Developing with PetaLinux for the Ultra96

SAN19-314

Tom Curran
Self Introduction

- Avnet – 12 years
- Global Products & Emerging Technologies Team
  - Technical training courses
  - Reference designs
  - PetaLinux BSPs
  - Dev board development
  - Embedded Linux & software development for Zynq, Zynq MPSoC, and MicroBlaze
Agenda

● Intro to Ultra96-V2
● Intro to Zynq MPSoC architecture
● Intro to PetaLinux
● Anatomy of a PetaLinux build
● Integrating a custom device driver
● Intro to Vivado and Petalinux build scripts
● Summary
Ultra96-V2 Kit

● 96boards Consumer Edition

● What’s in the box?
  ○ Ultra96-V2 featuring Zynq MPSoC ZU3EG
  ○ 16 GB MicroSD card + adapter
  ○ 1 Year SDS0C License Voucher
  ○ Quick Start Instructions

● Available online from Xilinx:
  ○ SDS0C, SDK, Vivado WebPACK (supports ZU3EG)
  ○ PetaLinux

● From Avnet: http://avnet.me/ultra96-v2
  ○ Schematics, BOM, UG, etc.
  ○ Tutorials and reference designs
Agenda

- Intro to Ultra96-V2
- **Intro to Zynq MPSoC architecture**
- Intro to PetaLinux
- Anatomy of a PetaLinux build
- Integrating a custom device driver
- Intro to Vivado and Petalinux build scripts
- Summary
Zynq UltraScale+ MPSoC Architecture

- Processing System (PS)
  - ARM® Quad Cortex-A53 & Dual Cortex-R5 System
- Custom Accelerators
- Memory Interfaces
  - Common Peripherals
  - Custom Peripherals
  - UltraScale+ Programmable Logic (PL)
- Common Peripherals
Agenda

- Intro to Ultra96-V2
- Intro to Zynq MPSoC architecture
- **Intro to PetaLinux**
  - Anatomy of a PetaLinux build
  - Integrating a custom device driver
  - Intro to Vivado and Petalinux build scripts
- Summary
Intro to PetaLinux

● A build system:
  ○ A set of open-source cross-platform development command line and menu tools for use running under an x86 Linux Host OS
  ○ Tailors embedded Linux to run on many Xilinx platforms: Zynq MPSoC, Zynq and MicroBlaze.
  ○ PetaLinux tools now use the open-source community Yocto system since v2016.3
    ■ This is very useful for embedded Linux developers!
  ○ Also includes an emulator (QEMU) that can emulate PetaLinux under x86 Linux
    ■ Very useful for Kernel and some device driver development

● A productivity layer:
  ○ Just six PetaLinux commands (with many options):
    petalinux-create, petalinux-config, petalinux-build, petalinux-boot, petalinux-package, petalinux-util
PetaLinux vs Other Linux Dev Tools

- Ease of integration of customized hardware platforms
  - Zynq SoCs and FPGAs are endlessly configurable in ways that other ARM SoCs are not
  - Import PL bitstream and PS configuration

- Automatic devicetree generation based on Vivado import

- Automatic system peripheral detection
  - STDIO UART, etc.

- Other Linux providers will run on Zynq & Zynq MPSoC, but their build tools don’t have a way to import the hardware definition from Vivado
What’s Included

● Source code, libraries, applications and Yocto recipes
  ○ Mostly open-source based with a few exceptions for some hardware drivers
  ○ PetaLinux includes and manages the Linux kernel sources and libraries
  ○ Various hardware drivers and modules
  ○ Traditional Linux applications and utilities

● Through Yocto, PetaLinux can make use of additional embedded Linux applications beyond what Xilinx supplies.
  ○ More on that later!
PetaLinux Build Products

- Bootloaders and firmware
  - FSBL, ARM Trusted Firmware, PMU Firmware, u-boot
- A Linux OS image that runs on Xilinx Zynq MPSoC, Zynq, and MicroBlaze CPUs
  - 2018.x uses Linux kernel version 4.14
  - 2019.x uses Linux kernel version 4.19
  - For single or multiple CPUs
- Linux applications, device drivers and lots of logs and configuration files
- Detailed hardware and memory requirements can be found here: [http://avnet.me/xilinx_ug1144](http://avnet.me/xilinx_ug1144)
PetaLinux Tools Testing, Release, and Documentation

- **u-boot, Linux kernel, Xen hypervisor**
  - **Sources:**
    - Kernel.org + Xilinx patches & additions
  - **Testing:**
    - Xilinx silicon
    - Xilinx reference board features
  - **Documentation:**
    - Source code
    - Open source communities/forums
    - Answer Records for select interim issues

- **Linux file system**
  - **Sources:**
    - Yocto community code base
  - **Testing:**
    - Build verification
    - Testing of Xilinx-unique software and features
  - **Release strategy:**
    - Bug fixes with next Yocto release integration
  - **Documentation:**
    - Source code
    - Open source communities/forums
Linux Boot for Zynq MPSoC

- Linux
- U-Boot
- ARM Trusted Firmware
- PMU Firmware
- FSBL

- Security
- ARM Cortex-A53
- Programmable Logic
- I/O
- GPU
- PMU
- RPU

Processing System
- Application Processing Unit
  - ARM Cortex-A53
  - Floating Point Unit
  - MMU
  - Trace
- GIC-400
- SCU
- GO/SA/MVU
- 2 MB L2

Graphics Processing Unit
- ARM Mali-400 MP
- Geometry Processor
- Pixel Processor
- MMU
- 64 KB L2 Cache
- 256 KB OCM with ECC

DDR Controller
- DDR4/3/1, LPDDR4/3 ECC Support

Real-Time Processing Unit
- ARM Cortex-R5
- Vector FPU
- Memory Protection Unit
- Trace
- V/T Monitor
- System Management
- Security
- Config AES, Decryption, Authentication, Secure Boot
- TrustZone

Security

Platform Management Unit
- Power
- System Management
- CMA, Timer, WDT, Resets, Clocking, Debug

Programmable Logic
- Storage & Signal Processing
  - Block RAM
  - UltraRAM
  - DSP
- General-Purpose I/O
  - GTH
  - GTP
  - PCIe Gen
- High-Speed Connectivity
  - 10G EMAC
  - Interlaken

Programmable Logic
- Video Codec
- AMS
PetaLinux Tools Flow

- Export from Vivado
- Select BSP or import from Vivado
- Configure
  - Kernel
  - Applications
  - System settings
- Build
  - Kernel
  - Root filesystem
- Boot
  - QEMU
  - Configure Zynq PL
  - Download image
- petalinux-create
- petalinux-config
- petalinux-build
- petalinux-boot
- petalinux-package (optional)
PetaLinux Install and Project Taxonomy

- Tools and default source components in install area
  - **petalinux**
  - **components**
    - tools
    - apps
    - misc
    - etc
    - yocto

- Per-project directory
  - Configuration files
  - Custom recipes
    - Applications, scripts, libraries, etc.
  - Build products
  - **myproj**
    - components
      - plnx_workspace
    - images
    - project-spec
    - build
    - conf
    - recipes-apps
    - recipes-bsp
    - recipes-dt
    - recipes-...
PetaLinux and Yocto

- PetaLinux supports and is supported by the Yocto ecosystem
- PetaLinux tools have the ability to integrate Yocto-generated RPM feeds into the native configured Linux filesystem
  - Allows designs to fully leverage in-house and Yocto-community software
- Yocto build system can generate the complete PetaLinux filesystem
  - Ensures that all Yocto-built software is library-compatible with Xilinx-provided software
  - Allows comprehensive revision and source control
- Xilinx provides Yocto recipes for all PetaLinux components
  - ARM Trusted Firmware
  - PMU Firmware
  - u-boot
  - Kernel
  - Root filesystem
PetaLinux and Bitbake

- PetaLinux commands abstract the details of the bitbake commands
  - Creation of new software applications, kernel modules (drivers), scripts, etc.
  - Add new bitbake recipes
  - Additions to configuration menus
  - Build, clean, config
  - Installation onto the root file system

- Easy edits to makefile and recipe as application complexity grows
  - Add source files, etc.

- PetaLinux command to build the root file system becomes bitbake command...
  
  ```$ petalinux-build -c rootfs
  $ bitbake petalinux-user-image -c do_rootfs```
Agenda

● Intro to Ultra96-V2
● Intro to Zynq MPSoC architecture
● Intro to PetaLinux
● **Anatomy of a PetaLinux build**
● Integrating a custom device driver
● Intro to Vivado and Petalinux build scripts
● Summary
Create a PetaLinux Project

- `petalinux-create`
  - Creates a new PetaLinux project or component
  - Usage
    
    ```
    petalinux-create [options] \
    -t|--type <TYPE> \ 
    -n|--name <COMPONENT_NAME>
    ```

<table>
<thead>
<tr>
<th>Required/Optional</th>
<th>Available Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-t, --type &lt;TYPE&gt;</code></td>
<td>project - Create a PetaLinux project</td>
</tr>
<tr>
<td></td>
<td>apps - Create an application</td>
</tr>
<tr>
<td></td>
<td>libs - Create a library</td>
</tr>
<tr>
<td></td>
<td>modules - Create a module</td>
</tr>
<tr>
<td><code>-n, --name &lt;COMPONENT_NAME&gt;</code></td>
<td>Specify name for the component/project</td>
</tr>
<tr>
<td></td>
<td>Not required if BSP is specified</td>
</tr>
</tbody>
</table>
Configuring a PetaLinux Project

- **petalinux-config**
  - Configures the PetaLinux project or component
  - Usage

  ```
  petalinux-config [options] {--component <COMPONENT> |\
  --get-hw-description[=SRC] |\
  --searchpath --<ACTION> [VALUE]}
  ```

<table>
<thead>
<tr>
<th>Required/Optional</th>
<th>Available Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>--get-hw-description[=SRC]</code></td>
<td>Get hardware description</td>
</tr>
<tr>
<td></td>
<td>Look in the location of Vivado SDK export</td>
</tr>
<tr>
<td><code>-p, --project &lt;PROJECT&gt;</code></td>
<td>Path to PetaLinux project</td>
</tr>
</tbody>
</table>
Customize the Ultra96-V2 Rootfs Config Menu

- Edit the bbappend file to add extra packages to rootfs config menu
  
  `<>/project-spec/meta-user/recipes-core/images/petalinux-image-full.bbappend`

- Add the desired packages and package groups (too many to show here)

  ```
  IMAGE_INSTALL_append = " packagegroup-petalinux-ultra96-webapp"
  IMAGE_INSTALL_append = " packagegroup-petalinux-v4lutils"
  IMAGE_INSTALL_append = " packagegroup-petalinux-96boards-sensors"
  IMAGE_INSTALL_append = " packagegroup-petalinux-x11"
  IMAGE_INSTALL_append = " packagegroup-petalinux-matchbox"
  ```
Configure the Ultra96-V2 Rootfs

- `petalinux-config -c rootfs`
Building a PetaLinux Project

- `petalinux-build`
  - Builds the PetaLinux project or component
  - Usage
    ```
    petalinux-build [options]
    ```

<table>
<thead>
<tr>
<th>Required/Optional</th>
<th>Available Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-p, --project &lt;PROJECT&gt;</code></td>
<td>Path to PetaLinux project</td>
</tr>
</tbody>
</table>
| `-c, --component <COMPONENT>` | Specify the component  
  If specified, e.g.:  
  `-c rootfs`  
  `-c rootfs/myapp` |
Agenda

- Intro to Ultra96-V2
- Intro to Zynq MPSoC architecture
- Intro to PetaLinux
- Anatomy of a PetaLinux build
- **Integrating a custom device driver**
- Intro to Vivado and Petalinux build scripts
- Summary
Microchip WiFi & Bluetooth LE for Ultra96-V2

- Replaces the TI WiLink™ 8 module on the Ultra96-V1
- Certified in 75 countries globally

But...
- Only tested with Microchip ARM Cortex-M host CPUs and Microchip dev boards
Custom Integration of Microchip WiFi & Bluetooth LE

- Correct device driver not in mainline
- Driver under continuous development by Microchip, so need a custom recipe
- Edits and fixes needed to run on ARM A53 CPU inside Zynq UltraScale+ MPSoC
  - ARM v7 vs ARM v8 (Zynq MPSoC)
  - Turn off power management during DHCP IP address fetch & refresh
  - Disable driver manipulation of module enable and reset pins
- Avnet forked the driver sources into own github repository
  - Extensive email threads and collaborative conversations with Microchip
  - Working with Microchip to push changes upstream
Create Custom Kernel Module

- Add kernel module to PetaLinux project
  ```
petalinux-create -t modules -name wilc -enable
  ```
- Create bitbake recipe file
  - Specify where to fetch source files:
    ```
    SRC_URI = "git://github.com/avnet/u96v2-wilc-driver;protocol=http;branch=v15_2"
    ```
  - Specify the git commit hash of the correct revision controlled sources
    ```
    SRCREV = "01ab7484e0e6b2191c69d7ec7c6e89da5ca51f0f"
    ```
  - Specify KCONFIG and MAKE options
    ```
    EXTRA_OEMAKE = 'CONFIG_WILC=y \
                   WLAN_VENDOR_MCHP=y \
                   CONFIG_WILC_SDIO=m \
                   CONFIG_WILC_SPI=n \
                   CONFIG_WILC1000_HW_OOB_INTR=n \
                   KERNEL_SRC="${STAGING_KERNEL_DIR}" \
                   O=${STAGING_KERNEL_BUILDDIR}'
    ```
Agenda

- Intro to Ultra96-V2
- Intro to Zynq MPSoC architecture
- Intro to PetaLinux
- Anatomy of a PetaLinux build
- Integrating a custom device driver
- **Intro to Vivado and Petalinux build scripts**
- Summary
Avnet Github Script Repositories

● Avnet git repositories
  ○ bdf
    https://github.com/Avnet/bdf
  ○ hdl
    https://github.com/Avnet/hdl
  ○ petalinux
    https://github.com/Avnet/petalinux

● Clone the repositories into the same parent folder
  ○ Scripts rely on relative paths

● Checkout the branch of the Vivado & PetaLinux tools you are using
  ○ 2018.2, 2018.3, etc.
Why Use Build Scripts to Build PetaLinux BSP?

- PetaLinux build scripts automate the steps to create and build the BSP
  - Provides consistency and avoids confusion
- Revision controlled so changes can be tracked
- Easily updated to new Vivado & PetaLinux tool versions
- Can build many BSPs at once (different SOM variants)
- Can build many boot binaries at once (SD card, QSPI, eMMC, etc.)
- Can create scripts for subsequent tasks
  - JTAG programming for booting via TFTP
  - QSPI & eMMC programming
Shell Script Variables

- Set boot method options
  - Many of Avnet’s SOMs and dev boards can boot from more than one source
    ```
    BUILD_BOOT_SD_EXT4_OPTION=yes
    ```

- Specify location of where Vivado & PetaLinux tools are installed (defaults are shown)
  - Needed for setting PetaLinux and Vivado environment variables
    ```
    APP_PETALINUX_INSTALL_PATH=/opt/petalinux-v2018.3-final
    APP_VIVADO_INSTALL_PATH=/opt/Xilinx/Vivado/2018.3
    ```

- Project and board specific variables identify input and output files
  - Identify where to fetch input files
    ```
    HDL_HARDWARE_NAME=ultra96v2_oob_hw
    HDL_PROJECT_NAME=ultra96v2_oob
    HDL_BOARD_NAME=ULTRA96V2
    PETALINUX_ROOTFS_NAME=ultra96v2_oob
    ```
  - Specify output files and locations
    ```
    PLNX_VER=2018_3
    FSBL_PROJECT_NAME=zynqmp_fsbl
    PETALINUX_PROJECT_NAME=ultra96v2_oob_${PLNX_VER}
    ```
Scripted Prerequisites for BSP Build

- Run settings scripts to set tools environment variables
  ```
  source_tools_settings()
  ```

- Build the Vivado hardware platform (if necessary)
  ```
  build_hw_platform()
  ```
Scripted BSP Build

- Build the PetaLinux BSP

  `create_petalinux_bsp()`
  - Create the project
  - Import the hw platform from the Vivado project
  - Configure the project with the git revision controlled config file inputs
    - Project config
    - Devicetree
    - Rootfs
    - u-boot
    - Kernel
  - Build the OS image, rootfs, u-boot, etc.
  - Create the boot.bin for booting
    - Includes the FSBL, ATF, PMU firmware, and u-boot
  - Package the BSP for archiving & sharing
Agenda

- Intro to Ultra96-V2
- Intro to Zynq MPSoC architecture
- Intro to PetaLinux
- Anatomy of a PetaLinux build
- Integrating a custom device driver
- Intro to Vivado and Petalinux build scripts
- Summary
Learn More

- Avnet Ultra96 Technical Training Courses
  https://www.avnet.me/ttc
  http://avnet.me/ttc_on_demand

- Download and install the BSP
  http://avnet.me/Ultra96-V2-2018.3-BSP

- Ultra96-V2 User Guide (and other documentation, etc.)
  http://avnet.me/ultra96-v2

- MicroZed Chronicle – Two Methods of Building PetaLinux for Ultra96
  http://avnet.me/two_methods_plnx_ultra96

- Xilinx Developer Forum 2018 – Linux Application Development with Ultra96
  http://avnet.me/linux_app_dev_ultra96
Helpful Resources

● Avnet git repositories
  ○ bdf  
    https://github.com/Avnet/bdf
  ○ hdl  
    https://github.com/Avnet/hdl
  ○ petalinux  
    https://github.com/Avnet/petalinux

● Avnet VirtualBox VM setup instructions  
  http://avnet.me/vbox-install-guide

● Xilinx UG1144 – PetaLinux Tools Documentation Reference Guide  
  http://avnet.me/xilinx_ug1144

● Xilinx UG1157 – PetaLinux Command Line Reference Guide  
  http://avnet.me/xilinx_ug1157
Thank you

Join Linaro to accelerate deployment of your Arm-based solutions through collaboration.

contactus@linaro.org