Altera SoC Linux Intro Workshop
Altera SW SoC Workshop Series

- SW Workshop #1 – Altera SoC SW Development Overview
- SW Workshop #2 – Introduction to Linux on Altera SoC
- SW Workshop #3 – Developing Drivers for Altera SoC Linux
Agenda

- Essential Information Resources
- SoC Device Overview
- SoC Physical Address Map
- SoCFPGA Development Flow & Tools
- Altera SoC Linux Overview
- Components of the SoC FPGA Linux BSP
- SoC Linux Upstreaming & Driver Support
- Altera SoC Linux Boot Flow
- Das U-Boot Bootloader
- Linux Device Tree for SoC FPGA
- Take Home Lab
Welcome. Here’s What You Can Expect Today

<table>
<thead>
<tr>
<th>Experienced Linux Developers</th>
<th>New Linux Developers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Find a familiar embedded Linux development flow</td>
<td>An exposure to the components of embedded Linux</td>
</tr>
<tr>
<td>Overview of upstreaming and driver support for mach_SoCFPGA architecture</td>
<td>Essential Linux learning and documentation resources</td>
</tr>
<tr>
<td>Guide to SoCFPGA resources</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hardware Developers</th>
<th>Everyone</th>
</tr>
</thead>
<tbody>
<tr>
<td>HW handoff to Linux build flow</td>
<td>SoC FPGA architecture-specific information</td>
</tr>
<tr>
<td>Boot and FPGA configuration for Linux</td>
<td>SoC FPGA recommendations and best practices</td>
</tr>
<tr>
<td>SW implications of HW architecture</td>
<td></td>
</tr>
</tbody>
</table>

Focused on SoC/Nios Linux Specific Topics
Essential Information Resources

Where to learn more...
...a non-exhaustive list
Linux Foundation Training

Linux Developer classes are designed to help participants:

- Learn how to develop an *embedded Linux* product
- Become familiar with and learn to write *device drivers*
- Get practical experience with the Linux kernel
- Learn how to work with the Linux developer community

**Developer Courses**

- **LFD331** – Developing Linux Device Drivers
- **LFD405** – Building Embedded Linux with the Yocto Project
- **LFD411** – Embedded Linux Development
- **LFD414** – Introduction to Embedded Android Development
- **LFD205** – How to Participate with the Linux Community
- **LFD211** – Introduction to Linux for Developers
- **LFD262** – Developing with Git
- **LFD312** – Developing Applications for Linux
- **LFD320** – Linux Kernel Internals & Debugging
- **LFD415** – Inside Android: An Intro to Android Internals
- **LFD432** – Optimizing Linux Device Drivers for Power Efficiency

[http://training.linuxfoundation.org/linux-courses/development-training](http://training.linuxfoundation.org/linux-courses/development-training)
Linux Documentation Resources

 GIT
 - Distributed revision control system to enable distributed collaboration
 - On-line documentation & training:
   - [http://git-scm.com/doc](http://git-scm.com/doc)
   - [https://training.github.com](https://training.github.com)

 Denx U-Boot Manual
 - Complete documentation from the folks who wrote Das U-Boot

 Free-Electrons:
 - Complete training materials posted free

 Device Tree for Dummies
The Two Best Sources for Linux Development Information

Linux Kernel Documentation

- The most complete and most essential Linux kernel documentation
- Included with the Linux kernel source code

<local GIT repo>/Documentation
The Two Best Sources for Linux Development Information

An open source OS breeds open source information

- Embedded Linux boot time reduction - Free Electrons
  free-electrons.com/services/boot-time/
  Making your embedded Linux systems boot faster: Investigating boot time issues and applying optimization techniques that don't require a redesign.

- Free Electrons: Embedded Linux Experts
  free-electrons.com/
  Embedded Linux, kernel and Android: development, training and consulting services. ... Buildroot commercial support: Embedded Linux boot time reduction ... Offering our broad embedded Linux development experience through our ... 42,295 views; How to boot an uncompressed Linux kernel on ARM - 38,199 views ....
  You've visited this page 4 times. Last visit: 11/1/14

[PDF] Update on boot time reduction techniques, with figures - T...
RocketBoards.org – Altera SoC Linux Community Portal

- The source for SoC FPGA Linux info
  - Golden System Reference Design (GSRD)
  - Updates on latest releases
  - Step-by-step getting started guides

- SoC FPGA Mailing List - RFI
  - Active community participation in answering SoC FPGA and Linux questions

- Example Projects, Applications, and Designs
  - From Altera and the SoC community

- Enables the SoC community to support Linux
RocketBoards.org Resources

Starting point for documentation

Mail Lists & Forum for Community Support

Information on your Development Kit

Link to GitHub Repos

DOCUMENTATION
Find information on boards, flows, and open hardware and software

CODE
Access the latest SoC Linux code from our git repositories

MAIL LISTS
Stay updated with latest news, features, questions and feedback

FORUM
Jump into the forum to get help and offer help

PROJECTS
Check out projects submitted by the community to get inspired

BOARDS
Explore the latest hardware
Welcome to the Documentation Web

<table>
<thead>
<tr>
<th>SoC Boards</th>
<th>GSRD (Golden System Reference Design) Documentation</th>
<th>GSRD Documentation - Arrow SoCKit Edition</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Altera Cyclone V Soc Board</td>
<td>- GHRD (Golden Hardware Reference Design) Overview</td>
<td>- GHRD (Golden Hardware Reference Design) Overview</td>
</tr>
<tr>
<td>- Altera Arria V Soc Board</td>
<td>- Booting Linux Using Prebuilt SD Card Image</td>
<td>- Overview</td>
</tr>
<tr>
<td>- Arrow SoCKit Evaluation Board</td>
<td>- Connecting to Board Web Server</td>
<td>- Booting Linux Using Prebuilt SD Card Image</td>
</tr>
<tr>
<td>- EBV SoCrates Evaluation Board</td>
<td>- Connecting to Board Using SSH</td>
<td>- Connecting to Board Web Server</td>
</tr>
<tr>
<td>- Marconi Hello Soc Evaluation Kit</td>
<td>- Running Sample Linux Applications</td>
<td>- Connecting to Board Using SSH</td>
</tr>
<tr>
<td>- DENX MCV SoM</td>
<td>- Angstrom On SoC FPGA</td>
<td>- Running Sample Linux Applications</td>
</tr>
<tr>
<td>- NOVPEK®Culti</td>
<td>- Using Yocto Source Package</td>
<td>- Using Yocto Source Package</td>
</tr>
<tr>
<td>- Enclustra SA series</td>
<td>- Compiling the Hardware Design</td>
<td>- Compiling the Hardware Design</td>
</tr>
<tr>
<td></td>
<td>- FPGA Programming</td>
<td>- FPGA Programming</td>
</tr>
<tr>
<td></td>
<td>- Generating and Compiling the Preloader</td>
<td>- Generating and Compiling the Preloader</td>
</tr>
<tr>
<td></td>
<td>- Generating the Device Tree</td>
<td>- Generating the Device Tree</td>
</tr>
<tr>
<td></td>
<td>- Using System Console</td>
<td>- Using System Console</td>
</tr>
<tr>
<td></td>
<td>- Using Git Trees</td>
<td>- Using Git Trees</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Preloader &amp; U-Boot</th>
<th>FPGA Programming</th>
<th>Example Designs</th>
</tr>
</thead>
<tbody>
<tr>
<td>- HPS Boot Flow</td>
<td>- FPGA Programming from HPS Software</td>
<td>- Device Wide AMP</td>
</tr>
<tr>
<td>- Preloader and U-Boot Source Code - Files &amp; Folders</td>
<td>- FPGA Programming with QuArtus II Programmer</td>
<td>- Arria V PCIe Root Port with MSI</td>
</tr>
<tr>
<td>- Adding a New Board to Preloader &amp; U-Boot</td>
<td></td>
<td>- Cyclone V PCIe Boot Port with MSI</td>
</tr>
<tr>
<td>- SPI / Serial NOR Flash Layout</td>
<td></td>
<td>- CycloneV SGMII Example Design</td>
</tr>
<tr>
<td>- SDMMC Flash Layout</td>
<td></td>
<td>- Altera SoC Triple Speed Ethernet Design Example</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Cyclone V RMII Example Design</td>
</tr>
</tbody>
</table>
RocketBoards.org – Useful Links

1. Select a Board
   Altera Cyclone V SoC Board

2. Select a Tool Version
   15.1

3. Select a Task
   Select One
   - 1 - Booting Linux Using Prebuilt SD Card Image
   - 2 - Connecting to Board Web Server
   - 3 - Running a Sample Linux Applications
   - 4 - Compiling Hardware Design
   - 5 - Generating and Compiling the Preloader
   - 6 - Generating the Device Tree
   - 7 - Compiling Linux
   - 8 - Creating and Updating SD Card
   - 9 - Programming FPGA
   - 10 - Programming FPGA With Quartus Programmer
   - 11 - Using System Console
   - 12 - Using Angstrom
   - 13 - Booting From FPGA

Device Tree Generator User Guide
   - http://www.rocketboards.org/foswiki/Documentation/GSRD141DeviceTreeGenerator

Programming FPGA from HPS
   - NOTE: the new FPGA manager core framework has recently been up streamed

GSRD Releases
   - http://releases.rocketboards.org

© 2016 Altera—Public
Several Ways to Learn!

**Instructor-led training**
- Face to face with an Altera expert Training Engineer
- 20+ courses to choose from (8 hour classes)

**Virtual classes (taught via WebEX)**
- Can ask questions to Altera expert Training Engineer
- Course content same as instructor-led classes (1/2 day sessions)

**Online training (free and always available)**
- 200+ topics available (~30 minutes in length)

**Videos (free and always available)**
- YouTube videos (~4 minutes each)
SoC Classes Available

- Instructor-led or virtual classes
  - Designing with an ARM-based SoC
  - Developing Software for an ARM-based SoC

- Online classes
  - Hardware Design Flow for an ARM-based SoC
  - Software Design Flow for an ARM-based SoC
  - SoC Hardware Overview: the Microprocessor Unit
  - SoC Hardware Overview: Interconnect and Memory
  - SoC Hardware Overview: System Management, Debug, and General Purpose Peripherals
  - SoC Hardware Overview: Flash Controllers and Interface Protocols
  - SoC Bare-metal Programming and Hardware Libraries
  - Getting Started with Linux for Altera SoCs
Essential SoC Software Tools Online Videos

- ARM DS-5 Altera Edition Toolchain
  - https://youtu.be/HV6NHR6gLx0
- DS-5 Altera Edition: Bare-metal Debug and Trace
  - https://youtu.be/u_xKybPhcHL
- DS-5 Altera Edition: FPGA-adaptive Linux Kernel Debug and Trace
  - https://youtu.be/lrR-SfVZd18
- Debugging Linux applications on the Altera SoC with ARM DS-5
- FPGA-adaptive debug on the Altera SoC using ARM DS-5
  - https://youtu.be/2NBcUrV2TxbI
- Streamline Profiling on Altera SoC FPGA. Part 1 - Setup
  - https://youtu.be/X-k9lmXQTYo
- Streamline Profiling on Altera SoC FPGA. Part 2 - Running Streamline
  - https://youtu.be/Tzbd7qldKqY
Essential SoC Hardware Documentation Resources

Hard Processor System Technical Reference Manuals
- Available in Device Handbooks:
- Contain Functional Descriptions Peripheral
- Contain Control Register Address Map and Definitions
  - These are also available online at the links above in HTML and PDF formats

HPS SoC Boot Guide
- Cyclone V SoC & Arria V SoC: [AN709 - HPS SoC Boot Guide](https://www.altera.com/products/soc/portfolio/cyclone-v-soc/support.html)
- Arria 10 SoC: included in HPS TRM in Arria 10 Device Handbook

ARM Documentation Site
- Documentation available for all ARM IP
  - Cortex-A9 & A53 MP Cores, FPU, NEON, GIC, ARM Peripherals, etc.
- Requires free registration
- Refer to HPS TRM for IP core names and revision information
- [http://infocenter.arm.com/help/index.jsp](http://infocenter.arm.com/help/index.jsp)
Essential SoC Software Documentation Resources

**Altera SoC Embedded Design Software (SoC EDS) Tools**

- User Guide:
- Linux & Baremetal Software Development Tools Overview
- HPS Preloader User Guide
- HPS Flash Programmer User Guide
- SD Card Boot Utility
- Getting Started Guides: Preloader, Linux, Bare Metal, Debug, HW Library
- SoC HPS Release Notes
- SoC Abstraction Layer (SoCAL) API Reference
  - `<SoC EDS install dir>/ip/altera/hps/altera_hps/doc/socal/html/index.html`
- Hardware Manager API Reference
  - `<SoC EDS install dir>/ip/altera/hps/altera_hps/doc/hwmgr/html/index.html`
- GCC Documentation
  - `<SoC EDS install dir>/ds-5/documents/gcc/getting_started.html`
- Bare Metal Compiler
  - `<SoC EDS installation directory>/host_tools/mentor/gnu/arm/baremetal/share/doc/sourceryg++-arm-altera-eabi`
SoC Device Overview
Altera Investment in Embedded Technologies

- Altera established Austin Technology Center (ATC) in 2011
- Altera’s primary embedded engineering center
- Austin provides access to one of the richest embedded processing talent bases in the world
Altera SoC Product Portfolio

### DEVICE AVAILABILITY

**SoC devices available across entire product portfolio ...**
<table>
<thead>
<tr>
<th>ARM Public Processor Offering</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CORTEX-A</strong></td>
</tr>
<tr>
<td>Cortex-A57</td>
</tr>
<tr>
<td>Cortex-A53</td>
</tr>
<tr>
<td>Cortex-A17</td>
</tr>
<tr>
<td>Cortex-A15</td>
</tr>
<tr>
<td>Cortex-A12</td>
</tr>
<tr>
<td>Cortex-A9</td>
</tr>
<tr>
<td>Cortex-A8</td>
</tr>
<tr>
<td>Cortex-A5</td>
</tr>
<tr>
<td><strong>CORTEX-R</strong></td>
</tr>
<tr>
<td>Cortex-R7</td>
</tr>
<tr>
<td>Cortex-R5</td>
</tr>
<tr>
<td>Cortex-R4</td>
</tr>
<tr>
<td><strong>CORTEX-M</strong></td>
</tr>
<tr>
<td>Cortex-M4</td>
</tr>
<tr>
<td>Cortex-M3</td>
</tr>
<tr>
<td>Cortex-M1</td>
</tr>
<tr>
<td>Cortex-M0+</td>
</tr>
<tr>
<td>Cortex-M0</td>
</tr>
<tr>
<td><strong>SECURCORE</strong></td>
</tr>
<tr>
<td>SC000</td>
</tr>
<tr>
<td>SC100</td>
</tr>
<tr>
<td>SC300</td>
</tr>
</tbody>
</table>

- **Stratix 10**: Application Processor
- **Cyclone V, Arria V, Arria 10**: Real-time control
- **Microcontroller**: Secure
28nm SoC System Architecture

**Processor**
- Dual-core ARM® Cortex™-A9 MPCore™ processor
- Up to 5,250 MIPS (1050 MHz per core maximum)
- NEON coprocessor with double-precision FPU
- 32-KB/32-KB L1 caches per core
- 512-KB shared L2 cache

**Multiport SDRAM controller**
- DDR3, DDR3L, DDR2, LPDDR2
- Integrated ECC support

**High-bandwidth on-chip interfaces**
- > 125-Gbps HPS-to-FPGA interface
- > 125-Gbps FPGA-to-SDRAM interface

**Cost- and power-optimized FPGA fabric**
- Lowest power transceivers
- Up to 1,600 GMACS, 300 GFLOPS
- Up to 25Mb on-chip RAM
- More hard intellectual property (IP): PCIe® and memory controllers

Notes:
(1) Integrated direct memory access (DMA)
(2) Integrated ECC
High-Level Block Diagram

- **EMAC (2)**
- **USB OTG (2)**
- **Flash Control**
- **DMA**
- **TMC (Trace)**
- **Debug Port**

**Interconnect**

- **FPGA to FPGA**
- **FPGA to HPS**
- **Configuration Control**
- **FPGA to SDRAM**

**FPGA**

- **ARM Cortex-A9 MPCore**
  - **CPU0**: ARM Cortex-A9 NEON/FPU, 32 KB I$, 32 KB D$
  - **CPU1**: ARM Cortex-A9 NEON/FPU, 32 KB I$, 32 KB D$

**Low Speed Peripherals**

- **Timers, GPIO, UART, SPI, I2C, CAN**

**HPS**

- **Boot ROM**
- **On-chip RAM 64 KB**
- **Multi-port DDR SDRAM Controller**

**Interconnect**

- **ACP**
- **SCU**
- **L2 Cache (512 KB)**
# A Comparison: Cyclone V SoC, Arria V SoC, Arria 10 SoC

<table>
<thead>
<tr>
<th>Metric</th>
<th>Cyclone V SoC</th>
<th>Arria V SoC</th>
<th>Arria 10 SoC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technology</td>
<td>28nm</td>
<td>28nm</td>
<td>20nm</td>
</tr>
<tr>
<td>Processor Performance</td>
<td>925 MHz</td>
<td>1.05 GHz</td>
<td>1.5 GHz</td>
</tr>
<tr>
<td>Total Power Dissipation</td>
<td>100%</td>
<td>100%</td>
<td>60% (40% Lower)</td>
</tr>
<tr>
<td>Max PCI Express Hard IP</td>
<td>Gen 2 x4</td>
<td>Gen 2 x8</td>
<td>Gen 3 x8</td>
</tr>
<tr>
<td>Memory Devices Supported</td>
<td>DDR2, DDR3, DDR3L, LPDDR2</td>
<td>DDR2, DDR3, DDR3L, LPDDR2</td>
<td>DDR4/3, LPDDR2/3, QDRIV, RLDRAM III, Hybrid Memory Cube*</td>
</tr>
<tr>
<td>Max. HPS DDR Data-Width</td>
<td>40-bit (32-bit + ECC)</td>
<td>40-bit (32-bit + ECC)</td>
<td>72-bit (64-bit + ECC)</td>
</tr>
<tr>
<td>EMAC Cores</td>
<td>EMAC x 2</td>
<td>EMAC x 2</td>
<td>EMAC x 3</td>
</tr>
<tr>
<td>NAND Device Supported</td>
<td>8-bit</td>
<td>8-bit</td>
<td>8-bit and 16-bit</td>
</tr>
<tr>
<td>SD/MMC devices supported</td>
<td>SD/SDIO/MMC</td>
<td>SD/SDIO/MMC</td>
<td>SD/SDIO/MMC 4.5 with eMMC</td>
</tr>
<tr>
<td>FPGA Logic Density Range (LEs)</td>
<td>25 - 110K</td>
<td>370 - 450K</td>
<td>160 - 660K</td>
</tr>
<tr>
<td>FPGA Core Performance</td>
<td>260 MHz</td>
<td>307 MHz</td>
<td>500 MHz</td>
</tr>
</tbody>
</table>

* Listed memory protocols are supported by FPGA EMIF interface, the HPS EMIF interface supports a subset of these.
SoC Physical Address Map

Essential HW information for SW Developers
Cyclone V & Arria V SoC HPS Physical Memory Map

- **L3 (Default)**
  - HPS Slaves
  - H2F FPGA Slaves
  - ACP Window
  - SDRAM Region
  - RAM/SDRAM

- **MPU**
  - HPS Slaves
  - H2F FPGA Slaves
  - SDRAM Region
  - Boot Region
  - RAM/ROM/SDRAM

- **FPGA to SDRAM**
  - SDRAM

- Memory Addresses:
  - 0xFF20_0000
  - 0xC000_0000
  - 0x8000_0000
  - 0x0000_0000

- **Remapping**
  - Default remap to 0x0
  - Remaps as RAM & ROM or SDRAM

© 2016 Altera—Public
Arria 10 SoC HPS Physical Memory Map

L3 (Default)

- HPS Slaves
- H2F FPGA Slaves

MPU

- HPS Slaves
- H2F FPGA Slaves

FPGA to SDRAM

- SDRAM

0x0000_0000

- RAM/SDRAM

0x8000_0000

- SDRAM Region

0xC000_0000

- RAM/ROM/SDRAM

0xFF20_0000

- Boot Region

Remaps as RAM & ROM or SDRAM

0x0000_0000

- 0 GB

0xC000_0000

- 2 GB

0xFF20_0000

- 3 GB

4 GB
Arria 10 SoC HPS Physical Memory Map
Physical Address Mapping
Physical Address Mapping – FPGA to SDRAM

- FPGA Masters have access to full 4GB of SDRAM address space
  - Subject to MPFE MPU restrictions
  - No coherency
  - No virtual addressing
MPU has access to the lower 3 GBytes of SDRAM

Kernel manages and can allocate memory in this 3GByte space

- Allocate for both user and kernel space
- Allocatable on 4K Byte Boundaries (page size)
Physical Address Mapping – MPU to FPGA

- MPU can access 960 MBytes of FPGA address space via HPS to FPGA Bridge
- MPU can access 2 MB of FPGA address space via HPS to FPGA Lightweight bridge
- Not allocatable in user space
- Space FPGA peripherals on Linux page size (4KB) boundaries
- Access methods discussed in Developing Linux Drivers for Custom Peripherals Workshop
Physical Address Mapping – FPGA to HPS

- FPGA to HPS (F2H) masters see 4 GByte address space
- F2H bandwidth to SDRAM limited vs. FPGA to SDRAM bridge
Cyclone V SoC Memory Map Example - SDRAM

1 GByte FPGA SDRAM

1 GByte HPS SDRAM
Cyclone V SoC GSRD Memory Map Example - SDRAM

- 960MB available for all FPGA slaves
- MPU cannot directly access full 1GByte of FPGA SDRAM
- Not allocatable by kernel

Default remap to 0x0
Cyclone V SoC Memory Map Example - PIO
Cyclone V SoC Memory Map Example - PIO
Cyclone V SoC GSRD Memory Map Example

Look at led_pio at Address 0x0001_0040
Cyclone V SoC GSRD Memory Map Example

It is connected to 3 Masters.
Cyclone V SoC GSRD Memory Map Example

- Each master sees the slave at a different address
- These addresses are offsets from the HPS bridge address

ARM sees LED through the H2F bridge:
H2F bridge: 0xff20_0000 + LED_PIO base: 0x0001_0040 = 0xff21_0040

ARM sees LED through the LW bridge:
H2F bridge: 0xff20_0000 + LED_PIO base: 0x0001_0040 = 0xff21_0040

JTAG Master sees LED H2F:
LED_PIO base: 0x0001_0040 = 0x0001_0040
SoCFPGA Development Flow & Tools
Why Does This Matter for Linux Development?

- **Altera SoCs offer unique advantages**
  - User specified peripheral set
  - Tightly-coupled MPU & FPGA fabric
  - One of a kind HW/SW debug capabilities

- **Altera SoCs have unique requirements**
  - Understanding of paths to exchange data between MPU and FPGA
  - Building custom BSP for user specified peripheral set
  - Correct handling and configuration of HPS/FPGA bridges

- **Altera SoCs have tools to enable the power of an integrated MPU and FPGA**
Altera SoC Embedded Design Suite

**FPGA Design Flow**

- **Design**
  - Quartus II design software
  - Qsys system integration tool
  - Standard RTL flow
  - Altera and partner IP

- **Simulate**
  - ModelSim, VCS, NCSim, etc.
  - AMBA-AXI and Avalon bus functional models (BFMs)

- **Debug**
  - SignalTap™ II logic analyzer
  - System Console

- **Release**
  - Quartus II Programmer
  - In-system Update

**Software Design Flow**

- **Design**
  - ARM Development Studio 5
  - GNU toolchain
  - OS/BSP: Linux, VxWorks
  - Hardware Libraries
  - Design Examples

- **Simulate**
  - Virtuoso Software Development
  - GNU toolchain

- **Debug**
  - GNU, Lauterbach, DS5

- **Release**
  - Flash Programmer
So... what exactly is Qsys?

GUI based system integration tool for HW system design using IP blocks.

- Simplifies complex system development
- Raises the level of design abstraction
- Provides a standard platform:
  - IP integration
  - Custom IP authoring
  - IP verification
- Enables design re-use
- Scales easily to meet the needs of end product
- Reduces time to market
Qsys System Integration Platform
Linux HW/SW Handoff – Cyclone V SoC and Arria V SoC

Qsys system info, SDRAM calibration files, ID / timestamp, HPS IOCSR data

Hardware

system.iswinfo

system.sopcinfo

board info

Software

Preloader Generator

Device Tree Generator

.c & .h source files (u-boot spl)

Linux Device Tree
Linux HW/SW Handoff – Arria 10 SoC

Hardware Project → Quartus II → Handoff Folder → Board Info

Device Tree Generator: sopc2dts

Boot Loader Generator: bsp-editor

Bootloader DT Source → DTC → Bootloader DT Blob

Regenerate when HW project is recompiled

Regenerate only when user options (boot source, etc.) change

Makefile → Make → U-Boot Binary

U-Boot Source Code → Make

mkpimage

Linux DT Blob

Different DTBs

Board Info

Intermediate File

Output File
Altera SoC Embedded Design Suite

Comprehensive Suite SW Dev Tools

- Hardware / software handoff tools
  - Preloader & Device Tree Generators

- Bare-metal application development
  - SoC Hardware Libraries
  - Bare-metal compiler tools

- FPGA-adaptive debugging
  - ARM DS-5 Altera Edition Toolkit

- Linux application development
  - Yocto Linux build environment
  - Pre-built binaries for Linux / U-Boot
  - Work in conjunction with the Community Portal

- Design examples

- Free Web Edition
- Subscription Edition
- Free 30-day Eval
Industry First: FPGA-Adaptive Debugging

ARM® Development Studio 5 (DS-5™) Altera® Edition Toolkit

- Removes debugging barrier between CPUs and FPGA
- Exclusive OEM agreement between Altera and ARM
- Result of innovation in silicon, software, and business model
- Supports FPGA-Adaptive Linux kernel, driver & application debug
Cross-Domain Debug 1

Trigger from software world to FPGA world
Cross-Domain Debug 2

Trigger from FPGA world to software world

HARDWARE TRIGGER

EXECUTION STOP OR HW TRACE TRIGGER

EXECUTION STOP OR SW TRACE TRIGGER

STOP
Correlate HW and SW Events

Debug event trigger point set from either:

SignalTap™ II Logic Analyzer
or
DS-5 debugger

Captured trace can then be analyzed using timestamp-correlated events

ARM® DS-5™ Toolkit

Timestamp Correlated
## SoC EDS Editions Summary

<table>
<thead>
<tr>
<th>Component</th>
<th>Key Feature</th>
<th>Web Edition</th>
<th>Subscription Edition</th>
<th>30-Day Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware/Software Handoff Tools</td>
<td>Preloader Image Generator</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>Flash Image Creator</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>Device Tree Generator (Linux)</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>ARM DS-5 Altera Edition</td>
<td>Eclipse IDE</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>Debugging over Ethernet (Linux)</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>Debugging over USB-Blaster II JTAG</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>Automatic FPGA Register Views</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>Hardware Cross-triggering</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>CPU/FPGA Event Correlation</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Compiler Tool Chains</td>
<td>Linaro Tool Chain (Linux)</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>CodeBench Lite EABI (Bare-metal)</td>
<td>X</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Hardware Libraries</td>
<td>Bare-metal programming Support</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>SoC Programming Examples</td>
<td>Golden System Reference Design</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>

Everything needed for Linux development is free & open source
Altera SoC Linux Overview
Linux for Altera SoCs

- High Quality Linux Support
- Modern release strategy
- Multiple Kernel Versions
- Community Enablement
Linux Strategy

Kernel
- Same kernel source tree for all SoC’s and NIOS II
  - Same kernel binary for all 32bit SoCFPGA
  - Same kernel binary for all 64bit SoCFPGA
- Device tree support (SoCs and NIOS II)
- Upstream and maintain to kernel.org

U-Boot
- Cyclone V & Arria V SoC currently 2013.01.01
  - Updating to mainline release at Altera 16.0 release
- Arria 10 SoC currently 2014.10
  - Updating to mainline release at Altera 16.1 release
- Same U-Boot source tree for all SoC’s and NIOS II

Toolchain
- Standard, un-patched Linaro Toolchain:
  gcc-linaro-arm-linux-gnueabihf-4.9-2014.09
Linux Strategy - Build Systems

Offer support for Angstrom for SoC
- Embedded Linux distribution, Yocto Project configuration, package manager (OPKG)
- Angstrom v2015.12 / Yocto 2.0
  - via kraj/meta-altera opensource meta-altera layer
  - Linaro GCC 4.9 for all kernels
  - Linaro GCC 5.2 for 4.1 LTSI kernel

Yocto Project Support for SoC
- SoCFPGA layer (meta-altera) upstreamed to Angstrom
- Yocto Project v2.0 ready

Buildroot for both SoC and Nios II
- Roll your own from relevant source
Build System Resources

- Angstrom flow for SoC
  
  http://rocketboards.org/foswiki/view/Documentation/AngstromOnSoCFPGA

- Yocto flow for SoC
  
  http://www.rocketboards.org/foswiki/Documentation/YoctoDoraBuildWithMetaAltera
  - Source Poky from the Yocto Project website
    
    git://git.yoctoproject.org/poky.git

- Buildroot flow for Nios II
  

- Buildroot flow for SoC
  
  http://www.rocketboards.org/foswiki/Documentation/BuildrootForSoCFPGA
Angstrom/Yocto Information Resources

- Yocto Project
  https://www.yoctoproject.org/documentation

- Angstrom Distribution
  http://www.angstrom-distribution.org/

- Open Embedded
  http://www.openembedded.org/wiki/Main_Page

- Building Linux w/ Yocto Linux Foundation class
  LFD405 – Building Embedded Linux with the Yocto Project
Altera SOC Linux Provides Customers Kernel Choices

Linux for Altera SoCs supports the latest stable kernel

Linux for Altera SoCs Linux also supports latest LTSI kernel

LTSI kernel available with or without Preempt Real Time Patches
Status of the Linux kernel for SoC FPGA

Current versions
- Latest stable: refer to linux-socfpga git repo tags: rel_socfpga-x.x…
  - We keep up with Linus Torvalds releases
  - Latest mainline kernel is currently 4.4

- LTSI v4.1
  - Maintained until next LTSI is released

- Real-Time: 4.1-ltsi-rt
  - LTSI kernel with PREEMP_RT patches

All branches are kept in sync
- Bug fixes
- New features
- No changes of API in the LTSI branches
Wind River Linux

- Wind River Linux version 8
- Linux SMP Kernel version 4.1 (LTSI)
  - Real Time patches & Carrier Grade Linux available
- Yocto 2.0 project user space
- Bitbake build system
- WR Workbench Tools
- GNU toolchain 5.2
- Available now from Wind River
- Technical Support:
  - www.windriver.com
  - support@windriver.com
Monta Vista Linux

- Monta Vista Linux CGE7
  - Carrier Grade Edition
- Linux SMP Kernel version 3.10 (LTSI)
- Yocto project user space
- Available from Monta Vista
- Technical Support:
  - www.mvista.com
# Latest Stable Kernel vs. LTSI Kernel

<table>
<thead>
<tr>
<th>Latest Stable Kernel</th>
<th>LTSI Kernel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access to latest kernel features</td>
<td>LTSI kernel versions supported for 2 years</td>
</tr>
<tr>
<td>New features and drivers often released only to the latest kernel version</td>
<td>Critical bug fixes, priority features, &amp; new device or driver support back-ported to LTSI kernel by the community</td>
</tr>
<tr>
<td>Significant investment in kernel maintenance:</td>
<td>Reduces investment in kernel maintenance</td>
</tr>
<tr>
<td>- Back porting features, bug fixes, device support, &amp; new drivers or…</td>
<td>- Features, bug fixes, device support, &amp; new drivers ported by the community</td>
</tr>
<tr>
<td>- Constant upgrades to latest stable kernel</td>
<td></td>
</tr>
</tbody>
</table>
LTSI – Economic value

Economic Value of LTSI?

- Cost of Back porting security and bug fixes are 3M$ per year per version
- Cost of maintaining in-house patch is 288K$ in case of LTSI 3.4
Linux Code Quality

Altera’s internal development process is similar to the community’s
- Code/Peer reviews
- Code style checked
- Copyright, licenses, etc checked
- All checks enforced

Daily Builds
- Automated builds run daily
- Complete system: boot loader, kernel, Angstrom
- SD card image produced

Daily Tests
- Linaro’s LAVA is used
- All kernel branches tested
Linux Code Tests

Objectives
- Daily test of the supported Linux kernel branches
- Provide feedback to developers

Infrastructure
- Linaro’s LAVA is used
  - Linaro Automated Validation Architecture
  - Runs our unit tests and log results
- Tests start automatically after each build is complete
Altera SoC Linux Support Model

Rocketboards.org
- SoC & Nios II Linux documentation
- SoC & Nios II SoC Linux reference & example designs

Rocketboards.org RFI & Linux Community
- Kernel/RFS/u-boot questions
- SoC/Nios II subsystem and driver questions

Altera.com and Rocketboards.org
- SoCEDS & Quartus/QSys documentation and questions
- SoC Preloader questions
- SoC HPS implementation specific questions
- Use myAltera for service requests

Support from Altera is focused on SoCFPGA and NiosII
Linux Board Support Package

Altera enables Linux community development on SoCFPGA
& Nios II
Break
Components of the SoC FPGA Linux BSP
Building a Custom Embedded Linux Distribution

How do I get from here... …to here?

- Altera provides a Linux BSP… …not a Linux distribution.
- The BSP enables the creation of a custom distribution
SoC Linux Board Support Package

- U-Boot
- Device Tree
- SoC Machine Specific Layer – mach-socfpga
- Drivers for SoC and board components
- Kernel
- Root File System

SoC Linux BSP release provides all of the components in fully non-proprietary source code form
SoC Linux Board Support Package

Example configuration to enable evaluation and initial development

Kernel configuration to enable evaluation and initial development

Up-streamed and community supported drivers

Up-streamed mach-socfpga architecture

Board-specific layer which enables common kernel binary

Open-source, community supported boot loader

Development kit or custom board

User Space
(debug, compiler, shell, etc)

Root File System

Kernel

Drivers

Machine Specific Layer

Device Tree

U-Boot

SoC

Board
Cyclone V SoC Development Kit

Everything you need to begin Linux development
- SoC Development Kit
- Golden System Reference Design
  - A Linux distribution for the dev. kit

Features:
- 1 user license for ARM DS-5 Altera Edition Toolkit
- Ethernet, USB, CAN, UART
- DDR3 (HPS and FPGA), SD Card, QSPI
- PCIe (rootport & endpoint)
- Expansion header
- Much more . . .

Linux GSRD for Development Kits

- Boot Linux from an SD card
  - Updated images on: [http://releases.rocketboards.org/](http://releases.rocketboards.org/)
    - Choose a release date folder, the “gsrd” folder, then the “bin” folder. Ex: [http://releases.rocketboards.org/release/2014.12/gsrd/bin](http://releases.rocketboards.org/release/2014.12/gsrd/bin)
  - Latest LTSI kernel
  - Angstrom Linux distribution for SoC
    - Package manager to load packages from Angstrom’s on-line package feed
    - Add whatever tools are needed for evaluation: gstreamer, usb-utils, etc…

- GSRD contents
  - Complete HW reference design w/ FPGA programming file
  - Bootable SD card image & component binaries
  - Tagged for rebuilding in angstrom-socfpga git repository
    - ACDSX.X_REL_GSRD_PR

A complete SD card ships with the board
- Take it out and stick it in a drawer
- Based on out-of-date 3.9 kernel
## Multiple Dev. Kit Options w/ Linux BSP

<table>
<thead>
<tr>
<th>Kit</th>
<th>Vendor</th>
<th>Family</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arria 10 SoC Dev. Kit</td>
<td>Altera</td>
<td>Arria 10 SoC</td>
</tr>
<tr>
<td>Arria V SoC Dev. Kit</td>
<td>Altera</td>
<td>Arria V SoC</td>
</tr>
<tr>
<td>Cyclone V SoC Dev. Kit</td>
<td>Altera</td>
<td>Cyclone V SoC</td>
</tr>
<tr>
<td>Atlas Board</td>
<td>Altera</td>
<td>Cyclone V SoC</td>
</tr>
<tr>
<td>SoCKit</td>
<td>Arrow</td>
<td>Cyclone V SoC</td>
</tr>
<tr>
<td>Helio SoC Eval. Platform</td>
<td>Macnica</td>
<td>Cyclone V SoC</td>
</tr>
<tr>
<td>SoCrates</td>
<td>EBV Elektronik</td>
<td>Cyclone V SoC</td>
</tr>
</tbody>
</table>
SoC Linux Up-streaming & Driver Support
Maintaining and Up-streaming

- Altera awarded maintainership for the ‘SoC FPGA’ architecture
  - Kernel (arch/arm/mach-socfpga)
  - U-Boot (altera/socfpga_cyclone5)

- Being a maintainer means
  - We upstream the SoC related code
  - We control the changes against the SoC code requested by the community
  - See kernel.org and git.denx.de

- Other community contributions
  - Device Tree Generator: sopc2dts
  - Yocto meta-altera layer
Public git repos for SoCFPGA

https://github.com/altera-opensource
## SoCFPGA Linux Code Repositories on GitHub

<table>
<thead>
<tr>
<th>Repository</th>
<th>Description</th>
</tr>
</thead>
</table>
| linux-socfpga.git/                 | SoCFPGA Linux development repository  
• Mirrors kernel.org linux repo releases
• Downstream branches for socfpga specific patches and updates |
| meta-altera.git/                   | Repository for Yocto recipes for SoCFPGA  
• Starting point for custom Yocto recipes |
| angstrom-socfpga.git/              | Setup scripts for SoCFPGA Angstrom distribution |
| uboot-socfpga.git/                 | SoCFPGA u-boot development repository |
| sopc2dts.git/                      | Device Tree Generator (sopc2dts) repository |
| linux-refdesigns.git/              | SW source code for Linux reference designs |

* Sourced from: [github.com/altera-opensource/](https://github.com/altera-opensource/)*
Kernel Release Cycle – Merge & Bug Fix

Long Term Stable Tree

Mainline Kernel Development Tree

Community Accepted Code

various development repositories

v3.17

v3.18

Merge Window

Stabilization

merge & bug fix

3.17.1

3.18.1

3.17.8

3.18.2

Kernel.org mainline repository

kernel.org linux-stable repo

2 Weeks

~12 Weeks

3.17

3.18

Community Accepted Code

New Features

No New Features

Bug Fixes Only

New Features

No New Features

Bug Fixes Only

various development repositories

© 2016 Altera—Public

now part of Intel
Kernel Release Cycle – Merge & Bug Fix

Long Term Stable Tree

Mainline Kernel Development Tree

Altera up streams with the community

Altera moves with the community to the latest stable kernel

kernel.org linux-stable repo

kernel.org mainline repository

github.com/altera-opensource/linux-socfpga development repository
Altera BSP Kernel Development

Long Term Stable Tree

Mainline Kernel Development Tree

Altera BSP Kernel Dev
Major Release & Frequent Updates

2 Weeks

~12 Weeks

kernel.org linux-stable repo

v3.17

3.17.1

branch: socfpga-v3.17
tag: rel_socfpga-3.17_14.11.01

New Features and all Bug Fixes & updates
tag: rel_socfpga-3.17_14.11.01

v3.18

3.18.1

branch: socfpga-v3.18
tag: rel_socfpga-3.18_15.04.01

New Features and all Bug Fixes & updates
tag: rel_socfpga-3.18_15.04.01

github.com/altera-opensource/linux-socfpga repository
Altera LTSI Kernel Development

Long Term Stable Tree

3.10.31

kernel.org linux-stable repo

LTS Industry Tree

3.10.31

kernel.org ltsi-kernel repository

LTSI patches

Altera BSP
Kernel Dev
Major Release & Frequent Updates

branch: socfpga-v3.10-ltsi

tag: rel_socfpga-3.10-ltsi_14.02.03

tag: rel_socfpga-3.10-ltsi YY.MM.WW

Latest tested release

github.com/altera-opensource/linux-socfpga repository
Altera LTSI Kernel Development

Long Term Stable Tree

LTS Industry Tree

Altera BSP
Kernel Dev
Major Release & Frequent Updates

branch: socfpga-v3.10-ltsi

tag: ACDS14.0.1_REL_GSRD_PR

tag: ACDS14.1_REL_GSRD_PR

Kernel tags for GSRD releases on Rocketboards

github.com/altera-opensource/linux-socfpga repository
SoC Hard IP Driver Support

- Altera SoC HPS built mainly with off the shelf Hard IP components
  - ARM
  - Synopsys DesignWare
  - Cadence

- Wide-spread usage and community support result in high-quality drivers

- Altera is actively contributing features, updates, and fixes to the community
# Linux Driver Support for HPS Peripherals

<table>
<thead>
<tr>
<th>Driver</th>
<th>Kernel Source Tree Location</th>
<th>Maintainer</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPI</td>
<td>drivers/spi/spidev.c</td>
<td>Community</td>
</tr>
<tr>
<td></td>
<td>drivers/spi/spi-dw.c</td>
<td></td>
</tr>
<tr>
<td></td>
<td>drivers/spi/spi-dw-mmio.c</td>
<td></td>
</tr>
<tr>
<td>CAN</td>
<td>drivers/net/can/c_can/c_can_platform.c</td>
<td>Community</td>
</tr>
<tr>
<td>Ethernet</td>
<td>drivers/net/ethernet/stmicro/stmmac/stmmac_platform.c</td>
<td>Community</td>
</tr>
<tr>
<td>NAND</td>
<td>mtd/nand/denali_dt.c</td>
<td>Community</td>
</tr>
<tr>
<td>I2C</td>
<td>drivers/i2c/busses/i2c-designware-platdrv.c</td>
<td>Community</td>
</tr>
<tr>
<td>USB</td>
<td>drivers/usb/dwc2/</td>
<td>Community</td>
</tr>
<tr>
<td>USB PHY</td>
<td>usb/phy/phy-generic.c</td>
<td>Community</td>
</tr>
<tr>
<td>SDMMC</td>
<td>drivers/mmc/host/dw_mmc-pltfm.c</td>
<td>Community</td>
</tr>
<tr>
<td>Timer</td>
<td>drivers/clocksource/dw_apb_timer_of.c</td>
<td>Community</td>
</tr>
</tbody>
</table>
## Linux Driver Support for HPS Peripherals

<table>
<thead>
<tr>
<th>Driver</th>
<th>Kernel Source Tree Location</th>
<th>Maintainer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Watchdog</td>
<td>drivers/watchdog/dw_wdt.c</td>
<td>Community</td>
</tr>
<tr>
<td>PL330 DMA</td>
<td>dma/pl330.c</td>
<td>Community</td>
</tr>
<tr>
<td>GIC</td>
<td>drivers/irqchip/irq-gic.c</td>
<td>Community</td>
</tr>
<tr>
<td>GPIO</td>
<td>drivers/gpio/gpio-dwapb.c</td>
<td>Community</td>
</tr>
<tr>
<td>Timer</td>
<td>drivers/clocksource/dw_apb_timer_of.c</td>
<td>Community</td>
</tr>
<tr>
<td>UART</td>
<td>drivers/tty/serial/8250/8250_dw.c</td>
<td>Community</td>
</tr>
<tr>
<td>QSPI</td>
<td>spi/spi-cadence-qspi.c</td>
<td>Altera</td>
</tr>
<tr>
<td>Clock Manager</td>
<td>drivers/clk/socfpga/clk.c</td>
<td>Altera</td>
</tr>
<tr>
<td>FPGA Manager</td>
<td>drivers/fpga/fpga-mgrs/altera.c</td>
<td>Altera</td>
</tr>
<tr>
<td>FPGA Bridges</td>
<td>drivers/misc/fpga-bridge/</td>
<td>Altera</td>
</tr>
<tr>
<td>EDAC (ECC)</td>
<td>drivers/edac/altera_*</td>
<td>Altera</td>
</tr>
</tbody>
</table>
## Linux Driver Support for soft Peripherals

<table>
<thead>
<tr>
<th>Driver</th>
<th>Kernel Source Tree Location</th>
<th>Maintainer</th>
</tr>
</thead>
<tbody>
<tr>
<td>TSE Ethernet</td>
<td>drivers/net/ethernet/altera</td>
<td>Altera</td>
</tr>
<tr>
<td>PCIe Root Port</td>
<td><a href="http://www.rocketboards.org">www.rocketboards.org</a></td>
<td>Altera</td>
</tr>
<tr>
<td></td>
<td><a href="http://www.rocketboards.org">www.rocketboards.org</a> – w/o MSI</td>
<td></td>
</tr>
<tr>
<td>Frame Buffer</td>
<td>video/altvipfb.c</td>
<td>Community</td>
</tr>
<tr>
<td>Avalon SPI</td>
<td>spi/spi-altera.c</td>
<td>Community</td>
</tr>
<tr>
<td>Avalon UART</td>
<td>tty/serial/altera_uart.c</td>
<td>Community</td>
</tr>
<tr>
<td>JTAG UART</td>
<td>tty/serial/altera_jtaguart.c</td>
<td>Community</td>
</tr>
<tr>
<td>QSYS Sys ID</td>
<td>misc/altera_sysid.c</td>
<td>Altera</td>
</tr>
<tr>
<td>Mailbox</td>
<td>drivers/mailbox/mailbox-altera.c</td>
<td>Altera</td>
</tr>
<tr>
<td>Altera 16550 UART</td>
<td>drivers/tty/serial/8250/8250_core.c</td>
<td>Community</td>
</tr>
<tr>
<td>Avalon PIO</td>
<td>drivers/gpio/gpio-altera.c</td>
<td>Altera</td>
</tr>
</tbody>
</table>
Altera SoC Linux Boot Flow
Altera SoC FPGA Configuration Options

Configuration Sources
- PCIe
- QSPI /SPI
- Passive Serial
- Passive Parallel
- User Specified I/F

Boot Sources
- Configuration Sources
- Boot code (RAM/ROM)
- On-chip RAM
- Boot ROM
- NAND Flash
- MMC /SD
- QSPI /SPI

Boot Source

SOC Device

FPGA

HPS

CPU

AXI
SoCFPGA Linux Boot Flow – Cyclone V & Arria V SoC

**Reset**
- Starts Running code at reset exception address
- Normal operation, BootROM is mapped to reset address

**Boot ROM**
- Hardcoded by Altera into device
- Read Boot source from BSEL pins
- Setup minimal configuration to read flash
- Load Preloader from Flash or execute from FPGA
- Jumps to Preloader

**PreLoader**
- U-Boot SPL
- Setup HPS IOs and pinmuxing
- Setup PLLs and clocking
- Initialize SDRAM
- Load subsequent stage from Flash into SDRAM
- Jump to subsequent stage (typically U-Boot)

**U-boot**
- Load Linux

**Linux**
- Starts Running code at reset exception address
- Normal operation, BootROM is mapped to reset address
Preloader Overview – Cyclone V & Arria V SoC

- Loaded by Boot ROM
  - From flash and executed from on-chip RAM
  - or run directly from FPGA
- Uses U-boot Secondary Program Loader (SPL)
  - Open source, GPL Licensed
- Loads U-boot into RAM and jumps to U-boot
- Always regenerate and recompile Preloader when
  - QSys system or HPS configuration changes
  - Quartus/QSys version changes

*Not regenerating and recompiling the Preloader is the single most common source of SoC SW problems!*

- Covered in detail in SoCSDS User Guide and *Designing Software for ARM-Base SoC* training class
SoCFPGA Linux Boot Flow – Arria 10

1. **Reset**
   - Hardcoded by Altera into device
   - Setup minimal configuration to read flash
   - Load Preloader from Flash in On-Chip memory (skipped if booting from FPGA)
   - Jumps to Preloader

2. **Boot ROM**
   - Starts Running code at reset exception address
   - Normal operation, BootROM is mapped to reset address

3. **U-boot**
   - Setup IOCSRs and pinmuxing
   - Setup PLLs and clocking
   - Initialize SDRAM
   - Load Linux

4. **Linux**
   - Starts Running code at reset exception address
   - Normal operation, BootROM is mapped to reset address
Booting from SD/eMMC – GSRD Flow - Simplified

**Reset**
- Reads MBR from SD/eMMC
- Locates custom Altera raw partition 1 type: 0xA2
- Checksums & loads Preloader image 0 from partition start address
- On fail, loads next preloader image

**Boot ROM**
- Checksums & loads U-Boot from raw A2 partition 1

**PreLoader**
- Loads kernel from FAT32 partition 2
- Boot script:
  - Configures FPGA from image on FAT32 partition 2
  - Enables HPS/FPGA Bridges
  - Loads dtb & boots kernel

**U-boot**
- Boots and mounts root file system from EXT partition 3

**Linux**

Provides ONE example of an SD/eMMC Linux boot flow
# GSRD SD Card Image for Cyclone V & Arria V

<table>
<thead>
<tr>
<th>Location</th>
<th>File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Partition 1</td>
<td>socfpga.dtb</td>
<td>Device Tree Blob</td>
</tr>
<tr>
<td>(FAT32)</td>
<td>soc_system.rbf</td>
<td>FPGA configuration file</td>
</tr>
<tr>
<td></td>
<td>u-boot.scr</td>
<td>U-Boot script: configures FPGA and loads kernel</td>
</tr>
<tr>
<td></td>
<td>zImage</td>
<td>Compressed Linux kernel image file</td>
</tr>
<tr>
<td>Partition 2</td>
<td>Various</td>
<td>Linux root file system</td>
</tr>
<tr>
<td>(EXT3)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Partition 3</td>
<td>n/a</td>
<td>Preloader image(s)</td>
</tr>
<tr>
<td>(A2 raw)</td>
<td>n/a</td>
<td>U-Boot image</td>
</tr>
</tbody>
</table>

- **Partition 1 (FAT32)**: Contains device tree blob, FPGA configuration file, U-Boot script, and compressed Linux kernel image file.
- **Partition 2 (EXT3)**: Contains various Linux root file system.
- **Partition 3 (A2 raw)**: Contains preloader image(s) and U-Boot image.
# GSRD SD Card Image for Arria 10

<table>
<thead>
<tr>
<th>Location</th>
<th>File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Partition 3</td>
<td>socfpga.dtb</td>
<td>Device Tree Blob</td>
</tr>
<tr>
<td>Type=A2 (raw)</td>
<td>soc_system.rbf</td>
<td>FPGA configuration file</td>
</tr>
<tr>
<td>Partition 2</td>
<td>u-boot.scr</td>
<td>U-Boot script: not currently used</td>
</tr>
<tr>
<td>Type=83 (EXT Linux)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Partition 1</td>
<td>zImage</td>
<td>Compressed Linux kernel image file</td>
</tr>
<tr>
<td>Type=B (FAT32 Windows)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Partition 2</td>
<td>Various</td>
<td>Linux root file system</td>
</tr>
<tr>
<td>Type=EXT3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Partition 3</td>
<td>n/a</td>
<td>U-Boot image</td>
</tr>
<tr>
<td>Type=A2 raw</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Creating SD Card Images

Create using Altera provided script
- See “tools” folder under GSRD release folders
  http://releases.rocketboards.org/
- Builds complete SD card image which can be directly copied

Use pre-built images
- In “bin” folder under GSRD release folders
  http://releases.rocketboards.org/
- <SoCEDS install directory>/examples/software
- Can be directly copied to SD card

Described in “Creating and Updating SD Card” section of GSRD User Manual
http://www.rocketboards.org/foswiki/Documentation/GSRD
FPGA Configuration – Cyclone V & Arria V SoC

- **Reset**
  - When FPGA is configured first, the HPS is held in reset until the FPGA releases the HPS reset

- **Boot ROM**

- **PreLoader**
  - Preloader can configure the FPGA
  - Currently requires boot from QSPI

- **Boot Loader**
  - U-boot can configure the FPGA - recommended for Linux
  - U-boot supports file systems for FPGA configuration data
  - All peripherals available at OS boot

- **Linux**
  - Linux can configure the FPGA, but tricky
  - Not all peripherals are available at boot
  - 28nm F2S Bridge Errata can lock the device

- **Application**
FPGA Configuration – Arria 10 SoC

- When FPGA is configured first, the HPS is held in reset until the FPGA releases the HPS reset

- U-boot can configure the FPGA
- U-boot supports file systems for FPGA configuration data
- All peripherals available at OS boot

- Partial reconfiguration from Linux: Future Quartus Release
Das U-Boot Bootloader
What is u-boot?

- Common embedded bootloader
  - Command line interface w/ decent help and lots of hardware support
  - Capable board bring up tool
  - Driver support for a wide variety of essential peripherals
- Loads the Device Tree and modifies Device Tree configuration at run-time
- Loads the kernel and passes boot arguments
  - From local file system, over network, or over a serial link
- Open-source & GPL licensed
  - [http://www.denx.de/wiki/U-Boot](http://www.denx.de/wiki/U-Boot)
U-Boot for SoCFPGA

- Sourced from GitHub Altera Opensource or SoCEDS
  
  https://github.com/altera-opensource/u-boot-socfpga.git
  
  <SoC EDS install dir>/examples/hardware/</*ghrd>/.../uboot-socfpga

- Supported u-boot versions:
  - Cyclone V & Arria V SoC currently 2013.01.01
    - Updating to mainline release at Altera 16.0 release
  - Arria 10 SoC currently 2014.10
    - Updating to mainline release at Altera 16.1 release

- u-boot-socfpga branch & tag convention similar to linux-socfpga
U-Boot for SoCFPGA

- SoCFPGA-specific u-boot documentation
  <u-boot-socfpga repo>/doc/README.SOCFPGA

- u-boot controls SDRAM size passed to kernel
  - u-boot SDRAM sizing algorithm overwrites device tree SDRAM memory node entry
  - Device tree entry overrides boot arguments

- SoCFPGA u-boot environment variables
  - Pass HPS/FPGA bridge status and configuration info from preloader
  - Enable FPGA programming from preloader
SoCFPGA HPS Peripherals Supported in U-boot

Peripheral support in u-boot-socfga 2013.01.01

- MPU Subsystem
- Cache/MMU
  - Cache Mgmt
  - MMU Mgmt
- Interrupt Ctrl
- FPGA Manager
- Clock Manager
- Reset Manager
- System Manager
- SDRAM Ctrl
- Interrupt Ctrl
- Pin I/O Cnf Mgmt
- ECC Mgmt
- Ethernet MAC
- Flash Memory
  - QSPI
  - NAND
  - SD/MMC
- Serial
  - UART
- System Manager
- Flash Memory
- Cache/MMU
- MPU Subsystem
- Interrupt Ctrl
- Pin I/O Cnf Mgmt
- ECC Mgmt
- Ethernet MAC
- Flash Memory
  - QSPI
  - NAND
  - SD/MMC
- Serial
  - UART
- System Manager
- Flash Memory
- Cache/MMU
- MPU Subsystem
- Interrupt Ctrl
- Pin I/O Cnf Mgmt
- ECC Mgmt
- Ethernet MAC
- Flash Memory
  - QSPI
  - NAND
  - SD/MMC
- Serial
  - UART
- System Manager
- Flash Memory
- Cache/MMU
- MPU Subsystem
- Interrupt Ctrl
- Pin I/O Cnf Mgmt
- ECC Mgmt
- Ethernet MAC
- Flash Memory
  - QSPI
  - NAND
  - SD/MMC
- Serial
  - UART
- System Manager
- Flash Memory
- Cache/MMU
- MPU Subsystem
- Interrupt Ctrl
- Pin I/O Cnf Mgmt
- ECC Mgmt
- Ethernet MAC
- Flash Memory
  - QSPI
  - NAND
  - SD/MMC
- Serial
  - UART
- System Manager
- Flash Memory
- Cache/MMU
- MPU Subsystem
- Interrupt Ctrl
- Pin I/O Cnf Mgmt
- ECC Mgmt
- Ethernet MAC
- Flash Memory
  - QSPI
  - NAND
  - SD/MMC
- Serial
  - UART
- System Manager
- Flash Memory
- Cache/MMU
- MPU Subsystem
- Interrupt Ctrl
- Pin I/O Cnf Mgmt
- ECC Mgmt
- Ethernet MAC
- Flash Memory
  - QSPI
  - NAND
  - SD/MMC
- Serial
  - UART
- System Manager
- Flash Memory
- Cache/MMU
- MPU Subsystem
- Interrupt Ctrl
- Pin I/O Cnf Mgmt
- ECC Mgmt
- Ethernet MAC
- Flash Memory
  - QSPI
  - NAND
  - SD/MMC
- Serial
  - UART
- System Manager
- Flash Memory
- Cache/MMU
- MPU Subsystem
- Interrupt Ctrl
- Pin I/O Cnf Mgmt
- ECC Mgmt
- Ethernet MAC
- Flash Memory
  - QSPI
  - NAND
  - SD/MMC
- Serial
  - UART
- System Manager
- Flash Memory
Useful U-boot Commands and Variables

- **Write memory location**
  
mw <address> <size> <optional count>
  
mw 0xC0000000 0x10 0x6

- **Read memory location**
  
md<optional .b .w .l> <address> <optional size>
  
md.w 0xC0000000 0x2

- **Control u-boot auto-boot**
  
  - Boot without delay:
    
    setenv bootdelay 0
  
  - Disable auto-boot (stop at u-boot command line):
    
    setenv bootdelay -1

- **Save environment variables to flash**
  
saveenv
Linux Device Tree for SoC FPGA
What’s a Linux Device Tree?

- A tree-like data structure for describing hardware in embedded systems
- Enables device drivers to be linked to Linux kernel at run-time
  - No Linux kernel recompile required
  - Drivers loaded dynamically after loading Device Tree
- Driver-specific Device Tree bindings are documented in the kernel documentation
  - Documentation/devicetree/bindings
- See Device Tree Generator User Guide
  - See link in GSRD User Guide
SoC FPGA Device Tree Bindings Example

Specify base address and size

Specify driver

Specify interrupts

Specify clock sources

Enable peripheral

Driver specific bindings

Load and configure drivers for sub-nodes
Device Trees for a Configurable Peripheral Set

- Typically developers build Linux for fixed form chips
- How do you build Linux for an FPGA fabric that changes?

You use the **Device Tree Generator**

- Device tree information can be added to custom Qsys peripherals and custom drivers
- Device Tree Source can be edited by hand
How to use Altera SoC Device Tree Generator

- .sopcinfo file describes HPS and FPGA system
- Board info file describes external devices on board
- Device Tree Generator creates a plain text representation of the device tree – called the Device Tree Source (DTS)
  - Device Tree Generator: SoCEDS sopc2dts tool
  - sopc2dts distributed as part or SoCEDS or on RBO GIT repo
- Compile the text into a binary representation called the Device Tree Blob (DTB)
- Optionally directly generate DTB from sopc2dts
Board Info File

- XML file required as an input to sop2dts
- Specifies information of which QSys isn’t aware
  - SPI or I2C timing or external peripherals and their properties
  - External flash (QSPI, SPI, or NAND) properties (organization, specs, etc)
- Board specific Ethernet & PHY information
- Allows the developer to disable peripherals which may be enabled in preloader or FPGA HW
Take Home Lab
Workshop 2 Lab Overview

Goal:
- Familiarize you with SoC FPGA Linux components and where to obtain them

Overview
- You will build the SoC-specific pieces of a Cyclone V or Arria V SoC Linux distribution and run it on your dev. kit
- This flow builds each component discretely without a build system. It does not use the optional Yocto or Angstrom based build systems.
- It configures the distribution in a way which works for this lab, which may differ from your actual system requirements
What You’ll Need

- A supported Dev. Kit
  - Altera Atlas Board
  - Altera Cyclone V SoC Board
  - Altera Arria V SoC Board
  - Arrow SoCKIT

- microSD Card

- Linux machine native or VM
  - 4GByte RAM minimum

- Serial Terminal Application
Obtaining Lab Files and Instructions

Posted on RocketBoards
- Link will be emailed out after class

WS2 Linux Kernel Introduction for Altera SoC Devices
This is the first of a series of workshops, to help users become familiar with software development for the Altera SoC family of parts.

Introduction
Link to Workshop Slides
Links to Lab Sections
Set Up for Labs
Install Required Tools
Obtain Development Board
Obtain SD card Image
Verify that the SD card is properly programmed and that your board and host PC are properly configured
Copy the Lab Materials from SD Card

Introduction
This workshop will take you through the manual steps of building the SoC FPGA specific pieces of a custom embedded Linux distribution. It is intended to familiarize you with the resources necessary to build an embedded Linux distribution for your own custom SoC FPGA based board.

This workshop is not an introduction to or training on Embedded Linux. It is only intended to be an overview of the SoC FPGA specific components of a custom Linux...
What You Will Accomplish

- Generate and build the preloader
- Generate and compile the device tree
- Obtain and build u-boot
- Obtain, configure, and build the kernel
- Program to an SD card
- Run a Simple Linux App
  - Verifies the lab was completed
- Submit results & feedback
Thank You