Jelly Bean Device Porting Walkthrough

Benjamin Zores

ABS 2013 – 18th February 2013 – San Francisco, CA

THE LINUX FOUNDATION

Alcatel·Lucent

COPYRIGHT © 2012 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough

About Me

ALCATEL LUCENT

ANDROID PLATFORM ARCHITECT

- Expert and Evangelist on Open Source Software.
- 9y experience on various multimedia/network embedded devices design.
- From low-level BSP integration to global applicative software architecture.

OPEN SOURCE

PROJECT FOUNDER, LEADER AND/OR CONTRIBUTOR FOR:

• OpenBricks	Embedded Linux cross-build framework.
• GeeXboX	Embedded multimedia HTPC distribution.
• uShare	UPnP A/V and DLNA Media Server.
• MPlayer	Linux media player application.

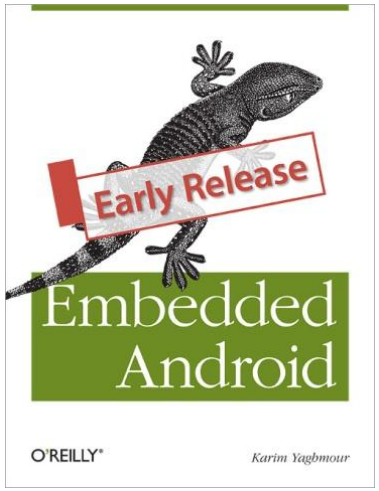
LINUX FOUNDATION CONFERENCES

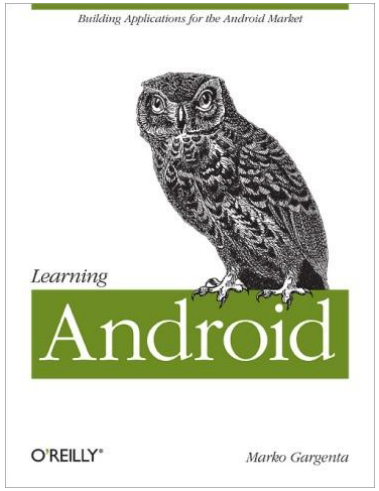
FORMER LINUX FOUNDATION'S EVENTS SPEAKER

• ELC 2010	GeeXboX Enna: Embedded Media Center
• ELC-E 2010	State of Multimedia in 2010 Embedded Linux Devices
• ELC-E 2011	Linux Optimization Techniques: How Not to Be Slow ?
• ABS 2012	Android Device Porting Walkthrough
• ELC-E 2012	Dive Into Android Networking: Adding Ethernet Connectivity

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Bibliographical References





My Android bibles,
from my Android mentors:

Karim Yaghmour
Marko Gargenta

Followed by my own publications:
« Discovering Android »

Series of articles published in
GNU/Linux Magazine France

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Agenda

1. What is Android ?
2. Device Porting How-To
3. Boot-Loader
4. The Linux Kernel
5. Android Build System
6. Android Ecosystem
7. Android Init
8. Storage Subsystem
9. Graphics Subsystem
10. Input Layer
11. Audio Subsystem
12. Multimedia Subsystem
13. Connectivity Subsystem
14. Miscellaneous Devices

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

What is Android ?

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough

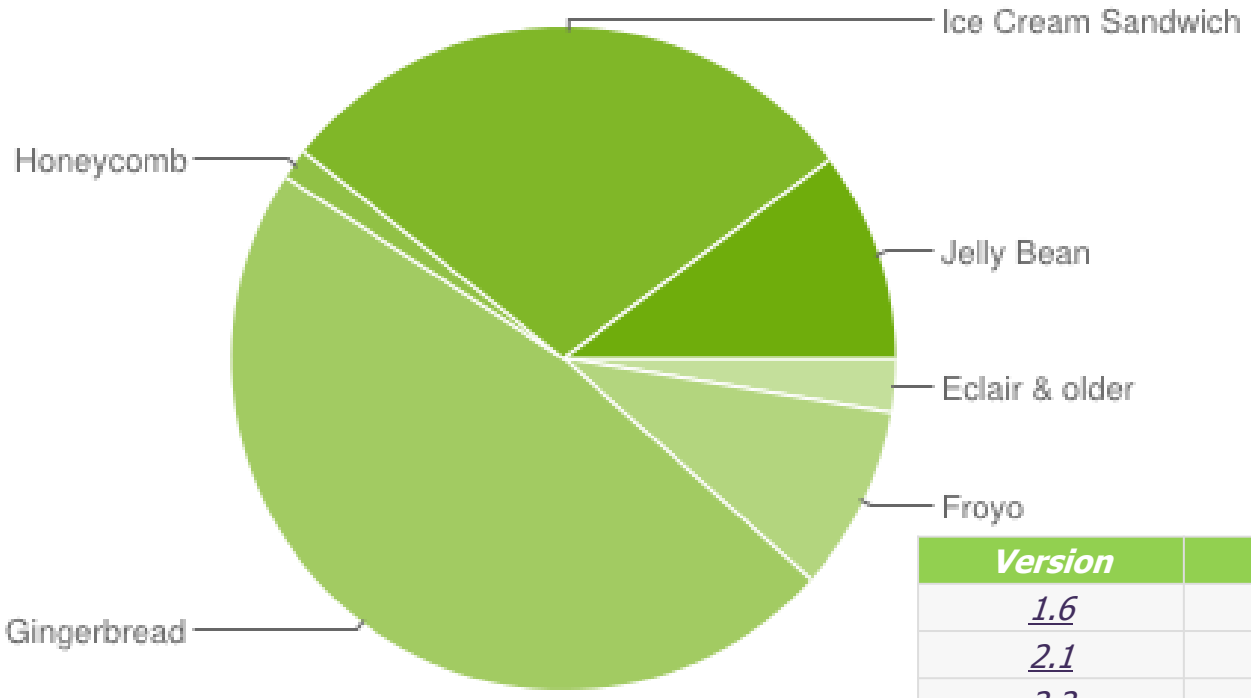
Releases History

Nickname	Version	SDK API Level	NDK API Level	Release Date	Linux Version
N.A.	1.0	1	N.A.	Sep. 08	2.6.25
Petit Four	1.1	2	N.A.	Feb. 09	2.6.25
Cupcake	1.5	3	1	Apr. 09	2.6.27
Donut	1.6	4	2	Sep. 09	2.6.29
Eclair	2.0	5	2	Oct. 09	2.6.29
	2.0.1	6	2	Dec. 09	2.6.29
	2.1	7	3	Jan. 10	2.6.29
Froyo	2.2	8	4	May. 10	2.6.32
Gingerbread	2.3 – 2.3.2	9	5	Nov. 10	2.6.35
	2.3.3 – 2.3.7	10	5	Feb. 11	2.6.35
Honeycomb	3.0	11	6	Feb. 11	2.6.36
	3.1	12	6	May. 11	2.6.36
	3.2	13	6	Jun. 11	2.6.36
Ice Cream Sandwich	4.0 – 4.0.2	14	7	Oct. 11	3.0.1
	4.0.3 – 4.0.4	15	7	Dec. 11	3.0.1
Jelly Bean	4.1.1 – 4.1.2	16	8	Jun. 12	3.0.31
	4.2	17	8	Nov. 12	3.0.31

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough

Fragmentation (Jan. 2013)



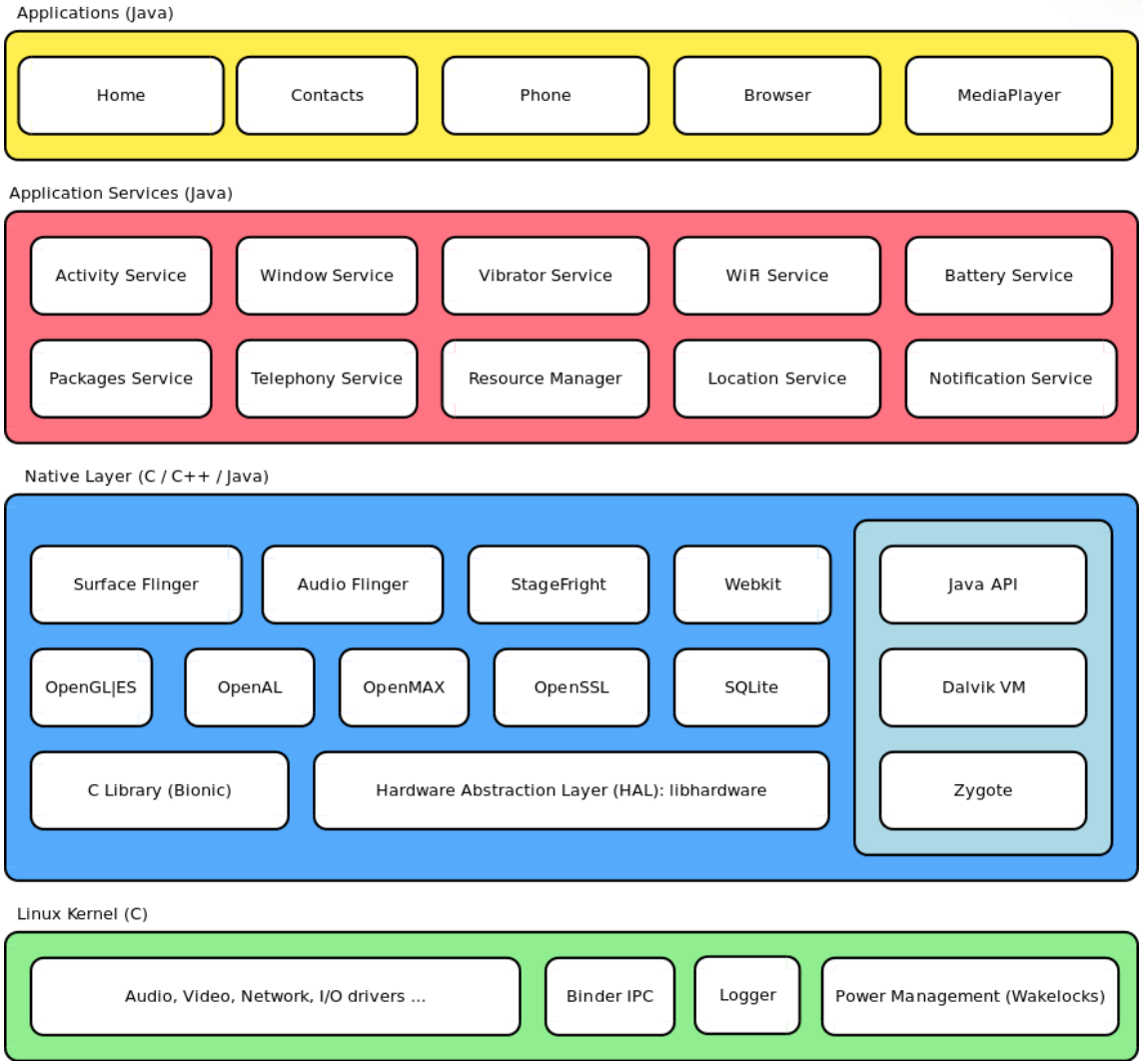
Source © Google

Version	Codename	Distribution
1.6	Donut	0.2%
2.1	Eclair	2.4%
2.2	Froyo	9.0%
2.3 - 2.3.2	Gingerbread	0.2%
2.3.3 - 2.3.7		47.4%
3.1	Honeycomb	0.4%
3.2		1.1%
4.0.3 - 4.0.4	Ice Cream Sandwich	29.1%
4.1	Jelly Bean	9.0%
4.2		1.2%

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Software Architecture

- **Android OS:**
  - Low-Level Linux Kernel & Drivers.
  - System Native Libraries, Services, Daemons and Supervisors.
  - Java-based Applicative Framework.
  - Java Applications.
    - Default ones.
    - User additions.



COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Which Android Sources to Start With ?

- **Google**
  - Up-to-date reference but limited devices support (reference design only).
  - **https://android.googlesource.com/platform/manifest**
- **Linaro**
  - Most integrated with wide hardware support (at least for SoC vendors reference boards).
  - Many packages addition, compiler optimizations and fixes.
  - **git://android.git.linaro.org/platform/manifest.git**
- **Cyanogen Mod**
  - Most features but mostly used for tuning already released commercial products.
  - **https://github.com/cyanogenmod**
- **Vendor BSP**
  - Potentially outdated but best support for a given platform/SoC.

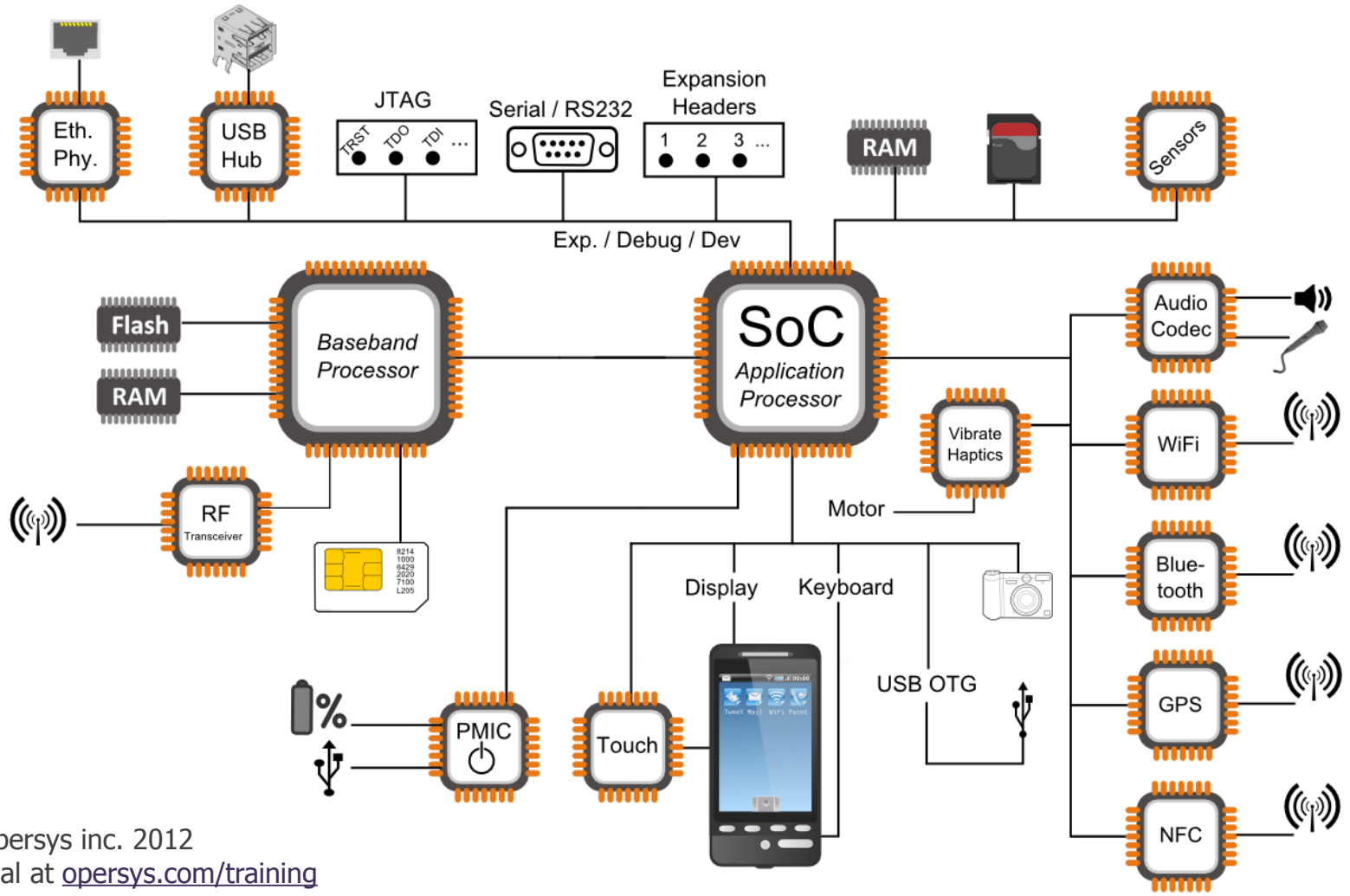
COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Device Porting How-To

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough

Device's HW Internals



(C) Opersys inc. 2012
Original at opersys.com/training

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Device Porting Checklist

- **Recommendation:**
  - Ensure your SoC is already supported by Linux kernel.

- 1. Design your custom board around a reference design.
- 2. Ensure your HW is compatible with **Android CDD**.
- 3. Whenever possible, select HW peripherals that you know are supported.
- 4. Ensure you have detailed specifications for each peripheral !
- 5. Think about SoC pin muxing and avoid multiplexing.
- 6. Ensure you have HW schematics at hand.

- **Let's start writing drivers ;-)**

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Building an Android-Compatible Device

- **Compatibility Goals**
  - Provide a consistent application and HW environment to application developers.
  - Enable a consistent application experience for consumers.
  - Enable device manufacturers to differentiate while being compatible.
  - Minimize costs and overhead associated with compatibility.
- **Compatibility Definition Document (CDD)**
  - List of hardware/software requirements per Android version.
  - Mandatory requirements for device certification and access to market place.
- **Compatibility Test Suite (CTS)**
  - Set of unit designed to be integrated into the daily workflow (such as via a continuous build system) of the engineers building a device.
  - Its intent is to reveal incompatibilities early on, and ensure that the software remains compatible throughout the development process.
- More info on **http://source.android.com/compatibility/overview.html**

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Jelly Bean Device Compatibility Guidelines

- **Goals**
  - Ensure Google device certification.
  - Provide Android **PlayStore** support.
  - Provide Android applications compatibility.
  - Allow access to Google Apps (GMail, Calendar ...).

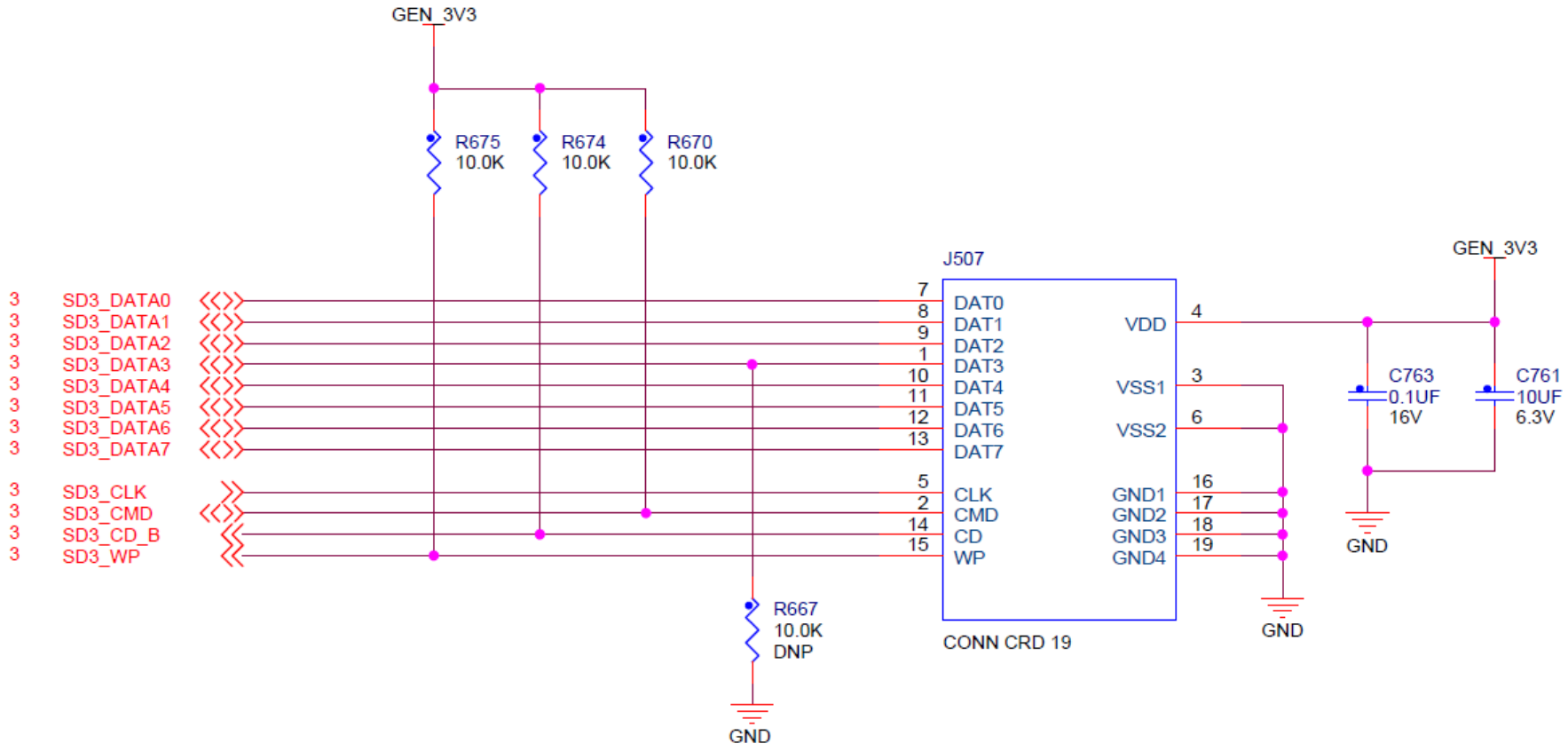
- **Key HW Mandatory Requirements**
  - Memory: Minimum of 340 MB (+ HW dedicated DMA memory).
  - Storage: Minimum of 2 GB.
  - Display: Minimum resolution of 426x320
Aspect ratio must be between 4:3 and 16:9.
  - Graphics: OpenGL ES 1.0 / 2.0 is mandatory.
Hardware 2D / 3D engine is more than recommended.

  - DO NOT modify Google APIs and framework.

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough

HW Schematics (Example)



COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
HW Pad/Pin Mux Table (Example)

Table 4-1. Pin Muxing (continued)

Pad Name	Mode	Instance	Port	Pad Settings
SD2_DAT2	ALT0	usdhc2	DAT2	Hyst. Enable - CFG(Enabled)
	ALT1	ecspi5	SS1	Drive Strength - CFG(R0DIV6)
	ALT2	eim	EIM_CS[3]	Pull Up / Down Config. - CFG(100KOhm PU)
	ALT3	audmux	AUD4_TXD	
	ALT4	kpp	ROW[6]	Pull / Keep Enable - CFG(Enabled)
	ALT5	gpio1	GPIO[13]	Open Drain Enable - CFG(Disabled)
				Speed - CFG(100MHz)
				Pull / Keep Select - CFG(Pull)
				Slew Rate - CFG(SLOW)
SD2_DAT0	ALT0	usdhc2	DAT0	Hyst. Enable - CFG(Enabled)
	ALT1	ecspi5	MISO	Drive Strength - CFG(R0DIV6)
	ALT3	audmux	AUD4_RXD	Pull Up / Down Config. - CFG(100KOhm PU)
	ALT4	kpp	ROW[7]	
	ALT5	gpio1	GPIO[15]	Pull / Keep Enable - CFG(Enabled)
				Open Drain Enable - CFG(Disabled)
				Speed - CFG(100MHz)
				Pull / Keep Select - CFG(Pull)
				Slew Rate - CFG(SLOW)

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Peripherals Integration

Type	Difficulty	Location	Comment
CPU Core	High	Bootloader, Kernel	*Hopefully provided by SoC manufacturer*
PMIC	High	Bootloader, Kernel	*Device Specific: to be configured …*
NAND/eMMC	Easy	Bootloader, Kernel	*Partitionning is up to you …*
LCD	Easy	Kernel	*Depends on selected peripheral …*
2D/3D GPU	Medium	Kernel, Android HAL	*Hopefully provided by SoC manufacturer*
Touchscreen	Easy	Kernel	*Depends on selected peripheral …*
Audio Codec	Medium	Kernel, Android HAL	*Depends on selected peripheral …*
Video Codec	Medium	Kernel, Android HAL, Framework	*Hopefully provided by SoC manufacturer*
USB	Easy	Kernel	
GSM Radio	Medium	Kernel, Android HAL	*Depends on selected peripheral …*
Wi-Fi	Medium	Kernel, Android HAL	*Depends on selected peripheral …*
Bluetooth	Medium	Kernel, Android HAL	*Depends on selected peripheral …*
Ethernet	Easy	Kernel, Android HAL, Framework	
NFC	Medium	Kernel, Android HAL	*Depends on selected peripheral …*
GPS	Medium	Kernel, Android HAL	*Depends on selected peripheral …*
Sensors	Medium	Kernel, Android HAL	*Depends on selected peripheral …*

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Current Context

- **For this session, we'll base our examples on:**
  - A custom board, designed in-house.
  - Based on Freescale i.MX6Q SoC.
  - Derived from Freescale SabreSD reference design.

- **With the following peripherals:**
  - WVGA parallel DPI LCD.
  - Cypress CTMG110 Touchscreen.
  - TI WiLink WL1273 BT/Wi-Fi combo chip.
  - Wolfson WM8958 Audio Codec.
  - NXP PN544 NFC chip.

- **Based on Google's JB 4.2r1 official source tree.**

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Boot-Loader

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Boot-Loader

- Google provides one in **bootable/bootloader/legacy**.
- But usually replaced by **U-Boot** with **Fastboot** protocol support.
  - See **http://goo.gl/WYyd5** for protocol details.
- Can be either out of AOSP sources or integrated.
- In **BoardConfig.mk**:
  - **TARGET_BOOTLOADER_BOARD_NAME := MY_DEVICE**
  - **TARGET_BOOTLOADER_CONFIG := my_device_android_config**

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
U-Boot SoC Initialization

```
void board_init(void) {
        mxc_iomux_v3_init((void *)IOMUXC_BASE_ADDR);

        /* board id for linux */
        gd->bd->bi_arch_number = MACH_TYPE_MY_DEVICE;

        /* address of boot parameters */
        gd->bd->bi_boot_params = PHYS_SDRAM_1 + 0x100;

        setup_leds();
        setup_uart();

        setup_i2c(CONFIG_SYS_I2C_PORT);
        i2c_bus_recovery();
        setup_pmic_voltages();
}
```

Initializes I/O

Tells kernel which board to boot (see MACH_TYPE_*)

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
U-Boot UART Initialization

```
static void setup_uart(void) {
        unsigned int reg;

        /* UART3 TXD/RXD */
        mxc_iomux_v3_setup_pad(MX6Q_PAD_EIM_D24__UART3_TXD);
        mxc_iomux_v3_setup_pad(MX6Q_PAD_EIM_D25__UART3_RXD);

        /* UART3 Data Enable */
        reg = readl(GPIO3_BASE_ADDR + GPIO_GDIR);
        reg |= (1 << 31);
        writel(reg, GPIO3_BASE_ADDR + GPIO_GDIR);

        reg = readl(GPIO3_BASE_ADDR + GPIO_DR);
        reg |= (1 << 31);
        writel(reg, GPIO3_BASE_ADDR + GPIO_DR);
}
```

PAD Setup for UART RX/TX

Set GPIO Output High

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
U-Boot eMMC Initialization

```
struct fsl_esdhc_cfg emmc_cfg = {
        .esdhc_base             = USDHC4_BASE_ADDR,
        .no_snoop               = 1,
        .clk_enable             = 1,
        .is_usdhc               = 1,
        .port_supports_uhs18v   = 0,
};
```

```
iomux_v3_cfg_t usdhc4_pads[] = {
        MX6Q_PAD_SD4_CLK__USDHC4_CLK,
        MX6Q_PAD_SD4_CMD__USDHC4_CMD,
        MX6Q_PAD_SD4_DAT0__USDHC4_DAT0,
        MX6Q_PAD_SD4_DAT1__USDHC4_DAT1,
        MX6Q_PAD_SD4_DAT2__USDHC4_DAT2,
        MX6Q_PAD_SD4_DAT3__USDHC4_DAT3,
        MX6Q_PAD_SD4_DAT4__USDHC4_DAT4,
        MX6Q_PAD_SD4_DAT5__USDHC4_DAT5,
        MX6Q_PAD_SD4_DAT6__USDHC4_DAT6,
        MX6Q_PAD_SD4_DAT7__USDHC4_DAT7,
};
```

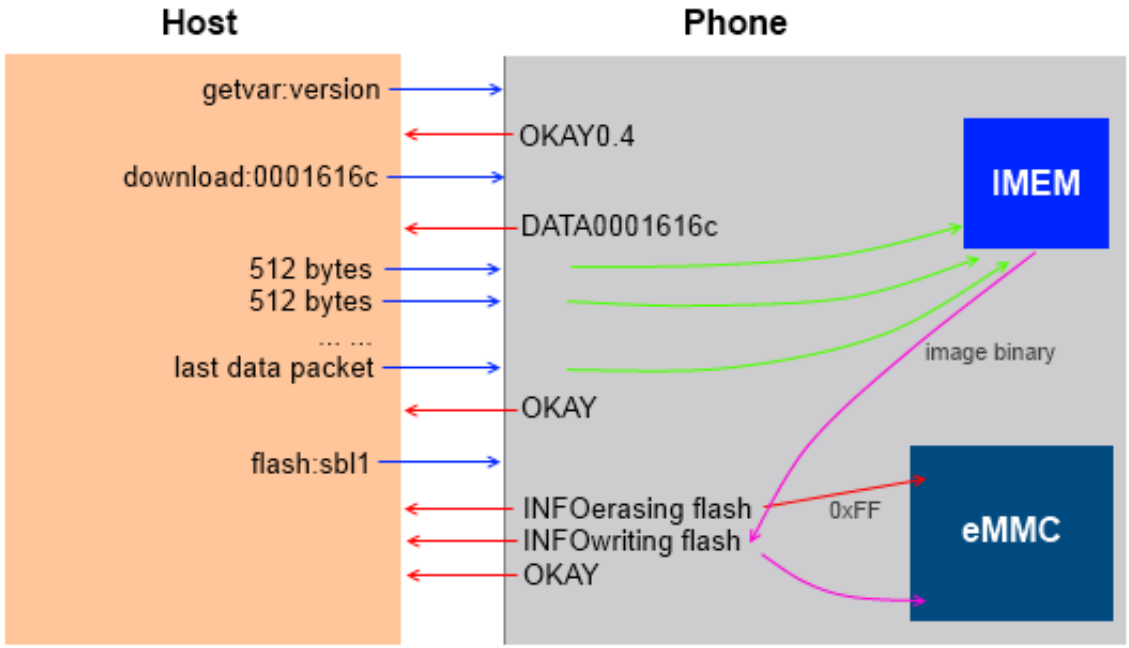
eMMC uses 8-bit data

```
void board_mmc_init(bd_t *bis) {
        mxc_iomux_v3_setup_multiple_pads(usdhc4_pads,
                sizeof(usdhc4_pads) / sizeof(usdhc4_pads[0]));
        fsl_esdhc_initialize(bis, &usdhc_cfg);
}
```


COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Fastboot and Partitions

- Allows remote flashing boot / system / recovery / data images to flash.
  - Handled by boot-loader when in USB mode.
  - Partitions location are hard-coded once for all in boot-loader.
  - e.g. (on host): **fastboot flash boot boot.img**



COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
U-Boot Flash Partitions for Fastboot

```
static void fastboot_init_mmc_sata_ptable(void) {
        int mmc_no = get_mmc_no(fastboot_env);
        dev_desc = get_dev("mmc", mmc_no);

        /* MBR */
        strcpy(ptable[PTN_MBR_INDEX].name, "mbr");
        ptable[PTN_MBR_INDEX].start = ANDROID_MBR_OFFSET / dev_desc->blksz;
        ptable[PTN_MBR_INDEX].length = ANDROID_MBR_SIZE / dev_desc->blksz;
        ptable[PTN_MBR_INDEX].partition_id = user_partition;

        /* Bootloader */
        strcpy(ptable[PTN_BOOTLOADER_INDEX].name, "bootloader");
        ptable[PTN_BOOTLOADER_INDEX].start = ANDROID_BOOTLOADER_OFFSET / dev_desc->blksz;
        ptable[PTN_BOOTLOADER_INDEX].length = ANDROID_BOOTLOADER_SIZE / dev_desc->blksz;
        ptable[PTN_BOOTLOADER_INDEX].partition_id = boot_partition;

        setup_ptable_mmc_partition(PTN_KERNEL_INDEX, CONFIG_ANDROID_BOOT_PARTITION_MMC,
                                   user_partition, "boot", dev_desc, ptable);
        setup_ptable_mmc_partition(PTN_RECOVERY_INDEX, CONFIG_ANDROID_RECOVERY_PARTITION_MMC,
                                   user_partition, "recovery", dev_desc, ptable);
        setup_ptable_mmc_partition(PTN_SYSTEM_INDEX, CONFIG_ANDROID_SYSTEM_PARTITION_MMC,
                                   user_partition, "system", dev_desc, ptable);

        for (i = 0; i <= PTN_RECOVERY_INDEX; i++) fastboot_flash_add_ptn(&ptable[i]);
}
```

```
#define CONFIG_ANDROID_MAIN_MMC_BUS 3
#define CONFIG_ANDROID_BOOT_PARTITION_MMC 1
#define CONFIG_ANDROID_SYSTEM_PARTITION_MMC 5
#define CONFIG_ANDROID_RECOVERY_PARTITION_MMC 2
#define CONFIG_ANDROID_CACHE_PARTITION_MMC 6
```

Maps partition numbers to partition names

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

The Linux Kernel

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Linux Kernel Androidisms

- **Various Kernel Androidisms:**
  - **Ashmem**, anonymous shared memory allocator.
  - **Binder** IPC and RMI system.
  - **Pmem / ION**, process contiguous memory allocator (vendor specific).
  - **Logger**, system logging facility.
  - **Wakelocks**, power management and suspend.
  - **Low Memory Killer**, OOM tuning for OOM-Killer.
  - **Alarm Timers**.
  - **Paranoid Network Security**.
  - **Timed GPIO**.
  - **RAM Console**.
  - **USB Gadget Driver**, for ADB.

- See **http://elinux.org/Android_Kernel_Features** for more details.
- Most of them have been integrated upstream with kernel 3.3 to 3.5.

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Linux Kernel Drivers Policy

- Usually no drivers built as modules:
  - No **udev**, smaller disk footprint, faster to load up.
  - Kernel is built for a static hardware configuration.

- Except for proprietary drivers (usually) or those requiring firmware (e.g. Wi-Fi).
- Can be added to Android FS separately as for firmwares.

- In **BoardConfig.mk**:

```
TARGET_KERNEL_DEFCONF      := my_device_android_defconfig

BOARD_KERNEL_CMDLINE       := console=ttymxc2,115200 init=/init rw
                              video=mxcfb0:dev=lcd,800x480@60,if=RGB24 fbmem=10M
                              vmalloc=512M androidboot.console=ttymxc2 maxcpus=4
                              consoleblank=0
```

Hint: Develop and debug raw drivers on Linux OS, not Android.

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Linux Pin Mux

```
static iomux_v3_cfg_t my_device_pads[] = {
        /* ANATOP */
        MX6Q_PAD_ENET_RX_ER__ANATOP_USBOTG_ID,  /* USBOTG ID pin */

        /* AUDMUX */
        MX6Q_PAD_CSI0_DAT4__AUDMUX_AUD3_TXC,
        MX6Q_PAD_CSI0_DAT5__AUDMUX_AUD3_TXD,
        MX6Q_PAD_CSI0_DAT6__AUDMUX_AUD3_TXFS,
        MX6Q_PAD_CSI0_DAT7__AUDMUX_AUD3_RXD,
        [...]
}

static void __init mx6_my_device_board_init(void) {
        [...]
        mxc_iomux_v3_setup_multiple_pads(my_device_pads, ARRAY_SIZE(my_device_pads));
        [...]
}
```

Must reflect HW schematics

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Linux UART Initialization

```
static iomux_v3_cfg_t uart_pads[] = {
        /* UART3 */
        MX6Q_PAD_EIM_D25__UART3_RXD,
        MX6Q_PAD_EIM_D24__UART3_TXD,
}

static inline void mx6q_init_uart(void) {
        imx6q_add_imx_uart(2, NULL);
}

static void __init mx6_timer_init(void) {
        struct clk *uart_clk;
        mx6_clocks_init(32768, 24000000, 0, 0);
        uart_clk = clk_get_sys("imx-uart.2", NULL);
        early_console_setup(UART3_BASE_ADDR, uart_clk);
}

mxc_iomux_v3_setup_multiple_pads(uart_pads, ARRAY_SIZE(uart_pads));
mx6q_init_uart();
mx6_timer_init();
```

Mux UART RX/TX PAD

Init UART clock for early console (prior to driver init)

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Linux SD/eMMC Initialization

```
static const struct esdhc_platform_data mx6q_sd_data __initconst = {
        .cd_gpio                = IMX_GPIO_NR(1, 4),
        .wp_gpio                = IMX_GPIO_NR(1, 2),
        .keep_power_at_suspend  = 1,
        .support_8bit           = 0,
        .support_18v            = 0,
        .delay_line             = 0,
        .cd_type                = ESDHC_CD_CONTROLLER,
};

static iomux_v3_cfg_t sd_pads[] = {
        MX6Q_PAD_GPIO_4__USDHC2_CD,
        MX6Q_PAD_SD2_CLK__USDHC2_CLK_50MHZ_48OHM,
        MX6Q_PAD_SD2_CMD__USDHC2_CMD_50MHZ_48OHM,
        MX6Q_PAD_SD2_DAT0__USDHC2_DAT0_50MHZ_48OHM,
        MX6Q_PAD_SD2_DAT1__USDHC2_DAT1_50MHZ_48OHM,
        MX6Q_PAD_SD2_DAT2__USDHC2_DAT2_50MHZ_48OHM,
        MX6Q_PAD_SD2_DAT3__USDHC2_DAT3_50MHZ_48OHM,
        MX6Q_PAD_GPIO_2__USDHC2_WP,
};
mxc_iomux_v3_setup_multiple_pads(sd_pads, ARRAY_SIZE(sd_pads));

  /* MMC-Mapping: mmcblk0 (eMMC), mmcblk1 (SD-Card) */
  imx6q_add_sdhci_usdhc_imx(3, &mx6q_emmc_data);
  imx6q_add_sdhci_usdhc_imx(1, &mx6q_sd_data);
```

```
static const struct esdhc_platform_data mx6q_emmc_data __initconst = {
        .always_present         = 1,
        .keep_power_at_suspend  = 1,
        .support_8bit           = 1,
        .support_18v            = 0,
        .delay_line             = 0,
        .cd_type                = ESDHC_CD_PERMANENT,
};

static iomux_v3_cfg_t emmc_pads[] = {
        MX6Q_PAD_SD4_CLK__USDHC4_CLK_50MHZ_48OHM,
        MX6Q_PAD_SD4_CMD__USDHC4_CMD_50MHZ_48OHM,
        MX6Q_PAD_SD4_DAT0__USDHC4_DAT0_50MHZ_48OHM,
        MX6Q_PAD_SD4_DAT1__USDHC4_DAT1_50MHZ_48OHM,
        MX6Q_PAD_SD4_DAT2__USDHC4_DAT2_50MHZ_48OHM,
        MX6Q_PAD_SD4_DAT3__USDHC4_DAT3_50MHZ_48OHM,
        MX6Q_PAD_SD4_DAT4__USDHC4_DAT4_50MHZ_48OHM,
        MX6Q_PAD_SD4_DAT5__USDHC4_DAT5_50MHZ_48OHM,
        MX6Q_PAD_SD4_DAT6__USDHC4_DAT6_50MHZ_48OHM,
        MX6Q_PAD_SD4_DAT7__USDHC4_DAT7_50MHZ_48OHM,
}
mxc_iomux_v3_setup_multiple_pads(emmc_pads, ARRAY_SIZE(emmc_pads));
```


COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Linux SD/eMMC Initialization

```
static struct regulator_consumer_supply usdhc_vmmc_consumers[] = {
        REGULATOR_SUPPLY("vmmc", "sdhci-esdhc-imx.1"),
        REGULATOR_SUPPLY("vmmc", "sdhci-esdhc-imx.3"),
};

static struct regulator_init_data usdhc_vmmc_init = {
        .num_consumer_supplies  = ARRAY_SIZE(usdhc_vmmc_consumers),
        .consumer_supplies      = usdhc_vmmc_consumers,
};

static struct fixed_voltage_config usdhc_vmmc_reg_config = {
        .supply_name            = "vmmc",
        .microvolts             = 3300000,
        .gpio                   = -1,
        .init_data              = &vhe3g_vmmc_init,
};

static struct platform_device vhe3g_vmmc_reg_devices = {
        .name   = "reg-fixed-voltage",
        .id     = 3,
        .dev    = {
                .platform_data = &usdhc_vmmc_reg_config,
        },
};
```

Ensure you list all SDIO devices (eMMC and SD)

Take care of voltage (3.3V here)

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Android Build System

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
AOSP Custom Device How-To

- Create your own **device/company/my_device** directory with complete product description.
- Mandatory **vendorsetup.sh**:

```
add_lunch_combo my_device-eng
add_lunch_combo my_device-userdebug
```

- Mandatory **Android.mk**:

```
LOCAL_PATH := $(call my-dir)
include $(call all-makefiles-under,$(LOCAL_PATH))
```

- Mandatory **AndroidProducts.mk**:

```
PRODUCT_MAKEFILES := $(LOCAL_DIR)/my_device_name.mk
```

- Mandatory **my_device_name.mk**:

```
$(call inherit-product, $(SRC_TARGET_DIR)/product/full_base.mk)
$(call inherit-product, device/company/common/common.mk)
$(call inherit-product, device/company/my_device/device.mk)

PRODUCT_BRAND                   := device_brand
PRODUCT_DEVICE                  := device_name
PRODUCT_NAME                    := device_name
```


COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
AOSP Custom Device How-To

- Mandatory **device.mk**:

```
PRODUCT_COPY_FILES          += device/company/my_device/init.rc:root/init.rc
PRODUCT_TAGS                += dalvik.gc.type-precise
$(call inherit-product, frameworks/base/build/tablet-dalvik-heap.mk)
```

- Mandatory **BoardConfig.mk** (product-specific compile-time definitions):
  - See **build/core/product.mk** for a list of nice-to-know build options.
- **# Platform**

```
BOARD_SOC_TYPE              := IMX6Q
BOARD_SOC_CLASS             := IMX6
```


COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
AOSP Custom Device How-To

- **# Target and compiler options**

```
TARGET_CPU_ABI                          := armeabi-v7a
TARGET_CPU_ABI2                         := armeabi
TARGET_CPU_SMP                          := true
```

- **# Enable NEON feature**

```
TARGET_ARCH_VARIANT                     := armv7-a-neon
ARCH_ARM_HAVE_TLS_REGISTER              := true
TARGET_EXTRA_CFLAGS       += $(call cc-option,"-march=armv7-a -mtune=cortex-a9",
                             $(call cc-option,"-march=armv7-a -mtune=cortex-a8"))
```

- **# Filesystem and partitioning**

```
BOARD_SYSTEMIMAGE_PARTITION_SIZE        := 512M
BOARD_USERDATAIMAGE_PARTITION_SIZE      := 512M
BOARD_CACHEIMAGE_PARTITION_SIZE         := 256M
BOARD_CACHEIMAGE_FILE_SYSTEM_TYPE       := ext4
BOARD_FLASH_BLOCK_SIZE                  := 4096
TARGET_USERIMAGES_USE_EXT4              := true
TARGET_USERIMAGES_SPARSE_EXT_DISABLED   := true
```

- **# System**

```
TARGET_NO_RECOVERY                      := true
TARGET_PROVIDES_INIT_RC                 := true
```


COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
AOSP Custom Device How-To

- Configuration Overlay:
  - In **device.mk**:

```
DEVICE_PACKAGE_OVERLAYS                    := device/company/my_device/overlay
```

  - Create a **device/company/my_device/overlay** directory with same depth and files you want to overwrite. e.g:
    - frameworks/base/core/res/res/values/config.xml
    - frameworks/base/core/res/res/values/dimens.xml
    - frameworks/base/core/res/res/xml/storage_list.xml

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Android Ecosystem

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Bionic C Library

- Small footprint non-POSIX BSD-licensed C library:
- No System V IPC (to avoid potential deny of services).
- No support for locales and wide chars (i.e. multi-byte characters).
  - I18N is done at **Dalvik**/Application level
- Custom pthread implementation, based on Linux futexes.
  - Bundled-in (no –lpthread), with no support for cancellation, process-shared mutexes and conditional variables.
- No support for C++ exceptions.
- Custom timezone support and DNS resolver library.
- Kernel Logger driver **liblog** implementation.
- Several functions are just stubs (i.e. runtime weirdness may happen).
- Non standard /etc config files:
  - Dynamic user/groups management (no pam, group, passwd or shadow)
  - No fstab, no SysV init, no /etc/services or /etc/protocol.
- See **ndk/docs/system/libc/OVERVIEW.html** for exhaustive list.

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
System Properties and Settings Databases

- In-memory Bionic **System Properties** management
  - Array of volatile **property=value** fields.
  - Can be get/set through **getprop/setprop** shell commands as well as Java API.
  - No documentation on existing properties.
  - Anyone can add his own custom properties.

- More persistent configuration settings are available in SQLite databases:
  - e.g: **/data/data/com.android.providers.settings/database/settings.db**
  - See **secure** and **system** tables.

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
User-Space Hardware Abstraction Layer (HAL)

- **Some drivers are implemented in user-space through an HAL**
  - Separates Android platform logic from hardware interface.
  - Offers "standard driver" definition for multiple components (e.g. Graphics, Audio, Camera, GPS, Radio …) and makes porting easier.
  - C/C++ vendor-specific libraries.
  - Communicate with Linux drivers through **/proc**, **/sys** and **/dev**.

- **Implementation**
  - Google offers generic **libhardware** and **libhardware_legacy** templates.
  - OEMs implement "drivers" libs for their specific hardware.
  - Code often remains proprietary.
  - Code is loaded at runtime through pre-determined naming strategies.

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Hardware Abstraction Library

- Series of dynamically loaded plugin (.so files):
  - MUST match board-specific plugin name
  - See APIs in **hardware/libhardware/include/hardware**
  - Default/dummy implementation provided in AOSP (**hardware/libhardware/modules**)
  - MUST be fully implemented by device vendor.
  - Can be closed-source.

- For each HAL class module plugins are checked:
  - based on **system properties** values in the following order:
    - ro.hardware
    - ro.product.board
    - ro.board.platform
    - ro.arch
  - in the following directories order:
    - /vendor/lib/hw
    - /system/lib/hw

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
HAL Components

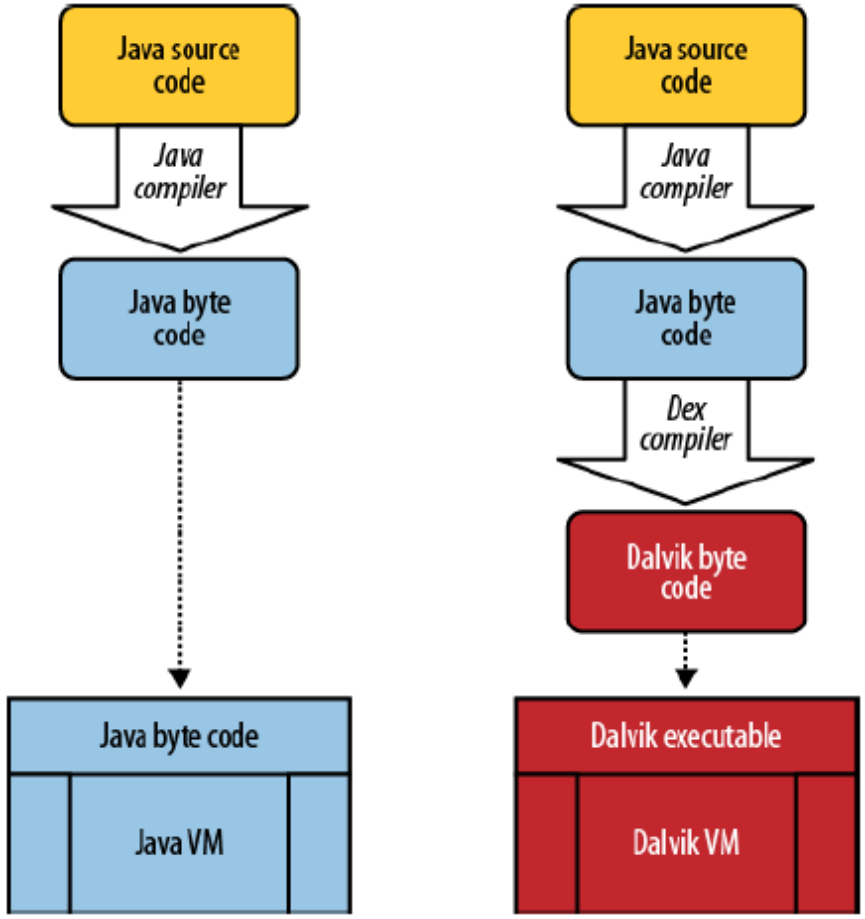
Subsystem	Component	Role
Audio	audio.primary.so	Configuration, Mixing, Routing, Streaming, Echo Cancellation, FX ...
Framebuffer	*Built-in*	Configuration, Composition, Display
GFX Allocator	gralloc.so	GPU Memory Buffer Management
GFX HW Composer	hwcomposer.so	Surface Composition and Transformation
Camera	camera.so	Facing, Orientation, Buffer Management, Pictures, Recording
Wi-Fi	*Built-in*	AP/STA/P2P Configuration and Firmware Support
Bluetooth	bluetooth.default.so	Low-level HW control
GPS	gps.so	Configuration, Location Data Acquisition
NFC	nfc.default.so	Low-level HW control
Ligths	lights.so	Backlight and LEDS control
Sensors	sensors.so	Accelerometer/Pressure/Proximity/Gravity/... Controls
Radio	libril-<company>-<version>.so	Low-level HW control

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Android Dalvik VM Optimizations

- **Dalvik VM:**
  - Register-based versus stack-based VM.
  - Runs **.dex** executable files.
  - More efficient and compact than JVM.
- **Possible Optimizations:**

```
# Dalvik VM tuning
dalvik.vm.heapstartsize=16m
dalvik.vm.heapgrowthlimit=128m
dalvik.vm.heapsize=320m
dalvik.vm.verify-bytecode=false
dalvik.vm.dexopt-flags=v=n,u=n,o=v
dalvik.vm.checkjni=false
dalvik.vm.execution-mode=int:jit
dalvik.gc.type=precise
ro.kernel.android.checkjni=0
ro.kernel.checkjni=0
```



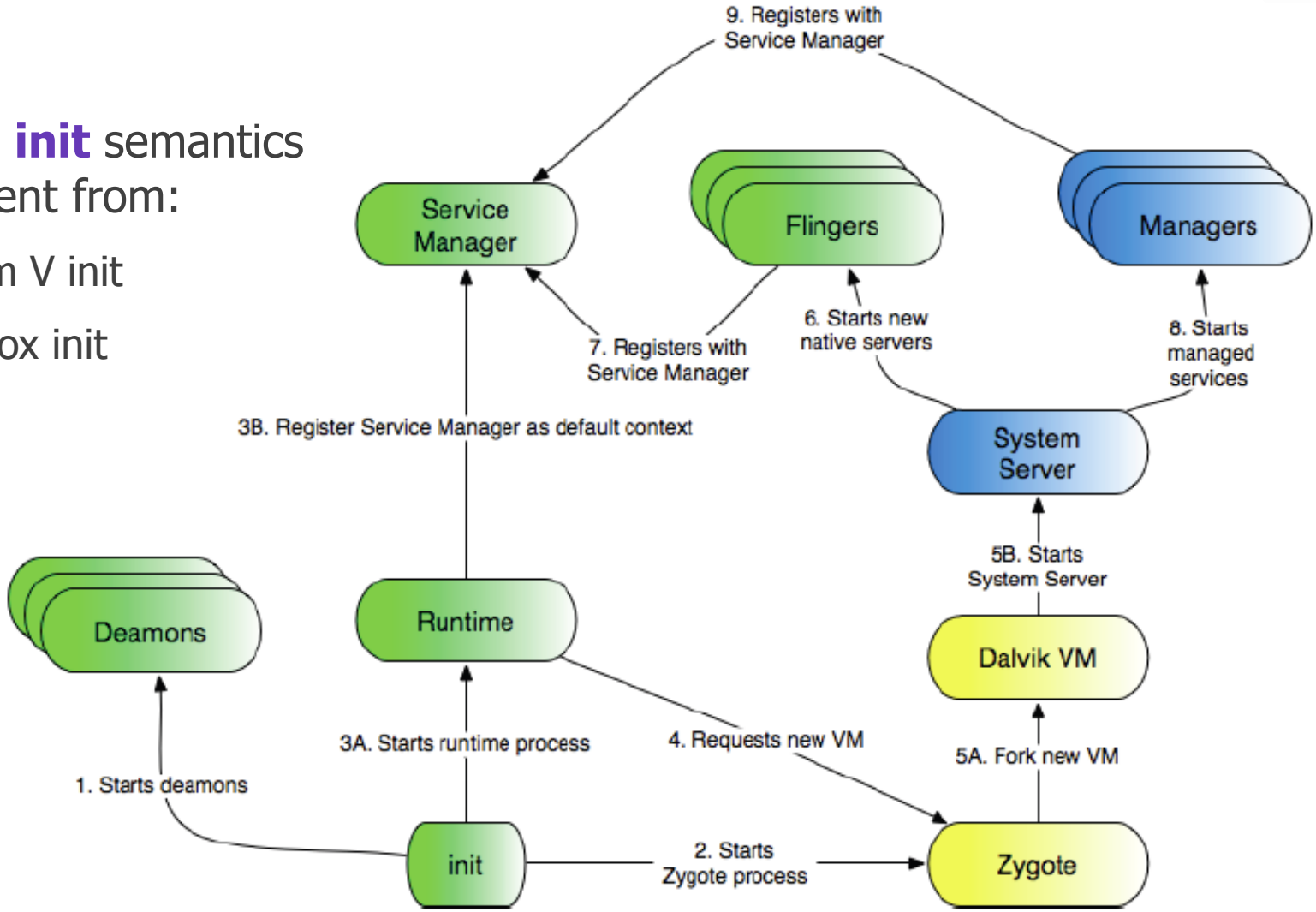
COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Android Init

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Init: Theory of Operations

- Android **init** semantics is different from:
  - System V init
  - BusyBox init



COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Android Init

- Proprietary init language based on rules and conditions.

- Able to spawn services and restart them on failures through signals and sockets.

- **Initialization Steps:**
  - Creates basic filesystem (/dev, /proc, /sys) and mounts it.
  - Parses **/init.rc**
  - Parses **/init.${hw_name}.rc** based on kernel command-line or /proc/cpuinfo
  - Build exec queues
  - Start triggers and associated actions and services.
    - e.g. "**early-init**", "**init**", "**early-fs**", "**fs**", "**post-fs**", "**early-boot**", "**boot**" …

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Android Init Language

- **Actions:**
  - Named sequences of commands queued and executed upon events/triggers.
  - Syntax: **on <trigger> <command> <command> ...**
- **Triggers:**
  - Strings which can be used to match certain kinds of events and used to cause an action to occur.
- **Services:**
  - Programs which init launches and (optionally) restarts when they exit.
  - Syntax: **service <name> <pathname> [<argument>]* <option> <option> ...**
- **Options:**
  - Modifiers to services. They affect how and when init runs the service.
- **Commands:**
  - Android proprietary built-in commands.

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Android Init Options

Option Syntax	Description
critical	This is a device-critical service. If it exits more than four times in four minutes, the device will reboot into recovery mode.
disabled	This service will not automatically start with its class. It must be explicitly started by name.
setenv <name> <value>	Set the environment variable <name> to <value> in the launched process.
socket <name> <type> <perm> [<user> [<group>]]	Create a unix domain socket named /dev/socket/<name> and pass its fd to the launched process.
group <groupname> [<groupname>]*	Change to groupname before exec'ing this service.
oneshot	Do not restart the service when it exits.
class <name>	Specify a class name for the service. All services in a named class may be started or stopped together
onrestart	Execute a Command when service restarts.
ioprio <rt|be|idle> <ioprio 0-7>	Set service priority and scheduling class.
console	Output logs on console.
capability <keycode>	Trigger service through keycode in /dev/keychord

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Android Init Triggers

- **boot**
  - This is the first trigger that will occur when init starts (after **/init.rc** is loaded)
- **<name>=<value>**
  - Triggers of this form occur when the property <name> is set to the specific value <value>.
- **device-added-<path>**
- **device-removed-<path>**
  - Triggers of these forms occur when a device node is added or removed.
- **service-exited-<name>**
  - Triggers of this form occur when the specified service exits.

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Android Init Commands

Command Syntax	Description
exec <path> [<argument>]*	Fork and execute a program (<path>). This will block until the program completes.
export <name> <value>	Set the environment variable <name> equal to <value> in the global environment.
ifup <interface>	Bring the network interface <interface> online.
hostname <name>	Set the host name.
chdir <directory>	Change working directory.
chmod <octal-mode> <path>	Change file access permissions.
chown <owner> <group> <path>	Change file owner and group.
chroot <directory>	Change process root directory.
class_start <serviceclass>	Start all services of the specified class if they are not already running.
class_stop <serviceclass>	Stop all services of the specified class if they are currently running.
class_reset <serviceclass>	Reset a class.
domainname <name>	Set the domain name.
insmod <path>	Install the module at <path>
mkdir <path> [mode] [owner] [group]	Create a directory at <path>, optionally with the given mode, owner, and group.
mount <type> <device> <dir> [<mountoption>]*	Attempt to mount the named device at the directory <dir>

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough

Android Init Commands

Command Syntax	Description
setprop <name> <value>	Set system property <name> to <value>.
setrlimit <resource> <cur> <max>	Set the rlimit for a resource.
start <service>	Start a service running if it is not already running.
stop <service>	Stop a service from running if it is currently running.
restart <service>	Restart a service from running if it is currently running.
symlink <target> <path>	Create a symbolic link at <path> with the value <target>
sysclktz <mins_west_of_gmt>	Set the system clock base (0 if system clock ticks in GMT)
trigger <event>	Trigger an event. Used to queue an action from another action.
write <path> <string> [<string>]*	Open the file at <path> and write one or more strings to it.
rm <path>	Removes a file.
rmdir <path>	Removes a directory.
wait <file>	Wait for file to exist or timeout to be reached.
copy <src> <dest>	Copy from source to dest.
loglevel <level>	Set kernel log level.
load_persist_props	Load properties from files in /data/property

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

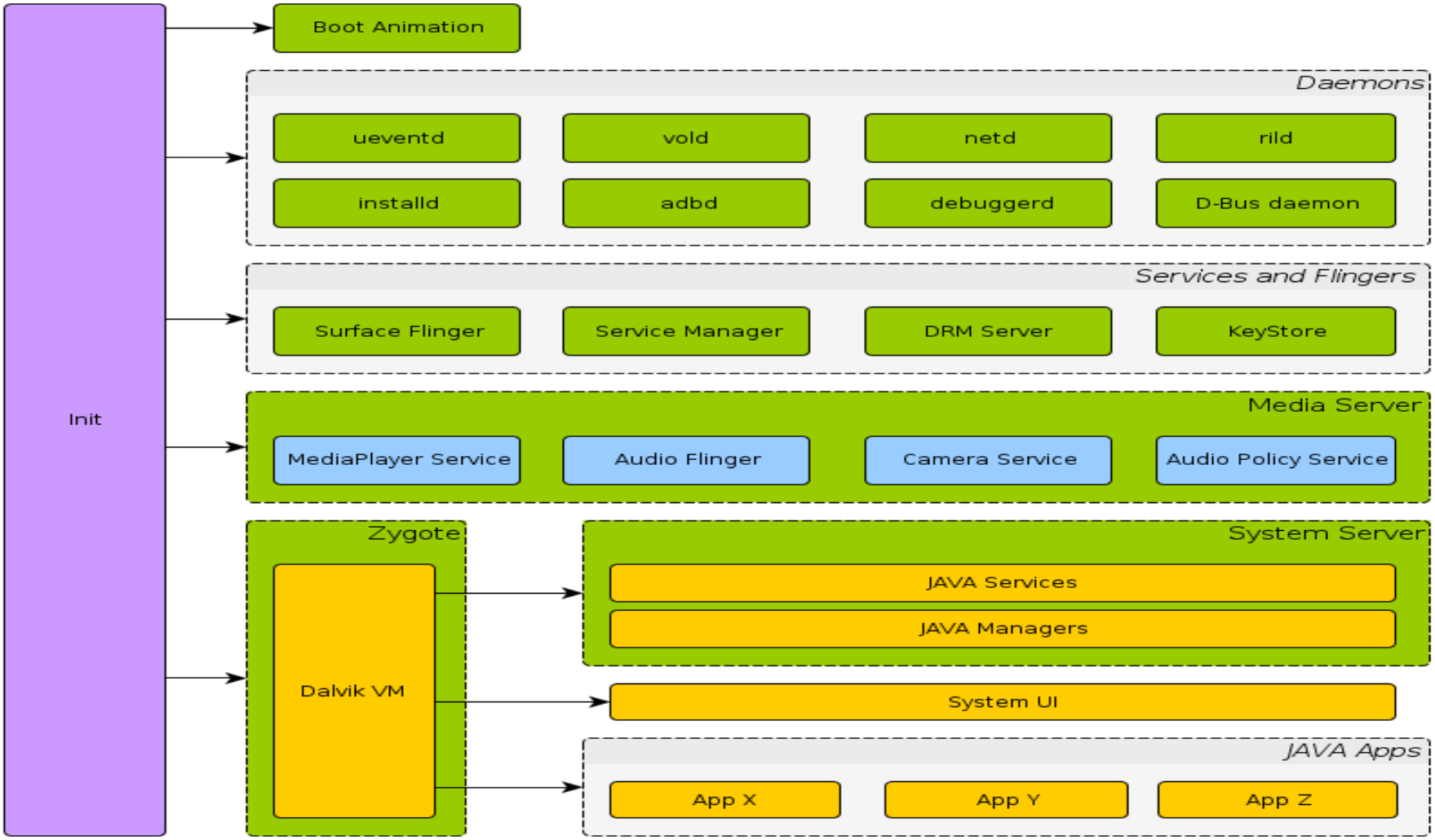
Jelly Bean Device Porting Walkthrough
uEventd

- Somehow replaces **udevd** from desktop Linux.
- Used to set user/group/permissions on /dev nodes.
- **Steps:**
  - Parses **/ueventd.rc**
  - Parses **/ueventd.${hw_name}.rc** based on kernel command-line or /proc/cpuinfo.
  - Set nodes permissions.

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough

Init: Summary



COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Fast & Furious Boot

```
# Give system ownership and permission
# to boost clock for specified timeout
chown system system /sys/devices/system/cpu/cpu0/cpufreq/boost_cpufreq
chmod 0664  /sys/devices/system/cpu/cpu0/cpufreq/boost_cpufreq

# Boost the CPU for 60 sec for boot optimization
write /sys/devices/system/cpu/cpufreq/hotplug/boost_timeout 60000000
write /sys/devices/system/cpu/cpu0/cpufreq/boost_cpufreq 1
```


COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Storage Subsystem

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Booting from eMMC

- Bootloaders loads **boot.img** with kernel + ramdisk
- Ramdisk's **init.rc** mounts rootfs from eMMC and partitions.

```
on fs
# Mount ext4 partitions (mmcblk0 is eMMC, mmcblk1 is SD)
    mount ext4 /dev/block/mmcblk0p5 /system
    mount ext4 /dev/block/mmcblk0p5 /system ro remount
    mount ext4 /dev/block/mmcblk0p7 /data nosuid nodev
    mount ext4 /dev/block/mmcblk0p6 /cache nosuid nodev
    mount ext4 /dev/block/mmcblk0p8 /vendor nosuid nodev
```


COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Volume Daemon and automount

- Overlay storage configuration file:
  - See **frameworks/base/core/res/res/xml/storage_list.xml**:

```
<StorageList xmlns:android="http://schemas.android.com/apk/res/android">
    <storage android:mountPoint="/mnt/sdcard"
            android:storageDescription="@string/storage_internal"
            android:primary="true"
            android:mtpReserve="100" />
    <storage android:mountPoint="/mnt/extsd"
            android:storageDescription="@string/storage_sd_card"
            android:primary="false"
            android:removable="true" />
    <storage android:mountPoint="/mnt/udisk"
            android:storageDescription="@string/storage_usb"
            android:primary="false"
            android:removable="true" />
</StorageList>
```

- Edit list of mount points in **vold**:
  - See **device/company/my_device/vold.fstab**:
    - dev_mount sdcard /mnt/sdcard auto /devices/platform/sdhci-esdhc-imx.1/mmc_host/mmc1

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Tuning your FS for performances

```
# Optimize eMMC read-ahead capabilities
write /sys/block/mmcblk0/queue/rotational 0
write /sys/block/mmcblk0/queue/read_ahead_kb 2048

# Optimize eMMC filesystem
mount ext4 /dev/block/mmcblk0p7 /data nosuid nodev nodiratime noatime noauto_da_alloc discard data=writeback
```

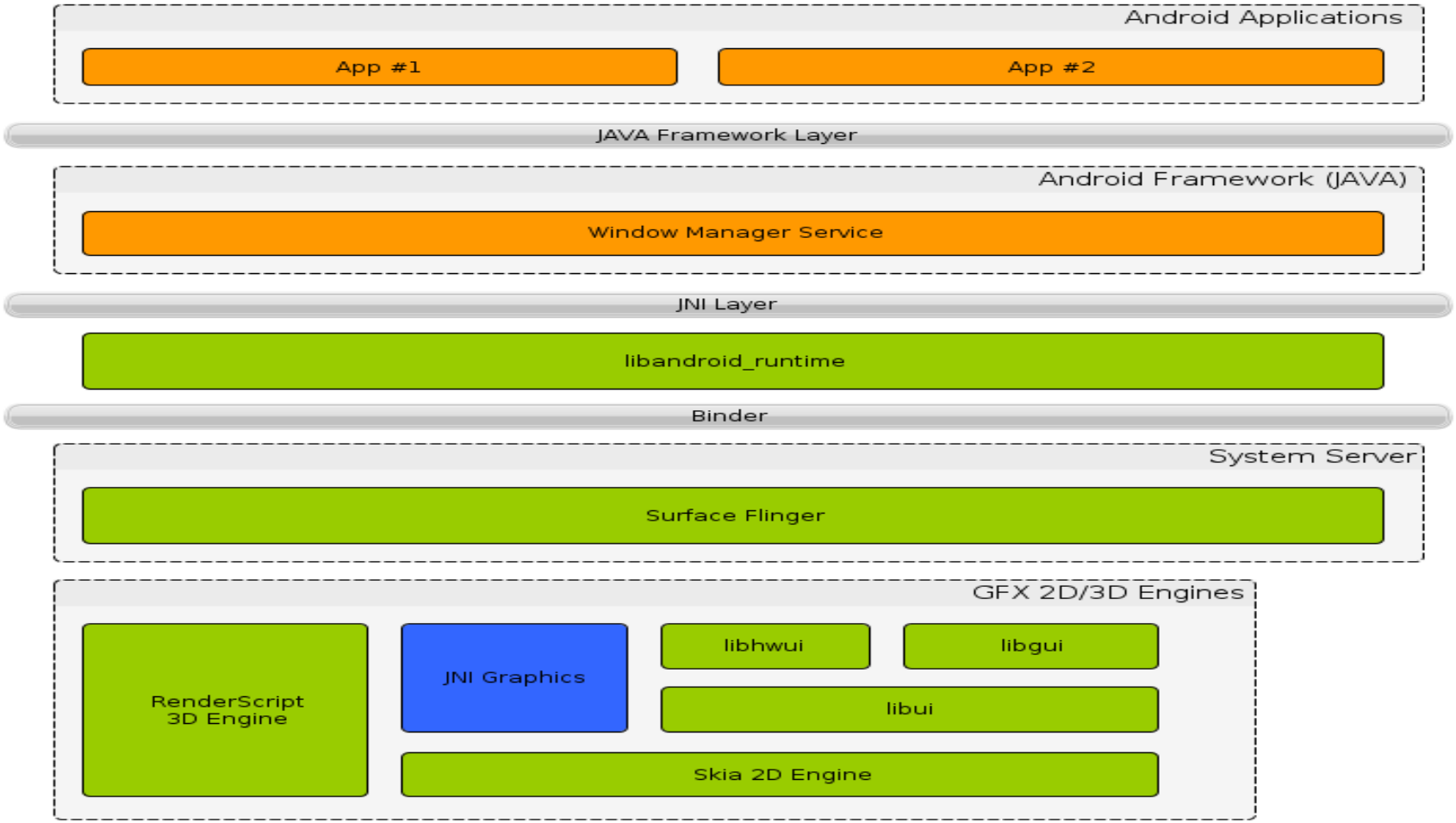
- Having a try at **F2FS** (*Flash-Friendly File System*)
  - Merged in Linux 3.8
  - Initiated by Samsung for SD, eMMC and SSD devices.
  - See LWN for details:
    - **https://lwn.net/Articles/518718/**

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Graphics Subsystem

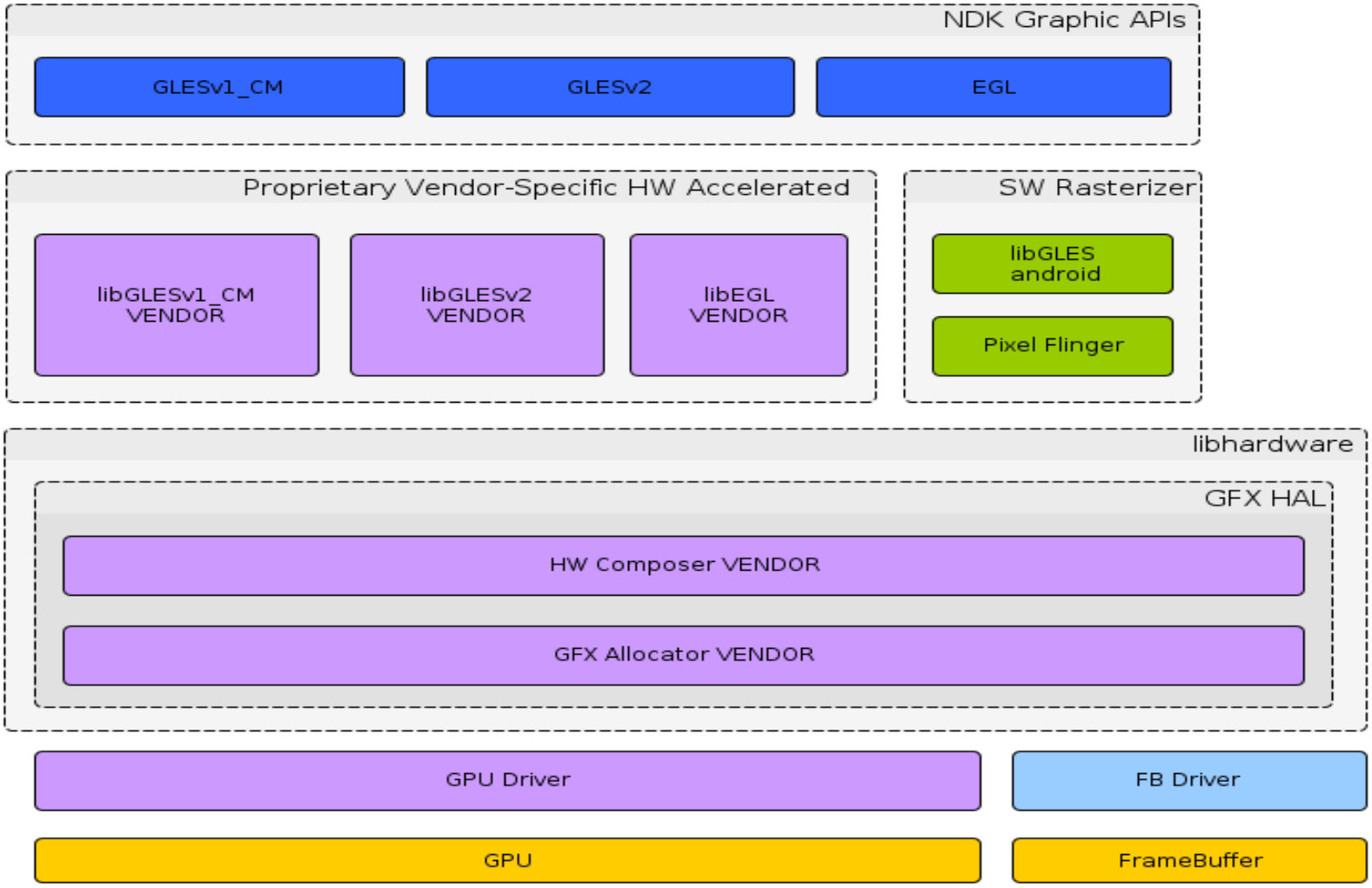
COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Graphics SW Architecture



COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

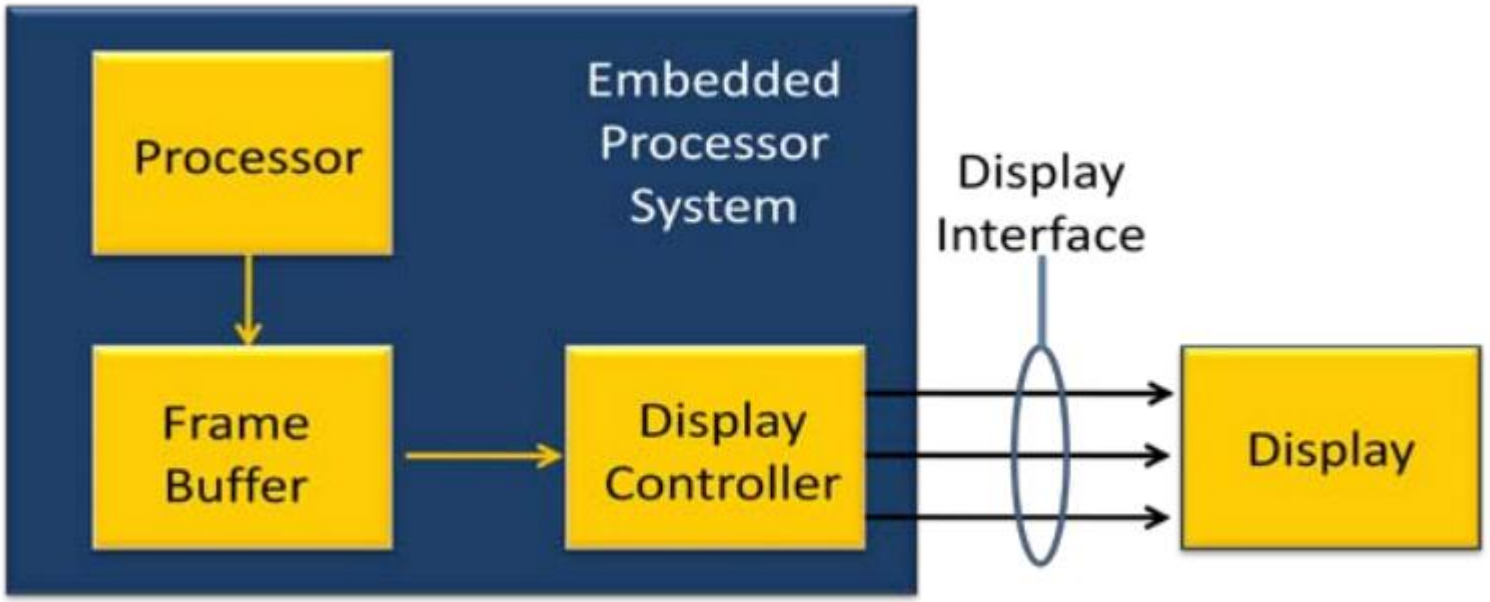
Jelly Bean Device Porting Walkthrough
Graphics SW Architecture



COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Back to the basics: the Framebuffer

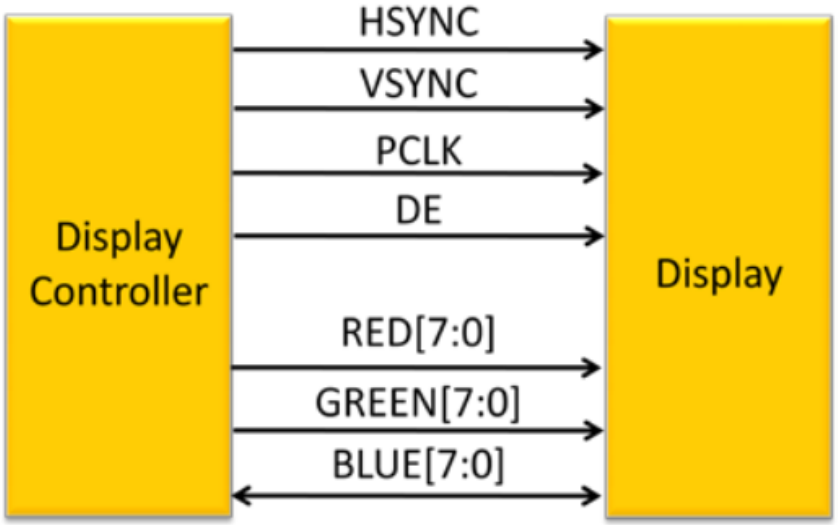
- At low-level, system outputs data to framebuffer.
- Which interacts with display controller
- Until you actually see something displayed on screen ;-)

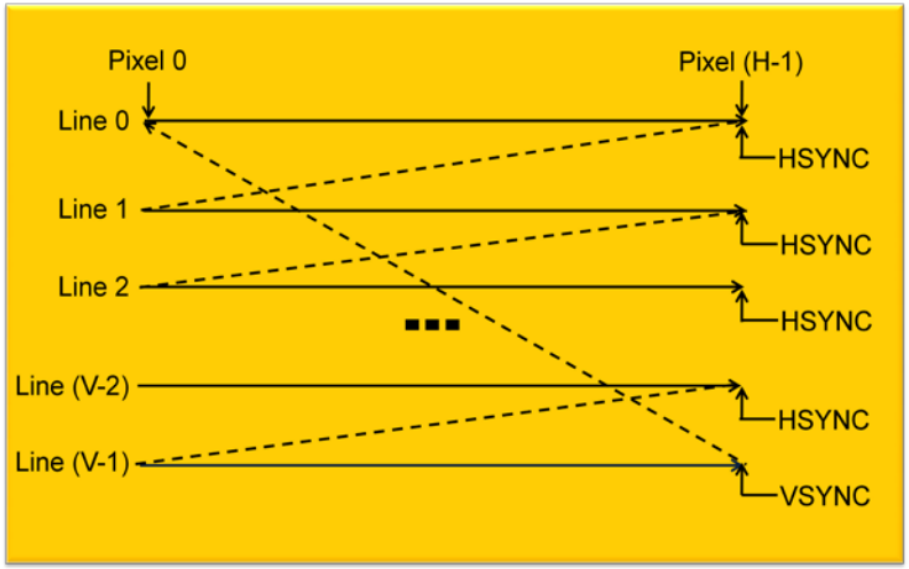


COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
DPI: Display Pixel Interface

- Old but simple parallel 32-bit display interface:
  - 24 bits for data (8/8/8 RGB)
  - 4 bits for horizontal/vertical sync, clock and data enable.

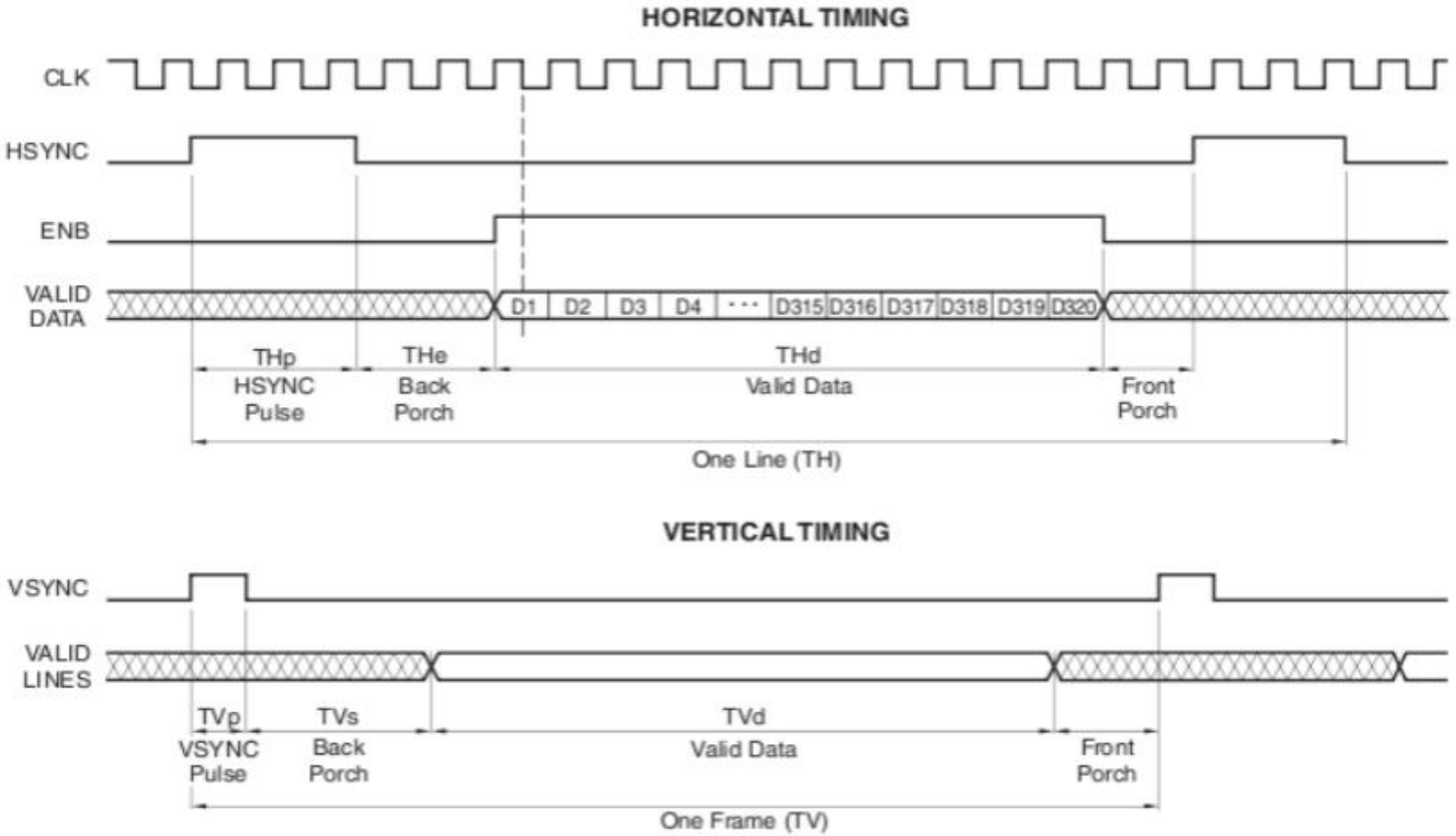




- From now on, you'll need the datasheets ;-)

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Electronic Signal



COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
LCD-DPI Driver Signal Mapping

- Hack **drivers/video/omap2/displays/panel-generic-dpi.c**:

Parameter	Value
HS Period	*TH*
HS Active Time	*THd*
HS Pulse Width	*THp*
HS First Horizontal Data Time	*THs*
VS Period	*TV*
VS Active Time	*TVd*
VS Pulse Width	*TVp*
VS First Vertical Data Time	*TVs*

```
static struct fb_videomode lcd_dpi = {
        .name           = "MYSCREEN",
        .refresh        = REFRESH_RATE,
        .xres           = THd,
        .yres           = TVd,
        .pixclock       = REFRESH_RATE * THd * TVd,
        .left_margin    = THs - THp,
        .right_margin   = TH - THs - THd,
        .upper_margin   = TVs - TVp,
        .lower_margin   = TV - TVs - TVd,
        .hsync_len      = THp,
        .vsync_len      = TVp,
        .sync           = FB_SYNC_CLK_LAT_FALL,
        .vmode          = FB_VMODE_NONINTERLACED,
        .flags          = 0,
};
```


COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough

Bringing up Display Controller

```
#define LCD_POWER_ENABLE          IMX_GPIO_NR(1, 30)
#define LCD_RESET                 IMX_GPIO_NR(3, 8)

static void enable_lcd_dpi(void) {
        gpio_request(LCD_POWER_ENABLE, "lcd_power");
        gpio_direction_output(LCD_POWER_ENABLE, 0);
        gpio_set_value(LCD_RESET, 1);
}

static struct ipuv3_fb_platform_data fb_data = {
        .disp_dev               = "lcd",
        .interface_pix_fmt      = IPU_PIX_FMT_RGB565,
        .mode_str               = "MYSCREEN",
        .default_bpp            = 32,
        .int_clk                = false,
};

static struct fsl_mxc_lcd_platform_data lcdif_data = {
        .ipu_id                 = 0,
        .disp_id                = 0,
        .default_ifmt           = IPU_PIX_FMT_RGB565,
};

imx6q_add_ipuv3fb(0, &fb_data);
imx6q_add_lcdif(&lcdif_data);
enable_lcd_dpi(void);
```

According to HW schematics

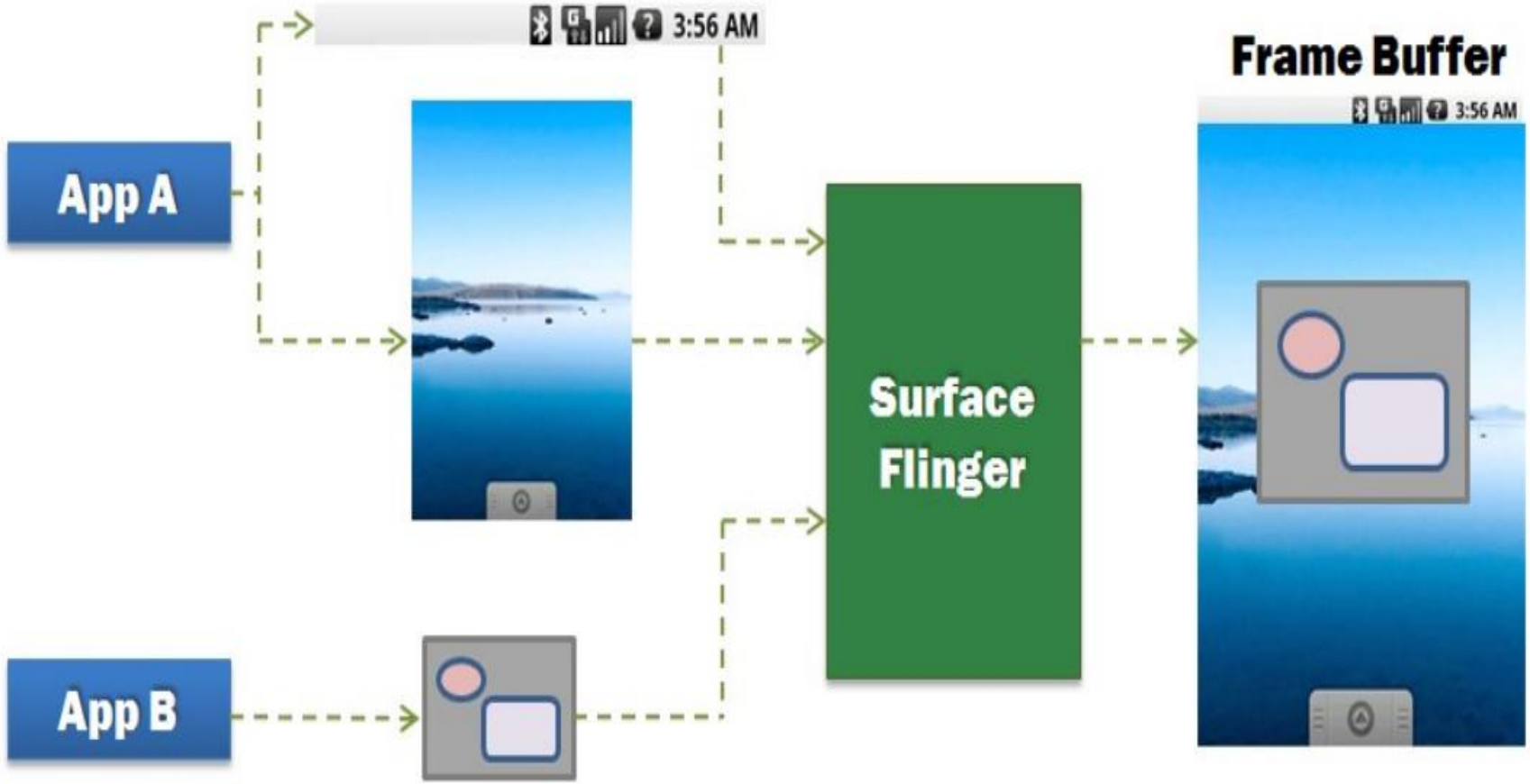
Turn on GPIO

Match the LCD framebuffer interface with « MYSCREEN » display

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough

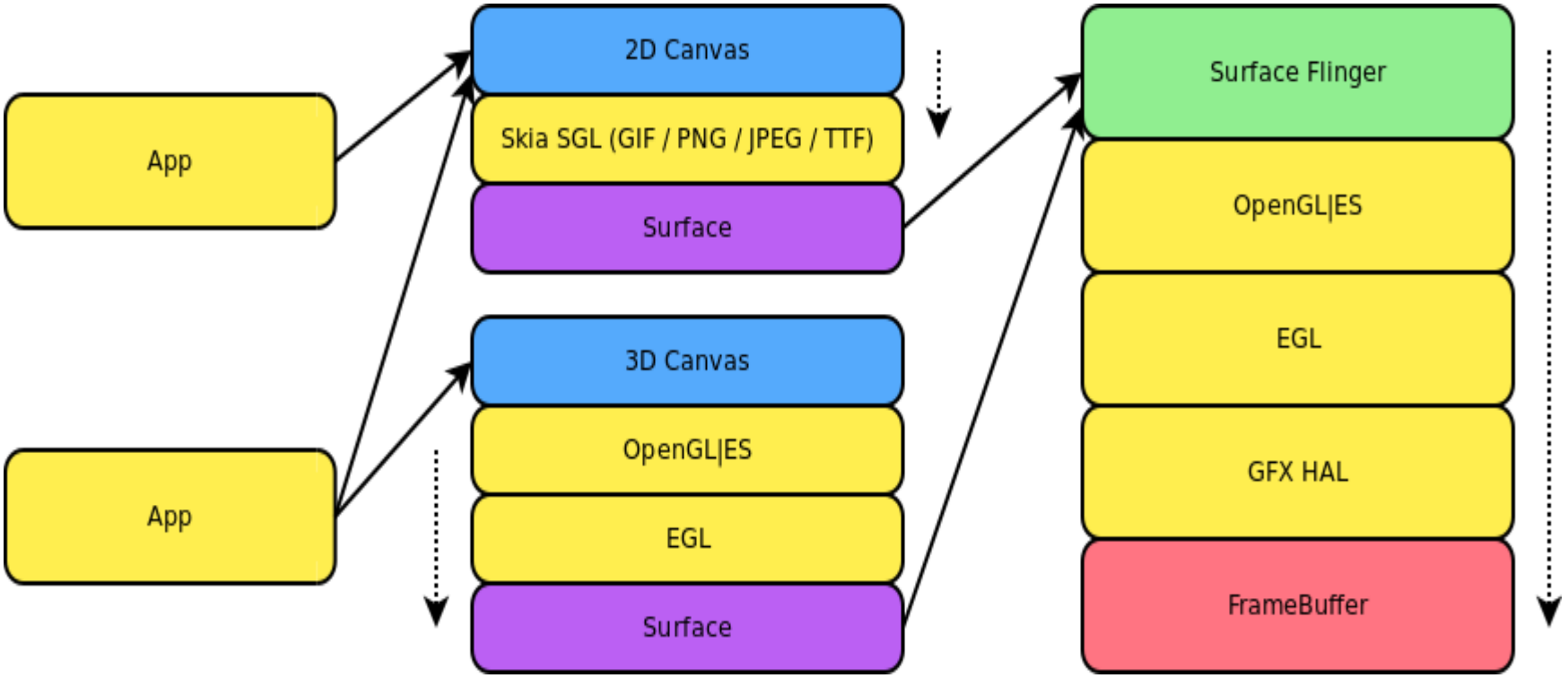
Surface Composition: Theory of Operations



COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough

Surface Composition: Theory of Operations



COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

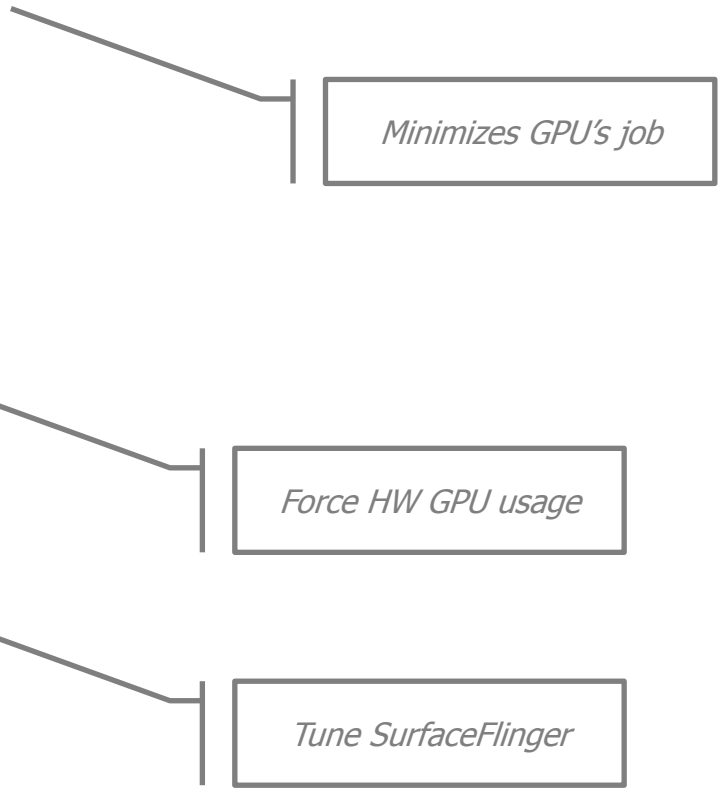
Jelly Bean Device Porting Walkthrough
2D/3D GPU

- Force HW GPU Usage
  - In **BoardConfig.mk**:
    - BOARD_EGL_CFG := device/company/my_device/egl.cfg
    - BOARD_USES_HGL := true
    - BOARD_USES_OVERLAY := true
    - USE_OPENGL_RENDERER := true
- Deploy (in **/vendor**) vendor-provided binary blobs of:
  - **Gralloc** and **HWcomposer** HAL Modules
  - OpenGL|ES 1.x and 2.x libraries.
- Optionally force **OpenGL|ES** libs version detection.
  - In **BoardConfig.mk**:
    - PRODUCT_PROPERTY_OVERRIDES += ro.opengles.version=131072
    - => Makes AngryBirds happy ... and so are we !

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
2D/3D GFX Optimizations

```
# 2D/3D acceleration
hwui.render_dirty_regions=false
debug.sf.hw=1
debug.sf.showfps=0
debug.sf.enable_hgl=1
debug.egl.hw=1
debug.composition.type=gpu
video.accelerate.hw=1
debug.performance.tuning=1
ro.max.fling_velocity=12000
ro.min.fling_velocity=8000
hwc.filter_stretch=1
persist.sys.ui.hw=1
windowsmgr.max_events_per_sec=55
```



COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
(GPU) Memory Management: Gralloc

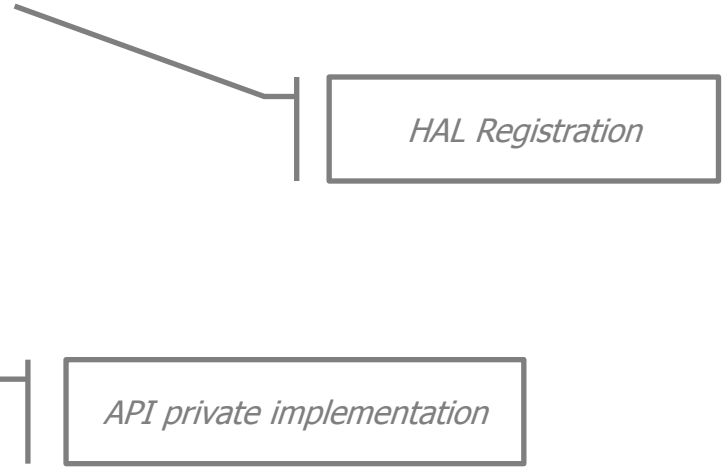
- Originally meant for GPU memory, but not only anymore !
- Implements **libhardware/include/hardware/gralloc.h**
- (Un)register and (un)lock memory buffers.
- Usually consist of contiguous memory (flag **GRALLOC_USAGE_FORCE_CONTIGUOUS**) for DMA operations.
- Jelly Bean now uses triple buffers for framebuffer.
- Support different kind of buffers:

  - GLES texture
  - GLES render surface
  - 2D HW blitter
  - HWComposer
  - Framebuffer
  - HW video encoder
  - Input HW camera pipeline
  - Output HW camera pipeline.

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
(GPU) Memory Management: Gralloc

```
struct private_module_t HAL_MODULE_INFO_SYM = {
    base: {
        common: {
            tag: HARDWARE_MODULE_TAG,
            version_major: 1,
            version_minor: 0,
            id: GRALLOC_HARDWARE_MODULE_ID,
            name: "Graphics Memory Allocator Module",
            author: "The Android Open Source Project",
            methods: &gralloc_module_methods
        },
        registerBuffer: gralloc_register_buffer,
        unregisterBuffer: gralloc_unregister_buffer,
        lock: gralloc_lock,
        unlock: gralloc_unlock,
    },
    framebuffer: 0,
    numBuffers: 0,
    bufferMask: 0,
    lock: PTHREAD_MUTEX_INITIALIZER,
    currentBuffer: 0,
};
```



COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
GPU Surface Composition: HWComposer

```
/* (*prepare)() is called for each frame before composition and is used by
 * SurfaceFlinger to determine what composition steps the HWC can handle. */
static int hwc_prepare(hwc_composer_device_t *dev, hwc_layer_list_t* list) {
        struct hwc_context_t *ctx = (struct hwc_context_t *)dev;
        if(ctx && ctx->m_viv_hwc && ctx->m_viv_hwc->prepare) {
                ctx->m_viv_hwc->prepare(ctx->m_viv_hwc, list);
        }
        return 0;
}

 /* (*set)() is used in place of eglSwapBuffers() */
static int hwc_set(hwc_composer_device_t *dev, hwc_display_t dpy,
                   hwc_surface_t sur, hwc_layer_list_t* list) {
    struct hwc_context_t *ctx = (struct hwc_context_t *) dev;

    if (ctx && ctx->m_viv_hwc && ctx->m_viv_hwc->set)
        return ctx->m_viv_hwc->set(ctx->m_viv_hwc, dpy, sur, list);
    else
        return eglSwapBuffers((EGLDisplay)dpy, (EGLSurface)sur);
}
```

- Relies on **gralloc.**
- Implements **hardware/libhardware/include/hardware/hwcomposer.h**
- Registers GPU specifics functions and maintains vertical synchronization.

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Phone vs. Tablet Mode

- LCD 800x480
  WVGA 7”
  (1.667 aspect ratio)

  => 133 dpi
  => Phone Mode.

- In frameworks/base/policy/src/com/android/internal/policy/impl/PhoneWindowManager.java:

```
// Determine whether the status bar can hide based on the size
// of the screen.  We assume sizes > 600dp are tablets where we
// will use the system bar.

int shortSizeDp = shortSize * DisplayMetrics.DENSITY_DEFAULT / DisplayMetrics.DENSITY_DEVICE;

 //tablet resolution width or height may define >= 480
//system according it to use system_bar or status_bar.
//and properity sys.devicy.type used to distinguish tablet and phone.

int standDp;
String deviceType = SystemProperties.get("sys.devicy.type");
if(! "".equals(deviceType) && deviceType.equals("tablet")) {
            standDp = 480;
} else {
            standDp = 600;
}
mStatusBarCanHide = shortSizeDp < standDp;
```

- Override in **BoardConfig.mk**:

```
PRODUCT_PROPERTY_OVERRIDES += ro.sf.lcd_density=120
PRODUCT_PROPERTY_OVERRIDES += sys.devicy.type=tablet
PRODUCT_CHARACTERISTICS := tablet
```


COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Custom Boot Logo

- Create a custom **bootanimation.zip** archive:

```
desc.txt
part0/f0000.jpg
...
part0/f0075.jpg
```

```
800 480 24
p 0 0 part0
```

- In **BoardConfig.mk**:
  - **PRODUCT_COPY_FILES += **
    device/company/my_device/bootanimation.zip:system/media/bootanimation.zip

- **Note:** Create the archive **_WITHOUT_** compression !!
  - => zip -0

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Enhanced Boot Animation

- Hack over **frameworks/base/cmds/bootanimation**:
  - Preloads the whole ZIP archive image files at once
  - Use an **OpenGL|ES** texture cache
  - **Bootanimation** doesn't stutter anymore !
  - => See patchset on **http://goo.gl/f1Lc4** (*CyanogenMod*)

- Hack over **BoardConfig.mk**:

```
# Bootanimation optimizations
TARGET_BOOTANIMATION_PRELOAD := true
TARGET_BOOTANIMATION_TEXTURE_CACHE := true
```

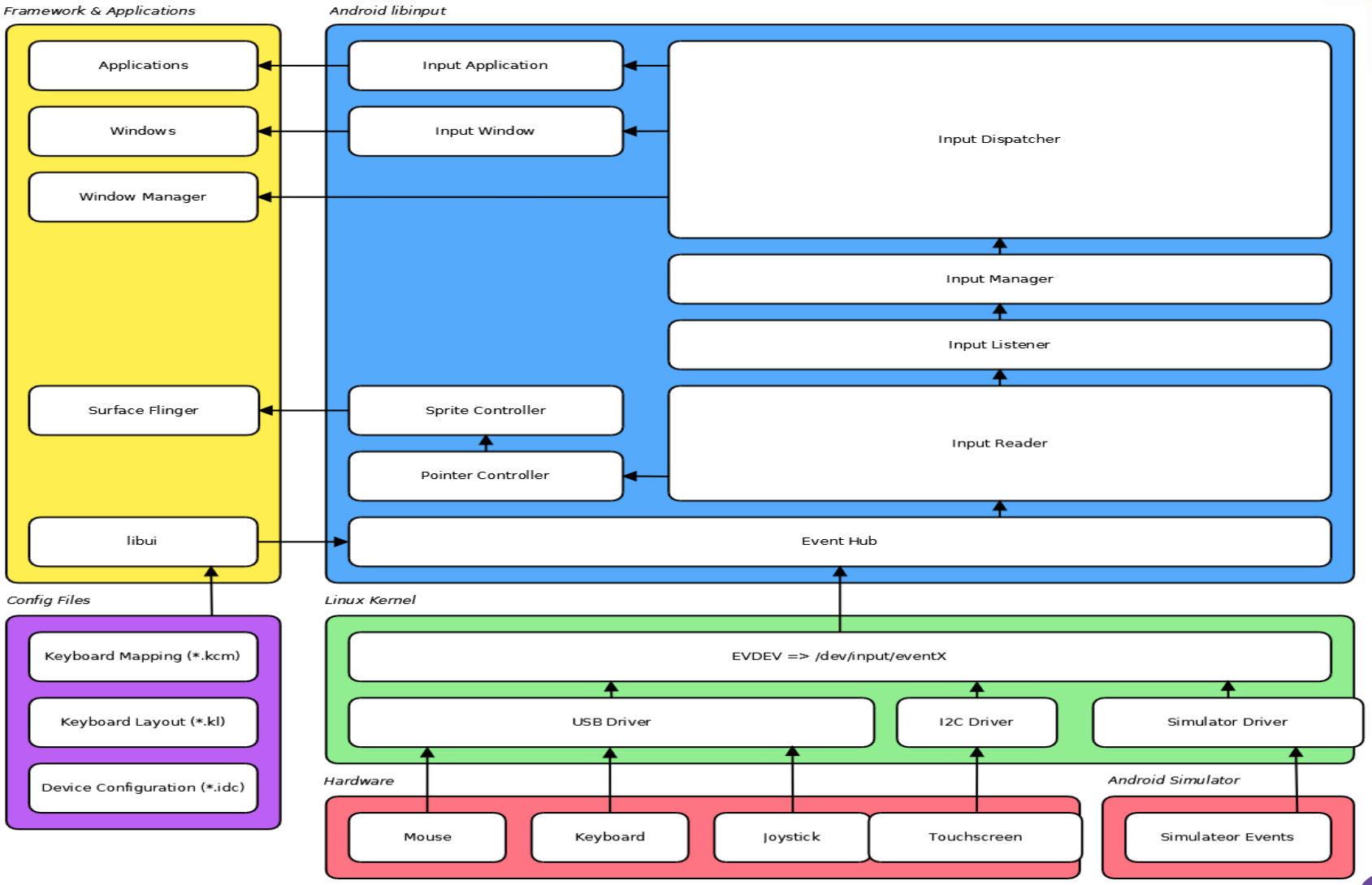

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Input Layer

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough

Input Layer Architecture



COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Input Device Configuration Files (.idc)

- Contains device-specific configuration properties that affect the behavior of input devices.

- Optional for standard peripherals such as HID keyboard and mouse.
- Mandatory for built-in embedded devices such as touch screens.

- Location:
  - Located by USB vendor, product (and optionally version) id or by input device name.
  - The following paths are consulted in order:

```
/system/usr/idc/Vendor_XXXX_Product_XXXX_Version_XXXX.idc
/system/usr/idc/Vendor_XXXX_Product_XXXX.idc
/system/usr/idc/DEVICE_NAME.idc

/data/system/devices/idc/Vendor_XXXX_Product_XXXX_Version_XXXX.idc
/data/system/devices/idc/Vendor_XXXX_Product_XXXX.idc
/data/system/devices/idc/DEVICE_NAME.idc
```


COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Input Key Layout Files (.kl)

- Maps Linux key and axis codes to Android key and axis codes.
- Required for all internal (built-in) input devices that have keys, including special keys such as volume, power and headset media keys.
- Location:
  - Located by USB vendor, product (and optionally version) id or by input device name.
  - The following paths are consulted in order:

```
/system/usr/keylayout/Vendor_XXXX_Product_XXXX_Version_XXXX.kl
/system/usr/keylayout/Vendor_XXXX_Product_XXXX.kl
/system/usr/keylayout/DEVICE_NAME.kl
/system/usr/keylayout/Generic.kl

/data/system/devices/keylayout/Vendor_XXXX_Product_XXXX_Version_XXXX.kl
/data/system/devices/keylayout/Vendor_XXXX_Product_XXXX.kl
/data/system/devices/keylayout/DEVICE_NAME.kl
/data/system/devices/keylayout/Generic.kl
```


COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Input Key Layout Files (.kl)

- Syntax:
  - **key - Linux scan code number - Android key code name - [policy flags]**

```
key  1     ESCAPE
key  114   VOLUME_DOWN WAKE
key  16    Q                    VIRTUAL  WAKE
```

- Policy flags:
  - **WAKE|WAKE_DROPPED**: The key should wake the device when it is asleep.
  - **SHIFT**: The key should be interpreted as if the SHIFT key was also pressed.
  - **CAPS_LOCK**: The key should be interpreted as if the CAPS LOCK key was also pressed.
  - **ALT**: The key should be interpreted as if the ALT key was also pressed.
  - **ALT_GR**: The key should be interpreted as if the RIGHT ALT key was also pressed.
  - **FUNCTION**: The key should be interpreted as if the FUNCTION key was also pressed.
  - **VIRTUAL**: The key is a virtual soft key that is adjacent to the main touch screen.

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Input Key Character Map (.kcm)

- Maps Android key codes with Unicode characters.

- Location:
  - Located by USB vendor, product (and optionally version) id or by input device name.
  - The following paths are consulted in order:

```
/system/usr/keychars/Vendor_XXXX_Product_XXXX_Version_XXXX.kcm
/system/usr/keychars/Vendor_XXXX_Product_XXXX.kcm
/system/usr/keychars/DEVICE_NAME.kcm

/data/system/devices/keychars/Vendor_XXXX_Product_XXXX_Version_XXXX.kcm
/data/system/devices/keychars/Vendor_XXXX_Product_XXXX.kcm
/data/system/devices/keychars/DEVICE_NAME.kcm

/system/usr/keychars/Generic.kcm
/data/system/devices/keychars/Generic.kcm

/system/usr/keychars/Virtual.kcm
/data/system/devices/keychars/Virtual.kcm
```


COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
I2C Touchscreen

```
#define TP_IRQ            IMX_GPIO_NR(2, 4)
#define TP_POWER_ENABLE   IMX_GPIO_NR(1, 7)

static struct cy8ctmg110_pdata ts_data __initdata = {
        .irq_pin        = TP_IRQ,
        .reset_pin      = TP_POWER_ENABLE,
};

static struct i2c_board_info mxc_i2c1[] __initdata = {
        {
                I2C_BOARD_INFO("cy8ctmg110", 0x20),
                .platform_data = &ts_data,
        },
};

gpio_request(TP_POWER_ENABLE, "tp_power");
gpio_direction_output(TP_POWER_ENABLE, 1);
i2c_register_board_info(1, mxc_i2c1, ARRAY_SIZE(mxc_i2c1));
```

Set IRQ/Reset pins

Match I2C address with driver's name

Turn on GPIO

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
I2C Touchscreen Driver

- **Initialization (driver name MUST match .idc filename):**

```
input_dev->name = CY8CTMG110_DRIVER_NAME;
```

```
input_dev->keybit[BIT_WORD(BTN_TOUCH)] = BIT_MASK(BTN_TOUCH);
input_dev->absbit[ABS_PRESSURE] = BIT(ABS_PRESSURE);
input_set_abs_params(input_dev, ABS_PRESSURE, 0, 1, 0, 0);
```

```
input_set_abs_params(input_dev, ABS_X, CY8CTMG110_X_MIN,
CY8CTMG110_X_MAX, 4, 0);
input_set_abs_params(input_dev, ABS_Y, CY8CTMG110_Y_MIN,
CY8CTMG110_Y_MAX, 4, 0);
```

- **Touch Press & Release Events:**

```
static void ts_press_on(struct input_dev *input, int x, int y) {
  input_report_abs(input, ABS_X, x);
  input_report_abs(input, ABS_Y, y);
  input_report_abs(input, ABS_PRESSURE, 1);
  input_report_key(input, BTN_TOUCH, 1);
  input_sync(input);
}
static void ts_press_off(struct input_dev *input, int x, int y) {
  input_report_abs(input, ABS_PRESSURE, 0);
  input_report_key(input, BTN_TOUCH, 0);
  input_sync(input);
}
```

Mandatory since ICS

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Touchscreen Device Configuration File

- Add **cy8ctmg110.idc** to system:
  - In **device.mk**, add:
    - PRODUCT_COPY_FILES +=
      device/company/my_device/cy8ctmg110.idc:system/usr/idc/cy8ctmg110.idc

- Example of **cy8ctmg110.idc**:

```
# Basic Parameters
 touch.deviceType = touchscreen
 touch.orientationAware = 0

# Size
 touch.size.calibration = none

# Orientation
 touch.pressure.calibration = none
```


COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
I2C Sense Keys

```
#define SENSEKEY_IRQ              IMX_GPIO_NR(6, 14)
#define SENSEKEY_POWER_ENABLE     IMX_GPIO_NR(6, 16)

static struct sensekey_pdata sensekey_data  __initdata = {
        .irq_pin        = SENSEKEY_IRQ,
        .reset_pin      = SENSEKEY_POWER_ENABLE,
};

static struct i2c_board_info mxc_i2c2[] __initdata = {
        {
                I2C_BOARD_INFO("so5g2000", 0x2d),
                .platform_data = &sensekey_data,
        },
 };

i2c_register_board_info(2, mxc_i2c2, ARRAY_SIZE(mxc_i2c2));
```

Set IRQ/Reset pins

Match I2C address with driver's name

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
I2C Sense Keys Configuration Files

- Example of **sensekey.kcm**:

```
type SPECIAL_FUNCTION
```

- Example of **sensekey.kl**:

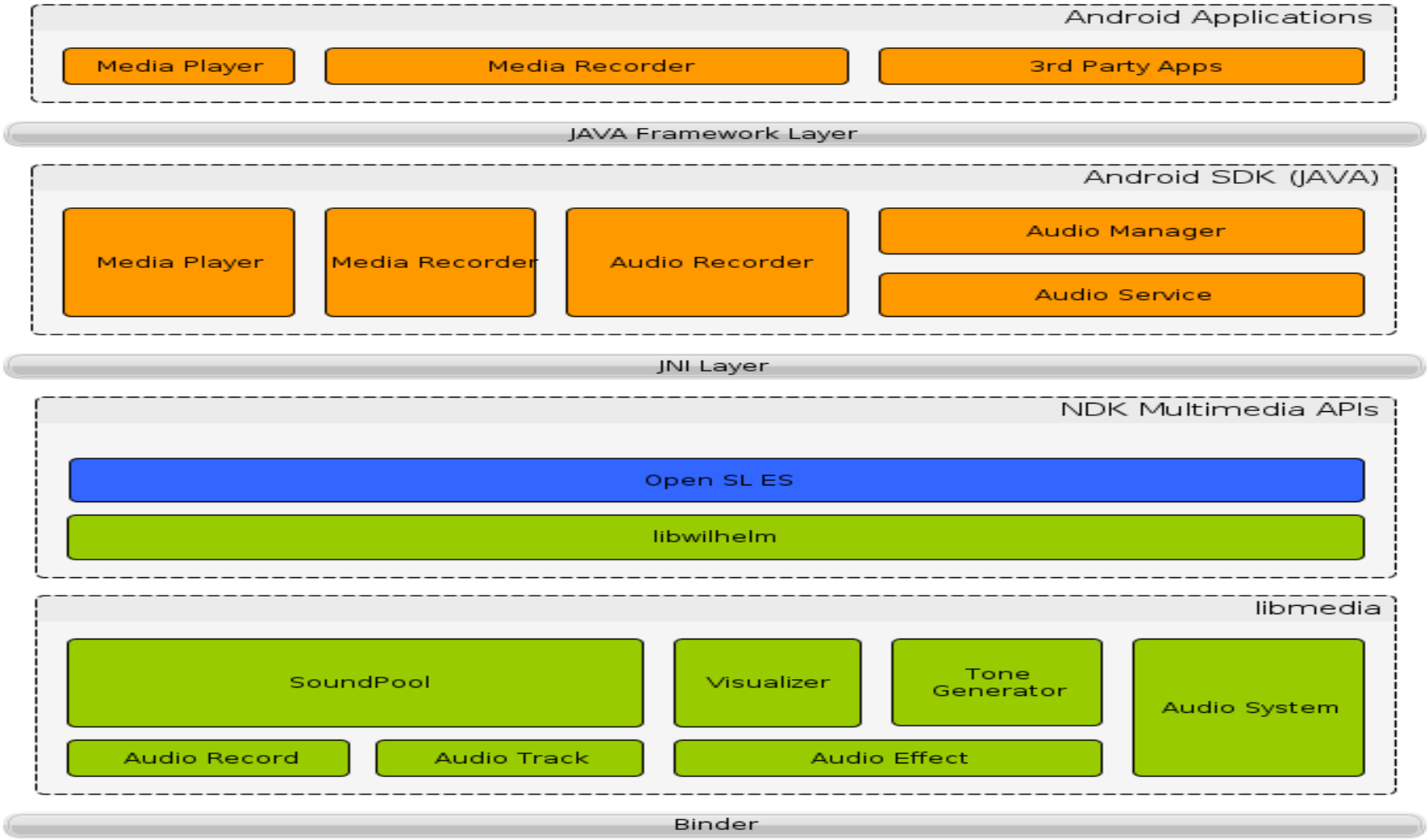
```
key 113   VOLUME_MUTE    WAKE    # KEY_MUTE
key 114   VOLUME_DOWN    WAKE    # KEY_VOLUMEDOWN
key 115   VOLUME_UP      WAKE    # KEY_VOLUMEUP
key 392   HEADSETHOOK    WAKE    # KEY_AUDIO
key 139   MENU           WAKE    # KEY_MENU
key 169   CONTACTS       WAKE    # KEY_PHONE
key 374   CALL           WAKE    # KEY_KEYBOARD
key 143   CALENDAR       WAKE    # KEY_WAKEUP
key 102   HOME           WAKE    # KEY_HOME
```


COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Audio Subsystem

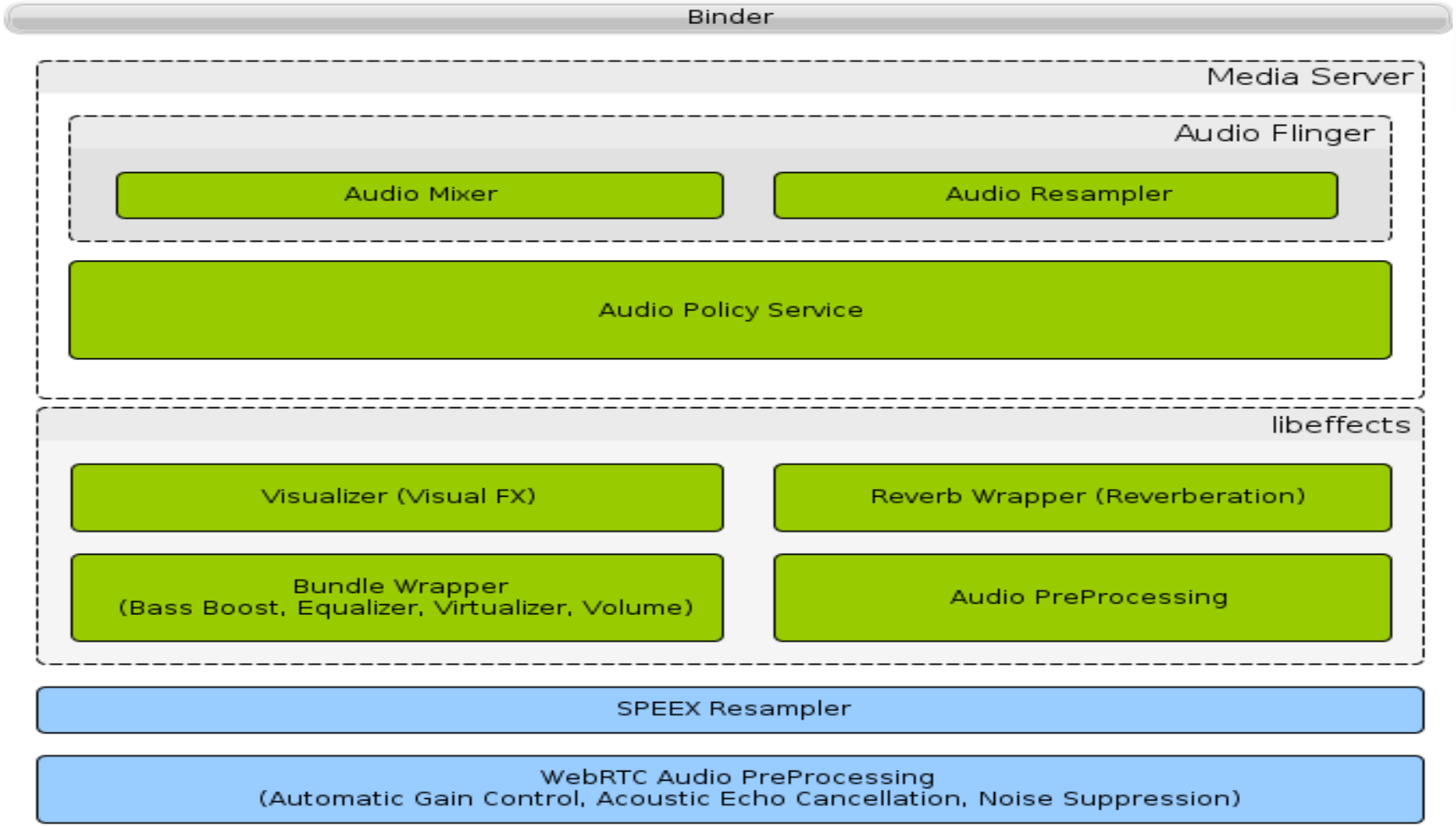
COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Audio SW Architecture



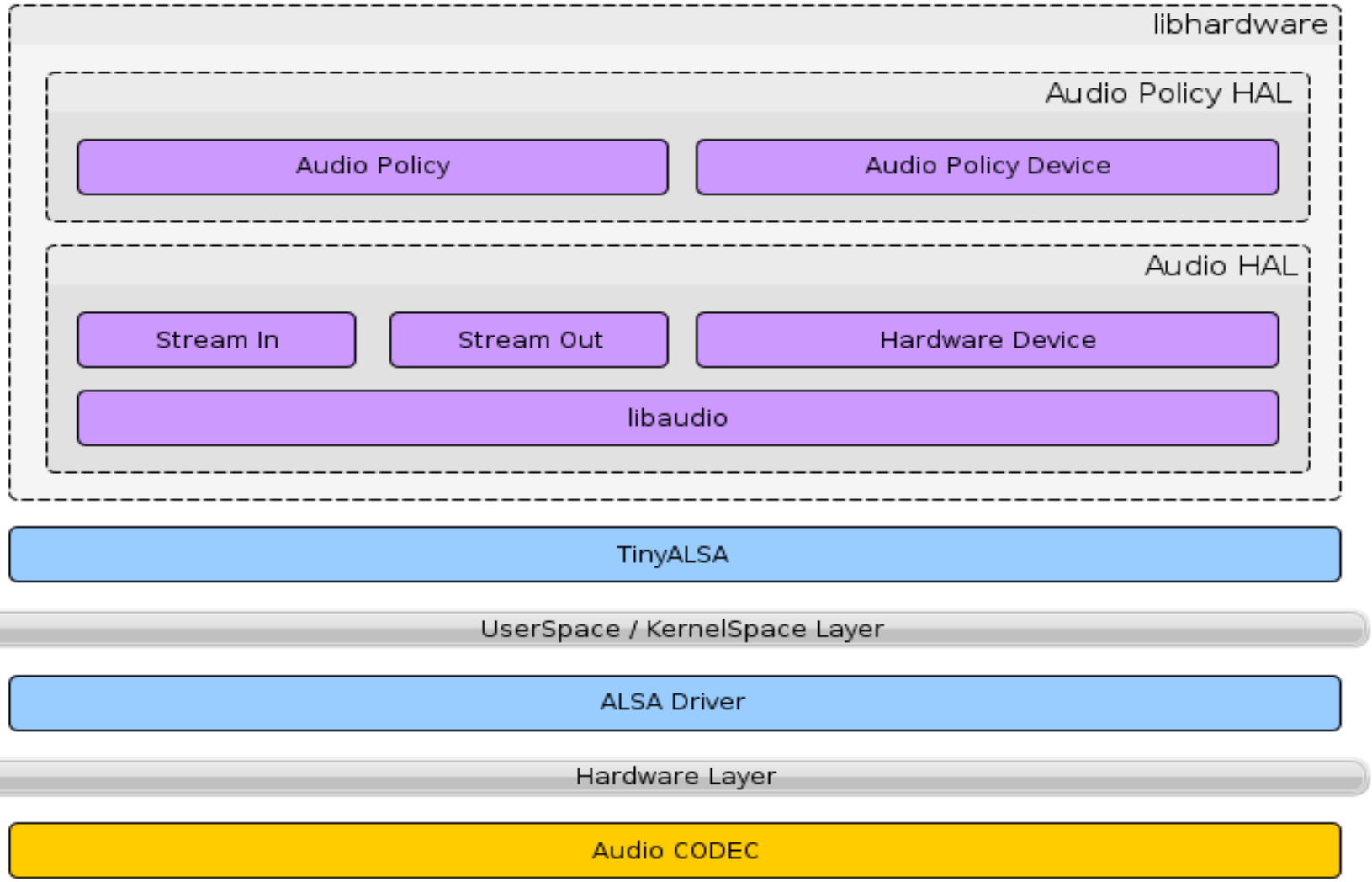
COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Audio SW Architecture



COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Audio SW Architecture



COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Audio Codec Driver

```
static struct platform_device audio_codec_device = {
        .name = "imx-wm8958",
};

static struct wm8994_pdata wm8958_config_data = {
        .gpio_defaults = {
                [0] = WM8994_GP_FN_GPIO | WM8994_GPN_DB,
                [1] = WM8994_GP_FN_GPIO | WM8994_GPN_DB | WM8994_GPN_PD,
                [...],
        },
};

static struct i2c_board_info mxc_i2c0[] __initdata = {
        {
                I2C_BOARD_INFO("wm8958", 0x1a),
                .platform_data = &wm8958_config_data,
                .irq = gpio_to_irq(IMX_GPIO_NR(1, 21)),
        },
};
```

Loads up platform-specific driver

Match I2C address with driver's name

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Audio Subsystem

- In **BoardConfig.mk**:
  - BOARD_USES_ALSA_AUDIO := true
    BUILD_WITH_ALSA_UTILS := true
  - PRODUCT_COPY_FILES += audio.primary.imx6
  - PRODUCT_COPY_FILES += audio_policy.conf

- MUST implement Audio HAL
  - See **hardware/libhardware/modules/audio** for example.
  - MUST declare as AUDIO_HARDWARE_MODULE_ID.
  - MUST implement audio HAL API; see **hardware/libhardware/include/hardware/audio.h**.
  - Implements audio in/out, volume get/set, mute get/set, gain get/set, list and configure all supported in/out devices, open/close input/output streams, configure mixer and router, add/remove audio effects (e.g. Automatic Gain Control and Acoustic Echo Cancellation).
  - MUST link against **TinyALSA** (new in ICS/JB) and **libaudioutils**.
  - To be implemented in: **device/company/my_device/audio/audio_hw.c**
  - MUST be declared in LOCAL_MODULE as "audio.primary.${ro.product.board}"
    - e.g. : audio.primary.imx6

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Audio HAL

```
static struct audio_card  wm8958_card = {
    .name = "wm8958-audio",
    .driver_name = "wm8958-audio",
    .supported_devices = (AUDIO_DEVICE_OUT_EARPIECE |
            AUDIO_DEVICE_OUT_SPEAKER | AUDIO_DEVICE_OUT_WIRED_HEADSET |
            AUDIO_DEVICE_OUT_WIRED_HEADPHONE | AUDIO_DEVICE_OUT_ANLG_DOCK_HEADSET |
            AUDIO_DEVICE_OUT_DGTL_DOCK_HEADSET | AUDIO_DEVICE_OUT_ALL_SCO |
            AUDIO_DEVICE_OUT_DEFAULT |
            /* IN */
            AUDIO_DEVICE_IN_COMMUNICATION | AUDIO_DEVICE_IN_AMBIENT |
            AUDIO_DEVICE_IN_BUILTIN_MIC | AUDIO_DEVICE_IN_WIRED_HEADSET |
            AUDIO_DEVICE_IN_BACK_MIC | AUDIO_DEVICE_IN_ALL_SCO |
            AUDIO_DEVICE_IN_DEFAULT),
    [...]
    .vx_bt_mic_input      = vx_bt_mic_input_wm8958,
    .mm_bt_mic_input      = mm_bt_mic_input_wm8958,
    [...]
};
```

Define Android framework supported input/output devices

Private API implementation

Turn on ALSA driver control name

```
static struct route_setting vx_bt_mic_input_wm8958[] = {
    {
        .ctl_name = "AIF1ADC1L Mixer AIF2 Switch",
        .intval = 1,
    },
    {
        .ctl_name = "AIF1ADC1R Mixer AIF2 Switch",
        .intval = 1,
    },
    {
        .ctl_name = NULL,
    },
};
```


COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough

Audio Policy (/etc/audio_policy.conf)

```
audio_hw_modules {
  primary {
    outputs {
      primary {
        sampling_rates 44100
        channel_masks AUDIO_CHANNEL_OUT_STEREO
        formats AUDIO_FORMAT_PCM_16_BIT
        devices AUDIO_DEVICE_OUT_EARPIECE|AUDIO_DEVICE_OUT_SPEAKER|AUDIO_DEVICE_OUT_WIRED_HEADSET|
AUDIO_DEVICE_OUT_WIRED_HEADPHONE|AUDIO_DEVICE_OUT_AUX_DIGITAL
        flags AUDIO_OUTPUT_FLAG_PRIMARY
      }
      hdmi {
        sampling_rates dynamic
        channel_masks dynamic
        formats AUDIO_FORMAT_PCM_16_BIT
        devices AUDIO_DEVICE_OUT_AUX_DIGITAL
        flags AUDIO_OUTPUT_FLAG_DIRECT
      }
    }
    inputs {
      primary {
        sampling_rates 8000|11025|16000|22050|24000|32000|44100|48000
        channel_masks AUDIO_CHANNEL_IN_MONO|AUDIO_CHANNEL_IN_STEREO
        formats AUDIO_FORMAT_PCM_16_BIT
        devices
AUDIO_DEVICE_IN_BUILTIN_MIC|AUDIO_DEVICE_IN_WIRED_HEADSET|AUDIO_DEVICE_IN_BACK_MIC|AUDIO_DEVICE_IN_USB_MIC
      }
    }
  }
}
```

List supported sampling rates, formats, devices … for each audio.<module>.so

```
global_configuration {
  attached_output_devices AUDIO_DEVICE_OUT_EARPIECE|AUDIO_DEVICE_OUT_SPEAKER
  default_output_device AUDIO_DEVICE_OUT_SPEAKER
  attached_input_devices AUDIO_DEVICE_IN_BUILTIN_MIC|AUDIO_DEVICE_IN_BACK_MIC
}
```

Turn on GPIO

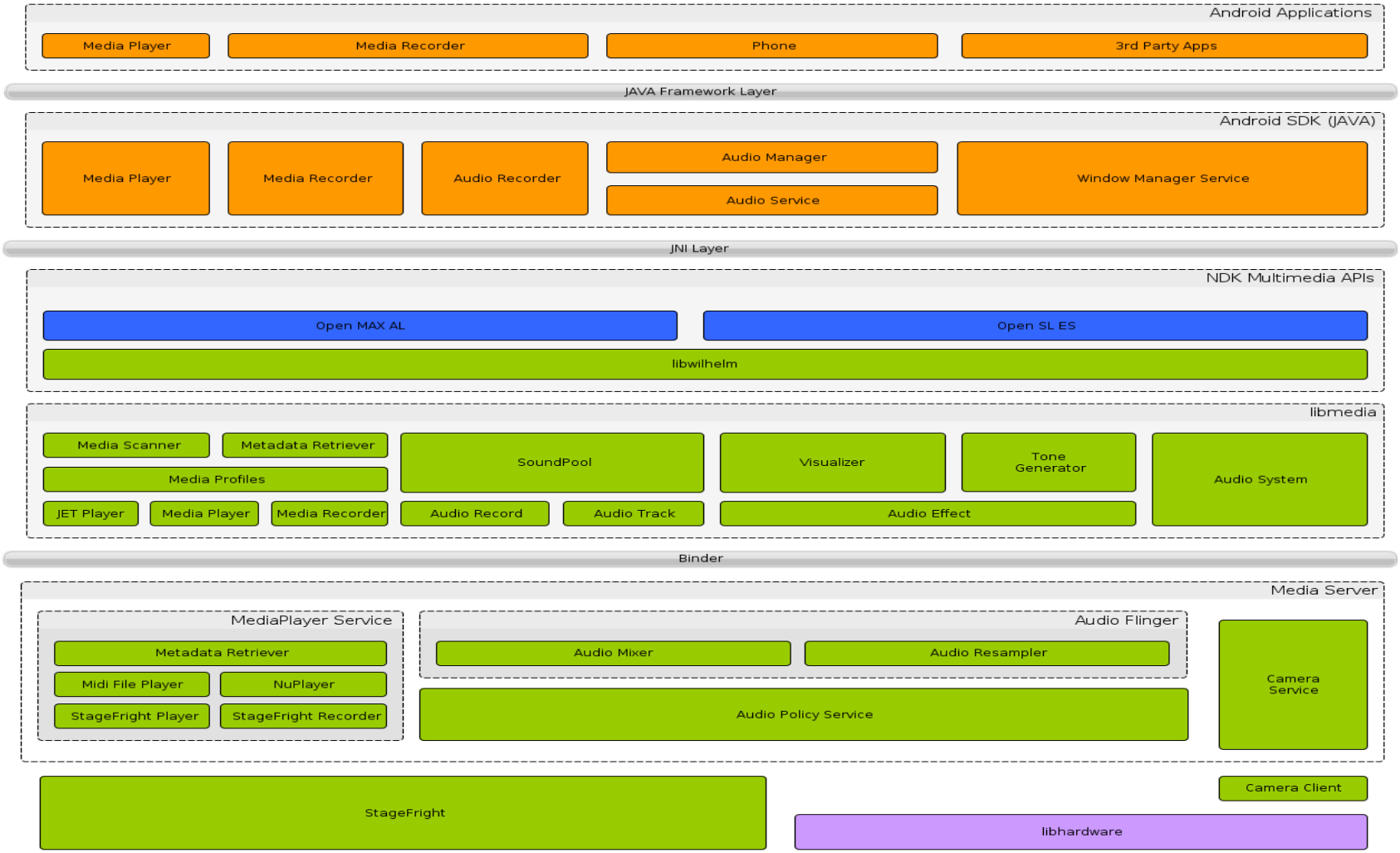
COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Multimedia Subsystem

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough

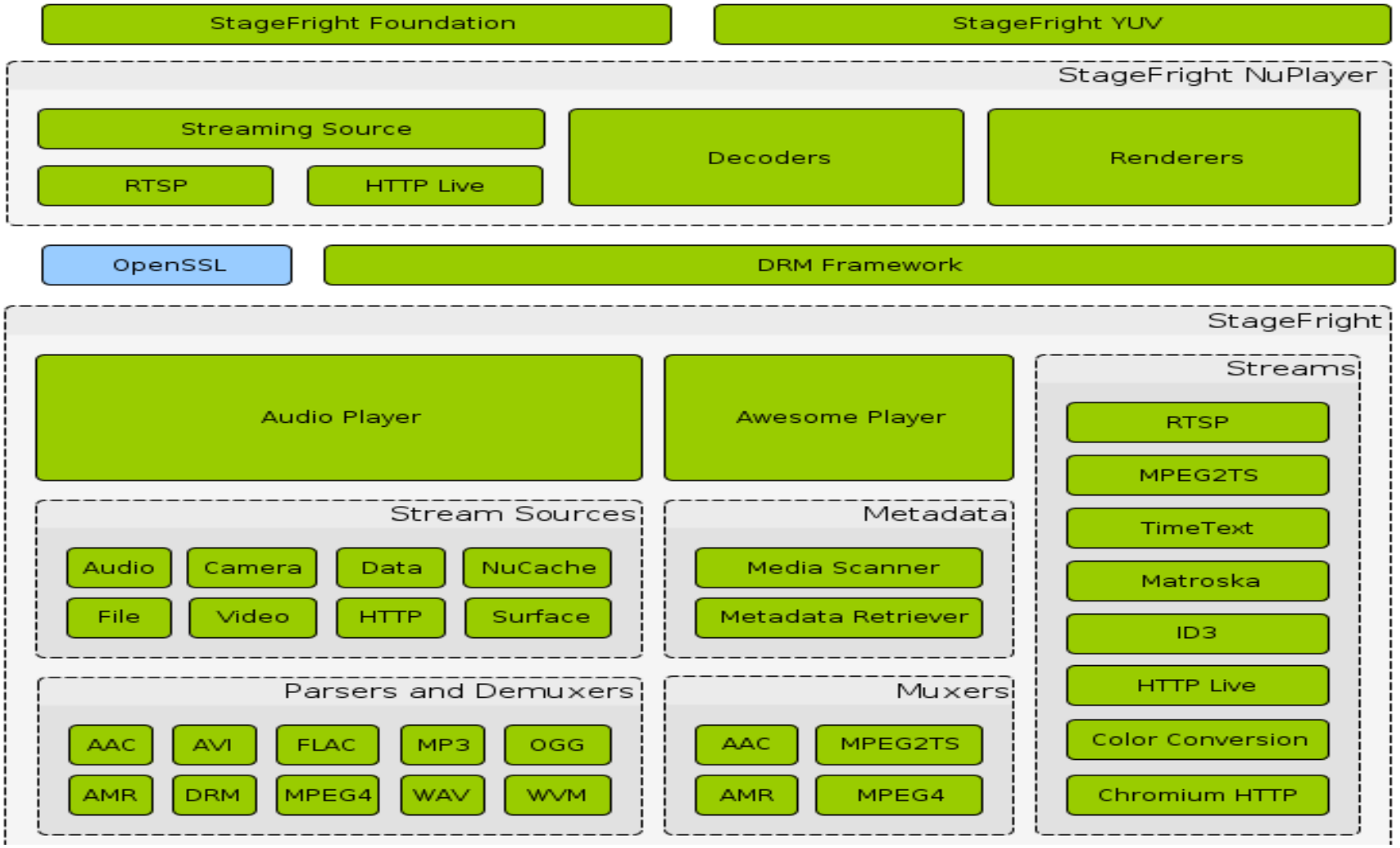
Multimedia SW Architecture



COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough

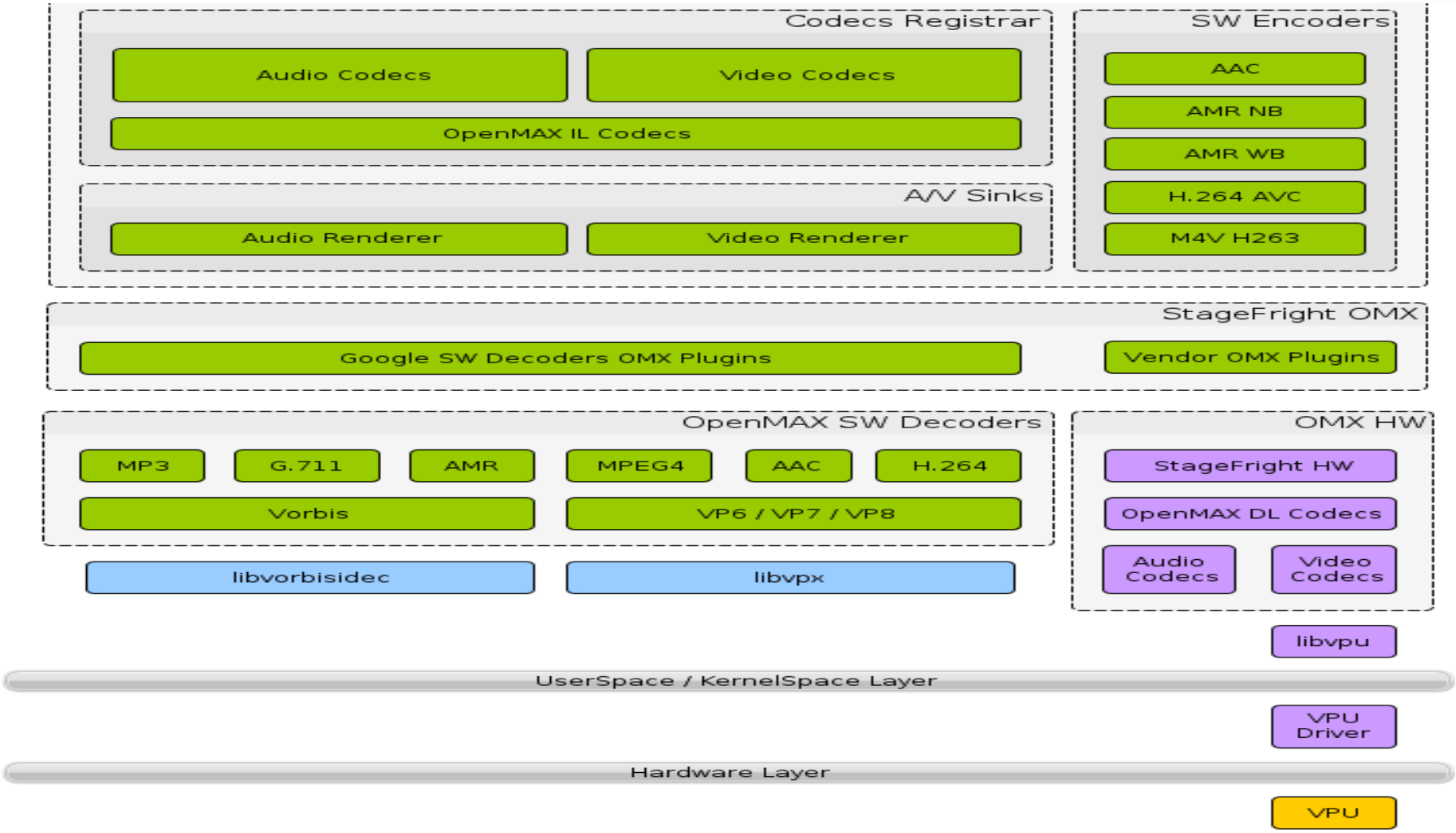
StageFright SW Architecture



COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough

StageFright SW Architecture



COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Camera Driver

```
#define OV5640_CSI_PWN          IMX_GPIO_NR(7, 8)
#define OV5640_CSI_RST          IMX_GPIO_NR(1, 20)

static void ov5640_csi_powerdown(int powerdown) {
        gpio_set_value(VHE3G_OV5640_CSI_PWN, powerdown);
}

static void ov5640_csi_sensor_io_init(void) {
        mxc_iomux_v3_setup_multiple_pads (mipi_sensor_pads, ARRAY_SIZE(mipi_sensor_pads));

        gpio_request(OV5640_CSI_RST, "cam-reset");
        gpio_direction_output(OV5640_CSI_RST, 1);

        gpio_request(OV5640_CSI_PWN, "cam-pwdn");
        gpio_direction_output(OV5640_CSI_PWN, 0);
}

static struct fsl_mxc_camera_platform_data ov5640_csi_data = {
        .io_init = ov5640_csi_sensor_io_init,
        .pwdn = ov5640_csi_powerdown,
};

static struct i2c_board_info mxc_i2c0[] __initdata = {
        {
                I2C_BOARD_INFO("ov5640_mipi", 0x3c),
                .platform_data = (void *)&ov5640_csi_data,
        },
};
```

Turn off GPIO for powerdown

Turn on GPIO for reset and power-up

Match I2C address to driver's name

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Camera HAL

- Major update in JellyBean which now features 2 APIs:
  - 1.0: **libhardware/include/hardware/camera.h**
  - 2.0: **libhardware/include/hardware/camera2.h**
- Implements start/stop recording, camera control, preview on/off …

Refer to:

"Camera 2.0: The New Camera Hardware Interface in Android 4.2"
by *Balwinder Kaur* & *Ashutosh Gupta* from *Aptina*
Same time and place ;-(

« Android 4.2 was released with a new Camera 2.0 HAL. Camera 2.0 has a big emphasis on collection and providing metadata associated with each frame. It also provides the ability to re-process streams. »

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Media Profiles

- In **media_profiles.xml**:
  - XML file that lists audio/video/image encoder/decoder capabilities.

```
<!-- If a codec is not enabled, it is invisible to the applications.
In other words, the applications won't be able to use the codec or query
the capabilities of the codec at all if it is disabled -->
<VideoEncoderCap name="h264" enabled="true"
                 minBitRate="64000" maxBitRate="3000000"
                 minFrameWidth="176" maxFrameWidth="800"
                 minFrameHeight="144" maxFrameHeight="480"
                 minFrameRate="1" maxFrameRate="30" />
```

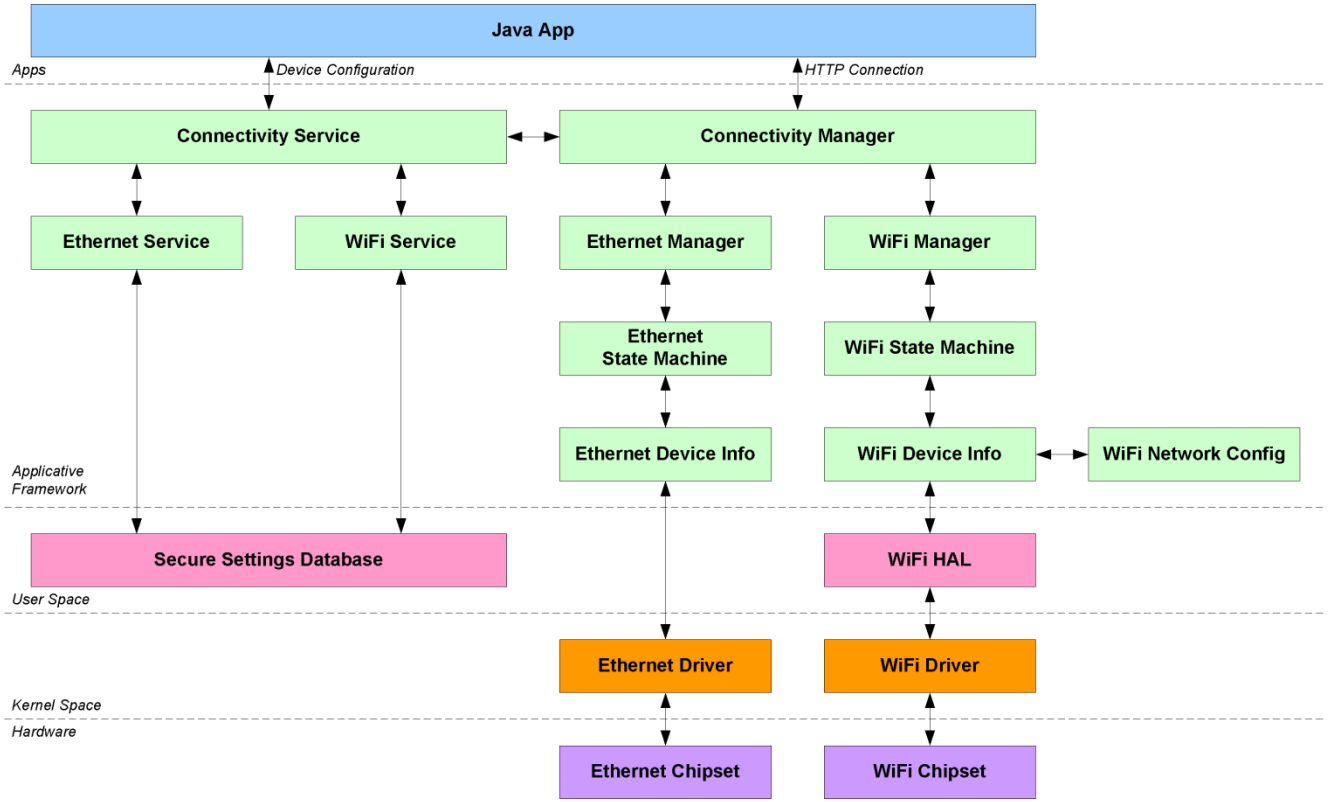
  - No checks on decoder capabilities so far.

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Connectivity Subsystem

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Network & Connectivity



- Low-level kernel drivers work just like a charm up to Linux user-space.
- For Java apps connectivity, each connection type must register a specific **ConnectivityManager** and associated **ConnectivityService** that handles device configuration, packet routing and HTTP(S) proxy settings.

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Network & Connectivity

- Overlay some resources in **frameworks/base/core/res/res/values/config.xml**:

```
<!-- Used by the connectivity manager to decide which networks can coexist -->
<string-array translatable="false" name="networkAttributes">
        <item>"wifi,1,1,1,-1,true"</item>
        <item>"bluetooth,7,7,0,-1,true"</item>
        <item>"ethernet,9,9,2,-1,true"</item>
</string-array>
<string-array translatable="false" name="radioAttributes">
        <item>"1,1"</item>
        <item>"7,1"</item>
        <item>"9,1"</item>
</string-array>
```


COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Ethernet Connectivity

Refer to:
"**Dive Into Android Networking: Adding Ethernet Connectivity**"

ABS 2013, *Benjamin Zores*,
19th February 2013.

Or find it back on **SlideShare**.

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Wi-Fi Driver

```
#define WLAN_IRQ                IMX_GPIO_NR(1, 28)
#define WLAN_EN                 IMX_GPIO_NR(2, 23)

static const struct esdhc_platform_data wlan_sdio_data = {
        .always_present         = 1,
        .keep_power_at_suspend  = 1,
        .support_8bit           = 0,
        .support_18v            = 1,
        .delay_line             = 0,
        .cd_type                = ESDHC_CD_PERMANENT,
};

static struct regulator_init_data vmmc_wlan_init = {
        .constraints = {
                .valid_ops_mask         = REGULATOR_CHANGE_STATUS,
        },
        .num_consumer_supplies  = 1,
        .consumer_supplies      = vmmc_wlan_consumers,
};

static struct fixed_voltage_config vmmc_wlan_reg_config = {
        .supply_name            = "vwl12731",
        .microvolts             = 1800000, /* 1.8V */
        .gpio                   = WLAN_EN,
        .startup_delay          = 70000, /* 70msec */
        .enable_high            = 1,
        .enabled_at_boot        = 0,
        .init_data              = &vmmc_wlan_init,
};
```

Wi-Fi SDIO is 4-bit only

Our chip is wired on board, so we don't need card-detect

Watch out for voltage (1.8V) and don't burn your chip !

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Wi-Fi Driver

```
static struct platform_device wlan_devices = {
        .name                           = "reg-fixed-voltage",
        .id                             = 4,
        .dev    = {
                .platform_data          = &vmmc_wlan_reg_config,
        },
};

struct wl12xx_platform_data wlan_data  __initdata = {
        .irq                            = gpio_to_irq(WLAN_IRQ),
        .board_ref_clock                = WL12XX_REFCLOCK_26_XTAL,
};

gpio_request(WLAN_EN, "wlan_power");
gpio_direction_output(WLAN_EN, 1);
imx6q_add_sdhci_usdhc_imx(2, &wlan_sdio_data);
wl12xx_set_platform_data(&wlan_data);
```

Set IRQ pin

Reference clock may vary from one packager to another

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Wi-Fi Connectivity

- HAL driver in **hardware/libhardware_legacy/wifi**.
  - Default implementation should be sufficient in most cases.
  - Loads/unloads kernel drivers, loads up firmware and registers UNIX socket connection to WPA supplicant for further control.
  - Used by JNI Java framework.

```
BOARD_HAVE_WIFI := true

WIFI_DRIVER_MODULE_PATH          := "/system/lib/modules/wl12xx_sdio.ko"
WIFI_DRIVER_MODULE_NAME          := "wl12xx_sdio"
WIFI_FIRMWARE_LOADER             := ""

PRODUCT_COPY_FILES += \
        kernel_imx/drivers/net/wireless/wl12xx/wl12xx_sdio.ko:system/lib/modules/wl12xx_sdio.ko

BOARD_WPA_SUPPLICANT_DRIVER      := WEXT
WPA_SUPPLICANT_VERSION           := VER_0_8_X
BOARD_WPA_SUPPLICANT_PRIVATE_LIB := private_lib_driver_cmd

PRODUCT_COPY_FILES +=
frameworks/base/data/etc/android.hardware.wifi.xml:system/etc/permissions/android.hardware.wifi.xml
```

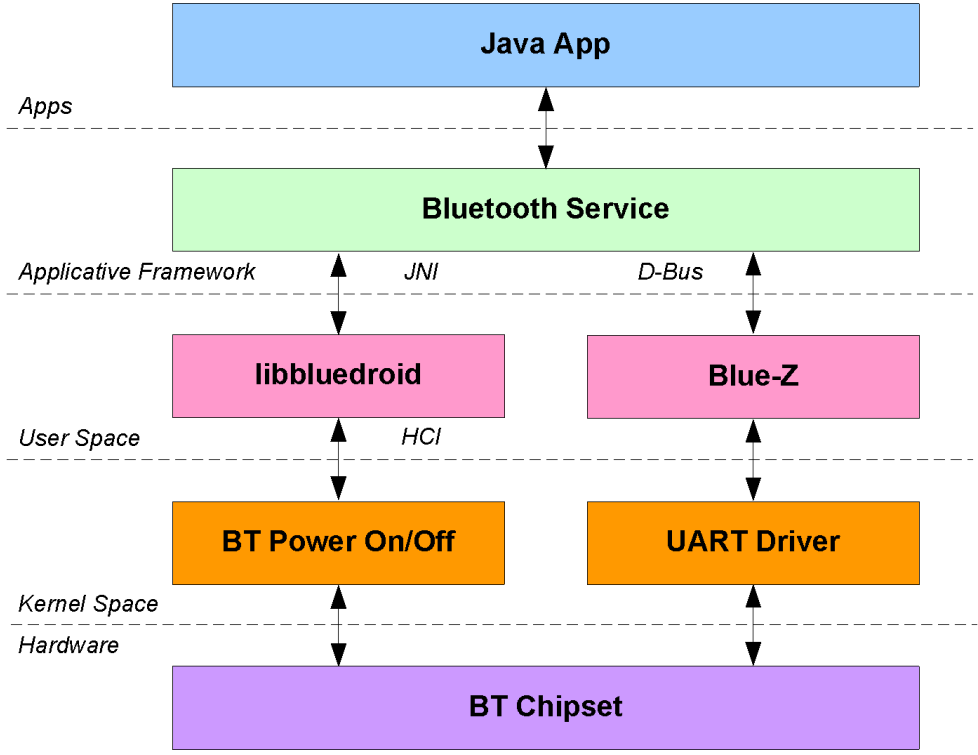
Used by Wi-Fi HAL

Build a WEXT compatible WPA Supplicant extension driver (derived from Broadcom reference one)

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
ICS Bluetooth Architecture

- In **BoardConfig.mk**:
  - BOARD_HAVE_BLUETOOTH := true
- Add Bluetooth hardware permission, in /system/etc/permissions/
  - <feature name="android.hardware.bluetooth" />
- May requires support in audio HAL for BT A2DP support.
- **libbluedroid** implements enables/disables BT interface and creates HCI socket through rfkill (see **system/bluetooth/bluedroid**).



COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

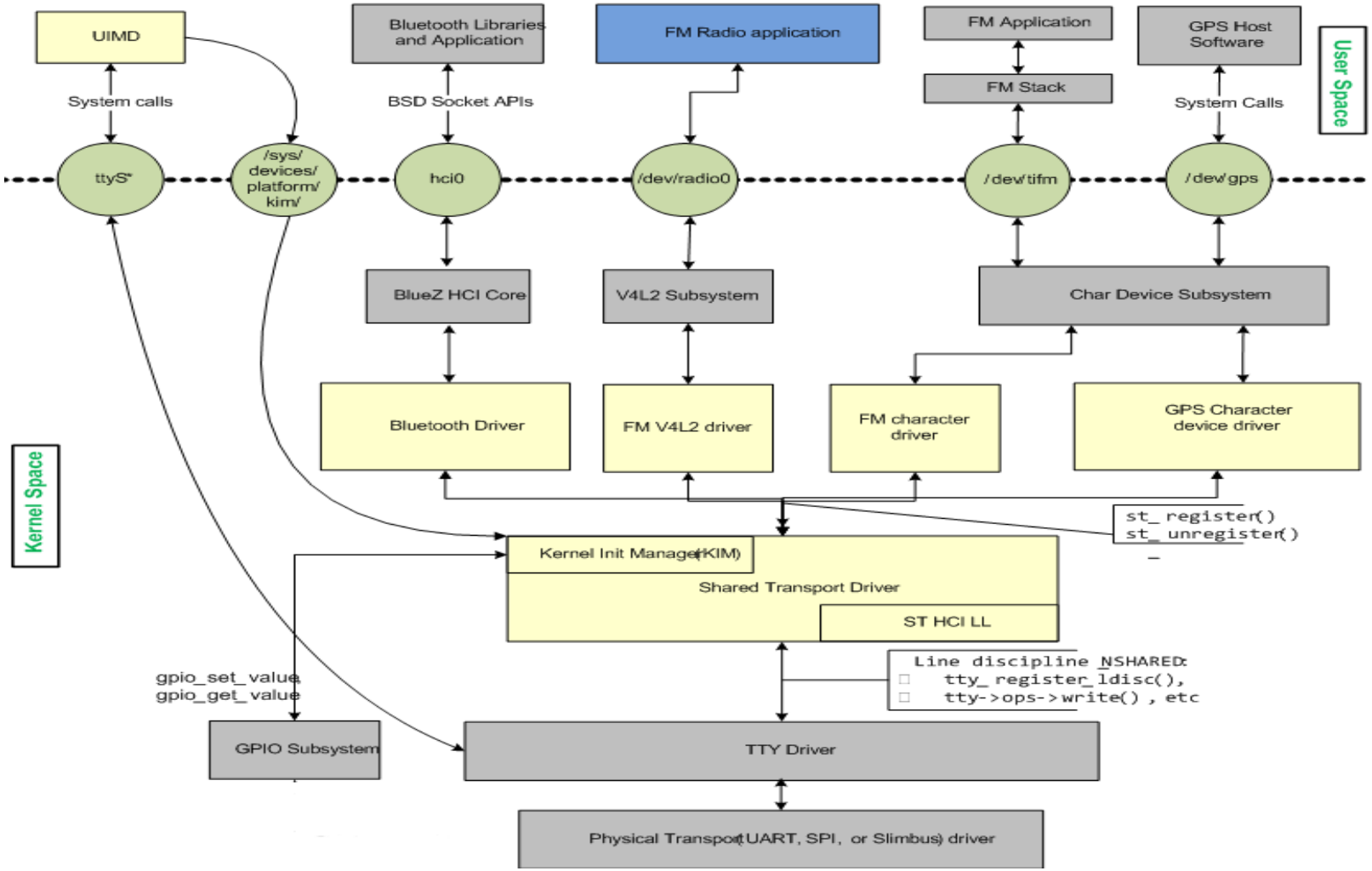
Jelly Bean Device Porting Walkthrough
JB Bluetooth Architecture

- **Blue-Z** has been removed in favor of Broadcom BT stack:
  - Wrongly called **bluedroid** in **external/bluetooth**.
  - **libbluedroid** still exist in **system/bluetooth** but is something else.
  - **libbluedroid_jni** exist in **packages/apps/Bluetooth**.
  - **D-Bus** dependency has been removed (that's a good thing).
  - New stack doesn't have **Blue-Z** profiles level (that's a bad thing).

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough

Bluetooth TI Shared Transport Architecture



COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Bluetooth Driver

```
#define BT_IRQ                  IMX_GPIO_NR(6, 11)
#define BT_EN                   IMX_GPIO_NR(2, 24)

static struct platform_device btwilink_device = {
        .name                   = "btwilink",
        .id                     = -1,
};

static void bt_chip_enable(struct kim_data_s *data) {
        gpio_direction_output(BT_EN, 1);
}

static void bt_chip_disable(struct kim_data_s *data) {
        gpio_direction_output(BT_EN, 0);
}

struct ti_st_plat_data kim_bt_pdata = {
        .nshutdown_gpio         = BT_EN,
        .dev_name               = "/dev/ttymxc0",
        .flow_cntrl             = 1,
        .baud_rate              = 3000000,
        .chip_enable            = bt_chip_enable,
        .chip_disable           = bt_chip_disable,
};
```

Loads up TI Bluetooth driver

BT GPIO enable/disable

Check HW schematics which UART you're plugged on

You NEED RTS/CTS

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Bluetooth Driver

```
static struct platform_device kim_bt_device = {
        .name                   = "kim",
        .id                     = ST_BT,
        .dev = {
                .platform_data  = &kim_bt_pdata,
        },
};

static struct platform_device *bt_devices[] __initdata = {
        &kim_bt_device,
        &btwilink_device,
};

platform_add_devices(bt_devices, ARRAY_SIZE(bt_devices));
platform_device_register(&bt_devices);
```

Loads TI ST (KIM) driver

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Bluetooth UIM

- Check **hardware/ti/wpan/ti_st/uim-sysfs** for details.

```
/* Forming the packet for Change speed command */
cmd.uart_prefix = HCI_COMMAND_PKT;
cmd.hci_hdr.opcode = HCI_HDR_OPCODE;
cmd.hci_hdr.plen = sizeof(unsigned long);
cmd.speed = custom_baud_rate;

/* Writing the change speed command to the UART
 * This will change the UART speed at the controller side */
len = write(dev_fd, &cmd, sizeof(cmd));

/* Read the response for the Change speed command */
read_command_complete(dev_fd, HCI_HDR_OPCODE);

/* Set the actual custom baud rate at the host side */
set_custom_baud_rate(custom_baud_rate, flow_ctrl);

/* Call IOCTL to set the line discipline to N_TI_WL */
ldisc = N_TI_WL; // 22
ioctl(dev_fd, TIOCSETD, &ldisc);
```

Remote control of WiLink through UART

Host UART control

Tell Kernel (UIM) we're done with init

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Bluetooth HAL

- Introduced in Jelly Bean.
- See **libhardware/include/hardware/bluetooth.h** for details.

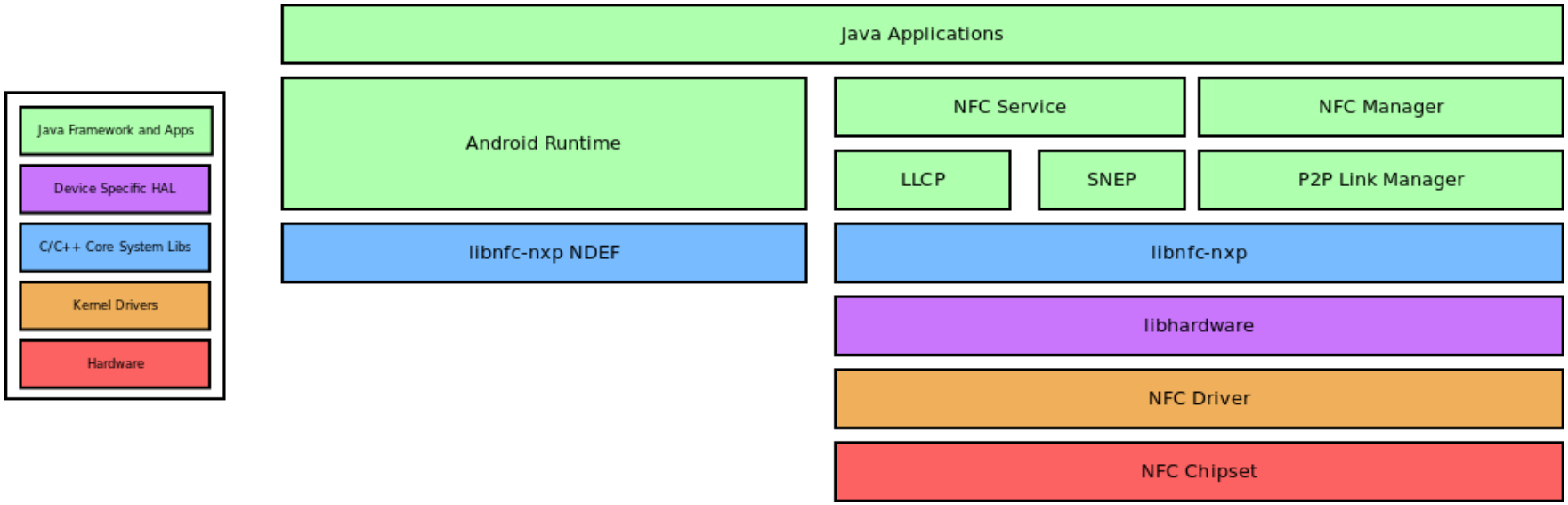
- Implements HW vendor specific interface to device:
  - Enable / Disable HW
  - Get / Set device properties
  - Start / Cancel discovery
  - …
- Implementation takes place in **packages/apps/Bluetooth/**
  - **jni/** provides **libbluetooth_jni** the interface with HAL.
  - **src/** implements Java Bluetooth service, using **libbluetooth_jni** and Broadcom's **bluetooth.default** library.

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Miscellaneous Devices

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
NFC Architecture



- **Supported NFC Features:**
  - NDEF (NFC Data Exchange Format) support.
  - Non-NDEF (Advanced NFC) support.
  - Passive NFC tags reading support.
  - Active NFC P2P connections exchanges (« Android Beam »).
  - NFC tags parsing and intent sending (to be catched by best suited app)

- **Supported NFC data types:**
  - Absolute URI
  - External URN type (for proprietary applications)
  - MIME types.

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
NFC Driver

```
#define NFC_IRQ                 IMX_GPIO_NR(7, 1)
#define NFC_POWER_ENABLE        IMX_GPIO_NR(7, 0)

static void pn544_nfc_enable(int fw) {
        gpio_request(NFC_POWER_ENABLE, "nfc_power");
        gpio_direction_output(NFC_POWER_ENABLE, 1);
}

static void pn544_nfc_disable(void) {
        gpio_request(NFC_POWER_ENABLE, "nfc_power");
        gpio_direction_output(NFC_POWER_ENABLE, 0);
}

static struct pn544_nfc_platform_data pn544_pdata = {
        .enable = pn544_nfc_enable,
        .disable = pn544_nfc_disable,
};

static struct i2c_board_info mxc_i2c0[] __initdata = {
        {
                I2C_BOARD_INFO("pn544", 0x28),
                .platform_data          = &pn544_pdata,
                .irq                    = gpio_to_irq(NFC_IRQ),
        },
};
```

Turn GPIO on

Turn GPIO off

Match I2C address with driver's name

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
NFC HAL

- Implement HAL in **hardware/libhardware/include/hardware/nfc.h**

- **Status in ICS:**
  - Stack introduction
  - Very *NXP pn544* specific
  - Supports I2C, UART and USB connectivity.

- **Status in JB:**
  - Add support for generic NCI-based NFC controllers
  - Implements NCI write calls.
  - Provides NFC stack / driver commands passing with callback support.

COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Power Management

- Initializes Power Management actions.
- Implements **libhardware/include/hardware/power.h**.

```
/*
 * (*setInteractive)() performs power management actions upon the system entering interactive state
 * (that is, the system is awake and ready for interaction, often with UI devices such as display and
 * touchscreen enabled) or non-interactive state (the system appears asleep, display usually turned
 * off).  The non-interactive state is usually entered after a period of inactivity, in order to
 * conserve battery power during such inactive periods.
 *
 * Typical actions are to turn on or off devices and adjust cpufreq parameters. This function may also
 * call the appropriate interfaces to allow the kernel to suspend the system to low-power sleep state
 * when entering non-interactive state, and to disallow low-power suspend when the system is in
 * interactive state.  When low-power suspend state is allowed, the kernel may suspend the system
 * whenever no wakelocks are held.
 *
 */
void (*setInteractive)(struct power_module *module, int on);
```


COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
Power Management

```
/*
 * (*powerHint) is called to pass hints on power requirements, which may result in adjustment of
 * power/performance parameters of the cpufreq governor and other controls. The possible hints are:
 *
 * POWER_HINT_VSYNC
 *     Foreground app has started or stopped requesting a VSYNC pulse from SurfaceFlinger. If the app
 *     has started requesting VSYNC then CPU and GPU load is expected soon, and it may be appropriate
 *     to raise speeds of CPU, memory bus, etc.
 *
 * POWER_HINT_INTERACTION
 *     User is interacting with the device, for example, touchscreen events are incoming.
 *     CPU and GPU load may be expected soon, and it may be appropriate to raise speeds of CPU,
 *     memory bus, etc.
 */
void (*powerHint)(struct power_module *module, power_hint_t hint, void *data);
```


COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Jelly Bean Device Porting Walkthrough
No-Battery Trick

- Trick for Android to believe it's running on power supply
- Add new system setting in **device.mk**:
  - PRODUCT_PROPERTY_OVERRIDES += hw.nobattery=true
  - Hack on **frameworks/base/services/java/com/android/server/BatteryService.java**

```
String hwNoBatteryStr = SystemProperties.get("hw.nobattery");
boolean hwNoBattery = Boolean.parseBoolean(hwNoBatteryStr);

if (!hwNoBattery)
        mPowerSupplyObserver.startObserving("SUBSYSTEM=power_supply");

private void stubUpdate() {
        // Hardcode values. We could read them from properties
        mAcOnline                       = true;
        mUsbOnline                      = false;
        mBatteryPresent                 = true;
        mBatteryLevel                   = 100;
        mBatteryVoltage                 = 4700;
        mBatteryTemperature             = 80;
        mBatteryStatus                  = BatteryManager.BATTERY_STATUS_FULL;
        mPlugType                       = BatteryManager.BATTERY_PLUGGED_AC;
}
```

```
private synchronized final void update() {
        if (hwNoBattery)
                stubUpdate();
        else {
                native_update();
                 processValues();
         }
}
```


COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

Thank You



COPYRIGHT © 2013 ALCATEL-LUCENT. ALL RIGHTS RESERVED.

AT THE SPEED OF IDEAS™

Alcatel·Lucent

www.alcatel-lucent.com