Zynq Booting & PetaLinux Tutorial + Demo

Keyshav Mor, Petr Žejdl
CMS-DAQ group

13 June 2019

Acknowledgements: M.Dobson, F.Meijers, A.Zahid

What is a Zynq?

- Xilinx SoCs/MPSoCs is an ASIC that integrates **processing system** - ARM microprocessor(s), I/O (memory, PCI Express, USB, Ethernet, I2C, serial line), and **programmable logic** (FPGA) in a single chip.
  - I/O is part of the ASIC (does not consume any part of the programmable logic)
  - SoC: System-on-chip
  - MPSoC: Multiprocessor system-on-chip (has GPU and real-time CPU)
  - RFSoC: Radio Frequency System-on-Chip (has high-speed ADC, DAC)

Zynq Versions

Zynq-7000 SoC
- Single/Dual ARM Cortex-A9
  - **32-bit**
  - Up to 1 GHz
  - L1 Cache 32KB
  - L2 Cache 512KB
  - On-chip Memory 256KB
- **I/O**
  - DDR3, DDR2 RAM
  - USB 2.0, Gigabit Ethernet
  - SD/SDI, UART, CAN, I2C, SPI, GPIO
- **FPGA**
  - PCI Express Gen2 x4/x8
  - Transceivers 6.25 / 12.5 Gb/s

Zynq UltraScale+ MPSoC
- Dual/Quad ARM Cortex-A53
  - **64-bit**
  - Up to 1.5 GHz
  - L1 Cache 32KB
  - L2 Cache 1MB
  - On-chip Memory 256KB
- Dual ARM Cortex-R5 (Hard Real-Time)
- ARM Mali-400 MP2 (GPU)
- **I/O**
  - DDR4, DDR3 RAM
  - USB 3.0, USB 2.0, Gigabit Ethernet
  - SD/SDI, UART, CAN, I2C, SPI, GPIO
  - **PCI Express Gen2 x4**
  - SATA 3.1, Display Port, ADC, DAC
- **FPGA**
  - PCI Express Gen3 x16
  - Transceivers 16.3 / 32.75 Gb/s
  - 100G Ethernet MAC
1. **Pre-configuration stage**
   - Reset and wake-up processes driven by ROM code

2. **Configuration stage**
   - Loads the first-stage boot loader (FSBL) code into the on-chip RAM (OCM). Can be executed either by APU or RPU.

3. **Post-configuration stage**
   - After FSBL execution starts, i.e. U-Boot, Linux, ...

Details: UG1228 - Zynq UltraScale+ MPSoC Embedded Design Methodology Guide
Boot process – Pre-configuration Stage – PMU

PMU - Platform Management Unit

- The PMU passes control to the CSU (Configuration and Security Unit), which checks whether authentication and/or decryption is required.
- PMU serves as Power and Safety Manager
  - Powers up and down peripherals
  - Manages clocks, resets and initializes PLL
  - Manages sleep modes
  - Wake-up system on various triggers

- Implementation
  - Triple Redundant MicroBlaze Processor
  - 128KB ram with ECC
  - 32KB ROM

- PMU Firmware Source:
  https://github.com/Xilinx/embeddedsw/tree/master/lib/sw_apps/zynqmp_pmufw
Boot process – Configuration Stage – CSU

CSU – Configuration and Security Unit

- Reads the FSBL (First-State Boot Loader) in OCM for execution by either the RPU or APU.
- Loads the PMU user firmware into the PMU RAM for execution.
- Supports image authentication and decryption (Secure Boot)

- Implementation
  
  - Triple-redundant embedded processor with ROM and a small RAM for sensitive data storage
  - A crypto interface contains AES-GCM, a key vault for key storage, DMA, SHA3, RSA
  - Processor configuration-access port (PCAP) interface
    - Used to configure PL (FPGA) part from FSBL
CSU – Configuration Security Unit

- Determines the boot mode / boot device (by bootstrapping Zynq external pins)
- BootROM can boot from following devices:
  - FLASH - Quad-SPI, SD, eMMC, NAND
  - JTAG
  - USB through Device Firmware Upgrade (DFU) protocol
- BootROM is looking for a valid boot header ("Golden Image Search")
  - Identification string “XLNX” and valid boot header checksum
  - Boot images can be located every 32 KB in the boot memory device (more than one is allowed)
  - On SD/eMMC it is looking for files on FAT16/32 filesystem:
    - boot.bin, alternatively boot0001.bin, boot0002.bin
  - FSBL MultiBoot / Fallback – FBSL asks CSU/BootROM to boot from another image
    - Possibility of having a fallback image if the primary one gets corrupted
Boot process – Post Configuration – FSBL, ATF, U-Boot

**FSBL – First-State Boot Loader**

- Execution by either the RPU or APU
- Brings the entire system
  - Configures PL (FPGA) part through PCAP port
  - Loads ATF (ARM Trusted Firmware)
    - Handles SMC (Secure Monitor Calls)
  - Loads the second state boot loader (e.g. U-Boot) or starts the application software (or Linux directly)

- Zynq MPSoC ATF Source: https://github.com/Xilinx/arm-trusted-firmware
- Zynq MPSoC FSBL Source: https://github.com/Xilinx/embeddedsw/tree/master/lib/sw_apps/zynqmp_fsbl
Boot Process for Zynq-7000

Boot process

- Internal BootROM code is executed on CPU0 (APU).
- Configures the system and fetches the First State Boot Loader to OCM.
- Starts executing the FSBL…
- Limitations
  - The SD card boot mode does not support header search or multiboot.
  - The SD card boot mode is not supported in 7z010 dual core and 7z007s single core CLG225 devices.
- Zynq-7000 FSBL Source: https://github.com/Xilinx/embeddedsw/tree/master/lib/sw_apps/zynq_fsbl
Required Files(Images) for Successful Linux Boot

- Boot image / boot.bin
  - First Stage Boot Loader (fsbl.elf)
  - Platform Management Firmware (pmufw.elf)
  - Optionally PL (FPGA) bitstream (system.bit)
  - ARM Trusted Firmware (bl31.elf)
  - Second Stage Boot Loader (e.g. uboot.elf for **U-Boot**) or application software

- Linux binary image – image.ub - U-Boot’s Flattened Image Tree (FIT)
  - Flattened Device Tree blob (system.dtb)
  - Linux Kernel (Image)

- A root-filesystem (in one form)
  - Ramdisk (initrd)
  - Disk partition (SD card, flash, …)
  - NFS file system

- All files are generated by PetaLinux Tools “automatically”, but plenty of combinations are possible
Flattened Device Tree

• Introduced in kernel 2.6 as a way to describe non-discoverable hardware
  – Embedded systems (usually) don’t have BIOS, ACPI and/or UEFI.
  – The same information was previously hard-coded in the source code

• Is a data structure, describing:
  – The number and type of CPUs, size of RAM
  – I/O memory mapping and resources
    • Base address, size, IRQs, ...
  – Kernel Command Line Arguments
  – The format allows to describe almost anything

• Can statically linked into the kernel, loaded by the boot loader and passed at boot time or even at Linux run time

• The device tree source is described in a text files (.dts) and compiled by Device Tree Compiler (dtc) into a binary format (.dtb blob file)
Booting summary, Outlook (1)

- Not covered secure boot
- Not covered RPU
- Not covered power considerations (different power domains)

- Large degree of freedom for customizations covering various use-cases
  - Different ARM and/or I/O configurations
  - Different boot devices
  - Different root filesystems
  - Different approaches (e.g. toolchain integrations into existing work flow…)
  - Different versions of development tools
Booting summary, Outlook (2)

- Can we benefit from a common approach (not only during this workshop)?
  - Can we have common Linux Distribution?
    - PetaLinux based?
    - Yocto based?
    - Centos7 based?
    - Arch based?
    - ... based?
    } Not a distribution per-se, more a framework to make a custom one.
    } Traditional Linux distributions, having pre-compiled packages.
    → Look at tutorials on “CentOS on Xilinx Zynq Ultrascale+ MPSoC”
  - Can we have a common Linux Kernel?
  - Both supported at CERN level?
  - What about system administration?
    - TFTP booting
    - NFS root filesystem
    - Common log server
    - Network support
    - ...
    } Look at “SysAdmin - Networking, Security & System-administration” on Friday
PetaLinux Tutorial + Demo

• Showing ZedBoard (ZYNQ-7000) TFTP boot with NFS root filesystem

• Prerequisites – PC with:
  – TFTP server
  – NFS server
  – DHCP server
    • Gives filename for TFTP boot
    • Gives NFS-root path

  – If in doubt, ask your friendly sysadmin team :-}