UEFI Porting Techniques on ARM SoCs

UEFI PlugFest—March 18-22, 2013
Presented by Prax Fang
(American Megatrends)
Agenda

• Introduction
• Power-on Process
• Boot to OS
• Porting Guidelines
• Summary
Introduction
U-Boot vs UEFI

**U-Boot**

1. Performs platform initialization
2. Sets the boot arguments
3. Passes control to the kernel image

**UEFI/Aptio**

1. Contains U-Boot base features
2. Cross-platform build environment
3. Modular design
4. Compatible with existing x86 module
Build Environment Requirements

ARMGCC
Linux

Vendor’s Compiler
Windows

Python
Supported Framework

• Code Name
  – CortexA8
  – CortexA9
  – CortexA15
  ...

• Instruction Set
  – Arm V7
  – Arm 9
  ...

Power-on Process
Power-on Process

• There are two cases of booting UEFI on ARM:
  – UEFI as the final stage
    • Some vendor supplied firmware does initialization and then gives control to a process that provides a basic UEFI layer to provide UEFI boot.
  – UEFI from power-on
    • UEFI is the only firmware image does the entire boot process from the first instruction executed after power-on to the start of the operating system on your platform
Power-on Process

- Example: TI OMAP Series

**Power-On**

1. Performs minimal clocks, memories
2. Loads the x-loader into SRAM and executes it

**HW Vendor ROM Firmware**

1. Sets up the pin muxing
2. Initializes clocks and memory
3. Loads the U-Boot into SRAM and executes it

**X-loader**

1. Performs some additional platform initialization
2. Sets the boot arguments
3. Passes control to the kernel image

**U-Boot**

1. Performs some additional platform initialization
2. Sets the boot arguments
3. Passes control to the kernel image

**OS Kernel**
Power-on Process

- UEFI as the final stage power-on flow:
  - Example: TI OMAP Series

Traditional:
- Power-On
  - HW Vendor ROM Firmware
  - X-loader
  - U-Boot
  - OS Kernel

UEFI:
- Power-On
  - HW Vendor ROM Firmware
  - X-loader
  - U-Boot
  - OS Kernel
Power-on Process

• UEFI power-on flow:
  – Follows traditional UEFI boot sequence through SEC through BDS
    • for more info please visit the website. www.uefi.org
  – Does same initialization of the platform throughout the phases
    • Such as: clock, memory and cpu initialization

Power-On → SEC → PEI → DXE → BDS → OS Kernel
Simplified UEFI Boot

– Not all UEFI phases are essential:
  • PEI phase can be skipped altogether if DRAM initialization is done in the SEC phase
Boot to OS
Boot to OS

• Follow the UEFI Framework
  – Look for supported boot devices
    • There is a FAT32 partition with an EFI directory
    • Inside it there is the BOOT folder that contains a file BootArm.efi (UEFI Boot Loader)
    • You can obtain UEFI Boot Loader by your OS.
  – Load image into memory and executes it
    • Image is verified if the Secure Boot is enabled
  – Pass control to the UEFI Boot Loader
    • The loader does the rest of the OS boot process
Porting Guidelines
Porting Guidelines

• Follow the porting guide released by SoC vendor to do the SoC initialization.
  – Using the porting guide you can create the required UEFI libraries
  – Very similar to x86 porting

• If U-Boot is available, use it as a reference to create the UEFI drivers and interfaces
Follow the steps to set the chip

- Example: Trigger software cold reset
  - Source: TI porting guide

<table>
<thead>
<tr>
<th>Physical Address</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x4A30 7B90</td>
<td>Global software cold and warm reset control. This register is auto-cleared. Only write 1 is possible. A read returns 0 only.</td>
</tr>
<tr>
<td>Type</td>
<td>RW</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bits</th>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:2</td>
<td>RESERVED</td>
<td>Global COLD software reset control. This bit is reset only upon a global cold source of reset. 0x0: Global COLD software reset is cleared. 0x1: Triggers a global COLD software reset. The software must ensure the SDHARM is properly put in self-refresh mode before applying this reset.</td>
</tr>
<tr>
<td></td>
<td>RST_GLOBAL_COLD_software</td>
<td>RW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x0000 0000</td>
</tr>
<tr>
<td>0</td>
<td>RST_GLOBAL_WARM_software</td>
<td>Global WARM software reset control. This bit is reset upon any global source of reset (warm and cold). 0x0: Global warm software reset is cleared. 0x1: Triggers a global warm software reset.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>
Follow the steps to set the chip

• Example: Trigger software cold reset
  – Target: UEFI ResetSystem Library

```c
EFI_STATUS
LibResetSystem (
  IN EFI_RESET_TYPE ResetType,
  IN EFI_STATUS ResetStatus,
  IN UINTN DataSize,
  IN CHAR16 *ResetData OPTIONAL
)
{
   :
   MmioOr32 (PRM_RSTCTRL, BIT1);
   :
}
```
Porting Guidelines

• Use the U-boot code
  – Most ARM vendors provide U-Boot code that initializes the following components:
    • cpu, board, serial, dram...etc
  – Using this U-Boot code as a reference, you can develop UEFI code to use these peripherals
Debugging Tips for UEFI on ARM

• Example: Write Debug Messages to Serial
  – Source: U-Boot Serial Code

```c
static void do_boot_linux ()
{
    debug ("## Transferring control to Linux ... \n");
}
```
Debugging Tips for UEFI on ARM

• Example: Write Debug Messages to Serial
  – Target: UEFI Code

```c
EFI_STATUS SomeFunction ( …
  ...
 )
{
  :
    DEBUG ((EFI_D_INFO, “This is a debug message\n”));
  :
}
```

• The serial debug messages for UEFI should be very familiar to x86 developers
Summary
Summary

• UEFI is a neutral boot firmware capable of booting both EFI and non-EFI compliant Operating Systems
  – Certain OSes like Windows already require UEFI for SecureBoot
• Porting on an ARM chipset is similar to porting an x86 platform
  – There are porting guides and reference code provided by the chip vendors
• ARM platforms can also use common debugging techniques like serial debug messages and JTAG debuggers
• UEFI also allows re-use of all features developed for x86 if it is hardware independent
Questions?
Thanks for attending the UEFI Spring PlugFest 2013

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

presented by

American Megatrends