Linux for Xilinx Zynq Ultrascale+ based Embedded Systems in the CMS DAQ Network

Authors: Keyshav Suresh Mor, Petr Zejdl and Marc Dobson
Contact: keyshav.suresh.mor@cern.ch

Motivation: CMS Phase-II Upgrade

- Upgrade of the CMS detector and electronics for HL-LHC (High Luminosity LHC) in 2024
- Extensive use of embedded systems (SoC) in the new electronics
  - Devices capable of running server-grade Linux OS
  - For control, configuration and monitoring
  - System requires network connection
  - Mostly based on Xilinx Zynq Ultrascale+ SoC
  - **Deployed at a scale of ~1000 devices**
- The scale poses challenges for the integration of SoCs in CMS
  - Hardware is not uniform due to the detector layout
  - Different software requirement and software life cycle
  - OS selection and support
  - Network, system administration and scalling challenges

Operating System Issues

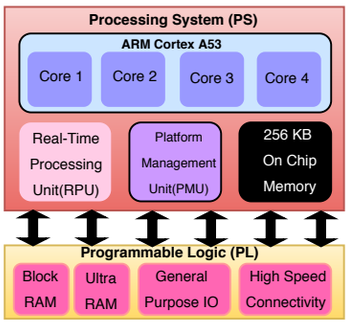
- Many Linux versions available for ARM (reference Linux from Xilinx: Patalinux, Yocto, Arch, CentOS)
- Hardware developers may prefer various OS distributions or versions
- Linux System Administratior may prefer to support only single OS version (reducing manpower)
- Can existing knowledge be re-used and pooled by using the same OS as PCs at CERN?
- CentOS (RHEL based Linux)
  - Not supported by Xilinx, but:
  - Kernel can be used from the Xilinx toolchain
  - Preferably CentOS default kernel could be used with Xilinx specific drivers (requires porting)
  - CentOS root file system remains unchanged

Hardware Issues

- Reliable, fault tolerant booting mechanism
- Automatic failover to golden image in case of failure to boot
- HW address (MAC) in standard EEPROM for all board designs
- Reliable and fault tolerant mechanism to update files on SD card (FSBL, U-Boot, and maybe firmware)

Xilinx Zynq Ultrascale+

- Zynq devices tightly integrate the **programmable logic** (PL) with the **processing system** (PS).



System Administration Issues

Integration Issues

- Devices need to be integrated into the CMS experiment technical and control network
- Network specific settings: IP addresses, DHCP, DNS, NTP
- Sufficient bandwidth for the primary task of control and supervision of the CMS electronics and for services (NFS, logging, etc…)

System Administration

- Centrally administered OS with regular updates and security patches: CentOS for common knowledge with PCs?
- Central Configuration Management System as for PCs: Puppet
- Mechanism to update Root file system or Board specific packages with Puppet
- Same user database (ldap, kerberos) across all platforms

Scaling Issues

NFS Root File System

- Aim to scale much less than by the number of connected devices
- Use a common Read Only root file system (most space is used by files which are only read)
- Use a Read/Write overlay (or copy on write) like VMs or Docker images
- Root file system can be updated for a number of devices at once and pushed centrally
- Add NFS servers as needed for performance and scale; or use commercial appliance e.g. NetApp NFS Filer

Logging

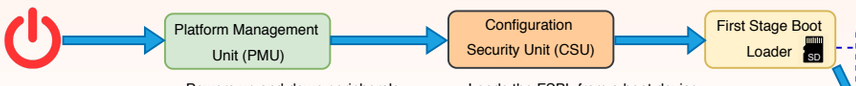
- No local storage on devices, NFS performance bottleneck if using root file system for log storage
- Central log server with local disks, and long term backup
- Maintain logs for all boards/devices in central location
- Access logs even if board crashes
- Add log servers as needed for performance and scale

CMS DAQ Proposed Solution

- Benefit from common knowledge across CMS/CERN and use a common approach for all the issues
- Minimize the manpower required for software development, integration and administration
- **Use centrally managed Linux distribution for SoC hardware at CERN**

- Linux distribution based on CentOS 7/8 with minimum changes
- CentOS default kernel with Xilinx patches backported
- Fault tolerant booting with automatic failover
- Minimum files required on SD card (FSBL, U-Boot)
- Fully booted from network (including all dependencies e.g. firmware, customizations)

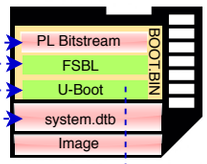
Power On Sequence



- Powers up and down peripherals
- Manages clocks, resets and initializes PLLs
- Manages sleep modes, monitors the system

- Loads the FSBL from a boot device into On-Chip Memory (OCM)
- Enables ARM CPU

Contents of SD Card



- For a complete boot from SD card, it must contain the **BOOT.BIN**, **system.dtb and Image** files.
- The Image file contains the **kernel image** and a compressed root file system.
- For network boot, only **BOOT.BIN** is required.

Files used for booting Linux on Xilinx Zynq devices

Hardware Design

Xilinx Vivado

Xilinx SDK

FSBL Generation

- **PL Bitstream (.bit):** The bitstream is used to program the programmable logic which would be eventually controlled by the processing system (PS).
- **Hardware Description File (.hdf):** File which describes the hardware, register and memory offsets being programmed by the bitstream, used to generate device tree.

- **First Stage Boot Loader (fsbl.elf):** Initialises peripherals and memory before handing over control to the ARM Trust Firmware, which then loads the U-Boot in the OCM.

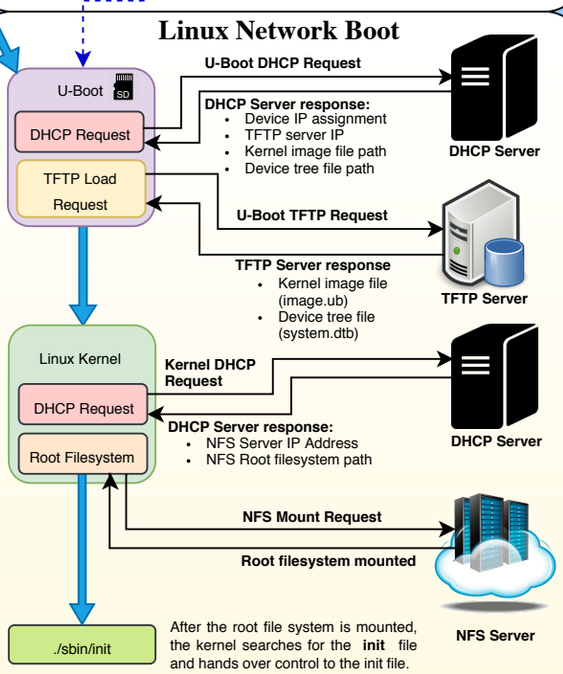
yocto PROJECT

GCC

- **U-Boot (u-boot.elf):** Intermediate bootloader which loads the device tree and the kernel in the memory.
- **Device Tree (system.dtb):** Device tree describes underlying hardware, register and memory offsets (like BIOS) to the Linux kernel when the kernel executes.
- **Linux Kernel (image.ub):** Linux kernel is responsible for enabling drivers for the configured underlying hardware and activating services and features supported by the hardware. **Linux Kernel** works as interface between userspace and hardware.

- **Root Filesystem** is the file system where the drivers, files, softwares and services used by the user are located and installed. **init** is the first process which executes after kernel finishes executing.

Linux Network Boot



After the root file system is mounted, the kernel searches for the **init** file and hands over control to the init file.