阿里云 | 平头哥

RISC-V® 中国峰会2022

QEMU & LINUX Support for RV32 Compat Mode

Liu Zhiwei & Guo Ren

T-Head Software Engineer

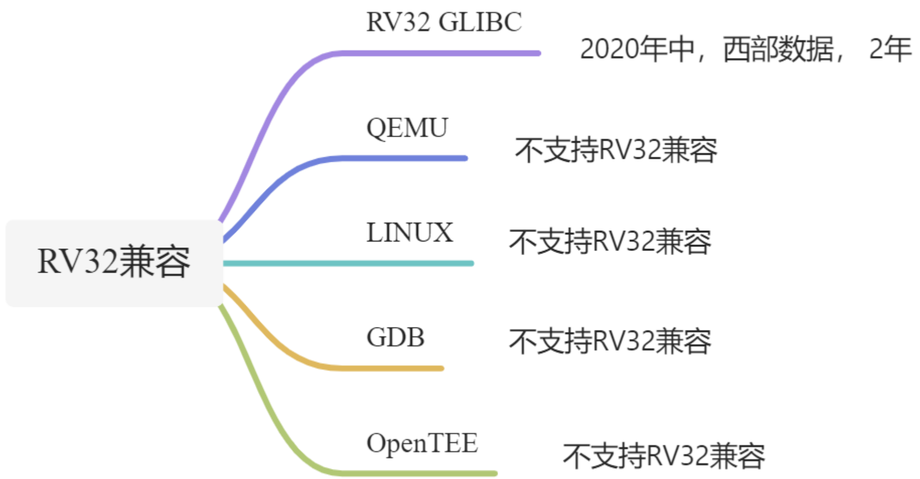
2022/08/24

Question

- Can we run RV32 application on 64-bit RISC-V CPU
  - Interesting & Complicated
  - Most architectures can run 32-bit application on 64-bit CPU
  - RISC-V does not have so many legacy 32-bit application

Exploring

- **Prerequisites**
  - Compiler for RV32 is ready
- **Still need to exploring**
  - Understanding the specification
  - Build an executive environment on QEMU
  - Necessary Linux kernel support



Run RV32 rootfs on RV64 Linux kernel

```
➜  build ./qemu-system-riscv64 -cpu rv64 -D 1.log -M virt -m 1024m -nographic -bios ../fw_jump.bin -kernel
 root=/dev/vda ro console=ttyS0 earlycon=sbi norandmaps" -netdev user,id=net0 -device virtio-net-device,net
...
OpenSBI v1.0-34-g7924a0b220b7
...
[    0.000000] Linux version 5.19.0+ (guoren@fedora) (riscv64-unknown-linux-gnu-gcc (g5964b5cd7272) 11.1.0,
[    0.000000] OF: fdt: Ignoring memory range 0x40000000 - 0x40200000
[    0.000000] Machine model: riscv-virtio,qemu
[    0.000000] earlycon: sbi0 at I/O port 0x0 (options '')
[    0.000000] printk: bootconsole [sbi0] enabled
[    0.000000] efi: UEFI not found.
[    0.000000] Zone ranges:
[    0.000000]   DMA32    [mem 0x0000000040200000-0x00000000ffffffff]
[    0.000000]   Normal   empty
[    0.000000] Movable zone start for each node
[    0.000000] Early memory node ranges
[    0.000000]   node   0: [mem 0x0000000040200000-0x00000000ffffffff]
[    0.000000] Initmem setup node 0 [mem 0x0000000040200000-0x00000000ffffffff]
...
[    0.000000] Virtual kernel memory layout:
[    0.000000]       fixmap : 0xff1bfffffee00000 - 0xff1bffffff000000   (2048 kB)
[    0.000000]       pci io : 0xff1bffffff000000 - 0xff1c000000000000   (  16 MB)
[    0.000000]      vmemmap : 0xff1c000000000000 - 0xff20000000000000   (1024 TB)
[    0.000000]      vmalloc : 0xff20000000000000 - 0xff60000000000000   (16384 TB)
[    0.000000]       lowmem : 0xff60000000000000 - 0xff6000003fe00000   (1022 MB)
[    0.000000]       kernel : 0xffffffff80000000 - 0xffffffffffffffff   (2047 MB)
[    0.000000] Memory: 1010044K/1046528K available (7985K kernel code, 4921K rwdata, 4096K rodata, 2171K in
...
[    0.777981] EXT4-fs (vda): warning: mounting unchecked fs, running e2fsck is recommended
[    0.785870] EXT4-fs (vda): re-mounted. Quota mode: disabled.
Starting syslogd: OK
Starting klogd: OK
Running sysctl: OK
Initializing random number generator: OK
Saving random seed: [    5.521212] random: crng init done
Starting network: udhcpc: started, v1.35.0
udhcpc: broadcasting discover
udhcpc: broadcasting select for 10.0.2.15, server 10.0.2.2
udhcpc: lease of 10.0.2.15 obtained from 10.0.2.2, lease time 86400
deleting routers
adding dns 10.0.2.3

Welcome to Buildroot
buildroot login: root
# uname -a
Linux buildroot 5.19.0+ #23 SMP Sun Aug 7 19:58:52 EDT 2022 riscv64 GNU/Linux
# ls /lib/
ld-linux-riscv32-ilp32d.so.1  libgcc_s.so.1
libanl.so.1                   libm.so.6
libatomic.so                  libnss_dns.so.2
libatomic.so.1                libnss_files.so.2
libatomic.so.1.2.0            libpthread.so.0
libc.so.6                     libresolv.so.2
libcrypt.so.1                 librt.so.1
libdl.so.2                    libutil.so.1
libgcc_s.so                   modules
# cat /proc/1/maps
55554000-55630000 r-xp 00000000 fe:00 17                                 /bin/busybox
55631000-55633000 r--p 000dc000 fe:00 17                                 /bin/busybox
55633000-55634000 rw-p 000de000 fe:00 17                                 /bin/busybox
55634000-55655000 rw-p 00000000 00:00 0                                  [heap]
77e8e000-77fbe000 r-xp 00000000 fe:00 137                                /lib/libc.so.6
77fbe000-77fc0000 r--p 0012f000 fe:00 137                                /lib/libc.so.6
77fc0000-77fc1000 rw-p 00131000 fe:00 137                                /lib/libc.so.6
77fc1000-77fcb000 rw-p 00000000 00:00 0
77fcb000-77fd3000 r-xp 00000000 fe:00 146                                /lib/libresolv.so.2
77fd3000-77fd4000 ---p 00008000 fe:00 146                                /lib/libresolv.so.2
77fd4000-77fd5000 r--p 00008000 fe:00 146                                /lib/libresolv.so.2
77fd5000-77fd6000 rw-p 00009000 fe:00 146                                /lib/libresolv.so.2
77fd6000-77fd8000 rw-p 00000000 00:00 0
77fd8000-77fda000 r--p 00000000 00:00 0                                  [vvar]
77fda000-77fdb000 r-xp 00000000 00:00 0                                  [vdso]
77fdb000-77ffc000 r-xp 00000000 fe:00 132                                /lib/ld-linux-riscv32-ilp32d.so.1
77ffc000-77ffe000 r--p 00020000 fe:00 132                                /lib/ld-linux-riscv32-ilp32d.so.1
77ffe000-77fff000 rw-p 00022000 fe:00 132                                /lib/ld-linux-riscv32-ilp32d.so.1
7ffde000-7ffff000 rw-p 00000000 00:00 0                                  [stack]
```

阿里云 | 平头哥

Contents
目录

Dynamic XLEN

01

Specification Rules

- When XLEN < the widest supported XLEN

- Ignore source operand register bits above the configured XLEN.
- Sign-extend results to fill the entire widest supported XLEN in the destination register.
- PC bits above XLEN are ignored, and when the pc is written, it is sign-extended to fill the widest supported XLEN.

- CSR width modulation

- Zero extension or truncate
- Specially specified

Specification not clarified

- Not seamless

It's important to understand that the defaults were not written to facilitate actively changing the effective XLEN for software "on the go". Rather, the expectation is that whatever software was running before is likely "done", and something else will now be run, without much carryover in the CSRs between the two, if indeed any(From John Hauser)

- MXLEN >= SXLEN >= UXLEN

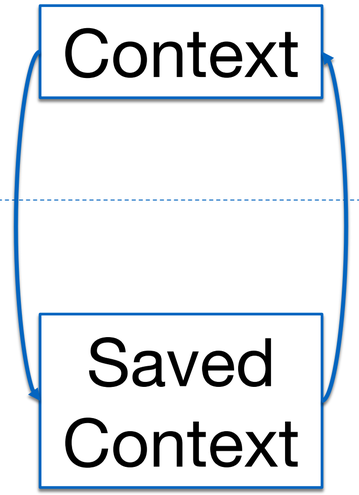
UXLEN support on QEMU

02

Ensure Context Switch Right

U-Mode(32bit)

S-Mode(64bit)



- 32-bit Regs
- PC
- 2^{XLEN} Address Space
- 32-bit CSR

- Sign extend Regs
- Sign extend PC
- 2^{XLEN} Address Space
- 64-bit CSR

Implementation

CSR Modulation

- User mode CSR
- Split XLEN related fields

32-bit Instruction

- Use 64 bit TCG op to emulate 32-bit instruction
- Sign extend or Zero extend the source and always sign extend the destination
- Define OLEN to process the *.w

PC Calculation

- Always sign extend for branch.
- When exception, always sign extend the PC.
- When restore PC in runtime, always sign extend the PC.
- When fetch PC and decoding , always mask the high bits

Address Calculation

- Module 2^{XLEN}
- Always use before pointer masking

Upstream Process

2021-08-09

UXLEN RFC Patch

[RFC PATCH 00/13] Support UXL field in mstatus

2021-08-30

Frédéric Pétrot RV128

[PATCH 0/8] RISC V partial support for 128-bit architecture

2021-10-07

Richard Henderson MXL

[RFC PATCH 00/13] target/riscv: Rationalize XLEN and operand length

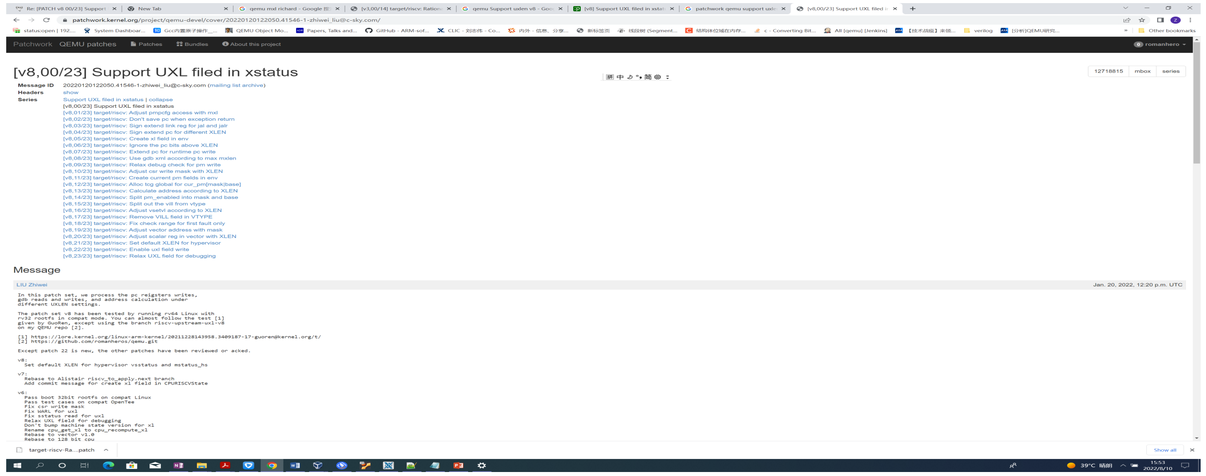
2022-1-21

V8 UXLEN Alistair merged

[PATCH v8 00/23] Support UXL filed in xstatus

Status

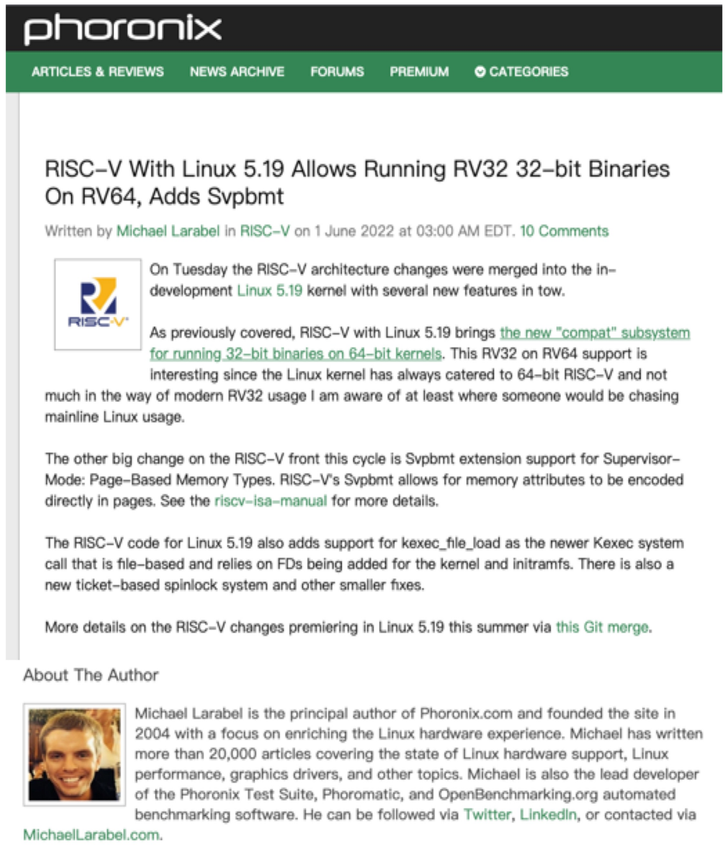
- The v8 UXLEN support patch set(23 patch) has been merged into QEMU v7.0.0
- Both Richard MXL and Frédéric Pétrot RV128 have been merged.
- Dynamic UXLEN support is ready.
- Neither dynamic SXLEN nor MXLEN is supported. Once both supported, we can remove RV32 system mode QEMU.



RV64 LINUX COMPAT for 32-bit U-mode

03

Linux-5.19 RISC-V statics



phoronix

ARTICLES & REVIEWS NEWS ARCHIVE FORUMS PREMIUM CATEGORIES

RISC-V With Linux 5.19 Allows Running RV32 32-bit Binaries On RV64, Adds Svpbmt

Written by Michael Larabel in RISC-V on 1 June 2022 at 03:00 AM EDT. 10 Comments

On Tuesday the RISC-V architecture changes were merged into the in-development Linux 5.19 kernel with several new features in tow.

As previously covered, RISC-V with Linux 5.19 brings the new "compat" subsystem for running 32-bit binaries on 64-bit kernels. This RV32 on RV64 support is interesting since the Linux kernel has always catered to 64-bit RISC-V and not much in the way of modern RV32 usage I am aware of at least where someone would be chasing mainline Linux usage.

The other big change on the RISC-V front this cycle is Svpbmt extension support for Supervisor-Mode: Page-Based Memory Types. RISC-V's Svpbmt allows for memory attributes to be encoded directly in pages. See the riscv-isa-manual for more details.

The RISC-V code for Linux 5.19 also adds support for kexec_file_load as the newer Kexec system call that is file-based and relies on FDs being added for the kernel and initramfs. There is also a new ticket-based spinlock system and other smaller fixes.

More details on the RISC-V changes premiering in Linux 5.19 this summer via this Git merge.

About The Author

Michael Larabel is the principal author of Phoronix.com and founded the site in 2004 with a focus on enriching the Linux hardware experience. Michael has written more than 20,000 articles covering the state of Linux hardware support, Linux performance, graphics drivers, and other topics. Michael is also the lead developer of the Phoronix Test Suite, Phoromatic, and OpenBenchmarking.org automated benchmarking software. He can be followed via Twitter, LinkedIn, or contacted via MichaelLarabel.com.

Linux-5.19 two important features:
COMPAT and Svpbmt

https://kernelnewbies.org/Linux_5.19#RISCV
https://www.cnx-software.com/2022/08/01/linux-5-19-release-main-changes-arm-risc-v-and-mips-architectures/
https://9to5linux.com/linux-kernel-5-19-officially-released-this-is-whats-new
https://tinylab.org/tinylab-weekly-6-1st-2022/

```
Guo Ren (21) // T-HEAD
Heiko Stuebner (16) // VRULL & T-HEAD
Palmer Dabbelt (14) // Rivos
Conor Dooley (11) // Microchip
Anup Patel (10) // ventana
Li Zhengyu (5) // Huawei
Christoph Hellwig (3) // ???
Alexandre Ghiti (3) // bootlin
Liao Chang (3) // Huawei
Nathan Chancellor (2) // ???
Zong Li (2) // Sifive
Xianting Tian (1)
Chuanhua Han (1)
Niklas Cassel (1)
Uwe Kleine-KÃ¶nig (1)
Jisheng Zhang (1)
Masahiro Yamada (1)
Samuel Holland (1)
Tobias Klauser (1)
Ben Dooks (1)
Krzysztof Kozlowski (1)
Emil Renner Berthing (1)
Julia Lawall (1)
Lukas Bulwahn (1)
Alexandre Ghiti (1)
Atish Patra (1)
Jiapeng Chong (1)
```

27 contributors from various organizations contributed 106 RISC-V patches. 43 of them are directly related to the XuanTie CPU. In addition, Huawei (6) and T-HEAD (1) have improved the Kdump/Kexec feature to prepare for the crash utils upstream.

```
Heiko Stuebner (16):
      riscv: integrate alternatives b
      riscv: allow different stages w
      riscv: implement module alterna
      riscv: implement ALTERNATIVE_2
      riscv: extend concatenated alte
      riscv: prevent compressed instr
      riscv: move boot alternatives t
      riscv: Fix accessing pfn bits i
      riscv: add RISC-V Svpbmt extens
      riscv: remove FIXMAP_PAGE_IO ar
      riscv: don't use global static
      riscv: add memory-type errata f
      riscv: fix dependency for t-hea
      riscv: drop cpufeature_apply_fe
      riscv: Improve description for
      riscv: don't warn for sifive er

Nathan Chancellor (1):
      riscv: Fix ALT_THEAD_PMA's asm
```

The Svpbmt feature contains 17 patches from two contributors. It makes D1's XuanTie CPU could run upstream kernel.

```
Christoph Hellwig (3):
      uapi: simplify __ARCH_FLOCK{,64}_PAD a littl
      uapi: always define F_GETLK64/F_SETLK64/F_SE
      compat: consolidate the compat_flock{,64} de

Guo Ren (21):
      arch: Add SYSVIPC_COMPAT for all architectur
      fs: stat: compat: Add __ARCH_WANT_COMPAT_STA
      asm-generic: compat: Cleanup duplicate defin
      syscalls: compat: Fix the missing part for _
      riscv: Fixup difference with defconfig
      riscv: compat: Add basic compat data type im
      riscv: compat: Support TASK_SIZE for compat
      riscv: compat: syscall: Add compat_sys_call_
      riscv: compat: syscall: Add entry.S implemen
      riscv: compat: process: Add UXL_32 support i
      riscv: compat: Add elf.h implementation
      riscv: compat: Add hw capability check for e
      riscv: compat: vdso: Add COMPAT_VDSO base co
      riscv: compat: vdso: Add setup additional pa
      riscv: compat: signal: Add rt_frame implemen
      riscv: compat: ptrace: Add compat_arch_ptrac
      riscv: compat: Add COMPAT Kbuild skeletal su
      riscv: atomic: Cleanup unnecessary definitio
      riscv: atomic: Optimize dec_if_positive func
      riscv: atomic: Add custom conditional atomic
      riscv: compat: Using seperated vdso_maps for

Palmer Dabbelt (1):
      RISC-V: Add support for rv32 userspace via C

Emil Renner Berthing (1):
      riscv: compat: vdso: Fix vdso_install target
```

The COMPAT feature contains 26 patches from 4 contributors.

COMPAT Introduction (V1~V9, cross two dev cycles)

Currently, in the Linux 64-bit architecture, x86, parisc, powerpc, arm64, s390, mips, and sparc all support COMPAT, and RISC-V has become the 8th architecture to support this feature. Different from its predecessors, RISC-V is the first to implement COMPAT based on asm-generic/unistd.h, and it is also the most standard. Here is work list:

- Cleanup syscall common code with community maintainers (Thx Arnd, Christoph)
- Create the base compat data structure
- Implement double TASK_SIZE
- Implement compat_sys_call_table
- Implement compat_syscall exception handler in entry.S
- Implement double ELF detect
- Implement compat_vdso
- Implement compat_signal
- Implement compat_ptrace

Agenda

- HW cost
- RV32 Linux kernel needs contributions
- eBPF JIT problem
- ELF size comparison
- Why not RV64 ILP32 ABI for Linux?

HW cost

U32 (Cheap)

- Zero extension or truncate
- Sign-extend BIT[31] to 64-bit GPR (x1~x31)
- Move between GPR and FPR/VPR

S32+M32 (Expensive)

- Sv32
- CSR xlen 32/64
- CSR h/l for 32-bit
- PMP, timer, CLIT/PLIT, PMU …

So, COMPAT is little in HW cost for an existed 64-bit core.

RV32 Linux kernel needs contributions

ch/riscv/Kconfig:

```
ect HAVE_GENERIC_VDSO if MMU && 64BIT
ect GENERIC_TIME_VSYSCALL if MMU && 64BIT
ect GENERIC_ATOMIC64 if !64BIT
ect HAVE_ARCH_KASAN if MMU && 64BIT
ect HAVE_ARCH_KASAN_VMALLOC if MMU && 64BIT
ect HAVE_ARCH_KFENCE if MMU && 64BIT
ect HAVE_ARCH_VMAP_STACK if MMU && 64BIT
ect HAVE_ARCH_TRANSPARENT_HUGEPAGE if 64BIT && MMU
ect ARCH_WANT_HUGE_PMD_SHARE if 64BIT
ect GENERIC_BUG_RELATIVE_POINTERS if 64BIT

nfig ARCH_RV32I
   bool "RV32I"
   depends on NONPORTABLE
   select 32BIT
   select GENERIC_LIB_ASHLDI3
   select GENERIC_LIB_ASHRDI3
   select GENERIC_LIB_LSHRDI3
   select GENERIC_LIB_UCMPDI2
   select MMU

nfig ARCH_RV64I
   bool "RV64I"
   select 64BIT
   select ARCH_SUPPORTS_INT128 if CC_HAS_INT128
   select HAVE_DYNAMIC_FTRACE
   select HAVE_DYNAMIC_FTRACE_WITH_REGS
   select HAVE_FTRACE_MCOUNT_RECORD if !XIP_KERNEL
   select HAVE_FUNCTION_GRAPH_TRACER
   select HAVE_FUNCTION_TRACER if !XIP_KERNEL
   select SWIOTLB if MMU

nfig RISCV_ISA_SVPBMT // 5.19
   bool "SVPBMT extension support"
   depends on 64BIT && MMU

nfig KEXEC_FILE // 5.19
   bool "kexec file based systmem call"
   ...
```

JavaScript RISC-V Instruction Set Architecture ...

RISC-V Mentorship: Porting V8 to RISC-V R32G

Terms

Spring 2022

The RISC-V Mentorship Program enables one or more 12-week internship-style projects per session, funded by RISC-V, to match mentors/project leaders together with mentees/interns . Mentees are guided through a series of milestones by one or more project mentors, with whom the mentees meet on a weekly basis.

This program pairs one mentee with an experienced mentor to deliver a V8 JavaScript engine port for a 32-bit RISC-V core.

The V8 JavaScript engine for RISCV64G has been upstreamed to Chromium recently. As a basic component for the Chromium web browser and node.js, it would enlarge RISCV's application scenario. Although RV32G V8 port would be quite similar to RV64G V8 port , it is still in the TODO list. Porting and enable the RV32G on V8 will bring the embedded RISCV software ecosystem more applications, make RISC-V embed processors more competitive.

Congrats! The Google V8 JavaScript Engine now supports RV32GC.

Wei Wu (吴伟)

Hi, everyone

We are glad to share that the RV32GC porting of V8 (the JavaScript engine of the Chrome browser and Node.js) is merged by Google V8 upstream today[1].

The porting has been carried out in a cooperative way as a mentorship project of RVI and the Linux Foundation[2]. Thanks to the communication channel provided by the mentorship project (special thanks to Megan!), people who are willing to contribute to RISC-V could join together more effectively and easily.

Now V8 can support both 32bit and 64bit backends for RISC-V ISA. The team from PLCT will maintain it for the long term. Reporting bugs at GitHub repo site[3] is welcome. We hope this would be a stepping stone to enable more applications for the 32bit RISC-V world.

[1] https://chromium-review.googlesource.com/c/v8/v8/+/3736732
[2] https://mentorship.lfx.linuxfoundation.org/project/2021e650-c533-4671-afed-bf87c089af09
[3] https://github.com/riscv-collab/v8

--
Best wishes,
Wei Wu (吴伟)
PLCT Lab, ISCAS

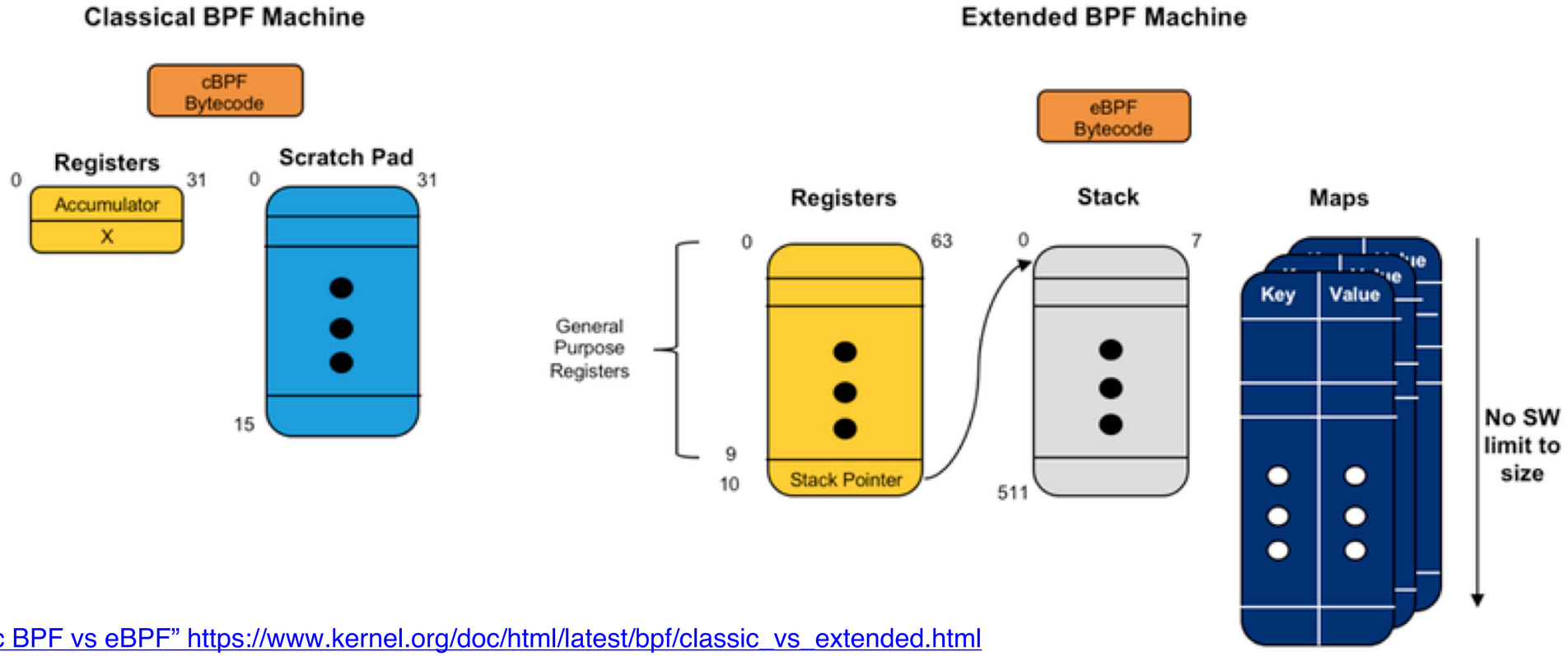
8月25日周四・下午・分会场B

16:30	**V8 RV32GC 移植项目进展介绍**	**邱吉** 中科院软件所
17:00	**OpenJDK的RV32G支持**	**史宁宁** 中科院软件所PLCT实验室

eBPF JIT problem

rv32-Linux with cBPF
rv64-Linux with eBPF

- Number of registers increase from 2 to 10
- Register width increases from 32-bit to 64-bit
- Conditional jt/jf targets replaced with jt/fall-through
- Introduces bpf_call insn and register passing convention for zero overhead calls from/to other kernel functions



"Classic BPF vs eBPF" https://www.kernel.org/doc/html/latest/bpf/classic_vs_extended.html

ELF size comparison

64kernel + 32rootfs

```
[    0.000000] Linux version 5.16.0-rc6-00017-g750f87086bdd-dirty (guoren@guoren-Z87-
HD3) (riscv64-unknown-linux-gnu-gcc (GCC) 10.2.0, GNU ld (GNU Binutils) 2.37) #96 SMP
Tue Dec 28 21:01:55 CST 2021
[    0.000000] OF: fdt: Ignoring memory range 0x80000000 - 0x80200000
[    0.000000] Machine model: riscv-virtio,qemu
[    0.000000] earlycon: sbi0 at I/O port 0x0 (options '')
[    0.000000] printk: bootconsole [sbi0] enabled
[    0.000000] efi: UEFI not found.
[    0.000000] Zone ranges:
[    0.000000]   DMA32    [mem 0x0000000080200000-0x00000000ffffffff]
[    0.000000]   Normal   empty
[    0.000000] Movable zone start for each node
[    0.000000] Early memory node ranges
[    0.000000]   node   0: [mem 0x0000000080200000-0x00000000ffffffff]
[    0.000000] Initmem setup node 0 [mem 0x0000000080200000-0x00000000ffffffff]
[    0.000000] SBI specification v0.2 detected
[    0.000000] SBI implementation ID=0x1 Version=0x9
[    0.000000] SBI TIME extension detected
[    0.000000] SBI IPI extension detected
[    0.000000] SBI RFENCE extension detected
[    0.000000] SBI v0.2 HSM extension detected
[    0.000000] riscv: ISA extensions acdfhimsu
[    0.000000] riscv: ELF capabilities acdfim
[    0.000000] percpu: Embedded 17 pages/cpu s30696 r8192 d30744 u69632
[    0.000000] Built 1 zonelists, mobility grouping on.  Total pages: 15655
[    0.000000] Kernel command line: rootwait root=/dev/vda ro console=ttyS0 earlycon=sbi
[    0.000000] Dentry cache hash table entries: 8192 (order: 4, 65536 bytes, linear)
[    0.000000] Inode-cache hash table entries: 4096 (order: 3, 32768 bytes, linear)
[    0.000000] mem auto-init: stack:off, heap alloc:off, heap free:off
[    0.000000] Virtual kernel memory layout:
[    0.000000]       fixmap : 0xffffffcefee00000 - 0xffffffceff000000   (2048 kB)
[    0.000000]       pci io : 0xffffffceff000000 - 0xffffffcf00000000   (  16 MB)
[    0.000000]      vmemmap : 0xffffffcf00000000 - 0xffffffcfffffffff   (4095 MB)
[    0.000000]      vmalloc : 0xffffffd000000000 - 0xffffffdfffffffff   (65535 MB)
[    0.000000]       lowmem : 0xffffffe000000000 - 0xffffffe003e00000   (  62 MB)
[    0.000000]       kernel : 0xffffffff80000000 - 0xffffffffffffffff   (2047 MB)
[    0.000000] Memory: 52788K/63488K available (6184K kernel code, 888K rwdata, 1917K
rodata, 294K init, 297K bss, 10700K reserved, 0K cma-reserved)
[    0.000000] SLUB: HWalign=64, Order=0-3, MinObjects=0, CPUs=1, Nodes=1
[    0.000000] rcu: Hierarchical RCU implementation.
[    0.000000] rcu:     RCU restricting CPUs from NR_CPUS=8 to nr_cpu_ids=1.
[    0.000000] rcu:     RCU debug extended QS entry/exit.
[    0.000000]  Tracing variant of Tasks RCU enabled.
[    0.000000] rcu: RCU calculated value of scheduler-enlistment delay is 25 jiffies.
[    0.000000] rcu: Adjusting geometry for rcu_fanout_leaf=16, nr_cpu_ids=1
[    0.000000] NR_IRQS: 64, nr_irqs: 64, preallocated irqs: 0
[    0.000000] riscv-intc: 64 local interrupts mapped
[    0.000000] plic: plic@c000000: mapped 53 interrupts with 1 handlers for 2 contexts.
...
Welcome to Buildroot
```

32kernel + 32rootfs

```
[    0.000000] Linux version 5.16.0-rc6-00017-g750f87086bdd-dirty (guoren@guoren-Z87-
HD3) (riscv32-buildroot-linux-gnu-gcc.br_real (Buildroot 2021.11-201-g7600ca7960-dirty)
10.3.0, GNU ld (GNU Binutils) 2.36.1) #7 SMP Tue Dec 28 21:02:21 CST 2021
[    0.000000] OF: fdt: Ignoring memory range 0x80000000 - 0x80400000
[    0.000000] Machine model: riscv-virtio,qemu
[    0.000000] earlycon: sbi0 at I/O port 0x0 (options '')
[    0.000000] printk: bootconsole [sbi0] enabled
[    0.000000] efi: UEFI not found.
[    0.000000] Zone ranges:
[    0.000000]   Normal   [mem 0x0000000080400000-0x00000000ffffffff]
[    0.000000] Movable zone start for each node
[    0.000000] Early memory node ranges
[    0.000000]   node   0: [mem 0x0000000080400000-0x00000000ffffffff]
[    0.000000] Initmem setup node 0 [mem 0x0000000080400000-0x00000000ffffffff]
[    0.000000] SBI specification v0.2 detected
[    0.000000] SBI implementation ID=0x1 Version=0x9
[    0.000000] SBI TIME extension detected
[    0.000000] SBI IPI extension detected
[    0.000000] SBI RFENCE extension detected
[    0.000000] SBI v0.2 HSM extension detected
[    0.000000] riscv: ISA extensions acdfhimsu
[    0.000000] riscv: ELF capabilities acdfim
[    0.000000] percpu: Embedded 12 pages/cpu s16600 r8192 d24360 u49152
[    0.000000] Built 1 zonelists, mobility grouping on.  Total pages: 15240
[    0.000000] Kernel command line: rootwait root=/dev/vda ro console=ttyS0 earlycon=sbi
[    0.000000] Dentry cache hash table entries: 8192 (order: 3, 32768 bytes, linear)
[    0.000000] Inode-cache hash table entries: 4096 (order: 2, 16384 bytes, linear)
[    0.000000] mem auto-init: stack:off, heap alloc:off, heap free:off
[    0.000000] Virtual kernel memory layout:
[    0.000000]       fixmap : 0x9dc00000 - 0x9e000000   (4096 kB)
[    0.000000]       pci io : 0x9e000000 - 0x9f000000   (  16 MB)
[    0.000000]      vmemmap : 0x9f000000 - 0x9fffffff   (  15 MB)
[    0.000000]      vmalloc : 0xa0000000 - 0xbfffffff   ( 511 MB)
[    0.000000]       lowmem : 0xc0000000 - 0xc3c00000   (  60 MB)
[    0.000000] Memory: 51924K/61440K available (6117K kernel code, 695K rwdata, 1594K
rodata, 255K init, 241K bss, 9516K reserved, 0K cma-reserved)
[    0.000000] SLUB: HWalign=64, Order=0-3, MinObjects=0, CPUs=1, Nodes=1
[    0.000000] rcu: Hierarchical RCU implementation.
[    0.000000] rcu:     RCU restricting CPUs from NR_CPUS=8 to nr_cpu_ids=1.
[    0.000000] rcu:     RCU debug extended QS entry/exit.
[    0.000000]  Tracing variant of Tasks RCU enabled.
[    0.000000] rcu: RCU calculated value of scheduler-enlistment delay is 25 jiffies.
[    0.000000] rcu: Adjusting geometry for rcu_fanout_leaf=16, nr_cpu_ids=1
[    0.000000] NR_IRQS: 64, nr_irqs: 64, preallocated irqs: 0
[    0.000000] riscv-intc: 32 local interrupts mapped
[    0.000000] plic: plic@c000000: mapped 53 interrupts with 1 handlers for 2 contexts.
...
Welcome to Buildroot
```

64: Memory: 52788K/63488K available (6184K kernel code, 888K rwdata, 1917K rodata, 294K init, 297K bss, 10700K reserved)
32: Memory: 51924K/61440K available (6117K kernel code, 695K rwdata, 1594K rodata, 255K init, 241K bss, 9516K reserved)
result: 1% 22% 17% 13% 19% 11%
rv32 kernel runtime KernelStack, Slab... are smaller
rv64: MemTotal: 53076 kB, MemFree: 40264 kB
rv32: MemTotal: 52176 + 2048 kB, MemFree: 41012 + 2048 kB
result : 2% 6%

Why not RV64 ILP32 ABI for Linux?

Question answered by software guys.

```
Subject: Re: [PATCH 00/13] riscv: compat: Add COMPAT mode support for rv64
Date: Mon, 27 Dec 2021 02:29:02 +0000   [thread overview]
Message-ID: <AA7091EA-C3AF-47CE-B0C5-E2ECE4133D09@jrtc27.com> (raw)
In-Reply-To: <CAJF2gTQ04utyOc9=q9Tu92togGuyygKgq3tWwfPBCuyTfLh2SQ@mail.gmail.com>

On 27 Dec 2021, at 01:16, Guo Ren <guoren@kernel.org> wrote:
>
> On Mon, Dec 27, 2021 at 4:31 AM Arnd Bergmann <arnd@arndb.de> wrote:
>>
>> On Sun, Dec 26, 2021 at 7:38 AM Guo Ren <guoren@kernel.org> wrote:
>>> On Sun, Dec 26, 2021 at 4:36 PM Jisheng Zhang <jszhang3@mail.ustc.edu.cn> wrote:
>>>> On Wed, 22 Dec 2021 20:59:30 +0800 Guo Ren <guoren@kernel.org> wrote:
>>>>> On Wed, Dec 22, 2021 at 2:10 AM Arnd Bergmann <arnd@arndb.de> wrote:
>>>>>
>>>>> What about adding RV64 ILP32 support instead? This don't need HW side
>>>>> modifications so can benefit all RV64.
>>>
>>> ILP32 is another topic in C Language Data Type Models and it couldn't
>>> replace the standard rv32 ecosystem.
>>> COMPAT is a common framework in Linux (7 arches have been supported),
>>> so let rv64 support COMPAT mode is considerable.
>>>
>>> Customers would choose ILP32 / RV32-compat by themself and that
>>> depends on which one has a better ecosystem.
>>
>> From a kernel perspective, supporting both is not much more work than
>> supporting either of them. We had the same debate for Arm64, and ended
>> up never merging the ILP32 patches despite them being well written
>> and maintainable, to limit the number of supported user space ABIs
>> as well as the possible attack vectors when there is an exploitable
>> bug that is specific to an ABI.
>>
>> arm64 does support big-endian mode, which is a similar niche, but it
>> can't easily be removed after it's already supported. Supporting normal
>> compat mode is the easiest here because it doesn't add another user
>> space ABI, but I'd strongly recommend not to add any other ones.
>
> @Palmer Dabbelt How do you think about supporting ILP32 & COMPAT both
> in rv64? And let users vote by foot which is better.

As psABI TG co-chair I really do not want an ILP32 RV64 to exist if it
can at all be avoided. Every single attempt at an ILP32 ABI for a
64-bit architecture has failed to take off in the past, so I struggle
to see why RV64 will be any different. So, in my opinion, there is a
relatively high barrier to entry for it to be an official frozen ABI,
and without it being that I doubt upstreams will want to go near it, be
it Linux, GCC, binutils or GCC, but they can speak for themselves if
they feel otherwise.

Also, with every year that goes by, ILP32 becomes more and more limited
in what you can use it for, due to increased memory footprints...

Jess
```

Q: Why not RV64 ILP32? How about apple watch's aarch64-ILP32?

ILP32 for ARM64

From: Yury Norov <ynorov-AT-caviumnetworks.com>
To: <linux-arm-kernel-AT-lists.infradead.org>, <linux-kernel-AT-vger.kernel.org>, <linux-doc-AT-vger
Arnd Bergmann" <arnd-AT-arndb.de>, Catalin Marinas <catalin.marinas-AT-arm.com>
Subject: [PATCH v7 00/20] ILP32 for ARM64
Date: Mon, 9 Jan 2017 16:59:37 +0530
Message-ID: <1483961397-8599-1-git-send-email-ynorov@caviumnetworks.com>
Cc: Yury Norov <ynorov-AT-caviumnetworks.com>, Andrew Pinski <pinskia-AT-gmail.com>, Andrew
caviumnetworks.com>, Adam Borowski <kilobyte-AT-angband.pl>, Chris Metcalf <cmetcalf-AT-ez
caviumnetworks.com>, Maxim Kuvyrkov <maxim.kuvyrkov-AT-linaro.org>, Ramana Radhakrishna
Weimer <fweimer-AT-redhat.com>, Bamvor Zhangjian <bamvor.zhangjian-AT-huawei.com>, And
Metcalf <cmetcalf-AT-mellanox.com>, Heiko Carstens <heiko.carstens-AT-de.ibm.com>, <schwid
kernel.org>, Joseph Myers <joseph-AT-codesourcery.com>, <christoph.muellner-AT-theobroma-s
<klimov.linux-AT-gmail.com>, <Nathan_Lynch-AT-mentor.com>, <agraf-AT-suse.de>, <Prasun.Ka
linux-m68k.org>, <philipp.tomsich-AT-theobroma-systems.com>, <manuel.montezelo-AT-gmail.
<davem-AT-davemloft.net>, <zhouchengming1-AT-huawei.com>

This series enables aarch64 with ilp32 mode.

As supporting work, it introduces ARCH_32BIT_OFF_T configuration
option that is enabled for existing 32-bit architectures but disabled
for new arches (so 64-bit off_t is is used by new userspace). Also it
deprecates getrlimit and setrlimit syscalls prior to prlimit64.

This version is based on linux-next from 2017-01-09. It works with
glibc-2.25, and tested with LTP, glibc testsuite, trinity, lmbench,
CPUSpec.

This is not RFC anymore. I believe that all ABI and implementation
issues are resolved now. The way that kernel clears registers top
halves is probably the last question, and because there's no objection
for current approach for more that 6 month, I think, community agrees
with it.

Patches 1, 2, 3 and 8 are general, and may be applied separately.

Current version does not introduce ABI changes comparing to RFC3.
Kernel and GLIBC trees:
https://github.com/norov/linux/tree/ilp32-2017-01-09
https://github.com/norov/glibc/tree/dev9

(GLIBC patches are managed by Steve Ellsey, so my tree is only for
reference.)

Changes:
v3: https://lkml.org/lkml/2014/9/3/704
v4: https://lkml.org/lkml/2015/4/13/691
v5: https://lkml.org/lkml/2015/9/29/911
v6: https://lkml.org/lkml/2016/5/23/661
v7: RFC nowrap: https://lkml.org/lkml/2016/6/17/990
v7: RFC2 nowrap: https://lkml.org/lkml/2016/8/17/245
v7: RFC3 nowrap: https://lkml.org/lkml/2016/10/21/883
v7: - 32-bit off_t deprecation is splitted for compat
and native 32-bit arches, as it was initially
done (patches 1, 2);
- getrlimit() and setrlimit() syscalls deprecated for
aarch64/ilp32 and all new architectures;
- documentation is cleaned up (patch 4);
- compat-related definitions moved from
aarch64/include/elf.h to binfmt_elf32.c (patch 11)
- for ptrace, execution mode detection is performed
at runtime, as it was in v4 (patch 18)

```
Andrew Pinski (6):
  arm64: rename COMPAT to AARCH32_EL0 in Kconfig
  arm64: ensure the kernel is compiled for LP64
  arm64:uapi: set __BITS_PER_LONG correctly for ILP32 and LP64
  arm64: ilp32: add sys_ilp32.c and a separate table (in entry.S) to use
    it
  arm64: ilp32: introduce ilp32-specific handlers for sigframe and
    ucontext
  arm64:ilp32: add ARM64_ILP32 to Kconfig

Philipp Tomsich (1):
  arm64:ilp32: add vdso-ilp32 and use for signal return

Yury Norov (13):
  compat ABI: use non-compat openat and open_by_handle_at variants
  32-bit ABI: introduce ARCH_32BIT_OFF_T config option
  asm-generic: Drop getrlimit and setrlimit syscalls from default list
  arm64: ilp32: add documentation on the ILP32 ABI for ARM64
  thread: move thread bits accessors to separated file
  arm64: introduce is_a32_task and is_a32_thread (for AArch32 compat)
  arm64: ilp32: add is_ilp32_compat_{task,thread} and TIF_32BIT_AARCH64
  arm64: introduce binfmt_elf32.c
  arm64: ilp32: introduce binfmt_ilp32.c
  arm64: ilp32: share aarch32 syscall handlers
  arm64: signal: share lp64 signal routines to ilp32
  arm64: signal32: move ilp32 and aarch32 common code to separated file
  arm64: ptrace: handle ptrace_request differently for aarch32 and ilp32

 Documentation/arm64/ilp32.txt                      |  45 +++++++
 arch/Kconfig                                       |   4 +
 arch/arc/Kconfig                                   |   1 +
 arch/arc/include/uapi/asm/unistd.h                 |   1 +
 arch/arm/Kconfig                                   |   1 +
 arch/arm64/Kconfig                                 |  19 ++-
 arch/arm64/Makefile                                |   5 +
 arch/arm64/include/asm/compat.h                    |  19 +--
 arch/arm64/include/asm/elf.h                       |  32 ++---
 arch/arm64/include/asm/fpsimd.h                    |   2 +-
 arch/arm64/include/asm/ftrace.h                    |   2 +-
 arch/arm64/include/asm/hwcap.h                     |   6 +-
 arch/arm64/include/asm/is_compat.h                 |  90 ++++++++++++++
 arch/arm64/include/asm/memory.h                    |   5 +-
 arch/arm64/include/asm/processor.h                 |  11 +-
 arch/arm64/include/asm/ptrace.h                    |   2 +-
 arch/arm64/include/asm/seccomp.h                   |   2 +-
 arch/arm64/include/asm/signal32.h                  |   9 +-
 arch/arm64/include/asm/signal32_common.h           |  27 ++++
 arch/arm64/include/asm/signal_common.h             |  33 +++++
 arch/arm64/include/asm/signal_ilp32.h              |  38 ++++++
 arch/arm64/include/asm/syscall.h                   |   2 +-
 arch/arm64/include/asm/thread_info.h               |   4 +-
 arch/arm64/include/asm/unistd.h                    |   8 +-
 arch/arm64/include/asm/vdso.h                      |   6 +
 arch/arm64/include/uapi/asm/bitsperlong.h          |   9 +-
 arch/arm64/include/uapi/asm/unistd.h               |  13 ++
 arch/arm64/kernel/Makefile                         |  18 +--
 arch/arm64/kernel/asm-offsets.c                    |   9 +-
 arch/arm64/kernel/binfmt_elf32.c                   |  32 +++++
 arch/arm64/kernel/binfmt_ilp32.c                   |  98 ++++++++++++++
 arch/arm64/kernel/cpufeature.c                     |   8 +-
 arch/arm64/kernel/cpuinfo.c                        |  20 ++--
 arch/arm64/kernel/entry.S                          |  34 +++++-
 arch/arm64/kernel/entry32.S                        |  80 ++++++++++++
 arch/arm64/kernel/entry32_common.S                 | 107 ++++++++++++++++
 arch/arm64/kernel/entry_ilp32.S                    |  22 ++++
 arch/arm64/kernel/head.S                           |   2 +-
 arch/arm64/kernel/hw_breakpoint.c                  |   8 +-
 arch/arm64/kernel/perf_regs.c                      |   2 +-
 arch/arm64/kernel/process.c                        |   7 +-
 arch/arm64/kernel/ptrace.c                         |  80 ++++++++++--
 arch/arm64/kernel/signal.c                         | 102 +++++++++++++++-
 arch/arm64/kernel/signal32.c                       | 107 ++++++++++++++++
 arch/arm64/kernel/signal32_common.c                | 135 +++++++++++++++++++++
 arch/arm64/kernel/signal_ilp32.c                   | 170 +++++++++++++++++++++++++
 arch/arm64/kernel/sys_ilp32.c                      | 100 +++++++++++++++
 arch/arm64/kernel/syscall.c                        |   2 +-
 arch/arm64/kernel/vdso-ilp32/.gitignore            |   2 +
 arch/arm64/kernel/vdso-ilp32/Makefile              |  74 +++++++++++
 arch/arm64/kernel/vdso-ilp32/vdso-ilp32.S          |  33 +++++
 arch/arm64/kernel/vdso-ilp32/vdso-ilp32.lds.S      |  95 +++++++++++++++
 arch/arm64/kernel/vdso.c                           |  70 ++++++++++--
```

https://lwn.net/Articles/711055/

Profiles' problem discussion

5.2.4. RVA20S64 Unsupported Optional Extensions

The following privileged extensions are unsupported:

- **Ssu32xl** `sstatus.UXL` =32

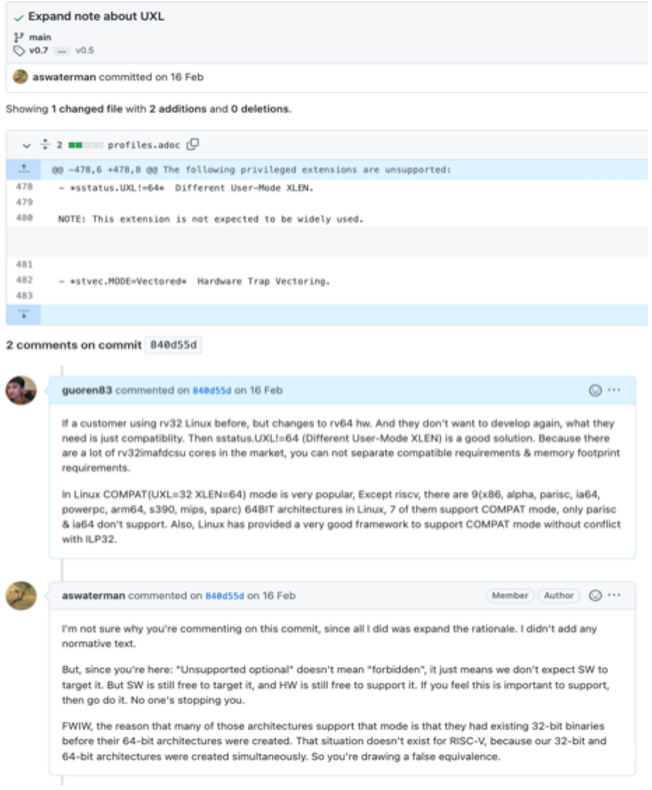
Note	This extension is not expected to be widely used, as RISC-V applications processors were initially, and are still predominantly, XLEN=64, avoiding the need to support legacy XLEN=32 binaries. Applications that want to use 32-bit pointers to reduce memory footprint should consider using a forthcoming RV64 ILP32 ABI instead.

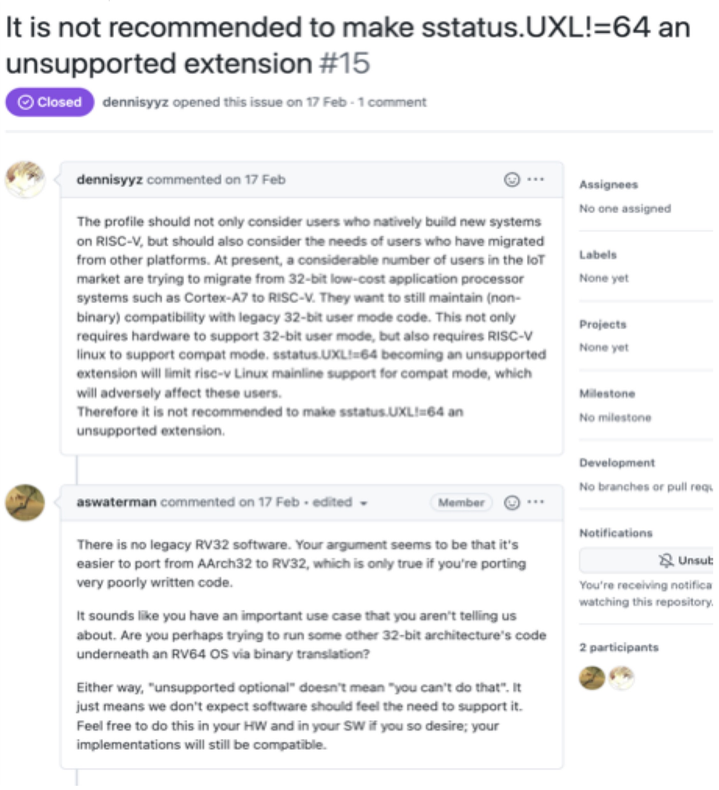
6.2.4. RVA22S64 Unsupported Optional Extensions

- **Ssu32xl** `sstatus.UXL` =32

Note	This extension is not expected to be widely used.

https://github.com/riscv/riscv-profiles/blob/main/profiles.adoc





Profiles' principle

Profiles are …

- A mechanism to identify a group of extensions (in
  behavior, state) that work together
- Like macros. Profiles consist of Extensions. Extens
  consist of a group of rules, state, and behaviors o
  extensions
- What we ask the upstream projects (gcc, binutils,
  and Linux distros to target. Major and minor releases
- The way we reduce fragmentation in the RISC-V community

Some suggestions at the end

- Focus on native ILP32 in Linux userspace
- Choose RV64 processors with COMPAT
- contribute rv32-Linux actively

谢 谢

THANK YOU

阿里云 | 平头哥