current workload and within the limits set by OS through min and max performance registers.

Active Mode

`amd_pstate=active`

This is the low-level firmware control mode which is implemented by `amd_pstate_epp` driver with `amd_pstate=active` passed to the kernel in the command line. In this mode, `amd_pstate_epp` driver provides a hint to the hardware if software wants to bias toward performance (0x0) or energy efficiency (0xff) to the CPPC firmware. then CPPC power algorithm will calculate the runtime workload and adjust the realtime cores frequency according to the power supply and thermal, core voltage and some other hardware conditions.

Passive Mode

`amd_pstate=passive`

It will be enabled if the `amd_pstate=passive` is passed to the kernel in the command line. In this mode, `amd_pstate` driver software specifies a desired QoS target in the CPPC performance scale as a relative number. This can be expressed as percentage of nominal performance (infrastructure max). Below the nominal sustained performance level, desired performance expresses the average performance level of the processor subject to the Performance Reduction Tolerance register. Above the nominal performance level, processor must provide at least nominal performance requested and go higher if current operating conditions allow.

Guided Mode

`amd_pstate=guided`

If `amd_pstate=guided` is passed to kernel command line option then this mode is activated. In this mode, driver requests minimum and maximum performance level and the platform autonomously selects a performance level in this range and appropriate to the current workload.

User Space Interface in `sysfs` - General

Global Attributes

`amd-pstate` exposes several global attributes (files) in `sysfs` to control its functionality at the system level. They are located in the `/sys/devices/system/cpu/amd_pstate/` directory and affect all CPUs.

`status`
Operation mode of the driver: "active", "passive" or "disable".

"active"
The driver is functional and in the `active mode`

"passive"
The driver is functional and in the `passive mode`

"guided"
The driver is functional and in the `guided mode`

"disable"
The driver is unregistered and not functional now.

This attribute can be written to in order to change the driver's operation mode or to unregister it. The string written to it must be one of the possible values of it and, if successful, writing one of these values to the sysfs file will cause the driver to switch over to the operation mode represented by that string - or to be unregistered in the "disable" case.

cpupower tool support for amd-pstate

amd-pstate is supported by the cpupower tool, which can be used to dump frequency information. Development is in progress to support more and more operations for the new amd-pstate module with this tool.

```
root@hr-test1:/home/ray# cpupower frequency-info
analyzing CPU 0:
  driver: amd-pstate
  CPUs which run at the same hardware frequency: 0
  CPUs which need to have their frequency coordinated by software: 0
  maximum transition latency: 131 us
  hardware limits: 400 MHz - 4.68 GHz
  available cpufreq governors: ondemand conservative powersave userspace␣
↪performance schedutil
  current policy: frequency should be within 400 MHz and 4.68 GHz.
                  The governor "schedutil" may decide which speed to use
                  within this range.
  current CPU frequency: Unable to call hardware
  current CPU frequency: 4.02 GHz (asserted by call to kernel)
  boost state support:
    Supported: yes
    Active: yes
    AMD PSTATE Highest Performance: 166. Maximum Frequency: 4.68 GHz.
    AMD PSTATE Nominal Performance: 117. Nominal Frequency: 3.30 GHz.
    AMD PSTATE Lowest Non-linear Performance: 39. Lowest Non-linear Frequency:␣
↪1.10 GHz.
    AMD PSTATE Lowest Performance: 15. Lowest Frequency: 400 MHz.
```

Diagnostics and Tuning

Trace Events

There are two static trace events that can be used for amd-pstate diagnostics. One of them is the cpu_frequency trace event generally used by CPUFreq, and the other one is the amd_pstate_perf trace event specific to amd-pstate. The following sequence of shell commands can be used to enable them and see their output (if the kernel is configured to support event tracing).

```
root@hr-test1:/home/ray# cd /sys/kernel/tracing/
root@hr-test1:/sys/kernel/tracing# echo 1 > events/amd_cpu/enable
root@hr-test1:/sys/kernel/tracing# cat trace
# tracer: nop
#
# entries-in-buffer/entries-written: 47827/42233061   #P:2
#
#                                _-----=> irqs-off
#                               / _----=> need-resched
#                              | / _---=> hardirq/softirq
#                              || / _--=> preempt-depth
```

```
#                                  ||| /     delay
#           TASK-PID     CPU#  ||||   TIMESTAMP  FUNCTION
#              | |         |   ||||      |         |
         <idle>-0       [015] dN...  4995.979886: amd_pstate_perf: amd_min_
↪perf=85 amd_des_perf=85 amd_max_perf=166 cpu_id=15 changed=false fast_
↪switch=true
         <idle>-0       [007] d.h..  4995.979893: amd_pstate_perf: amd_min_
↪perf=85 amd_des_perf=85 amd_max_perf=166 cpu_id=7 changed=false fast_
↪switch=true
            cat-2161    [000] d....  4995.980841: amd_pstate_perf: amd_min_
↪perf=85 amd_des_perf=85 amd_max_perf=166 cpu_id=0 changed=false fast_
↪switch=true
           sshd-2125    [004] d.s..  4995.980968: amd_pstate_perf: amd_min_
↪perf=85 amd_des_perf=85 amd_max_perf=166 cpu_id=4 changed=false fast_
↪switch=true
         <idle>-0       [007] d.s..  4995.980968: amd_pstate_perf: amd_min_
↪perf=85 amd_des_perf=85 amd_max_perf=166 cpu_id=7 changed=false fast_
↪switch=true
         <idle>-0       [003] d.s..  4995.980971: amd_pstate_perf: amd_min_
↪perf=85 amd_des_perf=85 amd_max_perf=166 cpu_id=3 changed=false fast_
↪switch=true
         <idle>-0       [011] d.s..  4995.980996: amd_pstate_perf: amd_min_
↪perf=85 amd_des_perf=85 amd_max_perf=166 cpu_id=11 changed=false fast_
↪switch=true
```

The `cpu_frequency` trace event will be triggered either by the `schedutil` scaling governor (for the policies it is attached to), or by the `CPUFreq` core (for the policies with other scaling governors).

Tracer Tool

`amd_pstate_tracer.py` can record and parse `amd-pstate` trace log, then generate performance plots. This utility can be used to debug and tune the performance of `amd-pstate` driver. The tracer tool needs to import intel pstate tracer.

Tracer tool located in `linux/tools/power/x86/amd_pstate_tracer`. It can be used in two ways. If trace file is available, then directly parse the file with command

```
./amd_pstate_trace.py [-c cpus] -t <trace_file> -n <test_name>
```

Or generate trace file with root privilege, then parse and plot with command

```
sudo ./amd_pstate_trace.py [-c cpus] -n <test_name> -i <interval> [-m kbytes]
```

The test result can be found in `results/test_name`. Following is the example about part of the output.

```
common_cpu  common_secs  common_usecs  min_perf  des_perf  max_perf  freq   ␣
↪mperf   apef    tsc       load   duration_ms  sample_num  elapsed_time ␣
↪common_comm
CPU_005     712          116384        39        49        166       0.7565 ␣
↪9645075 2214891 38431470  25.1   11.646       469         2.496          ␣
↪kworker/5:0-40
```

```
CPU_006     712          116408        39        49        166       0.6769
↪8950227 1839034 37192089  24.06  11.272       470         2.496
↪kworker/6:0-1264
```

Unit Tests for amd-pstate

`amd-pstate-ut` is a test module for testing the `amd-pstate` driver.

- It can help all users to verify their processor support (SBIOS/Firmware or Hardware).
- Kernel can have a basic function test to avoid the kernel regression during the update.
- We can introduce more functional or performance tests to align the result together, it will benefit power and performance scale optimization.

1. Test case descriptions

   1). Basic tests

   Test prerequisite and basic functions for the `amd-pstate` driver.

Index	Functions	Description
1	amd_pstate_ut_acpi_cpc_valid	Check whether the _CPC object is present in SBIOS. The detail refer to *Processor Support*.
2	amd_pstate_ut_check_enabled	Check whether AMD P-State is enabled. AMD P-States and ACPI hardware P-States always can be supported in one processor. But AMD P-States has the higher priority and if it is enabled with `MSR_AMD_CPPC_ENABLE` or `cppc_set_enable`, it will respond to the request from AMD P-States.
3	amd_pstate_ut_check_perf	Check if the each performance values are reasonable. highest_perf >= nominal_perf > lowest_nonlinear_perf > lowest_perf > 0.
4	amd_pstate_ut_check_freq	Check if the each frequency values and max freq when set support boost mode are reasonable. max_freq >= nominal_freq > lowest_nonlinear_freq > min_freq > 0 If boost is not active but supported, this maximum frequency will be larger than the one in `cpuinfo`.

2). Tbench test

Test and monitor the cpu changes when running tbench benchmark under the specified governor. These changes include desire performance, frequency, load, performance, energy etc. The specified governor is ondemand or schedutil. Tbench can also be tested on the `acpi-cpufreq` kernel driver for comparison.

3). Gitsource test

Test and monitor the cpu changes when running gitsource benchmark under the specified governor. These changes include desire performance, frequency, load, time, energy etc. The specified governor is ondemand or schedutil. Gitsource can also be tested on the `acpi-cpufreq` kernel driver for comparison.

2. How to execute the tests

We use test module in the kselftest frameworks to implement it. We create `amd-pstate-ut` module and tie it into kselftest.(for details refer to Linux Kernel Selftests[4]).

1). Build

- open the `CONFIG_X86_AMD_PSTATE` configuration option.
- set the `CONFIG_X86_AMD_PSTATE_UT` configuration option to M.
- make project
- make selftest

```
$ cd linux
$ make -C tools/testing/selftests
```

- make perf

```
$ cd tools/perf/
$ make
```

2). Installation & Steps

```
$ make -C tools/testing/selftests install INSTALL_PATH=~/kselftest
$ cp tools/perf/perf /usr/bin/perf
$ sudo ./kselftest/run_kselftest.sh -c amd-pstate
```

3). Specified test case

```
$ cd ~/kselftest/amd-pstate
$ sudo ./run.sh -t basic
$ sudo ./run.sh -t tbench
$ sudo ./run.sh -t tbench -m acpi-cpufreq
$ sudo ./run.sh -t gitsource
$ sudo ./run.sh -t gitsource -m acpi-cpufreq
$ ./run.sh --help
./run.sh: illegal option -- -
```

[4] Linux Kernel Selftests, https://www.kernel.org/doc/html/latest/dev-tools/kselftest.html

```
Usage: ./run.sh [OPTION...]
        [-h <help>]
        [-o <output-file-for-dump>]
        [-c <all: All testing,
             basic: Basic testing,
             tbench: Tbench testing,
             gitsource: Gitsource testing.>]
        [-t <tbench time limit>]
        [-p <tbench process number>]
        [-l <loop times for tbench>]
        [-i <amd tracer interval>]
        [-m <comparative test: acpi-cpufreq>]
```

4). Results

- basic

  When you finish test, you will get the following log info

```
$ dmesg | grep "amd_pstate_ut" | tee log.txt
[12977.570663] amd_pstate_ut: 1    amd_pstate_ut_acpi_
↪cpc_valid  success!
[12977.570673] amd_pstate_ut: 2    amd_pstate_ut_check_
↪enabled   success!
[12977.571207] amd_pstate_ut: 3    amd_pstate_ut_check_
↪perf      success!
[12977.571212] amd_pstate_ut: 4    amd_pstate_ut_check_
↪freq      success!
```

- tbench

  When you finish test, you will get selftest.tbench.csv and png images. The selftest.tbench.csv file contains the raw data and the drop of the comparative test. The png images shows the performance, energy and performan per watt of each test. Open selftest.tbench.csv :

Governor	Round	Desperf	Freq	Load	Performance	Energy	Performance Per Watt
Unit			GHz		MB/s	J	MB/J
amd-pstate-ondemand	1				2504.05	1563.67	158.5378
amd-pstate-ondemand	2				2243.64	1430.32	155.2941

continues on next page

Table 1 – continued from previous page

Governor	Round	Desperf	Freq	Load	Performance	Energy	Performance Per Watt
Unit			GHz		MB/s	J	MB/J
amd-pstate-ondemand	3				2183.88	1401.32	154.2860
amd-pstate-ondemand	Average				2310.52	1465.1	156.1268
amd-pstate-schedutil	1	165.329	1.62257	99.798	2136.54	1395.26	151.5971
amd-pstate-schedutil	2	166	1.49761	99.9993	2100.56	1380.5	150.6377
amd-pstate-schedutil	3	166	1.47806	99.9993	2084.12	1375.76	149.9737
amd-pstate-schedutil	Average	165.776	1.53275	99.9322	2107.07	1383.84	150.7399
acpi-cpufreq-ondemand	1				2529.9	1564.4	160.0997
acpi-cpufreq-ondemand	2				2249.76	1432.97	155.4297
acpi-cpufreq-ondemand	3				2181.46	1406.88	153.5060
acpi-cpufreq-ondemand	Average				2320.37	1468.08	156.4741
acpi-cpufreq-schedutil	1				2137.64	1385.24	152.7723
acpi-cpufreq-schedutil	2				2107.05	1372.23	152.0138
acpi-cpufreq-schedutil	3				2085.86	1365.35	151.2433
acpi-cpufreq-schedutil	Average				2110.18	1374.27	152.0136

continues on next page

Table 1 – continued from previous page

Governor	Round	Desperf	Freq	Load	Performance	Energy	Performance Per Watt
Unit			GHz		MB/s	J	MB/J
acpi-cpufreq-ondemand VS acpi-cpufreq-schedutil	Comprison(%)				-9.0584	-6.3899	-2.8506
amd-pstate-ondemand VS amd-pstate-schedutil	Comprison(%)				8.8053	-5.5463	-3.4503
acpi-cpufreq-ondemand VS amd-pstate-ondemand	Comprison(%)				-0.4245	-0.2029	-0.2219
acpi-cpufreq-schedutil VS amd-pstate-schedutil	Comprison(%)				-0.1473	0.6963	-0.8378

- gitsource

  When you finish test, you will get selftest.gitsource.csv and png images. The selftest.gitsource.csv file contains the raw data and the drop of the comparative test. The png images shows the performance, energy and performan per watt of each test. Open selftest.gitsource.csv :

Governor	Round	Desperf	Freq	Load	Time	Energy	Performance Per Watt
Unit			GHz		s	J	1/J
amd-pstate-ondemand	1	50.119	2.10509	23.3076	475.69	865.78	0.001155027
amd-pstate-ondemand	2	94.8006	1.98771	56.6533	467.1	839.67	0.001190944
amd-pstate-ondemand	3	76.6091	2.53251	43.7791	467.69	855.85	0.001168429
amd-pstate-ondemand	Average	73.8429	2.20844	41.2467	470.16	853.767	0.001171279
amd-pstate-schedutil	1	165.919	1.62319	98.3868	464.17	866.8	0.001153668
amd-pstate-schedutil	2	165.97	1.31309	99.5712	480.15	880.4	0.001135847
amd-pstate-schedutil	3	165.973	1.28448	99.9252	481.79	867.02	0.001153375
amd-pstate-schedutil	Average	165.954	1.40692	99.2944	475.37	871.407	0.001147569
acpi-cpufreq-ondemand	1				2379.62	742.96	0.001345967
acpi-cpufreq-ondemand	2				441.74	817.49	0.001223256
acpi-cpufreq-ondemand	3				455.48	820.01	0.001219497
acpi-cpufreq-ondemand	Average				425.613	793.487	0.001260260
acpi-cpufreq-schedutil	1				459.69	838.54	0.001192548
acpi-cpufreq-schedutil	2				466.55	830.89	0.001203528

continues on next page

Table 2 – continued from previous page

Governor	Round	Desperf	Freq	Load	Time	Energy	Performance Per Watt
Unit			GHz		s	J	1/J
acpi-cpufreq-schedutil	3				470.38	837.32	0.001194286
acpi-cpufreq-schedutil	Average				465.54	835.583	0.001196769
acpi-cpufreq-ondemand VS acpi-cpufreq-schedutil	Comparison(%)				9.3810	5.3051	-5.0379
amd-pstate-ondemand VS amd-pstate-schedutil	Comparison(%)	124.7392	36.2934	140.7329	1.1081	2.0661	-2.0242
acpi-cpufreq-ondemand VS amd-pstate-ondemand	Comparison(%)				10.4665	7.5968	-7.0605
acpi-cpufreq-schedutil VS amd-pstate-schedutil	Comparison(%)				2.1115	4.2873	-4.1110

Reference

67.3.6 Legacy Documentation of CPU Performance Scaling Drivers

Included below are historic documents describing assorted *CPU performance scaling* drivers. They are reproduced verbatim, with the original white space formatting and indentation preserved, except for the added leading space character in every line of text.

AMD PowerNow! Drivers

```
PowerNow! and Cool'n'Quiet are AMD names for frequency
management capabilities in AMD processors. As the hardware
implementation changes in new generations of the processors,
there is a different cpu-freq driver for each generation.

Note that the driver's will not load on the "wrong" hardware,
so it is safe to try each driver in turn when in doubt as to
which is the correct driver.

Note that the functionality to change frequency (and voltage)
is not available in all processors. The drivers will refuse
to load on processors without this capability. The capability
is detected with the cpuid instruction.

The drivers use BIOS supplied tables to obtain frequency and
voltage information appropriate for a particular platform.
Frequency transitions will be unavailable if the BIOS does
not supply these tables.

6th Generation: powernow-k6

7th Generation: powernow-k7: Athlon, Duron, Geode.

8th Generation: powernow-k8: Athlon, Athlon 64, Opteron, Sempron.
Documentation on this functionality in 8th generation processors
is available in the "BIOS and Kernel Developer's Guide", publication
26094, in chapter 9, available for download from www.amd.com.

BIOS supplied data, for powernow-k7 and for powernow-k8, may be
from either the PSB table or from ACPI objects. The ACPI support
is only available if the kernel config sets CONFIG_ACPI_PROCESSOR.
The powernow-k8 driver will attempt to use ACPI if so configured,
and fall back to PST if that fails.
The powernow-k7 driver will try to use the PSB support first, and
fall back to ACPI if the PSB support fails. A module parameter,
acpi_force, is provided to force ACPI support to be used instead
of PSB support.
```

cpufreq-nforce2

```
The cpufreq-nforce2 driver changes the FSB on nVidia nForce2 platforms.

This works better than on other platforms, because the FSB of the CPU
can be controlled independently from the PCI/AGP clock.

The module has two options:

       fid:     multiplier * 10 (for example 8.5 = 85)
       min_fsb: minimum FSB

If not set, fid is calculated from the current CPU speed and the FSB.
min_fsb defaults to FSB at boot time - 50 MHz.

IMPORTANT: The available range is limited downwards!
           Also the minimum available FSB can differ, for systems
           booting with 200 MHz, 150 should always work.
```

pcc-cpufreq

```
/*
 *  pcc-cpufreq.txt - PCC interface documentation
 *
 *  Copyright (C) 2009 Red Hat, Matthew Garrett <mjg@redhat.com>
 *  Copyright (C) 2009 Hewlett-Packard Development Company, L.P.
 *      Nagananda Chumbalkar <nagananda.chumbalkar@hp.com>
 */


                       Processor Clocking Control Driver
                       ---------------------------------

Contents:
---------
1.     Introduction
1.1    PCC interface
1.1.1  Get Average Frequency
1.1.2  Set Desired Frequency
1.2    Platforms affected
2.     Driver and /sys details
2.1    scaling_available_frequencies
2.2    cpuinfo_transition_latency
2.3    cpuinfo_cur_freq
2.4    related_cpus
3.     Caveats

1. Introduction:
----------------
```

```
Processor Clocking Control (PCC) is an interface between the platform
firmware and OSPM. It is a mechanism for coordinating processor
performance (ie: frequency) between the platform firmware and the OS.

The PCC driver (pcc-cpufreq) allows OSPM to take advantage of the PCC
interface.

OS utilizes the PCC interface to inform platform firmware what frequency the
OS wants for a logical processor. The platform firmware attempts to achieve
the requested frequency. If the request for the target frequency could not be
satisfied by platform firmware, then it usually means that power budget
conditions are in place, and "power capping" is taking place.

1.1 PCC interface:
------------------
The complete PCC specification is available here:
https://acpica.org/sites/acpica/files/Processor-Clocking-Control-v1p0.pdf

PCC relies on a shared memory region that provides a channel for communication
between the OS and platform firmware. PCC also implements a "doorbell" that
is used by the OS to inform the platform firmware that a command has been
sent.

The ACPI PCCH() method is used to discover the location of the PCC shared
memory region. The shared memory region header contains the "command" and
"status" interface. PCCH() also contains details on how to access the platform
doorbell.

The following commands are supported by the PCC interface:
* Get Average Frequency
* Set Desired Frequency

The ACPI PCCP() method is implemented for each logical processor and is
used to discover the offsets for the input and output buffers in the shared
memory region.

When PCC mode is enabled, the platform will not expose processor performance
or throttle states (_PSS, _TSS and related ACPI objects) to OSPM. Therefore,
the native P-state driver (such as acpi-cpufreq for Intel, powernow-k8 for
AMD) will not load.

However, OSPM remains in control of policy. The governor (eg: "ondemand")
computes the required performance for each processor based on server workload.
The PCC driver fills in the command interface, and the input buffer and
communicates the request to the platform firmware. The platform firmware is
responsible for delivering the requested performance.

Each PCC command is "global" in scope and can affect all the logical CPUs in
the system. Therefore, PCC is capable of performing "group" updates. With PCC
the OS is capable of getting/setting the frequency of all the logical CPUs in
```

```
the system with a single call to the BIOS.

1.1.1 Get Average Frequency:
----------------------------
This command is used by the OSPM to query the running frequency of the
processor since the last time this command was completed. The output buffer
indicates the average unhalted frequency of the logical processor expressed as
a percentage of the nominal (ie: maximum) CPU frequency. The output buffer
also signifies if the CPU frequency is limited by a power budget condition.

1.1.2 Set Desired Frequency:
----------------------------
This command is used by the OSPM to communicate to the platform firmware the
desired frequency for a logical processor. The output buffer is currently
ignored by OSPM. The next invocation of "Get Average Frequency" will inform
OSPM if the desired frequency was achieved or not.

1.2 Platforms affected:
-----------------------
The PCC driver will load on any system where the platform firmware:
* supports the PCC interface, and the associated PCCH() and PCCP() methods
* assumes responsibility for managing the hardware clocking controls in order
to deliver the requested processor performance

Currently, certain HP ProLiant platforms implement the PCC interface. On those
platforms PCC is the "default" choice.

However, it is possible to disable this interface via a BIOS setting. In
such an instance, as is also the case on platforms where the PCC interface
is not implemented, the PCC driver will fail to load silently.

2. Driver and /sys details:
---------------------------
When the driver loads, it merely prints the lowest and the highest CPU
frequencies supported by the platform firmware.

The PCC driver loads with a message such as:
pcc-cpufreq: (v1.00.00) driver loaded with frequency limits: 1600 MHz, 2933
MHz

This means that the OPSM can request the CPU to run at any frequency in
between the limits (1600 MHz, and 2933 MHz) specified in the message.

Internally, there is no need for the driver to convert the "target" frequency
to a corresponding P-state.

The VERSION number for the driver will be of the format v.xy.ab.
eg: 1.00.02
   ----- --
    |    |
```

```
    |    -- this will increase with bug fixes/enhancements to the driver
    |-- this is the version of the PCC specification the driver adheres to


The following is a brief discussion on some of the fields exported via the
/sys filesystem and how their values are affected by the PCC driver:

2.1 scaling_available_frequencies:
----------------------------------
scaling_available_frequencies is not created in /sys. No intermediate
frequencies need to be listed because the BIOS will try to achieve any
frequency, within limits, requested by the governor. A frequency does not have
to be strictly associated with a P-state.

2.2 cpuinfo_transition_latency:
-------------------------------
The cpuinfo_transition_latency field is 0. The PCC specification does
not include a field to expose this value currently.

2.3 cpuinfo_cur_freq:
---------------------
A) Often cpuinfo_cur_freq will show a value different than what is declared
in the scaling_available_frequencies or scaling_cur_freq, or scaling_max_freq.
This is due to "turbo boost" available on recent Intel processors. If certain
conditions are met the BIOS can achieve a slightly higher speed than requested
by OSPM. An example:

scaling_cur_freq       : 2933000
cpuinfo_cur_freq       : 3196000

B) There is a round-off error associated with the cpuinfo_cur_freq value.
Since the driver obtains the current frequency as a "percentage" (%) of the
nominal frequency from the BIOS, sometimes, the values displayed by
scaling_cur_freq and cpuinfo_cur_freq may not match. An example:

scaling_cur_freq       : 1600000
cpuinfo_cur_freq       : 1583000

In this example, the nominal frequency is 2933 MHz. The driver obtains the
current frequency, cpuinfo_cur_freq, as 54% of the nominal frequency:

       54% of 2933 MHz = 1583 MHz

Nominal frequency is the maximum frequency of the processor, and it usually
corresponds to the frequency of the P0 P-state.

2.4 related_cpus:
-----------------
The related_cpus field is identical to affected_cpus.
```

```
affected_cpus  : 4
related_cpus   : 4

Currently, the PCC driver does not evaluate _PSD. The platforms that support
PCC do not implement SW_ALL. So OSPM doesn't need to perform any coordination
to ensure that the same frequency is requested of all dependent CPUs.

3. Caveats:
-----------
The "cpufreq_stats" module in its present form cannot be loaded and
expected to work with the PCC driver. Since the "cpufreq_stats" module
provides information wrt each P-state, it is not applicable to the PCC driver.
```

67.3.7 Intel Performance and Energy Bias Hint

Copyright
© 2019 Intel Corporation

Author
Rafael J. Wysocki <rafael.j.wysocki@intel.com>

The Performance and Energy Bias Hint (EPB) allows software to specify its preference with respect to the power-performance tradeoffs present in the processor. Generally, the EPB is expected to be set by user space (directly via sysfs or with the help of the x86_energy_perf_policy tool), but there are two reasons for the kernel to update it.

First, there are systems where the platform firmware resets the EPB during system-wide transitions from sleep states back into the working state effectively causing the previous EPB updates by user space to be lost. Thus the kernel needs to save the current EPB values for all CPUs during system-wide transitions to sleep states and restore them on the way back to the working state. That can be achieved by saving EPB for secondary CPUs when they are taken offline during transitions into system sleep states and for the boot CPU in a syscore suspend operation, so that it can be restored for the boot CPU in a syscore resume operation and for the other CPUs when they are brought back online. However, CPUs that are already offline when a system-wide PM transition is started are not taken offline again, but their EPB values may still be reset by the platform firmware during the transition, so in fact it is necessary to save the EPB of any CPU taken offline and to restore it when the given CPU goes back online at all times.

Second, on many systems the initial EPB value coming from the platform firmware is 0 ('performance') and at least on some of them that is because the platform firmware does not initialize EPB at all with the assumption that the OS will do that anyway. That sometimes is problematic, as it may cause the system battery to drain too fast, for example, so it is better to adjust it on CPU bring-up and if the initial EPB value for a given CPU is 0, the kernel changes it to 6 ('normal').

Intel Performance and Energy Bias Attribute in sysfs

The Intel Performance and Energy Bias Hint (EPB) value for a given (logical) CPU can be checked or updated through a `sysfs` attribute (file) under `/sys/devices/system/cpu/cpu<N>/power/`, where the CPU number `<N>` is allocated at the system initialization time:

energy_perf_bias

Shows the current EPB value for the CPU in a sliding scale 0 - 15, where a value of 0 corresponds to a hint preference for highest performance and a value of 15 corresponds to the maximum energy savings.

In order to update the EPB value for the CPU, this attribute can be written to, either with a number in the 0 - 15 sliding scale above, or with one of the strings: "performance", "balance-performance", "normal", "balance-power", "power" that represent values reflected by their meaning.

This attribute is present for all online CPUs supporting the EPB feature.

Note that while the EPB interface to the processor is defined at the logical CPU level, the physical register backing it may be shared by multiple CPUs (for example, SMT siblings or cores in one package). For this reason, updating the EPB value for one CPU may cause the EPB values for other CPUs to change.

67.3.8 Intel(R) Speed Select Technology User Guide

The Intel(R) Speed Select Technology (Intel(R) SST) provides a powerful new collection of features that give more granular control over CPU performance. With Intel(R) SST, one server can be configured for power and performance for a variety of diverse workload requirements.

Refer to the links below for an overview of the technology:

- https://www.intel.com/content/www/us/en/architecture-and-technology/speed-select-technology-article.html
- https://builders.intel.com/docs/networkbuilders/intel-speed-select-technology-base-frequency-en pdf

These capabilities are further enhanced in some of the newer generations of server platforms where these features can be enumerated and controlled dynamically without pre-configuring via BIOS setup options. This dynamic configuration is done via mailbox commands to the hardware. One way to enumerate and configure these features is by using the Intel Speed Select utility.

This document explains how to use the Intel Speed Select tool to enumerate and control Intel(R) SST features. This document gives example commands and explains how these commands change the power and performance profile of the system under test. Using this tool as an example, customers can replicate the messaging implemented in the tool in their production software.

intel-speed-select configuration tool

Most Linux distribution packages may include the "intel-speed-select" tool. If not, it can be built by downloading the Linux kernel tree from kernel.org. Once downloaded, the tool can be built without building the full kernel.

From the kernel tree, run the following commands:

```
# cd tools/power/x86/intel-speed-select/
# make
# make install
```

Getting Help

To get help with the tool, execute the command below:

```
# intel-speed-select --help
```

The top-level help describes arguments and features. Notice that there is a multi-level help structure in the tool. For example, to get help for the feature "perf-profile":

```
# intel-speed-select perf-profile --help
```

To get help on a command, another level of help is provided. For example for the command info "info":

```
# intel-speed-select perf-profile info --help
```

Summary of platform capability

To check the current platform and driver capabilities, execute:

```
#intel-speed-select --info
```

For example on a test system:

```
# intel-speed-select --info
Intel(R) Speed Select Technology
Executing on CPU model: X
Platform: API version : 1
Platform: Driver version : 1
Platform: mbox supported : 1
Platform: mmio supported : 1
Intel(R) SST-PP (feature perf-profile) is supported
TDP level change control is unlocked, max level: 4
Intel(R) SST-TF (feature turbo-freq) is supported
Intel(R) SST-BF (feature base-freq) is not supported
Intel(R) SST-CP (feature core-power) is supported
```

Intel(R) Speed Select Technology - Performance Profile (Intel(R) SST-PP)

This feature allows configuration of a server dynamically based on workload performance requirements. This helps users during deployment as they do not have to choose a specific server configuration statically. This Intel(R) Speed Select Technology - Performance Profile (Intel(R) SST-PP) feature introduces a mechanism that allows multiple optimized performance profiles per system. Each profile defines a set of CPUs that need to be online and rest offline to sustain a guaranteed base frequency. Once the user issues a command to use a specific performance profile and meet CPU online/offline requirement, the user can expect a change in the base frequency dynamically. This feature is called ”perf-profile” when using the Intel Speed Select tool.

Number or performance levels

There can be multiple performance profiles on a system. To get the number of profiles, execute the command below:

```
# intel-speed-select perf-profile get-config-levels
Intel(R) Speed Select Technology
Executing on CPU model: X
package-0
 die-0
   cpu-0
       get-config-levels:4
package-1
 die-0
   cpu-14
       get-config-levels:4
```

On this system under test, there are 4 performance profiles in addition to the base performance profile (which is performance level 0).

Lock/Unlock status

Even if there are multiple performance profiles, it is possible that they are locked. If they are locked, users cannot issue a command to change the performance state. It is possible that there is a BIOS setup to unlock or check with your system vendor.

To check if the system is locked, execute the following command:

```
# intel-speed-select perf-profile get-lock-status
Intel(R) Speed Select Technology
Executing on CPU model: X
package-0
 die-0
   cpu-0
       get-lock-status:0
package-1
 die-0
```

```
   cpu-14
       get-lock-status:0
```

In this case, lock status is 0, which means that the system is unlocked.

Properties of a performance level

To get properties of a specific performance level (For example for the level 0, below), execute the command below:

```
# intel-speed-select perf-profile info -l 0
Intel(R) Speed Select Technology
Executing on CPU model: X
package-0
 die-0
   cpu-0
     perf-profile-level-0
       cpu-count:28
       enable-cpu-mask:000003ff,f0003fff
       enable-cpu-list:0,1,2,3,4,5,6,7,8,9,10,11,12,13,28,29,30,31,32,33,34,35,
↪36,37,38,39,40,41
       thermal-design-power-ratio:26
       base-frequency(MHz):2600
       speed-select-turbo-freq:disabled
       speed-select-base-freq:disabled
       ...
       ...
```

Here -l option is used to specify a performance level.

If the option -l is omitted, then this command will print information about all the performance levels. The above command is printing properties of the performance level 0.

For this performance profile, the list of CPUs displayed by the ”enable-cpu-mask/enable-cpu-list” at the max can be ”online.” When that condition is met, then base frequency of 2600 MHz can be maintained. To understand more, execute ”intel-speed-select perf-profile info” for performance level 4:

```
# intel-speed-select perf-profile info -l 4
Intel(R) Speed Select Technology
Executing on CPU model: X
package-0
 die-0
   cpu-0
     perf-profile-level-4
       cpu-count:28
       enable-cpu-mask:000000fa,f0000faf
       enable-cpu-list:0,1,2,3,5,7,8,9,10,11,28,29,30,31,33,35,36,37,38,39
       thermal-design-power-ratio:28
       base-frequency(MHz):2800
       speed-select-turbo-freq:disabled
```

```
        speed-select-base-freq:unsupported
        ...
        ...
```

There are fewer CPUs in the "enable-cpu-mask/enable-cpu-list". Consequently, if the user only keeps these CPUs online and the rest "offline," then the base frequency is increased to 2.8 GHz compared to 2.6 GHz at performance level 0.

Get current performance level

To get the current performance level, execute:

```
# intel-speed-select perf-profile get-config-current-level
Intel(R) Speed Select Technology
Executing on CPU model: X
package-0
 die-0
   cpu-0
       get-config-current_level:0
```

First verify that the base_frequency displayed by the cpufreq sysfs is correct:

```
# cat /sys/devices/system/cpu/cpu0/cpufreq/base_frequency
2600000
```

This matches the base-frequency (MHz) field value displayed from the "perf-profile info" command for performance level 0(cpufreq frequency is in KHz).

To check if the average frequency is equal to the base frequency for a 100% busy workload, disable turbo:

```
# echo 1 > /sys/devices/system/cpu/intel_pstate/no_turbo
```

Then runs a busy workload on all CPUs, for example:

```
#stress -c 64
```

To verify the base frequency, run turbostat:

```
#turbostat -c 0-13 --show Package,Core,CPU,Bzy_MHz -i 1

 Package       Core    CPU     Bzy_MHz
               -       -       2600
 0             0       0       2600
 0             1       1       2600
 0             2       2       2600
 0             3       3       2600
 0             4       4       2600
 .             .       .       .
```

Changing performance level

To the change the performance level to 4, execute:

```
# intel-speed-select -d perf-profile set-config-level -l 4 -o
Intel(R) Speed Select Technology
Executing on CPU model: X
package-0
 die-0
   cpu-0
     perf-profile
       set_tdp_level:success
```

In the command above, ”-o” is optional. If it is specified, then it will also offline CPUs which are not present in the enable_cpu_mask for this performance level.

Now if the base_frequency is checked:

```
#cat /sys/devices/system/cpu/cpu0/cpufreq/base_frequency
2800000
```

Which shows that the base frequency now increased from 2600 MHz at performance level 0 to 2800 MHz at performance level 4. As a result, any workload, which can use fewer CPUs, can see a boost of 200 MHz compared to performance level 0.

Changing performance level via BMC Interface

It is possible to change SST-PP level using out of band (OOB) agent (Via some remote management console, through BMC ”Baseboard Management Controller” interface). This mode is supported from the Sapphire Rapids processor generation. The kernel and tool change to support this mode is added to Linux kernel version 5.18. To enable this feature, kernel config ”CONFIG_INTEL_HFI_THERMAL” is required. The minimum version of the tool is ”v1.12” to support this feature, which is part of Linux kernel version 5.18.

To support such configuration, this tool can be used as a daemon. Add a command line option --oob:

```
# intel-speed-select --oob
Intel(R) Speed Select Technology
Executing on CPU model:143[0x8f]
OOB mode is enabled and will run as daemon
```

In this mode the tool will online/offline CPUs based on the new performance level.

Check presence of other Intel(R) SST features

Each of the performance profiles also specifies weather there is support of other two Intel(R) SST features (Intel(R) Speed Select Technology - Base Frequency (Intel(R) SST-BF) and Intel(R) Speed Select Technology - Turbo Frequency (Intel SST-TF)).

For example, from the output of ”perf-profile info” above, for level 0 and level 4:

For level 0::
speed-select-turbo-freq:disabled speed-select-base-freq:disabled

For level 4::
speed-select-turbo-freq:disabled speed-select-base-freq:unsupported

Given these results, the ”speed-select-base-freq” (Intel(R) SST-BF) in level 4 changed from ”disabled” to ”unsupported” compared to performance level 0.

This means that at performance level 4, the ”speed-select-base-freq” feature is not supported. However, at performance level 0, this feature is ”supported”, but currently ”disabled”, meaning the user has not activated this feature. Whereas ”speed-select-turbo-freq” (Intel(R) SST-TF) is supported at both performance levels, but currently not activated by the user.

The Intel(R) SST-BF and the Intel(R) SST-TF features are built on a foundation technology called Intel(R) Speed Select Technology - Core Power (Intel(R) SST-CP). The platform firmware enables this feature when Intel(R) SST-BF or Intel(R) SST-TF is supported on a platform.

Intel(R) Speed Select Technology Core Power (Intel(R) SST-CP)

Intel(R) Speed Select Technology Core Power (Intel(R) SST-CP) is an interface that allows users to define per core priority. This defines a mechanism to distribute power among cores when there is a power constrained scenario. This defines a class of service (CLOS) configuration.

The user can configure up to 4 class of service configurations. Each CLOS group configuration allows definitions of parameters, which affects how the frequency can be limited and power is distributed. Each CPU core can be tied to a class of service and hence an associated priority. The granularity is at core level not at per CPU level.

Enable CLOS based prioritization

To use CLOS based prioritization feature, firmware must be informed to enable and use a priority type. There is a default per platform priority type, which can be changed with optional command line parameter.

To enable and check the options, execute:

```
# intel-speed-select core-power enable --help
Intel(R) Speed Select Technology
Executing on CPU model: X
Enable core-power for a package/die
       Clos Enable: Specify priority type with [--priority|-p]
                0: Proportional, 1: Ordered
```

There are two types of priority types:

- Ordered

Priority for ordered throttling is defined based on the index of the assigned CLOS group. Where CLOS0 gets highest priority (throttled last).

Priority order is: CLOS0 > CLOS1 > CLOS2 > CLOS3.

- Proportional

When proportional priority is used, there is an additional parameter called frequency_weight, which can be specified per CLOS group. The goal of proportional priority is to provide each core with the requested min., then distribute all remaining (excess/deficit) budgets in proportion to a defined weight. This proportional priority can be configured using "core-power config" command.

To enable with the platform default priority type, execute:

```
# intel-speed-select core-power enable
Intel(R) Speed Select Technology
Executing on CPU model: X
package-0
 die-0
   cpu-0
     core-power
       enable:success
package-1
 die-0
   cpu-6
     core-power
       enable:success
```

The scope of this enable is per package or die scoped when a package contains multiple dies. To check if CLOS is enabled and get priority type, "core-power info" command can be used. For example to check the status of core-power feature on CPU 0, execute:

```
# intel-speed-select -c 0 core-power info
Intel(R) Speed Select Technology
Executing on CPU model: X
package-0
 die-0
   cpu-0
     core-power
       support-status:supported
       enable-status:enabled
       clos-enable-status:enabled
       priority-type:proportional
package-1
 die-0
   cpu-24
     core-power
       support-status:supported
       enable-status:enabled
       clos-enable-status:enabled
       priority-type:proportional
```

Configuring CLOS groups

Each CLOS group has its own attributes including min, max, freq_weight and desired. These parameters can be configured with "core-power config" command. Defaults will be used if user skips setting a parameter except clos id, which is mandatory. To check core-power config options, execute:

```
# intel-speed-select core-power config --help
Intel(R) Speed Select Technology
Executing on CPU model: X
Set core-power configuration for one of the four clos ids
       Specify targeted clos id with [--clos|-c]
       Specify clos Proportional Priority [--weight|-w]
       Specify clos min in MHz with [--min|-n]
       Specify clos max in MHz with [--max|-m]
```

For example:

```
# intel-speed-select core-power config -c 0
Intel(R) Speed Select Technology
Executing on CPU model: X
clos epp is not specified, default: 0
clos frequency weight is not specified, default: 0
clos min is not specified, default: 0 MHz
clos max is not specified, default: 25500 MHz
clos desired is not specified, default: 0
package-0
 die-0
   cpu-0
     core-power
       config:success
package-1
 die-0
   cpu-6
     core-power
       config:success
```

The user has the option to change defaults. For example, the user can change the "min" and set the base frequency to always get guaranteed base frequency.

Get the current CLOS configuration

To check the current configuration, "core-power get-config" can be used. For example, to get the configuration of CLOS 0:

```
# intel-speed-select core-power get-config -c 0
Intel(R) Speed Select Technology
Executing on CPU model: X
package-0
 die-0
   cpu-0
```

```
     core-power
       clos:0
       epp:0
       clos-proportional-priority:0
       clos-min:0 MHz
       clos-max:Max Turbo frequency
       clos-desired:0 MHz
package-1
 die-0
   cpu-24
     core-power
       clos:0
       epp:0
       clos-proportional-priority:0
       clos-min:0 MHz
       clos-max:Max Turbo frequency
       clos-desired:0 MHz
```

Associating a CPU with a CLOS group

To associate a CPU to a CLOS group "core-power assoc" command can be used:

```
# intel-speed-select core-power assoc --help
Intel(R) Speed Select Technology
Executing on CPU model: X
Associate a clos id to a CPU
       Specify targeted clos id with [--clos|-c]
```

For example to associate CPU 10 to CLOS group 3, execute:

```
# intel-speed-select -c 10 core-power assoc -c 3
Intel(R) Speed Select Technology
Executing on CPU model: X
package-0
 die-0
   cpu-10
     core-power
       assoc:success
```

Once a CPU is associated, its sibling CPUs are also associated to a CLOS group. Once associated, avoid changing Linux "cpufreq" subsystem scaling frequency limits.

To check the existing association for a CPU, "core-power get-assoc" command can be used. For example, to get association of CPU 10, execute:

```
# intel-speed-select -c 10 core-power get-assoc
Intel(R) Speed Select Technology
Executing on CPU model: X
package-1
 die-0
```

```
  cpu-10
    get-assoc
      clos:3
```

This shows that CPU 10 is part of a CLOS group 3.

Disable CLOS based prioritization

To disable, execute:

```
# intel-speed-select core-power disable
```

Some features like Intel(R) SST-TF can only be enabled when CLOS based prioritization is enabled. For this reason, disabling while Intel(R) SST-TF is enabled can cause Intel(R) SST-TF to fail. This will cause the "disable" command to display an error if Intel(R) SST-TF is already enabled. In turn, to disable, the Intel(R) SST-TF feature must be disabled first.

Intel(R) Speed Select Technology - Base Frequency (Intel(R) SST-BF)

The Intel(R) Speed Select Technology - Base Frequency (Intel(R) SST-BF) feature lets the user control base frequency. If some critical workload threads demand constant high guaranteed performance, then this feature can be used to execute the thread at higher base frequency on specific sets of CPUs (high priority CPUs) at the cost of lower base frequency (low priority CPUs) on other CPUs. This feature does not require offline of the low priority CPUs.

The support of Intel(R) SST-BF depends on the Intel(R) Speed Select Technology - Performance Profile (Intel(R) SST-PP) performance level configuration. It is possible that only certain performance levels support Intel(R) SST-BF. It is also possible that only base performance level (level = 0) has support of Intel SST-BF. Consequently, first select the desired performance level to enable this feature.

In the system under test here, Intel(R) SST-BF is supported at the base performance level 0, but currently disabled. For example for the level 0:

```
# intel-speed-select -c 0 perf-profile info -l 0
Intel(R) Speed Select Technology
Executing on CPU model: X
package-0
 die-0
   cpu-0
     perf-profile-level-0
       ...

       speed-select-base-freq:disabled
       ...
```

Before enabling Intel(R) SST-BF and measuring its impact on a workload performance, execute some workload and measure performance and get a baseline performance to compare against.

Here the user wants more guaranteed performance. For this reason, it is likely that turbo is disabled. To disable turbo, execute:

```
#echo 1 > /sys/devices/system/cpu/intel_pstate/no_turbo
```

Based on the output of the ”intel-speed-select perf-profile info -l 0” base frequency of guaranteed frequency 2600 MHz.

Measure baseline performance for comparison

To compare, pick a multi-threaded workload where each thread can be scheduled on separate CPUs. ”Hackbench pipe” test is a good example on how to improve performance using Intel(R) SST-BF.

Below, the workload is measuring average scheduler wakeup latency, so a lower number means better performance:

```
# taskset -c 3,4 perf bench -r 100 sched pipe
# Running 'sched/pipe' benchmark:
# Executed 1000000 pipe operations between two processes
    Total time: 6.102 [sec]
      6.102445 usecs/op
        163868 ops/sec
```

While running the above test, if we take turbostat output, it will show us that 2 of the CPUs are busy and reaching max. frequency (which would be the base frequency as the turbo is disabled). The turbostat output:

```
#turbostat -c 0-13 --show Package,Core,CPU,Bzy_MHz -i 1
Package        Core    CPU     Bzy_MHz
0              0       0       1000
0              1       1       1005
0              2       2       1000
0              3       3       2600
0              4       4       2600
0              5       5       1000
0              6       6       1000
0              7       7       1005
0              8       8       1005
0              9       9       1000
0              10      10      1000
0              11      11      995
0              12      12      1000
0              13      13      1000
```

From the above turbostat output, both CPU 3 and 4 are very busy and reaching full guaranteed frequency of 2600 MHz.

Intel(R) SST-BF Capabilities

To get capabilities of Intel(R) SST-BF for the current performance level 0, execute:

```
# intel-speed-select base-freq info -l 0
Intel(R) Speed Select Technology
Executing on CPU model: X
package-0
 die-0
   cpu-0
     speed-select-base-freq
       high-priority-base-frequency(MHz):3000
       high-priority-cpu-mask:00000216,00002160
       high-priority-cpu-list:5,6,8,13,33,34,36,41
       low-priority-base-frequency(MHz):2400
       tjunction-temperature(C):125
       thermal-design-power(W):205
```

The above capabilities show that there are some CPUs on this system that can offer base frequency of 3000 MHz compared to the standard base frequency at this performance levels. Nevertheless, these CPUs are fixed, and they are presented via high-priority-cpu-list/high-priority-cpu-mask. But if this Intel(R) SST-BF feature is selected, the low priorities CPUs (which are not in high-priority-cpu-list) can only offer up to 2400 MHz. As a result, if this clipping of low priority CPUs is acceptable, then the user can enable Intel SST-BF feature particularly for the above "sched pipe" workload since only two CPUs are used, they can be scheduled on high priority CPUs and can get boost of 400 MHz.

Enable Intel(R) SST-BF

To enable Intel(R) SST-BF feature, execute:

```
# intel-speed-select base-freq enable -a
Intel(R) Speed Select Technology
Executing on CPU model: X
package-0
 die-0
   cpu-0
     base-freq
       enable:success
package-1
 die-0
   cpu-14
     base-freq
       enable:success
```

In this case, -a option is optional. This not only enables Intel(R) SST-BF, but it also adjusts the priority of cores using Intel(R) Speed Select Technology Core Power (Intel(R) SST-CP) features. This option sets the minimum performance of each Intel(R) Speed Select Technology - Performance Profile (Intel(R) SST-PP) class to maximum performance so that the hardware will give maximum performance possible for each CPU.

If -a option is not used, then the following steps are required before enabling Intel(R) SST-BF:

- Discover Intel(R) SST-BF and note low and high priority base frequency
- Note the high priority CPU list
- Enable CLOS using core-power feature set
- Configure CLOS parameters. Use CLOS.min to set to minimum performance
- Subscribe desired CPUs to CLOS groups

With this configuration, if the same workload is executed by pinning the workload to high priority CPUs (CPU 5 and 6 in this case):

```
#taskset -c 5,6 perf bench -r 100 sched pipe
# Running 'sched/pipe' benchmark:
# Executed 1000000 pipe operations between two processes
    Total time: 5.627 [sec]
      5.627922 usecs/op
        177685 ops/sec
```

This way, by enabling Intel(R) SST-BF, the performance of this benchmark is improved (latency reduced) by 7.79%. From the turbostat output, it can be observed that the high priority CPUs reached 3000 MHz compared to 2600 MHz. The turbostat output:

```
#turbostat -c 0-13 --show Package,Core,CPU,Bzy_MHz -i 1
Package        Core    CPU     Bzy_MHz
0              0       0       2151
0              1       1       2166
0              2       2       2175
0              3       3       2175
0              4       4       2175
0              5       5       3000
0              6       6       3000
0              7       7       2180
0              8       8       2662
0              9       9       2176
0              10      10      2175
0              11      11      2176
0              12      12      2176
0              13      13      2661
```

Disable Intel(R) SST-BF

To disable the Intel(R) SST-BF feature, execute:

```
# intel-speed-select base-freq disable -a
```

Intel(R) Speed Select Technology - Turbo Frequency (Intel(R) SST-TF)

This feature enables the ability to set different "All core turbo ratio limits" to cores based on the priority. By using this feature, some cores can be configured to get higher turbo frequency by designating them as high priority at the cost of lower or no turbo frequency on the low priority cores.

For this reason, this feature is only useful when system is busy utilizing all CPUs, but the user wants some configurable option to get high performance on some CPUs.

The support of Intel(R) Speed Select Technology - Turbo Frequency (Intel(R) SST-TF) depends on the Intel(R) Speed Select Technology - Performance Profile (Intel SST-PP) performance level configuration. It is possible that only a certain performance level supports Intel(R) SST-TF. It is also possible that only the base performance level (level = 0) has the support of Intel(R) SST-TF. Hence, first select the desired performance level to enable this feature.

In the system under test here, Intel(R) SST-TF is supported at the base performance level 0, but currently disabled:

```
# intel-speed-select -c 0 perf-profile info -l 0
Intel(R) Speed Select Technology
package-0
 die-0
   cpu-0
     perf-profile-level-0
       ...
       ...
       speed-select-turbo-freq:disabled
       ...
       ...
```

To check if performance can be improved using Intel(R) SST-TF feature, get the turbo frequency properties with Intel(R) SST-TF enabled and compare to the base turbo capability of this system.

Get Base turbo capability

To get the base turbo capability of performance level 0, execute:

```
# intel-speed-select perf-profile info -l 0
Intel(R) Speed Select Technology
Executing on CPU model: X
package-0
 die-0
   cpu-0
     perf-profile-level-0
       ...
       ...
       turbo-ratio-limits-sse
         bucket-0
           core-count:2
           max-turbo-frequency(MHz):3200
         bucket-1
```

```
            core-count:4
            max-turbo-frequency(MHz):3100
          bucket-2
            core-count:6
            max-turbo-frequency(MHz):3100
          bucket-3
            core-count:8
            max-turbo-frequency(MHz):3100
          bucket-4
            core-count:10
            max-turbo-frequency(MHz):3100
          bucket-5
            core-count:12
            max-turbo-frequency(MHz):3100
          bucket-6
            core-count:14
            max-turbo-frequency(MHz):3100
          bucket-7
            core-count:16
            max-turbo-frequency(MHz):3100
```

Based on the data above, when all the CPUS are busy, the max. frequency of 3100 MHz can be achieved. If there is some busy workload on cpu 0 - 11 (e.g. stress) and on CPU 12 and 13, execute "hackbench pipe" workload:

```
# taskset -c 12,13 perf bench -r 100 sched pipe
# Running 'sched/pipe' benchmark:
# Executed 1000000 pipe operations between two processes
    Total time: 5.705 [sec]
      5.705488 usecs/op
        175269 ops/sec
```

The turbostat output:

```
#turbostat -c 0-13 --show Package,Core,CPU,Bzy_MHz -i 1
Package        Core    CPU     Bzy_MHz
0              0       0       3000
0              1       1       3000
0              2       2       3000
0              3       3       3000
0              4       4       3000
0              5       5       3100
0              6       6       3100
0              7       7       3000
0              8       8       3100
0              9       9       3000
0              10      10      3000
0              11      11      3000
0              12      12      3100
0              13      13      3100
```

Based on turbostat output, the performance is limited by frequency cap of 3100 MHz. To check if the hackbench performance can be improved for CPU 12 and CPU 13, first check the capability of the Intel(R) SST-TF feature for this performance level.

Get Intel(R) SST-TF Capability

To get the capability, the "turbo-freq info" command can be used:

```
# intel-speed-select turbo-freq info -l 0
Intel(R) Speed Select Technology
Executing on CPU model: X
package-0
 die-0
   cpu-0
     speed-select-turbo-freq
         bucket-0
           high-priority-cores-count:2
           high-priority-max-frequency(MHz):3200
           high-priority-max-avx2-frequency(MHz):3200
           high-priority-max-avx512-frequency(MHz):3100
         bucket-1
           high-priority-cores-count:4
           high-priority-max-frequency(MHz):3100
           high-priority-max-avx2-frequency(MHz):3000
           high-priority-max-avx512-frequency(MHz):2900
         bucket-2
           high-priority-cores-count:6
           high-priority-max-frequency(MHz):3100
           high-priority-max-avx2-frequency(MHz):3000
           high-priority-max-avx512-frequency(MHz):2900
         speed-select-turbo-freq-clip-frequencies
           low-priority-max-frequency(MHz):2600
           low-priority-max-avx2-frequency(MHz):2400
           low-priority-max-avx512-frequency(MHz):2100
```

Based on the output above, there is an Intel(R) SST-TF bucket for which there are two high priority cores. If only two high priority cores are set, then max. turbo frequency on those cores can be increased to 3200 MHz. This is 100 MHz more than the base turbo capability for all cores.

In turn, for the hackbench workload, two CPUs can be set as high priority and rest as low priority. One side effect is that once enabled, the low priority cores will be clipped to a lower frequency of 2600 MHz.

Enable Intel(R) SST-TF

To enable Intel(R) SST-TF, execute:

```
# intel-speed-select -c 12,13 turbo-freq enable -a
Intel(R) Speed Select Technology
Executing on CPU model: X
package-0
 die-0
   cpu-12
     turbo-freq
       enable:success
package-0
 die-0
   cpu-13
     turbo-freq
       enable:success
package--1
 die-0
   cpu-63
     turbo-freq --auto
       enable:success
```

In this case, the option ”-a” is optional. If set, it enables Intel(R) SST-TF feature and also sets the CPUs to high and low priority using Intel Speed Select Technology Core Power (Intel(R) SST-CP) features. The CPU numbers passed with ”-c” arguments are marked as high priority, including its siblings.

If -a option is not used, then the following steps are required before enabling Intel(R) SST-TF:

- Discover Intel(R) SST-TF and note buckets of high priority cores and maximum frequency
- Enable CLOS using core-power feature set - Configure CLOS parameters
- Subscribe desired CPUs to CLOS groups making sure that high priority cores are set to the maximum frequency

If the same hackbench workload is executed, schedule hackbench threads on high priority CPUs:

```
#taskset -c 12,13 perf bench -r 100 sched pipe
# Running 'sched/pipe' benchmark:
# Executed 1000000 pipe operations between two processes
    Total time: 5.510 [sec]
      5.510165 usecs/op
        180826 ops/sec
```

This improved performance by around 3.3% improvement on a busy system. Here the turbostat output will show that the CPU 12 and CPU 13 are getting 100 MHz boost. The turbostat output:

```
#turbostat -c 0-13 --show Package,Core,CPU,Bzy_MHz -i 1
Package        Core    CPU     Bzy_MHz
...
```

```
0               12      12      3200
0               13      13      3200
```

67.3.9 Intel Uncore Frequency Scaling

Copyright
© 2022-2023 Intel Corporation

Author
Srinivas Pandruvada <srinivas.pandruvada@linux.intel.com>

Introduction

The uncore can consume significant amount of power in Intel's Xeon servers based on the workload characteristics. To optimize the total power and improve overall performance, SoCs have internal algorithms for scaling uncore frequency. These algorithms monitor workload usage of uncore and set a desirable frequency.

It is possible that users have different expectations of uncore performance and want to have control over it. The objective is similar to allowing users to set the scaling min/max frequencies via cpufreq sysfs to improve CPU performance. Users may have some latency sensitive workloads where they do not want any change to uncore frequency. Also, users may have workloads which require different core and uncore performance at distinct phases and they may want to use both cpufreq and the uncore scaling interface to distribute power and improve overall performance.

Sysfs Interface

To control uncore frequency, a sysfs interface is provided in the directory: */sys/devices/system/cpu/intel_uncore_frequency/*.

There is one directory for each package and die combination as the scope of uncore scaling control is per die in multiple die/package SoCs or per package for single die per package SoCs. The name represents the scope of control. For example: 'package_00_die_00' is for package id 0 and die 0.

Each package_*_die_* contains the following attributes:

`initial_max_freq_khz`
Out of reset, this attribute represent the maximum possible frequency. This is a read-only attribute. If users adjust max_freq_khz, they can always go back to maximum using the value from this attribute.

`initial_min_freq_khz`
Out of reset, this attribute represent the minimum possible frequency. This is a read-only attribute. If users adjust min_freq_khz, they can always go back to minimum using the value from this attribute.

`max_freq_khz`
This attribute is used to set the maximum uncore frequency.

min_freq_khz
This attribute is used to set the minimum uncore frequency.

current_freq_khz
This attribute is used to get the current uncore frequency.

SoCs with TPMI (Topology Aware Register and PM Capsule Interface)

An SoC can contain multiple power domains with individual or collection of mesh partitions. This partition is called fabric cluster.

Certain type of meshes will need to run at the same frequency, they will be placed in the same fabric cluster. Benefit of fabric cluster is that it offers a scalable mechanism to deal with partitioned fabrics in a SoC.

The current sysfs interface supports controls at package and die level. This interface is not enough to support more granular control at fabric cluster level.

SoCs with the support of TPMI (Topology Aware Register and PM Capsule Interface), can have multiple power domains. Each power domain can contain one or more fabric clusters.

To represent controls at fabric cluster level in addition to the controls at package and die level (like systems without TPMI support), sysfs is enhanced. This granular interface is presented in the sysfs with directories names prefixed with ”uncore”. For example: uncore00, uncore01 etc.

The scope of control is specified by attributes ”package_id”, ”domain_id” and ”fabric_cluster_id” in the directory.

Attributes in each directory:

domain_id
This attribute is used to get the power domain id of this instance.

fabric_cluster_id
This attribute is used to get the fabric cluster id of this instance.

package_id
This attribute is used to get the package id of this instance.

The other attributes are same as presented at package_*_die_* level.

In most of current use cases, the ”max_freq_khz” and ”min_freq_khz” is updated at ”package_*_die_*” level. This model will be still supported with the following approach:

When user uses controls at ”package_*_die_*” level, then every fabric cluster is affected in that package and die. For example: user changes ”max_freq_khz” in the package_00_die_00, then ”max_freq_khz” for uncore* directory with the same package id will be updated. In this case user can still update ”max_freq_khz” at each uncore* level, which is more restrictive. Similarly, user can update ”min_freq_khz” at ”package_*_die_*” level to apply at each uncore* level.

Support for ”current_freq_khz” is available only at each fabric cluster level (i.e., in uncore* directory).

CHAPTER

SIXTYEIGHT

SET UDEV RULES FOR PMF SMART PC BUILDER

AMD PMF(Platform Management Framework) Smart PC Solution builder has to set the system states like S0i3, Screen lock, hibernate etc, based on the output actions provided by the PMF TA (Trusted Application).

In order for this to work the PMF driver generates a uevent for userspace to react to. Below are sample udev rules that can facilitate this experience when a machine has PMF Smart PC solution builder enabled.

Please add the following line(s) to `/etc/udev/rules.d/99-local.rules`:

```
DRIVERS=="amd-pmf", ACTION=="change", ENV{EVENT_ID}=="0", RUN+="/usr/bin/
↪systemctl suspend"
DRIVERS=="amd-pmf", ACTION=="change", ENV{EVENT_ID}=="1", RUN+="/usr/bin/
↪systemctl hibernate"
DRIVERS=="amd-pmf", ACTION=="change", ENV{EVENT_ID}=="2", RUN+="/bin/loginctl␣
↪lock-sessions"
```

EVENT_ID values: 0= Put the system to S0i3/S2Idle 1= Put the system to hibernate 2= Lock the screen

CHAPTER

SIXTYNINE

LINUX PLUG AND PLAY DOCUMENTATION

Author
Adam Belay <ambx1@neo.rr.com>

Last updated
Oct. 16, 2002

69.1 Overview

Plug and Play provides a means of detecting and setting resources for legacy or otherwise unconfigurable devices. The Linux Plug and Play Layer provides these services to compatible drivers.

69.2 The User Interface

The Linux Plug and Play user interface provides a means to activate PnP devices for legacy and user level drivers that do not support Linux Plug and Play. The user interface is integrated into sysfs.

In addition to the standard sysfs file the following are created in each device's directory: - id - displays a list of support EISA IDs - options - displays possible resource configurations - resources - displays currently allocated resources and allows resource changes

69.2.1 activating a device

```
# echo "auto" > resources
```

this will invoke the automatic resource config system to activate the device

69.2.2 manually activating a device

```
# echo "manual <depnum> <mode>" > resources

<depnum> - the configuration number
<mode> - static or dynamic
         static = for next boot
         dynamic = now
```

69.2.3 disabling a device

```
# echo "disable" > resources
```

EXAMPLE:

Suppose you need to activate the floppy disk controller.

1. change to the proper directory, in my case it is /driver/bus/pnp/devices/00:0f:

   ```
   # cd /driver/bus/pnp/devices/00:0f
   # cat name
   PC standard floppy disk controller
   ```

2. check if the device is already active:

   ```
   # cat resources
   DISABLED
   ```

- Notice the string ”DISABLED”. This means the device is not active.

3. check the device's possible configurations (optional):

   ```
   # cat options
   Dependent: 01 - Priority acceptable
       port 0x3f0-0x3f0, align 0x7, size 0x6, 16-bit address decoding
       port 0x3f7-0x3f7, align 0x0, size 0x1, 16-bit address decoding
       irq 6
       dma 2 8-bit compatible
   Dependent: 02 - Priority acceptable
       port 0x370-0x370, align 0x7, size 0x6, 16-bit address decoding
       port 0x377-0x377, align 0x0, size 0x1, 16-bit address decoding
       irq 6
       dma 2 8-bit compatible
   ```

4. now activate the device:

   ```
   # echo "auto" > resources
   ```

5. finally check if the device is active:

   ```
   # cat resources
   io 0x3f0-0x3f5
   ```

```
io 0x3f7-0x3f7
irq 6
dma 2
```

also there are a series of kernel parameters:

```
pnp_reserve_irq=irq1[,irq2] ....
pnp_reserve_dma=dma1[,dma2] ....
pnp_reserve_io=io1,size1[,io2,size2] ....
pnp_reserve_mem=mem1,size1[,mem2,size2] ....
```

69.3 The Unified Plug and Play Layer

All Plug and Play drivers, protocols, and services meet at a central location called the Plug and Play Layer. This layer is responsible for the exchange of information between PnP drivers and PnP protocols. Thus it automatically forwards commands to the proper protocol. This makes writing PnP drivers significantly easier.

The following functions are available from the Plug and Play Layer:

pnp_get_protocol
increments the number of uses by one

pnp_put_protocol
deincrements the number of uses by one

pnp_register_protocol
use this to register a new PnP protocol

pnp_unregister_protocol
use this function to remove a PnP protocol from the Plug and Play Layer

pnp_register_driver
adds a PnP driver to the Plug and Play Layer

this includes driver model integration returns zero for success or a negative error number for failure; count calls to the .add() method if you need to know how many devices bind to the driver

pnp_unregister_driver
removes a PnP driver from the Plug and Play Layer

69.4 Plug and Play Protocols

This section contains information for PnP protocol developers.

The following Protocols are currently available in the computing world:

- **PNPBIOS:**
  used for system devices such as serial and parallel ports.
- **ISAPNP:**
  provides PnP support for the ISA bus

- **ACPI:**
  among its many uses, ACPI provides information about system level devices.

It is meant to replace the PNPBIOS. It is not currently supported by Linux Plug and Play but it is planned to be in the near future.

Requirements for a Linux PnP protocol: 1. the protocol must use EISA IDs 2. the protocol must inform the PnP Layer of a device's current configuration

- the ability to set resources is optional but preferred.

The following are PnP protocol related functions:

pnp_add_device
use this function to add a PnP device to the PnP layer

only call this function when all wanted values are set in the pnp_dev structure

pnp_init_device
call this to initialize the PnP structure

pnp_remove_device
call this to remove a device from the Plug and Play Layer. it will fail if the device is still in use. automatically will free mem used by the device and related structures

pnp_add_id
adds an EISA ID to the list of supported IDs for the specified device

For more information consult the source of a protocol such as /drivers/pnp/pnpbios/core.c.

69.5 Linux Plug and Play Drivers

This section contains information for Linux PnP driver developers.

69.5.1 The New Way

1. first make a list of supported EISA IDS

   ex:

```
static const struct pnp_id pnp_dev_table[] = {
        /* Standard LPT Printer Port */
        {.id = "PNP0400", .driver_data = 0},
        /* ECP Printer Port */
        {.id = "PNP0401", .driver_data = 0},
        {.id = ""}
};
```

   Please note that the character 'X' can be used as a wild card in the function portion (last four characters).

   ex:

```
/* Unknown PnP modems */
{       "PNPCXXX",              UNKNOWN_DEV     },
```

Supported PnP card IDs can optionally be defined. ex:

```
static const struct pnp_id pnp_card_table[] = {
        {       "ANYDEVS",              0       },
        {       "",                     0       }
};
```

2. Optionally define probe and remove functions. It may make sense not to define these functions if the driver already has a reliable method of detecting the resources, such as the parport_pc driver.

   ex:

```
static int
serial_pnp_probe(struct pnp_dev * dev, const struct pnp_id *card_id, const
                struct pnp_id *dev_id)
{
. . .
```

   ex:

```
static void serial_pnp_remove(struct pnp_dev * dev)
{
. . .
```

   consult /drivers/serial/8250_pnp.c for more information.

3. create a driver structure

   ex:

```
static struct pnp_driver serial_pnp_driver = {
        .name           = "serial",
        .card_id_table  = pnp_card_table,
        .id_table       = pnp_dev_table,
        .probe          = serial_pnp_probe,
        .remove         = serial_pnp_remove,
};
```

   - name and id_table cannot be NULL.

4. register the driver

   ex:

```
static int __init serial8250_pnp_init(void)
{
        return pnp_register_driver(&serial_pnp_driver);
}
```

69.5.2 The Old Way

A series of compatibility functions have been created to make it easy to convert ISAPNP drivers. They should serve as a temporary solution only.

They are as follows:

```
struct pnp_dev *pnp_find_dev(struct pnp_card *card,
                             unsigned short vendor,
                             unsigned short function,
                             struct pnp_dev *from)
```

CHAPTER

SEVENTY

RAPIDIO SUBSYSTEM GUIDE

Author
Matt Porter

70.1 Introduction

RapidIO is a high speed switched fabric interconnect with features aimed at the embedded market. RapidIO provides support for memory-mapped I/O as well as message-based transactions over the switched fabric network. RapidIO has a standardized discovery mechanism not unlike the PCI bus standard that allows simple detection of devices in a network.

This documentation is provided for developers intending to support RapidIO on new architectures, write new drivers, or to understand the subsystem internals.

70.2 Known Bugs and Limitations

70.2.1 Bugs

None. ;)

70.2.2 Limitations

1. Access/management of RapidIO memory regions is not supported
2. Multiple host enumeration is not supported

70.3 RapidIO driver interface

Drivers are provided a set of calls in order to interface with the subsystem to gather info on devices, request/map memory region resources, and manage mailboxes/doorbells.

70.3.1 Functions

int **rio_local_read_config_32**(struct *rio_mport* *port, u32 offset, u32 *data)

Read 32 bits from local configuration space

Parameters

struct rio_mport *port
Master port

u32 offset
Offset into local configuration space

u32 * data
Pointer to read data into

Description

Reads 32 bits of data from the specified offset within the local device's configuration space.

int **rio_local_write_config_32**(struct *rio_mport* *port, u32 offset, u32 data)

Write 32 bits to local configuration space

Parameters

struct rio_mport *port
Master port

u32 offset
Offset into local configuration space

u32 data
Data to be written

Description

Writes 32 bits of data to the specified offset within the local device's configuration space.

int **rio_local_read_config_16**(struct *rio_mport* *port, u32 offset, u16 *data)

Read 16 bits from local configuration space

Parameters

struct rio_mport *port
Master port

u32 offset
Offset into local configuration space

u16 * data
Pointer to read data into

Description

Reads 16 bits of data from the specified offset within the local device's configuration space.

int **rio_local_write_config_16**(struct *rio_mport* *port, u32 offset, u16 data)

Write 16 bits to local configuration space

Parameters

struct rio_mport *port
Master port

u32 offset
Offset into local configuration space

u16 data
Data to be written

Description

Writes 16 bits of data to the specified offset within the local device's configuration space.

int **rio_local_read_config_8**(struct *rio_mport* *port, u32 offset, u8 *data)
Read 8 bits from local configuration space

Parameters

struct rio_mport *port
Master port

u32 offset
Offset into local configuration space

u8 * data
Pointer to read data into

Description

Reads 8 bits of data from the specified offset within the local device's configuration space.

int **rio_local_write_config_8**(struct *rio_mport* *port, u32 offset, u8 data)
Write 8 bits to local configuration space

Parameters

struct rio_mport *port
Master port

u32 offset
Offset into local configuration space

u8 data
Data to be written

Description

Writes 8 bits of data to the specified offset within the local device's configuration space.

int **rio_read_config_32**(struct *rio_dev* *rdev, u32 offset, u32 *data)
Read 32 bits from configuration space

Parameters

struct rio_dev *rdev
RIO device

u32 offset
Offset into device configuration space

u32 * data
Pointer to read data into

Description

Reads 32 bits of data from the specified offset within the RIO device's configuration space.

int **rio_write_config_32**(struct *rio_dev* *rdev, u32 offset, u32 data)

Write 32 bits to configuration space

Parameters

struct rio_dev *rdev
RIO device

u32 offset
Offset into device configuration space

u32 data
Data to be written

Description

Writes 32 bits of data to the specified offset within the RIO device's configuration space.

int **rio_read_config_16**(struct *rio_dev* *rdev, u32 offset, u16 *data)

Read 16 bits from configuration space

Parameters

struct rio_dev *rdev
RIO device

u32 offset
Offset into device configuration space

u16 * data
Pointer to read data into

Description

Reads 16 bits of data from the specified offset within the RIO device's configuration space.

int **rio_write_config_16**(struct *rio_dev* *rdev, u32 offset, u16 data)

Write 16 bits to configuration space

Parameters

struct rio_dev *rdev
RIO device

u32 offset
Offset into device configuration space

u16 data
Data to be written

Description

Writes 16 bits of data to the specified offset within the RIO device's configuration space.

int **rio_read_config_8**(struct *rio_dev* *rdev, u32 offset, u8 *data)

Read 8 bits from configuration space

Parameters

struct rio_dev *rdev
RIO device

u32 offset
Offset into device configuration space

u8 * data
Pointer to read data into

Description

Reads 8 bits of data from the specified offset within the RIO device's configuration space.

int **rio_write_config_8**(struct *rio_dev* *rdev, u32 offset, u8 data)
Write 8 bits to configuration space

Parameters

struct rio_dev *rdev
RIO device

u32 offset
Offset into device configuration space

u8 data
Data to be written

Description

Writes 8 bits of data to the specified offset within the RIO device's configuration space.

int **rio_send_doorbell**(struct *rio_dev* *rdev, u16 data)
Send a doorbell message to a device

Parameters

struct rio_dev *rdev
RIO device

u16 data
Doorbell message data

Description

Send a doorbell message to a RIO device. The doorbell message has a 16-bit info field provided by the **data** argument.

void **rio_init_mbox_res**(struct resource *res, int start, int end)
Initialize a RIO mailbox resource

Parameters

struct resource *res
resource struct

int start
start of mailbox range

int end
end of mailbox range

Description

This function is used to initialize the fields of a resource for use as a mailbox resource. It initializes a range of mailboxes using the start and end arguments.

void **rio_init_dbell_res**(struct resource *res, u16 start, u16 end)

Initialize a RIO doorbell resource

Parameters

struct resource *res
resource struct

u16 start
start of doorbell range

u16 end
end of doorbell range

Description

This function is used to initialize the fields of a resource for use as a doorbell resource. It initializes a range of doorbell messages using the start and end arguments.

RIO_DEVICE

RIO_DEVICE (dev, ven)

macro used to describe a specific RIO device

Parameters

dev
the 16 bit RIO device ID

ven
the 16 bit RIO vendor ID

Description

This macro is used to create a struct rio_device_id that matches a specific device. The assembly vendor and assembly device fields will be set to `RIO_ANY_ID`.

int **rio_add_outb_message**(struct *rio_mport* *mport, struct *rio_dev* *rdev, int mbox, void *buffer, size_t len)

Add RIO message to an outbound mailbox queue

Parameters

struct rio_mport *mport
RIO master port containing the outbound queue

struct rio_dev *rdev
RIO device the message is be sent to

int mbox
The outbound mailbox queue

void *buffer
Pointer to the message buffer

size_t len
Length of the message buffer

Description

Adds a RIO message buffer to an outbound mailbox queue for transmission. Returns 0 on success.

int **rio_add_inb_buffer**(struct *rio_mport* *mport, int mbox, void *buffer)
Add buffer to an inbound mailbox queue

Parameters

struct rio_mport *mport
Master port containing the inbound mailbox

int mbox
The inbound mailbox number

void *buffer
Pointer to the message buffer

Description

Adds a buffer to an inbound mailbox queue for reception. Returns 0 on success.

void ***rio_get_inb_message**(struct *rio_mport* *mport, int mbox)
Get A RIO message from an inbound mailbox queue

Parameters

struct rio_mport *mport
Master port containing the inbound mailbox

int mbox
The inbound mailbox number

Description

Get a RIO message from an inbound mailbox queue. Returns 0 on success.

const char ***rio_name**(struct *rio_dev* *rdev)
Get the unique RIO device identifier

Parameters

struct rio_dev *rdev
RIO device

Description

Get the unique RIO device identifier. Returns the device identifier string.

void ***rio_get_drvdata**(struct *rio_dev* *rdev)
Get RIO driver specific data

Parameters

struct rio_dev *rdev
RIO device

Description

Get RIO driver specific data. Returns a pointer to the driver specific data.

void **rio_set_drvdata**(struct *rio_dev* *rdev, void *data)

Set RIO driver specific data

Parameters

struct rio_dev *rdev

RIO device

void *data

Pointer to driver specific data

Description

Set RIO driver specific data. device struct driver data pointer is set to the **data** argument.

struct *rio_dev* ***rio_dev_get**(struct *rio_dev* *rdev)

Increments the reference count of the RIO device structure

Parameters

struct rio_dev *rdev

RIO device being referenced

Description

Each live reference to a device should be refcounted.

Drivers for RIO devices should normally record such references in their probe() methods, when they bind to a device, and release them by calling `rio_dev_put()`, in their disconnect() methods.

void **rio_dev_put**(struct *rio_dev* *rdev)

Release a use of the RIO device structure

Parameters

struct rio_dev *rdev

RIO device being disconnected

Description

Must be called when a user of a device is finished with it. When the last user of the device calls this function, the memory of the device is freed.

int **rio_register_driver**(struct *rio_driver* *rdrv)

register a new RIO driver

Parameters

struct rio_driver *rdrv

the RIO driver structure to register

Adds a `struct rio_driver` to the list of registered drivers. Returns a negative value on error, otherwise 0. If no error occurred, the driver remains registered even if no device was claimed during registration.

void **rio_unregister_driver**(struct *rio_driver* *rdrv)

unregister a RIO driver

Parameters

struct rio_driver *rdrv
the RIO driver structure to unregister

Deletes the *struct rio_driver* from the list of registered RIO drivers, gives it a chance to clean up by calling its remove() function for each device it was responsible for, and marks those devices as driverless.

u16 **rio_local_get_device_id**(struct *rio_mport* *port)
Get the base/extended device id for a port

Parameters

struct rio_mport *port
RIO master port from which to get the deviceid

Description

Reads the base/extended device id from the local device implementing the master port. Returns the 8/16-bit device id.

int **rio_query_mport**(struct *rio_mport* *port, struct *rio_mport_attr* *mport_attr)
Query mport device attributes

Parameters

struct rio_mport *port
mport device to query

struct rio_mport_attr *mport_attr
mport attributes data structure

Description

Returns attributes of specified mport through the pointer to attributes data structure.

struct *rio_net* ***rio_alloc_net**(struct *rio_mport* *mport)
Allocate and initialize a new RIO network data structure

Parameters

struct rio_mport *mport
Master port associated with the RIO network

Description

Allocates a RIO network structure, initializes per-network list heads, and adds the associated master port to the network list of associated master ports. Returns a RIO network pointer on success or `NULL` on failure.

void **rio_local_set_device_id**(struct *rio_mport* *port, u16 did)
Set the base/extended device id for a port

Parameters

struct rio_mport *port
RIO master port

u16 did
Device ID value to be written

Description

Writes the base/extended device id from a device.

int **rio_add_device**(struct *rio_dev* *rdev)

Adds a RIO device to the device model

Parameters

struct rio_dev *rdev
RIO device

Description

Adds the RIO device to the global device list and adds the RIO device to the RIO device list. Creates the generic sysfs nodes for an RIO device.

int **rio_request_inb_mbox**(struct *rio_mport* *mport, void *dev_id, int mbox, int entries, void (*minb)(struct *rio_mport* *mport, void *dev_id, int mbox, int slot))

request inbound mailbox service

Parameters

struct rio_mport *mport
RIO master port from which to allocate the mailbox resource

void *dev_id
Device specific pointer to pass on event

int mbox
Mailbox number to claim

int entries
Number of entries in inbound mailbox queue

void (*minb) (struct rio_mport * mport, void *dev_id, int mbox, int slot)
Callback to execute when inbound message is received

Description

Requests ownership of an inbound mailbox resource and binds a callback function to the resource. Returns `0` on success.

int **rio_release_inb_mbox**(struct *rio_mport* *mport, int mbox)

release inbound mailbox message service

Parameters

struct rio_mport *mport
RIO master port from which to release the mailbox resource

int mbox
Mailbox number to release

Description

Releases ownership of an inbound mailbox resource. Returns 0 if the request has been satisfied.

int **rio_request_outb_mbox**(struct *rio_mport* *mport, void *dev_id, int mbox, int entries, void (*moutb)(struct *rio_mport* *mport, void *dev_id, int mbox, int slot))

request outbound mailbox service

Parameters

struct rio_mport *mport
RIO master port from which to allocate the mailbox resource

void *dev_id
Device specific pointer to pass on event

int mbox
Mailbox number to claim

int entries
Number of entries in outbound mailbox queue

void (*moutb) (struct rio_mport * mport, void *dev_id, int mbox, int slot)
Callback to execute when outbound message is sent

Description

Requests ownership of an outbound mailbox resource and binds a callback function to the resource. Returns 0 on success.

int **rio_release_outb_mbox**(struct *rio_mport* *mport, int mbox)
release outbound mailbox message service

Parameters

struct rio_mport *mport
RIO master port from which to release the mailbox resource

int mbox
Mailbox number to release

Description

Releases ownership of an inbound mailbox resource. Returns 0 if the request has been satisfied.

int **rio_request_inb_dbell**(struct *rio_mport* *mport, void *dev_id, u16 start, u16 end, void (*dinb)(struct *rio_mport* *mport, void *dev_id, u16 src, u16 dst, u16 info))
request inbound doorbell message service

Parameters

struct rio_mport *mport
RIO master port from which to allocate the doorbell resource

void *dev_id
Device specific pointer to pass on event

u16 start
Doorbell info range start

u16 end
Doorbell info range end

void (*dinb) (struct rio_mport * mport, void *dev_id, u16 src, u16 dst, u16 info)
Callback to execute when doorbell is received

Description

Requests ownership of an inbound doorbell resource and binds a callback function to the resource. Returns 0 if the request has been satisfied.

int **rio_release_inb_dbell**(struct *rio_mport* *mport, u16 start, u16 end)

release inbound doorbell message service

Parameters

struct rio_mport *mport
RIO master port from which to release the doorbell resource

u16 start
Doorbell info range start

u16 end
Doorbell info range end

Description

Releases ownership of an inbound doorbell resource and removes callback from the doorbell event list. Returns 0 if the request has been satisfied.

struct resource ***rio_request_outb_dbell**(struct *rio_dev* *rdev, u16 start, u16 end)

request outbound doorbell message range

Parameters

struct rio_dev *rdev
RIO device from which to allocate the doorbell resource

u16 start
Doorbell message range start

u16 end
Doorbell message range end

Description

Requests ownership of a doorbell message range. Returns a resource if the request has been satisfied or `NULL` on failure.

int **rio_release_outb_dbell**(struct *rio_dev* *rdev, struct resource *res)

release outbound doorbell message range

Parameters

struct rio_dev *rdev
RIO device from which to release the doorbell resource

struct resource *res
Doorbell resource to be freed

Description

Releases ownership of a doorbell message range. Returns 0 if the request has been satisfied.

int **rio_add_mport_pw_handler**(struct *rio_mport* *mport, void *context, int (*pwcback)(struct *rio_mport* *mport, void *context, union rio_pw_msg *msg, int step))

add port-write message handler into the list of mport specific pw handlers

Parameters

struct rio_mport *mport
RIO master port to bind the portwrite callback

void *context
Handler specific context to pass on event

int (*pwcback)(struct rio_mport *mport, void *context, union rio_pw_msg *msg, int step)
Callback to execute when portwrite is received

Description

Returns 0 if the request has been satisfied.

int **rio_del_mport_pw_handler**(struct *rio_mport* *mport, void *context, int (*pwcback)(struct *rio_mport* *mport, void *context, union rio_pw_msg *msg, int step))

remove port-write message handler from the list of mport specific pw handlers

Parameters

struct rio_mport *mport
RIO master port to bind the portwrite callback

void *context
Registered handler specific context to pass on event

int (*pwcback)(struct rio_mport *mport, void *context, union rio_pw_msg *msg, int step)
Registered callback function

Description

Returns 0 if the request has been satisfied.

int **rio_request_inb_pwrite**(struct *rio_dev* *rdev, int (*pwcback)(struct *rio_dev* *rdev, union rio_pw_msg *msg, int step))

request inbound port-write message service for specific RapidIO device

Parameters

struct rio_dev *rdev
RIO device to which register inbound port-write callback routine

int (*pwcback)(struct rio_dev *rdev, union rio_pw_msg *msg, int step)
Callback routine to execute when port-write is received

Description

Binds a port-write callback function to the RapidIO device. Returns 0 if the request has been satisfied.

int **rio_release_inb_pwrite**(struct *rio_dev* *rdev)

release inbound port-write message service associated with specific RapidIO device

Parameters

struct rio_dev *rdev
RIO device which registered for inbound port-write callback

Description

Removes callback from the rio_dev structure. Returns 0 if the request has been satisfied.

void **rio_pw_enable**(struct *rio_mport* *mport, int enable)
Enables/disables port-write handling by a master port

Parameters

struct rio_mport *mport
Master port associated with port-write handling

int enable
1=enable, 0=disable

int **rio_map_inb_region**(struct *rio_mport* *mport, dma_addr_t local, u64 rbase, u32 size, u32 rflags)

- Map inbound memory region.

Parameters

struct rio_mport *mport
Master port.

dma_addr_t local
physical address of memory region to be mapped

u64 rbase
RIO base address assigned to this window

u32 size
Size of the memory region

u32 rflags
Flags for mapping.

Return

0 -- Success.

Description

This function will create the mapping from RIO space to local memory.

void **rio_unmap_inb_region**(struct *rio_mport* *mport, dma_addr_t lstart)

- Unmap the inbound memory region

Parameters

struct rio_mport *mport
Master port

dma_addr_t lstart
physical address of memory region to be unmapped

int **rio_map_outb_region**(struct *rio_mport* *mport, u16 destid, u64 rbase, u32 size, u32 rflags, dma_addr_t *local)

- Map outbound memory region.

Parameters

struct rio_mport *mport
Master port.

u16 destid
destination id window points to

u64 rbase
RIO base address window translates to

u32 size
Size of the memory region

u32 rflags
Flags for mapping.

dma_addr_t *local
physical address of memory region mapped

Return

0 -- Success.

Description

This function will create the mapping from RIO space to local memory.

void **rio_unmap_outb_region**(struct *rio_mport* *mport, u16 destid, u64 rstart)

- Unmap the inbound memory region

Parameters

struct rio_mport *mport
Master port

u16 destid
destination id mapping points to

u64 rstart
RIO base address window translates to

u32 **rio_mport_get_physefb**(struct *rio_mport* *port, int local, u16 destid, u8 hopcount, u32 *rmap)

Helper function that returns register offset for Physical Layer Extended Features Block.

Parameters

struct rio_mport *port
Master port to issue transaction

int local
Indicate a local master port or remote device access

u16 destid
Destination ID of the device

u8 hopcount
Number of switch hops to the device

u32 *rmap
pointer to location to store register map type info

struct *rio_dev* ***rio_get_comptag**(u32 comp_tag, struct *rio_dev* *from)
Begin or continue searching for a RIO device by component tag

Parameters

u32 comp_tag
RIO component tag to match

struct rio_dev *from
Previous RIO device found in search, or NULL for new search

Description

Iterates through the list of known RIO devices. If a RIO device is found with a matching **comp_tag**, a pointer to its device structure is returned. Otherwise, NULL is returned. A new search is initiated by passing NULL to the **from** argument. Otherwise, if **from** is not NULL, searches continue from next device on the global list.

int **rio_set_port_lockout**(struct *rio_dev* *rdev, u32 pnum, int lock)
Sets/clears LOCKOUT bit (RIO EM 1.3) for a switch port.

Parameters

struct rio_dev *rdev
Pointer to RIO device control structure

u32 pnum
Switch port number to set LOCKOUT bit

int lock
Operation : set (=1) or clear (=0)

int **rio_enable_rx_tx_port**(struct *rio_mport* *port, int local, u16 destid, u8 hopcount, u8 port_num)
enable input receiver and output transmitter of given port

Parameters

struct rio_mport *port
Master port associated with the RIO network

int local
local=1 select local port otherwise a far device is reached

u16 destid
Destination ID of the device to check host bit

u8 hopcount
Number of hops to reach the target

u8 port_num
Port (-number on switch) to enable on a far end device

Description

Returns 0 or 1 from on General Control Command and Status Register (EXT_PTR+0x3C)

int **rio_mport_chk_dev_access**(struct *rio_mport* *mport, u16 destid, u8 hopcount)

Validate access to the specified device.

Parameters

struct rio_mport *mport
Master port to send transactions

u16 destid
Device destination ID in network

u8 hopcount
Number of hops into the network

int **rio_inb_pwrite_handler**(struct *rio_mport* *mport, union rio_pw_msg *pw_msg)

inbound port-write message handler

Parameters

struct rio_mport *mport
mport device associated with port-write

union rio_pw_msg *pw_msg
pointer to inbound port-write message

Description

Processes an inbound port-write message. Returns 0 if the request has been satisfied.

u32 **rio_mport_get_efb**(struct *rio_mport* *port, int local, u16 destid, u8 hopcount, u32 from)

get pointer to next extended features block

Parameters

struct rio_mport *port
Master port to issue transaction

int local
Indicate a local master port or remote device access

u16 destid
Destination ID of the device

u8 hopcount
Number of switch hops to the device

u32 from
Offset of current Extended Feature block header (if 0 starts from ExtFeaturePtr)

u32 **rio_mport_get_feature**(struct *rio_mport* *port, int local, u16 destid, u8 hopcount, int ftr)

query for devices' extended features

Parameters

struct rio_mport * port
Master port to issue transaction

int local
Indicate a local master port or remote device access

u16 destid
Destination ID of the device

u8 hopcount
Number of switch hops to the device

int ftr
Extended feature code

Description

Tell if a device supports a given RapidIO capability. Returns the offset of the requested extended feature block within the device's RIO configuration space or 0 in case the device does not support it.

int **rio_lock_device**(struct *rio_mport* *port, u16 destid, u8 hopcount, int wait_ms)

Acquires host device lock for specified device

Parameters

struct rio_mport *port
Master port to send transaction

u16 destid
Destination ID for device/switch

u8 hopcount
Hopcount to reach switch

int wait_ms
Max wait time in msec (0 = no timeout)

Description

Attepts to acquire host device lock for specified device Returns 0 if device lock acquired or EINVAL if timeout expires.

int **rio_unlock_device**(struct *rio_mport* *port, u16 destid, u8 hopcount)

Releases host device lock for specified device

Parameters

struct rio_mport *port
Master port to send transaction

u16 destid
Destination ID for device/switch

u8 hopcount
Hopcount to reach switch

Description

Returns 0 if device lock released or EINVAL if fails.

int **rio_route_add_entry**(struct *rio_dev* *rdev, u16 table, u16 route_destid, u8 route_port, int lock)

Add a route entry to a switch routing table

Parameters

struct rio_dev *rdev
RIO device

u16 table
Routing table ID

u16 route_destid
Destination ID to be routed

u8 route_port
Port number to be routed

int lock
apply a hardware lock on switch device flag (1=lock, 0=no_lock)

Description

If available calls the switch specific add_entry() method to add a route entry into a switch routing table. Otherwise uses standard RT update method as defined by RapidIO specification. A specific routing table can be selected using the **table** argument if a switch has per port routing tables or the standard (or global) table may be used by passing `RIO_GLOBAL_TABLE` in **table**.

Returns `0` on success or `-EINVAL` on failure.

int **rio_route_get_entry**(struct *rio_dev* *rdev, u16 table, u16 route_destid, u8 *route_port, int lock)

Read an entry from a switch routing table

Parameters

struct rio_dev *rdev
RIO device

u16 table
Routing table ID

u16 route_destid
Destination ID to be routed

u8 *route_port
Pointer to read port number into

int lock
apply a hardware lock on switch device flag (1=lock, 0=no_lock)

Description

If available calls the switch specific get_entry() method to fetch a route entry from a switch routing table. Otherwise uses standard RT read method as defined by RapidIO specification. A specific routing table can be selected using the **table** argument if a switch has per port routing tables or the standard (or global) table may be used by passing `RIO_GLOBAL_TABLE` in **table**.

Returns `0` on success or `-EINVAL` on failure.

int **rio_route_clr_table**(struct *rio_dev* *rdev, u16 table, int lock)

Clear a switch routing table

Parameters

struct rio_dev *rdev
RIO device

u16 table
Routing table ID

int lock
apply a hardware lock on switch device flag (1=lock, 0=no_lock)

Description

If available calls the switch specific clr_table() method to clear a switch routing table. Otherwise uses standard RT write method as defined by RapidIO specification. A specific routing table can be selected using the **table** argument if a switch has per port routing tables or the standard (or global) table may be used by passing `RIO_GLOBAL_TABLE` in **table**.

Returns `0` on success or `-EINVAL` on failure.

struct dma_chan ***rio_request_mport_dma**(struct *rio_mport* *mport)
request RapidIO capable DMA channel associated with specified local RapidIO mport device.

Parameters

struct rio_mport *mport
RIO mport to perform DMA data transfers

Description

Returns pointer to allocated DMA channel or NULL if failed.

struct dma_chan ***rio_request_dma**(struct *rio_dev* *rdev)
request RapidIO capable DMA channel that supports specified target RapidIO device.

Parameters

struct rio_dev *rdev
RIO device associated with DMA transfer

Description

Returns pointer to allocated DMA channel or NULL if failed.

void **rio_release_dma**(struct dma_chan *dchan)
release specified DMA channel

Parameters

struct dma_chan *dchan
DMA channel to release

struct dma_async_tx_descriptor ***rio_dma_prep_xfer**(struct dma_chan *dchan, u16 destid, struct rio_dma_data *data, enum dma_transfer_direction direction, unsigned long flags)
RapidIO specific wrapper for device_prep_slave_sg callback defined by DMAENGINE.

Parameters

struct dma_chan *dchan
DMA channel to configure

u16 destid
target RapidIO device destination ID

struct rio_dma_data *data
RIO specific data descriptor

enum dma_transfer_direction direction
DMA data transfer direction (TO or FROM the device)

unsigned long flags
dmaengine defined flags

Description

Initializes RapidIO capable DMA channel for the specified data transfer. Uses DMA channel private extension to pass information related to remote target RIO device.

Return

pointer to DMA transaction descriptor if successful,
error-valued pointer or NULL if failed.

struct dma_async_tx_descriptor ***rio_dma_prep_slave_sg**(struct *rio_dev* *rdev, struct dma_chan *dchan, struct rio_dma_data *data, enum dma_transfer_direction direction, unsigned long flags)

RapidIO specific wrapper for device_prep_slave_sg callback defined by DMAENGINE.

Parameters

struct rio_dev *rdev
RIO device control structure

struct dma_chan *dchan
DMA channel to configure

struct rio_dma_data *data
RIO specific data descriptor

enum dma_transfer_direction direction
DMA data transfer direction (TO or FROM the device)

unsigned long flags
dmaengine defined flags

Description

Initializes RapidIO capable DMA channel for the specified data transfer. Uses DMA channel private extension to pass information related to remote target RIO device.

Return

pointer to DMA transaction descriptor if successful,
error-valued pointer or NULL if failed.

int **rio_register_scan**(int mport_id, struct *rio_scan* *scan_ops)

enumeration/discovery method registration interface

Parameters

int mport_id
mport device ID for which fabric scan routine has to be set (RIO_MPORT_ANY = set for all available mports)

struct rio_scan *scan_ops
enumeration/discovery operations structure

Description

Registers enumeration/discovery operations with RapidIO subsystem and attaches it to the specified mport device (or all available mports if RIO_MPORT_ANY is specified).

Returns error if the mport already has an enumerator attached to it. In case of RIO_MPORT_ANY skips mports with valid scan routines (no error).

int **rio_unregister_scan**(int mport_id, struct *rio_scan* *scan_ops)
removes enumeration/discovery method from mport

Parameters

int mport_id
mport device ID for which fabric scan routine has to be unregistered (RIO_MPORT_ANY = apply to all mports that use the specified scan_ops)

struct rio_scan *scan_ops
enumeration/discovery operations structure

Description

Removes enumeration or discovery method assigned to the specified mport device. If RIO_MPORT_ANY is specified, removes the specified operations from all mports that have them attached.

70.4 Internals

This chapter contains the autogenerated documentation of the RapidIO subsystem.

70.4.1 Structures

struct **rio_switch**
RIO switch info

Definition:

```
struct rio_switch {
    struct list_head node;
    u8 *route_table;
    u32 port_ok;
    struct rio_switch_ops *ops;
    spinlock_t lock;
    struct rio_dev *nextdev[];
};
```

Members

node
Node in global list of switches

route_table
Copy of switch routing table

port_ok
Status of each port (one bit per port) - OK=1 or UNINIT=0

ops
pointer to switch-specific operations

lock
lock to serialize operations updates

nextdev
Array of per-port pointers to the next attached device

struct **rio_switch_ops**
Per-switch operations

Definition:

```
struct rio_switch_ops {
    struct module *owner;
    int (*add_entry) (struct rio_mport *mport, u16 destid, u8 hopcount, u16␣
↪table, u16 route_destid, u8 route_port);
    int (*get_entry) (struct rio_mport *mport, u16 destid, u8 hopcount, u16␣
↪table, u16 route_destid, u8 *route_port);
    int (*clr_table) (struct rio_mport *mport, u16 destid, u8 hopcount, u16␣
↪table);
    int (*set_domain) (struct rio_mport *mport, u16 destid, u8 hopcount, u8 sw_
↪domain);
    int (*get_domain) (struct rio_mport *mport, u16 destid, u8 hopcount, u8␣
↪*sw_domain);
    int (*em_init) (struct rio_dev *dev);
    int (*em_handle) (struct rio_dev *dev, u8 swport);
};
```

Members

owner
The module owner of this structure

add_entry
Callback for switch-specific route add function

get_entry
Callback for switch-specific route get function

clr_table
Callback for switch-specific clear route table function

set_domain
Callback for switch-specific domain setting function

get_domain
Callback for switch-specific domain get function

em_init
Callback for switch-specific error management init function

em_handle
Callback for switch-specific error management handler function

Description

Defines the operations that are necessary to initialize/control a particular RIO switch device.

struct **rio_dev**
RIO device info

Definition:

```
struct rio_dev {
    struct list_head global_list;
    struct list_head net_list;
    struct rio_net *net;
    bool do_enum;
    u16 did;
    u16 vid;
    u32 device_rev;
    u16 asm_did;
    u16 asm_vid;
    u16 asm_rev;
    u16 efptr;
    u32 pef;
    u32 swpinfo;
    u32 src_ops;
    u32 dst_ops;
    u32 comp_tag;
    u32 phys_efptr;
    u32 phys_rmap;
    u32 em_efptr;
    u64 dma_mask;
    struct rio_driver *driver;
    struct device dev;
    struct resource riores[RIO_MAX_DEV_RESOURCES];
    int (*pwcback) (struct rio_dev *rdev, union rio_pw_msg *msg, int step);
    u16 destid;
    u8 hopcount;
    struct rio_dev *prev;
    atomic_t state;
    struct rio_switch rswitch[];
};
```

Members

global_list
Node in list of all RIO devices

net_list
Node in list of RIO devices in a network

net
Network this device is a part of

do_enum
Enumeration flag

did
Device ID

vid
Vendor ID

device_rev
Device revision

asm_did
Assembly device ID

asm_vid
Assembly vendor ID

asm_rev
Assembly revision

efptr
Extended feature pointer

pef
Processing element features

swpinfo
Switch port info

src_ops
Source operation capabilities

dst_ops
Destination operation capabilities

comp_tag
RIO component tag

phys_efptr
RIO device extended features pointer

phys_rmap
LP-Serial Register Map Type (1 or 2)

em_efptr
RIO Error Management features pointer

dma_mask
Mask of bits of RIO address this device implements

driver
Driver claiming this device

dev
Device model device

riores
RIO resources this device owns

pwcback
port-write callback function for this device

destid
Network destination ID (or associated destid for switch)

hopcount
Hopcount to this device

prev
Previous RIO device connected to the current one

state
device state

rswitch
struct rio_switch (if valid for this device)

struct **rio_msg**
RIO message event

Definition:

```
struct rio_msg {
    struct resource *res;
    void (*mcback) (struct rio_mport * mport, void *dev_id, int mbox, int␣
↪slot);
};
```

Members

res
Mailbox resource

mcback
Message event callback

struct **rio_dbell**
RIO doorbell event

Definition:

```
struct rio_dbell {
    struct list_head node;
    struct resource *res;
    void (*dinb) (struct rio_mport *mport, void *dev_id, u16 src, u16 dst, u16␣
↪info);
    void *dev_id;
};
```

Members

node
Node in list of doorbell events

res
Doorbell resource

dinb
Doorbell event callback

dev_id
Device specific pointer to pass on event

struct **rio_mport**
RIO master port info

Definition:

```
struct rio_mport {
    struct list_head dbells;
    struct list_head pwrites;
    struct list_head node;
    struct list_head nnode;
    struct rio_net *net;
    struct mutex lock;
    struct resource iores;
    struct resource riores[RIO_MAX_MPORT_RESOURCES];
    struct rio_msg inb_msg[RIO_MAX_MBOX];
    struct rio_msg outb_msg[RIO_MAX_MBOX];
    int host_deviceid;
    struct rio_ops *ops;
    unsigned char id;
    unsigned char index;
    unsigned int sys_size;
    u32 phys_efptr;
    u32 phys_rmap;
    unsigned char name[RIO_MAX_MPORT_NAME];
    struct device dev;
    void *priv;
#ifdef CONFIG_RAPIDIO_DMA_ENGINE;
    struct dma_device       dma;
#endif;
    struct rio_scan *nscan;
    atomic_t state;
    unsigned int pwe_refcnt;
};
```

Members

dbells
List of doorbell events

pwrites
List of portwrite events

node
Node in global list of master ports

nnode

Node in network list of master ports

net
RIO net this mport is attached to

lock
lock to synchronize lists manipulations

iores
I/O mem resource that this master port interface owns

riores
RIO resources that this master port interfaces owns

inb_msg
RIO inbound message event descriptors

outb_msg
RIO outbound message event descriptors

host_deviceid
Host device ID associated with this master port

ops
configuration space functions

id
Port ID, unique among all ports

index
Port index, unique among all port interfaces of the same type

sys_size
RapidIO common transport system size

phys_efptr
RIO port extended features pointer

phys_rmap
LP-Serial EFB Register Mapping type (1 or 2).

name
Port name string

dev
device structure associated with an mport

priv
Master port private data

dma
DMA device associated with mport

nscan
RapidIO network enumeration/discovery operations

state
mport device state

pwe_refcnt
port-write enable ref counter to track enable/disable requests

struct **rio_net**

RIO network info

Definition:

```
struct rio_net {
    struct list_head node;
    struct list_head devices;
    struct list_head switches;
    struct list_head mports;
    struct rio_mport *hport;
    unsigned char id;
    struct device dev;
    void *enum_data;
    void (*release)(struct rio_net *net);
};
```

Members

node
Node in global list of RIO networks

devices
List of devices in this network

switches
List of switches in this network

mports
List of master ports accessing this network

hport
Default port for accessing this network

id
RIO network ID

dev
Device object

enum_data
private data specific to a network enumerator

release
enumerator-specific release callback

struct **rio_mport_attr**

RIO mport device attributes

Definition:

```
struct rio_mport_attr {
    int flags;
    int link_speed;
    int link_width;
    int dma_max_sge;
    int dma_max_size;
```

```
    int dma_align;
};
```

Members

flags
mport device capability flags

link_speed
SRIO link speed value (as defined by RapidIO specification)

link_width
SRIO link width value (as defined by RapidIO specification)

dma_max_sge
number of SG list entries that can be handled by DMA channel(s)

dma_max_size
max number of bytes in single DMA transfer (SG entry)

dma_align
alignment shift for DMA operations (as for other DMA operations)

struct **rio_ops**
Low-level RIO configuration space operations

Definition:

```
struct rio_ops {
    int (*lcread) (struct rio_mport *mport, int index, u32 offset, int len,␣
↪u32 *data);
    int (*lcwrite) (struct rio_mport *mport, int index, u32 offset, int len,␣
↪u32 data);
    int (*cread) (struct rio_mport *mport, int index, u16 destid, u8 hopcount,␣
↪u32 offset, int len, u32 *data);
    int (*cwrite) (struct rio_mport *mport, int index, u16 destid, u8 hopcount,
↪ u32 offset, int len, u32 data);
    int (*dsend) (struct rio_mport *mport, int index, u16 destid, u16 data);
    int (*pwenable) (struct rio_mport *mport, int enable);
    int (*open_outb_mbox)(struct rio_mport *mport, void *dev_id, int mbox, int␣
↪entries);
    void (*close_outb_mbox)(struct rio_mport *mport, int mbox);
    int (*open_inb_mbox)(struct rio_mport *mport, void *dev_id, int mbox, int␣
↪entries);
    void (*close_inb_mbox)(struct rio_mport *mport, int mbox);
    int (*add_outb_message)(struct rio_mport *mport, struct rio_dev *rdev, int␣
↪mbox, void *buffer, size_t len);
    int (*add_inb_buffer)(struct rio_mport *mport, int mbox, void *buf);
    void *(*get_inb_message)(struct rio_mport *mport, int mbox);
    int (*map_inb)(struct rio_mport *mport, dma_addr_t lstart, u64 rstart, u64␣
↪size, u32 flags);
    void (*unmap_inb)(struct rio_mport *mport, dma_addr_t lstart);
    int (*query_mport)(struct rio_mport *mport, struct rio_mport_attr *attr);
    int (*map_outb)(struct rio_mport *mport, u16 destid, u64 rstart, u32 size,␣
↪u32 flags, dma_addr_t *laddr);
```

```
    void (*unmap_outb)(struct rio_mport *mport, u16 destid, u64 rstart);
};
```

Members

lcread
Callback to perform local (master port) read of config space.

lcwrite
Callback to perform local (master port) write of config space.

cread
Callback to perform network read of config space.

cwrite
Callback to perform network write of config space.

dsend
Callback to send a doorbell message.

pwenable
Callback to enable/disable port-write message handling.

open_outb_mbox
Callback to initialize outbound mailbox.

close_outb_mbox
Callback to shut down outbound mailbox.

open_inb_mbox
Callback to initialize inbound mailbox.

close_inb_mbox
Callback to shut down inbound mailbox.

add_outb_message
Callback to add a message to an outbound mailbox queue.

add_inb_buffer
Callback to add a buffer to an inbound mailbox queue.

get_inb_message
Callback to get a message from an inbound mailbox queue.

map_inb
Callback to map RapidIO address region into local memory space.

unmap_inb
Callback to unmap RapidIO address region mapped with map_inb().

query_mport
Callback to query mport device attributes.

map_outb
Callback to map outbound address region into local memory space.

unmap_outb
Callback to unmap outbound RapidIO address region.

struct **rio_driver**

RIO driver info

Definition:

```
struct rio_driver {
    struct list_head node;
    char *name;
    const struct rio_device_id *id_table;
    int (*probe) (struct rio_dev * dev, const struct rio_device_id * id);
    void (*remove) (struct rio_dev * dev);
    void (*shutdown)(struct rio_dev *dev);
    int (*suspend) (struct rio_dev * dev, u32 state);
    int (*resume) (struct rio_dev * dev);
    int (*enable_wake) (struct rio_dev * dev, u32 state, int enable);
    struct device_driver driver;
};
```

Members

node
Node in list of drivers

name
RIO driver name

id_table
RIO device ids to be associated with this driver

probe
RIO device inserted

remove
RIO device removed

shutdown
shutdown notification callback

suspend
RIO device suspended

resume
RIO device awakened

enable_wake
RIO device enable wake event

driver
LDM driver struct

Description

Provides info on a RIO device driver for insertion/removal and power management purposes.

struct **rio_scan**

RIO enumeration and discovery operations

Definition:

```
struct rio_scan {
    struct module *owner;
    int (*enumerate)(struct rio_mport *mport, u32 flags);
    int (*discover)(struct rio_mport *mport, u32 flags);
};
```

Members

owner
The module owner of this structure

enumerate
Callback to perform RapidIO fabric enumeration.

discover
Callback to perform RapidIO fabric discovery.

struct **rio_scan_node**
list node to register RapidIO enumeration and discovery methods with RapidIO core.

Definition:

```
struct rio_scan_node {
    int mport_id;
    struct list_head node;
    struct rio_scan *ops;
};
```

Members

mport_id
ID of an mport (net) serviced by this enumerator

node
node in global list of registered enumerators

ops
RIO enumeration and discovery operations

70.4.2 Enumeration and Discovery

u16 **rio_destid_alloc**(struct *rio_net* *net)
Allocate next available destID for given network

Parameters

struct rio_net *net
RIO network

Description

Returns next available device destination ID for the specified RIO network. Marks allocated ID as one in use. Returns RIO_INVALID_DESTID if new destID is not available.

int **rio_destid_reserve**(struct *rio_net* *net, u16 destid)
Reserve the specified destID

Parameters

struct rio_net *net
RIO network

u16 destid
destID to reserve

Description

Tries to reserve the specified destID. Returns 0 if successful.

void **rio_destid_free**(struct *rio_net* *net, u16 destid)
free a previously allocated destID

Parameters

struct rio_net *net
RIO network

u16 destid
destID to free

Description

Makes the specified destID available for use.

u16 **rio_destid_first**(struct *rio_net* *net)
return first destID in use

Parameters

struct rio_net *net
RIO network

u16 **rio_destid_next**(struct *rio_net* *net, u16 from)
return next destID in use

Parameters

struct rio_net *net
RIO network

u16 from
destination ID from which search shall continue

u16 **rio_get_device_id**(struct *rio_mport* *port, u16 destid, u8 hopcount)
Get the base/extended device id for a device

Parameters

struct rio_mport *port
RIO master port

u16 destid
Destination ID of device

u8 hopcount
Hopcount to device

Description

Reads the base/extended device id from a device. Returns the 8/16-bit device ID.

void **rio_set_device_id**(struct *rio_mport* *port, u16 destid, u8 hopcount, u16 did)

Set the base/extended device id for a device

Parameters

struct rio_mport *port
RIO master port

u16 destid
Destination ID of device

u8 hopcount
Hopcount to device

u16 did
Device ID value to be written

Description

Writes the base/extended device id from a device.

int **rio_clear_locks**(struct *rio_net* *net)

Release all host locks and signal enumeration complete

Parameters

struct rio_net *net
RIO network to run on

Description

Marks the component tag CSR on each device with the enumeration complete flag. When complete, it then release the host locks on each device. Returns 0 on success or `-EINVAL` on failure.

int **rio_enum_host**(struct *rio_mport* *port)

Set host lock and initialize host destination ID

Parameters

struct rio_mport *port
Master port to issue transaction

Description

Sets the local host master port lock and destination ID register with the host device ID value. The host device ID value is provided by the platform. Returns `0` on success or `-1` on failure.

int **rio_device_has_destid**(struct *rio_mport* *port, int src_ops, int dst_ops)

Test if a device contains a destination ID register

Parameters

struct rio_mport *port
Master port to issue transaction

int src_ops
RIO device source operations

int dst_ops
RIO device destination operations

Description

Checks the provided **src_ops** and **dst_ops** for the necessary transaction capabilities that indicate whether or not a device will implement a destination ID register. Returns 1 if true or 0 if false.

void **rio_release_dev**(struct device *dev)

Frees a RIO device struct

Parameters

struct device *dev
LDM device associated with a RIO device struct

Description

Gets the RIO device struct associated a RIO device struct. The RIO device struct is freed.

int **rio_is_switch**(struct *rio_dev* *rdev)

Tests if a RIO device has switch capabilities

Parameters

struct rio_dev *rdev
RIO device

Description

Gets the RIO device Processing Element Features register contents and tests for switch capabilities. Returns 1 if the device is a switch or 0 if it is not a switch. The RIO device struct is freed.

struct *rio_dev* ***rio_setup_device**(struct *rio_net* *net, struct *rio_mport* *port, u16 destid, u8 hopcount, int do_enum)

Allocates and sets up a RIO device

Parameters

struct rio_net *net
RIO network

struct rio_mport *port
Master port to send transactions

u16 destid
Current destination ID

u8 hopcount
Current hopcount

int do_enum
Enumeration/Discovery mode flag

Description

Allocates a RIO device and configures fields based on configuration space contents. If device has a destination ID register, a destination ID is either assigned in enumeration mode or read from configuration space in discovery mode. If the device has switch capabilities, then a switch is allocated and configured appropriately. Returns a pointer to a RIO device on success or NULL on failure.

int **rio_sport_is_active**(struct *rio_dev* *rdev, int sp)

Tests if a switch port has an active connection.

Parameters

struct rio_dev *rdev
RapidIO device object

int sp
Switch port number

Description

Reads the port error status CSR for a particular switch port to determine if the port has an active link. Returns `RIO_PORT_N_ERR_STS_PORT_OK` if the port is active or `0` if it is inactive.

u16 **rio_get_host_deviceid_lock**(struct *rio_mport* *port, u8 hopcount)

Reads the Host Device ID Lock CSR on a device

Parameters

struct rio_mport *port
Master port to send transaction

u8 hopcount
Number of hops to the device

Description

Used during enumeration to read the Host Device ID Lock CSR on a RIO device. Returns the value of the lock register.

int **rio_enum_peer**(struct *rio_net* *net, struct *rio_mport* *port, u8 hopcount, struct *rio_dev* *prev, int prev_port)

Recursively enumerate a RIO network through a master port

Parameters

struct rio_net *net
RIO network being enumerated

struct rio_mport *port
Master port to send transactions

u8 hopcount
Number of hops into the network

struct rio_dev *prev
Previous RIO device connected to the enumerated one

int prev_port
Port on previous RIO device

Description

Recursively enumerates a RIO network. Transactions are sent via the master port passed in **port**.

int **rio_enum_complete**(struct *rio_mport* *port)

Tests if enumeration of a network is complete

Parameters

struct rio_mport *port
Master port to send transaction

Description

Tests the PGCCSR discovered bit for non-zero value (enumeration complete flag). Return `1` if enumeration is complete or `0` if enumeration is incomplete.

int **rio_disc_peer**(struct *rio_net* *net, struct *rio_mport* *port, u16 destid, u8 hopcount, struct *rio_dev* *prev, int prev_port)

Recursively discovers a RIO network through a master port

Parameters

struct rio_net *net
RIO network being discovered

struct rio_mport *port
Master port to send transactions

u16 destid
Current destination ID in network

u8 hopcount
Number of hops into the network

struct rio_dev *prev
previous rio_dev

int prev_port
previous port number

Description

Recursively discovers a RIO network. Transactions are sent via the master port passed in **port**.

int **rio_mport_is_active**(struct *rio_mport* *port)

Tests if master port link is active

Parameters

struct rio_mport *port
Master port to test

Description

Reads the port error status CSR for the master port to determine if the port has an active link. Returns `RIO_PORT_N_ERR_STS_PORT_OK` if the master port is active or `0` if it is inactive.

void **rio_update_route_tables**(struct *rio_net* *net)

Updates route tables in switches

Parameters

struct rio_net *net
RIO network to run update on

Description

For each enumerated device, ensure that each switch in a system has correct routing entries. Add routes for devices that where unknown during the first enumeration pass through the switch.

void **rio_init_em**(struct *rio_dev* *rdev)

Initializes RIO Error Management (for switches)

Parameters

struct rio_dev *rdev

RIO device

Description

For each enumerated switch, call device-specific error management initialization routine (if supplied by the switch driver).

int **rio_enum_mport**(struct *rio_mport* *mport, u32 flags)

Start enumeration through a master port

Parameters

struct rio_mport *mport

Master port to send transactions

u32 flags

Enumeration control flags

Description

Starts the enumeration process. If somebody has enumerated our master port device, then give up. If not and we have an active link, then start recursive peer enumeration. Returns `0` if enumeration succeeds or `-EBUSY` if enumeration fails.

void **rio_build_route_tables**(struct *rio_net* *net)

Generate route tables from switch route entries

Parameters

struct rio_net *net

RIO network to run route tables scan on

Description

For each switch device, generate a route table by copying existing route entries from the switch.

int **rio_disc_mport**(struct *rio_mport* *mport, u32 flags)

Start discovery through a master port

Parameters

struct rio_mport *mport

Master port to send transactions

u32 flags

discovery control flags

Description

Starts the discovery process. If we have an active link, then wait for the signal that enumeration is complete (if wait is allowed). When enumeration completion is signaled, start recursive peer discovery. Returns `0` if discovery succeeds or `-EBUSY` on failure.

int **rio_basic_attach**(void)

Parameters

void
no arguments

Description

When this enumeration/discovery method is loaded as a module this function registers its specific enumeration and discover routines for all available RapidIO mport devices. The "scan" command line parameter controls ability of the module to start RapidIO enumeration/discovery automatically.

Returns 0 for success or -EIO if unable to register itself.

This enumeration/discovery method cannot be unloaded and therefore does not provide a matching cleanup_module routine.

70.4.3 Driver functionality

int **rio_setup_inb_dbell**(struct *rio_mport* *mport, void *dev_id, struct resource *res, void (*dinb)(struct *rio_mport* *mport, void *dev_id, u16 src, u16 dst, u16 info))

bind inbound doorbell callback

Parameters

struct rio_mport *mport
RIO master port to bind the doorbell callback

void *dev_id
Device specific pointer to pass on event

struct resource *res
Doorbell message resource

void (*dinb) (struct rio_mport * mport, void *dev_id, u16 src, u16 dst, u16 info)
Callback to execute when doorbell is received

Description

Adds a doorbell resource/callback pair into a port's doorbell event list. Returns 0 if the request has been satisfied.

int **rio_chk_dev_route**(struct *rio_dev* *rdev, struct *rio_dev* **nrdev, int *npnum)

Validate route to the specified device.

Parameters

struct rio_dev *rdev
RIO device failed to respond

struct rio_dev **nrdev
Last active device on the route to rdev

int *npnum
nrdev's port number on the route to rdev

Description

Follows a route to the specified RIO device to determine the last available device (and corresponding RIO port) on the route.

int **rio_chk_dev_access**(struct *rio_dev* *rdev)

Validate access to the specified device.

Parameters

struct rio_dev *rdev
Pointer to RIO device control structure

int **rio_get_input_status**(struct *rio_dev* *rdev, int pnum, u32 *lnkresp)

Sends a Link-Request/Input-Status control symbol and returns link-response (if requested).

Parameters

struct rio_dev *rdev
RIO devive to issue Input-status command

int pnum
Device port number to issue the command

u32 *lnkresp
Response from a link partner

int **rio_clr_err_stopped**(struct *rio_dev* *rdev, u32 pnum, u32 err_status)

Clears port Error-stopped states.

Parameters

struct rio_dev *rdev
Pointer to RIO device control structure

u32 pnum
Switch port number to clear errors

u32 err_status
port error status (if 0 reads register from device)

Description

TODO: Currently this routine is not compatible with recovery process specified for idt_gen3 RapidIO switch devices. It has to be reviewed to implement universal recovery process that is compatible full range off available devices. IDT gen3 switch driver now implements HW-specific error handler that issues soft port reset to the port to reset ERR_STOP bits and ackIDs.

int **rio_std_route_add_entry**(struct *rio_mport* *mport, u16 destid, u8 hopcount, u16 table, u16 route_destid, u8 route_port)

Add switch route table entry using standard registers defined in RIO specification rev.1.3

Parameters

struct rio_mport *mport
Master port to issue transaction

u16 destid
Destination ID of the device

u8 hopcount
Number of switch hops to the device

u16 table
routing table ID (global or port-specific)

u16 route_destid
destID entry in the RT

u8 route_port
destination port for specified destID

int **rio_std_route_get_entry**(struct *rio_mport* *mport, u16 destid, u8 hopcount, u16 table, u16 route_destid, u8 *route_port)

Read switch route table entry (port number) associated with specified destID using standard registers defined in RIO specification rev.1.3

Parameters

struct rio_mport *mport
Master port to issue transaction

u16 destid
Destination ID of the device

u8 hopcount
Number of switch hops to the device

u16 table
routing table ID (global or port-specific)

u16 route_destid
destID entry in the RT

u8 *route_port
returned destination port for specified destID

int **rio_std_route_clr_table**(struct *rio_mport* *mport, u16 destid, u8 hopcount, u16 table)

Clear swotch route table using standard registers defined in RIO specification rev.1.3.

Parameters

struct rio_mport *mport
Master port to issue transaction

u16 destid
Destination ID of the device

u8 hopcount
Number of switch hops to the device

u16 table
routing table ID (global or port-specific)

struct *rio_mport* ***rio_find_mport**(int mport_id)

find RIO mport by its ID

Parameters

int mport_id
number (ID) of mport device

Description

Given a RIO mport number, the desired mport is located in the global list of mports. If the mport is found, a pointer to its data structure is returned. If no mport is found, `NULL` is returned.

int **`rio_mport_scan`**(int mport_id)

execute enumeration/discovery on the specified mport

Parameters

`int mport_id`
number (ID) of mport device

`RIO_LOP_READ`

`RIO_LOP_READ (size, type, len)`

Generate rio_local_read_config_* functions

Parameters

`size`
Size of configuration space read (8, 16, 32 bits)

`type`
C type of value argument

`len`
Length of configuration space read (1, 2, 4 bytes)

Description

Generates rio_local_read_config_* functions used to access configuration space registers on the local device.

`RIO_LOP_WRITE`

`RIO_LOP_WRITE (size, type, len)`

Generate rio_local_write_config_* functions

Parameters

`size`
Size of configuration space write (8, 16, 32 bits)

`type`
C type of value argument

`len`
Length of configuration space write (1, 2, 4 bytes)

Description

Generates rio_local_write_config_* functions used to access configuration space registers on the local device.

`RIO_OP_READ`

`RIO_OP_READ (size, type, len)`

Generate rio_mport_read_config_* functions

Parameters

size
Size of configuration space read (8, 16, 32 bits)

type
C type of value argument

len
Length of configuration space read (1, 2, 4 bytes)

Description

Generates rio_mport_read_config_* functions used to access configuration space registers on the local device.

RIO_OP_WRITE

`RIO_OP_WRITE (size, type, len)`

Generate rio_mport_write_config_* functions

Parameters

size
Size of configuration space write (8, 16, 32 bits)

type
C type of value argument

len
Length of configuration space write (1, 2, 4 bytes)

Description

Generates rio_mport_write_config_* functions used to access configuration space registers on the local device.

70.4.4 Device model support

const struct rio_device_id ***rio_match_device**(const struct rio_device_id *id, const struct *rio_dev* *rdev)

Tell if a RIO device has a matching RIO device id structure

Parameters

const struct rio_device_id *id
the RIO device id structure to match against

const struct rio_dev *rdev
the RIO device structure to match against

Used from driver probe and bus matching to check whether a RIO device matches a device id structure provided by a RIO driver. Returns the matching `struct rio_device_id` or `NULL` if there is no match.

int **rio_device_probe**(struct device *dev)

Tell if a RIO device structure has a matching RIO device id structure

Parameters

struct device *dev
the RIO device structure to match against

Description

return 0 and set rio_dev->driver when drv claims rio_dev, else error

void **rio_device_remove**(struct device *dev)
Remove a RIO device from the system

Parameters

struct device *dev
the RIO device structure to match against

Description

Remove a RIO device from the system. If it has an associated driver, then run the driver remove() method. Then update the reference count.

int **rio_match_bus**(struct device *dev, struct device_driver *drv)
Tell if a RIO device structure has a matching RIO driver device id structure

Parameters

struct device *dev
the standard device structure to match against

struct device_driver *drv
the standard driver structure containing the ids to match against

Used by a driver to check whether a RIO device present in the system is in its list of supported devices. Returns 1 if there is a matching `struct rio_device_id` or 0 if there is no match.

int **rio_bus_init**(void)
Register the RapidIO bus with the device model

Parameters

void
no arguments

Description

Registers the RIO mport device class and RIO bus type with the Linux device model.

70.4.5 PPC32 support

int **fsl_local_config_read**(struct *rio_mport* *mport, int index, u32 offset, int len, u32 *data)
Generate a MPC85xx local config space read

Parameters

struct rio_mport *mport
RapidIO master port info

int index
ID of RapdiIO interface

u32 offset
Offset into configuration space

int len
Length (in bytes) of the maintenance transaction

u32 *data
Value to be read into

Description

Generates a MPC85xx local configuration space read. Returns `0` on success or `-EINVAL` on failure.

int **fsl_local_config_write**(struct *rio_mport* *mport, int index, u32 offset, int len, u32 data)
Generate a MPC85xx local config space write

Parameters

struct rio_mport *mport
RapidIO master port info

int index
ID of RapdiIO interface

u32 offset
Offset into configuration space

int len
Length (in bytes) of the maintenance transaction

u32 data
Value to be written

Description

Generates a MPC85xx local configuration space write. Returns `0` on success or `-EINVAL` on failure.

int **fsl_rio_config_read**(struct *rio_mport* *mport, int index, u16 destid, u8 hopcount, u32 offset, int len, u32 *val)
Generate a MPC85xx read maintenance transaction

Parameters

struct rio_mport *mport
RapidIO master port info

int index
ID of RapdiIO interface

u16 destid
Destination ID of transaction

u8 hopcount
Number of hops to target device

u32 offset
Offset into configuration space

int len
Length (in bytes) of the maintenance transaction

u32 *val
Location to be read into

Description

Generates a MPC85xx read maintenance transaction. Returns `0` on success or `-EINVAL` on failure.

int **fsl_rio_config_write**(struct *rio_mport* *mport, int index, u16 destid, u8 hopcount, u32 offset, int len, u32 val)

Generate a MPC85xx write maintenance transaction

Parameters

struct rio_mport *mport
RapidIO master port info

int index
ID of RapdiIO interface

u16 destid
Destination ID of transaction

u8 hopcount
Number of hops to target device

u32 offset
Offset into configuration space

int len
Length (in bytes) of the maintenance transaction

u32 val
Value to be written

Description

Generates an MPC85xx write maintenance transaction. Returns `0` on success or `-EINVAL` on failure.

int **fsl_rio_setup**(struct platform_device *dev)

Setup Freescale PowerPC RapidIO interface

Parameters

struct platform_device *dev
platform_device pointer

Description

Initializes MPC85xx RapidIO hardware interface, configures master port with system-specific info, and registers the master port with the RapidIO subsystem.

70.5 Credits

The following people have contributed to the RapidIO subsystem directly or indirectly:

1. Matt Portermporter@kernel.crashing.org
2. Randy Vinsonrvinson@mvista.com
3. Dan Malekdan@embeddedalley.com

The following people have contributed to this document:

1. Matt Portermporter@kernel.crashing.org

CHAPTER

SEVENTYONE

RELIABILITY, AVAILABILITY AND SERVICEABILITY

71.1 RAS concepts

Reliability, Availability and Serviceability (RAS) is a concept used on servers meant to measure their robustness.

Reliability
is the probability that a system will produce correct outputs.

- Generally measured as Mean Time Between Failures (MTBF)
- Enhanced by features that help to avoid, detect and repair hardware faults

Availability
is the probability that a system is operational at a given time

- Generally measured as a percentage of downtime per a period of time
- Often uses mechanisms to detect and correct hardware faults in runtime;

Serviceability (or maintainability)
is the simplicity and speed with which a system can be repaired or maintained

- Generally measured on Mean Time Between Repair (MTBR)

71.1.1 Improving RAS

In order to reduce systems downtime, a system should be capable of detecting hardware errors, and, when possible correcting them in runtime. It should also provide mechanisms to detect hardware degradation, in order to warn the system administrator to take the action of replacing a component before it causes data loss or system downtime.

Among the monitoring measures, the most usual ones include:

- CPU – detect errors at instruction execution and at L1/L2/L3 caches;
- Memory – add error correction logic (ECC) to detect and correct errors;
- I/O – add CRC checksums for transferred data;
- Storage – RAID, journal file systems, checksums, Self-Monitoring, Analysis and Reporting Technology (SMART).

By monitoring the number of occurrences of error detections, it is possible to identify if the probability of hardware errors is increasing, and, on such case, do a preventive maintenance to replace a degraded component while those errors are correctable.

71.1.2 Types of errors

Most mechanisms used on modern systems use technologies like Hamming Codes that allow error correction when the number of errors on a bit packet is below a threshold. If the number of errors is above, those mechanisms can indicate with a high degree of confidence that an error happened, but they can't correct.

Also, sometimes an error occur on a component that it is not used. For example, a part of the memory that it is not currently allocated.

That defines some categories of errors:

- **Correctable Error (CE)** - the error detection mechanism detected and corrected the error. Such errors are usually not fatal, although some Kernel mechanisms allow the system administrator to consider them as fatal.
- **Uncorrected Error (UE)** - the amount of errors happened above the error correction threshold, and the system was unable to auto-correct.
- **Fatal Error** - when an UE error happens on a critical component of the system (for example, a piece of the Kernel got corrupted by an UE), the only reliable way to avoid data corruption is to hang or reboot the machine.
- **Non-fatal Error** - when an UE error happens on an unused component, like a CPU in power down state or an unused memory bank, the system may still run, eventually replacing the affected hardware by a hot spare, if available.

  Also, when an error happens on a userspace process, it is also possible to kill such process and let userspace restart it.

The mechanism for handling non-fatal errors is usually complex and may require the help of some userspace application, in order to apply the policy desired by the system administrator.

71.1.3 Identifying a bad hardware component

Just detecting a hardware flaw is usually not enough, as the system needs to pinpoint to the minimal replaceable unit (MRU) that should be exchanged to make the hardware reliable again.

So, it requires not only error logging facilities, but also mechanisms that will translate the error message to the silkscreen or component label for the MRU.

Typically, it is very complex for memory, as modern CPUs interlace memory from different memory modules, in order to provide a better performance. The DMI BIOS usually have a list of memory module labels, with can be obtained using the `dmidecode` tool. For example, on a desktop machine, it shows:

```
Memory Device
        Total Width: 64 bits
        Data Width: 64 bits
        Size: 16384 MB
        Form Factor: SODIMM
        Set: None
        Locator: ChannelA-DIMM0
        Bank Locator: BANK 0
        Type: DDR4
        Type Detail: Synchronous
```

```
        Speed: 2133 MHz
        Rank: 2
        Configured Clock Speed: 2133 MHz
```

On the above example, a DDR4 SO-DIMM memory module is located at the system's memory labeled as "BANK 0", as given by the *bank locator* field. Please notice that, on such system, the *total width* is equal to the *data width*. It means that such memory module doesn't have error detection/correction mechanisms.

Unfortunately, not all systems use the same field to specify the memory bank. On this example, from an older server, `dmidecode` shows:

```
Memory Device
        Array Handle: 0x1000
        Error Information Handle: Not Provided
        Total Width: 72 bits
        Data Width: 64 bits
        Size: 8192 MB
        Form Factor: DIMM
        Set: 1
        Locator: DIMM_A1
        Bank Locator: Not Specified
        Type: DDR3
        Type Detail: Synchronous Registered (Buffered)
        Speed: 1600 MHz
        Rank: 2
        Configured Clock Speed: 1600 MHz
```

There, the DDR3 RDIMM memory module is located at the system's memory labeled as "DIMM_A1", as given by the *locator* field. Please notice that this memory module has 64 bits of *data width* and 72 bits of *total width*. So, it has 8 extra bits to be used by error detection and correction mechanisms. Such kind of memory is called Error-correcting code memory (ECC memory).

To make things even worse, it is not uncommon that systems with different labels on their system's board to use exactly the same BIOS, meaning that the labels provided by the BIOS won't match the real ones.

71.1.4 ECC memory

As mentioned in the previous section, ECC memory has extra bits to be used for error correction. In the above example, a memory module has 64 bits of *data width*, and 72 bits of *total width*. The extra 8 bits which are used for the error detection and correction mechanisms are referred to as the *syndrome*[1][2].

[1] Please notice that several memory controllers allow operation on a mode called "Lock-Step", where it groups two memory modules together, doing 128-bit reads/writes. That gives 16 bits for error correction, with significantly improves the error correction mechanism, at the expense that, when an error happens, there's no way to know what memory module is to blame. So, it has to blame both memory modules.

[2] Some memory controllers also allow using memory in mirror mode. On such mode, the same data is written to two memory modules. At read, the system checks both memory modules, in order to check if both provide identical data. On such configuration, when an error happens, there's no way to know what memory module is to blame. So, it has to blame both memory modules (or 4 memory modules, if the system is also on Lock-step mode).

So, when the cpu requests the memory controller to write a word with *data width*, the memory controller calculates the *syndrome* in real time, using Hamming code, or some other error correction code, like SECDED+, producing a code with *total width* size. Such code is then written on the memory modules.

At read, the *total width* bits code is converted back, using the same ECC code used on write, producing a word with *data width* and a *syndrome*. The word with *data width* is sent to the CPU, even when errors happen.

The memory controller also looks at the *syndrome* in order to check if there was an error, and if the ECC code was able to fix such error. If the error was corrected, a Corrected Error (CE) happened. If not, an Uncorrected Error (UE) happened.

The information about the CE/UE errors is stored on some special registers at the memory controller and can be accessed by reading such registers, either by BIOS, by some special CPUs or by Linux EDAC driver. On x86 64 bit CPUs, such errors can also be retrieved via the Machine Check Architecture (MCA)[3].

71.2 EDAC - Error Detection And Correction

Note: ”bluesmoke” was the name for this device driver subsystem when it was ”out-of-tree” and maintained at http://bluesmoke.sourceforge.net. That site is mostly archaic now and can be used only for historical purposes.

When the subsystem was pushed upstream for the first time, on Kernel 2.6.16, it was renamed to `EDAC`.

71.2.1 Purpose

The `edac` kernel module's goal is to detect and report hardware errors that occur within the computer system running under linux.

71.2.2 Memory

Memory Correctable Errors (CE) and Uncorrectable Errors (UE) are the primary errors being harvested. These types of errors are harvested by the `edac_mc` device.

Detecting CE events, then harvesting those events and reporting them, **can** but must not necessarily be a predictor of future UE events. With CE events only, the system can and will continue to operate as no data has been damaged yet.

However, preventive maintenance and proactive part replacement of memory modules exhibiting CEs can reduce the likelihood of the dreaded UE events and system panics.

[3] For more details about the Machine Check Architecture (MCA), please read Documentation/arch/x86/x86_64/machinecheck.rst at the Kernel tree.

71.2.3 Other hardware elements

A new feature for EDAC, the `edac_device` class of device, was added in the 2.6.23 version of the kernel.

This new device type allows for non-memory type of ECC hardware detectors to have their states harvested and presented to userspace via the sysfs interface.

Some architectures have ECC detectors for L1, L2 and L3 caches, along with DMA engines, fabric switches, main data path switches, interconnections, and various other hardware data paths. If the hardware reports it, then a edac_device device probably can be constructed to harvest and present that to userspace.

71.2.4 PCI bus scanning

In addition, PCI devices are scanned for PCI Bus Parity and SERR Errors in order to determine if errors are occurring during data transfers.

The presence of PCI Parity errors must be examined with a grain of salt. There are several add-in adapters that do **not** follow the PCI specification with regards to Parity generation and reporting. The specification says the vendor should tie the parity status bits to 0 if they do not intend to generate parity. Some vendors do not do this, and thus the parity bit can ”float” giving false positives.

There is a PCI device attribute located in sysfs that is checked by the EDAC PCI scanning code. If that attribute is set, PCI parity/error scanning is skipped for that device. The attribute is:

```
broken_parity_status
```

and is located in `/sys/devices/pci<XXX>/0000:XX:YY.Z` directories for PCI devices.

71.2.5 Versioning

EDAC is composed of a ”core” module (`edac_core.ko`) and several Memory Controller (MC) driver modules. On a given system, the CORE is loaded and one MC driver will be loaded. Both the CORE and the MC driver (or `edac_device` driver) have individual versions that reflect current release level of their respective modules.

Thus, to ”report” on what version a system is running, one must report both the CORE's and the MC driver's versions.

71.2.6 Loading

If `edac` was statically linked with the kernel then no loading is necessary. If `edac` was built as modules then simply modprobe the `edac` pieces that you need. You should be able to modprobe hardware-specific modules and have the dependencies load the necessary core modules.

Example:

```
$ modprobe amd76x_edac
```

loads both the `amd76x_edac.ko` memory controller module and the `edac_mc.ko` core module.

71.2.7 Sysfs interface

EDAC presents a `sysfs` interface for control and reporting purposes. It lives in the /sys/devices/system/edac directory.

Within this directory there currently reside 2 components:

mc	memory controller(s) system
pci	PCI control and status system

71.2.8 Memory Controller (mc) Model

Each `mc` device controls a set of memory modules[4]. These modules are laid out in a Chip-Select Row (`csrowX`) and Channel table (`chX`). There can be multiple csrows and multiple channels.

Memory controllers allow for several csrows, with 8 csrows being a typical value. Yet, the actual number of csrows depends on the layout of a given motherboard, memory controller and memory module characteristics.

Dual channels allow for dual data length (e. g. 128 bits, on 64 bit systems) data transfers to/from the CPU from/to memory. Some newer chipsets allow for more than 2 channels, like Fully Buffered DIMMs (FB-DIMMs) memory controllers. The following example will assume 2 channels:

CS Rows	Channels	
	`ch0`	`ch1`
	DIMM_A0	**DIMM_B0**
`csrow0`	rank0	rank0
`csrow1`	rank1	rank1
	DIMM_A1	**DIMM_B1**
`csrow2`	rank0	rank0
`csrow3`	rank1	rank1

In the above example, there are 4 physical slots on the motherboard for memory DIMMs:

DIMM_A0	DIMM_B0
DIMM_A1	DIMM_B1

Labels for these slots are usually silk-screened on the motherboard. Slots labeled `A` are channel 0 in this example. Slots labeled `B` are channel 1. Notice that there are two csrows possible on a physical DIMM. These csrows are allocated their csrow assignment based on the slot into which the memory DIMM is placed. Thus, when 1 DIMM is placed in each Channel, the csrows cross both DIMMs.

Memory DIMMs come single or dual "ranked". A rank is a populated csrow. In the example above 2 dual ranked DIMMs are similarly placed. Thus, both csrow0 and csrow1 are populated.

[4] Nowadays, the term DIMM (Dual In-line Memory Module) is widely used to refer to a memory module, although there are other memory packaging alternatives, like SO-DIMM, SIMM, etc. The UEFI specification (Version 2.7) defines a memory module in the Common Platform Error Record (CPER) section to be an SMBIOS Memory Device (Type 17). Along this document, and inside the EDAC subsystem, the term "dimm" is used for all memory modules, even when they use a different kind of packaging.

On the other hand, when 2 single ranked DIMMs are placed in slots DIMM_A0 and DIMM_B0, then they will have just one csrow (csrow0) and csrow1 will be empty. The pattern repeats itself for csrow2 and csrow3. Also note that some memory controllers don't have any logic to identify the memory module, see `rankX` directories below.

The representation of the above is reflected in the directory tree in EDAC's sysfs interface. Starting in directory `/sys/devices/system/edac/mc`, each memory controller will be represented by its own `mcX` directory, where `X` is the index of the MC:

```
..../edac/mc/
           |
           |->mc0
           |->mc1
           |->mc2
           ....
```

Under each `mcX` directory each `csrowX` is again represented by a `csrowX`, where `X` is the csrow index:

```
.../mc/mc0/
        |
        |->csrow0
        |->csrow2
        |->csrow3
        ....
```

Notice that there is no csrow1, which indicates that csrow0 is composed of a single ranked DIMMs. This should also apply in both Channels, in order to have dual-channel mode be operational. Since both csrow2 and csrow3 are populated, this indicates a dual ranked set of DIMMs for channels 0 and 1.

Within each of the `mcX` and `csrowX` directories are several EDAC control and attribute files.

71.2.9 `mcX` directories

In `mcX` directories are EDAC control and attribute files for this `X` instance of the memory controllers.

For a description of the sysfs API, please see:

> Documentation/ABI/testing/sysfs-devices-edac

71.2.10 `dimmX` or `rankX` directories

The recommended way to use the EDAC subsystem is to look at the information provided by the `dimmX` or `rankX` directories[5].

[5] On some systems, the memory controller doesn't have any logic to identify the memory module. On such systems, the directory is called `rankX` and works on a similar way as the `csrowX` directories. On modern Intel memory controllers, the memory controller identifies the memory modules directly. On such systems, the directory is called `dimmX`.

A typical EDAC system has the following structure under `/sys/devices/system/edac/`[6]:

```
/sys/devices/system/edac/
├── mc
│   ├── mc0
│   │   ├── ce_count
│   │   ├── ce_noinfo_count
│   │   ├── dimm0
│   │   │   ├── dimm_ce_count
│   │   │   ├── dimm_dev_type
│   │   │   ├── dimm_edac_mode
│   │   │   ├── dimm_label
│   │   │   ├── dimm_location
│   │   │   ├── dimm_mem_type
│   │   │   ├── dimm_ue_count
│   │   │   ├── size
│   │   │   └── uevent
│   │   ├── max_location
│   │   ├── mc_name
│   │   ├── reset_counters
│   │   ├── seconds_since_reset
│   │   ├── size_mb
│   │   ├── ue_count
│   │   ├── ue_noinfo_count
│   │   └── uevent
│   ├── mc1
│   │   ├── ce_count
│   │   ├── ce_noinfo_count
│   │   ├── dimm0
│   │   │   ├── dimm_ce_count
│   │   │   ├── dimm_dev_type
│   │   │   ├── dimm_edac_mode
│   │   │   ├── dimm_label
│   │   │   ├── dimm_location
│   │   │   ├── dimm_mem_type
│   │   │   ├── dimm_ue_count
│   │   │   ├── size
│   │   │   └── uevent
│   │   ├── max_location
│   │   ├── mc_name
│   │   ├── reset_counters
│   │   ├── seconds_since_reset
│   │   ├── size_mb
│   │   ├── ue_count
│   │   ├── ue_noinfo_count
│   │   └── uevent
│   └── uevent
```

[6] There are also some `power` directories and `subsystem` symlinks inside the sysfs mapping that are automatically created by the sysfs subsystem. Currently, they serve no purpose.

```
└── uevent
```

In the `dimmX` directories are EDAC control and attribute files for this `X` memory module:

- `size` - Total memory managed by this csrow attribute file

  This attribute file displays, in count of megabytes, the memory that this csrow contains.

- `dimm_ue_count` - Uncorrectable Errors count attribute file

  This attribute file displays the total count of uncorrectable errors that have occurred on this DIMM. If panic_on_ue is set this counter will not have a chance to increment, since EDAC will panic the system.

- `dimm_ce_count` - Correctable Errors count attribute file

  This attribute file displays the total count of correctable errors that have occurred on this DIMM. This count is very important to examine. CEs provide early indications that a DIMM is beginning to fail. This count field should be monitored for non-zero values and report such information to the system administrator.

- `dimm_dev_type` - Device type attribute file

  This attribute file will display what type of DRAM device is being utilized on this DIMM. Examples:

  - x1
  - x2
  - x4
  - x8

- `dimm_edac_mode` - EDAC Mode of operation attribute file

  This attribute file will display what type of Error detection and correction is being utilized.

- `dimm_label` - memory module label control file

  This control file allows this DIMM to have a label assigned to it. With this label in the module, when errors occur the output can provide the DIMM label in the system log. This becomes vital for panic events to isolate the cause of the UE event.

  DIMM Labels must be assigned after booting, with information that correctly identifies the physical slot with its silk screen label. This information is currently very motherboard specific and determination of this information must occur in userland at this time.

- `dimm_location` - location of the memory module

  The location can have up to 3 levels, and describe how the memory controller identifies the location of a memory module. Depending on the type of memory and memory controller, it can be:

  - *csrow* and *channel* - used when the memory controller doesn't identify a single DIMM - e. g. in `rankX` dir;

  - *branch*, *channel*, *slot* - typically used on FB-DIMM memory controllers;
  - *channel*, *slot* - used on Nehalem and newer Intel drivers.

- `dimm_mem_type` - Memory Type attribute file

  This attribute file will display what type of memory is currently on this csrow. Normally, either buffered or unbuffered memory. Examples:

  - Registered-DDR
  - Unbuffered-DDR

71.2.11 `csrowX` directories

When CONFIG_EDAC_LEGACY_SYSFS is enabled, sysfs will contain the `csrowX` directories. As this API doesn't work properly for Rambus, FB-DIMMs and modern Intel Memory Controllers, this is being deprecated in favor of `dimmX` directories.

In the `csrowX` directories are EDAC control and attribute files for this `X` instance of csrow:

- `ue_count` - Total Uncorrectable Errors count attribute file

  This attribute file displays the total count of uncorrectable errors that have occurred on this csrow. If panic_on_ue is set this counter will not have a chance to increment, since EDAC will panic the system.

- `ce_count` - Total Correctable Errors count attribute file

  This attribute file displays the total count of correctable errors that have occurred on this csrow. This count is very important to examine. CEs provide early indications that a DIMM is beginning to fail. This count field should be monitored for non-zero values and report such information to the system administrator.

- `size_mb` - Total memory managed by this csrow attribute file

  This attribute file displays, in count of megabytes, the memory that this csrow contains.

- `mem_type` - Memory Type attribute file

  This attribute file will display what type of memory is currently on this csrow. Normally, either buffered or unbuffered memory. Examples:

  - Registered-DDR
  - Unbuffered-DDR

- `edac_mode` - EDAC Mode of operation attribute file

  This attribute file will display what type of Error detection and correction is being utilized.

- `dev_type` - Device type attribute file

  This attribute file will display what type of DRAM device is being utilized on this DIMM. Examples:

  - x1
  - x2

  - x4
  - x8

- `ch0_ce_count` - Channel 0 CE Count attribute file

  This attribute file will display the count of CEs on this DIMM located in channel 0.

- `ch0_ue_count` - Channel 0 UE Count attribute file

  This attribute file will display the count of UEs on this DIMM located in channel 0.

- `ch0_dimm_label` - Channel 0 DIMM Label control file

  This control file allows this DIMM to have a label assigned to it. With this label in the module, when errors occur the output can provide the DIMM label in the system log. This becomes vital for panic events to isolate the cause of the UE event.

  DIMM Labels must be assigned after booting, with information that correctly identifies the physical slot with its silk screen label. This information is currently very motherboard specific and determination of this information must occur in userland at this time.

- `ch1_ce_count` - Channel 1 CE Count attribute file

  This attribute file will display the count of CEs on this DIMM located in channel 1.

- `ch1_ue_count` - Channel 1 UE Count attribute file

  This attribute file will display the count of UEs on this DIMM located in channel 0.

- `ch1_dimm_label` - Channel 1 DIMM Label control file

  This control file allows this DIMM to have a label assigned to it. With this label in the module, when errors occur the output can provide the DIMM label in the system log. This becomes vital for panic events to isolate the cause of the UE event.

  DIMM Labels must be assigned after booting, with information that correctly identifies the physical slot with its silk screen label. This information is currently very motherboard specific and determination of this information must occur in userland at this time.

71.2.12 System Logging

If logging for UEs and CEs is enabled, then system logs will contain information indicating that errors have been detected:

```
EDAC MC0: CE page 0x283, offset 0xce0, grain 8, syndrome 0x6ec3, row 0,␣
↪channel 1 "DIMM_B1": amd76x_edac
EDAC MC0: CE page 0x1e5, offset 0xfb0, grain 8, syndrome 0xb741, row 0,␣
↪channel 1 "DIMM_B1": amd76x_edac
```

The structure of the message is:

Content	Example
The memory controller	MC0
Error type	CE
Memory page	0x283
Offset in the page	0xce0
The byte granularity or resolution of the error	grain 8
The error syndrome	0xb741
Memory row	row 0
Memory channel	channel 1
DIMM label, if set prior	DIMM B1
And then an optional, driver-specific message that may have additional information.	

Both UEs and CEs with no info will lack all but memory controller, error type, a notice of "no info" and then an optional, driver-specific error message.

71.2.13 PCI Bus Parity Detection

On Header Type 00 devices, the primary status is looked at for any parity error regardless of whether parity is enabled on the device or not. (The spec indicates parity is generated in some cases). On Header Type 01 bridges, the secondary status register is also looked at to see if parity occurred on the bus on the other side of the bridge.

71.2.14 Sysfs configuration

Under `/sys/devices/system/edac/pci` are control and attribute files as follows:

- `check_pci_parity` - Enable/Disable PCI Parity checking control file

  This control file enables or disables the PCI Bus Parity scanning operation. Writing a 1 to this file enables the scanning. Writing a 0 to this file disables the scanning.

  Enable:

  ```
  echo "1" >/sys/devices/system/edac/pci/check_pci_parity
  ```

  Disable:

  ```
  echo "0" >/sys/devices/system/edac/pci/check_pci_parity
  ```

- `pci_parity_count` - Parity Count

  This attribute file will display the number of parity errors that have been detected.

71.2.15 Module parameters

- `edac_mc_panic_on_ue` - Panic on UE control file

  An uncorrectable error will cause a machine panic. This is usually desirable. It is a bad idea to continue when an uncorrectable error occurs - it is indeterminate what was uncorrected and the operating system context might be so mangled that continuing will lead to further corruption. If the kernel has MCE configured, then EDAC will never notice the UE.

  LOAD TIME:

  ```
  module/kernel parameter: edac_mc_panic_on_ue=[0|1]
  ```

  RUN TIME:

  ```
  echo "1" > /sys/module/edac_core/parameters/edac_mc_panic_on_ue
  ```

- `edac_mc_log_ue` - Log UE control file

  Generate kernel messages describing uncorrectable errors. These errors are reported through the system message log system. UE statistics will be accumulated even when UE logging is disabled.

  LOAD TIME:

  ```
  module/kernel parameter: edac_mc_log_ue=[0|1]
  ```

  RUN TIME:

  ```
  echo "1" > /sys/module/edac_core/parameters/edac_mc_log_ue
  ```

- `edac_mc_log_ce` - Log CE control file

  Generate kernel messages describing correctable errors. These errors are reported through the system message log system. CE statistics will be accumulated even when CE logging is disabled.

  LOAD TIME:

  ```
  module/kernel parameter: edac_mc_log_ce=[0|1]
  ```

  RUN TIME:

  ```
  echo "1" > /sys/module/edac_core/parameters/edac_mc_log_ce
  ```

- `edac_mc_poll_msec` - Polling period control file

  The time period, in milliseconds, for polling for error information. Too small a value wastes resources. Too large a value might delay necessary handling of errors and might loose valuable information for locating the error. 1000 milliseconds (once each second) is the current default. Systems which require all the bandwidth they can get, may increase this.

  LOAD TIME:

```
module/kernel parameter: edac_mc_poll_msec=[0|1]
```

RUN TIME:

```
echo "1000" > /sys/module/edac_core/parameters/edac_mc_poll_msec
```

- `panic_on_pci_parity` - Panic on PCI PARITY Error

  This control file enables or disables panicking when a parity error has been detected.

  module/kernel parameter:

```
edac_panic_on_pci_pe=[0|1]
```

  Enable:

```
echo "1" > /sys/module/edac_core/parameters/edac_panic_on_pci_pe
```

  Disable:

```
echo "0" > /sys/module/edac_core/parameters/edac_panic_on_pci_pe
```

71.2.16 EDAC device type

In the header file, edac_pci.h, there is a series of edac_device structures and APIs for the EDAC_DEVICE.

User space access to an edac_device is through the sysfs interface.

At the location `/sys/devices/system/edac` (sysfs) new edac_device devices will appear.

There is a three level tree beneath the above `edac` directory. For example, the `test_device_edac` device (found at the http://bluesmoke.sourceforget.net website) installs itself as:

```
/sys/devices/system/edac/test-instance
```

in this directory are various controls, a symlink and one or more `instance` directories.

The standard default controls are:

log_ce	boolean to log CE events
log_ue	boolean to log UE events
panic_on_ue	boolean to `panic` the system if an UE is encountered (default off, can be set true via startup script)
poll_msec	time period between POLL cycles for events

The test_device_edac device adds at least one of its own custom control:

test_bits	which in the current test driver does nothing but show how it is installed. A ported driver can add one or more such controls and/or attributes for specific uses. One out-of-tree driver uses controls here to allow for ERROR INJECTION operations to hardware injection registers

The symlink points to the 'struct dev' that is registered for this edac_device.

71.2.17 Instances

One or more instance directories are present. For the `test_device_edac` case:

test-instance0

In this directory there are two default counter attributes, which are totals of counter in deeper subdirectories.

ce_count	total of CE events of subdirectories
ue_count	total of UE events of subdirectories

71.2.18 Blocks

At the lowest directory level is the `block` directory. There can be 0, 1 or more blocks specified in each instance:

test-block0

In this directory the default attributes are:

ce_count	which is counter of CE events for this `block` of hardware being monitored
ue_count	which is counter of UE events for this `block` of hardware being monitored

The `test_device_edac` device adds 4 attributes and 1 control:

test-block-bits-0	for every POLL cycle this counter is incremented
test-block-bits-1	every 10 cycles, this counter is bumped once, and test-block-bits-0 is set to 0
test-block-bits-2	every 100 cycles, this counter is bumped once, and test-block-bits-1 is set to 0
test-block-bits-3	every 1000 cycles, this counter is bumped once, and test-block-bits-2 is set to 0

reset-counters	writing ANY thing to this control will reset all the above counters.

Use of the `test_device_edac` driver should enable any others to create their own unique drivers for their hardware systems.

The `test_device_edac` sample driver is located at the http://bluesmoke.sourceforge.net project site for EDAC.

71.2.19 Usage of EDAC APIs on Nehalem and newer Intel CPUs

On older Intel architectures, the memory controller was part of the North Bridge chipset. Nehalem, Sandy Bridge, Ivy Bridge, Haswell, Sky Lake and newer Intel architectures integrated an enhanced version of the memory controller (MC) inside the CPUs.

This chapter will cover the differences of the enhanced memory controllers found on newer Intel CPUs, such as `i7core_edac`, `sb_edac` and `sbx_edac` drivers.

Note: The Xeon E7 processor families use a separate chip for the memory controller, called Intel Scalable Memory Buffer. This section doesn't apply for such families.

---

1) There is one Memory Controller per Quick Patch Interconnect (QPI). At the driver, the term "socket" means one QPI. This is associated with a physical CPU socket.

   Each MC have 3 physical read channels, 3 physical write channels and 3 logic channels. The driver currently sees it as just 3 channels. Each channel can have up to 3 DIMMs.

   The minimum known unity is DIMMs. There are no information about csrows. As EDAC API maps the minimum unity is csrows, the driver sequentially maps channel/DIMM into different csrows.

   For example, supposing the following layout:

```
Ch0 phy rd0, wr0 (0x063f4031): 2 ranks, UDIMMs
  dimm 0 1024 Mb offset: 0, bank: 8, rank: 1, row: 0x4000, col: 0x400
  dimm 1 1024 Mb offset: 4, bank: 8, rank: 1, row: 0x4000, col: 0x400
Ch1 phy rd1, wr1 (0x063f4031): 2 ranks, UDIMMs
  dimm 0 1024 Mb offset: 0, bank: 8, rank: 1, row: 0x4000, col: 0x400
Ch2 phy rd3, wr3 (0x063f4031): 2 ranks, UDIMMs
  dimm 0 1024 Mb offset: 0, bank: 8, rank: 1, row: 0x4000, col: 0x400
```

   The driver will map it as:

```
csrow0: channel 0, dimm0
csrow1: channel 0, dimm1
csrow2: channel 1, dimm0
csrow3: channel 2, dimm0
```

   exports one DIMM per csrow.

   Each QPI is exported as a different memory controller.

2) The MC has the ability to inject errors to test drivers. The drivers implement this functionality via some error injection nodes:

   For injecting a memory error, there are some sysfs nodes, under `/sys/devices/system/edac/mc/mc?/`:

- **inject_addrmatch/*:**
  Controls the error injection mask register. It is possible to specify several characteristics of the address to match an error code:

  ```
  dimm = the affected dimm. Numbers are relative to a channel;
  rank = the memory rank;
  channel = the channel that will generate an error;
  bank = the affected bank;
  page = the page address;
  column (or col) = the address column.
  ```

  each of the above values can be set to "any" to match any valid value.

  At driver init, all values are set to any.

  For example, to generate an error at rank 1 of dimm 2, for any channel, any bank, any page, any column:

  ```
          echo 2 >/sys/devices/system/edac/mc/mc0/inject_addrmatch/
  ↪dimm
          echo 1 >/sys/devices/system/edac/mc/mc0/inject_addrmatch/
  ↪rank

  To return to the default behaviour of matching any, you can do::

          echo any >/sys/devices/system/edac/mc/mc0/inject_addrmatch/
  ↪dimm
          echo any >/sys/devices/system/edac/mc/mc0/inject_addrmatch/
  ↪rank
  ```

- **inject_eccmask:**
  specifies what bits will have troubles,

- **inject_section:**
  specifies what ECC cache section will get the error:

  ```
  3 for both
  2 for the highest
  1 for the lowest
  ```

- **inject_type:**
  specifies the type of error, being a combination of the following bits:

  ```
  bit 0 - repeat
  bit 1 - ecc
  bit 2 - parity
  ```

- **inject_enable:**
  starts the error generation when something different than 0 is written.

All inject vars can be read. root permission is needed for write.

Datasheet states that the error will only be generated after a write on an address that matches inject_addrmatch. It seems, however, that reading will also produce an error.

For example, the following code will generate an error for any write access at socket 0, on any DIMM/address on channel 2:

```
echo 2 >/sys/devices/system/edac/mc/mc0/inject_addrmatch/channel
echo 2 >/sys/devices/system/edac/mc/mc0/inject_type
echo 64 >/sys/devices/system/edac/mc/mc0/inject_eccmask
echo 3 >/sys/devices/system/edac/mc/mc0/inject_section
echo 1 >/sys/devices/system/edac/mc/mc0/inject_enable
dd if=/dev/mem of=/dev/null seek=16k bs=4k count=1 >& /dev/null
```

For socket 1, it is needed to replace "mc0" by "mc1" at the above commands.

The generated error message will look like:

```
EDAC MC0: UE row 0, channel-a= 0 channel-b= 0 labels "-": NON_FATAL (addr␣
↪= 0x0075b980, socket=0, Dimm=0, Channel=2, syndrome=0x00000040, count=1,␣
↪Err=8c0000400001009f:4000080482 (read error: read ECC error))
```

3) Corrected Error memory register counters

Those newer MCs have some registers to count memory errors. The driver uses those registers to report Corrected Errors on devices with Registered DIMMs.

However, those counters don't work with Unregistered DIMM. As the chipset offers some counters that also work with UDIMMs (but with a worse level of granularity than the default ones), the driver exposes those registers for UDIMM memories.

They can be read by looking at the contents of `all_channel_counts/`:

```
$ for i in /sys/devices/system/edac/mc/mc0/all_channel_counts/*; do echo
↪$i; cat $i; done
  /sys/devices/system/edac/mc/mc0/all_channel_counts/udimm0
  0
  /sys/devices/system/edac/mc/mc0/all_channel_counts/udimm1
  0
  /sys/devices/system/edac/mc/mc0/all_channel_counts/udimm2
  0
```

What happens here is that errors on different csrows, but at the same dimm number will increment the same counter. So, in this memory mapping:

```
csrow0: channel 0, dimm0
csrow1: channel 0, dimm1
csrow2: channel 1, dimm0
csrow3: channel 2, dimm0
```

The hardware will increment udimm0 for an error at the first dimm at either csrow0, csrow2 or csrow3;

The hardware will increment udimm1 for an error at the second dimm at either csrow0, csrow2 or csrow3;

The hardware will increment udimm2 for an error at the third dimm at either csrow0, csrow2 or csrow3;

4) Standard error counters

The standard error counters are generated when an mcelog error is received by the driver. Since, with UDIMM, this is counted by software, it is possible that some errors could be lost. With RDIMM's, they display the contents of the registers

71.2.20 Reference documents used on `amd64_edac`

`amd64_edac` module is based on the following documents (available from http://support.amd.com/en-us/search/tech-docs):

1. **Title**
   BIOS and Kernel Developer's Guide for AMD Athlon 64 and AMD Opteron Processors

   AMD publication #
   26094

   Revision
   3.26

   Link
   http://support.amd.com/TechDocs/26094.PDF

2. **Title**
   BIOS and Kernel Developer's Guide for AMD NPT Family 0Fh Processors

   AMD publication #
   32559

   Revision
   3.00

   Issue Date
   May 2006

   Link
   http://support.amd.com/TechDocs/32559.pdf

3. **Title**
   BIOS and Kernel Developer's Guide (BKDG) For AMD Family 10h Processors

   AMD publication #
   31116

   Revision
   3.00

   Issue Date
   September 07, 2007

   Link
   http://support.amd.com/TechDocs/31116.pdf

4. **Title**
   BIOS and Kernel Developer's Guide (BKDG) for AMD Family 15h Models 30h-3Fh Processors

   AMD publication #
   49125

Revision
3.06

Issue Date
2/12/2015 (latest release)

Link
http://support.amd.com/TechDocs/49125_15h_Models_30h-3Fh_BKDG.pdf

5. **Title**
BIOS and Kernel Developer's Guide (BKDG) for AMD Family 15h Models 60h-6Fh Processors

AMD publication #
50742

Revision
3.01

Issue Date
7/23/2015 (latest release)

Link
http://support.amd.com/TechDocs/50742_15h_Models_60h-6Fh_BKDG.pdf

6. **Title**
BIOS and Kernel Developer's Guide (BKDG) for AMD Family 16h Models 00h-0Fh Processors

AMD publication #
48751

Revision
3.03

Issue Date
2/23/2015 (latest release)

Link
http://support.amd.com/TechDocs/48751_16h_bkdg.pdf

Credits

- Written by Doug Thompson <dougthompson@xmission.com>
  - 7 Dec 2005
  - 17 Jul 2007 Updated
- © Mauro Carvalho Chehab
  - 05 Aug 2009 Nehalem interface
  - 26 Oct 2016 Converted to ReST and cleanups at the Nehalem section
- EDAC authors/maintainers:
  - Doug Thompson, Dave Jiang, Dave Peterson et al,
  - Mauro Carvalho Chehab

- Borislav Petkov
- original author: Thayne Harbaugh

CHAPTER

SEVENTYTWO

REAL TIME CLOCK (RTC) DRIVERS FOR LINUX

When Linux developers talk about a ”Real Time Clock”, they usually mean something that tracks wall clock time and is battery backed so that it works even with system power off. Such clocks will normally not track the local time zone or daylight savings time -- unless they dual boot with MS-Windows -- but will instead be set to Coordinated Universal Time (UTC, formerly ”Greenwich Mean Time”).

The newest non-PC hardware tends to just count seconds, like the time(2) system call reports, but RTCs also very commonly represent time using the Gregorian calendar and 24 hour time, as reported by gmtime(3).

Linux has two largely-compatible userspace RTC API families you may need to know about:

- /dev/rtc ... is the RTC provided by PC compatible systems, so it's not very portable to non-x86 systems.
- /dev/rtc0, /dev/rtc1 ... are part of a framework that's supported by a wide variety of RTC chips on all systems.

Programmers need to understand that the PC/AT functionality is not always available, and some systems can do much more. That is, the RTCs use the same API to make requests in both RTC frameworks (using different filenames of course), but the hardware may not offer the same functionality. For example, not every RTC is hooked up to an IRQ, so they can't all issue alarms; and where standard PC RTCs can only issue an alarm up to 24 hours in the future, other hardware may be able to schedule one any time in the upcoming century.

72.1 Old PC/AT-Compatible driver: /dev/rtc

All PCs (even Alpha machines) have a Real Time Clock built into them. Usually they are built into the chipset of the computer, but some may actually have a Motorola MC146818 (or clone) on the board. This is the clock that keeps the date and time while your computer is turned off.

ACPI has standardized that MC146818 functionality, and extended it in a few ways (enabling longer alarm periods, and wake-from-hibernate). That functionality is NOT exposed in the old driver.

However it can also be used to generate signals from a slow 2Hz to a relatively fast 8192Hz, in increments of powers of two. These signals are reported by interrupt number 8. (Oh! So *that* is what IRQ 8 is for...) It can also function as a 24hr alarm, raising IRQ 8 when the alarm goes off. The alarm can also be programmed to only check any subset of the three programmable values, meaning that it could be set to ring on the 30th second of the 30th minute of every hour,

for example. The clock can also be set to generate an interrupt upon every clock update, thus generating a 1Hz signal.

The interrupts are reported via /dev/rtc (major 10, minor 135, read only character device) in the form of an unsigned long. The low byte contains the type of interrupt (update-done, alarm-rang, or periodic) that was raised, and the remaining bytes contain the number of interrupts since the last read. Status information is reported through the pseudo-file /proc/driver/rtc if the /proc filesystem was enabled. The driver has built in locking so that only one process is allowed to have the /dev/rtc interface open at a time.

A user process can monitor these interrupts by doing a read(2) or a select(2) on /dev/rtc -- either will block/stop the user process until the next interrupt is received. This is useful for things like reasonably high frequency data acquisition where one doesn't want to burn up 100% CPU by polling gettimeofday etc. etc.

At high frequencies, or under high loads, the user process should check the number of interrupts received since the last read to determine if there has been any interrupt ”pileup” so to speak. Just for reference, a typical 486-33 running a tight read loop on /dev/rtc will start to suffer occasional interrupt pileup (i.e. > 1 IRQ event since last read) for frequencies above 1024Hz. So you really should check the high bytes of the value you read, especially at frequencies above that of the normal timer interrupt, which is 100Hz.

Programming and/or enabling interrupt frequencies greater than 64Hz is only allowed by root. This is perhaps a bit conservative, but we don't want an evil user generating lots of IRQs on a slow 386sx-16, where it might have a negative impact on performance. This 64Hz limit can be changed by writing a different value to /proc/sys/dev/rtc/max-user-freq. Note that the interrupt handler is only a few lines of code to minimize any possibility of this effect.

Also, if the kernel time is synchronized with an external source, the kernel will write the time back to the CMOS clock every 11 minutes. In the process of doing this, the kernel briefly turns off RTC periodic interrupts, so be aware of this if you are doing serious work. If you don't synchronize the kernel time with an external source (via ntp or whatever) then the kernel will keep its hands off the RTC, allowing you exclusive access to the device for your applications.

The alarm and/or interrupt frequency are programmed into the RTC via various ioctl(2) calls as listed in ./include/linux/rtc.h Rather than write 50 pages describing the ioctl() and so on, it is perhaps more useful to include a small test program that demonstrates how to use them, and demonstrates the features of the driver. This is probably a lot more useful to people interested in writing applications that will be using this driver. See the code at the end of this document.

(The original /dev/rtc driver was written by Paul Gortmaker.)

72.2 New portable ”RTC Class” drivers: /dev/rtcN

Because Linux supports many non-ACPI and non-PC platforms, some of which have more than one RTC style clock, it needed a more portable solution than expecting a single battery-backed MC146818 clone on every system. Accordingly, a new ”RTC Class” framework has been defined. It offers three different userspace interfaces:

- /dev/rtcN ... much the same as the older /dev/rtc interface
- /sys/class/rtc/rtcN ... sysfs attributes support readonly access to some RTC attributes.
- /proc/driver/rtc ... the system clock RTC may expose itself using a procfs interface. If there is no RTC for the system clock, rtc0 is used by default. More information is (currently)

shown here than through sysfs.

The RTC Class framework supports a wide variety of RTCs, ranging from those integrated into embeddable system-on-chip (SOC) processors to discrete chips using I2C, SPI, or some other bus to communicate with the host CPU. There's even support for PC-style RTCs … including the features exposed on newer PCs through ACPI.

The new framework also removes the ”one RTC per system” restriction. For example, maybe the low-power battery-backed RTC is a discrete I2C chip, but a high functionality RTC is integrated into the SOC. That system might read the system clock from the discrete RTC, but use the integrated one for all other tasks, because of its greater functionality.

Check out tools/testing/selftests/rtc/rtctest.c for an example usage of the ioctl interface.

CHAPTER

SEVENTYTHREE

LINUX SERIAL CONSOLE

To use a serial port as console you need to compile the support into your kernel - by default it is not compiled in. For PC style serial ports it's the config option next to menu option:

Character devices → Serial drivers → 8250/16550 and compatible serial support → Console on 8250/16550 and compatible serial port

You must compile serial support into the kernel and not as a module.

It is possible to specify multiple devices for console output. You can define a new kernel command line option to select which device(s) to use for console output.

The format of this option is:

```
console=device,options

device:         tty0 for the foreground virtual console
                ttyX for any other virtual console
                ttySx for a serial port
                lp0 for the first parallel port
                ttyUSB0 for the first USB serial device

options:        depend on the driver. For the serial port this
                defines the baudrate/parity/bits/flow control of
                the port, in the format BBBBPNF, where BBBB is the
                speed, P is parity (n/o/e), N is number of bits,
                and F is flow control ('r' for RTS). Default is
                9600n8. The maximum baudrate is 115200.
```

You can specify multiple console= options on the kernel command line.

The behavior is well defined when each device type is mentioned only once. In this case, the output will appear on all requested consoles. And the last device will be used when you open `/dev/console`. So, for example:

```
console=ttyS1,9600 console=tty0
```

defines that opening `/dev/console` will get you the current foreground virtual console, and kernel messages will appear on both the VGA console and the 2nd serial port (ttyS1 or COM2) at 9600 baud.

The behavior is more complicated when the same device type is defined more times. In this case, there are the following two rules:

1. The output will appear only on the first device of each defined type.
2. `/dev/console` will be associated with the first registered device. Where the registration order depends on how kernel initializes various subsystems.

   This rule is used also when the last console= parameter is not used for other reasons. For example, because of a typo or because the hardware is not available.

The result might be surprising. For example, the following two command lines have the same result:

```
console=ttyS1,9600 console=tty0 console=tty1
console=tty0 console=ttyS1,9600 console=tty1
```

The kernel messages are printed only on `tty0` and `ttyS1`. And `/dev/console` gets associated with `tty0`. It is because kernel tries to register graphical consoles before serial ones. It does it because of the default behavior when no console device is specified, see below.

Note that the last `console=tty1` parameter still makes a difference. The kernel command line is used also by systemd. It would use the last defined `tty1` as the login console.

If no console device is specified, the first device found capable of acting as a system console will be used. At this time, the system first looks for a VGA card and then for a serial port. So if you don't have a VGA card in your system the first serial port will automatically become the console.

You will need to create a new device to use `/dev/console`. The official `/dev/console` is now character device 5,1.

(You can also use a network device as a console. See `Documentation/networking/netconsole.rst` for information on that.)

Here's an example that will use `/dev/ttyS1` (COM2) as the console. Replace the sample values as needed.

1. Create `/dev/console` (real console) and `/dev/tty0` (master virtual console):

   ```
   cd /dev
   rm -f console tty0
   mknod -m 622 console c 5 1
   mknod -m 622 tty0 c 4 0
   ```

2. LILO can also take input from a serial device. This is a very useful option. To tell LILO to use the serial port: In lilo.conf (global section):

   ```
   serial  = 1,9600n8 (ttyS1, 9600 bd, no parity, 8 bits)
   ```

3. Adjust to kernel flags for the new kernel, again in lilo.conf (kernel section):

   ```
   append = "console=ttyS1,9600"
   ```

4. Make sure a getty runs on the serial port so that you can login to it once the system is done booting. This is done by adding a line like this to `/etc/inittab` (exact syntax depends on your getty):

   ```
   S1:23:respawn:/sbin/getty -L ttyS1 9600 vt100
   ```

5. Init and `/etc/ioctl.save`

   Sysvinit remembers its stty settings in a file in `/etc`, called `/etc/ioctl.save`. REMOVE THIS FILE before using the serial console for the first time, because otherwise init will probably set the baudrate to 38400 (baudrate of the virtual console).

6. `/dev/console` and X Programs that want to do something with the virtual console usually open `/dev/console`. If you have created the new `/dev/console` device, and your console is NOT the virtual console some programs will fail. Those are programs that want to access the VT interface, and use `/dev/console instead of /dev/tty0`. Some of those programs are:

   ```
   Xfree86, svgalib, gpm, SVGATextMode
   ```

   It should be fixed in modern versions of these programs though.

   Note that if you boot without a `console=` option (or with `console=/dev/tty0`), `/dev/console` is the same as `/dev/tty0`. In that case everything will still work.

7. Thanks

   Thanks to Geert Uytterhoeven <geert@linux-m68k.org> for porting the patches from 2.1.4x to 2.1.6x for taking care of the integration of these patches into m68k, ppc and alpha.

Miquel van Smoorenburg <miquels@cistron.nl>, 11-Jun-2000

CHAPTER

SEVENTYFOUR

VIDEO MODE SELECTION SUPPORT 2.13

Copyright
© 1995--1999 Martin Mares, <mj@ucw.cz>

74.1 Intro

This small document describes the ”Video Mode Selection” feature which allows the use of various special video modes supported by the video BIOS. Due to usage of the BIOS, the selection is limited to boot time (before the kernel decompression starts) and works only on 80X86 machines that are booted through BIOS firmware (as opposed to through UEFI, kexec, etc.).

Note: Short intro for the impatient: Just use vga=ask for the first time, enter `scan` on the video mode prompt, pick the mode you want to use, remember its mode ID (the four-digit hexadecimal number) and then set the vga parameter to this number (converted to decimal first).

The video mode to be used is selected by a kernel parameter which can be specified in the kernel Makefile (the SVGA_MODE=... line) or by the ”vga=...” option of LILO (or some other boot loader you use) or by the ”xrandr” utility (present in standard Linux utility packages). You can use the following values of this parameter:

```
NORMAL_VGA - Standard 80x25 mode available on all display adapters.

EXTENDED_VGA - Standard 8-pixel font mode: 80x43 on EGA, 80x50 on VGA.

ASK_VGA - Display a video mode menu upon startup (see below).

0..35 - Menu item number (when you have used the menu to view the list of
   modes available on your adapter, you can specify the menu item you want
   to use). 0..9 correspond to "0".."9", 10..35 to "a".."z". Warning: the
   mode list displayed may vary as the kernel version changes, because the
   modes are listed in a "first detected -- first displayed" manner. It's
   better to use absolute mode numbers instead.

0x.... - Hexadecimal video mode ID (also displayed on the menu, see below
   for exact meaning of the ID). Warning: LILO doesn't support
   hexadecimal numbers -- you have to convert it to decimal manually.
```

74.2 Menu

The ASK_VGA mode causes the kernel to offer a video mode menu upon bootup. It displays a ”Press <RETURN> to see video modes available, <SPACE> to continue or wait 30 secs” message. If you press <RETURN>, you enter the menu, if you press <SPACE> or wait 30 seconds, the kernel will boot up in the standard 80x25 mode.

The menu looks like:

```
Video adapter: <name-of-detected-video-adapter>
Mode:    COLSxROWS:
0  0F00  80x25
1  0F01  80x50
2  0F02  80x43
3  0F03  80x26
....
Enter mode number or ``scan``: <flashing-cursor-here>
```

<name-of-detected-video-adapter> tells what video adapter did Linux detect -- it's either a generic adapter name (MDA, CGA, HGC, EGA, VGA, VESA VGA [a VGA with VESA-compliant BIOS]) or a chipset name (e.g., Trident). Direct detection of chipsets is turned off by default as it's inherently unreliable due to absolutely insane PC design.

”0 0F00 80x25” means that the first menu item (the menu items are numbered from ”0” to ”9” and from ”a” to ”z”) is a 80x25 mode with ID=0x0f00 (see the next section for a description of mode IDs).

<flashing-cursor-here> encourages you to enter the item number or mode ID you wish to set and press <RETURN>. If the computer complains something about ”Unknown mode ID”, it is trying to tell you that it isn't possible to set such a mode. It's also possible to press only <RETURN> which leaves the current mode.

The mode list usually contains a few basic modes and some VESA modes. In case your chipset has been detected, some chipset-specific modes are shown as well (some of these might be missing or unusable on your machine as different BIOSes are often shipped with the same card and the mode numbers depend purely on the VGA BIOS).

The modes displayed on the menu are partially sorted: The list starts with the standard modes (80x25 and 80x50) followed by ”special” modes (80x28 and 80x43), local modes (if the local modes feature is enabled), VESA modes and finally SVGA modes for the auto-detected adapter.

If you are not happy with the mode list offered (e.g., if you think your card is able to do more), you can enter ”scan” instead of item number / mode ID. The program will try to ask the BIOS for all possible video mode numbers and test what happens then. The screen will be probably flashing wildly for some time and strange noises will be heard from inside the monitor and so on and then, really all consistent video modes supported by your BIOS will appear (plus maybe some `ghost modes`). If you are afraid this could damage your monitor, don't use this function.

After scanning, the mode ordering is a bit different: the auto-detected SVGA modes are not listed at all and the modes revealed by `scan` are shown before all VESA modes.

74.3 Mode IDs

Because of the complexity of all the video stuff, the video mode IDs used here are also a bit complex. A video mode ID is a 16-bit number usually expressed in a hexadecimal notation (starting with "0x"). You can set a mode by entering its mode directly if you know it even if it isn't shown on the menu.

The ID numbers can be divided to those regions:

```
0x0000 to 0x00ff - menu item references. 0x0000 is the first item. Don't use
     outside the menu as this can change from boot to boot (especially if you
     have used the ``scan`` feature).

0x0100 to 0x017f - standard BIOS modes. The ID is a BIOS video mode number
     (as presented to INT 10, function 00) increased by 0x0100.

0x0200 to 0x08ff - VESA BIOS modes. The ID is a VESA mode ID increased by
     0x0100. All VESA modes should be autodetected and shown on the menu.

0x0900 to 0x09ff - Video7 special modes. Set by calling INT 0x10, AX=0x6f05.
     (Usually 940=80x43, 941=132x25, 942=132x44, 943=80x60, 944=100x60,
     945=132x28 for the standard Video7 BIOS)

0x0f00 to 0x0fff - special modes (they are set by various tricks -- usually
     by modifying one of the standard modes). Currently available:
     0x0f00  standard 80x25, don't reset mode if already set (=FFFF)
     0x0f01  standard with 8-point font: 80x43 on EGA, 80x50 on VGA
     0x0f02  VGA 80x43 (VGA switched to 350 scanlines with a 8-point font)
     0x0f03  VGA 80x28 (standard VGA scans, but 14-point font)
     0x0f04  leave current video mode
     0x0f05  VGA 80x30 (480 scans, 16-point font)
     0x0f06  VGA 80x34 (480 scans, 14-point font)
     0x0f07  VGA 80x60 (480 scans, 8-point font)
     0x0f08  Graphics hack (see the VIDEO_GFX_HACK paragraph below)

0x1000 to 0x7fff - modes specified by resolution. The code has a "0xRRCC"
     form where RR is a number of rows and CC is a number of columns.
     E.g., 0x1950 corresponds to a 80x25 mode, 0x2b84 to 132x43 etc.
     This is the only fully portable way to refer to a non-standard mode,
     but it relies on the mode being found and displayed on the menu
     (remember that mode scanning is not done automatically).

0xff00 to 0xffff - aliases for backward compatibility:
     0xffff  equivalent to 0x0f00 (standard 80x25)
     0xfffe  equivalent to 0x0f01 (EGA 80x43 or VGA 80x50)
```

If you add 0x8000 to the mode ID, the program will try to recalculate vertical display timing according to mode parameters, which can be used to eliminate some annoying bugs of certain VGA BIOSes (usually those used for cards with S3 chipsets and old Cirrus Logic BIOSes) -- mainly extra lines at the end of the display.

74.4 Options

Build options for arch/x86/boot/* are selected by the kernel kconfig utility and the kernel .config file.

VIDEO_GFX_HACK - includes special hack for setting of graphics modes to be used later by special drivers. Allows to set _any_ BIOS mode including graphic ones and forcing specific text screen resolution instead of peeking it from BIOS variables. Don't use unless you think you know what you're doing. To activate this setup, use mode number 0x0f08 (see the Mode IDs section above).

74.5 Still doesn't work?

When the mode detection doesn't work (e.g., the mode list is incorrect or the machine hangs instead of displaying the menu), try to switch off some of the configuration options listed under ”Options”. If it fails, you can still use your kernel with the video mode set directly via the kernel parameter.

In either case, please send me a bug report containing what _exactly_ happens and how do the configuration switches affect the behaviour of the bug.

If you start Linux from M$-DOS, you might also use some DOS tools for video mode setting. In this case, you must specify the 0x0f04 mode (”leave current settings”) to Linux, because if you don't and you use any non-standard mode, Linux will switch to 80x25 automatically.

If you set some extended mode and there's one or more extra lines on the bottom of the display containing already scrolled-out text, your VGA BIOS contains the most common video BIOS bug called ”incorrect vertical display end setting”. Adding 0x8000 to the mode ID might fix the problem. Unfortunately, this must be done manually -- no autodetection mechanisms are available.

74.6 History

1.0 (??-Nov-95)	First version supporting all adapters supported by the old setup.S + Cirrus Logic 54XX. Present in some 1.3.4? kernels and then removed due to instability on some machines.
2.0 (28-Jan-96)	Rewritten from scratch. Cirrus Logic 64XX support added, almost everything is configurable, the VESA support should be much more stable, explicit mode numbering allowed, ”scan” implemented etc.
2.1 (30-Jan-96)	VESA modes moved to 0x200-0x3ff. Mode selection by resolution supported. Few bugs fixed. VESA modes are listed prior to modes supplied by SVGA autodetection as they are more reliable. CLGD autodetect works better. Doesn't depend on 80x25 being active when started. Scanning fixed. 80x43 (any VGA) added. Code cleaned up.
2.2 (01-Feb-96)	EGA 80x43 fixed. VESA extended to 0x200-0x4ff (non-standard 02XX VESA modes work now). Display end bug workaround supported. Special modes renumbered to allow adding of the ”recalculate” flag, 0xffff and 0xfffe became aliases instead of real IDs. Screen contents retained during mode changes.
2.3 (15-Mar-96)	Changed to work with 1.3.74 kernel.
2.4 (18-Mar-96)	Added patches by Hans Lermen fixing a memory overwrite problem with some boot loaders. Memory management rewritten to reflect these changes. Unfortunately, screen contents retaining works only with some loaders now. Added a Tseng 132x60 mode.
2.5 (19-Mar-96)	Fixed a VESA mode scanning bug introduced in 2.4.
2.6 (25-Mar-96)	Some VESA BIOS errors not reported -- it fixes error reports on several cards with broken VESA code (e.g., ATI VGA).
2.7 (09-Apr-96)	• Accepted all VESA modes in range 0x100 to 0x7ff, because some cards use very strange mode numbers. • Added Realtek VGA modes (thanks to Gonzalo Tornaria). • Hardware testing order slightly changed, tests based on ROM contents done as first. • Added support for special Video7 mode switching functions (thanks to Tom Vander Aa). • Added 1400x1050 modes (especially useful for notebooks, original version written by hhanemaa@cs.ruu.nl, patched by Jeff Chua, rewritten by me).

CHAPTER

SEVENTYFIVE

SYSCALL USER DISPATCH

75.1 Background

Compatibility layers like Wine need a way to efficiently emulate system calls of only a part of their process - the part that has the incompatible code - while being able to execute native syscalls without a high performance penalty on the native part of the process. Seccomp falls short on this task, since it has limited support to efficiently filter syscalls based on memory regions, and it doesn't support removing filters. Therefore a new mechanism is necessary.

Syscall User Dispatch brings the filtering of the syscall dispatcher address back to userspace. The application is in control of a flip switch, indicating the current personality of the process. A multiple-personality application can then flip the switch without invoking the kernel, when crossing the compatibility layer API boundaries, to enable/disable the syscall redirection and execute syscalls directly (disabled) or send them to be emulated in userspace through a SIGSYS.

The goal of this design is to provide very quick compatibility layer boundary crosses, which is achieved by not executing a syscall to change personality every time the compatibility layer executes. Instead, a userspace memory region exposed to the kernel indicates the current personality, and the application simply modifies that variable to configure the mechanism.

There is a relatively high cost associated with handling signals on most architectures, like x86, but at least for Wine, syscalls issued by native Windows code are currently not known to be a performance problem, since they are quite rare, at least for modern gaming applications.

Since this mechanism is designed to capture syscalls issued by non-native applications, it must function on syscalls whose invocation ABI is completely unexpected to Linux. Syscall User Dispatch, therefore doesn't rely on any of the syscall ABI to make the filtering. It uses only the syscall dispatcher address and the userspace key.

As the ABI of these intercepted syscalls is unknown to Linux, these syscalls are not instrumentable via ptrace or the syscall tracepoints.

75.2 Interface

A thread can setup this mechanism on supported kernels by executing the following prctl:

prctl(PR_SET_SYSCALL_USER_DISPATCH, <op>, <offset>, <length>, [selector])

<op> is either PR_SYS_DISPATCH_ON or PR_SYS_DISPATCH_OFF, to enable and disable the mechanism globally for that thread. When PR_SYS_DISPATCH_OFF is used, the other fields must be zero.

[<offset>, <offset>+<length>) delimit a memory region interval from which syscalls are always executed directly, regardless of the userspace selector. This provides a fast path for the C library, which includes the most common syscall dispatchers in the native code applications, and also provides a way for the signal handler to return without triggering a nested SIGSYS on (rt_)sigreturn. Users of this interface should make sure that at least the signal trampoline code is included in this region. In addition, for syscalls that implement the trampoline code on the vDSO, that trampoline is never intercepted.

[selector] is a pointer to a char-sized region in the process memory region, that provides a quick way to enable disable syscall redirection thread-wide, without the need to invoke the kernel directly. selector can be set to SYSCALL_DISPATCH_FILTER_ALLOW or SYSCALL_DISPATCH_FILTER_BLOCK. Any other value should terminate the program with a SIGSYS.

Additionally, a tasks syscall user dispatch configuration can be peeked and poked via the PTRACE_(GET|SET)_SYSCALL_USER_DISPATCH_CONFIG ptrace requests. This is useful for checkpoint/restart software.

75.3 Security Notes

Syscall User Dispatch provides functionality for compatibility layers to quickly capture system calls issued by a non-native part of the application, while not impacting the Linux native regions of the process. It is not a mechanism for sandboxing system calls, and it should not be seen as a security mechanism, since it is trivial for a malicious application to subvert the mechanism by jumping to an allowed dispatcher region prior to executing the syscall, or to discover the address and modify the selector value. If the use case requires any kind of security sandboxing, Seccomp should be used instead.

Any fork or exec of the existing process resets the mechanism to PR_SYS_DISPATCH_OFF.

CHAPTER

SEVENTYSIX

LINUX MAGIC SYSTEM REQUEST KEY HACKS

Documentation for sysrq.c

76.1 What is the magic SysRq key?

It is a 'magical' key combo you can hit which the kernel will respond to regardless of whatever else it is doing, unless it is completely locked up.

76.2 How do I enable the magic SysRq key?

You need to say "yes" to 'Magic SysRq key (CONFIG_MAGIC_SYSRQ)' when configuring the kernel. When running a kernel with SysRq compiled in, /proc/sys/kernel/sysrq controls the functions allowed to be invoked via the SysRq key. The default value in this file is set by the CONFIG_MAGIC_SYSRQ_DEFAULT_ENABLE config symbol, which itself defaults to 1. Here is the list of possible values in /proc/sys/kernel/sysrq:

- 0 - disable sysrq completely
- 1 - enable all functions of sysrq
- >1 - bitmask of allowed sysrq functions (see below for detailed function description):

```
  2 =   0x2 - enable control of console logging level
  4 =   0x4 - enable control of keyboard (SAK, unraw)
  8 =   0x8 - enable debugging dumps of processes etc.
 16 =  0x10 - enable sync command
 32 =  0x20 - enable remount read-only
 64 =  0x40 - enable signalling of processes (term, kill, oom-kill)
128 =  0x80 - allow reboot/poweroff
256 = 0x100 - allow nicing of all RT tasks
```

You can set the value in the file by the following command:

```
echo "number" >/proc/sys/kernel/sysrq
```

The number may be written here either as decimal or as hexadecimal with the 0x prefix. CONFIG_MAGIC_SYSRQ_DEFAULT_ENABLE must always be written in hexadecimal.

Note that the value of `/proc/sys/kernel/sysrq` influences only the invocation via a keyboard. Invocation of any operation via `/proc/sysrq-trigger` is always allowed (by a user with admin privileges).

76.3 How do I use the magic SysRq key?

On x86
You press the key combo `ALT-SysRq-<command key>`.

> **Note:** Some keyboards may not have a key labeled 'SysRq'. The 'SysRq' key is also known as the 'Print Screen' key. Also some keyboards cannot handle so many keys being pressed at the same time, so you might have better luck with press `Alt`, press `SysRq`, release `SysRq`, press `<command key>`, release everything.

On SPARC
You press `ALT-STOP-<command key>`, I believe.

On the serial console (PC style standard serial ports only)
You send a `BREAK`, then within 5 seconds a command key. Sending `BREAK` twice is interpreted as a normal BREAK.

On PowerPC
Press `ALT - Print Screen` (or `F13`) - `<command key>`. `Print Screen` (or `F13`) - `<command key>` may suffice.

On other
If you know of the key combos for other architectures, please submit a patch to be included in this section.

On all
Write a single character to /proc/sysrq-trigger. Only the first character is processed, the rest of the string is ignored. However, it is not recommended to write any extra characters as the behavior is undefined and might change in the future versions. E.g.:

```
echo t > /proc/sysrq-trigger
```

Alternatively, write multiple characters prepended by underscore. This way, all characters will be processed. E.g.:

```
echo _reisub > /proc/sysrq-trigger
```

The `<command key>` is case sensitive.

76.4 What are the 'command' keys?

Command	Function
b	Will immediately reboot the system without syncing or unmounting your disks.
c	Will perform a system crash and a crashdump will be taken if configured.
d	Shows all locks that are held.
e	Send a SIGTERM to all processes, except for init.
f	Will call the oom killer to kill a memory hog process, but do not panic if nothing can be killed.
g	Used by kgdb (kernel debugger)
h	Will display help (actually any other key than those listed here will display help. but h is easy to remember :-)
i	Send a SIGKILL to all processes, except for init.
j	Forcibly "Just thaw it" - filesystems frozen by the FIFREEZE ioctl.
k	Secure Access Key (SAK) Kills all programs on the current virtual console. NOTE: See important comments below in SAK section.
l	Shows a stack backtrace for all active CPUs.
m	Will dump current memory info to your console.
n	Used to make RT tasks nice-able
o	Will shut your system off (if configured and supported).
p	Will dump the current registers and flags to your console.
q	Will dump per CPU lists of all armed hrtimers (but NOT regular timer_list timers) and detailed information about all clockevent devices.
r	Turns off keyboard raw mode and sets it to XLATE.
s	Will attempt to sync all mounted filesystems.
t	Will dump a list of current tasks and their information to your console.
u	Will attempt to remount all mounted filesystems read-only.
v	Forcefully restores framebuffer console
v	Causes ETM buffer dump [ARM-specific]
w	Dumps tasks that are in uninterruptible (blocked) state.
x	Used by xmon interface on ppc/powerpc platforms. Show global PMU Registers on sparc64. Dump all TLB entries on MIPS.
y	Show global CPU Registers [SPARC-64 specific]
z	Dump the ftrace buffer
0-9	Sets the console log level, controlling which kernel messages will be printed to your console. (0, for example would make it so that only emergency messages like PANICs or OOPSes would make it to your console.)

76.5 Okay, so what can I use them for?

Well, unraw(r) is very handy when your X server or a svgalib program crashes.

sak(k) (Secure Access Key) is useful when you want to be sure there is no trojan program running at console which could grab your password when you would try to login. It will kill all programs on given console, thus letting you make sure that the login prompt you see is actually the one from init, not some trojan program.

Important: In its true form it is not a true SAK like the one in a c2 compliant system, and it

should not be mistaken as such.

It seems others find it useful as (System Attention Key) which is useful when you want to exit a program that will not let you switch consoles. (For example, X or a svgalib program.)

`reboot(b)` is good when you're unable to shut down, it is an equivalent of pressing the ”reset” button.

`crash(c)` can be used to manually trigger a crashdump when the system is hung. Note that this just triggers a crash if there is no dump mechanism available.

`sync(s)` is handy before yanking removable medium or after using a rescue shell that provides no graceful shutdown -- it will ensure your data is safely written to the disk. Note that the sync hasn't taken place until you see the ”OK” and ”Done” appear on the screen.

`umount(u)` can be used to mark filesystems as properly unmounted. From the running system's point of view, they will be remounted read-only. The remount isn't complete until you see the ”OK” and ”Done” message appear on the screen.

The loglevels `0`-`9` are useful when your console is being flooded with kernel messages you do not want to see. Selecting `0` will prevent all but the most urgent kernel messages from reaching your console. (They will still be logged if syslogd/klogd are alive, though.)

`term(e)` and `kill(i)` are useful if you have some sort of runaway process you are unable to kill any other way, especially if it's spawning other processes.

”just thaw `it(j)`” is useful if your system becomes unresponsive due to a frozen (probably root) filesystem via the FIFREEZE ioctl.

76.6 Sometimes SysRq seems to get 'stuck' after using it, what can I do?

When this happens, try tapping shift, alt and control on both sides of the keyboard, and hitting an invalid sysrq sequence again. (i.e., something like `alt-sysrq-z`).

Switching to another virtual console (`ALT+Fn`) and then back again should also help.

76.7 I hit SysRq, but nothing seems to happen, what's wrong?

There are some keyboards that produce a different keycode for SysRq than the pre-defined value of 99 (see `KEY_SYSRQ` in `include/uapi/linux/input-event-codes.h`), or which don't have a SysRq key at all. In these cases, run `showkey -s` to find an appropriate scancode sequence, and use `setkeycodes <sequence> 99` to map this sequence to the usual SysRq code (e.g., `setkeycodes e05b 99`). It's probably best to put this command in a boot script. Oh, and by the way, you exit `showkey` by not typing anything for ten seconds.

76.8 I want to add SysRQ key events to a module, how does it work?

In order to register a basic function with the table, you must first include the header `include/linux/sysrq.h`, this will define everything else you need. Next, you must create a `sysrq_key_op` struct, and populate it with A) the key handler function you will use, B) a help_msg string, that will print when SysRQ prints help, and C) an action_msg string, that will print right before your handler is called. Your handler must conform to the prototype in 'sysrq.h'.

After the `sysrq_key_op` is created, you can call the kernel function `register_sysrq_key(int key, const struct sysrq_key_op *op_p);` this will register the operation pointed to by `op_p` at table key 'key', if that slot in the table is blank. At module unload time, you must call the function `unregister_sysrq_key(int key, const struct sysrq_key_op *op_p)`, which will remove the key op pointed to by 'op_p' from the key 'key', if and only if it is currently registered in that slot. This is in case the slot has been overwritten since you registered it.

The Magic SysRQ system works by registering key operations against a key op lookup table, which is defined in 'drivers/tty/sysrq.c'. This key table has a number of operations registered into it at compile time, but is mutable, and 2 functions are exported for interface to it:

```
register_sysrq_key and unregister_sysrq_key.
```

Of course, never ever leave an invalid pointer in the table. I.e., when your module that called register_sysrq_key() exits, it must call unregister_sysrq_key() to clean up the sysrq key table entry that it used. Null pointers in the table are always safe. :)

If for some reason you feel the need to call the handle_sysrq function from within a function called by handle_sysrq, you must be aware that you are in a lock (you are also in an interrupt handler, which means don't sleep!), so you must call `__handle_sysrq_nolock` instead.

76.9 When I hit a SysRq key combination only the header appears on the console?

Sysrq output is subject to the same console loglevel control as all other console output. This means that if the kernel was booted 'quiet' as is common on distro kernels the output may not appear on the actual console, even though it will appear in the dmesg buffer, and be accessible via the dmesg command and to the consumers of `/proc/kmsg`. As a specific exception the header line from the sysrq command is passed to all console consumers as if the current loglevel was maximum. If only the header is emitted it is almost certain that the kernel loglevel is too low. Should you require the output on the console channel then you will need to temporarily up the console loglevel using `alt-sysrq-8` or:

```
echo 8 > /proc/sysrq-trigger
```

Remember to return the loglevel to normal after triggering the sysrq command you are interested in.

76.10 I have more questions, who can I ask?

Just ask them on the linux-kernel mailing list:
linux-kernel@vger.kernel.org

76.11 Credits

- Written by Mydraal <vulpyne@vulpyne.net>
- Updated by Adam Sulmicki <adam@cfar.umd.edu>
- Updated by Jeremy M. Dolan <jmd@turbogeek.org> 2001/01/28 10:15:59
- Added to by Crutcher Dunnavant <crutcher+kernel@datastacks.com>

CHAPTER

SEVENTYSEVEN

THERMAL SUBSYSTEM

77.1 Intel Powerclamp Driver

By:

- Arjan van de Ven <arjan@linux.intel.com>
- Jacob Pan <jacob.jun.pan@linux.intel.com>

77.1.1 INTRODUCTION

Consider the situation where a system's power consumption must be reduced at runtime, due to power budget, thermal constraint, or noise level, and where active cooling is not preferred. Software managed passive power reduction must be performed to prevent the hardware actions that are designed for catastrophic scenarios.

Currently, P-states, T-states (clock modulation), and CPU offlining are used for CPU throttling.

On Intel CPUs, C-states provide effective power reduction, but so far they're only used opportunistically, based on workload. With the development of intel_powerclamp driver, the method of synchronizing idle injection across all online CPU threads was introduced. The goal is to achieve forced and controllable C-state residency.

Test/Analysis has been made in the areas of power, performance, scalability, and user experience. In many cases, clear advantage is shown over taking the CPU offline or modulating the CPU clock.

77.1.2 THEORY OF OPERATION

Idle Injection

On modern Intel processors (Nehalem or later), package level C-state residency is available in MSRs, thus also available to the kernel.

These MSRs are:

```
#define MSR_PKG_C2_RESIDENCY      0x60D
#define MSR_PKG_C3_RESIDENCY      0x3F8
#define MSR_PKG_C6_RESIDENCY      0x3F9
#define MSR_PKG_C7_RESIDENCY      0x3FA
```

If the kernel can also inject idle time to the system, then a closed-loop control system can be established that manages package level C-state. The intel_powerclamp driver is conceived as such a control system, where the target set point is a user-selected idle ratio (based on power reduction), and the error is the difference between the actual package level C-state residency ratio and the target idle ratio.

Injection is controlled by high priority kernel threads, spawned for each online CPU.

These kernel threads, with SCHED_FIFO class, are created to perform clamping actions of controlled duty ratio and duration. Each per-CPU thread synchronizes its idle time and duration, based on the rounding of jiffies, so accumulated errors can be prevented to avoid a jittery effect. Threads are also bound to the CPU such that they cannot be migrated, unless the CPU is taken offline. In this case, threads belong to the offlined CPUs will be terminated immediately.

Running as SCHED_FIFO and relatively high priority, also allows such scheme to work for both preemptible and non-preemptible kernels. Alignment of idle time around jiffies ensures scalability for HZ values. This effect can be better visualized using a Perf timechart. The following diagram shows the behavior of kernel thread kidle_inject/cpu. During idle injection, it runs monitor/mwait idle for a given ”duration”, then relinquishes the CPU to other tasks, until the next time interval.

The NOHZ schedule tick is disabled during idle time, but interrupts are not masked. Tests show that the extra wakeups from scheduler tick have a dramatic impact on the effectiveness of the powerclamp driver on large scale systems (Westmere system with 80 processors).

```
CPU0
                    ____________          ____________
kidle_inject/0   |   sleep    |  mwait |  sleep     |
        _________|            |________|            |_______
                              duration
CPU1
                    ____________          ____________
kidle_inject/1   |   sleep    |  mwait |  sleep     |
        _________|            |________|            |_______
                             ^
                             |
                             |
                             roundup(jiffies, interval)
```

Only one CPU is allowed to collect statistics and update global control parameters. This CPU is referred to as the controlling CPU in this document. The controlling CPU is elected at runtime, with a policy that favors BSP, taking into account the possibility of a CPU hot-plug.

In terms of dynamics of the idle control system, package level idle time is considered largely as a non-causal system where its behavior cannot be based on the past or current input. Therefore, the intel_powerclamp driver attempts to enforce the desired idle time instantly as given input (target idle ratio). After injection, powerclamp monitors the actual idle for a given time window and adjust the next injection accordingly to avoid over/under correction.

When used in a causal control system, such as a temperature control, it is up to the user of this driver to implement algorithms where past samples and outputs are included in the feedback. For example, a PID-based thermal controller can use the powerclamp driver to maintain a desired target temperature, based on integral and derivative gains of the past samples.

Calibration

During scalability testing, it is observed that synchronized actions among CPUs become challenging as the number of cores grows. This is also true for the ability of a system to enter package level C-states.

To make sure the intel_powerclamp driver scales well, online calibration is implemented. The goals for doing such a calibration are:

a) determine the effective range of idle injection ratio

b) determine the amount of compensation needed at each target ratio

Compensation to each target ratio consists of two parts:

a) steady state error compensation

   This is to offset the error occurring when the system can enter idle without extra wakeups (such as external interrupts).

b) dynamic error compensation

   When an excessive amount of wakeups occurs during idle, an additional idle ratio can be added to quiet interrupts, by slowing down CPU activities.

A debugfs file is provided for the user to examine compensation progress and results, such as on a Westmere system:

```
[jacob@nex01 ~]$ cat
/sys/kernel/debug/intel_powerclamp/powerclamp_calib
controlling cpu: 0
pct confidence steady dynamic (compensation)
0       0       0       0
1       1       0       0
2       1       1       0
3       3       1       0
4       3       1       0
5       3       1       0
6       3       1       0
7       3       1       0
8       3       1       0
...
30      3       2       0
31      3       2       0
32      3       1       0
33      3       2       0
34      3       1       0
35      3       2       0
36      3       1       0
37      3       2       0
38      3       1       0
39      3       2       0
40      3       3       0
41      3       1       0
42      3       2       0
43      3       1       0
```

```
44      3       1       0
45      3       2       0
46      3       3       0
47      3       0       0
48      3       2       0
49      3       3       0
```

Calibration occurs during runtime. No offline method is available. Steady state compensation is used only when confidence levels of all adjacent ratios have reached satisfactory level. A confidence level is accumulated based on clean data collected at runtime. Data collected during a period without extra interrupts is considered clean.

To compensate for excessive amounts of wakeup during idle, additional idle time is injected when such a condition is detected. Currently, we have a simple algorithm to double the injection ratio. A possible enhancement might be to throttle the offending IRQ, such as delaying EOI for level triggered interrupts. But it is a challenge to be non-intrusive to the scheduler or the IRQ core code.

CPU Online/Offline

Per-CPU kernel threads are started/stopped upon receiving notifications of CPU hotplug activities. The intel_powerclamp driver keeps track of clamping kernel threads, even after they are migrated to other CPUs, after a CPU offline event.

77.1.3 Performance Analysis

This section describes the general performance data collected on multiple systems, including Westmere (80P) and Ivy Bridge (4P, 8P).

Effectiveness and Limitations

The maximum range that idle injection is allowed is capped at 50 percent. As mentioned earlier, since interrupts are allowed during forced idle time, excessive interrupts could result in less effectiveness. The extreme case would be doing a ping -f to generated flooded network interrupts without much CPU acknowledgement. In this case, little can be done from the idle injection threads. In most normal cases, such as scp a large file, applications can be throttled by the powerclamp driver, since slowing down the CPU also slows down network protocol processing, which in turn reduces interrupts.

When control parameters change at runtime by the controlling CPU, it may take an additional period for the rest of the CPUs to catch up with the changes. During this time, idle injection is out of sync, thus not able to enter package C- states at the expected ratio. But this effect is minor, in that in most cases change to the target ratio is updated much less frequently than the idle injection frequency.

Scalability

Tests also show a minor, but measurable, difference between the 4P/8P Ivy Bridge system and the 80P Westmere server under 50% idle ratio. More compensation is needed on Westmere for the same amount of target idle ratio. The compensation also increases as the idle ratio gets larger. The above reason constitutes the need for the calibration code.

On the IVB 8P system, compared to an offline CPU, powerclamp can achieve up to 40% better performance per watt. (measured by a spin counter summed over per CPU counting threads spawned for all running CPUs).

77.1.4 Usage and Interfaces

The powerclamp driver is registered to the generic thermal layer as a cooling device. Currently, it's not bound to any thermal zones:

```
jacob@chromoly:/sys/class/thermal/cooling_device14$ grep . *
cur_state:0
max_state:50
type:intel_powerclamp
```

cur_state allows user to set the desired idle percentage. Writing 0 to cur_state will stop idle injection. Writing a value between 1 and max_state will start the idle injection. Reading cur_state returns the actual and current idle percentage. This may not be the same value set by the user in that current idle percentage depends on workload and includes natural idle. When idle injection is disabled, reading cur_state returns value -1 instead of 0 which is to avoid confusing 100% busy state with the disabled state.

Example usage:

- To inject 25% idle time:

  ```
  $ sudo sh -c "echo 25 > /sys/class/thermal/cooling_device80/cur_state
  ```

If the system is not busy and has more than 25% idle time already, then the powerclamp driver will not start idle injection. Using Top will not show idle injection kernel threads.

If the system is busy (spin test below) and has less than 25% natural idle time, powerclamp kernel threads will do idle injection. Forced idle time is accounted as normal idle in that common code path is taken as the idle task.

In this example, 24.1% idle is shown. This helps the system admin or user determine the cause of slowdown, when a powerclamp driver is in action:

```
Tasks: 197 total,   1 running, 196 sleeping,   0 stopped,   0 zombie
Cpu(s): 71.2%us,  4.7%sy,  0.0%ni, 24.1%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:   3943228k total,  1689632k used,  2253596k free,    74960k buffers
Swap:  4087804k total,        0k used,  4087804k free,   945336k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 3352 jacob     20   0  262m  644  428 S  286  0.0   0:17.16 spin
 3341 root     -51   0     0    0    0 D   25  0.0   0:01.62 kidle_inject/0
 3344 root     -51   0     0    0    0 D   25  0.0   0:01.60 kidle_inject/3
 3342 root     -51   0     0    0    0 D   25  0.0   0:01.61 kidle_inject/1
```

```
 3343 root     -51   0     0    0    0 D   25  0.0   0:01.60 kidle_inject/2
 2935 jacob     20   0  696m 125m  35m S    5  3.3   0:31.11 firefox
 1546 root      20   0  158m  20m 6640 S    3  0.5   0:26.97 Xorg
 2100 jacob     20   0 1223m  88m  30m S    3  2.3   0:23.68 compiz
```

Tests have shown that by using the powerclamp driver as a cooling device, a PID based userspace thermal controller can manage to control CPU temperature effectively, when no other thermal influence is added. For example, a UltraBook user can compile the kernel under certain temperature (below most active trip points).

77.1.5 Module Parameters

`cpumask` (RW)

A bit mask of CPUs to inject idle. The format of the bitmask is same as used in other subsystems like in /proc/irq/*/smp_affinity. The mask is comma separated 32 bit groups. Each CPU is one bit. For example for a 256 CPU system the full mask is: ffffffff,ffffffff,ffffffff,ffffffff,ffffffff,ffffffff,ffffffff,ffffffff

The rightmost mask is for CPU 0-32.

`max_idle` (RW)

Maximum injected idle time to the total CPU time ratio in percent range from 1 to 100. Even if the cooling device max_state is always 100 (100%), this parameter allows to add a max idle percent limit. The default is 50, to match the current implementation of powerclamp driver. Also doesn't allow value more than 75, if the cpumask includes every CPU present in the system.

CHAPTER

SEVENTYEIGHT

USB4 AND THUNDERBOLT

USB4 is the public specification based on Thunderbolt 3 protocol with some differences at the register level among other things. Connection manager is an entity running on the host router (host controller) responsible for enumerating routers and establishing tunnels. A connection manager can be implemented either in firmware or software. Typically PCs come with a firmware connection manager for Thunderbolt 3 and early USB4 capable systems. Apple systems on the other hand use software connection manager and the later USB4 compliant devices follow the suit.

The Linux Thunderbolt driver supports both and can detect at runtime which connection manager implementation is to be used. To be on the safe side the software connection manager in Linux also advertises security level `user` which means PCIe tunneling is disabled by default. The documentation below applies to both implementations with the exception that the software connection manager only supports `user` security level and is expected to be accompanied with an IOMMU based DMA protection.

78.1 Security levels and how to use them

The interface presented here is not meant for end users. Instead there should be a userspace tool that handles all the low-level details, keeps a database of the authorized devices and prompts users for new connections.

More details about the sysfs interface for Thunderbolt devices can be found in `Documentation/ABI/testing/sysfs-bus-thunderbolt`.

Those users who just want to connect any device without any sort of manual work can add following line to `/etc/udev/rules.d/99-local.rules`:

```
ACTION=="add", SUBSYSTEM=="thunderbolt", ATTR{authorized}=="0", ATTR
↪{authorized}="1"
```

This will authorize all devices automatically when they appear. However, keep in mind that this bypasses the security levels and makes the system vulnerable to DMA attacks.

Starting with Intel Falcon Ridge Thunderbolt controller there are 4 security levels available. Intel Titan Ridge added one more security level (usbonly). The reason for these is the fact that the connected devices can be DMA masters and thus read contents of the host memory without CPU and OS knowing about it. There are ways to prevent this by setting up an IOMMU but it is not always available for various reasons.

Some USB4 systems have a BIOS setting to disable PCIe tunneling. This is treated as another security level (nopcie).

The security levels are as follows:

none
All devices are automatically connected by the firmware. No user approval is needed. In BIOS settings this is typically called *Legacy mode*.

user
User is asked whether the device is allowed to be connected. Based on the device identification information available through `/sys/bus/thunderbolt/devices`, the user then can make the decision. In BIOS settings this is typically called *Unique ID*.

secure
User is asked whether the device is allowed to be connected. In addition to UUID the device (if it supports secure connect) is sent a challenge that should match the expected one based on a random key written to the `key` sysfs attribute. In BIOS settings this is typically called *One time saved key*.

dponly
The firmware automatically creates tunnels for Display Port and USB. No PCIe tunneling is done. In BIOS settings this is typically called *Display Port Only*.

usbonly
The firmware automatically creates tunnels for the USB controller and Display Port in a dock. All PCIe links downstream of the dock are removed.

nopcie
PCIe tunneling is disabled/forbidden from the BIOS. Available in some USB4 systems.

The current security level can be read from `/sys/bus/thunderbolt/devices/domainX/security` where `domainX` is the Thunderbolt domain the host controller manages. There is typically one domain per Thunderbolt host controller.

If the security level reads as `user` or `secure` the connected device must be authorized by the user before PCIe tunnels are created (e.g the PCIe device appears).

Each Thunderbolt device plugged in will appear in sysfs under `/sys/bus/thunderbolt/devices`. The device directory carries information that can be used to identify the particular device, including its name and UUID.

78.2 Authorizing devices when security level is `user` or `secure`

When a device is plugged in it will appear in sysfs as follows:

```
/sys/bus/thunderbolt/devices/0-1/authorized   - 0
/sys/bus/thunderbolt/devices/0-1/device       - 0x8004
/sys/bus/thunderbolt/devices/0-1/device_name  - Thunderbolt to FireWire Adapter
/sys/bus/thunderbolt/devices/0-1/vendor       - 0x1
/sys/bus/thunderbolt/devices/0-1/vendor_name  - Apple, Inc.
/sys/bus/thunderbolt/devices/0-1/unique_id    - e0376f00-0300-0100-ffff-
↪ffffffffffff
```

The `authorized` attribute reads 0 which means no PCIe tunnels are created yet. The user can authorize the device by simply entering:

```
# echo 1 > /sys/bus/thunderbolt/devices/0-1/authorized
```

This will create the PCIe tunnels and the device is now connected.

If the device supports secure connect, and the domain security level is set to `secure`, it has an additional attribute `key` which can hold a random 32-byte value used for authorization and challenging the device in future connects:

```
/sys/bus/thunderbolt/devices/0-3/authorized   - 0
/sys/bus/thunderbolt/devices/0-3/device       - 0x305
/sys/bus/thunderbolt/devices/0-3/device_name  - AKiTiO Thunder3 PCIe Box
/sys/bus/thunderbolt/devices/0-3/key          -
/sys/bus/thunderbolt/devices/0-3/vendor       - 0x41
/sys/bus/thunderbolt/devices/0-3/vendor_name  - inXtron
/sys/bus/thunderbolt/devices/0-3/unique_id    - dc010000-0000-8508-a22d-
↪32ca6421cb16
```

Notice the key is empty by default.

If the user does not want to use secure connect they can just `echo 1` to the `authorized` attribute and the PCIe tunnels will be created in the same way as in the `user` security level.

If the user wants to use secure connect, the first time the device is plugged a key needs to be created and sent to the device:

```
# key=$(openssl rand -hex 32)
# echo $key > /sys/bus/thunderbolt/devices/0-3/key
# echo 1 > /sys/bus/thunderbolt/devices/0-3/authorized
```

Now the device is connected (PCIe tunnels are created) and in addition the key is stored on the device NVM.

Next time the device is plugged in the user can verify (challenge) the device using the same key:

```
# echo $key > /sys/bus/thunderbolt/devices/0-3/key
# echo 2 > /sys/bus/thunderbolt/devices/0-3/authorized
```

If the challenge the device returns back matches the one we expect based on the key, the device is connected and the PCIe tunnels are created. However, if the challenge fails no tunnels are created and error is returned to the user.

If the user still wants to connect the device they can either approve the device without a key or write a new key and write 1 to the `authorized` file to get the new key stored on the device NVM.

78.3 De-authorizing devices

It is possible to de-authorize devices by writing `0` to their `authorized` attribute. This requires support from the connection manager implementation and can be checked by reading domain `deauthorization` attribute. If it reads `1` then the feature is supported.

When a device is de-authorized the PCIe tunnel from the parent device PCIe downstream (or root) port to the device PCIe upstream port is torn down. This is essentially the same thing as PCIe hot-remove and the PCIe toplogy in question will not be accessible anymore until the device is authorized again. If there is storage such as NVMe or similar involved, there is a risk for data loss if the filesystem on that storage is not properly shut down. You have been warned!

78.4 DMA protection utilizing IOMMU

Recent systems from 2018 and forward with Thunderbolt ports may natively support IOMMU. This means that Thunderbolt security is handled by an IOMMU so connected devices cannot access memory regions outside of what is allocated for them by drivers. When Linux is running on such system it automatically enables IOMMU if not enabled by the user already. These systems can be identified by reading `1` from `/sys/bus/thunderbolt/devices/domainX/iommu_dma_protection` attribute.

The driver does not do anything special in this case but because DMA protection is handled by the IOMMU, security levels (if set) are redundant. For this reason some systems ship with security level set to `none`. Other systems have security level set to `user` in order to support downgrade to older OS, so users who want to automatically authorize devices when IOMMU DMA protection is enabled can use the following `udev` rule:

```
ACTION=="add", SUBSYSTEM=="thunderbolt", ATTRS{iommu_dma_protection}=="1", ATTR
↪{authorized}=="0", ATTR{authorized}="1"
```

78.5 Upgrading NVM on Thunderbolt device, host or retimer

Since most of the functionality is handled in firmware running on a host controller or a device, it is important that the firmware can be upgraded to the latest where possible bugs in it have been fixed. Typically OEMs provide this firmware from their support site.

There is also a central site which has links where to download firmware for some machines:

> Thunderbolt Updates

Before you upgrade firmware on a device, host or retimer, please make sure it is a suitable upgrade. Failing to do that may render the device in a state where it cannot be used properly anymore without special tools!

Host NVM upgrade on Apple Macs is not supported.

Once the NVM image has been downloaded, you need to plug in a Thunderbolt device so that the host controller appears. It does not matter which device is connected (unless you are upgrading NVM on a device - then you need to connect that particular device).

Note an OEM-specific method to power the controller up (”force power”) may be available for your system in which case there is no need to plug in a Thunderbolt device.

After that we can write the firmware to the non-active parts of the NVM of the host or device. As an example here is how Intel NUC6i7KYK (Skull Canyon) Thunderbolt controller NVM is upgraded:

```
# dd if=KYK_TBT_FW_0018.bin of=/sys/bus/thunderbolt/devices/0-0/nvm_non_
↪active0/nvmem
```

Once the operation completes we can trigger NVM authentication and upgrade process as follows:

```
# echo 1 > /sys/bus/thunderbolt/devices/0-0/nvm_authenticate
```

If no errors are returned, the host controller shortly disappears. Once it comes back the driver notices it and initiates a full power cycle. After a while the host controller appears again and this time it should be fully functional.

We can verify that the new NVM firmware is active by running the following commands:

```
# cat /sys/bus/thunderbolt/devices/0-0/nvm_authenticate
0x0
# cat /sys/bus/thunderbolt/devices/0-0/nvm_version
18.0
```

If `nvm_authenticate` contains anything other than 0x0 it is the error code from the last authentication cycle, which means the authentication of the NVM image failed.

Note names of the NVMem devices `nvm_activeN` and `nvm_non_activeN` depend on the order they are registered in the NVMem subsystem. N in the name is the identifier added by the NVMem subsystem.

78.6 Upgrading on-board retimer NVM when there is no cable connected

If the platform supports, it may be possible to upgrade the retimer NVM firmware even when there is nothing connected to the USB4 ports. When this is the case the `usb4_portX` devices have two special attributes: `offline` and `rescan`. The way to upgrade the firmware is to first put the USB4 port into offline mode:

```
# echo 1 > /sys/bus/thunderbolt/devices/0-0/usb4_port1/offline
```

This step makes sure the port does not respond to any hotplug events, and also ensures the retimers are powered on. The next step is to scan for the retimers:

```
# echo 1 > /sys/bus/thunderbolt/devices/0-0/usb4_port1/rescan
```

This enumerates and adds the on-board retimers. Now retimer NVM can be upgraded in the same way than with cable connected (see previous section). However, the retimer is not disconnected as we are offline mode) so after writing `1` to `nvm_authenticate` one should wait for 5 or more seconds before running rescan again:

```
# echo 1 > /sys/bus/thunderbolt/devices/0-0/usb4_port1/rescan
```

This point if everything went fine, the port can be put back to functional state again:

```
# echo 0 > /sys/bus/thunderbolt/devices/0-0/usb4_port1/offline
```

78.7 Upgrading NVM when host controller is in safe mode

If the existing NVM is not properly authenticated (or is missing) the host controller goes into safe mode which means that the only available functionality is flashing a new NVM image. When in this mode, reading `nvm_version` fails with `ENODATA` and the device identification information is missing.

To recover from this mode, one needs to flash a valid NVM image to the host controller in the same way it is done in the previous chapter.

78.8 Networking over Thunderbolt cable

Thunderbolt technology allows software communication between two hosts connected by a Thunderbolt cable.

It is possible to tunnel any kind of traffic over a Thunderbolt link but currently we only support Apple ThunderboltIP protocol.

If the other host is running Windows or macOS, the only thing you need to do is to connect a Thunderbolt cable between the two hosts; the `thunderbolt-net` driver is loaded automatically. If the other host is also Linux you should load `thunderbolt-net` manually on one host (it does not matter which one):

```
# modprobe thunderbolt-net
```

This triggers module load on the other host automatically. If the driver is built-in to the kernel image, there is no need to do anything.

The driver will create one virtual ethernet interface per Thunderbolt port which are named like `thunderbolt0` and so on. From this point you can either use standard userspace tools like `ifconfig` to configure the interface or let your GUI handle it automatically.

78.9 Forcing power

Many OEMs include a method that can be used to force the power of a Thunderbolt controller to an ”On” state even if nothing is connected. If supported by your machine this will be exposed by the WMI bus with a sysfs attribute called ”force_power”.

For example the intel-wmi-thunderbolt driver exposes this attribute in:

/sys/bus/wmi/devices/86CCFD48-205E-4A77-9C48-2021CBEDE341/force_power

To force the power to on, write 1 to this attribute file. To disable force power, write 0 to this attribute file.

Note: it's currently not possible to query the force power state of a platform.

CHAPTER

SEVENTYNINE

USING UFS

mount -t ufs -o ufstype=type_of_ufs device dir

79.1 UFS Options

ufstype=type_of_ufs

UFS is a file system widely used in different operating systems. The problem are differences among implementations. Features of some implementations are undocumented, so its hard to recognize type of ufs automatically. That's why user must specify type of ufs manually by mount option ufstype. Possible values are:

old
: old format of ufs default value, supported as read-only

44bsd
: used in FreeBSD, NetBSD, OpenBSD supported as read-write

ufs2
: used in FreeBSD 5.x supported as read-write

5xbsd
: synonym for ufs2

sun
: used in SunOS (Solaris) supported as read-write

sunx86
: used in SunOS for Intel (Solarisx86) supported as read-write

hp
: used in HP-UX supported as read-only

nextstep
: used in NextStep supported as read-only

nextstep-cd
: used for NextStep CDROMs (block_size == 2048) supported as read-only

openstep
: used in OpenStep supported as read-only

79.1.1 Possible Problems

See next section, if you have any.

79.1.2 Bug Reports

Any ufs bug report you can send to daniel.pirkl@email.cz or to dushistov@mail.ru (do not send partition tables bug reports).

CHAPTER

EIGHTY

UNICODE SUPPORT

Last update: 2005-01-17, version 1.4

Note: The original version of this document, which was maintained at lanana.org as part of the Linux Assigned Names And Numbers Authority (LANANA) project, is no longer existent. So, this version in the mainline Linux kernel is now the maintained main document.

80.1 Introduction

The Linux kernel code has been rewritten to use Unicode to map characters to fonts. By downloading a single Unicode-to-font table, both the eight-bit character sets and UTF-8 mode are changed to use the font as indicated.

This changes the semantics of the eight-bit character tables subtly. The four character tables are now:

Map symbol	Map name	Escape code (G0)
LAT1_MAP	Latin-1 (ISO 8859-1)	ESC (B
GRAF_MAP	DEC VT100 pseudographics	ESC (0
IBMPC_MAP	IBM code page 437	ESC (U
USER_MAP	User defined	ESC (K

In particular, ESC (U is no longer "straight to font", since the font might be completely different than the IBM character set. This permits for example the use of block graphics even with a Latin-1 font loaded.

Note that although these codes are similar to ISO 2022, neither the codes nor their uses match ISO 2022; Linux has two 8-bit codes (G0 and G1), whereas ISO 2022 has four 7-bit codes (G0-G3).

In accordance with the Unicode standard/ISO 10646 the range U+F000 to U+F8FF has been reserved for OS-wide allocation (the Unicode Standard refers to this as a "Corporate Zone", since this is inaccurate for Linux we call it the "Linux Zone"). U+F000 was picked as the starting point since it lets the direct-mapping area start on a large power of two (in case 1024- or 2048-character fonts ever become necessary). This leaves U+E000 to U+EFFF as End User Zone.

[v1.2]: The Unicodes range from U+F000 and up to U+F7FF have been hard-coded to map directly to the loaded font, bypassing the translation table. The user-defined map now defaults to U+F000 to U+F0FF, emulating the previous behaviour. In practice, this range might

be shorter; for example, vgacon can only handle 256-character (U+F000..U+F0FF) or 512-character (U+F000..U+F1FF) fonts.

80.2 Actual characters assigned in the Linux Zone

In addition, the following characters not present in Unicode 1.1.4 have been defined; these are used by the DEC VT graphics map. [v1.2] THIS USE IS OBSOLETE AND SHOULD NO LONGER BE USED; PLEASE SEE BELOW.

U+F800	DEC VT GRAPHICS HORIZONTAL LINE SCAN 1
U+F801	DEC VT GRAPHICS HORIZONTAL LINE SCAN 3
U+F803	DEC VT GRAPHICS HORIZONTAL LINE SCAN 7
U+F804	DEC VT GRAPHICS HORIZONTAL LINE SCAN 9

The DEC VT220 uses a 6x10 character matrix, and these characters form a smooth progression in the DEC VT graphics character set. I have omitted the scan 5 line, since it is also used as a block-graphics character, and hence has been coded as U+2500 FORMS LIGHT HORIZONTAL.

[v1.3]: These characters have been officially added to Unicode 3.2.0; they are added at U+23BA, U+23BB, U+23BC, U+23BD. Linux now uses the new values.

[v1.2]: The following characters have been added to represent common keyboard symbols that are unlikely to ever be added to Unicode proper since they are horribly vendor-specific. This, of course, is an excellent example of horrible design.

U+F810	KEYBOARD SYMBOL FLYING FLAG
U+F811	KEYBOARD SYMBOL PULLDOWN MENU
U+F812	KEYBOARD SYMBOL OPEN APPLE
U+F813	KEYBOARD SYMBOL SOLID APPLE

80.3 Klingon language support

In 1996, Linux was the first operating system in the world to add support for the artificial language Klingon, created by Marc Okrand for the "Star Trek" television series. This encoding was later adopted by the ConScript Unicode Registry and proposed (but ultimately rejected) for inclusion in Unicode Plane 1. Thus, it remains as a Linux/CSUR private assignment in the Linux Zone.

This encoding has been endorsed by the Klingon Language Institute. For more information, contact them at:

http://www.kli.org/

Since the characters in the beginning of the Linux CZ have been more of the dingbats/symbols/forms type and this is a language, I have located it at the end, on a 16-cell boundary in keeping with standard Unicode practice.

Note: This range is now officially managed by the ConScript Unicode Registry. The normative reference is at:

https://www.evertype.com/standards/csur/klingon.html

Klingon has an alphabet of 26 characters, a positional numeric writing system with 10 digits, and is written left-to-right, top-to-bottom.

Several glyph forms for the Klingon alphabet have been proposed. However, since the set of symbols appear to be consistent throughout, with only the actual shapes being different, in keeping with standard Unicode practice these differences are considered font variants.

U+F8D0	KLINGON LETTER A
U+F8D1	KLINGON LETTER B
U+F8D2	KLINGON LETTER CH
U+F8D3	KLINGON LETTER D
U+F8D4	KLINGON LETTER E
U+F8D5	KLINGON LETTER GH
U+F8D6	KLINGON LETTER H
U+F8D7	KLINGON LETTER I
U+F8D8	KLINGON LETTER J
U+F8D9	KLINGON LETTER L
U+F8DA	KLINGON LETTER M
U+F8DB	KLINGON LETTER N
U+F8DC	KLINGON LETTER NG
U+F8DD	KLINGON LETTER O
U+F8DE	KLINGON LETTER P
U+F8DF	KLINGON LETTER Q - Written <q> in standard Okrand Latin transliteration
U+F8E0	KLINGON LETTER QH - Written <Q> in standard Okrand Latin transliteration
U+F8E1	KLINGON LETTER R
U+F8E2	KLINGON LETTER S
U+F8E3	KLINGON LETTER T
U+F8E4	KLINGON LETTER TLH
U+F8E5	KLINGON LETTER U
U+F8E6	KLINGON LETTER V
U+F8E7	KLINGON LETTER W
U+F8E8	KLINGON LETTER Y
U+F8E9	KLINGON LETTER GLOTTAL STOP
U+F8F0	KLINGON DIGIT ZERO
U+F8F1	KLINGON DIGIT ONE
U+F8F2	KLINGON DIGIT TWO
U+F8F3	KLINGON DIGIT THREE
U+F8F4	KLINGON DIGIT FOUR
U+F8F5	KLINGON DIGIT FIVE
U+F8F6	KLINGON DIGIT SIX
U+F8F7	KLINGON DIGIT SEVEN
U+F8F8	KLINGON DIGIT EIGHT
U+F8F9	KLINGON DIGIT NINE
U+F8FD	KLINGON COMMA
U+F8FE	KLINGON FULL STOP
U+F8FF	KLINGON SYMBOL FOR EMPIRE

80.4 Other Fictional and Artificial Scripts

Since the assignment of the Klingon Linux Unicode block, a registry of fictional and artificial scripts has been established by John Cowan <jcowan@reutershealth.com> and Michael Everson <everson@evertype.com>. The ConScript Unicode Registry is accessible at:

> https://www.evertype.com/standards/csur/

The ranges used fall at the low end of the End User Zone and can hence not be normatively assigned, but it is recommended that people who wish to encode fictional scripts use these codes, in the interest of interoperability. For Klingon, CSUR has adopted the Linux encoding. The CSUR people are driving adding Tengwar and Cirth into Unicode Plane 1; the addition of Klingon to Unicode Plane 1 has been rejected and so the above encoding remains official.

CHAPTER

EIGHTYONE

SOFTWARE CURSOR FOR VGA

by Pavel Machek <pavel@atrey.karlin.mff.cuni.cz> and Martin Mares <mj@atrey.karlin.mff.cuni.cz>

Linux now has some ability to manipulate cursor appearance. Normally, you can set the size of hardware cursor. You can now play a few new tricks: you can make your cursor look like a non-blinking red block, make it inverse background of the character it's over or to highlight that character and still choose whether the original hardware cursor should remain visible or not. There may be other things I have never thought of.

The cursor appearance is controlled by a `<ESC>[?1;2;3c` escape sequence where 1, 2 and 3 are parameters described below. If you omit any of them, they will default to zeroes.

first Parameter
specifies cursor size:

```
0=default
1=invisible
2=underline,
...
8=full block
+ 16 if you want the software cursor to be applied
+ 32 if you want to always change the background color
+ 64 if you dislike having the background the same as the
     foreground.
```

Highlights are ignored for the last two flags.

second parameter
selects character attribute bits you want to change (by simply XORing them with the value of this parameter). On standard VGA, the high four bits specify background and the low four the foreground. In both groups, low three bits set color (as in normal color codes used by the console) and the most significant one turns on highlight (or sometimes blinking -- it depends on the configuration of your VGA).

third parameter
consists of character attribute bits you want to set.

Bit setting takes place before bit toggling, so you can simply clear a bit by including it in both the set mask and the toggle mask.

81.1 Examples

To get normal blinking underline, use:

```
echo -e '\033[?2c'
```

To get blinking block, use:

```
echo -e '\033[?6c'
```

To get red non-blinking block, use:

```
echo -e '\033[?17;0;64c'
```

CHAPTER

EIGHTYTWO

VIDEO OUTPUT SWITCHER CONTROL

2006 luming.yu@intel.com

The output sysfs class driver provides an abstract video output layer that can be used to hook platform specific methods to enable/disable video output device through common sysfs interface. For example, on my IBM ThinkPad T42 laptop, The ACPI video driver registered its output devices and read/write method for 'state' with output sysfs class. The user interface under sysfs is:

```
linux:/sys/class/video_output # tree .
.
|-- CRT0
|   |-- device -> ../../../devices/pci0000:00/0000:00:01.0
|   |-- state
|   |-- subsystem -> ../../../class/video_output
|   `-- uevent
|-- DVI0
|   |-- device -> ../../../devices/pci0000:00/0000:00:01.0
|   |-- state
|   |-- subsystem -> ../../../class/video_output
|   `-- uevent
|-- LCD0
|   |-- device -> ../../../devices/pci0000:00/0000:00:01.0
|   |-- state
|   |-- subsystem -> ../../../class/video_output
|   `-- uevent
`-- TV0
   |-- device -> ../../../devices/pci0000:00/0000:00:01.0
   |-- state
   |-- subsystem -> ../../../class/video_output
   `-- uevent
```

CHAPTER

EIGHTYTHREE

THE SGI XFS FILESYSTEM

XFS is a high performance journaling filesystem which originated on the SGI IRIX platform. It is completely multi-threaded, can support large files and large filesystems, extended attributes, variable block sizes, is extent based, and makes extensive use of Btrees (directories, extents, free space) to aid both performance and scalability.

Refer to the documentation at https://xfs.wiki.kernel.org/ for further details. This implementation is on-disk compatible with the IRIX version of XFS.

83.1 Mount Options

When mounting an XFS filesystem, the following options are accepted.

allocsize=size

Sets the buffered I/O end-of-file preallocation size when doing delayed allocation writeout (default size is 64KiB). Valid values for this option are page size (typically 4KiB) through to 1GiB, inclusive, in power-of-2 increments.

The default behaviour is for dynamic end-of-file preallocation size, which uses a set of heuristics to optimise the preallocation size based on the current allocation patterns within the file and the access patterns to the file. Specifying a fixed `allocsize` value turns off the dynamic behaviour.

attr2 or noattr2

The options enable/disable an "opportunistic" improvement to be made in the way inline extended attributes are stored on-disk. When the new form is used for the first time when `attr2` is selected (either when setting or removing extended attributes) the on-disk superblock feature bit field will be updated to reflect this format being in use.

The default behaviour is determined by the on-disk feature bit indicating that `attr2` behaviour is active. If either mount option is set, then that becomes the new default used by the filesystem.

CRC enabled filesystems always use the `attr2` format, and so will reject the `noattr2` mount option if it is set.

discard or nodiscard (default)

Enable/disable the issuing of commands to let the block device reclaim space freed by the filesystem. This is useful for SSD devices, thinly provisioned LUNs and virtual machine images, but may have a performance impact.

Note: It is currently recommended that you use the `fstrim` application to `discard` unused blocks rather than the `discard` mount option because the performance impact of this option is quite severe.

grpid/bsdgroups or nogrpid/sysvgroups (default)
These options define what group ID a newly created file gets. When `grpid` is set, it takes the group ID of the directory in which it is created; otherwise it takes the `fsgid` of the current process, unless the directory has the `setgid` bit set, in which case it takes the `gid` from the parent directory, and also gets the `setgid` bit set if it is a directory itself.

filestreams
Make the data allocator use the filestreams allocation mode across the entire filesystem rather than just on directories configured to use it.

ikeep or noikeep (default)
When `ikeep` is specified, XFS does not delete empty inode clusters and keeps them around on disk. When `noikeep` is specified, empty inode clusters are returned to the free space pool.

inode32 or inode64 (default)
When `inode32` is specified, it indicates that XFS limits inode creation to locations which will not result in inode numbers with more than 32 bits of significance.

When `inode64` is specified, it indicates that XFS is allowed to create inodes at any location in the filesystem, including those which will result in inode numbers occupying more than 32 bits of significance.

`inode32` is provided for backwards compatibility with older systems and applications, since 64 bits inode numbers might cause problems for some applications that cannot handle large inode numbers. If applications are in use which do not handle inode numbers bigger than 32 bits, the `inode32` option should be specified.

largeio or nolargeio (default)
If `nolargeio` is specified, the optimal I/O reported in `st_blksize` by **stat(2)** will be as small as possible to allow user applications to avoid inefficient read/modify/write I/O. This is typically the page size of the machine, as this is the granularity of the page cache.

If `largeio` is specified, a filesystem that was created with a `swidth` specified will return the `swidth` value (in bytes) in `st_blksize`. If the filesystem does not have a `swidth` specified but does specify an `allocsize` then `allocsize` (in bytes) will be returned instead. Otherwise the behaviour is the same as if `nolargeio` was specified.

logbufs=value
Set the number of in-memory log buffers. Valid numbers range from 2-8 inclusive.

The default value is 8 buffers.

If the memory cost of 8 log buffers is too high on small systems, then it may be reduced at some cost to performance on metadata intensive workloads. The `logbsize` option below controls the size of each buffer and so is also relevant to this case.

logbsize=value

Set the size of each in-memory log buffer. The size may be specified in bytes, or in kilobytes with a "k" suffix. Valid sizes for version 1 and version 2 logs are 16384 (16k) and 32768 (32k). Valid sizes for version 2 logs also include 65536 (64k), 131072 (128k) and 262144 (256k). The logbsize must be an integer multiple of the log stripe unit configured at **mkfs(8)** time.

The default value for version 1 logs is 32768, while the default value for version 2 logs is MAX(32768, log_sunit).

logdev=device and rtdev=device
Use an external log (metadata journal) and/or real-time device. An XFS filesystem has up to three parts: a data section, a log section, and a real-time section. The real-time section is optional, and the log section can be separate from the data section or contained within it.

noalign
Data allocations will not be aligned at stripe unit boundaries. This is only relevant to filesystems created with non-zero data alignment parameters (`sunit`, `swidth`) by **mkfs(8)**.

norecovery
The filesystem will be mounted without running log recovery. If the filesystem was not cleanly unmounted, it is likely to be inconsistent when mounted in `norecovery` mode. Some files or directories may not be accessible because of this. Filesystems mounted `norecovery` must be mounted read-only or the mount will fail.

nouuid
Don't check for double mounted file systems using the file system `uuid`. This is useful to mount LVM snapshot volumes, and often used in combination with `norecovery` for mounting read-only snapshots.

noquota
Forcibly turns off all quota accounting and enforcement within the filesystem.

uquota/usrquota/uqnoenforce/quota
User disk quota accounting enabled, and limits (optionally) enforced. Refer to **xfs_quota(8)** for further details.

gquota/grpquota/gqnoenforce
Group disk quota accounting enabled and limits (optionally) enforced. Refer to **xfs_quota(8)** for further details.

pquota/prjquota/pqnoenforce
Project disk quota accounting enabled and limits (optionally) enforced. Refer to **xfs_quota(8)** for further details.

sunit=value and swidth=value
Used to specify the stripe unit and width for a RAID device or a stripe volume. "value" must be specified in 512-byte block units. These options are only relevant to filesystems that were created with non-zero data alignment parameters.

The `sunit` and `swidth` parameters specified must be compatible with the existing filesystem alignment characteristics. In general, that means the only valid changes to `sunit` are increasing it by a power-of-2 multiple. Valid `swidth` values are any integer multiple of a valid `sunit` value.

Typically the only time these mount options are necessary if after an underlying RAID device has had its geometry modified, such as adding a new disk to a RAID5 lun and reshaping it.

swalloc
Data allocations will be rounded up to stripe width boundaries when the current end of file is being extended and the file size is larger than the stripe width size.

wsync
When specified, all filesystem namespace operations are executed synchronously. This ensures that when the namespace operation (create, unlink, etc) completes, the change to the namespace is on stable storage. This is useful in HA setups where failover must not result in clients seeing inconsistent namespace presentation during or after a failover event.

83.2 Deprecation of V4 Format

The V4 filesystem format lacks certain features that are supported by the V5 format, such as metadata checksumming, strengthened metadata verification, and the ability to store timestamps past the year 2038. Because of this, the V4 format is deprecated. All users should upgrade by backing up their files, reformatting, and restoring from the backup.

Administrators and users can detect a V4 filesystem by running xfs_info against a filesystem mountpoint and checking for a string containing ”crc=”. If no such string is found, please upgrade xfsprogs to the latest version and try again.

The deprecation will take place in two parts. Support for mounting V4 filesystems can now be disabled at kernel build time via Kconfig option. The option will default to yes until September 2025, at which time it will be changed to default to no. In September 2030, support will be removed from the codebase entirely.

Note: Distributors may choose to withdraw V4 format support earlier than the dates listed above.

83.3 Deprecated Mount Options

Name	Removal Schedule
Mounting with V4 filesystem	September 2030
Mounting ascii-ci filesystem	September 2030
ikeep/noikeep	September 2025
attr2/noattr2	September 2025

83.4 Removed Mount Options

Name	Removed
delaylog/nodelaylog	v4.0
ihashsize	v4.0
irixsgid	v4.0
osyncisdsync/osyncisosync	v4.0
barrier	v4.19
nobarrier	v4.19

83.5 sysctls

The following sysctls are available for the XFS filesystem:

fs.xfs.stats_clear (Min: 0 Default: 0 Max: 1)
Setting this to "1" clears accumulated XFS statistics in /proc/fs/xfs/stat. It then immediately resets to "0".

fs.xfs.xfssyncd_centisecs (Min: 100 Default: 3000 Max: 720000)
The interval at which the filesystem flushes metadata out to disk and runs internal cache cleanup routines.

fs.xfs.filestream_centisecs (Min: 1 Default: 3000 Max: 360000)
The interval at which the filesystem ages filestreams cache references and returns timed-out AGs back to the free stream pool.

fs.xfs.speculative_prealloc_lifetime
(Units: seconds Min: 1 Default: 300 Max: 86400) The interval at which the background scanning for inodes with unused speculative preallocation runs. The scan removes unused preallocation from clean inodes and releases the unused space back to the free pool.

fs.xfs.speculative_cow_prealloc_lifetime
This is an alias for speculative_prealloc_lifetime.

fs.xfs.error_level (Min: 0 Default: 3 Max: 11)
A volume knob for error reporting when internal errors occur. This will generate detailed messages & backtraces for filesystem shutdowns, for example. Current threshold values are:

XFS_ERRLEVEL_OFF: 0 XFS_ERRLEVEL_LOW: 1
XFS_ERRLEVEL_HIGH: 5

fs.xfs.panic_mask (Min: 0 Default: 0 Max: 511)
Causes certain error conditions to call BUG(). Value is a bitmask; OR together the tags which represent errors which should cause panics:

XFS_NO_PTAG 0 XFS_PTAG_IFLUSH 0x00000001
XFS_PTAG_LOGRES 0x00000002 XFS_PTAG_AILDELETE
0x00000004 XFS_PTAG_ERROR_REPORT 0x00000008
XFS_PTAG_SHUTDOWN_CORRUPT 0x00000010
XFS_PTAG_SHUTDOWN_IOERROR 0x00000020
XFS_PTAG_SHUTDOWN_LOGERROR 0x00000040

XFS_PTAG_FSBLOCK_ZERO 0x00000080 XFS_PTAG_VERIFIER_ERROR 0x00000100

This option is intended for debugging only.

fs.xfs.irix_symlink_mode (Min: 0 Default: 0 Max: 1)
Controls whether symlinks are created with mode 0777 (default) or whether their mode is affected by the umask (irix mode).

fs.xfs.irix_sgid_inherit (Min: 0 Default: 0 Max: 1)
Controls files created in SGID directories. If the group ID of the new file does not match the effective group ID or one of the supplementary group IDs of the parent dir, the ISGID bit is cleared if the irix_sgid_inherit compatibility sysctl is set.

fs.xfs.inherit_sync (Min: 0 Default: 1 Max: 1)
Setting this to ”1” will cause the ”sync” flag set by the **xfs_io(8)** chattr command on a directory to be inherited by files in that directory.

fs.xfs.inherit_nodump (Min: 0 Default: 1 Max: 1)
Setting this to ”1” will cause the ”nodump” flag set by the **xfs_io(8)** chattr command on a directory to be inherited by files in that directory.

fs.xfs.inherit_noatime (Min: 0 Default: 1 Max: 1)
Setting this to ”1” will cause the ”noatime” flag set by the **xfs_io(8)** chattr command on a directory to be inherited by files in that directory.

fs.xfs.inherit_nosymlinks (Min: 0 Default: 1 Max: 1)
Setting this to ”1” will cause the ”nosymlinks” flag set by the **xfs_io(8)** chattr command on a directory to be inherited by files in that directory.

fs.xfs.inherit_nodefrag (Min: 0 Default: 1 Max: 1)
Setting this to ”1” will cause the ”nodefrag” flag set by the **xfs_io(8)** chattr command on a directory to be inherited by files in that directory.

fs.xfs.rotorstep (Min: 1 Default: 1 Max: 256)
In ”inode32” allocation mode, this option determines how many files the allocator attempts to allocate in the same allocation group before moving to the next allocation group. The intent is to control the rate at which the allocator moves between allocation groups when allocating extents for new files.

83.6 Deprecated Sysctls

Name	Removal Schedule
fs.xfs.irix_sgid_inherit	September 2025
fs.xfs.irix_symlink_mode	September 2025
fs.xfs.speculative_cow_prealloc_lifetime	September 2025

83.7 Removed Sysctls

Name	Removed
fs.xfs.xfsbufd_centisec	v4.0
fs.xfs.age_buffer_centisecs	v4.0

83.8 Error handling

XFS can act differently according to the type of error found during its operation. The implementation introduces the following concepts to the error handler:

-failure speed:
Defines how fast XFS should propagate an error upwards when a specific error is found during the filesystem operation. It can propagate immediately, after a defined number of retries, after a set time period, or simply retry forever.

-error classes:
Specifies the subsystem the error configuration will apply to, such as metadata IO or memory allocation. Different subsystems will have different error handlers for which behaviour can be configured.

-error handlers:
Defines the behavior for a specific error.

The filesystem behavior during an error can be set via `sysfs` files. Each error handler works independently - the first condition met by an error handler for a specific class will cause the error to be propagated rather than reset and retried.

The action taken by the filesystem when the error is propagated is context dependent - it may cause a shut down in the case of an unrecoverable error, it may be reported back to userspace, or it may even be ignored because there's nothing useful we can with the error or anyone we can report it to (e.g. during unmount).

The configuration files are organized into the following hierarchy for each mounted filesystem:

/sys/fs/xfs/<dev>/error/<class>/<error>/

Where:

<dev>
The short device name of the mounted filesystem. This is the same device name that shows up in XFS kernel error messages as "XFS(<dev>): ..."

<class>
The subsystem the error configuration belongs to. As of 4.9, the defined classes are:

- "metadata": applies metadata buffer write IO

<error>
The individual error handler configurations.

Each filesystem has "global" error configuration options defined in their top level directory:

/sys/fs/xfs/<dev>/error/

fail_at_unmount (Min: 0 Default: 1 Max: 1)
Defines the filesystem error behavior at unmount time.

If set to a value of 1, XFS will override all other error configurations during unmount and replace them with ”immediate fail” characteristics. i.e. no retries, no retry timeout. This will always allow unmount to succeed when there are persistent errors present.

If set to 0, the configured retry behaviour will continue until all retries and/or timeouts have been exhausted. This will delay unmount completion when there are persistent errors, and it may prevent the filesystem from ever unmounting fully in the case of ”retry forever” handler configurations.

Note: there is no guarantee that fail_at_unmount can be set while an unmount is in progress. It is possible that the `sysfs` entries are removed by the unmounting filesystem before a ”retry forever” error handler configuration causes unmount to hang, and hence the filesystem must be configured appropriately before unmount begins to prevent unmount hangs.

Each filesystem has specific error class handlers that define the error propagation behaviour for specific errors. There is also a ”default” error handler defined, which defines the behaviour for all errors that don't have specific handlers defined. Where multiple retry constraints are configured for a single error, the first retry configuration that expires will cause the error to be propagated. The handler configurations are found in the directory:

/sys/fs/xfs/<dev>/error/<class>/<error>/

max_retries (Min: -1 Default: Varies Max: INTMAX)
Defines the allowed number of retries of a specific error before the filesystem will propagate the error. The retry count for a given error context (e.g. a specific metadata buffer) is reset every time there is a successful completion of the operation.

Setting the value to ”-1” will cause XFS to retry forever for this specific error.

Setting the value to ”0” will cause XFS to fail immediately when the specific error is reported.

Setting the value to ”N” (where 0 < N < Max) will make XFS retry the operation ”N” times before propagating the error.

retry_timeout_seconds (Min: -1 Default: Varies Max: 1 day)
Define the amount of time (in seconds) that the filesystem is allowed to retry its operations when the specific error is found.

Setting the value to ”-1” will allow XFS to retry forever for this specific error.

Setting the value to ”0” will cause XFS to fail immediately when the specific error is reported.

Setting the value to ”N” (where 0 < N < Max) will allow XFS to retry the operation for up to ”N” seconds before propagating the error.

Note: The default behaviour for a specific error handler is dependent on both the class and error context. For example, the default values for ”metadata/ENODEV” are ”0” rather than ”-1” so that this error handler defaults to ”fail immediately” behaviour. This is done because ENODEV is a fatal, unrecoverable error no matter how many times the metadata IO is retried.

83.9 Workqueue Concurrency

XFS uses kernel workqueues to parallelize metadata update processes. This enables it to take advantage of storage hardware that can service many IO operations simultaneously. This interface exposes internal implementation details of XFS, and as such is explicitly not part of any userspace API/ABI guarantee the kernel may give userspace. These are undocumented features of the generic workqueue implementation XFS uses for concurrency, and they are provided here purely for diagnostic and tuning purposes and may change at any time in the future.

The control knobs for a filesystem's workqueues are organized by task at hand and the short name of the data device. They all can be found in:

/sys/bus/workqueue/devices/${task}!${device}

Task	Description
xfs_iwalk-$pid	Inode scans of the entire filesystem. Currently limited to mount time quotacheck.
xfs-gc	Background garbage collection of disk space that have been speculatively allocated beyond EOF or for staging copy on write operations.

For example, the knobs for the quotacheck workqueue for /dev/nvme0n1 would be found in /sys/bus/workqueue/devices/xfs_iwalk-1111!nvme0n1/.

The interesting knobs for XFS workqueues are as follows:

Knob	Description
max_active	Maximum number of background threads that can be started to run the work.
cpumask	CPUs upon which the threads are allowed to run.
nice	Relative priority of scheduling the threads. These are the same nice levels that can be applied to userspace processes.

INDEX

F

fsl_local_config_read (*C function*), 2653
fsl_local_config_write (*C function*), 2654
fsl_rio_config_read (*C function*), 2654
fsl_rio_config_write (*C function*), 2655
fsl_rio_setup (*C function*), 2655

P

pstore_blk_config (*C struct*), 1599
pstore_blk_get_config (*C function*), 1599
pstore_device_info (*C struct*), 1599
pstore_zone (*C struct*), 1595
pstore_zone_info (*C struct*), 1597
psz_buffer (*C struct*), 1594
psz_context (*C struct*), 1595
psz_flush_mode (*C enum*), 1597
psz_kmsg_header (*C struct*), 1594
psz_recovery (*C function*), 1597

R

register_pstore_device (*C function*), 1592
rio_add_device (*C function*), 2618
rio_add_inb_buffer (*C function*), 2615
rio_add_mport_pw_handler (*C function*), 2620
rio_add_outb_message (*C function*), 2614
rio_alloc_net (*C function*), 2617
rio_basic_attach (*C function*), 2647
rio_build_route_tables (*C function*), 2647
rio_bus_init (*C function*), 2653
rio_chk_dev_access (*C function*), 2649
rio_chk_dev_route (*C function*), 2648
rio_clear_locks (*C function*), 2643
rio_clr_err_stopped (*C function*), 2649
rio_dbell (*C struct*), 2634
rio_del_mport_pw_handler (*C function*), 2621
rio_destid_alloc (*C function*), 2641
rio_destid_first (*C function*), 2642
rio_destid_free (*C function*), 2642
rio_destid_next (*C function*), 2642
rio_destid_reserve (*C function*), 2641
rio_dev (*C struct*), 2632
rio_dev_get (*C function*), 2616
rio_dev_put (*C function*), 2616
RIO_DEVICE (*C macro*), 2614
rio_device_has_destid (*C function*), 2643
rio_device_probe (*C function*), 2652
rio_device_remove (*C function*), 2653
rio_disc_mport (*C function*), 2647
rio_disc_peer (*C function*), 2646
rio_dma_prep_slave_sg (*C function*), 2629
rio_dma_prep_xfer (*C function*), 2628
rio_driver (*C struct*), 2639
rio_enable_rx_tx_port (*C function*), 2624
rio_enum_complete (*C function*), 2645
rio_enum_host (*C function*), 2643
rio_enum_mport (*C function*), 2647
rio_enum_peer (*C function*), 2645
rio_find_mport (*C function*), 2650
rio_get_comptag (*C function*), 2624
rio_get_device_id (*C function*), 2642
rio_get_drvdata (*C function*), 2615
rio_get_host_deviceid_lock (*C function*), 2645
rio_get_inb_message (*C function*), 2615
rio_get_input_status (*C function*), 2649
rio_inb_pwrite_handler (*C function*), 2625
rio_init_dbell_res (*C function*), 2614
rio_init_em (*C function*), 2647
rio_init_mbox_res (*C function*), 2613
rio_is_switch (*C function*), 2644
rio_local_get_device_id (*C function*), 2617
rio_local_read_config_16 (*C function*), 2610
rio_local_read_config_32 (*C function*), 2610
rio_local_read_config_8 (*C function*), 2611
rio_local_set_device_id (*C function*), 2617
rio_local_write_config_16 (*C function*), 2610
rio_local_write_config_32 (*C function*),

2610
rio_local_write_config_8 (*C function*), 2611
rio_lock_device (*C function*), 2626
RIO_LOP_READ (*C macro*), 2651
RIO_LOP_WRITE (*C macro*), 2651
rio_map_inb_region (*C function*), 2622
rio_map_outb_region (*C function*), 2622
rio_match_bus (*C function*), 2653
rio_match_device (*C function*), 2652
rio_mport (*C struct*), 2635
rio_mport_attr (*C struct*), 2637
rio_mport_chk_dev_access (*C function*), 2625
rio_mport_get_efb (*C function*), 2625
rio_mport_get_feature (*C function*), 2625
rio_mport_get_physefb (*C function*), 2623
rio_mport_is_active (*C function*), 2646
rio_mport_scan (*C function*), 2651
rio_msg (*C struct*), 2634
rio_name (*C function*), 2615
rio_net (*C struct*), 2636
RIO_OP_READ (*C macro*), 2651
RIO_OP_WRITE (*C macro*), 2652
rio_ops (*C struct*), 2638
rio_pw_enable (*C function*), 2622
rio_query_mport (*C function*), 2617
rio_read_config_16 (*C function*), 2612
rio_read_config_32 (*C function*), 2611
rio_read_config_8 (*C function*), 2612
rio_register_driver (*C function*), 2616
rio_register_scan (*C function*), 2629
rio_release_dev (*C function*), 2644
rio_release_dma (*C function*), 2628
rio_release_inb_dbell (*C function*), 2620
rio_release_inb_mbox (*C function*), 2618
rio_release_inb_pwrite (*C function*), 2621
rio_release_outb_dbell (*C function*), 2620
rio_release_outb_mbox (*C function*), 2619
rio_request_dma (*C function*), 2628
rio_request_inb_dbell (*C function*), 2619
rio_request_inb_mbox (*C function*), 2618
rio_request_inb_pwrite (*C function*), 2621
rio_request_mport_dma (*C function*), 2628
rio_request_outb_dbell (*C function*), 2620
rio_request_outb_mbox (*C function*), 2618
rio_route_add_entry (*C function*), 2626
rio_route_clr_table (*C function*), 2627
rio_route_get_entry (*C function*), 2627
rio_scan (*C struct*), 2640
rio_scan_node (*C struct*), 2641
rio_send_doorbell (*C function*), 2613
rio_set_device_id (*C function*), 2642
rio_set_drvdata (*C function*), 2616
rio_set_port_lockout (*C function*), 2624
rio_setup_device (*C function*), 2644
rio_setup_inb_dbell (*C function*), 2648
rio_sport_is_active (*C function*), 2644
rio_std_route_add_entry (*C function*), 2649
rio_std_route_clr_table (*C function*), 2650
rio_std_route_get_entry (*C function*), 2650
rio_switch (*C struct*), 2630
rio_switch_ops (*C struct*), 2631
rio_unlock_device (*C function*), 2626
rio_unmap_inb_region (*C function*), 2622
rio_unmap_outb_region (*C function*), 2623
rio_unregister_driver (*C function*), 2616
rio_unregister_scan (*C function*), 2630
rio_update_route_tables (*C function*), 2646
rio_write_config_16 (*C function*), 2612
rio_write_config_32 (*C function*), 2612
rio_write_config_8 (*C function*), 2613

U

unregister_pstore_device (*C function*), 1592

X

xbc_array_for_each_value (*C macro*), 1688
xbc_calc_checksum (*C function*), 1687
xbc_exit (*C function*), 1694
xbc_find_node (*C function*), 1688
xbc_find_value (*C function*), 1688
xbc_for_each_key_value (*C macro*), 1690
xbc_get_info (*C function*), 1691
xbc_init (*C function*), 1694
xbc_node_compose_key (*C function*), 1690
xbc_node_compose_key_after (*C function*), 1693
xbc_node_find_next_key_value (*C function*), 1693
xbc_node_find_next_leaf (*C function*), 1693
xbc_node_find_subkey (*C function*), 1692
xbc_node_find_value (*C function*), 1692
xbc_node_for_each_array_value (*C macro*), 1689
xbc_node_for_each_child (*C macro*), 1689
xbc_node_for_each_key_value (*C macro*), 1690
xbc_node_for_each_subkey (*C macro*), 1689
xbc_node_get_child (*C function*), 1691
xbc_node_get_data (*C function*), 1692
xbc_node_get_next (*C function*), 1692

xbc_node_get_parent (*C function*), 1691
xbc_node_get_subkey (*C function*), 1688
xbc_node_index (*C function*), 1691
xbc_node_is_array (*C function*), 1687
xbc_node_is_key (*C function*), 1687
xbc_node_is_leaf (*C function*), 1687
xbc_node_is_value (*C function*), 1687
xbc_root_node (*C function*), 1691