Android多核心嵌入式多媒體系統設計與實作

Android Architecture

賴槿峰 (Chin-Feng Lai)
Assistant Professor, institute of CSIE, National Ilan University
Oct. 13th 2012

Outline

- Introduction to Android
- Android Architecture
- Android Multimedia Framework
- Android Porting
- Android start-up program
- LAB : Mount Android Filesystem

- ***Introduction to Android***
- *Android Architecture*
- *Android Multimedia Framework*
- *Android Porting*
- *Android start-up programming*
- *LAB : Mount Android Filesystem*

Introduction to Android

- What's Android ?

Galaxy tab ***Motolola XOOM***

- Android is an operating system released by Google at 5th of November 2007,the goal is to develop open standards for mobile devices by Open Handset Alliance

Introduction to Android

Android Millstone

Date	Notes
2003	Andy Rubin Founded Android
2005.7	Google buy Android
2007.11	Handset Alliance announces Android
2007.11	Early look Android SDK releas
2008.8	Android Market announced
2008.9	Android 1.1 release
2008.9	T-Mobile G1, Android 1.0 SDK release 1 available
2008.10	Android Open Source Project
2009.4	Android 1.5 release
2009.9	Android 1.6 release
2009.10	Android 2.0 release
2009.10	Android 2.1 release
2010.5	Android 2.2 release
2010.12	Android 2.3 release
2011	Android 3.0 release

Introduction to Android

- Version of Android

1.5 (Cupcake)
Based on Linux Kernel 2.6.27

1.6 (Donut)
Based on Linux Kernel 2.6.29

2.0 / 2.1 (Eclair)
Based on Linux Kernel 2.6.29

2.2 (Froyo)
Based on Linux Kernel 2.6.32

2.3 (Gingerbread)
Based on Linux Kernel 2.6.35

3.x (Honeycomb)
Based on Linux Kernel 2.6.38

Introduction to Android

Google G1

Nexus one

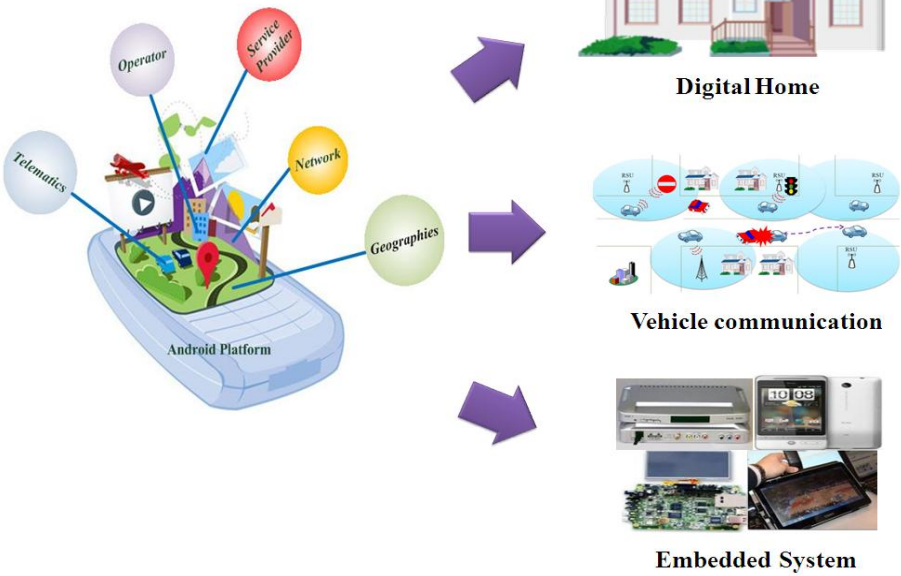
Nexus S

Nexus galaxy(android 4.0)

Nexus 7(android 4.1)

MMN Lab.

Introduction to Android



Cooperation：
PacketVideo OpenCore、Droid Fonts Family

And many other freeware：
Linux Kernel、
SQLite、
Apache Harmony、
FreeType、
Webkit、
OpenGL/ES、
OpenSSL、
BSD libc(Bionic libc) etc.

Introduction to Android - Features

- **Handset layouts**
  - VGA, 2D graphics library, 3D graphics library based on OpenGL ES 2.0 specifications
- **Storage**
  - SQLite , a lightweight relational database
- **Connectivity**
  - GSM/EDGE, IDEN, CDMA, EVDO, UMTS, Bluetooth, WiFi
- **Messaging**
  - SMS and MMS
    are available forms of messaging, also support Android Cloud to Device Messaging Framework (C2DM)
- **Multiple Language Support**
  - Multiple languages are available on Android

Introduction to Android - Features

- **Web browser**
  - The web browser available in Android is based on the open-source WebKit layout engine
- **Java support**
  - Java classes are compiled into Dalvik executables and run on the Dalvik virtual machine,
  - Specialized virtual machine designed specifically for Android and optimizedfor battery-powered mobile devices with limited memory and CPU

Introduction to Android - Features

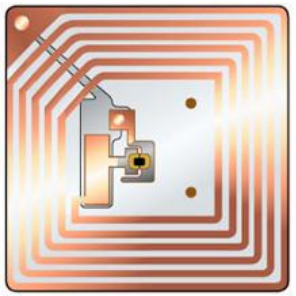
- **Media support**
  - Android supports the following audio/video/still media formats: H.263, H.264 (in 3GP or MP4 container),MPEG-4 SP,AMR, AMR-WB(in 3GP container), AAC, HE-AAC (in MP4 or 3GP container), MP3, MIDI, Ogg Vorbis, WAV, JPEG, PNG,GIF, BMP
- **Streaming media support**
  - RTP/RTSP streaming (3GPP PSS, ISMA), HTML progressive download,
  - Adobe Flash Streaming (RTP) and HTTP Dynamic Streaming ar e supported by the Flash 10.1 plugin
- **Additional hardware support**
  - Android can use video/still cameras, touch screens, GPS, accelerometers, gyroscopes, magnetometers

Introduction to Android - Features

- **Multi-touch**
  - Android has native support for multi-touch which was initially made available in handsets such as the HTC Hero
- **Bluetooth**
  - Supports A2DP, AVRCP, sending files (OPP)
- **Multitasking**
  - Multitasking of applications is available
- **Voice based features**
  - Google search through voice has been available since initial rele ase, Voice actions for calling, texting, navigation ,etc

Introduction to Android - Features

- Near Field Communication
  - a set of short-range wireless technologies
  - a type of RFID
  - requiring a distance of 4cm or less
  - Application
    - Electronic wallet
    - P2P Communication
    - Tag Reader/Writer



Introduction to Android - Features

- Android Market(Google Play)
  - an online software store developed by Google for Android devices
  - allows users to browse and download apps published by third-party developers, hosted on Android Market

Date	Applications	Downloads to date
March 2009	2,300[7]	
December 2009	20,000[8]	
August 2010	80,000[9][10]	1 billion
May 2011	200,000[2]	3 billion[11]

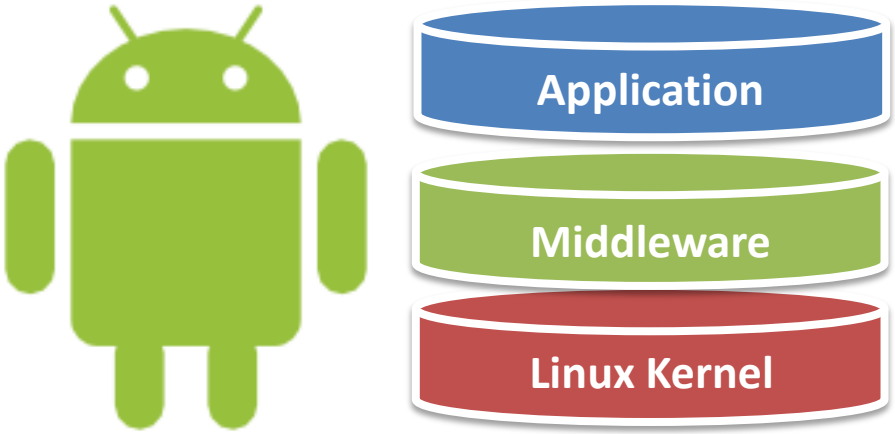
Introduction to Android

- Android source code has been available under a free software/open source license since October,21 2008
- Google published the entire source code (including network and telephony stacks) under an **Apache License**
- We can get the source code from http://source.android.com/

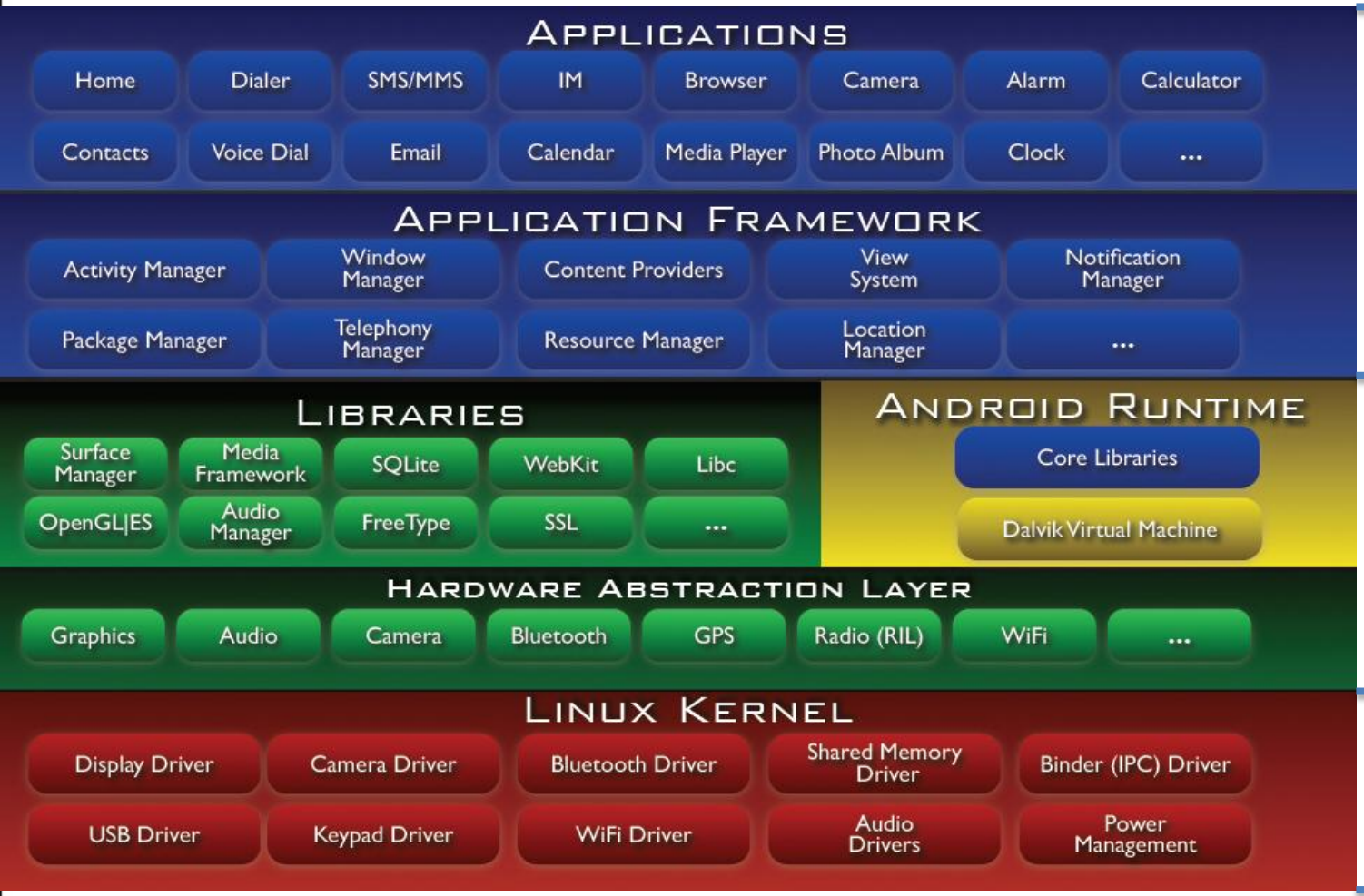
- *Introduction to Android*
- ***Android Architecture***
- *Android Multimedia Framework*
- *Android Porting*
- *Android start-up programming*
- *LAB : Mount Android Filesystem*

Android Architecture

- Android is software stack for mobile devices
- It includes
  - Application
    - Google map、Alarm、widget
  - Middleware
    - Framework、libraries、Dalvik VM
  - Linux kernel



Android Architecture



Application
Java based

Middleware
C++/C based

Middleware
C/Assembly based

MMN Lab.

Android Architecture

- Kernel Layer
  - Android is built on the Linux kernel 2.6.24+ and Provide core system services such as process, memory, power management, network stack, driver model and security
  - Android Driver Ashmem, Binder, Power Management, LowMemKillrer, logger

Introduction

- Android additional kernel driver

driver Features	2.6.23	2.6.35	2.6.27	2.6.29	2.6.32 (Froyo)	2.6.33+
Alarm Driver	✓	✓	✓	✓	✓	×
Android Logger (logcat)	✓	✓	✓	✓	✓	×
Low Memory Killer	✓	✓	✓	✓	✓	×
Wakelock (power management)	✓	✓	✓	✓	✓	×
USB Gadget	✓	✓	✓	✓	✓	×
ASHMEM (shared memory)	×	✓	✓	✓	✓	×
PMEM (memory allocator)	×	×	✓	✓	✓	×
X86 Support	×	×	✓	✓	✓	×
driver/staging/Android	×	×	×	✓	✓	×

Android Architecture

- Hardware Abstraction Layer
  - Abstract the hardware features , and separate the kernel space and user space
  - It is user space, not kernel space
  - GPL problem

Android Architecture

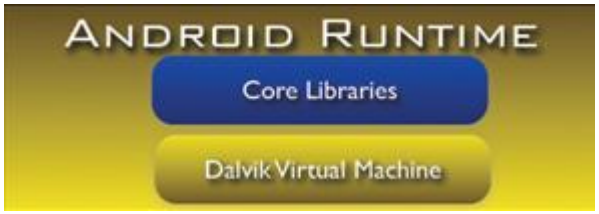
- The library runs in the system.
  - Bionic Libc
  - Function Libraries
  - Hardware Abstraction Libraries

Android Architecture

- Bionic Libc
  - C/C++ library, custom libc implementation, optimized for embedded use.
- WebKit- Apple Safari
  - Based on open source WebKit browser
  - Full CSS, Javascript, DOM, AJAX support
- Media Framework
  - Based on PacketVideo OpenCORE platform
  - Supports standard video, audio, still-frame formats
  - Might be replaced by Stagefright framework
- Surface manager
- Audio manager
- OpenGL

Android Architecture

- Android Runtime.
  - Not use java runtime, and java virtual machine
  - Core library already contained more originally java API
  - Unlike most of virtual machines that are stack based, Dalvik architecture is register based.



Android Architecture

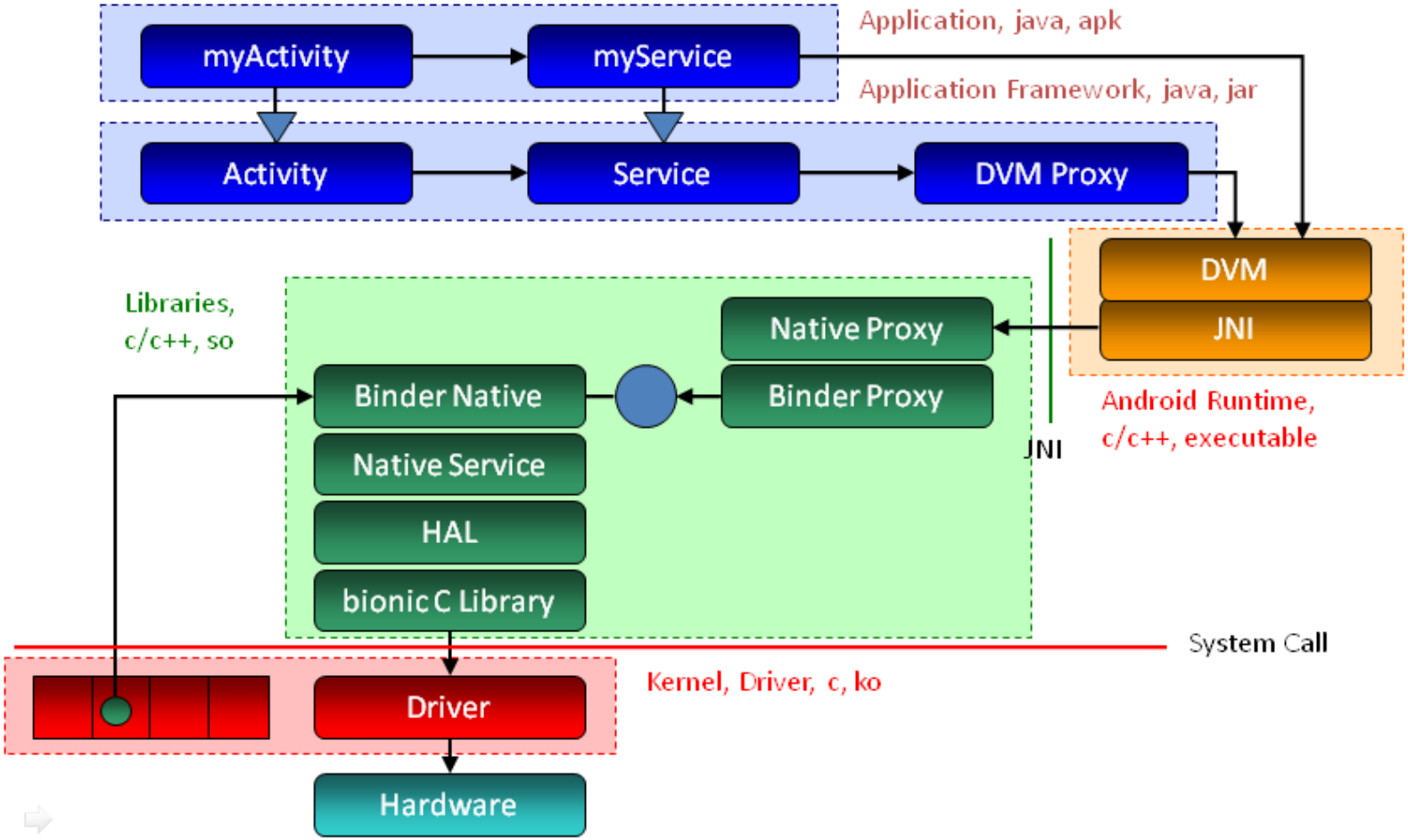
- Application Framework
  - Provide developer with complete application programming interface
  - Application is composed of Services with System
    - Core system
      - Activity manager (manages application lifecycle)
      - Package manager (loads apk files)
      - Window manager (handles applications window manager interaction with surface flinger)
      - Resource manager (handles media resources)
      - Content providers (provides data to application)
      - View system (provides widgets, views, layouts to applications)
    - Hardware Service
      - Provides low-level access to hardware device
      - Location manager, Telephony manager, Bluetooth service, WiFi service, USB service, Sensor service

Android Architecture

- Application
  - Java programming
  - Default Application : Email, Browser, Clock, Calendar, etc.

Android Architecture

- Program flow



- *Introduction to Android*
- *Android Architecture*
- ***Android Multimedia Framework***
- *Android Porting*
- *Android start-up programming*
- *LAB : Mount Android Filesystem*

Android Multimedia Framework

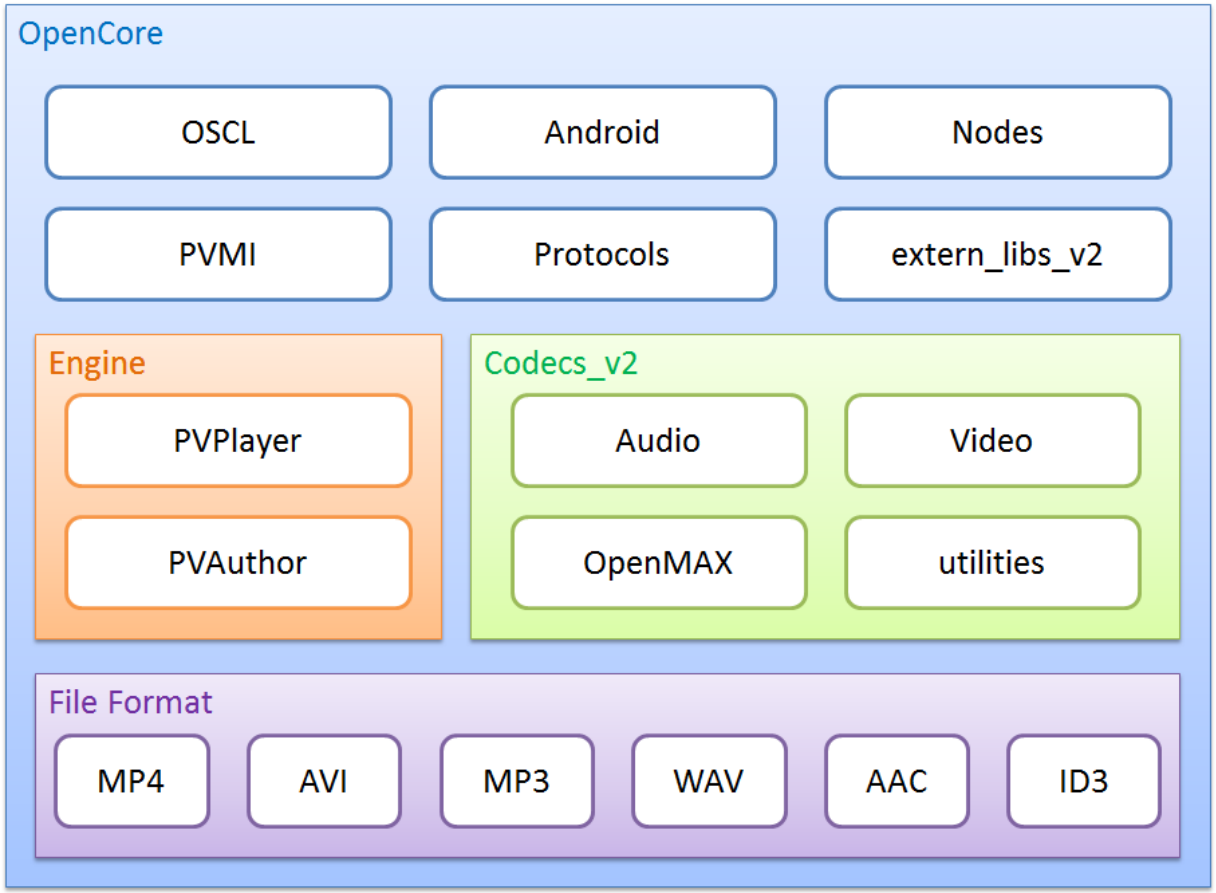
- Android Multimedia Framework?
  - Media framework in Android
    - ***OpenCORE(before Android 2.2)***
    - ***Stagefright(after Android 2.3)***

Android Multimedia Framework

- OpenCORE
  - Interfaces for third-party and hardware media codecs, input and output devices, and content policies
  - Media playback, streaming, downloading, and progressive playback, including 3rd Generation Partnership
  - Ensure robustness and stability
  - But the framework is too complicated to maintain

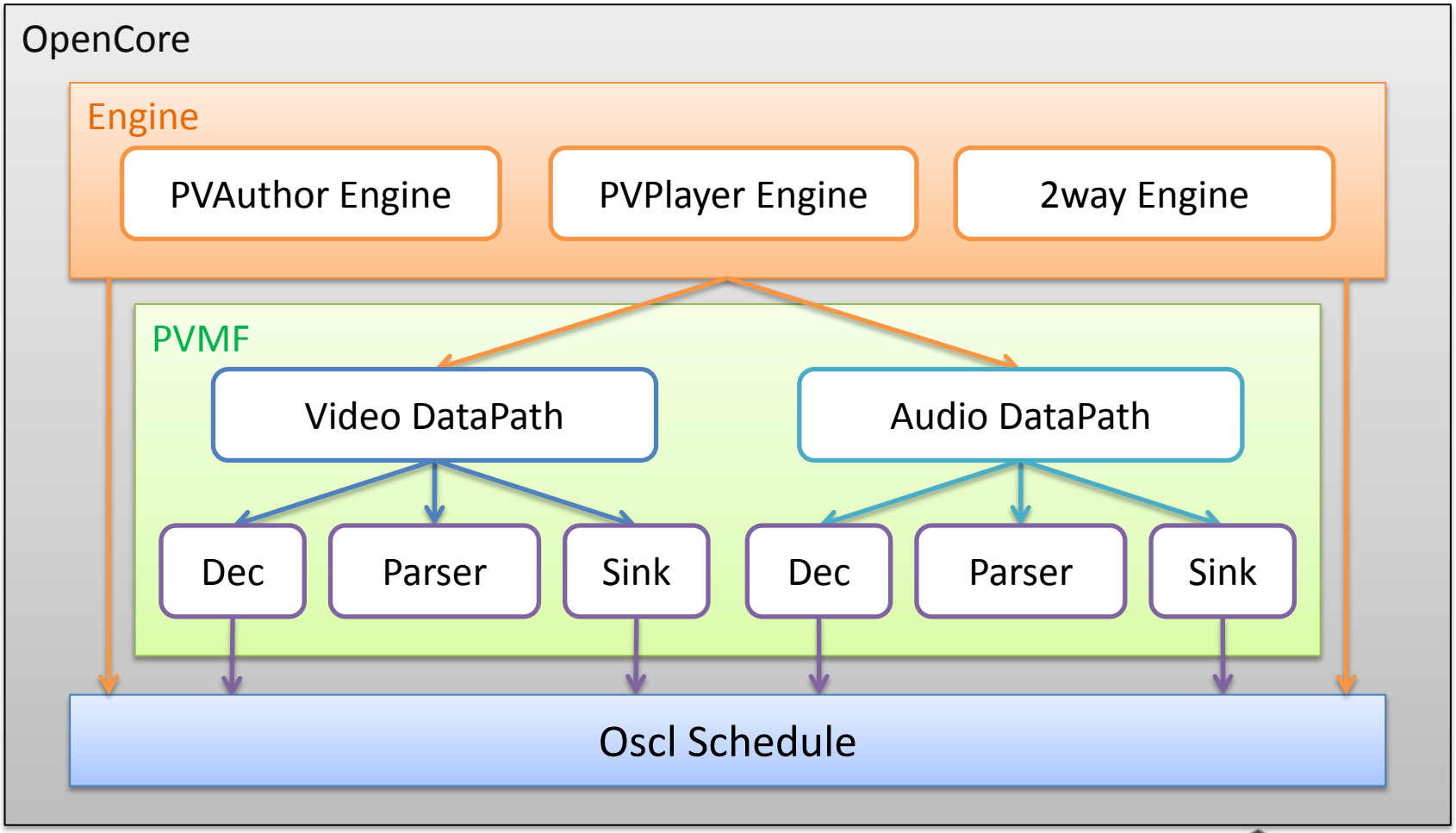
Android Multimedia Framework

- OpenCORE media framework



Android Multimedia Framework

- OpenCORE workflow



Android Multimedia Framework

- OpenCORE source tree (simplied)
  - **<android_src>/external/opencore**

|-- engine :

- Resolve the events from the application layer (play, pause, record, stop)
- Maintain the state machine of player
- Invoke the nodes

|--nodes :

- Audio/video decode node, parser node, encode node, sink node
- Media input/output node
- Streaming protocols
- Each node has its own state machine

|--pvmi :

- Include media recognizer

|--fileformats :

- Invoked by parser node, get information of media

|-- codec_v2 :

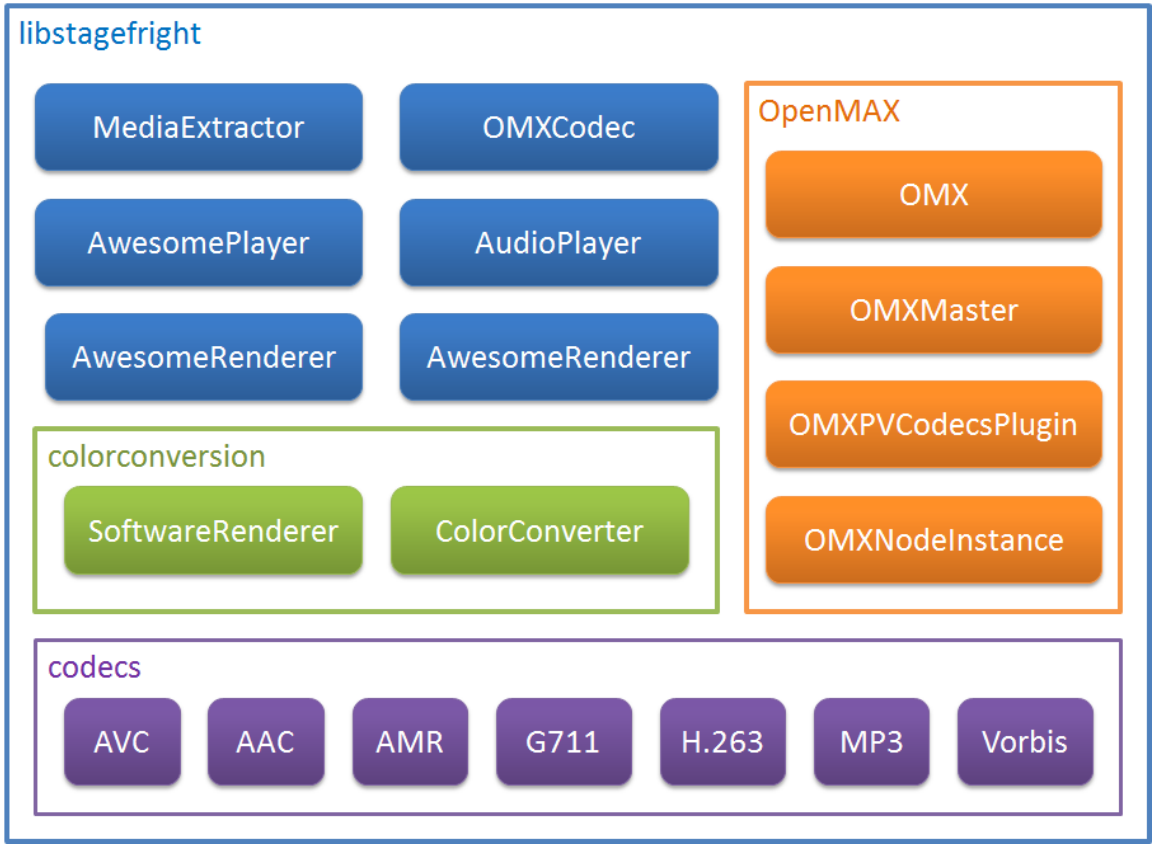
- Implementation of OpenMAX components and media codecs

Android Multimedia Framework

- Stagefright
  - An lightweight media framework released in Android 2.0 and used after Android 2.3
  - Easy to add the codec into Stagefright as an OpenMAX component
  - Or, implement the media extractor and media encoder/decoder

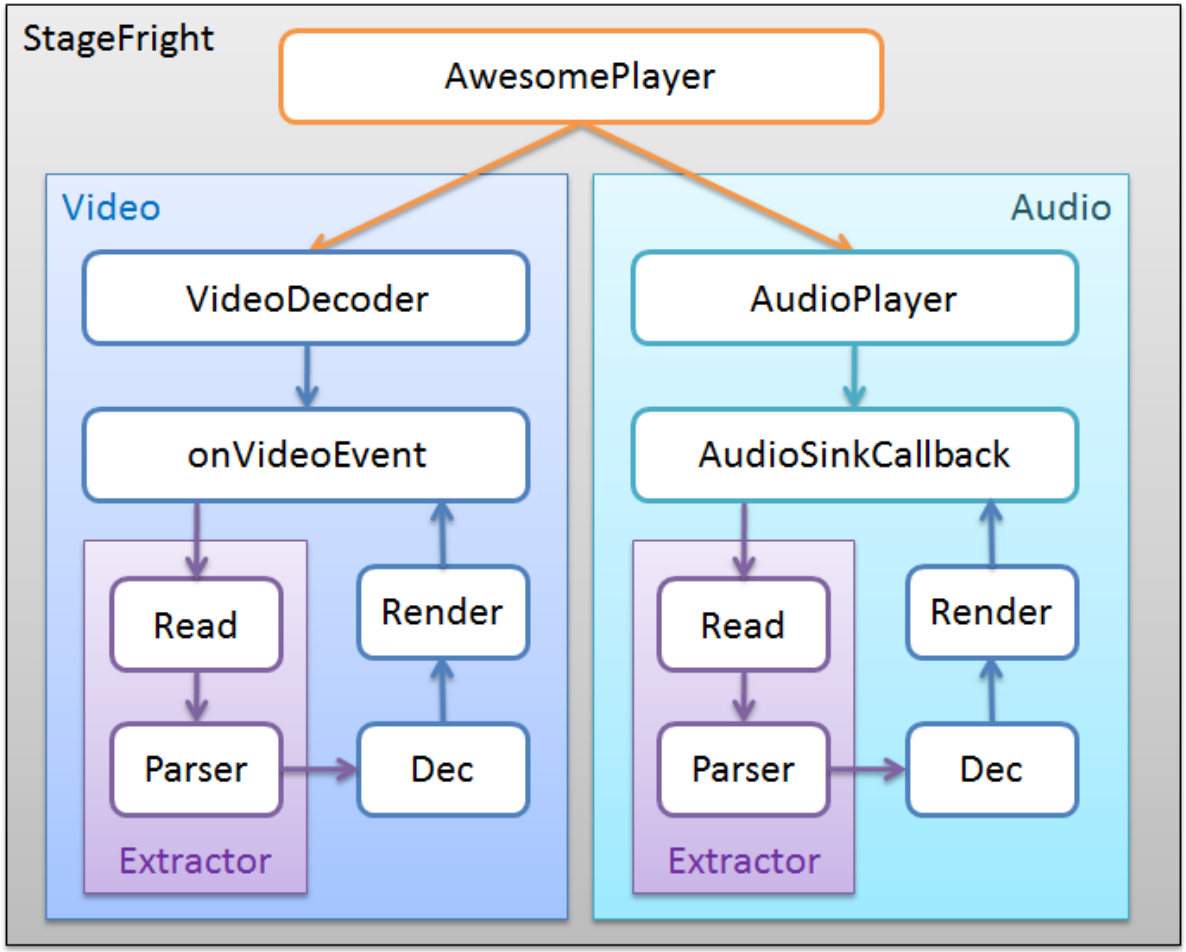
Android Multimedia Framework

- Stagefright media framework
  - An lightweight media framework released in Android 2.0 and replace opencore in Android 2.3



Android Multimedia Framework

- Stagefright workflow



Android Multimedia Framework

- Stagefright source tree(simplied)
  - **<android_src>/framework/base/media/libstagefright**

|-- AwesomePlayer.cpp

- Resolve the events from the application layer (play, pause, record, stop)
- Create AudioPlayer thread
- Manage the video buffer for decoding/encoding/rendering
- Manage Audio/Video Synchronous

|--AudioPlayer.cpp

- Manage the audio buffer for decoding/encoding and output to the audio device

|--*Extractor.cpp

- Recognize the file format of media stream
- Manage the buffer filling for decoding/encoding

|--*Writer.cpp

- Encode the media raw data to the requested media format

|-- codec :

- Raw codec for decoding and encoding

|-- colorconversion :

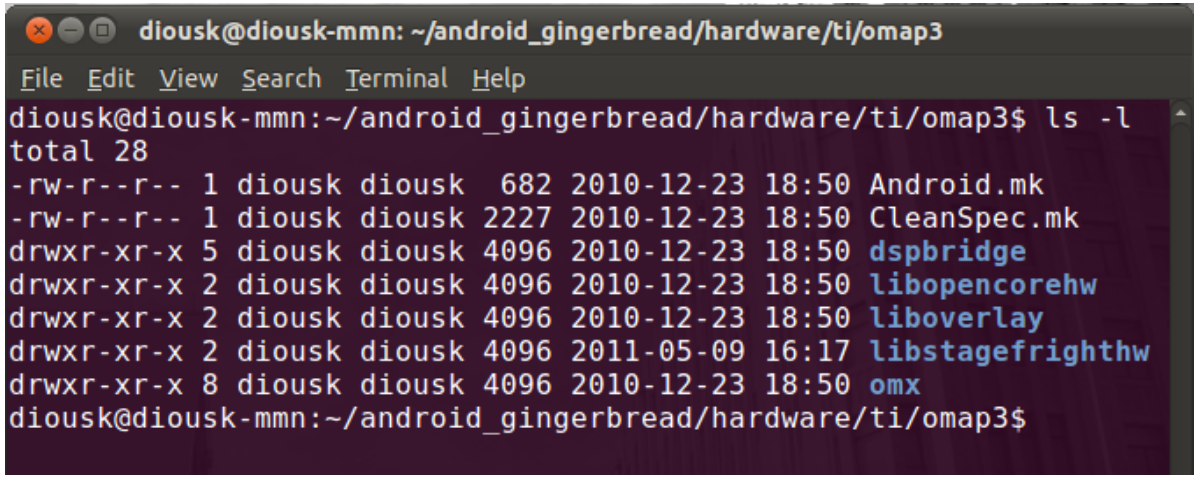
- Conver the YUV raw data to RGB for display

|--omx :

- Compatible for openmax components in opencore

Android Multimedia Framework

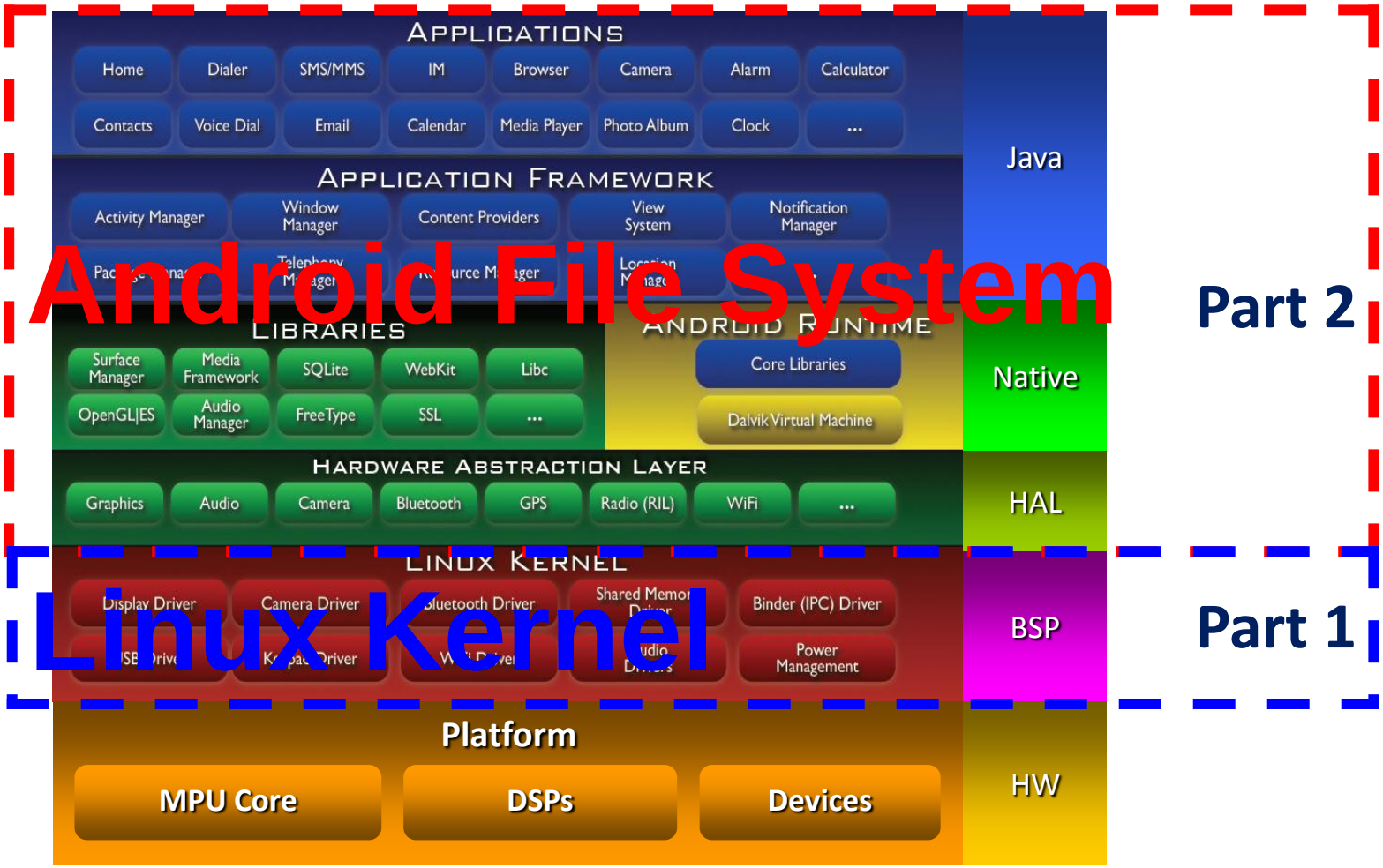
- In order to fit Android media framework , TI implement the OpenMAX Integration Layer for processing media decode/encode via DSP in Android
- We can see the implementation in android source code
  - <android_src>/hardware/ti/omap3



Connections between opencore or stagefright and DSP bridge

- *Introduction to Android*
- *Android Architecture*
- *Android Multimedia Framework*
- ***Android Porting***
- *Android start-up programming*
- *LAB : Mount Android Filesystem*

Android Porting



Android Porting

1. Get Android Source and setup develop environment

1. Install tool:
   - Git and Python (get android source tool)
   - JDK6 for 2.3+ or JDk5 for 2.1- (compiler application, framework)
   - zlib1g-dev libc6-dev lib32ncurses5-dev ia32-libs, x11proto-core-dev libx11-dev lib32readline5-dev lib32z-dev (compiler native libraries)
2. Get android source

2. Build LinuxKernel

1. According your platform to configure kernel
2. Porting Android Driver to your kernel
3. Build kernel image

Generate uImage

3. Build Android Filesystem

1. Compiler Android source
2. Modify initial produce

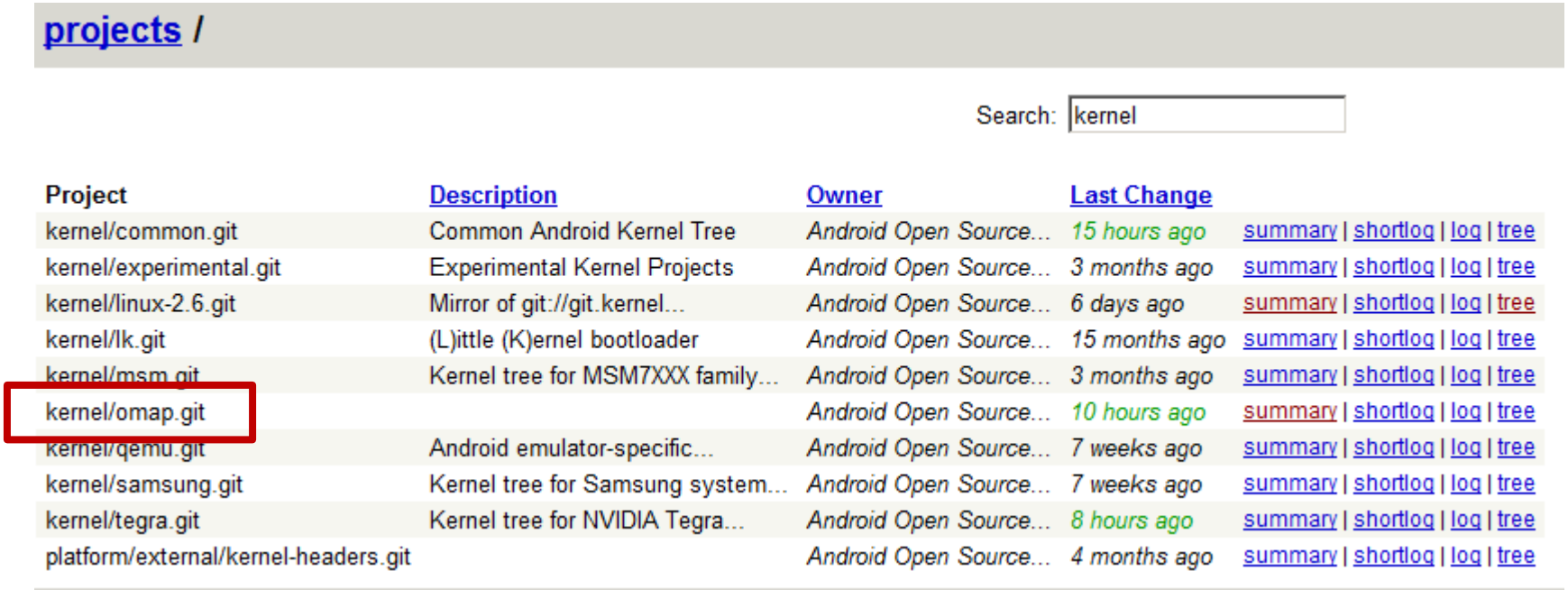
Generate file system image(yaffs2)

Android Porting

- What do we need ?
  - Android filesystem
  - Linux kernel
- Android Filesystem
  - office
    - http://source.android.com/source/download.html
  - Third-party android team
    - http://code.google.com/p/0xdroid/
      - **Support devkit8000 , beagleboard**
    - http://code.google.com/p/rowboat/
      - **Support most of TI platforms for android**
- Linux kernel
  - linux-02.01.03.11(version 2.6.29 for TI Platform)
  - To be modified

Android Porting

- **Get Android kernel???**
  1. Go to http://android.git.kernel.org/
  2. Search “kernel”



projects /

Search: kernel

Project	Description	Owner	Last Change	
kernel/common.git	Common Android Kernel Tree	Android Open Source...	15 hours ago	summary | shortlog | log | tree
kernel/experimental.git	Experimental Kernel Projects	Android Open Source...	3 months ago	summary | shortlog | log | tree
kernel/linux-2.6.git	Mirror of git://git.kernel...	Android Open Source...	6 days ago	summary | shortlog | log | tree
kernel/lk.git	(L)ittle (K)ernel bootloader	Android Open Source...	15 months ago	summary | shortlog | log | tree
kernel/msm.git	Kernel tree for MSM7XXX family...	Android Open Source...	3 months ago	summary | shortlog | log | tree
kernel/omap.git		Android Open Source...	10 hours ago	summary | shortlog | log | tree
kernel/qemu.git	Android emulator-specific...	Android Open Source...	7 weeks ago	summary | shortlog | log | tree
kernel/samsung.git	Kernel tree for Samsung system...	Android Open Source...	7 weeks ago	summary | shortlog | log | tree
kernel/tegra.git	Kernel tree for NVIDIA Tegra...	Android Open Source...	8 hours ago	summary | shortlog | log | tree
platform/external/kernel-headers.git		Android Open Source...	4 months ago	summary | shortlog | log | tree

Android Porting

- **Get Android kernel???**
  1. Go to http://android.git.kernel.org/
  2. Search “kernel”
  3. ***Host$ git clone git://android.git.kernel.org/kernel/omap.git***

```
Host$ git clone git://android.git.kernel.org/kernel/omap.git
Initialized empty Git repository in /home/mad/kernel_source/omap/.git/
remote: Counting objects: 236683
....
```

  4. ***Host$ git branch -a***

Android Porting

- **Get Android kernel???**
  - ***4. Host$ git branch -a***

```
mad@mad-desktop:~/kernel_source/test2/omap$ git branch -a
* (no branch)
  linux-omap-2.6.38
  origin/android-omap-2.6.38
  origin/android-omap-2.6.39
  origin/HEAD
  origin/android-omap-2.6.38
  origin/android-omap-2.6.39
  origin/android-omap-3.0
  origin/archive/android-omap-2.6.29
  origin/archive/android-omap-2.6.29-eclair
  origin/archive/android-omap-2.6.32
  origin/linux-omap-2.6.38
  origin/linux-omap-2.6.39
  origin/linux-omap-3.0
  ......
  ......
```

heads

11 hours ago	android-omap-3.0	shortlog | log | tree
11 hours ago	linux-omap-3.0	shortlog | log | tree
13 hours ago	linux-omap-pm-3.0	shortlog | log | tree
21 hours ago	users/simonwilson/linux-omap-audio-3.0	shortlog | log | tree
5 days ago	linux-omap-dss-3.0	shortlog | log | tree
12 days ago	android-omap-2.6.39	shortlog | log | tree
3 weeks ago	linux-omap-mm-3.0	shortlog | log | tree
7 weeks ago	linux-omap-audio-3.0	shortlog | log | tree
7 weeks ago	linux-omap-2.6.39	shortlog | log | tree
7 weeks ago	linux-omap-mm-2.6.39	shortlog | log | tree
8 weeks ago	linux-omap-dss-2.6.39	shortlog | log | tree
2 months ago	sandbox/ccross/linux-omap-2.6.39-pm	shortlog | log | tree
2 months ago	linux-omap-audio-2.6.39	shortlog | log | tree
2 months ago	sandbox/ccross/linux-omap-4460-2.6.39	shortlog | log | tree
4 months ago	android-omap-2.6.38	shortlog | log | tree
4 months ago	linux-omap-2.6.38	shortlog | log | tree

...

Android Porting

- **Get Android kernel???**
  5. ***Host$ git checkout origin/android-omap-3.0***
  6. ***Host$ Make menuconfig ARCH=arm***
  7. ***Configure your kernel source and build image***

heads

11 hours ago	**android-omap-3.0**	shortlog | log | tree
11 hours ago	**linux-omap-3.0**	shortlog | log | tree
13 hours ago	**linux-omap-pm-3.0**	shortlog | log | tree
21 hours ago	**users/simonwilson/linux-omap-audio-3.0**	shortlog | log | tree
5 days ago	**linux-omap-dss-3.0**	shortlog | log | tree
12 days ago	**android-omap-2.6.39**	shortlog | log | tree
3 weeks ago	**linux-omap-mm-3.0**	shortlog | log | tree
7 weeks ago	**linux-omap-audio-3.0**	shortlog | log | tree
7 weeks ago	**linux-omap-2.6.39**	shortlog | log | tree
7 weeks ago	**linux-omap-mm-2.6.39**	shortlog | log | tree
8 weeks ago	**linux-omap-dss-2.6.39**	shortlog | log | tree
2 months ago	**sandbox/ccross/linux-omap-2.6.39-pm**	shortlog | log | tree
2 months ago	**linux-omap-audio-2.6.39**	shortlog | log | tree
2 months ago	**sandbox/ccross/linux-omap-4460-2.6.39**	shortlog | log | tree
4 months ago	**android-omap-2.6.38**	shortlog | log | tree
4 months ago	**linux-omap-2.6.38**	shortlog | log | tree

shortlog

2011-03-29	*Erik Gilling*	**gpu: pvr: remove reference to dbgdrv in makefile** linux-omap-2.6.38
2011-03-29	*Vikram Pandita*	**OMAP4: SGX-KM: Enable SGX initialisation**
2011-03-29	*Tony Lofthouse*	**OMAP: SYSLINK: cacheflush (modified)**
2011-03-29	*Tony Lofthouse*	**OMAP: SGX-KM: Port PVR services to .38**
2011-03-29	*Hemant Hariyani*	**Kernel changes for hwmod and omap_device initialization...**
2011-03-29	*Imagination...*	**gpu: pvr: Update to DDK 1.7.17.4142**
2011-03-29	*Gustavo Diaz...*	**SGX: UDD: Changing the early suspend registration level**
2011-03-29	*Gustavo Diaz...*	**SGX: UDD: Create sysfs entry to allow ignoring the...**
2011-03-29	*Gustavo Diaz...*	**SGX: UDD: Use correct stride when TILER memory is used**
2011-03-29	*Tony Lofthouse*	**V4L2-GFX: (NEW) video capture driver for SGX texture...**
2011-03-29	*Rodrigo Obregon*	**SGX-KM: Add PDump build option**
2011-03-29	*Gustavo Diaz...*	**OMAP4: Virtual display: Add manual update support for...**
2011-03-29	*Gustavo Diaz...*	**SGX: Enable early suspend flag for all OMAP3/4 builds**
2011-03-29	*Gustavo Diaz...*	**SGX: UDD: Fix crash when cloning with FB sysfs entries**
2011-03-29	*Rodrigo Obregon*	**OMAP4: SGX-KM: Update DDK version to 1.6.16.4061**
2011-03-29	*Lajos Molnar*	**OMAP4: SGX: UDD: Fixed checking of manual update suppor...**

Android Porting

- Android kernel feature
  - Original linux kernel doesn't support to run android
  - Need to add some specific kernel modules for running its android operating system

Android Porting

- **Android kernel feature**
  - ***Configure kernel reference : Documentation/android.txt***

1.Required enabled config options

ANDROID_PARANOID_NETWORK
ASHMEM
CONFIG_FB_MODE_HELPERS
CONFIG_FONT_8x16
CONFIG_FONT_8x8
CONFIG_YAFFS_SHORT_NAMES_IN_RAM
DAB
EARLYSUSPEND
FB
FB_CFB_COPYAREA
FB_CFB_FILLRECT
FB_CFB_IMAGEBLIT
FB_DEFERRED_IO
FB_TILEBLITTING
HIGH_RES_TIMERS
INOTIFY
INOTIFY_USER
INPUT_EVDEV
INPUT_GPIO
INPUT_MISC
LEDS_CLASS
LEDS_GPIO
LOCK_KERNEL
LkOGGER
LOW_MEMORY_KILLER
MISC_DEVICES
NEW_LEDS
NO_HZ
POWER_SUPPLY
PREEMPT
RAMFS
RTC_CLASS
RTC_LIB
SWITCH
SWITCH_GPIO
TMPFS
UID_STAT
UID16
USB_FUNCTION
USB_FUNCTION_ADB
USER_WAKELOCK
VIDEO_OUTPUT_CONTROL
WAKELOCK
YAFFS_AUTO_YAFFS2
YAFFS_FS
YAFFS_YAFFS1
YAFFS_YAFFS2

2. Required disabled config options

CONFIG_YAFFS_DISABLE_LAZY_LOAD
DNOTIFY

3. Recommended enabled config options

ANDROID_PMEM
ANDROID_RAM_CONSOLE
ANDROID_RAM_CONSOLE_ERROR_CORRECTION
SCHEDSTATS
DEBUG_PREEMPT
DEBUG_MUTEXES
DEBUG_SPINLOCK_SLEEP
DEBUG_INFO
FRAME_POINTER
CPU_FREQ
CPU_FREQ_TABLE
CPU_FREQ_DEFAULT_GOV_ONDEMAND
CPU_FREQ_GOV_ONDEMAND
CRC_CCITT
EMBEDDED
INPUT_TOUCHSCREEN
I2C
I2C_BOARDINFO
LOG_BUF_SHIFT=17
SERIAL_CORE
SERIAL_CORE_CONSOLE

Android Porting

- Android kernel feature
  - Android Binder
    - The mechanism used to manage the inter-processes communication (IPC)
  - Ashmem : anonymous shared memory map
    - The mechanism to use/allocate the shared memory between processes communication

Modules for Android

Android Porting

Modules for Android

- Android kernel feature
  - Android Low memory killer
    - Used to kill process for more memory when the memory is not enough
  - Android PMEM : Physical memory map
    - Used to allocate the continuous memory for devices

Android Porting

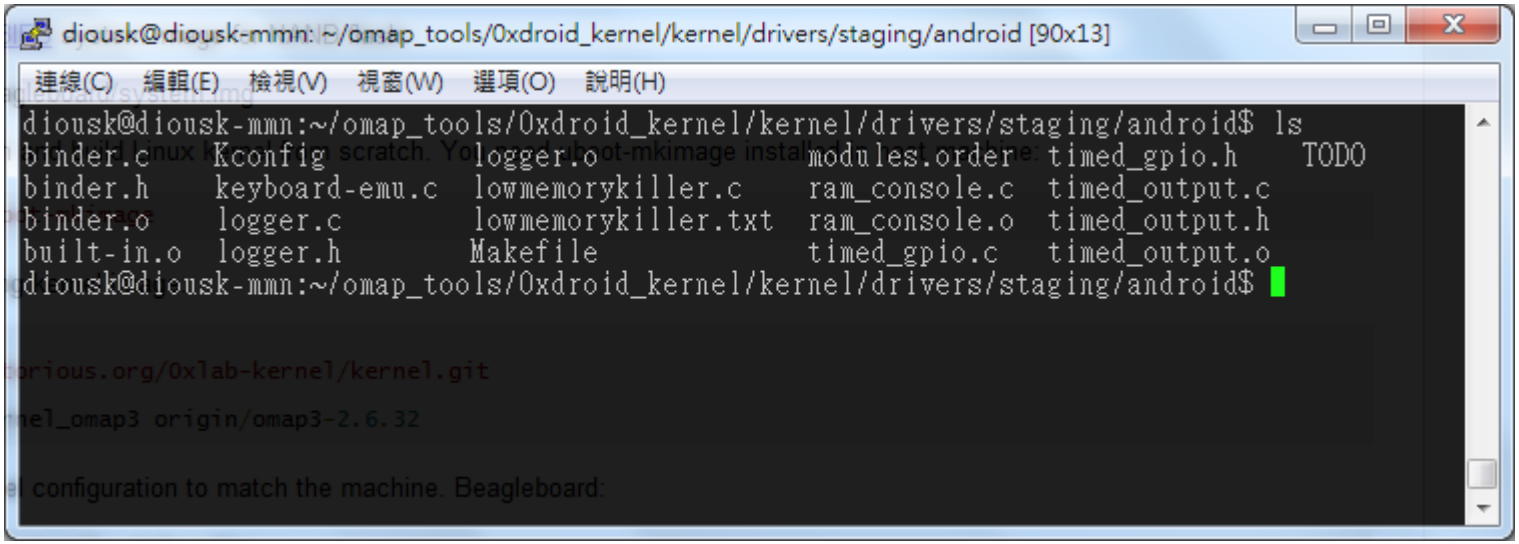
Modules for Android

- Android kernel feature
  - Android Logger
    - system logging facility , and support for the 'logcat' command for log from processes
  - Android timed gpio/output class
    - A mechanism to allow programs to access and manipulate gpio registers from user space.
  - Wakelock
    - used for power management
    - Holds machine awake on a per-event basis until wakelock is released

Android Porting

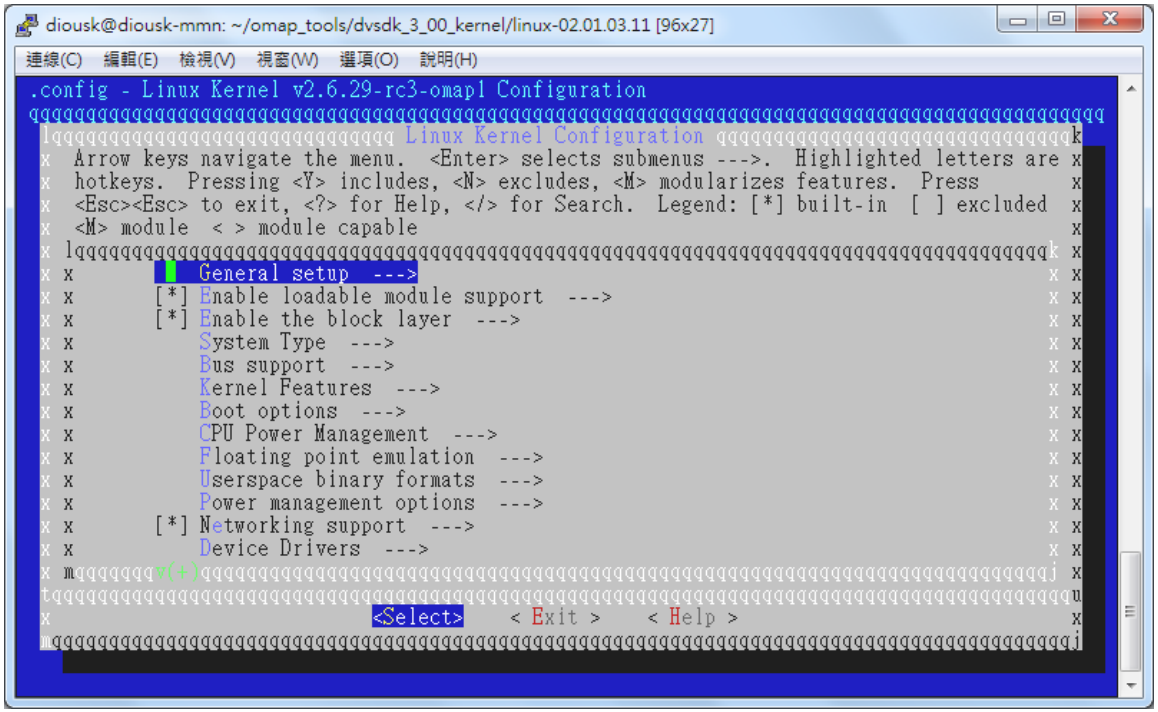
- Android kernel feature
  - Where to get those modules ?
  - The modules are located at <kernel_src>/driver/staging/android

Modules for Android



Android Porting

- Android kernel feature
  - All we have to do :
    - Move the modules source to our kernel source
    - Revise Kconfig for selection in “menuconfig”
    - Revice Makefile for building modules into kernel



MMN Lab.

Android Porting

- **Get Android filesystem source**
  - Precondition: git and repo
    - $sudo apt-get install git-core
    - $ curl -o ~/bin/repo http://android.git.kernel.org/repo
    - $ chmod a+x ~/bin/repo
  - Get android source code
    - $ mkdir beagle-donut
    - $ cd beagle-donut
      1. $ repo init -u git://android.git.kernel.org/platform/manifest.git
      2. $ repo init -u git://gitorious.org/0xdroid/manifest.git -b beagle-éclair
    - $ repo sync

PS:
Host$ repo init -u git://android.git.kernel.org/platform/manifest.git -b cupcake

Android Porting

- **Android Source Tree**
- |-- Makefile
  |-- bionic （bionic C libraries）
  |-- bootable （android boot stage initializtion）
  |-- build （rules to build corresponding to user setting makefile）
  |-- cts （Android test benchmark）
  |-- dalvik （dalvik JAVA virtual machine）
  |-- development（tools for development like usb driver）
  |-- external （open-source or third-party modules in android）
  |-- frameworks （android core framework）
  |-- hardware （third-party defined HAL code）
  |-- out （store the built filesystem and image）
  |-- packages （application built with android filesystem）
  |-- prebuilt （resources prebuilt for use of other tools like eclipse）
  |-- sdk （sdk and emulator）
  |-- system （system tool like adb,init）

Android Porting

- **Android filesystem feature**
  - Building filesystem from source
    - $cd <android_src>

•Host$ ***make –jx***

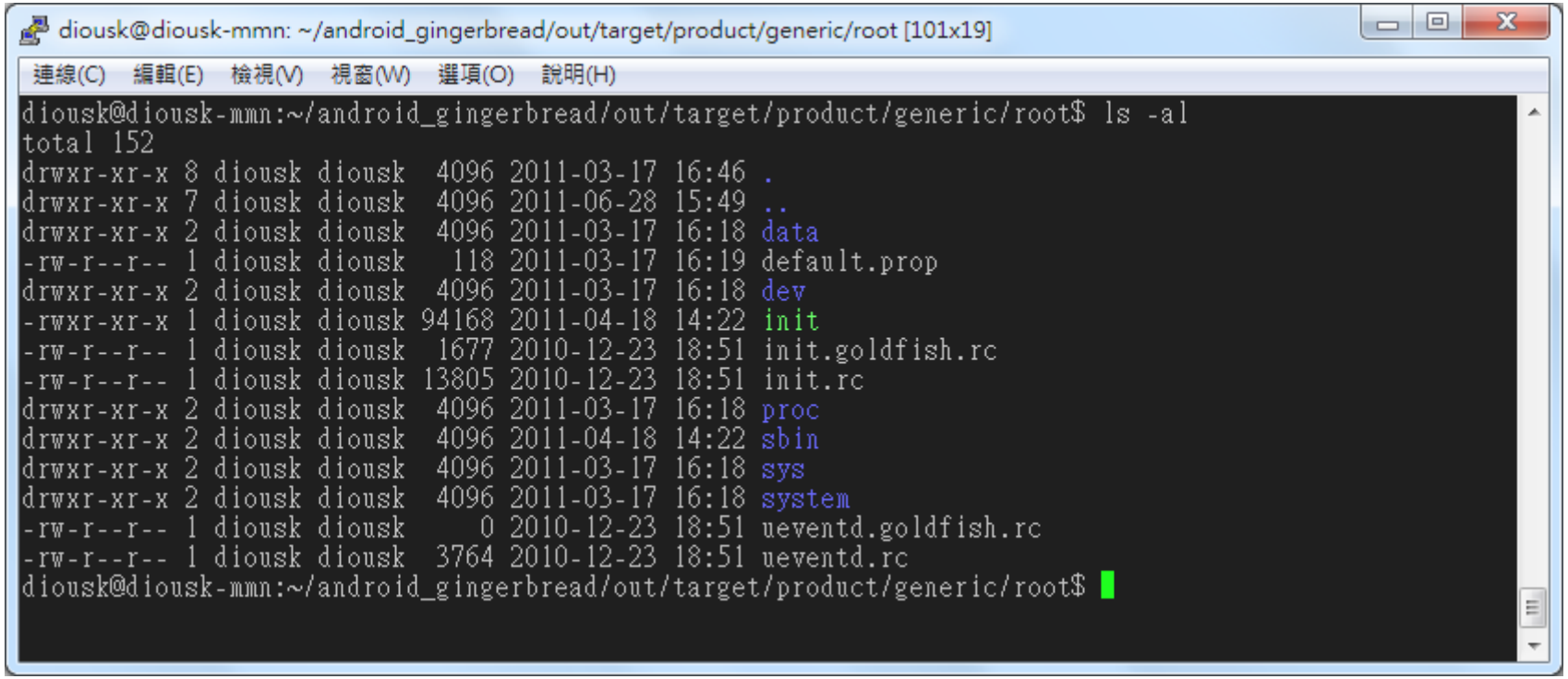
  - Take a break (first time : about 1 hours)

Android Porting

- **Android filesystem feature**
  - Generating filesystem
    - Generating path: out/target/product/generic
    - root/: rootfs and init program
    - system:/ include libraries, framework, application
    - data/: android runtime cache
    - ramdisk.img: cpio image of root/
    - system.img: yaffs2 image of system/
    - userdata.img: yaffs2 image of data/
  - Generating filesystem for nfs
    - $mkdir android_fs
    - $cp android_src/out/target/product/generic/root/* android_fs/
    - $cp android_src/out/target/product/generic/system/* android_fs/system/

Android Porting

- **Android filesystem feature**
  - After building android filesystem, it look likes:



Android Porting

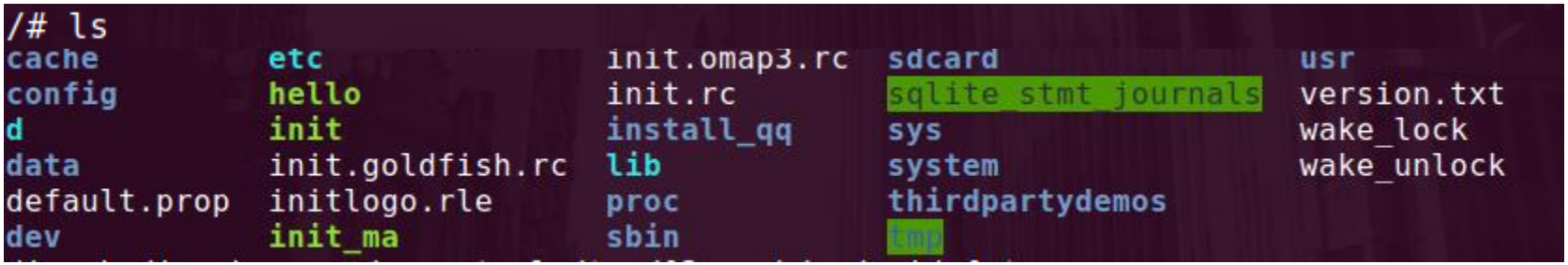
- Building busybox into Android for ease of use
  - Download the source code of busybus
    - http://www.busybox.net/
  - Build via cross compiler
    - $cd <busybox_src>
    - $make menuconfig
    - Build as static binary:

```
Busybox Settings --->
Build Options --->
[*] Build BusyBox as a static binary (no shared libs)
```

    - $make
    - $cp –rf _**install/bin**/* <android_fs>/system/bin
    - $cp –rf _**install/sbin**/* <android_fs>/system/sbin

Android Porting

- Building busybox into Android for ease of use
  - After porting busybox, we can using “tab” and see the colorful filesystem in android shell like host PC



Android Porting

- Android kernel feature
  - Rebuild kernel
    - $make uImage
  - Using the modified kernel to boot devkit8000
  - Try and error…
  - Until see the boot animation

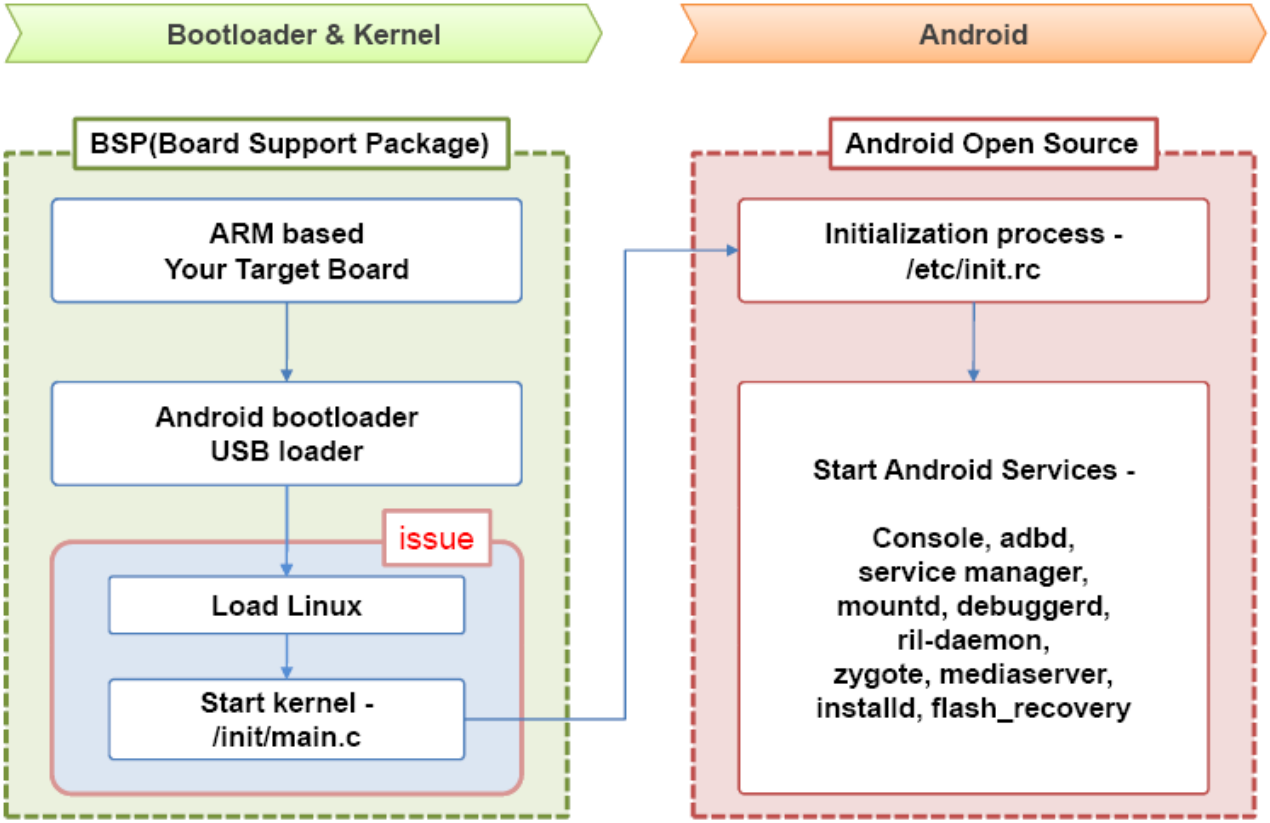
Android Porting

- envsetup.sh
  - ***Host$ Sourcd build/envsetup.sh***
    - **help : view command**
    - **m : Makes from the top of the tree**
    - **mm : Builds all of the modules in the current directory**
    - **mmm : Builds all of the modules in the supplied directories**
    - **croot : Changes directory to the top of the tree**
    - **godir : Go to the directory containing a file**
    - **printconfig : show configure**
    - **…**
    - **…**

- *Introduction to Android*
- *Android Architecture*
- *Android Multimedia Framework*
- *Android Porting*
- ***Android start-up programming***
- *LAB : Mount Android Filesystem*

Android start-up program

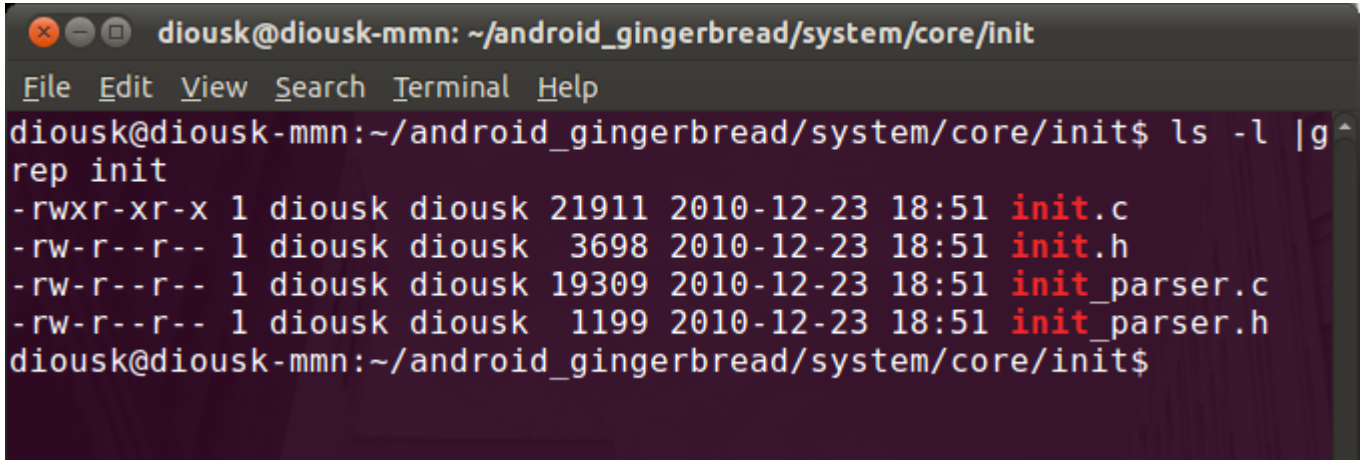
- What happened during Android booting stage ?(con'd)



From Korea Android Community- www.kandroid.org

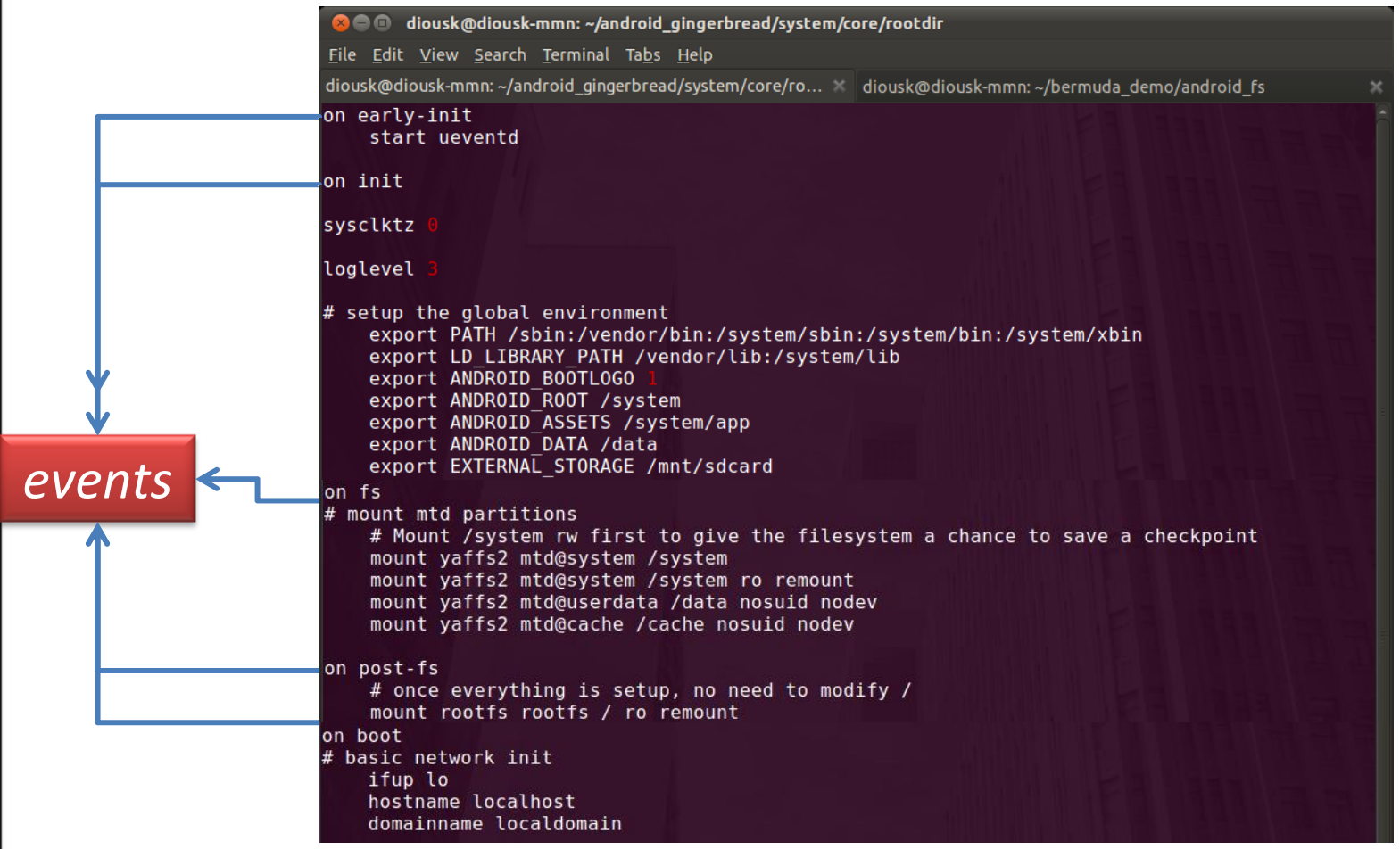
Android start-up program

- What happened during Android booting stage ?(con'd)
  - Kernel will execute "**init**" for starting android initialization
  - "**init**" will read the init.rc file to set up the environment variable or properties and start android services
- "**init**" is the first process after kernel started. The corresponding source code lies in <android_src>/system/core/init



Android start-up program

- Init.rc (under android_src/system/core/rootdir/init.rc)



Android start-up program

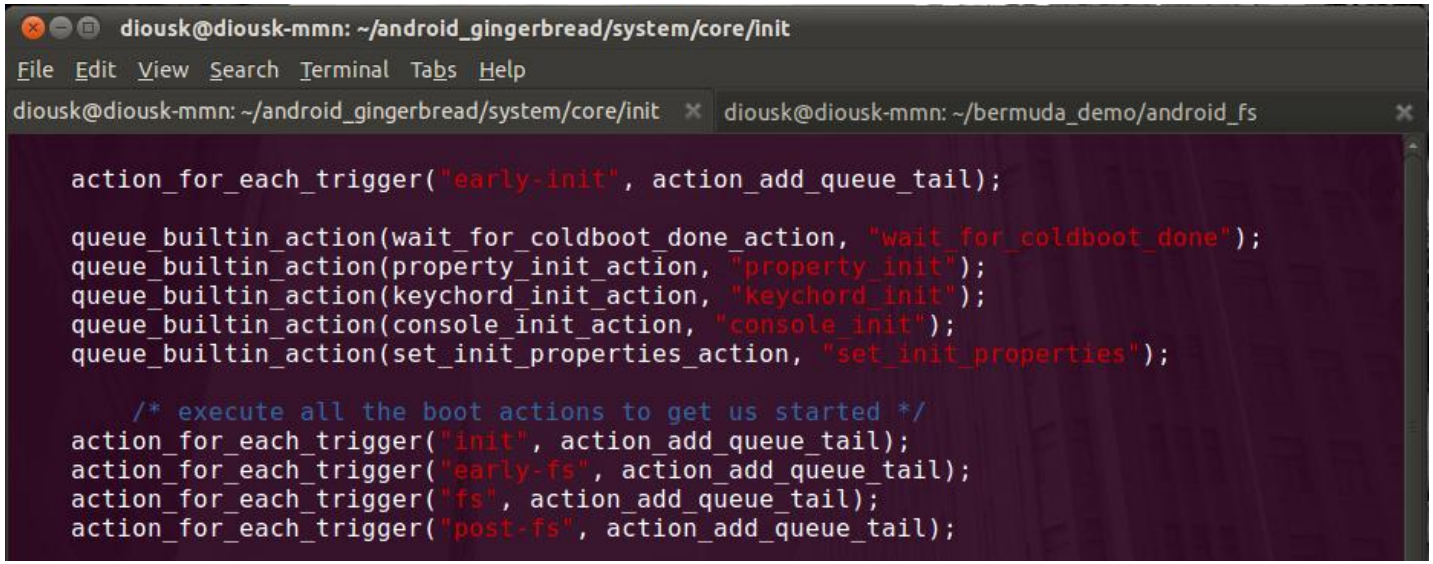
- "**init**" does the following tasks step by step:
  - 1.Initialize log system.
  - 2.Parse /init.rc
  - 3.Execute ***early-init action*** parsed in step 2.

<init.c>

```
int main(int argc, char **argv)
{
    INFO("reading config file\n");
    init_parse_config_file("/init.rc");

    /* pull the kernel commandline and ramdisk properties file in */
    import_kernel_cmdline(0);

    get_hardware_name(hardware, &revision);
    snprintf(tmp, sizeof(tmp), "/init.%s.rc", hardware);
    init_parse_config_file(tmp);

    action_for_each_trigger("early-init", action_add_queue_tail);
```

Android start-up program

- "**init**" does the following tasks step by step(con'd):
  - 4.Device specific initialize. For example, make all device node in /dev
  - 5.Initialize property system. Actually the property system is working as a share memory. Logically it looks like a registry under Windows system.
  - 6.Execute ***init action*** in the two files parsed in step 2.



Android start-up program

- "**init**" does the following tasks step by step(con'd):
  - 7.Start property service.
  - 8.Execute ***early-boot and boot actions*** in the two files parsed in step 2.
  - 9.Execute property action in init.rc parsed in step 2.
  - 10.Enter into an indefinite loop to wait for device/property set/child process exit events.

```
/* execute all the boot actions to get us started */
action_for_each_trigger("early-boot", action_add_queue_tail);
action_for_each_trigger("boot", action_add_queue_tail);
```

Android start-up program

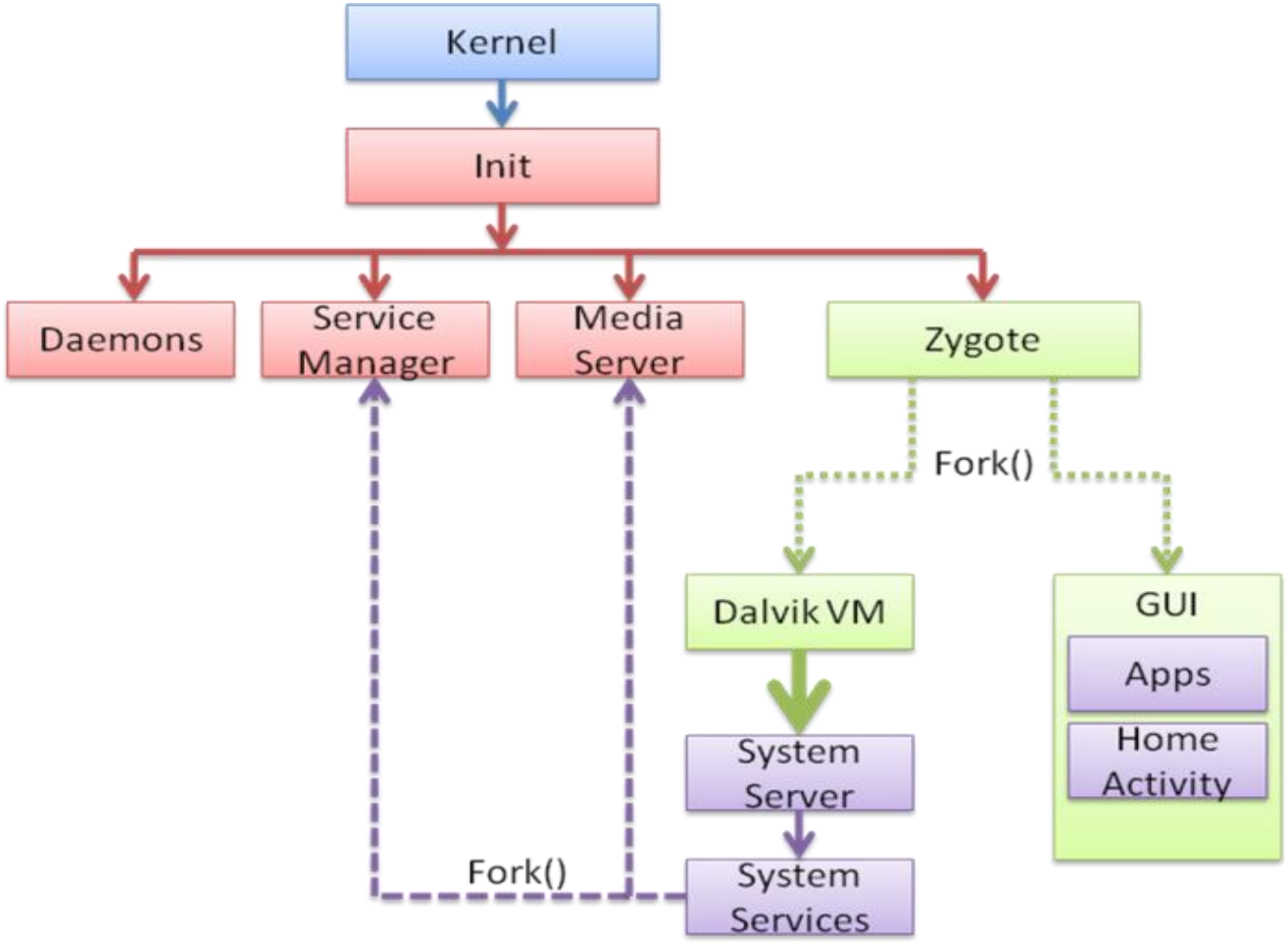
- After init process , there are two main functions (Zygote, System server)in booting:
  - Zygote does the following tasks step by step:
    - 1.Create JAVA VM.
    - 2.Register android native function for JAVA VM.
    - 3.Call the main function in the JAVA class named com.android.internal.os.ZygoteInit
      - **Call Zygote::forkSystemServer (implemented in dalvik/vm/native/dalvik_system_Zygote.c)to fork a new process.**
    - 4. call IPCThreadState::self()->joinThreadPool() to enter into service dispatcher.

Android start-up program

- SystemServer will start a new thread to start all JAVA services as follows:
- Core Services:
  - 1.Starting Power Manager
  - 2.Creating Activity Manager
  - 3.Starting Telephony Registry
  - 4.Starting Package Manager
  - 5.Set Activity Manager Service as System Process
  - 6.Starting Context Manager
  - 7.Starting System Context Providers
  - 8.Starting Battery Service
  - 9.Starting Alarm Manager
  - 10.Starting Sensor Service
  - 11.Starting Window Manager
  - 12.Starting Bluetooth Service
  - 13.Starting Mount Service

Android start-up program

- Booting diagram



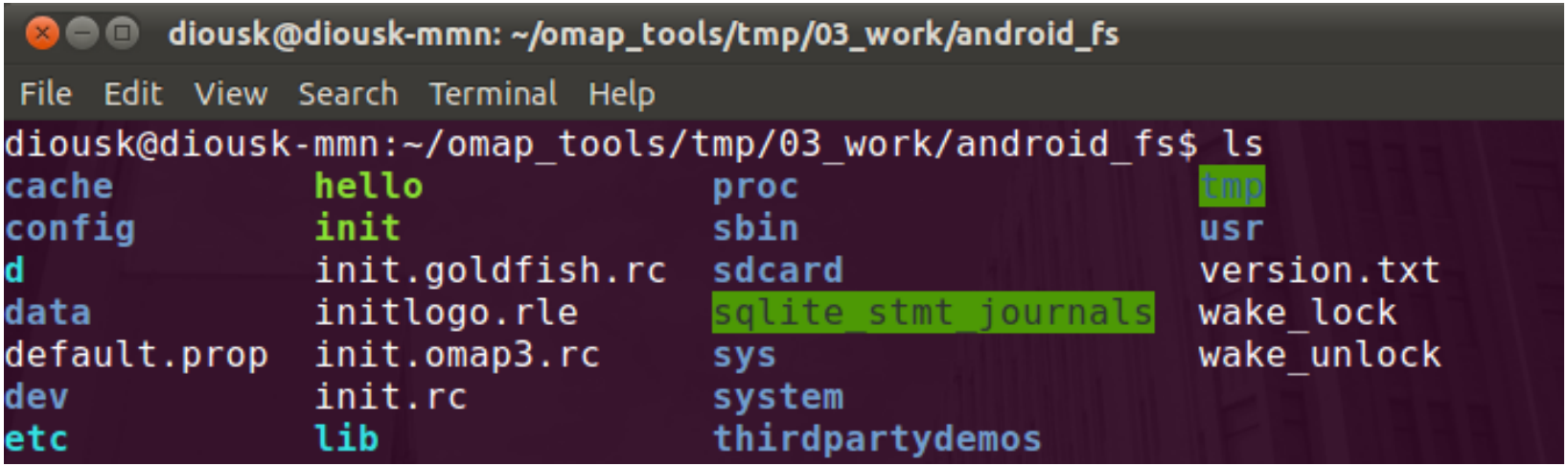
Android start-up program

- Service in android

- *Introduction to Android*
- *Android Architecture*
- *Android Multimedia Framework*
- *Android Porting*
- *Android start-up programming*
- ***LAB : Mount Android Filesystem***

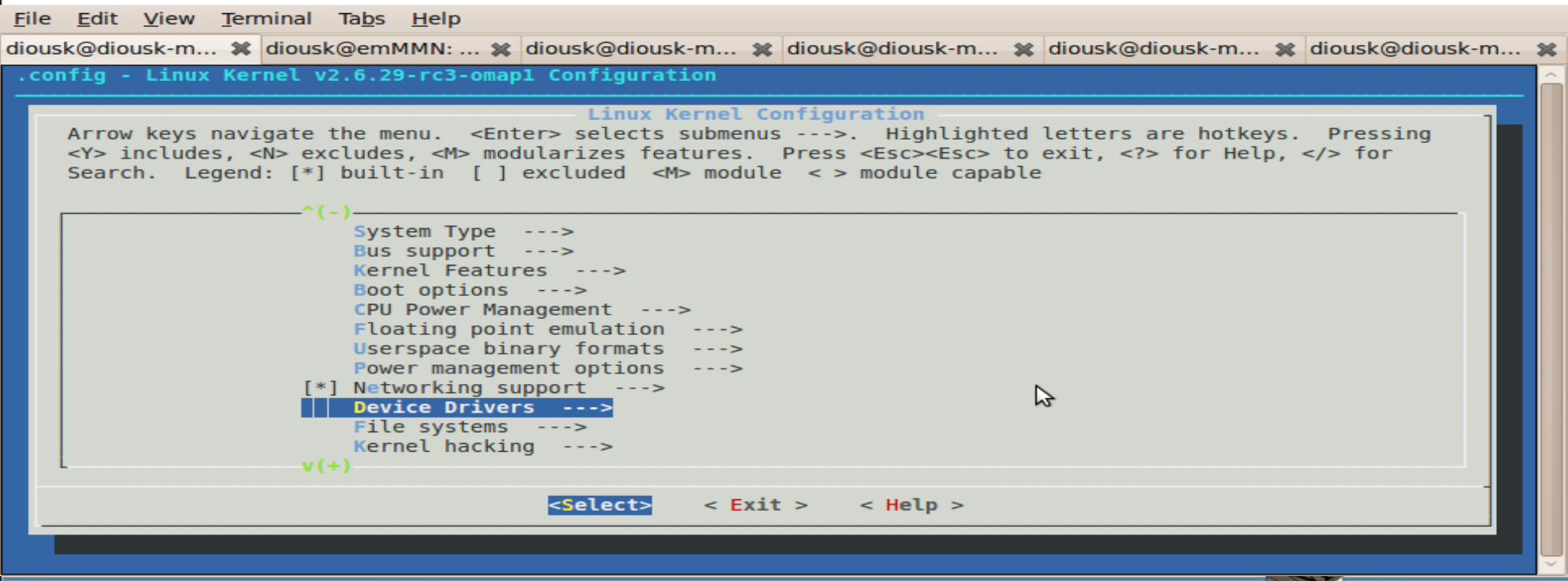
Step1. Uncompressing android fs

- $cd dvsdk_lab
- $sudo tar –zxvf android_fs_course.tgz
- $cd android_fs
- Check the filesystem



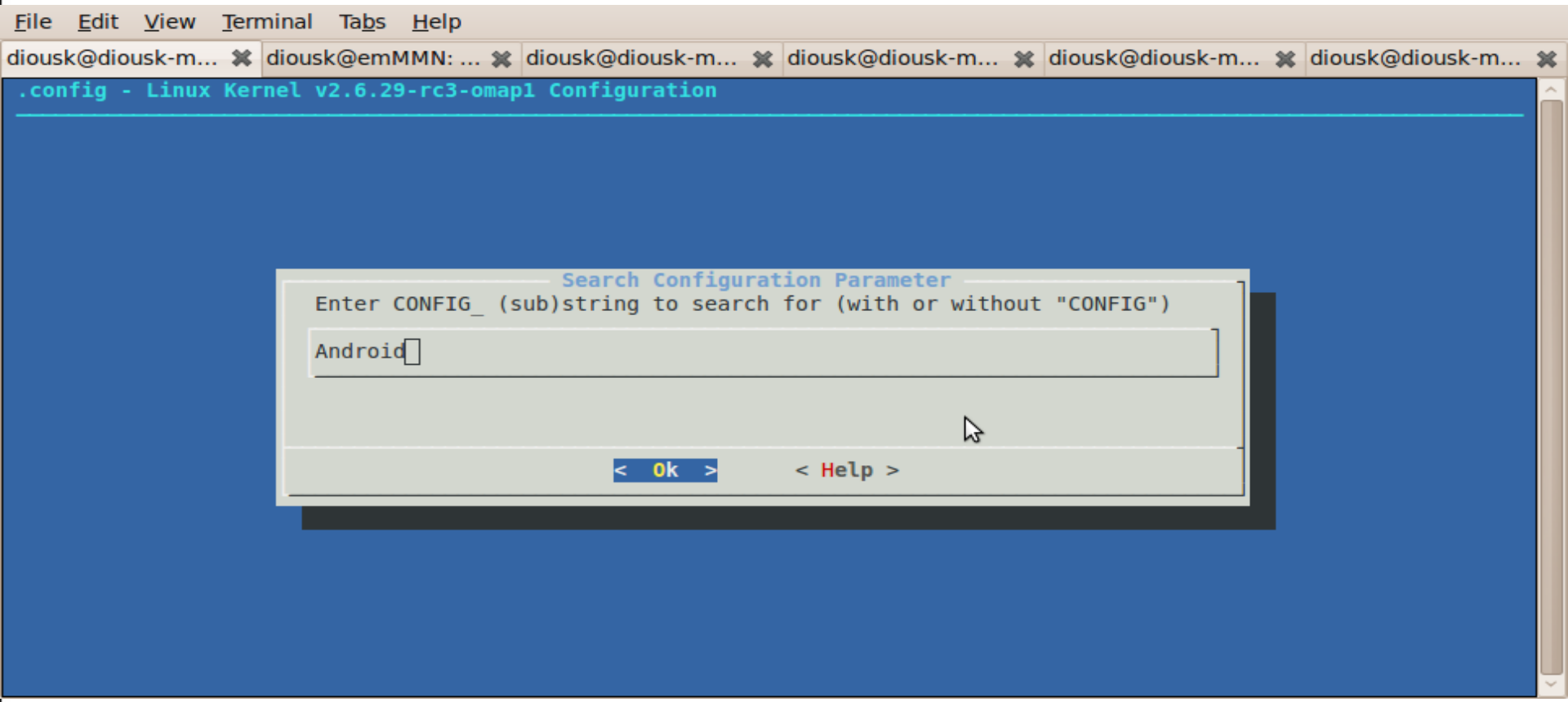
Step2.Activate the android drivers

- Select the android related drivers mentioned previously by menuconfig
  - $cd dvsdk_lab/dvsdk_kernel/linux-02-01-03-11
  - $make menuconfig
- Using “/” to find the needed drivers by keywords



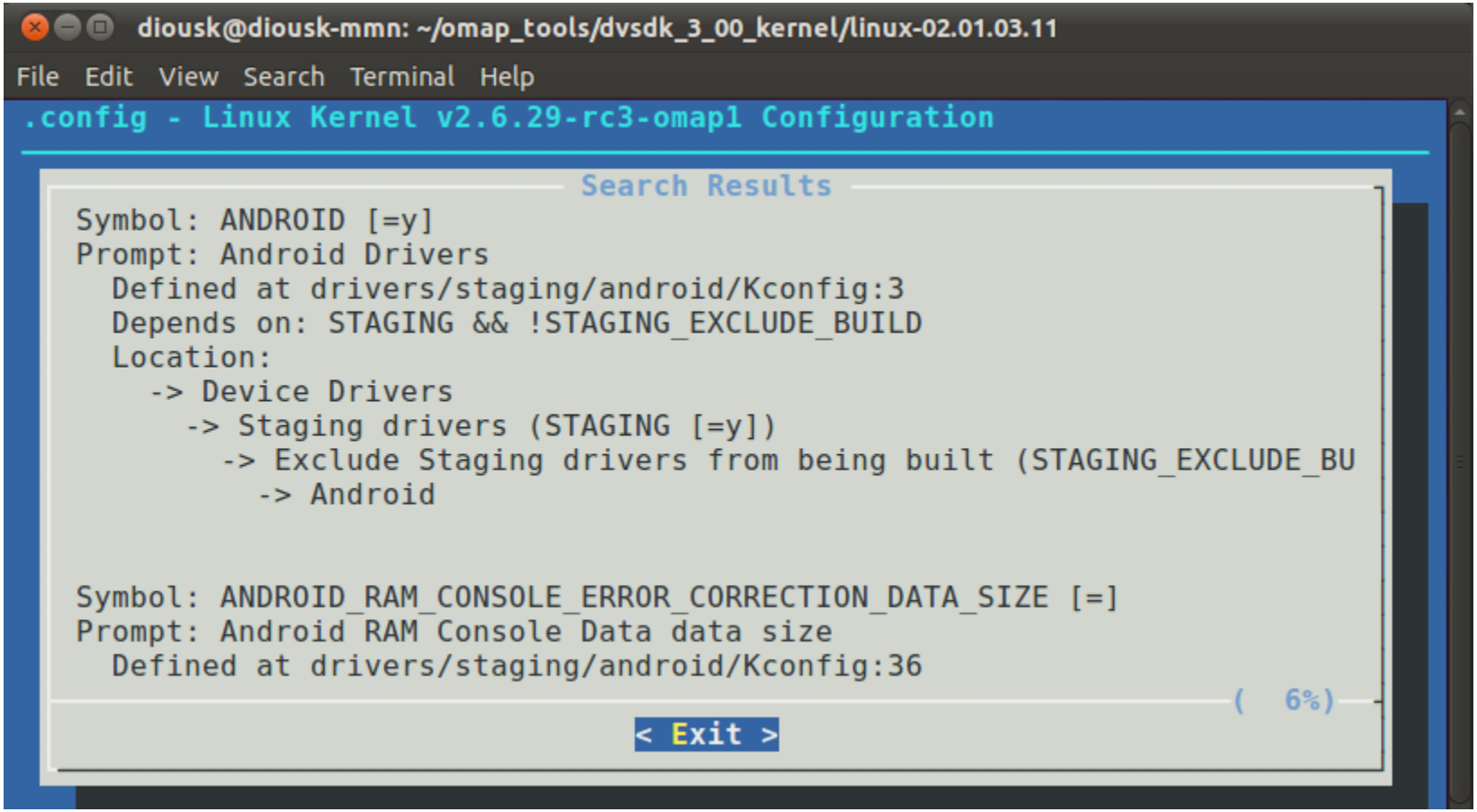
Step2.Activate the android drivers

- Keywords : **android**, pmem, ashmem



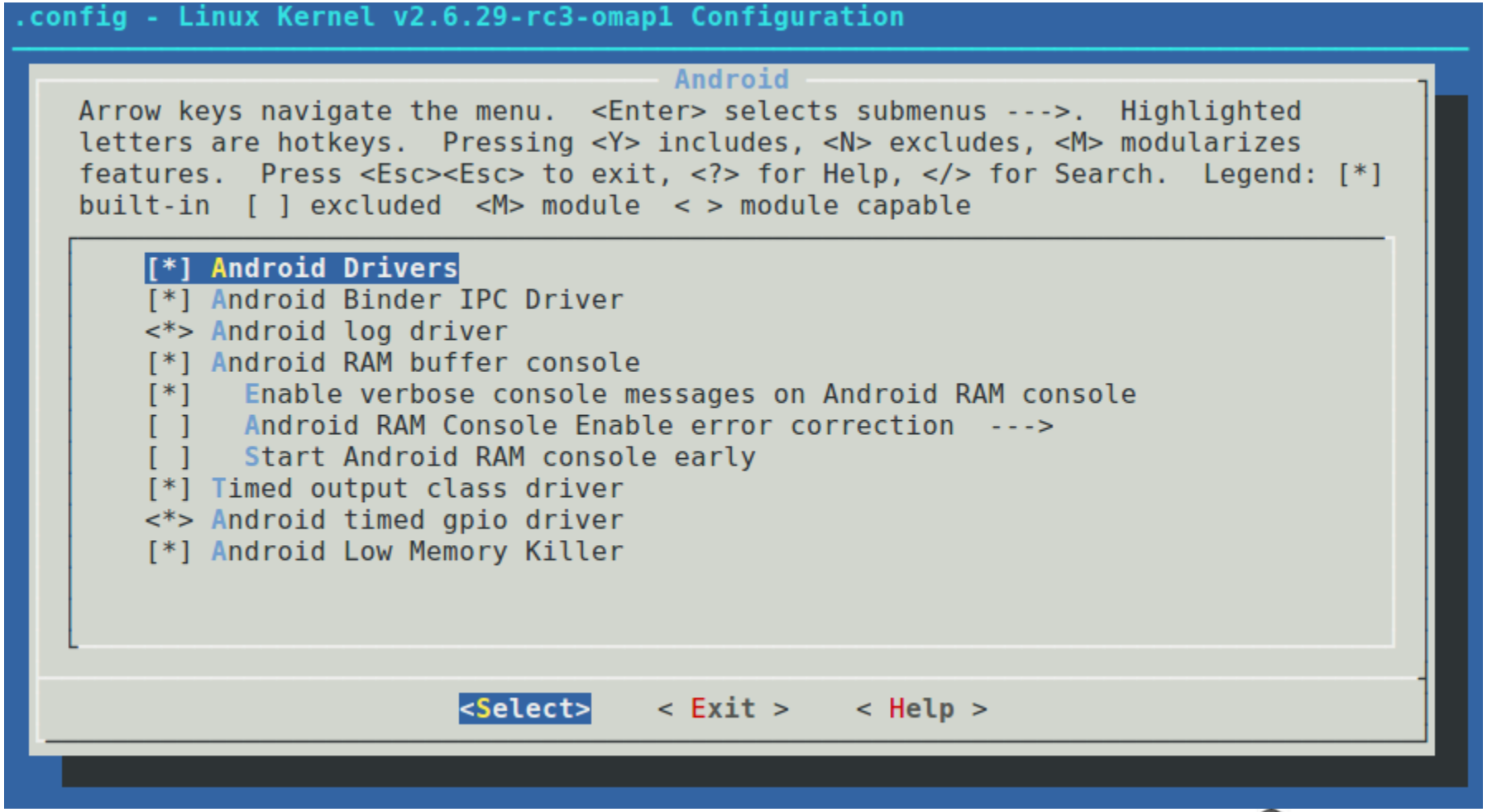
Step2.Activate the android drivers

- Searching result



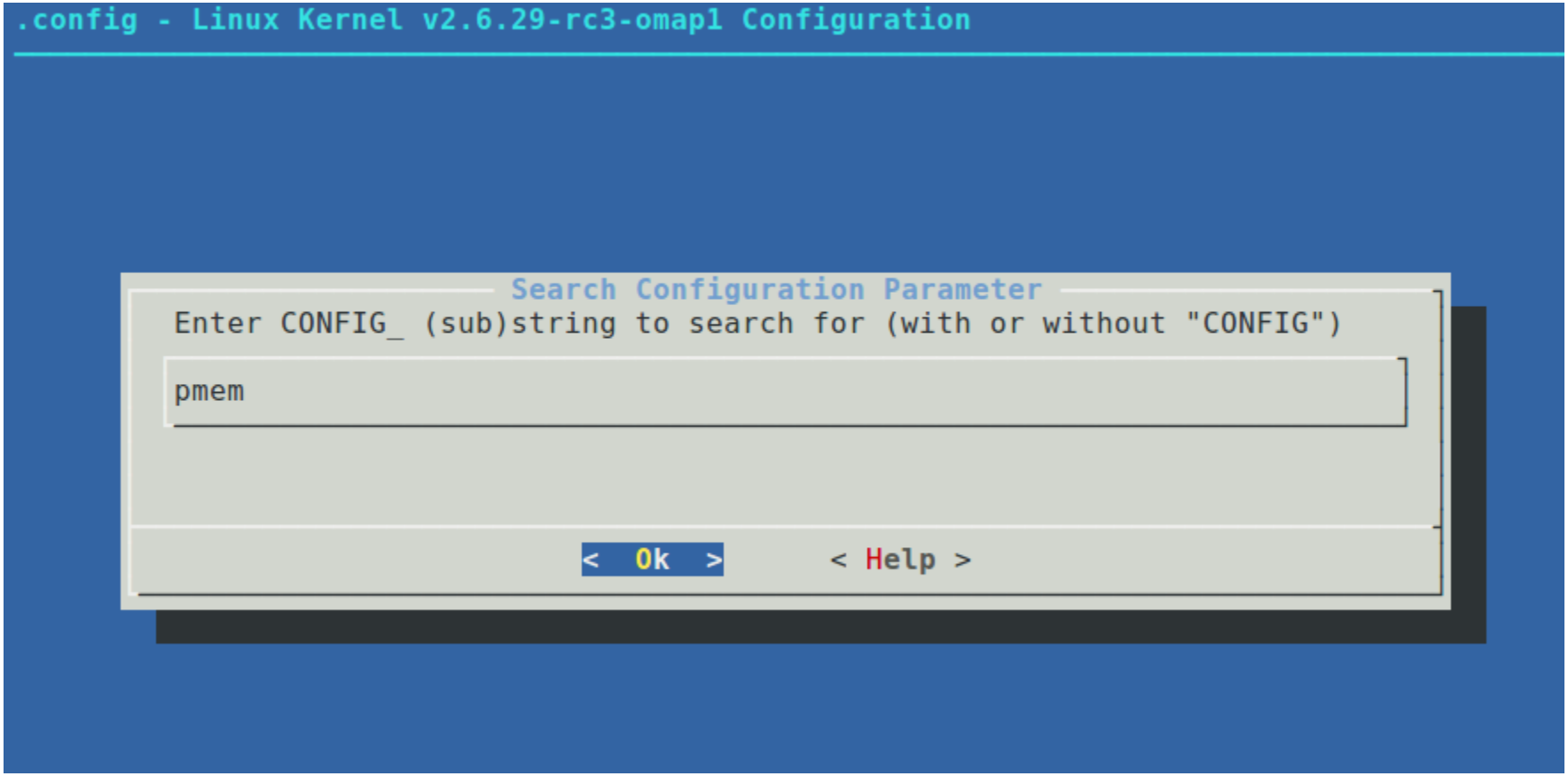
Step2.Activate the android drivers

- Build-in the android modules



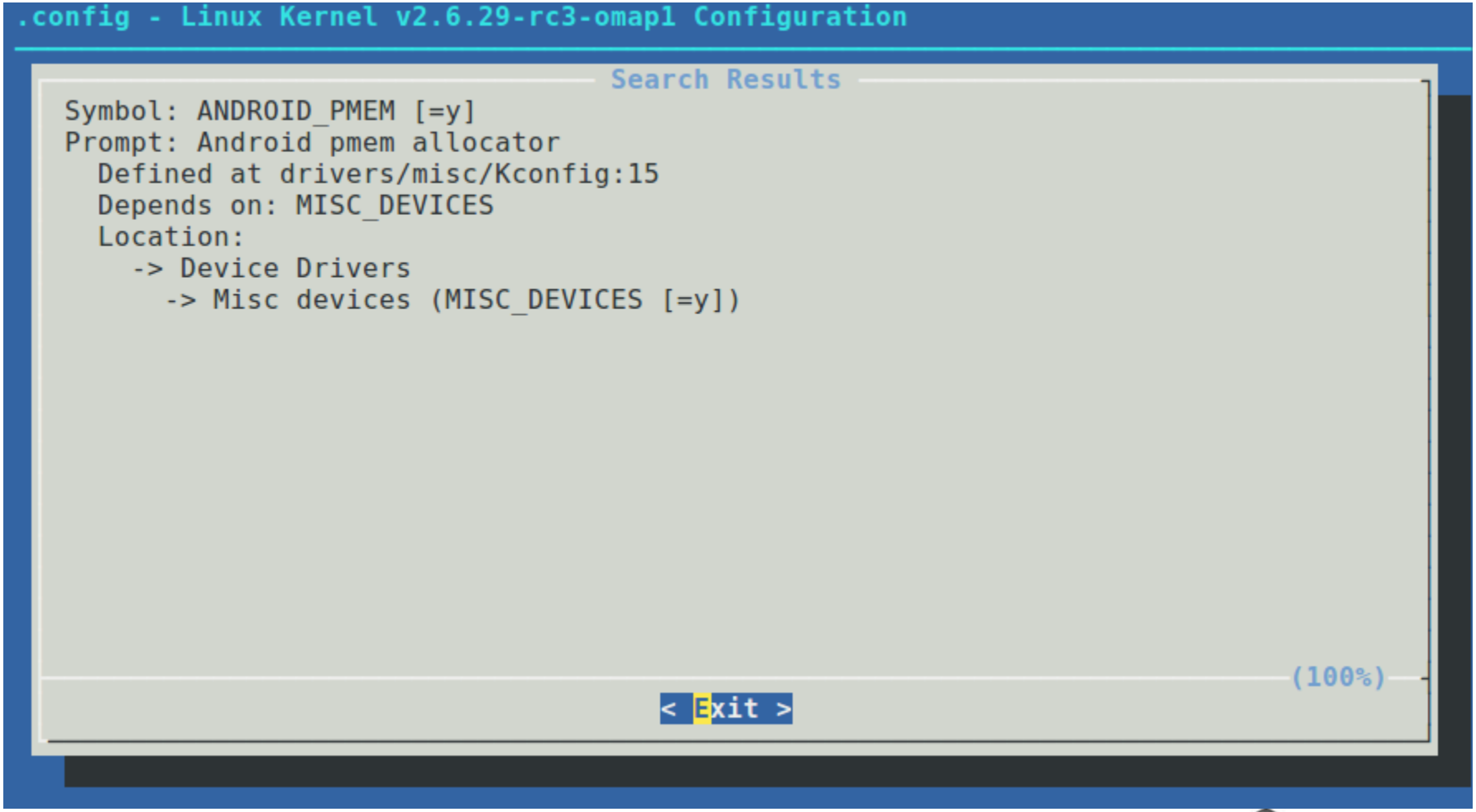
Step2.Activate the android drivers

- Keywords : android, **pmem**, ashmem



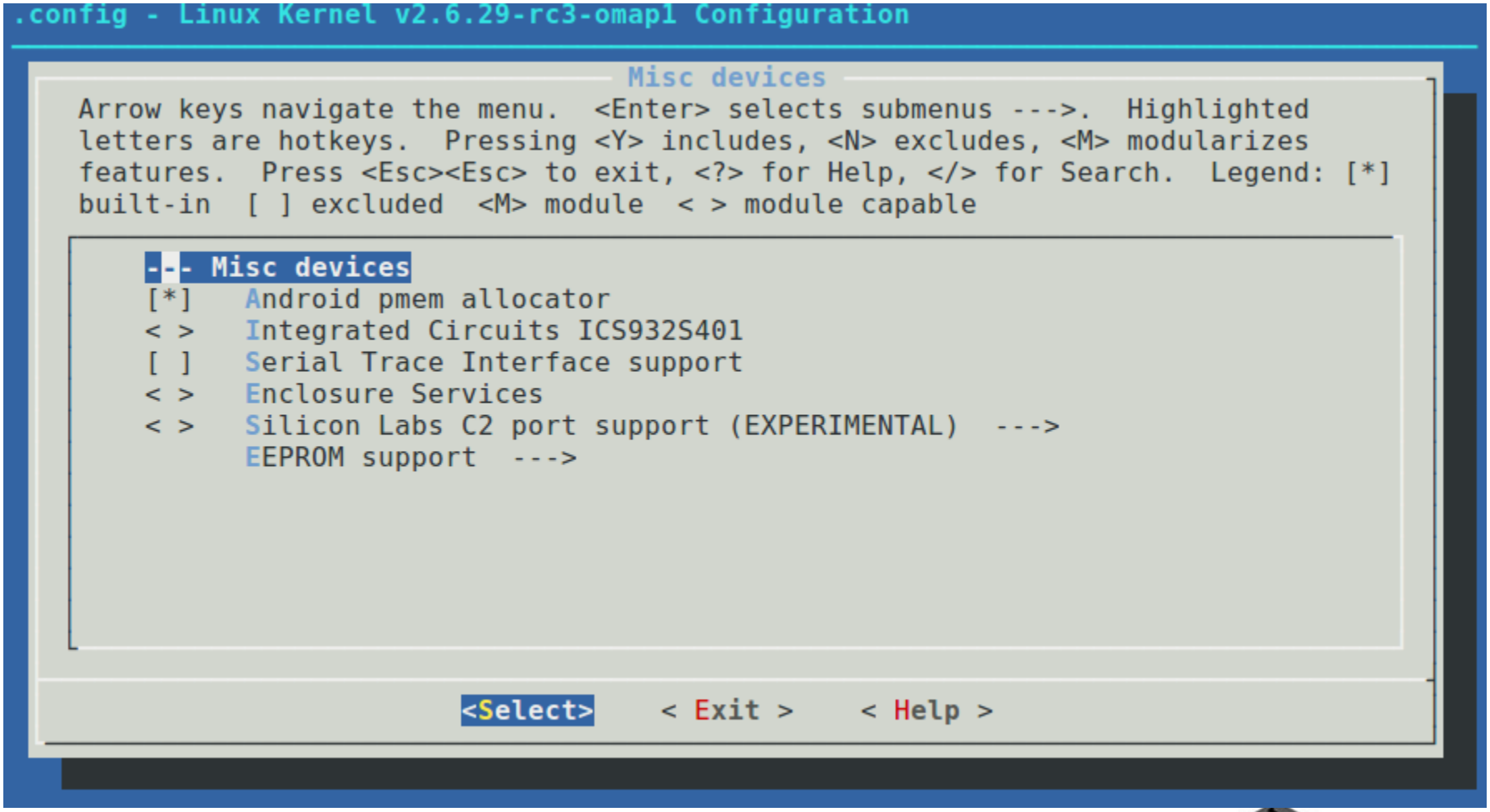
Step2.Activate the android drivers

- Searching result



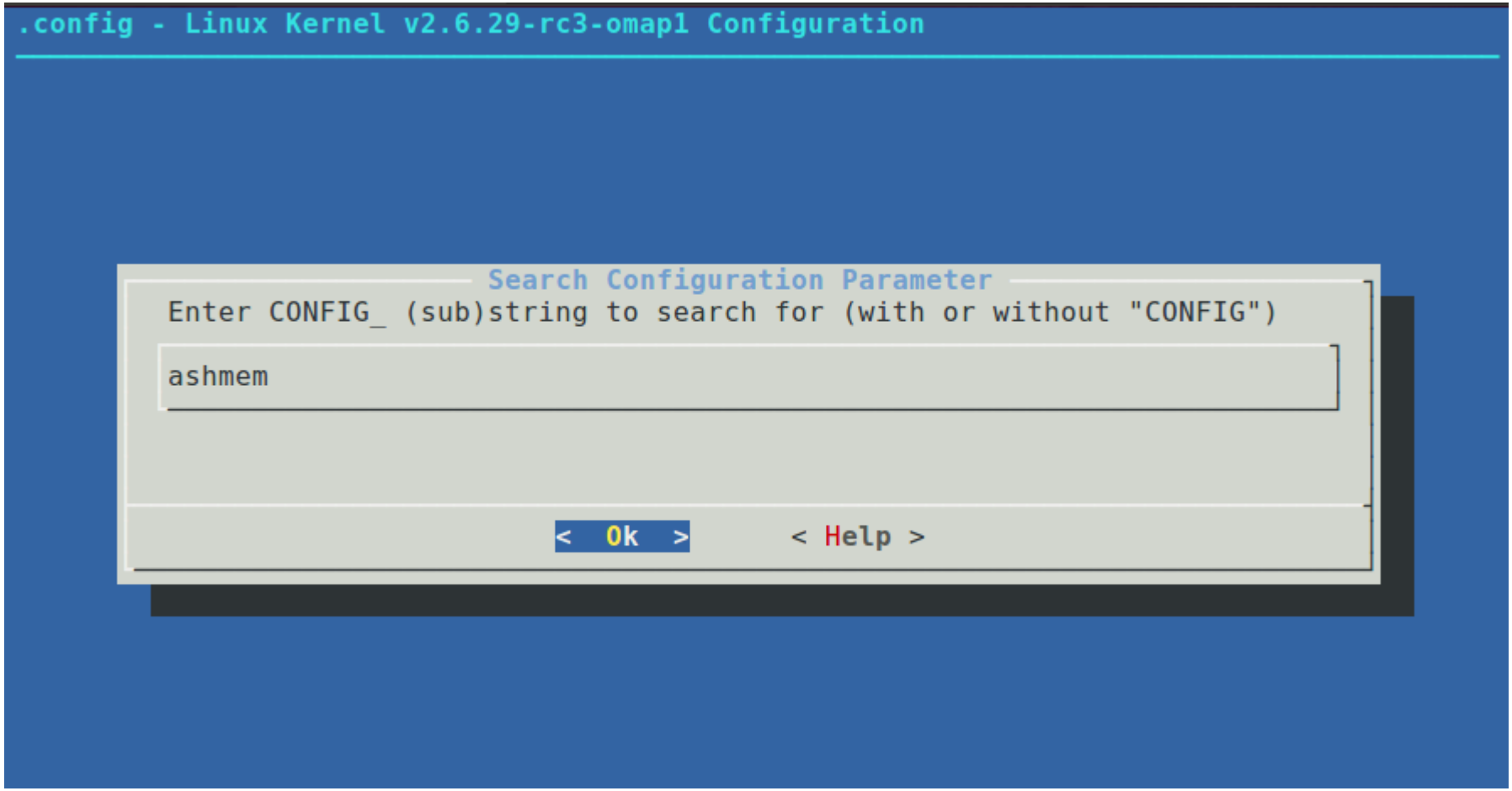
Step2.Activate the android drivers

- Build-in the android modules



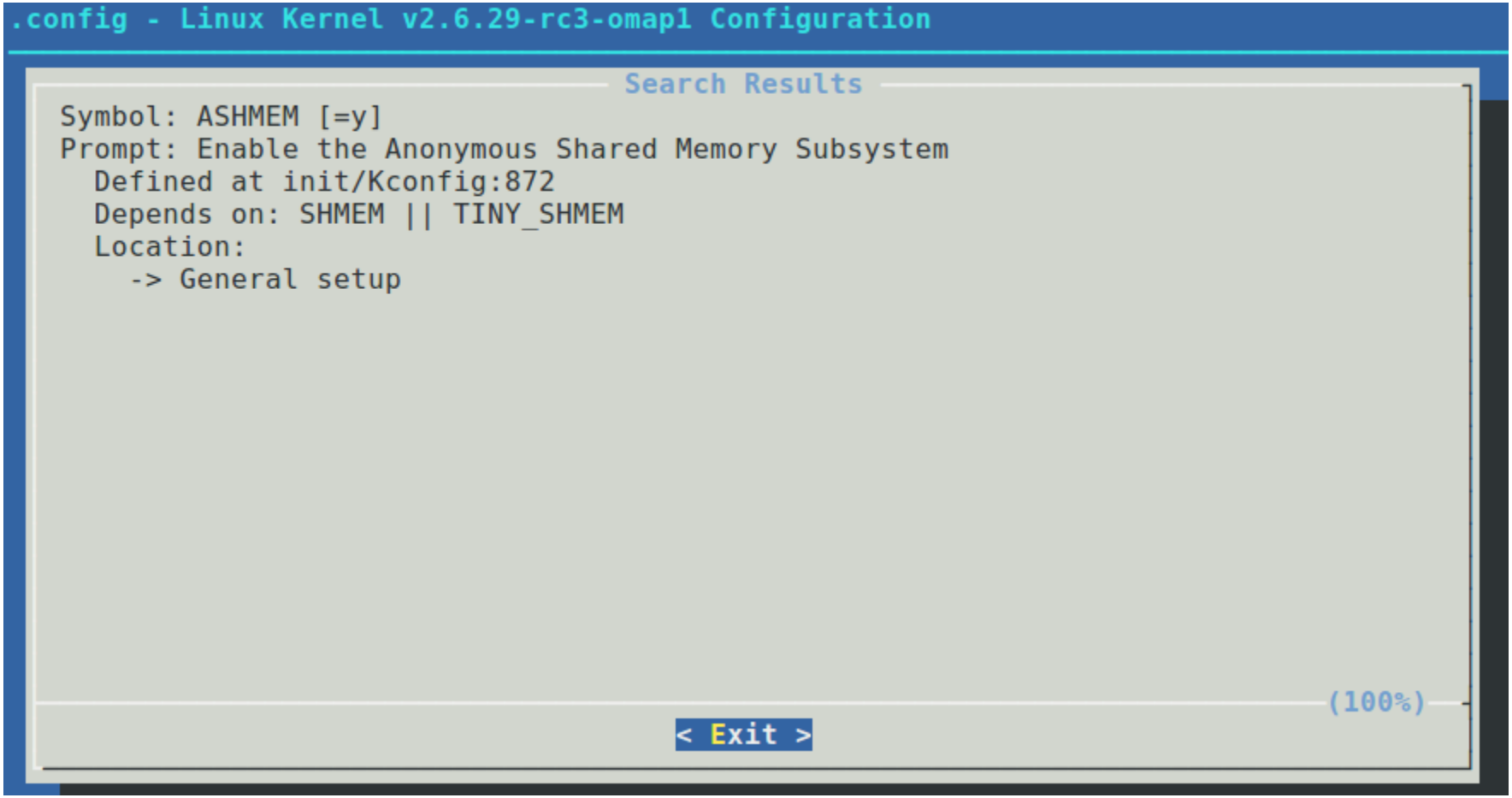
Step2.Activate the android drivers

- Keywords : android, pmem, **ashmem**



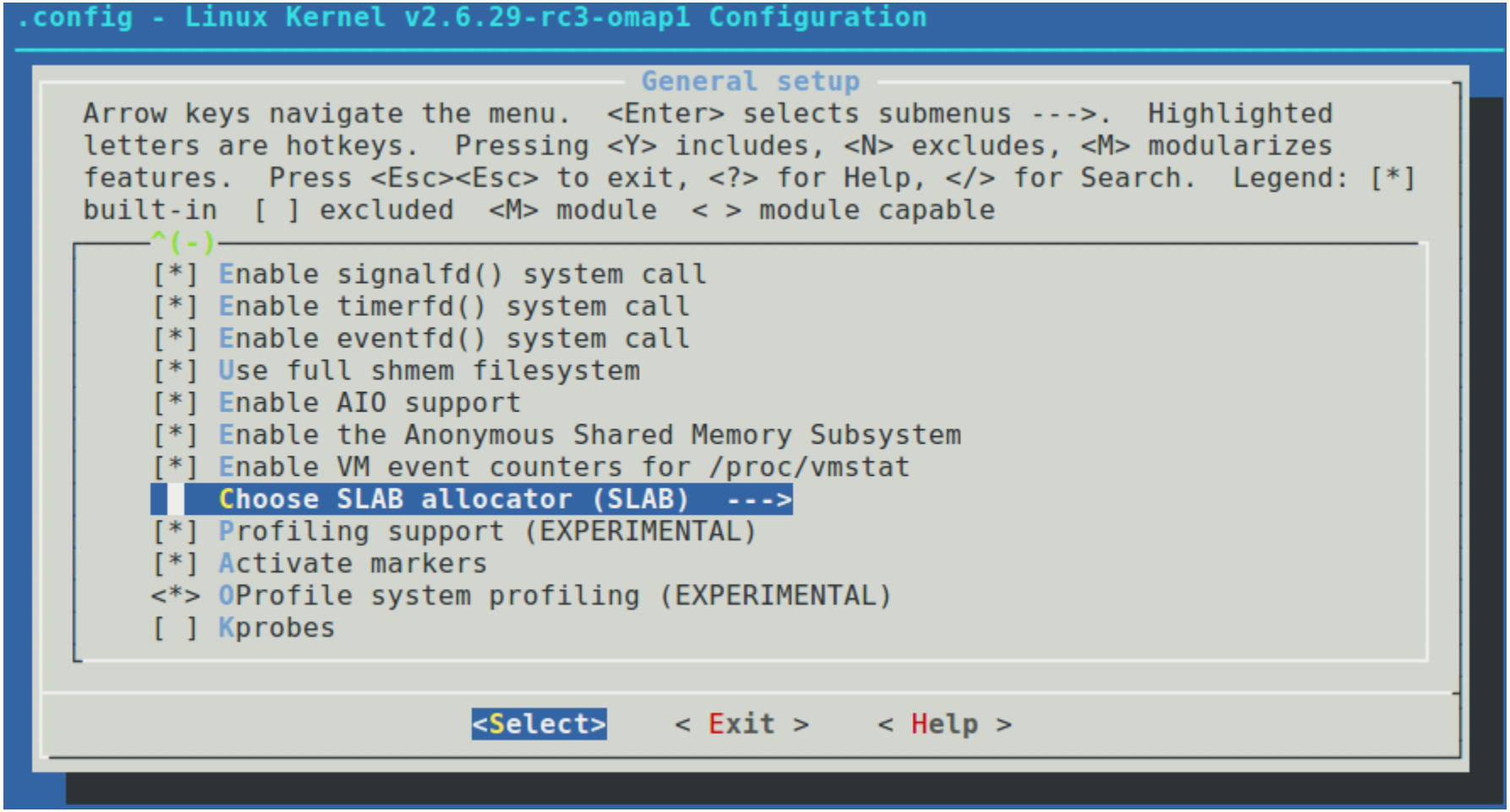
Step2.Activate the android drivers

- Searching result



Step2.Activate the android drivers

- Build-in the android modules



Step3.Build kernel

- After select all the needed android drivers , the ".config" under kernel source folder will be updated ,then
  - $make ARCH=arm CROSS_COMPILE=arm-none-linux-gnueabi-
  - $make ARCH=arm CROSS_COMPILE=arm-none-linux-gnueabi- uImage
- If compile error , remove include/asm
  - $sudo rm include/asm
- After compile , generate the "uImage" under arch/arm/boot
  - Use this uImage to boot (tftp or RS232)
  - $sudo cp arch/arm/boot/uImage /var/lib/tftpboot/

Step4.Mount Android Filesystem

- On your host PC
  - Add the path of android filesystem to /etc/exports as network filesystem
- On devkit8000
  - revise bootargs for android network filesystem path and add "init=/init noinitrd rw" to bootargs
  - Boot!

