UEFI Port to RISC-V Processor Architecture

UEFI Spring Plugfest – March 30, 2016
Abner Chang, SW/FW Technologist
Agenda

• Introduction
• RISC-V UEFI port on EDKII
• Spec changes for RISC-V
• Appendix
RISC and CISC

**RISC** Reduced instruction set computer, like ARM, Power PC, SPARC processor
- Fixed length of instruction, simple instructions to be executed in one CPU clock.
- Less instruction sets, RISC-V has around 90+ instructions for example.
- Better performance of instruction fetch and pipeline.
- Slightly complex when writing program in assembly language.
- Large code size

**CISC** Complex instruction set computer, like Intel processor
- Variable length of instructions, multiple clocks instruction
- To complete a task in few line assembly as possible.
- Specific instructions for specific purpose
- Easy to write program in assembly language
- Less code size
What is RISC-V processor

- From UC Berkeley, the fifth major RISC ISA design.
- Roman numeral “V” to signify “variations” and “vectors”
- New and open instruction set architecture (ISA) originally designed for computer architecture research and education
- Now become a standard open ISA for industrial
What is RISC-V processor

- 32-bit, 64-bit and 128 bit processor.
- Instruction groups
  - Base integer instruction set
  - Standard extension for integer multiplication and division
  - Standard extension for atomic instructions
  - Standard extension for single-precision floating point
  - Standard extension for double-precision floating point
  - Standard extension for quad-precision floating point
  - Standard extension for compressed instructions
  - Standard extension for bit manipulation
  - Standard extension for SIMD instruction
What is RISC-V processor

- 4 privileged operation modes

<table>
<thead>
<tr>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Machine mode</td>
</tr>
<tr>
<td>Hypervisor mode</td>
</tr>
<tr>
<td>Supervisor mode</td>
</tr>
<tr>
<td>User mode</td>
</tr>
</tbody>
</table>
RISC-V UEFI Port

EDKII OVMF RISC-V Package on QEMU

Security
SecMain.efi (SEC)

Pre EFI Initialization
(PEI)

Driver Execution Environment
(DXE)

Device, Bus, or Service Driver

Boot Dev Select
(BDS)

Transient System Load
(TSL)

Runtime
(RT)

After Life
(AL)

Reset Vector

CPU Init

Chipset Init

Board init

SEC to PEI handoff

DXE initial program loader

Boot Manager

Final OS Loader

Final OS Environment

OS-Absent App

Transient OS Loader

Transient OS Environment

OS-Present APP

UEFI/PI Execution Phases

Power on

[....Platform initialization..]

[....OS boot...]

Shutdown
UEFI/PI Execution Phases

Power on ➔ [....Platform initialization..] ➔ [....OS boot...] ➔ Shutdown
UEFI/PI Execution Phases

Power on [….Platform initialization..] [….OS boot...] Shutdown
**UEFI/PI Execution Phases**

- **Power on**
  - [..Platform initialization..]
- **Pre EFI Initialization**
  - Security SecMain.efi
  - Pre EFI Initialization (PEI)
    - CPU HOB
    - Memory address: 32-bit, I/O address: 32-bit
    - Set Jump/Long Jump
    - Set Jump requires return value on EDKII
    - Memory and I/O
    - Processor intrinsic function, Ena/Dis interrupt (CSR), Stack switch, Breakpoint (EBREAK), CPU pause (NOP)
    - BaseLib
    - Processor Binding (structure alignment, variable alignment)
    - Generate EFI image PE COFF
    - PE COFF RISC-V relocation type.
    - Prepare Temporary memory
    - Platform memory initialization
      - CPU HOB to address memory address size
      - RISC-V SET_JUMP/LONG_JUMP to switch
      - stack to permanent
      - RISC-V memory map read/write
      - RISC-V I/O read/write (memory map)
      - RISC-V BaseLib
      - RISC-V specific PEI service pointer retrieval
- **Boot Device Select**
  - Boot Device Select (BDS)
- **Transient System Load**
  - Transient System Load (TSL)
- **Runtime**
  - Runtime (RT)
- **After Life**
  - After Life (AL)

**FD, Flash Device (ROM)**

- Reset Vector (VTF) for RISC-V
- Generate Reset Vector VTF for RISC-V
- RISC-V Reset Vector (0xF...FF00)
- Volume Top
- FD, Flash Device (ROM)
- [UEFI/PI Execution Phases]
UEFI/PI Execution Phases

Power on → [....Platform initialization..] → [....OS boot...] → Shutdown

Reset Vector (VTF) → Security SecMain.efi (SEC) → Pre EFI Initialization (PEI) → Driver Execution Environment (DXE) → Boot Dev Select (BDS) → Transient System Load (TSL) → Runtime (RT) → After Life (AL)

- ProcessorBinding initialization
- CPU HOB to declare memory address size
- RISC-V SET_JUMP/LONGJ UMP to switch stack to permanent
- RISC-V memory map read/write
- RISC-V I/O read/write (memory map)
- EDKII BaseLib for RISC-V
- RISC-V specific PEI service pointer retrieval

Maintain RISC-V machine trap handler in MSCRATCH CSR

RISC-V Machine Mode

- RISC-V DXE Timer Arch protocol
- RISC-V DXE CPU arch protocol
- CpuFlushDataCache
- CpuEnableInterrupt
- CpuDisableInterrupt
- CpuGetInterruptState
- CpuInit
- CpuRegisterInterruptHandle
- CpuGetTimerValue
- CpuSetMemoryAttribute

mcause CSR
mtime CSR
mtimecmp CSR
mip CSR
mie CSR
mcause CSR
mtime CSR
mtimecmp CSR
mip CSR
mie CSR
FD, Flash Device (ROM)

Power on → [....Platform initialization..] → [....OS boot...] → Shutdown

UEFI/PI Execution Phases
FD, Flash Device (ROM)

UEFI/PI Execution Phases

Power on → [....Platform initialization..] → [....OS boot...] → Shutdown
### Security
- `SecMain.efi`

### Pre EFI Initialization
- `Pre EFI Initialization`
- `Driver Execution Environment`
- `Boot Dev Select`
- `Transient System Load`
- `Runtime`
- `After Life`

### UEFI/PI Execution Phases

**Power on**[

*Platform initialization]*

**[UEFI/PI Execution Phases]**

**Shutdown**
Power on                                 
[….Platform initialization..]        
[…. OS boot...]                     
Shutdown 

Security
SecMain.efi (SEC)

Pre EFI Initialization (PEI)
Driver Execution Environment (DXE)

Boot Dev
Select (BDS)
Transient System Load (TSL)
Runtime (RT)

After Life (AL)

Reset Vector (VTF)

UEFI/PI Execution Phases

FD, Flash Device (ROM)

UEFI Interactive Shell v2.1
EDK II
UEFI v2.58 (EDK II, 0x00010000)
map: No mapping found.
Press ESC in 1 seconds to skip startup.nsh or any other key to continue.
Shell> _

142,147,584 bytes of system memory tested OK
Specification changes for RISC-V
UEFI/PI Spec changes for RISC-V (in progress)

UEFI spec change for RISC-V
- 2.1.1. UEFI Images
- 2.3. Calling Conventions
- 2.3. RISC-V 32 (64/128) Platforms
- 17.2 EFI Debug Support Protocol

PI spec change for RISC-V
- Volume 1: 5.4 RISC-V PEI Services Table Retrieval
- Volume 3: PI Status code
Microsoft will release RISC-V related definition in next PE/COFF specification.

**PE/COFF image machine type,**
- IMAGE_FILE_MACHINE_RISCV32 0x5032
- IMAGE_FILE_MACHINE_RISCV64 0x5064
- IMAGE_FILE_MACHINE_RISCV128 0x5128

**RISC-V image relocation types,**
- IMAGE_REL_BASED_RISCV_HI20 5
- IMAGE_REL_BASED_RISCV_LO12I 7
- IMAGE_REL_BASED_RISCV_LO12S 8
Appendix
RISC-V QEMU

- QEMU RISC-V PC/AT board
  Built up RISC-V PC/AT board on QEMU with some PC peripherals.
- QEMU PC/AT memory map devices (CMOS, PM, PCI and other devices)
  Changed these PC peripherals to memory map I/O device because RISC-V uses memory map I/O.
• BERI (Bluespec Extensible RISC Implementation) Programmable Interrupt Controller

BERI PIC is attached to BERI processor. BERI PIC supports large number of external interrupts, total up to 1024 interrupt sources.
Need more in RISC-V spec

- Bus Interface
- PI Management Mode support
- ACPI support
- MP support
- Reset mechanism
RISC-V Implementations

- lowRisc Rocket chip (RISC-V based SoC)
- RISC-V in AXIOM First Fully Open Source 4K Film Camera
- A 32-bit 100MHz RISC-V Microcontroller with 10-bit SAR ADC
- SoC for a Satellite Navigation Unit based on the RISC-V single-core Rocket chip
- RISC-V-based photonic processor
Thanks for attending the UEFI Spring Plugfest 2016

For more information on the Unified EFI Forum and UEFI Specifications, visit http://www.uefi.org

presented by

Hewlett Packard Enterprise