Department of Electrical & Computer Engineering
EE8205 Embedded Computer Systems

Ryerson University

Porting Android to DE1 SoC

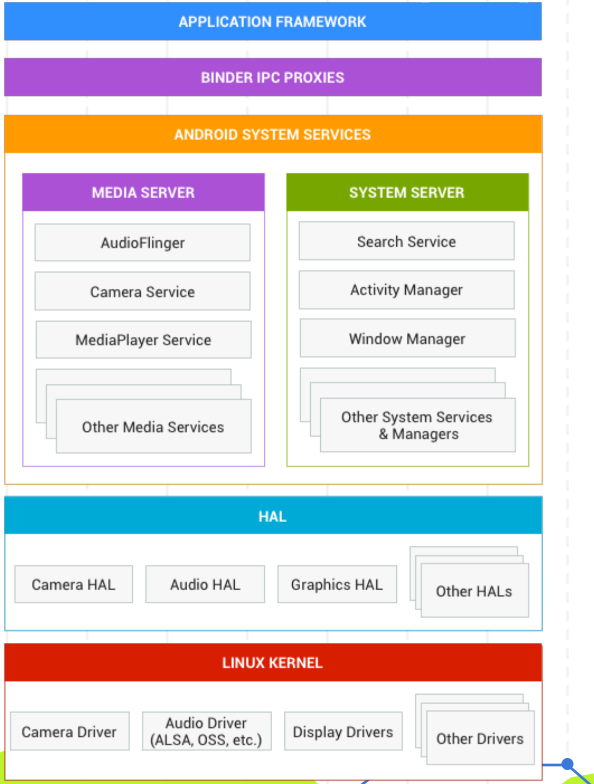
By: Muhammad Obaidullah
Supervised By: Dr. Gul N. Khan

Overview

- Introduction
- Inspiration
- Literature Review
- Implementation
- Results
- Conclusion

INTRODUCTION 1

What is Android ?



APPLICATION FRAMEWORK

BINDER IPC PROXIES

ANDROID SYSTEM SERVICES

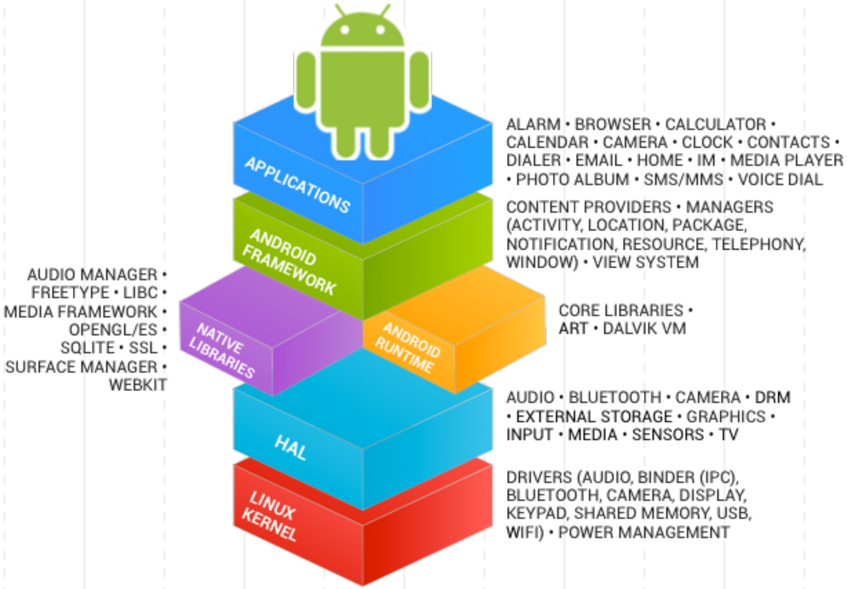
MEDIA SERVER
- AudioFlinger
- Camera Service
- MediaPlayer Service
- Other Media Services

SYSTEM SERVER
- Search Service
- Activity Manager
- Window Manager
- Other System Services & Managers

HAL
- Camera HAL
- Audio HAL
- Graphics HAL
- Other HALs

LINUX KERNEL
- Camera Driver
- Audio Driver (ALSA, OSS, etc.)
- Display Drivers
- Other Drivers

[1]



APPLICATIONS: ALARM • BROWSER • CALCULATOR • CALENDAR • CAMERA • CLOCK • CONTACTS • DIALER • EMAIL • HOME • IM • MEDIA PLAYER • PHOTO ALBUM • SMS/MMS • VOICE DIAL

ANDROID FRAMEWORK: CONTENT PROVIDERS • MANAGERS (ACTIVITY, LOCATION, PACKAGE, NOTIFICATION, RESOURCE, TELEPHONY, WINDOW) • VIEW SYSTEM

NATIVE LIBRARIES: AUDIO MANAGER • FREETYPE • LIBC • MEDIA FRAMEWORK • OPENGL/ES • SQLITE • SSL • SURFACE MANAGER • WEBKIT

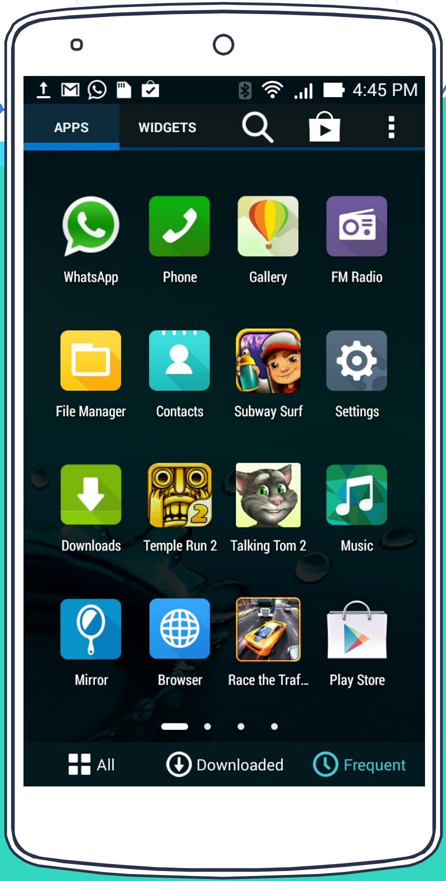
ANDROID RUNTIME: CORE LIBRARIES • ART • DALVIK VM

HAL: AUDIO • BLUETOOTH • CAMERA • DRM • EXTERNAL STORAGE • GRAPHICS • INPUT • MEDIA • SENSORS • TV

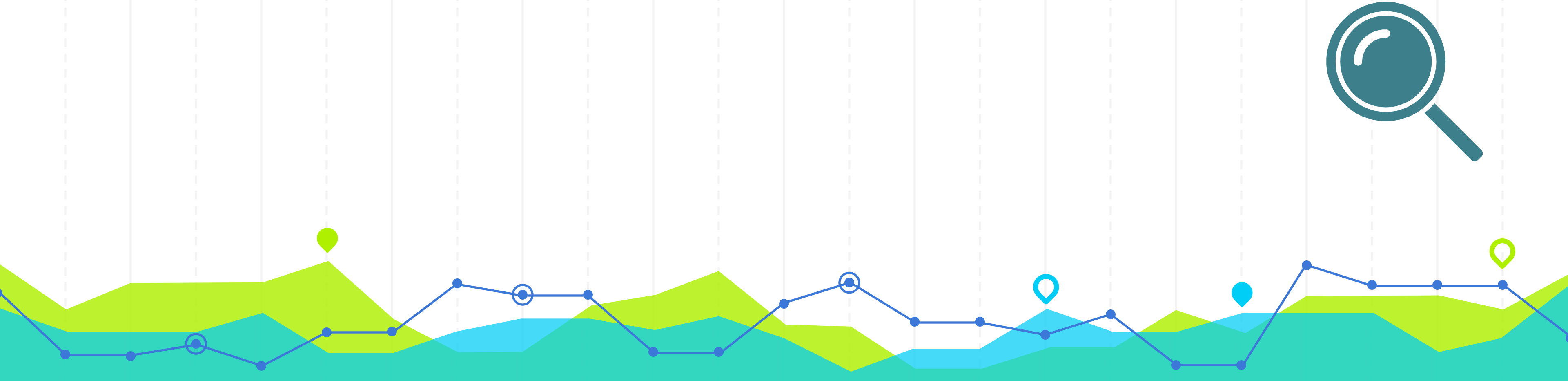
LINUX KERNEL: DRIVERS (AUDIO, BINDER (IPC), BLUETOOTH, CAMERA, DISPLAY, KEYPAD, SHARED MEMORY, USB, WIFI) • POWER MANAGEMENT

[1]

INSPIRATION 2

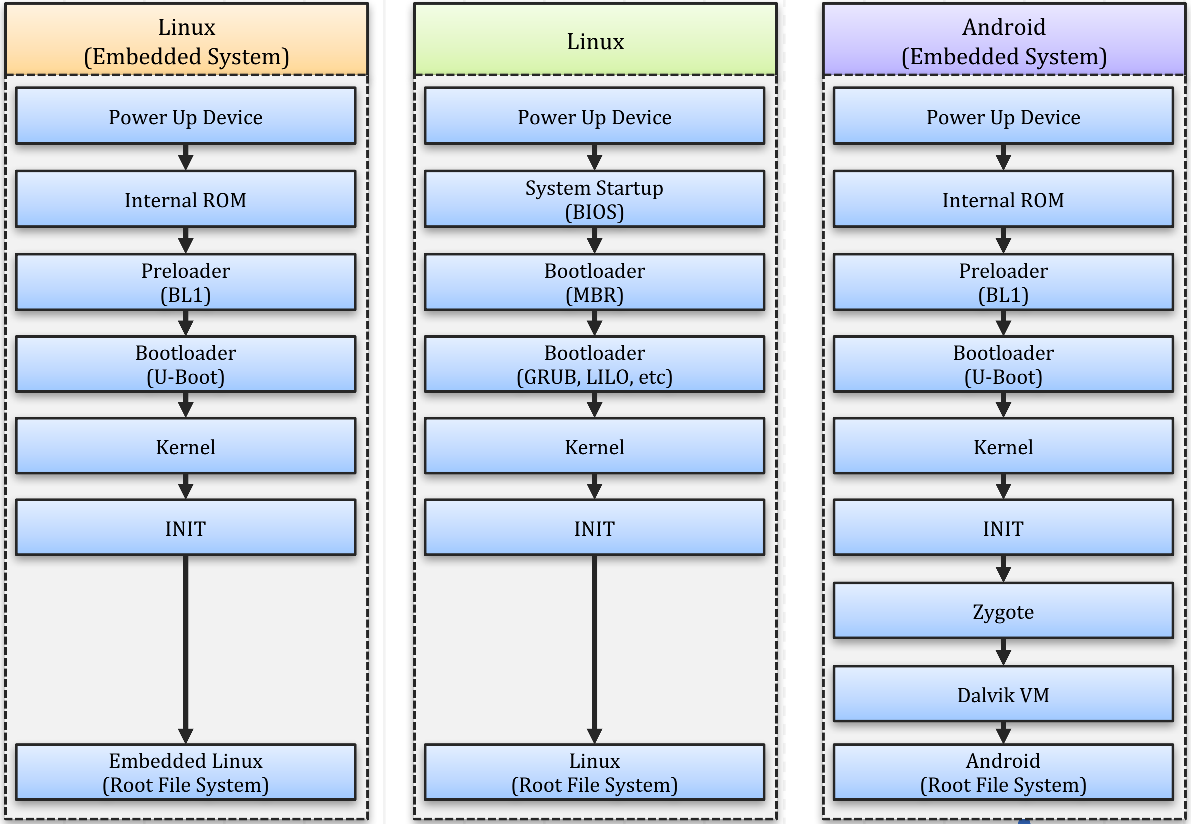


- Code is available online and managed by Google Android Open Source Project (AOSP).
- Based on Linux.
- Designed for embedded devices.
- Suited to low power devices
- Android is Everywhere



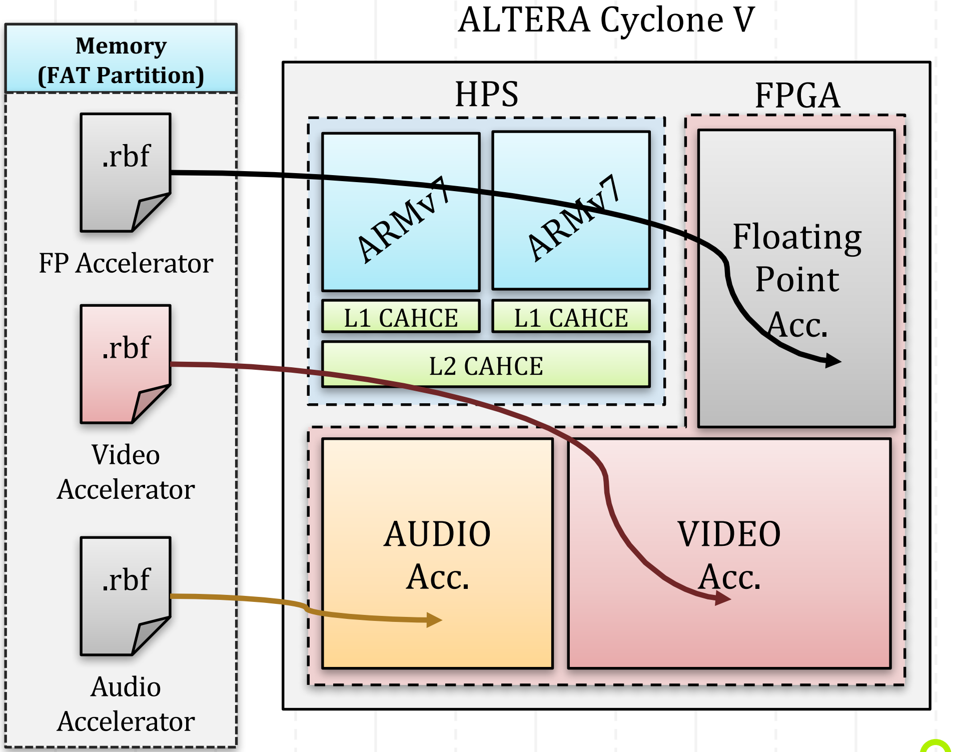
LITERATURE REVIEW 3

Difference Between Linux and Android

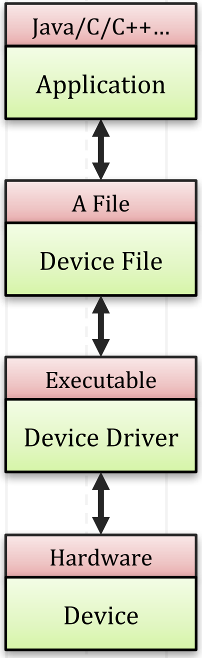


Raw Binary Files (.rbf)

- Write hardware code in VHDL/Verilog/System Verilog
- Compile and verify the design using Quartus to get SRAM Object File (.sof)
- Convert .sof file to .rbf
- Copy the generated .rbf file to SD card's FAT partition
- Decide on how to load the configuration bit stream (Choose from following)
  - Pre-loader script
  - U-Boot source code
  - U-Boot script
  - Linux init
  - Linux application (runtime)



Linux Device Files



Java/C/C++…

Application

A File

Device File

Executable

Device Driver

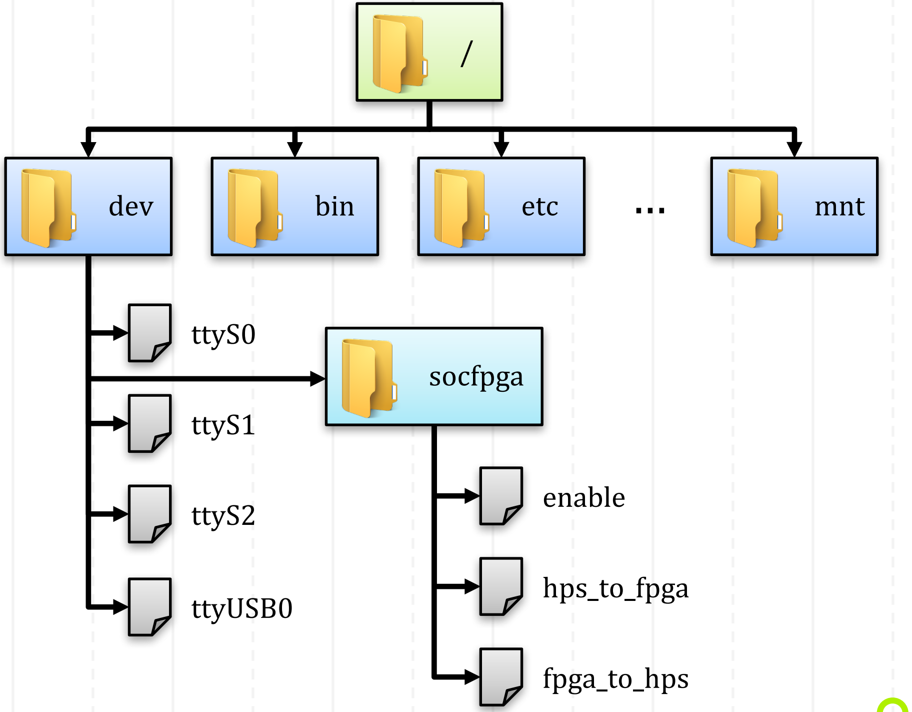
Hardware

Device

The application finds the device file based on device file name

Device file finds the device driver based on the major number

Device driver finds device based on the minor number



/

dev

bin

etc

…

mnt

ttyS0

socfpga

ttyS1

ttyS2

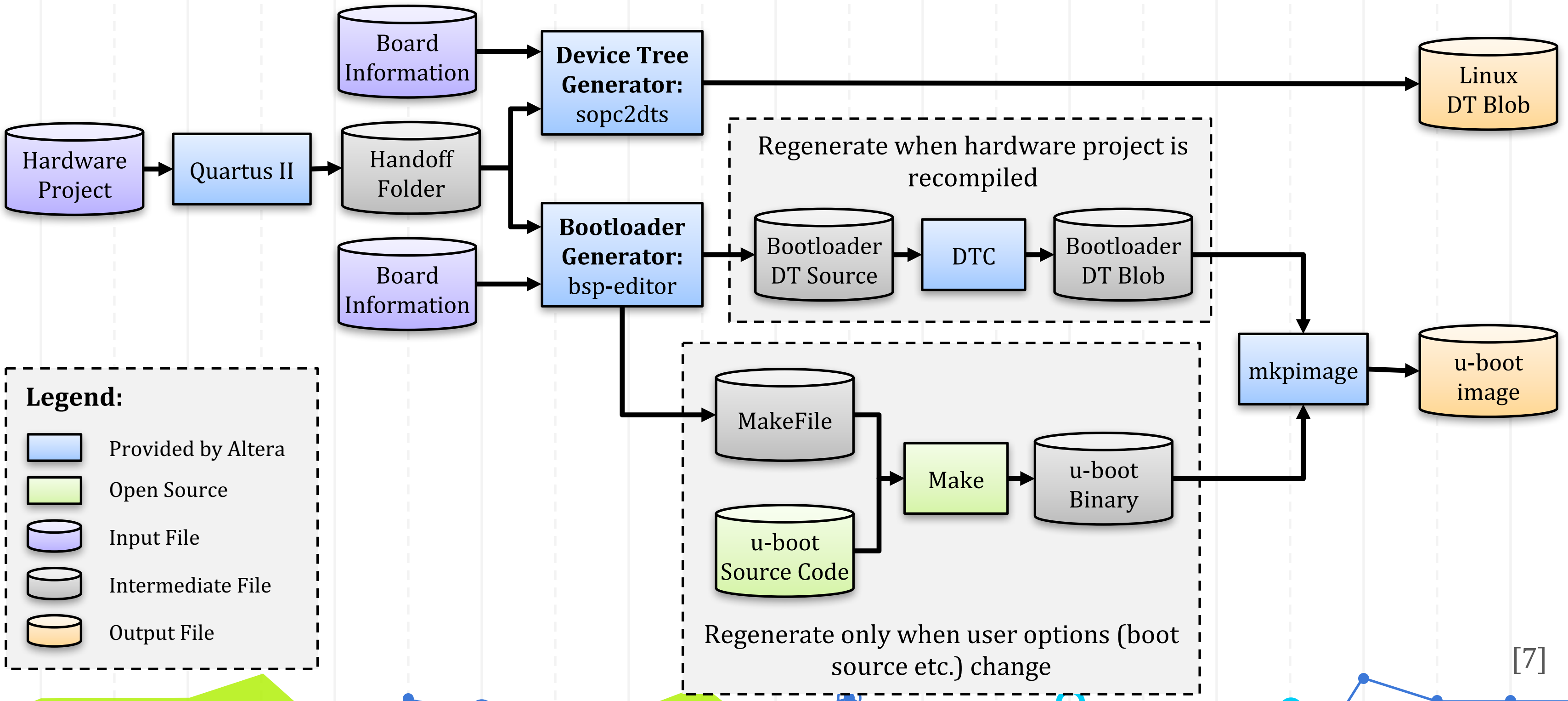
ttyUSB0

enable

hps_to_fpga

fpga_to_hps

Altera SoC Linux Hardware/Software Handoff

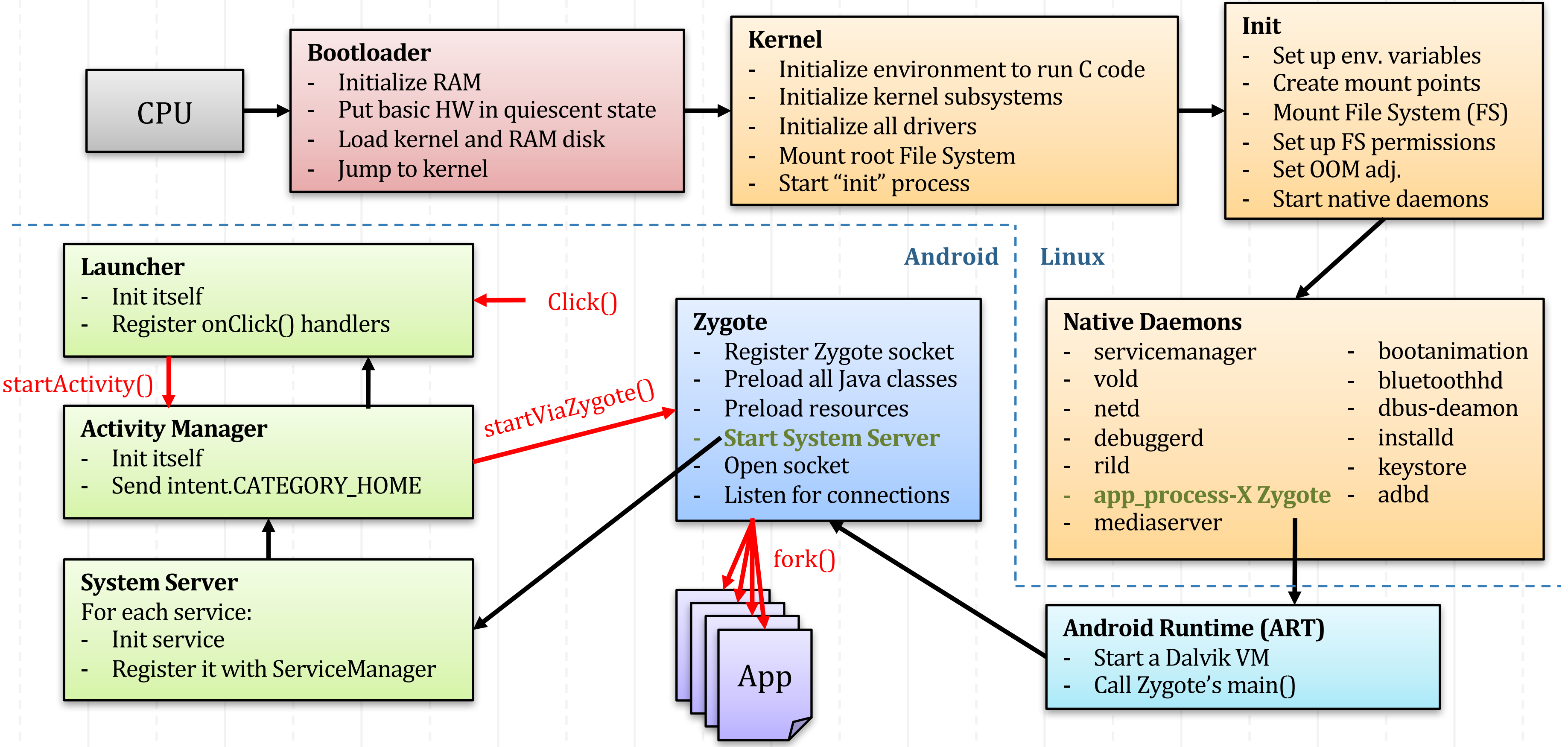


Legend:

- 🟦 Provided by Altera
- 🟩 Open Source
- Input File
- Intermediate File
- Output File

[7]

Android Boot Sequence



CPU

Bootloader

- Initialize RAM
- Put basic HW in quiescent state
- Load kernel and RAM disk
- Jump to kernel

Kernel

- Initialize environment to run C code
- Initialize kernel subsystems
- Initialize all drivers
- Mount root File System
- Start "init" process

Init

- Set up env. variables
- Create mount points
- Mount File System (FS)
- Set up FS permissions
- Set OOM adj.
- Start native daemons

Native Daemons

- servicemanager
- vold
- netd
- debuggerd
- rild
- app_process-X Zygote
- mediaserver
- bootanimation
- bluetoothhd
- dbus-deamon
- installd
- keystore
- adbd

Android Runtime (ART)

- Start a Dalvik VM
- Call Zygote's main()

Zygote

- Register Zygote socket
- Preload all Java classes
- Preload resources
- Start System Server
- Open socket
- Listen for connections

System Server

For each service:

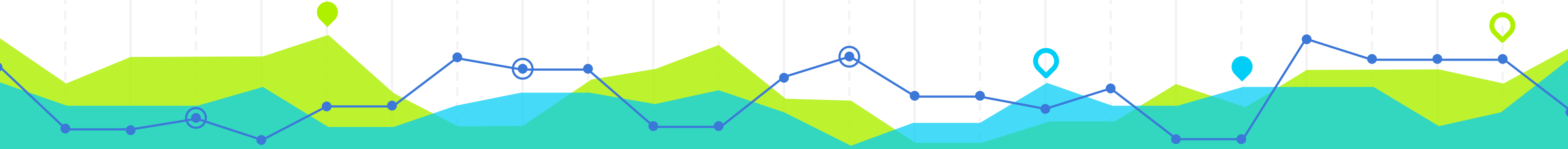
- Init service
- Register it with ServiceManager

Activity Manager

- Init itself
- Send intent.CATEGORY_HOME

Launcher

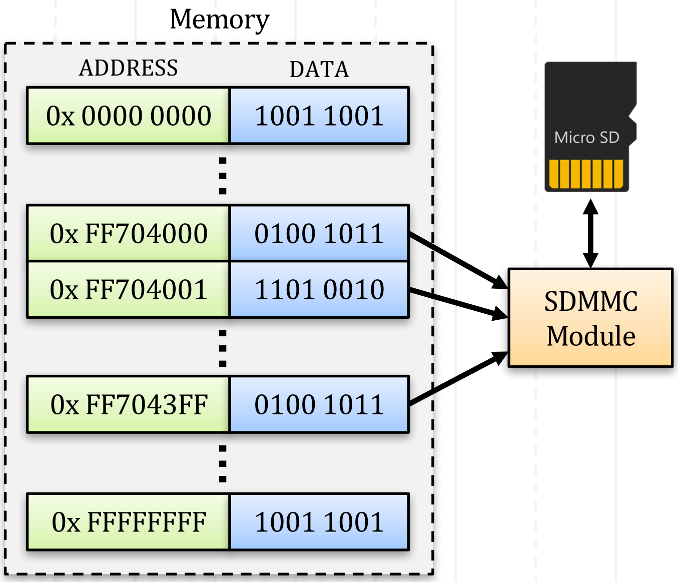
- Init itself
- Register onClick() handlers



IMPLEMENTATION 4

Step 1: Studying Cyclone V HPS Memory Map

[5]



ADDRESS	DATA
0x 0000 0000	1001 1001
⋮	
0x FF704000	0100 1011
0x FF704001	1101 0010
⋮	
0x FF7043FF	0100 1011
⋮	
0x FFFFFFFF	1001 1001

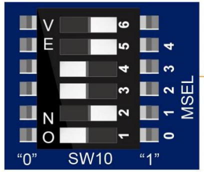
0x C0000000 - 0x FBFFFFFF	Slaves via HP AXI Bridge
0x FF200000 - 0x FF3FFFFF	Slaves via LW AXI Bridge
0x FF704000 - 0x FF7043FF	SDMMC Module
0x FF705000 - 0x FF7050FF	QSPI Flash Controller Module
0x FF708000 - 0x FF70A07F	GPIO Module
0x FFB00000 - 0x FFB7FFFF	USB OTG Controller
0x FFFE0000 - 0x FFE01FFF	DMA Module
0x FFFD0000 - 0x FFFDFFFF	Boot ROM Module
0x FFFD0000 - 0x FFFFFFFF	On-Chip RAM Module

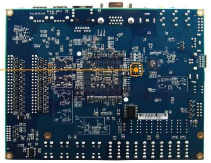
Step 2: Finding Resources

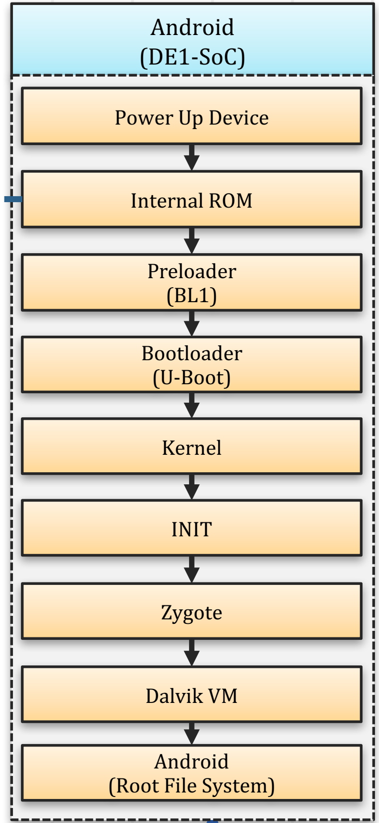
S.No.	Component	Repository Name	GitHub URL
1.	Angstrom Scripts	angstrom-socfpga	https://github.com/altera-opensource/angstrom-socfpga
2.	Boot Loader	u-Boot-socfpga	https://github.com/altera-opensource/u-boot-socfpga
3.	Device Tree Generator	sopc2dts	https://github.com/altera-opensource/sopc2dts
4.	Linux Kernel	linux-socfpga	https://github.com/altera-opensource/linux-socfpga
5.	Reference Designs	linux-refdesigns	https://github.com/altera-opensource/linux-refdesigns
6.	Yocoto Layer	meta-altera	https://github.com/altera-opensource/meta-altera

[6]

Step 3: Internal ROM Configuration

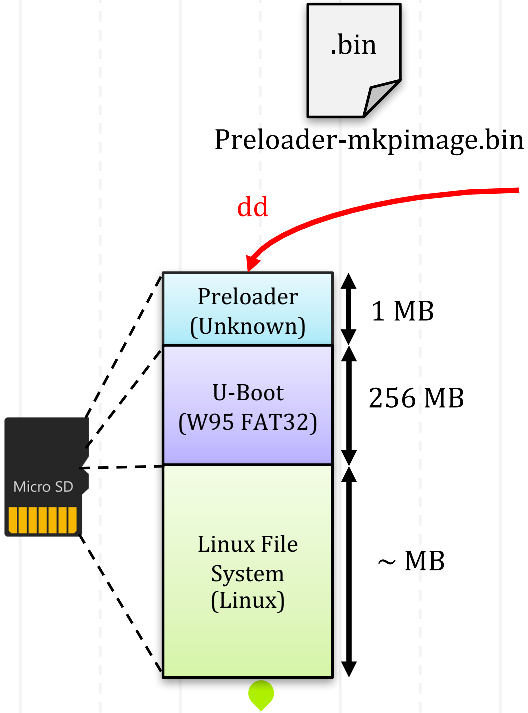


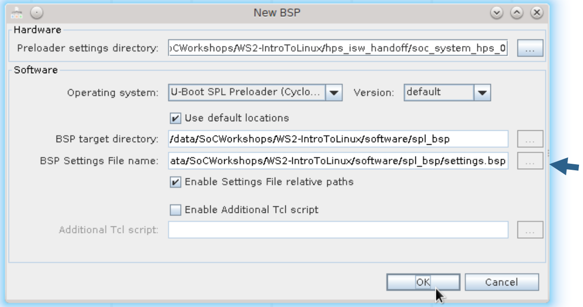




MSEL[4:0]	**Configuration Scheme**	**Description**
10010	AS	FPGA configured from EPCQ (default)
01010	FPPx32	FPGA configured from HPS software: Linux
00000	FPPx16	FPGA configured from HPS software: U-Boot, with image stored on the SD card, like LXDE Desktop or console Linux with frame buffer edition

Step 4: Generating Preloader





New BSP

Hardware

Preloader settings directory: ›CWorkshops/WS2-IntroToLinux/hps_isw_handoff/soc_system_hps_0

Software

Operating system: U-Boot SPL Preloader (Cyclo...   Version: default

☑ Use default locations

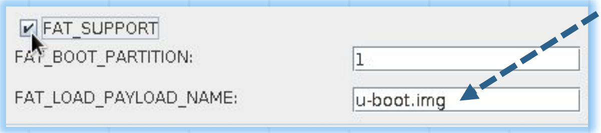
BSP target directory: /data/SoCWorkshops/WS2-IntroToLinux/software/spl_bsp

BSP Settings File name: ata/SoCWorkshops/WS2-IntroToLinux/software/spl_bsp/settings.bsp

☑ Enable Settings File relative paths

□ Enable Additional Tcl script

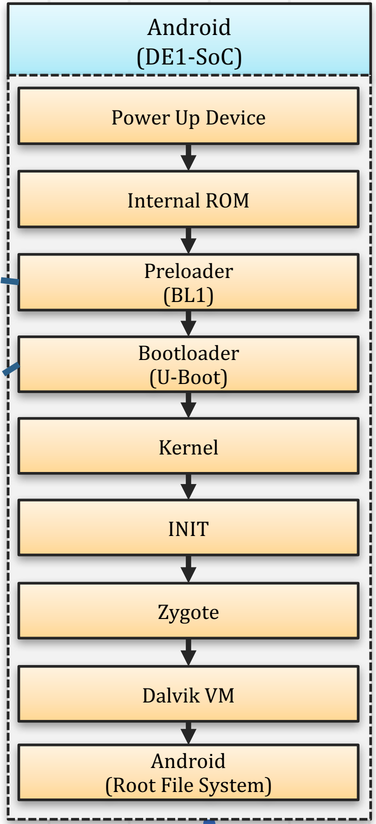
Additional Tcl script:



☑ FAT_SUPPORT

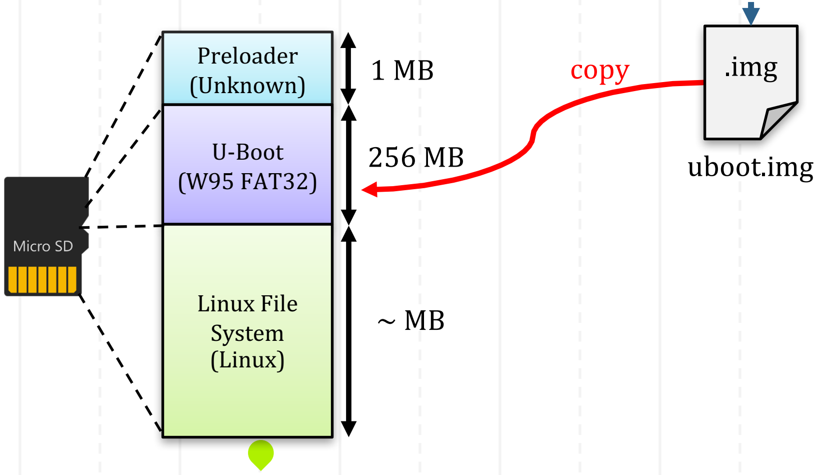
FAT_BOOT_PARTITION: 1

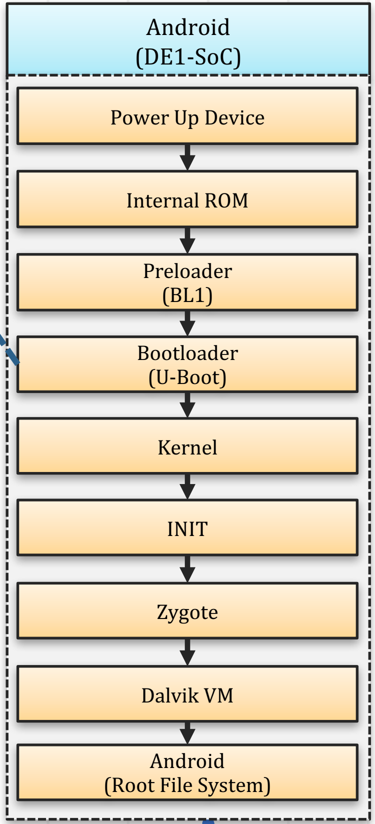
FAT_LOAD_PAYLOAD_NAME: u-boot.img



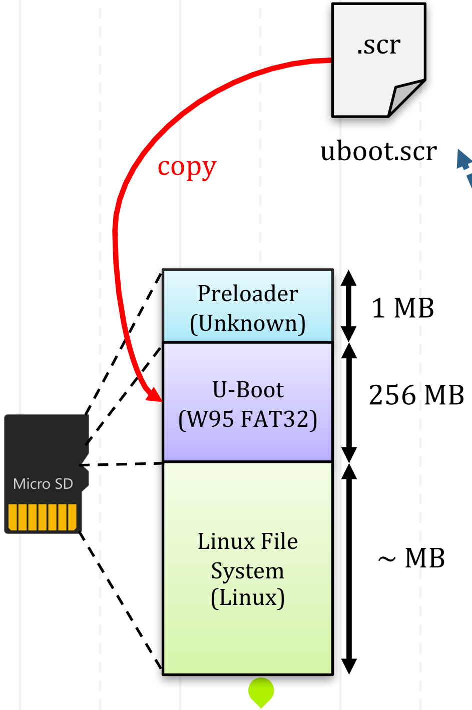
Step 5: Generating U-Boot

```
$ git clone https://github.com/altera-opensource/u-boot-socfpga.git
$ make mrproper
$ make socfpga_cyclone5_config
$ make
```

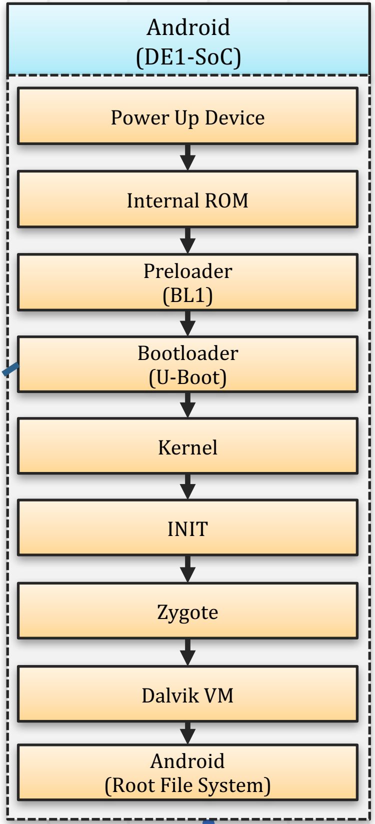




Step 6: Generating U-Boot Script

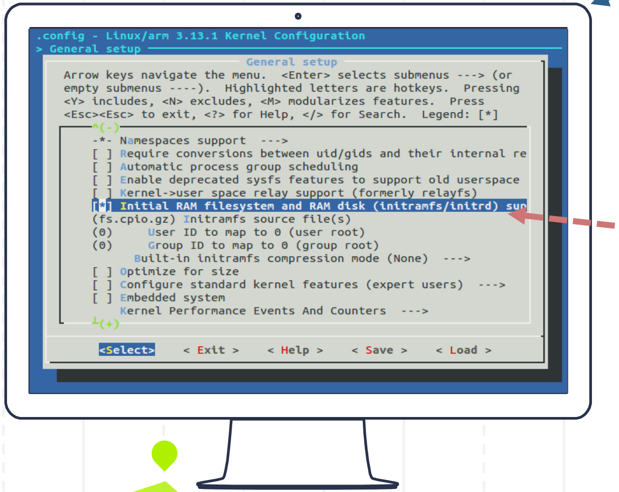


```
echo -- Programming FPGA –
fatload mmc 0:1 $fpgadata soc_config.rbf;
fpga load 0 $fpgadata $filesize;
run bridge_enable_handoff;

echo -- Setting Env Variables –
setenv fdtimage soc_system.dtb;
setenv mmcroot /dev/ram;
setenv mmcload 'mmc rescan;
${mmcloadcmd} mmc 0:${mmcloadpart} ${loadaddr}
${bootimage};
${mmcloadcmd} mmc 0:${mmcloadpart} ${fdtaddr}
${fdtimage};
setenv mmcboot 'setenv bootargs
console=ttyS0,115200 root=${mmcroot} rw
rootwait;
bootz ${loadaddr} - ${fdtaddr}';

run mmcload;
run mmcboot;
```



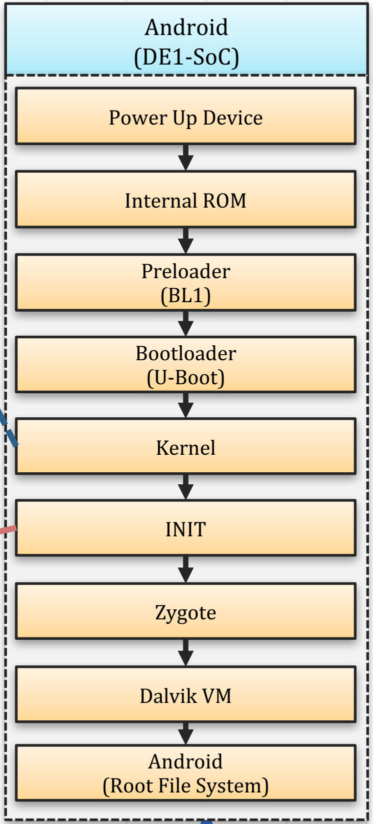
Step 7: Configuring Linux

```
$ git clone https://github.com/altera-opensource/linux-socfpga.git
$ make ARCH=arm socfpga_custom_defconfig
$ make ARCH=arm menuconfig
```

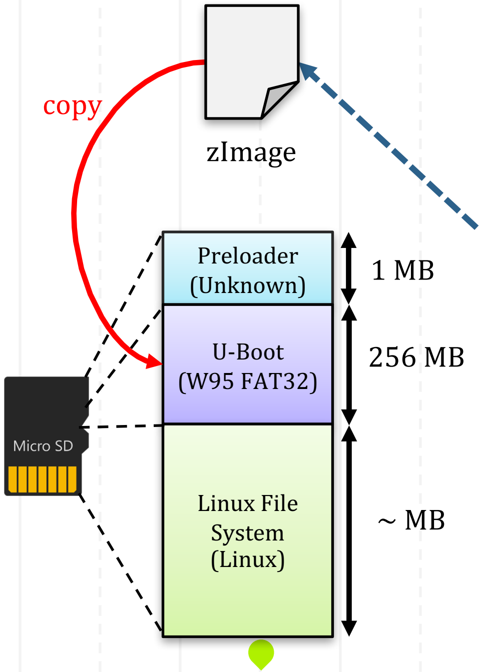


```
.config - Linux/arm 3.13.1 Kernel Configuration
> General setup ──────────────────────────────────────────────
                          General setup
  Arrow keys navigate the menu.  <Enter> selects submenus ---> (or
  empty submenus ----).  Highlighted letters are hotkeys.  Pressing
  <Y> includes, <N> excludes, <M> modularizes features.  Press
  <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*]
      ^(-)
      -*- Namespaces support  --->
      [ ] Require conversions between uid/gids and their internal re
      [ ] Automatic process group scheduling
      [ ] Enable deprecated sysfs features to support old userspace
      [ ] Kernel->user space relay support (formerly relayfs)
      [*] Initial RAM filesystem and RAM disk (initramfs/initrd) sup
      (fs.cpio.gz) Initramfs source file(s)
      (0)     User ID to map to 0 (user root)
      (0)     Group ID to map to 0 (group root)
            Built-in initramfs compression mode (None)  --->
      [ ] Optimize for size
      [ ] Configure standard kernel features (expert users)  --->
      [ ] Embedded system
          Kernel Performance Events And Counters  --->
      ┴(+)

    <Select>    < Exit >    < Help >    < Save >    < Load >
```

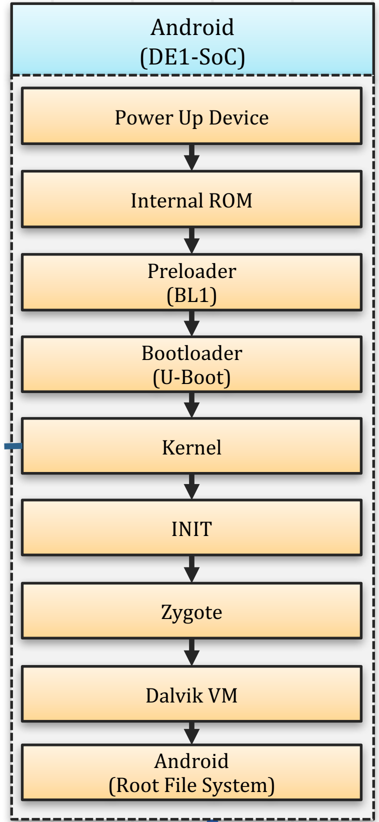
Specify the file system and RAM disk to load



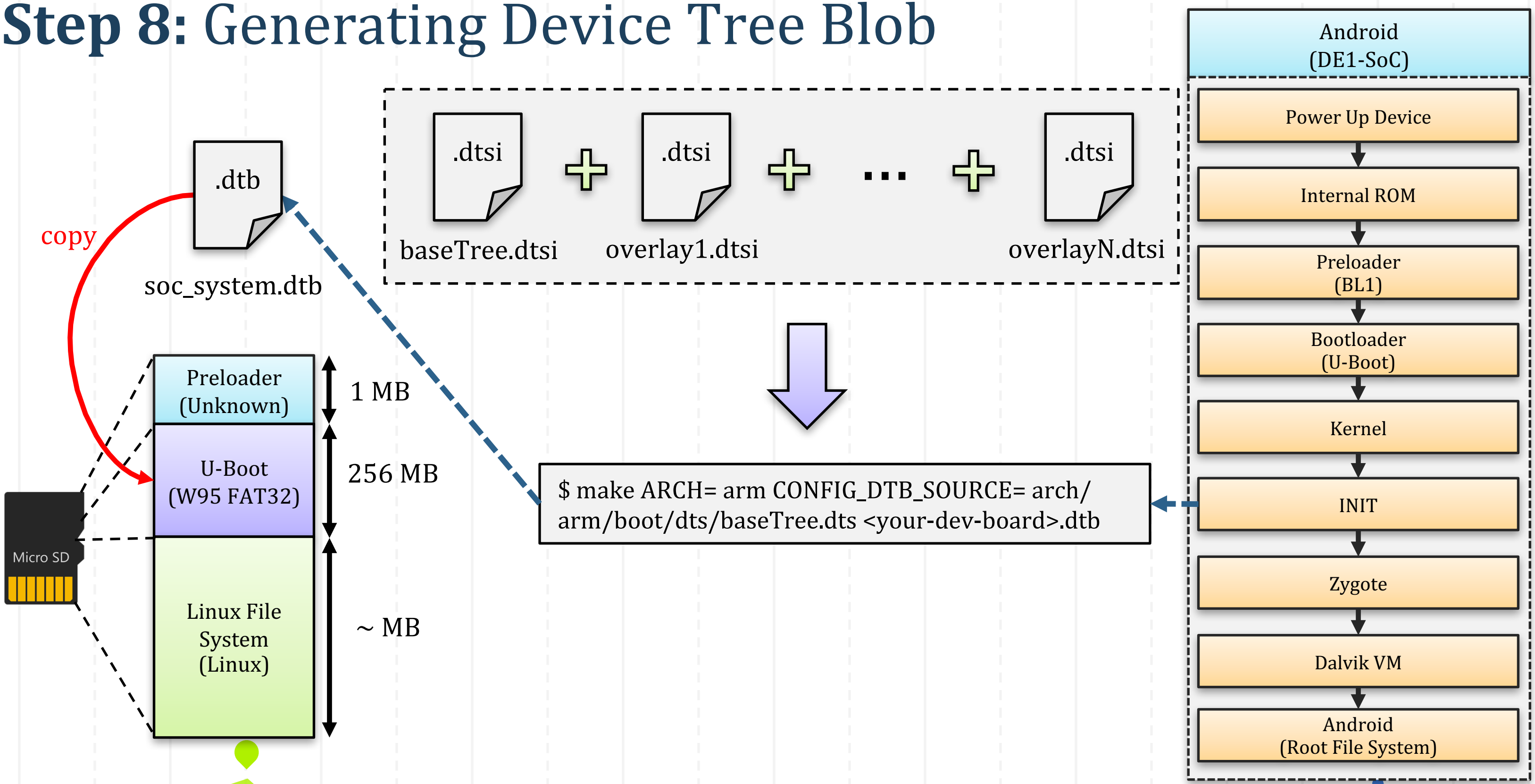
Step 8: Compiling Linux



```
$ make ARCH=arm LOCALVERSION= zImage
```



Step 8: Generating Device Tree Blob

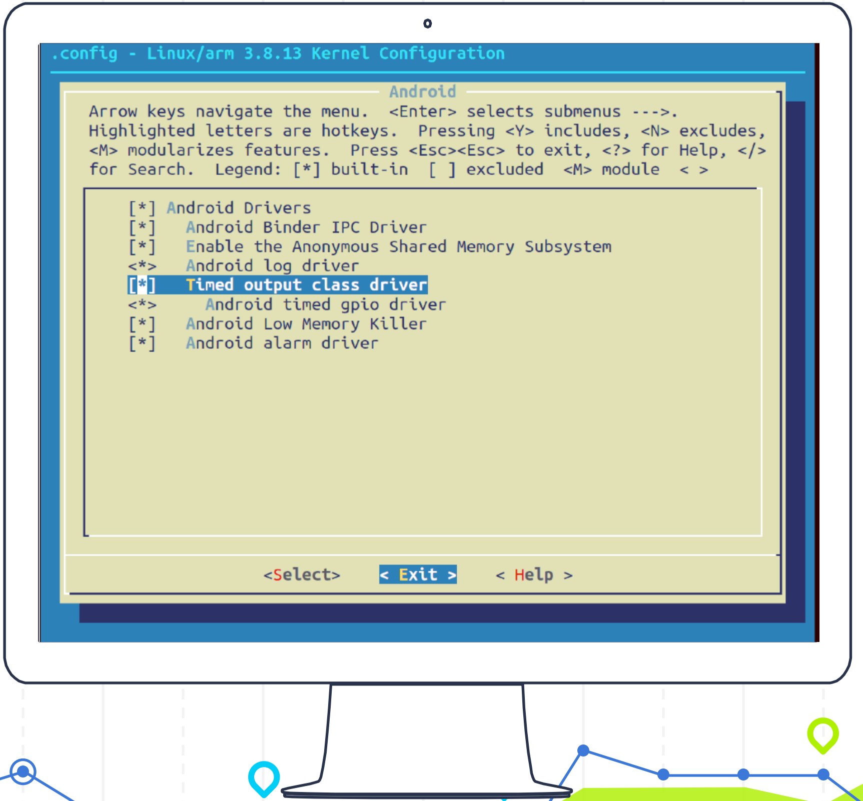


```
$ make ARCH= arm CONFIG_DTB_SOURCE= arch/
arm/boot/dts/baseTree.dts <your-dev-board>.dtb
```

Step 9: Android Bindings

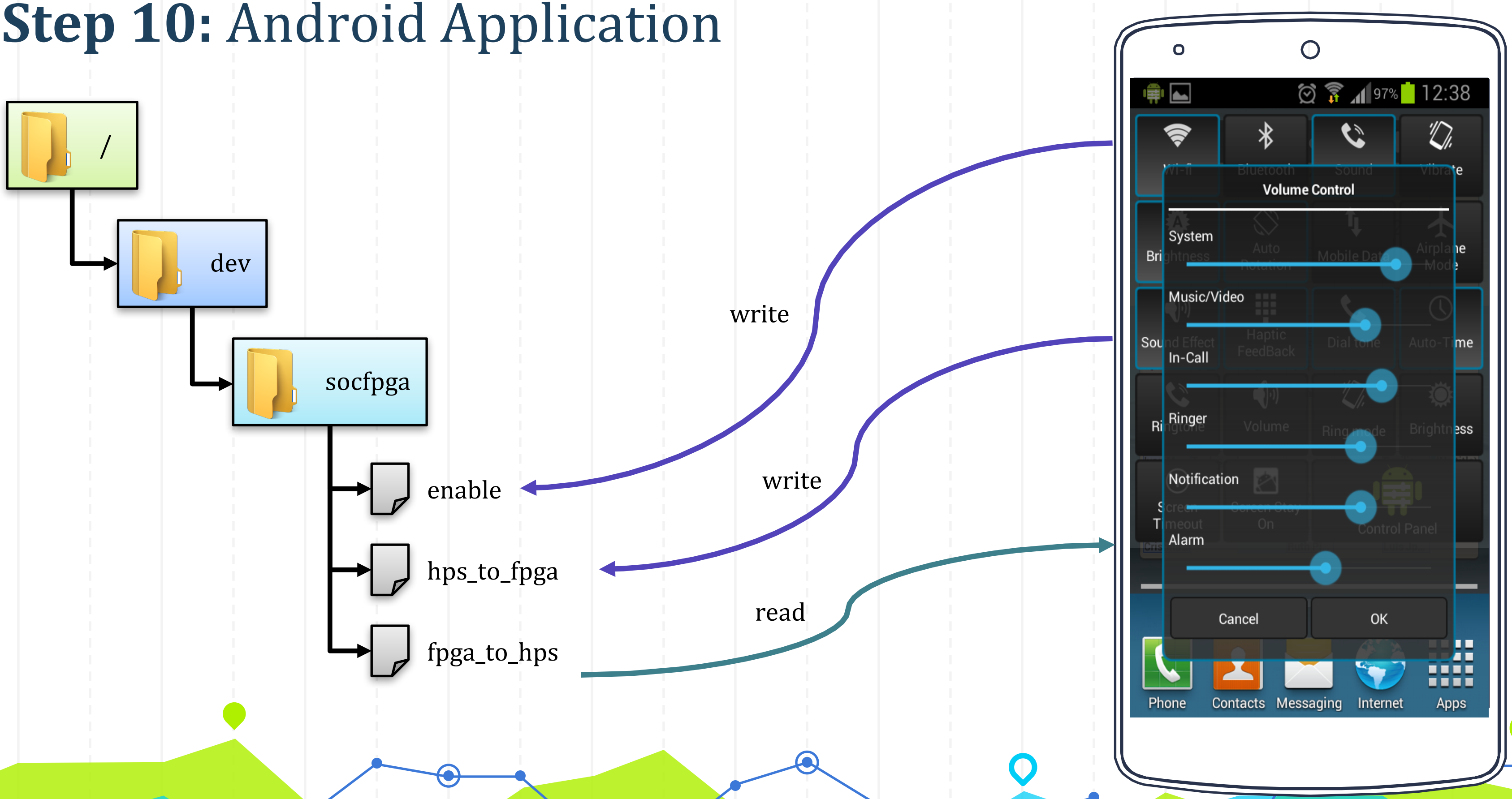
Developing your device drivers is similar to developing a typical Linux device driver. Android uses a version of the Linux kernel with a few special additions such as wake locks (a memory management system that is more aggressive in preserving memory), the Binder IPC driver, and other features important for a mobile embedded platform. These additions are primarily for system functionality and do not affect driver development.

You can use any version of the kernel as long as it supports the required features (such as the binder driver).



```
.config - Linux/arm 3.8.13 Kernel Configuration
                              Android
  Arrow keys navigate the menu.  <Enter> selects submenus --->.
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
  <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </>
  for Search.  Legend: [*] built-in  [ ] excluded  <M> module  < >

      [*] Android Drivers
      [*]   Android Binder IPC Driver
      [*]   Enable the Anonymous Shared Memory Subsystem
      <*>   Android log driver
      [*]   Timed output class driver
      <*>     Android timed gpio driver
      [*]   Android Low Memory Killer
      [*]   Android alarm driver

                 <Select>    < Exit >    < Help >
```

Step 10: Android Application



/

dev

socfpga

enable

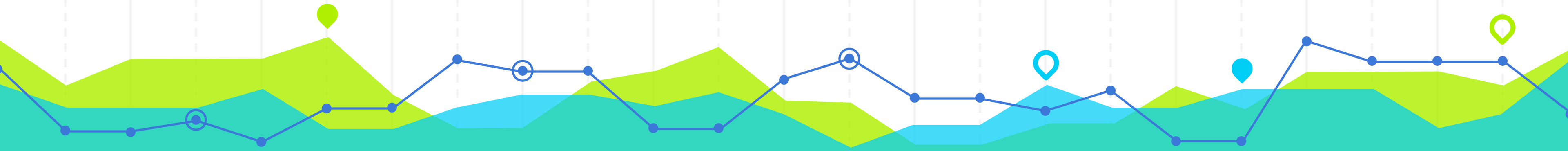
hps_to_fpga

fpga_to_hps

write

write

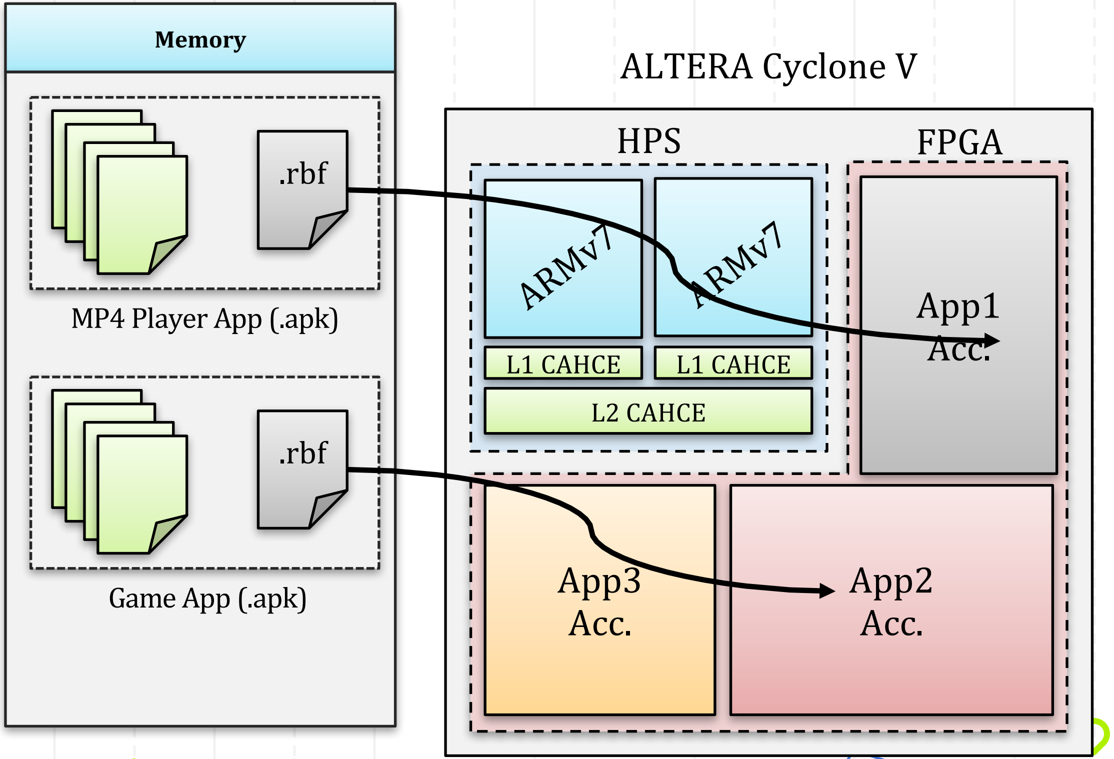
read



RESULTS 5

Raw Binary Files (.rbf) in Android Application Package (.apk)

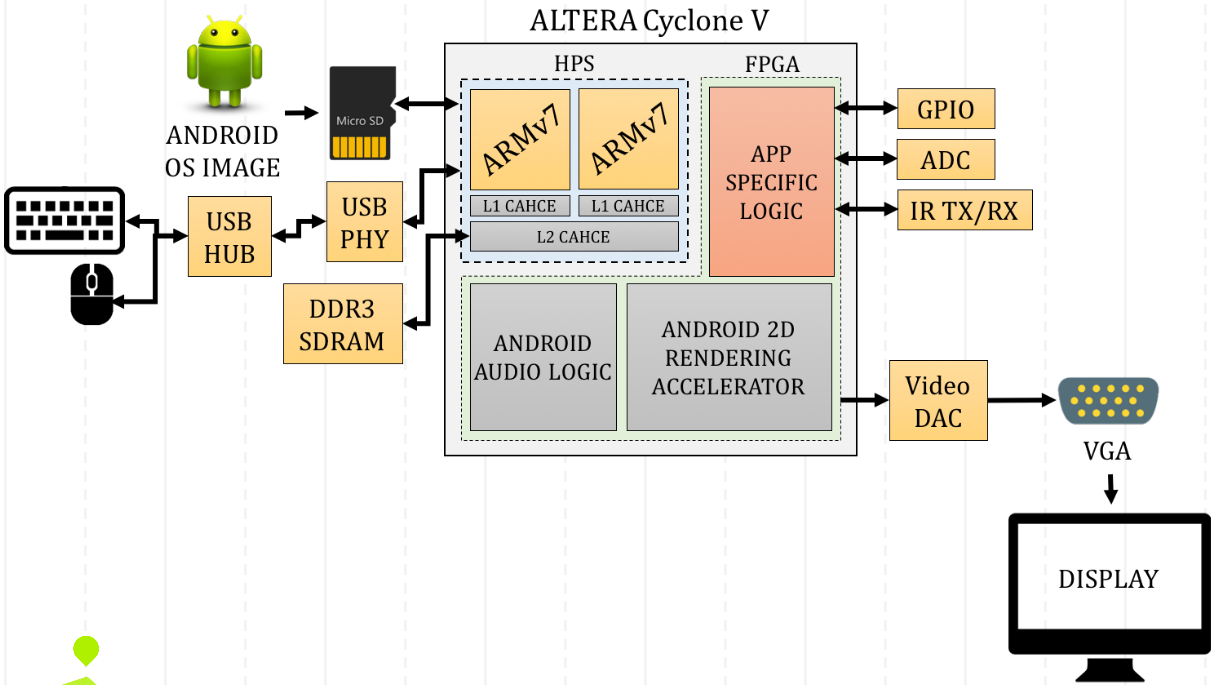
- Load .rbf file on activity create event onto the fpga for acceleration
- Remove .rbf file on activity close event
- Android Init() should configure the fpga with base configuration file and several PRRs (Partial Reconfiguration Regions).



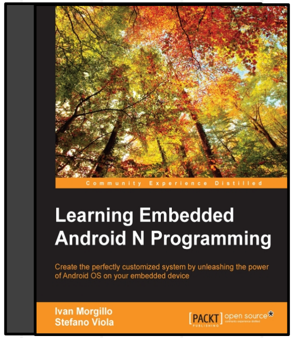
Demo

CONCLUSION 6

App-based Hardware Acceleration



Books Used



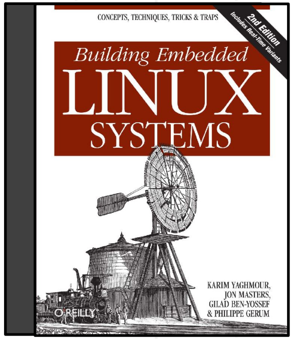
Community Experience Distilled

Learning Embedded Android N Programming

Create the perfectly customized system by unleashing the power of Android OS on your embedded device

Ivan Morgillo
Stefano Viola

[PACKT] open source*



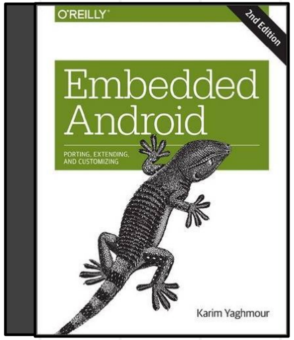
CONCEPTS, TECHNIQUES, TRICKS & TRAPS

2nd Edition
Includes Real-Time Variants

Building Embedded
LINUX
SYSTEMS

KARIM YAGHMOUR,
JON MASTERS,
GILAD BEN-YOSSEF
& PHILIPPE GERUM

O'REILLY



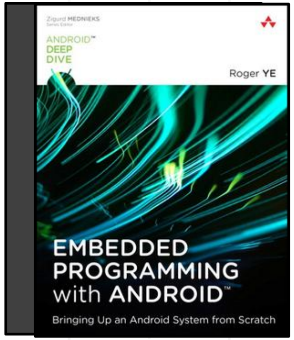
O'REILLY

2nd Edition

Embedded Android

PORTING, EXTENDING, AND CUSTOMIZING

Karim Yaghmour



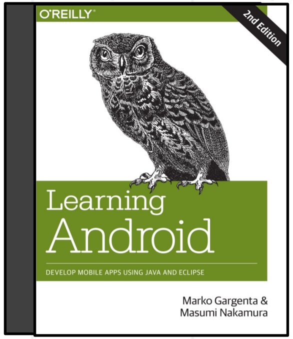
Zigurd MEDNIEKS
Series Editor

ANDROID™
DEEP
DIVE

Roger YE

EMBEDDED
PROGRAMMING
with ANDROID™

Bringing Up an Android System from Scratch



O'REILLY

2nd Edition

Learning Android

DEVELOP MOBILE APPS USING JAVA AND ECLIPSE

Marko Gargenta &
Masumi Nakamura

People who are really serious about software should make their own hardware.

~ Alan Kay (Computer Scientist)

References

1. Google Inc., "Android Open Source Project," Google, 23 August 2016. [Online]. Available: https://source.android.com. [Accessed 9 November 2016].
2. M. Daum, "Android for DE1-SoC Board," RocketBoards.org, 6 October 2016. [Online]. Available: https://rocketboards.org/foswiki/view/Projects/AndroidForDE1SoCBoard. [Accessed 9 November 2016].
3. "Graphics Accelerator for Android," FUJISOFT INCORPORATED, 15 November 2013. [Online]. Available: https://www.fsi-embedded.jp/e/_emb/gaforandroid_e/. [Accessed 9 November 2016].
4. Intel, "Implementation of an Android™ Operating System on an Altera SoC," Intel, 8 January 2014. [Online]. Available: https://youtu.be/zHqS_yWiMNI. [Accessed 9 November 2016].
5. Altera, "Cyclone V HPS Memory Map," Altera, 16 November 2016. [Online]. Available: https://www.altera.com/hps/en_us/cyclone-v/hps.html#topic/sfo1418687413697.html. [Accessed 16 November 2016].
6. Yves Vandervennet, "Move to GitHub," RocketBoards.org, 7 August 2015. [Online]. Available: https://rocketboards.org/foswiki/view/Documentation/MoveToGitHub. [Accessed 21 November 2016].
7. Altera, "Workshop #1 – Altera SoC Software Development Overview," RocketBoards.org, 23 April 2016. [Online]. Available: https://rocketboards.org/foswiki/pub/Documentation/WS1IntroToAlteraSoCDevices/WS_1_Intro_To_SoC_SW_Workshop.pdf. [Accessed 23 November 2016].

THANKS!

Any questions?

You can find me at
ENG 402 / mobaidullah@ryerson.ca