features.txt

FUJITSU FR-V LINUX FEATURES

This kernel port has a number of features of which the user should be aware:

(*) Linux and uClinux

The FR-V architecture port supports both normal MMU linux and uClinux out of the same sources.

(*) CPU support

Support for the FR401, FR403, FR405, FR451 and FR555 CPUs should work with the same uClinux kernel configuration.

In normal (MMU) Linux mode, only the FR451 CPU will work as that is the only one with a suitably featured CPU.

The kernel is written and compiled with the assumption that only the bottom 32 GR registers and no FR registers will be used by the kernel itself, however all extra userspace registers will be saved on context switch. Note that since most CPUs can't support lazy switching, no attempt is made to do lazy register saving where that would be possible (FR555 only currently).

(*) Board support

The board on which the kernel will run can be configured on the "Processor type and features" configuration tab.

Set the System to "MB93093-PDK" to boot from the MB93093 (FR403) PDK.

Set the System to "MB93091-VDK" to boot from the CB11, CB30, CB41, CB60, CB70 or CB451 VDK boards. Set the Motherboard setting to "MB93090-MB00" to boot with the standard ATA90590B VDK motherboard, and set it to "None" to boot without any motherboard.

(*) Binary Formats

The only userspace binary format supported is FDPIC ELF. Normal ELF, FLAT and AOUT binaries are not supported for this architecture.

FDPIC ELF supports shared library and program interpreter facilities.

(*) Scheduler Speed

The kernel scheduler runs at 100Hz irrespective of the clock speed on this architecture. This value is set in asm/param.h (see the HZ macro defined there).

(*) Normal (MMU) Linux Memory Layout.

See mmu-layout.txt in this directory for a description of the normal linux memory layout

See include/asm-frv/mem-layout.h for constants pertaining to the memory layout.

See include/asm-frv/mb-regs.h for the constants pertaining to the I/O bus controller configuration.

(*) uClinux Memory Layout

The memory layout used by the uClinux kernel is as follows:

```
0x00000000 - 0x00000FFF         Null pointer catch page
0x20000000 - 0x200FFFFF CS2#    [PDK] FPGA
0xC0000000 - 0xCFFFFFFF         SDRAM
0xC0000000                      Base of Linux kernel image
0xE0000000 - 0xEFFFFFFF CS2#    [VDK] SLBUS/PCI window
0xF0000000 - 0xF0FFFFFF CS5#    MB93493 CSC area (DAV daughter board)
0xF1000000 - 0xF1FFFFFF CS7#    [CB70/CB451] CPU-card PCMCIA port space
0xFC000000 - 0xFC0FFFFF CS1#    [VDK] MB86943 config space
0xFC100000 - 0xFC1FFFFF CS6#    [CB70/CB451] CPU-card DM9000 NIC space
0xFC100000 - 0xFC1FFFFF CS6#    [PDK] AX88796 NIC space
0xFC200000 - 0xFC2FFFFF CS3#    MB93493 CSR area (DAV daughter board)
0xFD000000 - 0xFDFFFFFF CS4#    [CB70/CB451] CPU-card extra flash space
0xFE000000 - 0xFEFFFFFF         Internal CPU peripherals
0xFF000000 - 0xFF1FFFFF CS0#    Flash 1
0xFF200000 - 0xFF3FFFFF CS0#    Flash 2
0xFFC00000 - 0xFFC0001F CS0#    [VDK] FPGA
```

The kernel reads the size of the SDRAM from the memory bus controller registers by default.

The kernel initialisation code (1) adjusts the SDRAM base addresses to move the SDRAM to desired address, (2) moves the kernel image down to the bottom of SDRAM, (3) adjusts the bus controller registers to move I/O windows, and (4) rearranges the protection registers to protect all of this.

The reasons for doing this are: (1) the page at address 0 should be inaccessible so that NULL pointer errors can be caught; and (2) the bottom three quarters are left unoccupied so that an FR-V CPU with an MMU can use it for virtual userspace mappings.

See include/asm-frv/mem-layout.h for constants pertaining to the memory layout.

See include/asm-frv/mb-regs.h for the constants pertaining to the I/O bus controller configuration.

(*) uClinux Memory Protection

A DAMPR register is used to cover the entire region used for I/O (0xE0000000 - 0xFFFFFFFF). This permits the kernel to make uncached accesses to this region. Userspace is not permitted to access it.

The DAMPR/IAMPR protection registers not in use for any other purpose are tiled over the top of the SDRAM such that:

(1) The core kernel image is covered by as small a tile as possible granting only the kernel access to the underlying data, whilst making sure no SDRAM is actually made unavailable by this approach.

(2) All other tiles are arranged to permit userspace access to the rest of the SDRAM.

Barring point (1), there is nothing to protect kernel data against userspace damage - but this is uClinux.

(*) Exceptions and Fixups

Since the FR40x and FR55x CPUs that do not have full MMUs generate imprecise data error exceptions, there are currently no automatic fixup services available in uClinux. This includes misaligned memory access fixups.

Userspace EFAULT errors can be trapped by issuing a MEMBAR instruction and forcing the fault to happen there.

On the FR451, however, data exceptions are mostly precise, and so exception fixup handling is implemented as normal.

(*) Userspace Breakpoints

The ptrace() system call supports the following userspace debugging features:

(1) Hardware assisted single step.

(2) Breakpoint via the FR-V "BREAK" instruction.

(3) Breakpoint via the FR-V "TIRA GR0, #1" instruction.

(4) Syscall entry/exit trap.

Each of the above generates a SIGTRAP.

(*) On-Chip Serial Ports

The FR-V on-chip serial ports are made available as ttyS0 and ttyS1. Note that if the GDB stub is compiled in, ttyS1 will not actually be available as it will be being used for the GDB stub.

These ports can be made by:

```
mknod /dev/ttyS0 c 4 64
mknod /dev/ttyS1 c 4 65
```

(*) Maskable Interrupts

Level 15 (Non-maskable) interrupts are dealt with by the GDB stub if present, and cause a panic if not. If the GDB stub is present, ttyS1's interrupts are rated at level 15.

All other interrupts are distributed over the set of available priorities so that no IRQs are shared where possible. The arch interrupt handling routines attempt to disentangle the various sources available through the CPU's own multiplexor, and those on off-CPU peripherals.

(*) Accessing PCI Devices

Where PCI is available, care must be taken when dealing with drivers that access PCI devices. PCI devices present their data in little-endian form, but the CPU sees it in big-endian form. The macros in asm/io.h try to get this right, but may not under all circumstances...

(*) Ax88796 Ethernet Driver

The MB93093 PDK board has an Ax88796 ethernet chipset (an NE2000 clone). A driver has been written to deal specifically with this. The driver provides MII services for the card.

The driver can be configured by running make xconfig, and going to:

```
(*) Network device support
    - turn on "Network device support"
    (*) Ethernet (10 or 100Mbit)
        - turn on "Ethernet (10 or 100Mbit)"
        - turn on "AX88796 NE2000 compatible chipset"
```

The driver can be found in:

```
drivers/net/ax88796.c
include/asm/ax88796.h
```

(*) WorkRAM Driver

This driver provides a character device that permits access to the WorkRAM that can be found on the FR451 CPU. Each page is accessible through a separate minor number, thereby permitting each page to have its own filesystem permissions set on the device file.

The device files should be:

```
mknod /dev/frv/workram0 c 240 0
mknod /dev/frv/workram1 c 240 1
mknod /dev/frv/workram2 c 240 2
```

```
   ...
```

The driver will not permit the opening of any device file that does not correspond to at least a partial page of WorkRAM. So the first device file is the only one available on the FR451. If any other CPU is detected, none of the devices will be openable.

The devices can be accessed with read, write and llseek, and can also be mmapped. If they're mmapped, they will only map at the appropriate 0x7e8nnnnn address on linux and at the 0xfe8nnnnn address on uClinux. If MAP_FIXED is not specified, the appropriate address will be chosen anyway.

The mappings must be MAP_SHARED not MAP_PRIVATE, and must not be PROT_EXEC. They must also start at file offset 0, and must not be longer than one page in size.

This driver can be configured by running make xconfig, and going to:

```
   (*) Character devices
       - turn on "Fujitsu FR-V CPU WorkRAM support"
```

(*) Dynamic data cache write mode changing

It is possible to view and to change the data cache's write mode through the /proc/sys/frv/cache-mode file while the kernel is running. There are two modes available:

```
   NAME    MEANING
   =====   ========================================
   wthru   Data cache is in Write-Through mode
   wback   Data cache is in Write-Back/Copy-Back mode
```

To read the cache mode:

```
   # cat /proc/sys/frv/cache-mode
   wthru
```

To change the cache mode:

```
   # echo wback >/proc/sys/frv/cache-mode
   # cat /proc/sys/frv/cache-mode
   wback
```

(*) MMU Context IDs and Pinning

On MMU Linux the CPU supports the concept of a context ID in its MMU to make it more efficient (TLB entries are labelled with a context ID to link them to specific tasks).

Normally once a context ID is allocated, it will remain affixed to a task or CLONE_VM'd group of tasks for as long as it exists. However, since the kernel is capable of supporting more tasks than there are possible ID numbers, the kernel will pass context IDs from one task to another if there are insufficient available.

The context ID currently in use by a task can be viewed in /proc:

```
# grep CXNR /proc/1/status
CXNR: 1
```

Note that kernel threads do not have a userspace context, and so will not show a CXNR entry in that file.

Under some circumstances, however, it is desirable to pin a context ID on a process such that the kernel won't pass it on. This can be done by writing the process ID of the target process to a special file:

```
# echo 17 >/proc/sys/frv/pin-cxnr
```

Reading from the file will then show the context ID pinned.

```
# cat /proc/sys/frv/pin-cxnr
4
```

The context ID will remain pinned as long as any process is using that context, i.e.: when the all the subscribing processes have exited or exec'd; or when an unpinning request happens:

```
# echo 0 >/proc/sys/frv/pin-cxnr
```

When there isn't a pinned context, the file shows -1:

```
# cat /proc/sys/frv/pin-cxnr
-1
```