Embedded Software Strategy & Development

Presented By

Tony McDowell
System Software & SoC Solutions – Product and Technical Marketing
If Microsoft ever does applications for Linux it means I've won.

-Linus Torvalds, 1998

When software developers drive hardware design it means adaptable SoC’s have won.

-Xilinx, 2018
How Do You Want to Do This?

fetch

build

integrate

package

deploy

GCC

yocto

SDK

PetaLinux
Open and Public Code

- GitHub.com/Xilinx
- Nearly 100 repositories
- All of our embedded software stack
- All of our Yocto recipes
- Scripts for Vivado
- Tutorials and Examples
Staying Up-to-Date

The same for every device family!
Rebase Kernel Tree

> **Merge-Tree**
  > Merges two separate branches into a single new branch going forward
  > Lose the history of what was different between the branches

> **Rebase Tree**
  > Creates a series of patches that can be applied cleanly to the HEAD node
  > Maintain history of development in the separate development paths

> Single upstream kernel version per year
> Rebase patchsets with Vivado releases
> Rolling merge tree
Compilers and Toolchains

> AArch32 – ARMv7 – Zynq-7000
> AArch64 – ARMv8 – Zynq UltraScale+, Versal
> Cortex-R5 – ARMv7 – Zynq UltraScale+, Versal
> MicroBlaze – MMU / Linux Configuration
> MicroBlaze – Microcontroller Configuration

GCC 8 Support in 2019
Enabling Yocto

- `meta-xilinx` – BSP support for Xilinx device families

- `meta-xilinx-tools` – Yocto infrastructure to interface with Xilinx tools

- `meta-petalinux` – Infrastructure to replicate the default PetaLinux root filesystem
Integrating with Yocto

GitHub Sources

- User Configuration
- Metadata (.bb + patches)
- Machine (BSP) Configuration
- Policy Configuration

.meta-xilinx-tools

.meta-petalinux

Openembedded Architecture Workflow

- Upstream Source
- Metadata/Inputs
- Build system
- Output Packages
- Process steps (tasks)
- Output Image Data

Packaging Stages

- Source Fetching
- Patch Application
- Configuration / Compile / Autoreconf as needed
- Output Analysis for package splitting plus package relationships
- .rpm Generation
- .deb Generation
- .ipk Generation

QA Tests

Package Feeds

- Image Generation
- SDK Generation
- Images
- Application Development SDK

Images

© Copyright 2018 Xilinx
Abstracting Yocto

Diagram: Yocto project flowchart showing sources, compilation, and abstraction leading to PetaLinux export and deploy.
Multiprocessing with Xen

- Reducing code Size
- Working toward certifiability
- Dom0-less boot
- Automatic static partitioning
OpenAMP and Interprocessor Comms

- Built on standard remoteproc and rpmsg infrastructure

- Open and public on GitHub.com/OpenAMP
FPGA Manager

$ /sys/class/fpga_manager/fpga0/$
New Community Portal

> Centralized Clearinghouse
  >> References other resources, doesn’t replace them

> Increasing number of developers use Open Source Content
  >> Converge content and make navigation to desired location easier

> Xilinx has lots of Open Source content to filter
  >> GitHub, AWS, Wiki, Ultra96
One more thing...
Decoupling PetaLinux Projects

2019.1
Project A
SW Components A

2019.2
Project A
SW Components A

Interchange

Decoupling Runtime Components

2019.x
rootfs
kernel
U-boot
FSBL
bitstream

Update

2019.x+
rootfs
kernel
U-boot
FSBL
bitstream

Decoupling Linux from Vivado

2019.X+
Vivado
DSA
XSCT
Open Format
PetaLinux

Decoupled from Vivado

2018.3
Vivado
DSA
TCL
PetaLinux

Tightly coupled with Vivado

Decoupling Packages from Each Other

Package Update

Package Repo

Decoupling Runtime Components

Decoupling Linux from Vivado

Decoupling Packages from Each Other