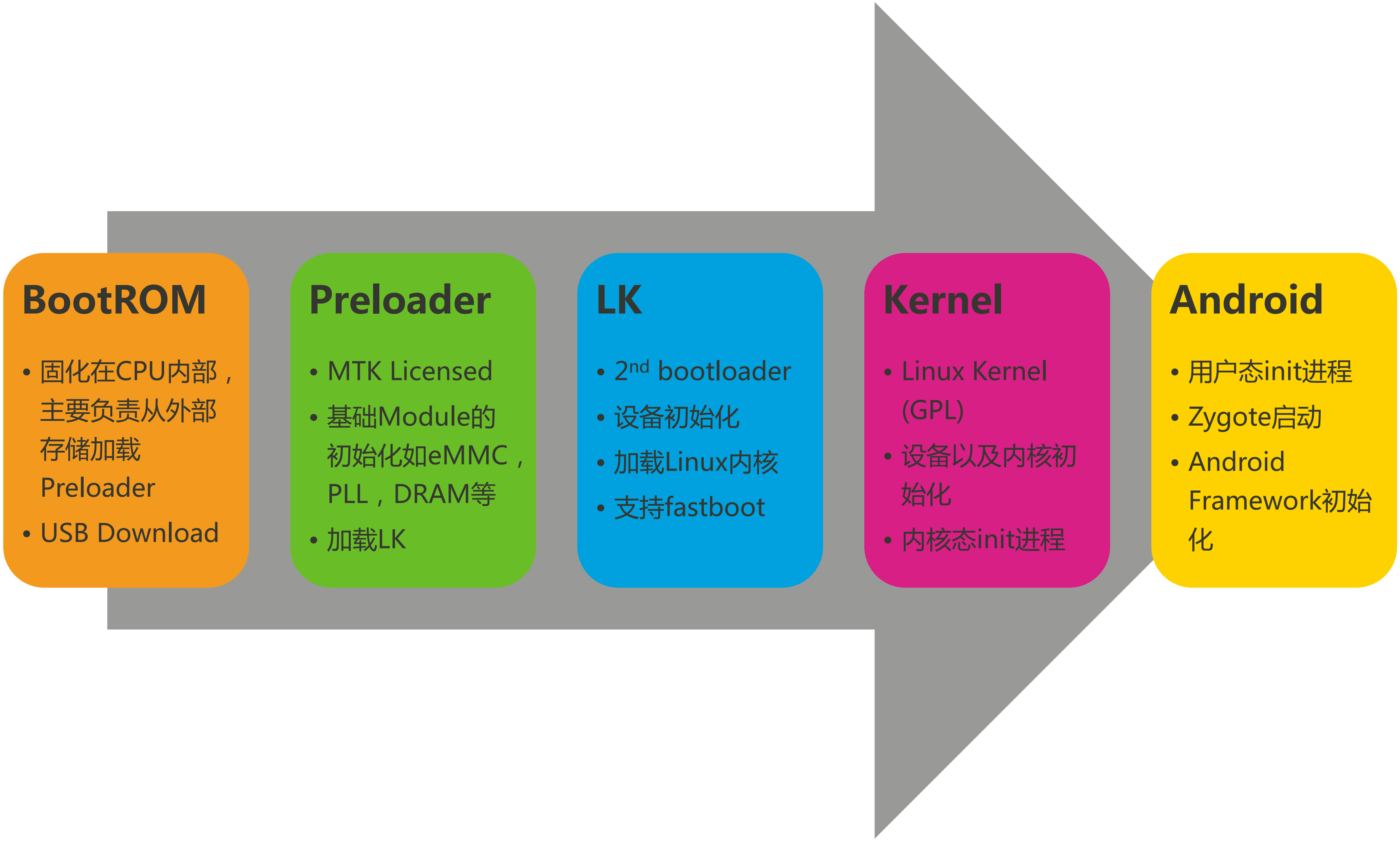
CONFIDENTIAL B

Bootup

启动流程



BootROM

- 固化在CPU内部，主要负责从外部存储加载Preloader
- USB Download

Preloader

- MTK Licensed
- 基础Module的初始化如eMMC，PLL，DRAM等
- 加载LK

LK

- 2nd bootloader
- 设备初始化
- 加载Linux内核
- 支持fastboot

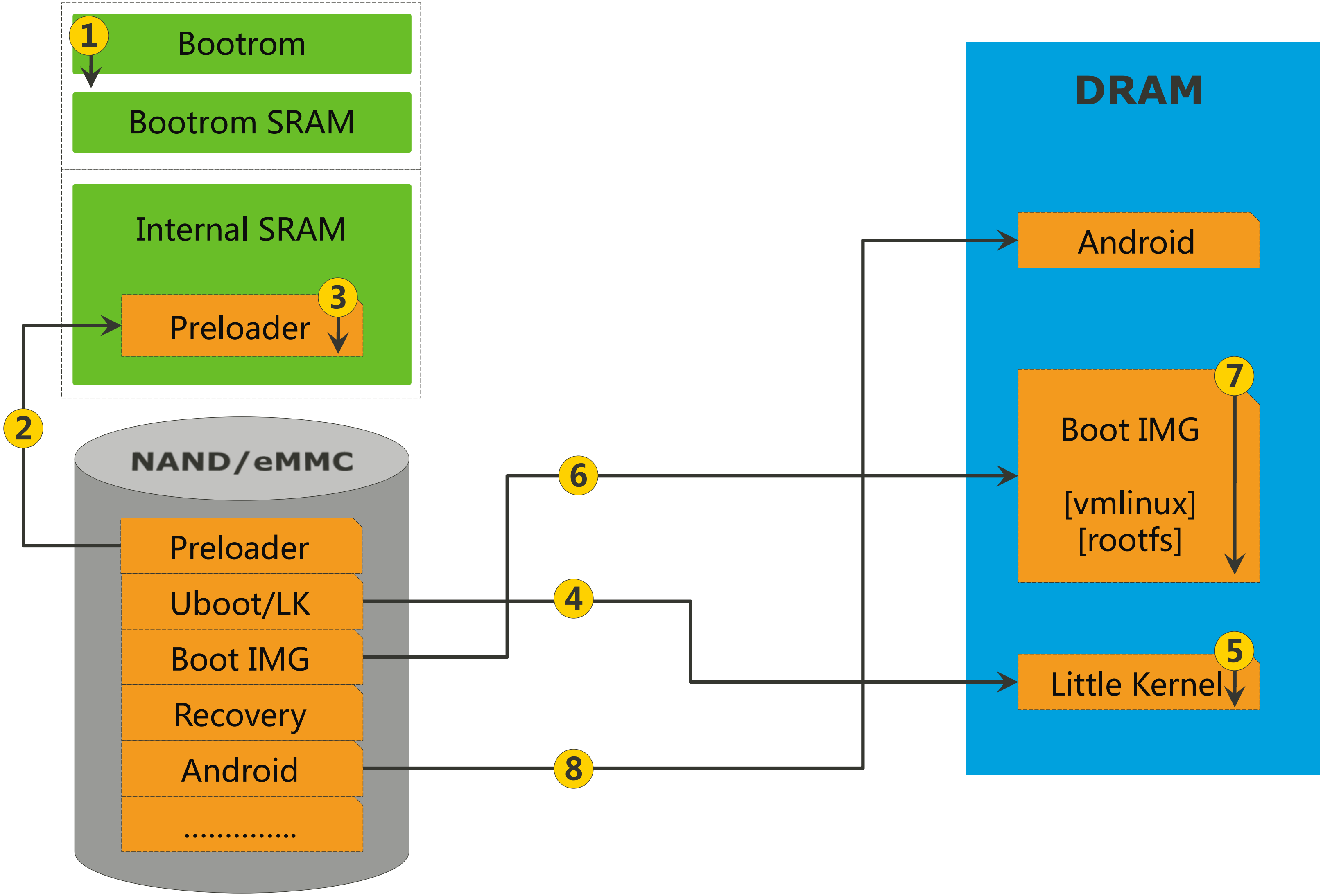
Kernel

- Linux Kernel (GPL)
- 设备以及内核初始化
- 内核态init进程

Android

- 用户态init进程
- Zygote启动
- Android Framework初始化

IMAGE 加载过程


1
Bootrom
Bootrom SRAM
Internal SRAM
Preloader
3
2
NAND/eMMC
Preloader
Uboot/LK
Boot IMG
Recovery
Android
....
DRAM
Android
6
7
Boot IMG
[vmlinux]
[rootfs]
4
5
Little Kernel
8

CONFIDENTIAL B

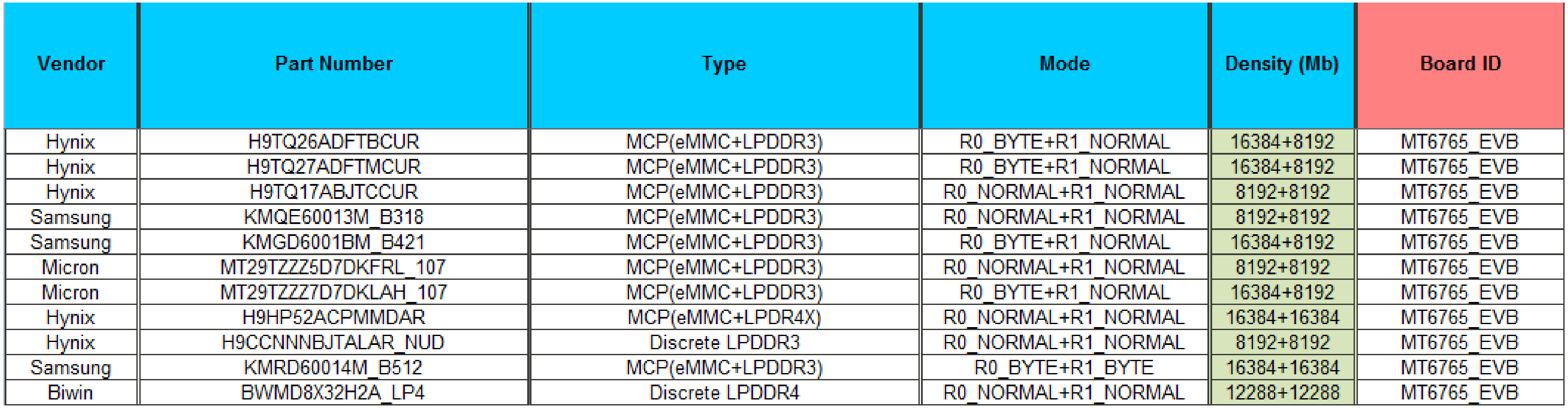
EMI

EMI Introduction

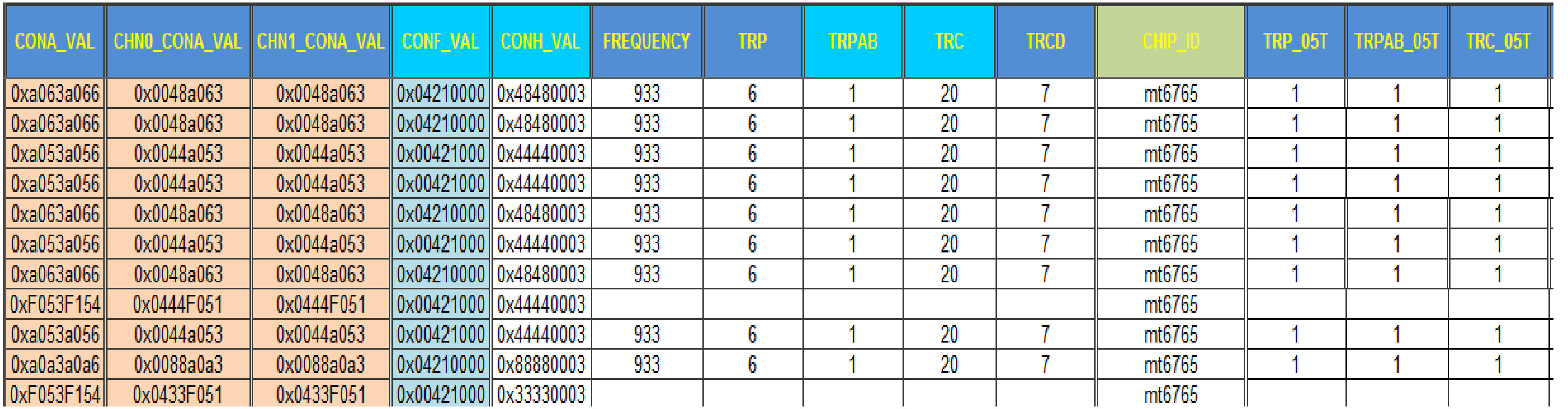
- Introduction
  - Perl script is used to auto generate source file and header file of DDR initialization.
    - Location: alps/vendor/mediatek/proprietary/bootable/bootloader/**preloader**/tools/emigen/${platform}/emigen.pl
  - Memory DB file
    - Location: alps/vendor/mediatek/proprietary/bootable/bootloader/**preloader**/tools/emigen/${platform}/ MemoryDeviceList_MTxxxx.xls
    - Note:

      Please confirm if the memory[to use] has been verified in MTK Online->QVL(New). If verified, get information from MTK Online->QVL(New) and merge it in the last line of this file. If not, submit eService for memory verification.

MemoryDeviceList_MTxxxx.xls (e.g)

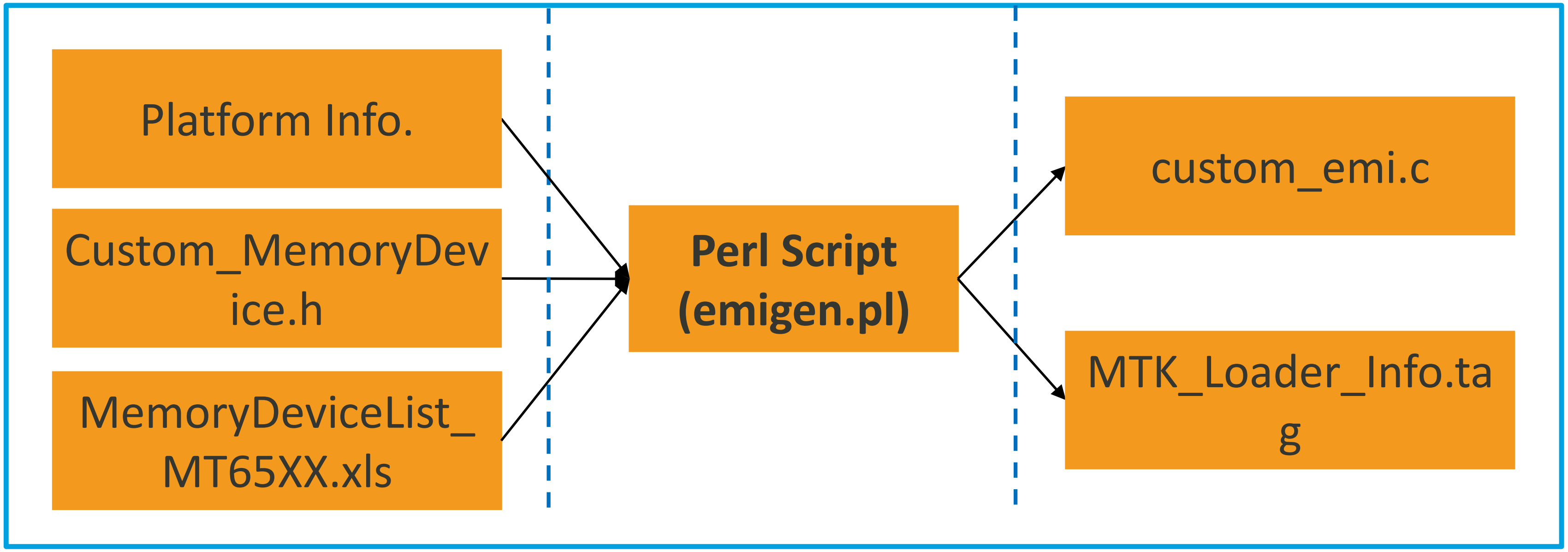


Vendor	Part Number	Type	Mode	Density (Mb)	Board ID
Hynix	H9TQ26ADFTBCUR	MCP(eMMC+LPDDR3)	R0_BYTE+R1_NORMAL	16384+8192	MT6765_EVB
Hynix	H9TQ27ADFTMCUR	MCP(eMMC+LPDDR3)	R0_BYTE+R1_NORMAL	16384+8192	MT6765_EVB
Hynix	H9TQ17ABJTCCUR	MCP(eMMC+LPDDR3)	R0_NORMAL+R1_NORMAL	8192+8192	MT6765_EVB
Samsung	KMQE60013M_B318	MCP(eMMC+LPDDR3)	R0_NORMAL+R1_NORMAL	8192+8192	MT6765_EVB
Samsung	KMGD6001BM_B421	MCP(eMMC+LPDDR3)	R0_BYTE+R1_NORMAL	16384+8192	MT6765_EVB
Micron	MT29TZZZ5D7DKFRL_107	MCP(eMMC+LPDDR3)	R0_NORMAL+R1_NORMAL	8192+8192	MT6765_EVB
Micron	MT29TZZZ7D7DKLAH_107	MCP(eMMC+LPDDR3)	R0_BYTE+R1_NORMAL	16384+8192	MT6765_EVB
Hynix	H9HP52ACPMMDAR	MCP(eMMC+LPDR4X)	R0_NORMAL+R1_NORMAL	16384+16384	MT6765_EVB
Hynix	H9CCNNNBJTALAR_NUD	Discrete LPDDR3	R0_NORMAL+R1_NORMAL	8192+8192	MT6765_EVB
Samsung	KMRD60014M_B512	MCP(eMMC+LPDDR3)	R0_BYTE+R1_BYTE	16384+16384	MT6765_EVB
Biwin	BWMD8X32H2A_LP4	Discrete LPDDR4	R0_NORMAL+R1_NORMAL	12288+12288	MT6765_EVB



CONA_VAL	CHN0_CONA_VAL	CHN1_CONA_VAL	CONF_VAL	CONH_VAL	FREQUENCY	TRP	TRPAB	TRC	TRCD	CHIP_ID	TRP_05T	TRPAB_05T	TRC_05T
0xa063a066	0x0048a063	0x0048a063	0x04210000	0x48480003	933	6	1	20	7	mt6765	1	1	1
0xa063a066	0x0048a063	0x0048a063	0x04210000	0x48480003	933	6	1	20	7	mt6765	1	1	1
0xa053a056	0x0044a053	0x0044a053	0x00421000	0x44440003	933	6	1	20	7	mt6765	1	1	1
0xa053a056	0x0044a053	0x0044a053	0x00421000	0x44440003	933	6	1	20	7	mt6765	1	1	1
0xa063a066	0x0048a063	0x0048a063	0x04210000	0x48480003	933	6	1	20	7	mt6765	1	1	1
0xa053a056	0x0044a053	0x0044a053	0x00421000	0x44440003	933	6	1	20	7	mt6765	1	1	1
0xa063a066	0x0048a063	0x0048a063	0x04210000	0x48480003	933	6	1	20	7	mt6765	1	1	1
0xF053F154	0x0444F051	0x0444F051	0x00421000	0x44440003						mt6765			
0xa053a056	0x0044a053	0x0044a053	0x00421000	0x44440003	933	6	1	20	7	mt6765	1	1	1
0xa0a3a0a6	0x0088a0a3	0x0088a0a3	0x04210000	0x88880003	933	6	1	20	7	mt6765	1	1	1
0xF053F154	0x0433F051	0x0433F051	0x00421000	0x33330003						mt6765			

EMIGEN Flow



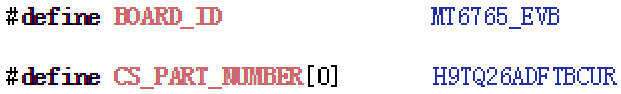
Input Parameters **Processing** **Output Files**

EMI Customization

- Customization Files

File	Description
alps/vendor/mediatek/proprietary/bootable/bootloader/**preloader**/custom/${PROJECT}/inc/custom_MemoryDevice.h	
custom_MemoryDevice.h	The customization file for EMI setting

- How to customize
  - Config memory in custom_MemoryDevice.h
    - E.g, MT6765 project supports two MCPs



```
#define BOARD_ID                    MT6765_EVB

#define CS_PART_NUMBER[0]           H9TQ26ADFTBCUR
```

  - Make sure the memory is verified(refer to page **EMI Introduction**)
  - Rebuild preloader when memory is config/changed
    - **Build command**: make -j24 pl 2>&1 | tee pl.log
- **Remind: Must run ETT procedure before EMI customization**
  - **Reference**: MTxxxx ETT & stress test reference V0.1.pdf
    (File Path: MTK Online -> QVL(New) -> memory -> MTK_MVG_TOOLs.rar -> MT6xxx_ETT_and_stress_test_reference)

Combo Memory Feature

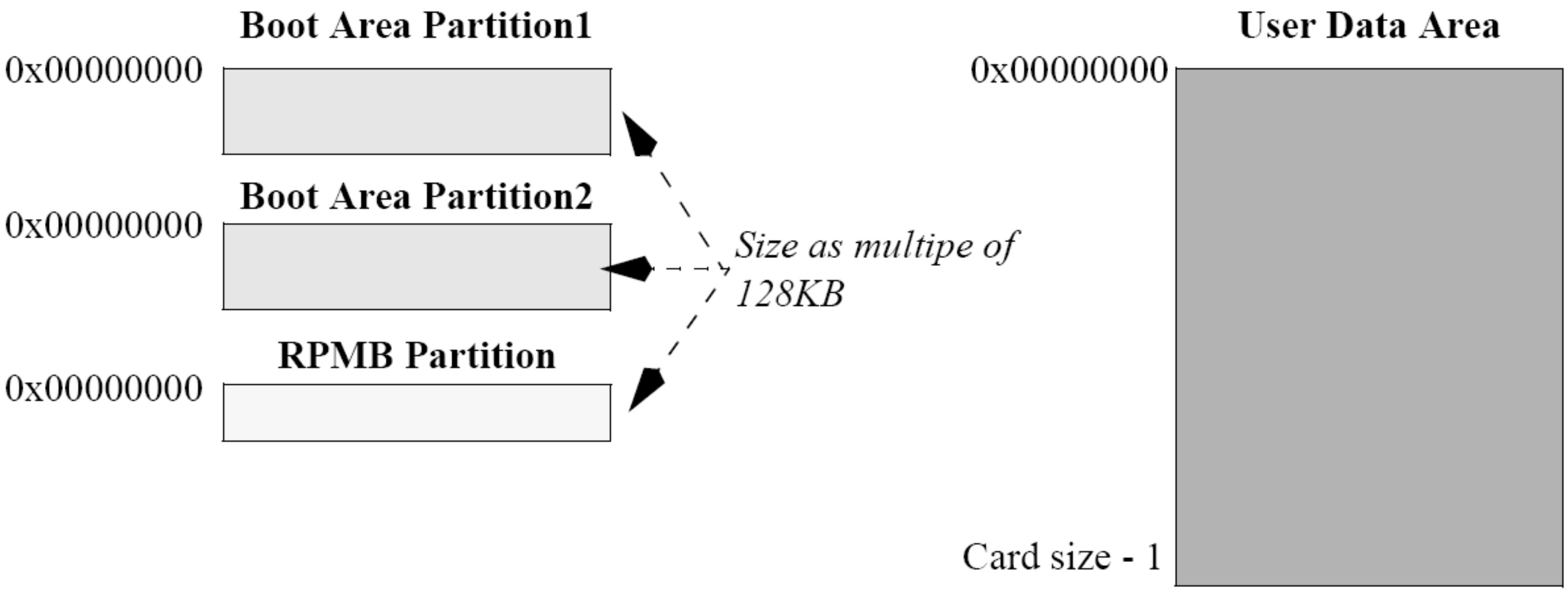
- Collect EMI settings of specified memory into codebase in compile time.
- Select correct EMI settings of one memory in runtime.
- User can change memory without re-compiling/downloading pre-loader image if required MCP devices have already been specified in configure files.

CONFIDENTIAL B

eMMC

eMMC Partition Management

- 4 Default Areas of Memory Device
  - 2 x Boot Area Partitions for Booting
  - 1 x Replay Protected Memory Block Area Partition
  - 1 x User Data Area



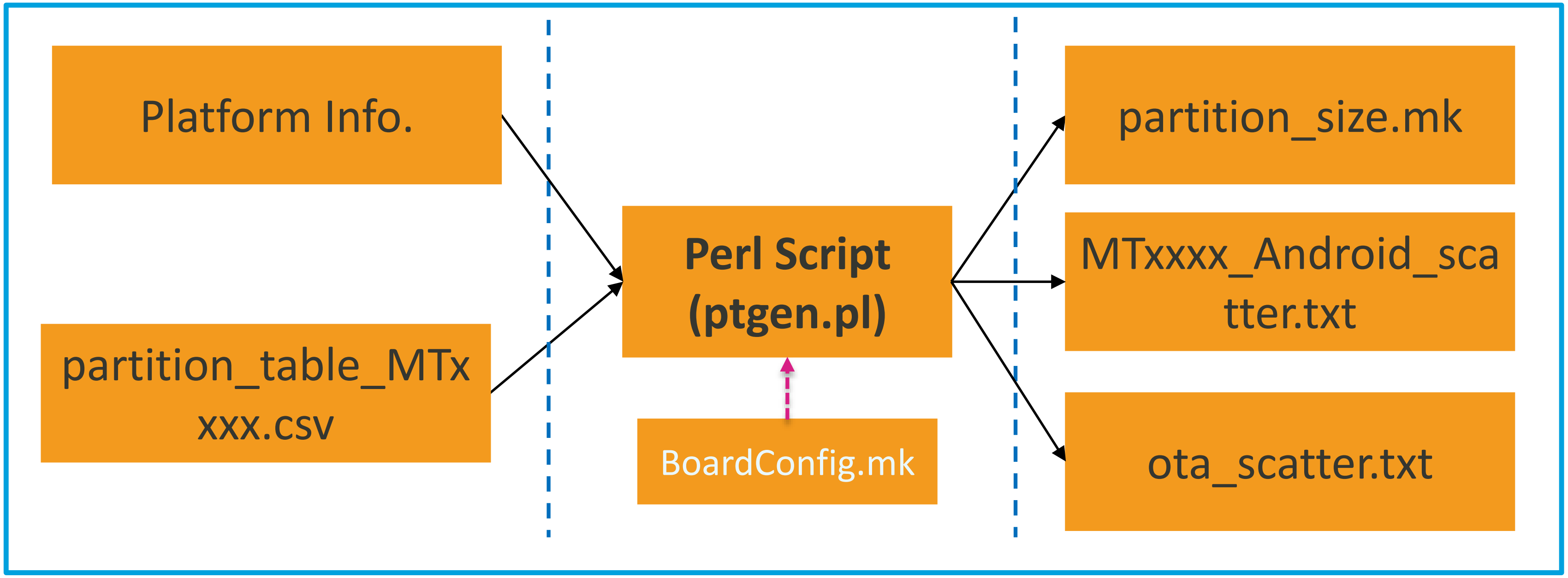
Partition Table & PTGEN

- Partition Table
  - There is **ONLY** one excel file in the codebase, which pre-config the partition layout **by platform**.
  - **Location**

    alps/device/mediatek/build/build/tools/ptgen/${platform}/partition_table_MTxxxx_emmc.csv
- PTGEN
  - Perl script is used to parse partition table, and generate source and header files including partition layout information.
  - **Location**

    alps/device/mediatek/build/build/tools/ptgen/${platform}/ptgen.pl

PTGEN Flow



Platform Info.

partition_table_MTxxxx.csv

Perl Script (ptgen.pl)

BoardConfig.mk

partition_size.mk

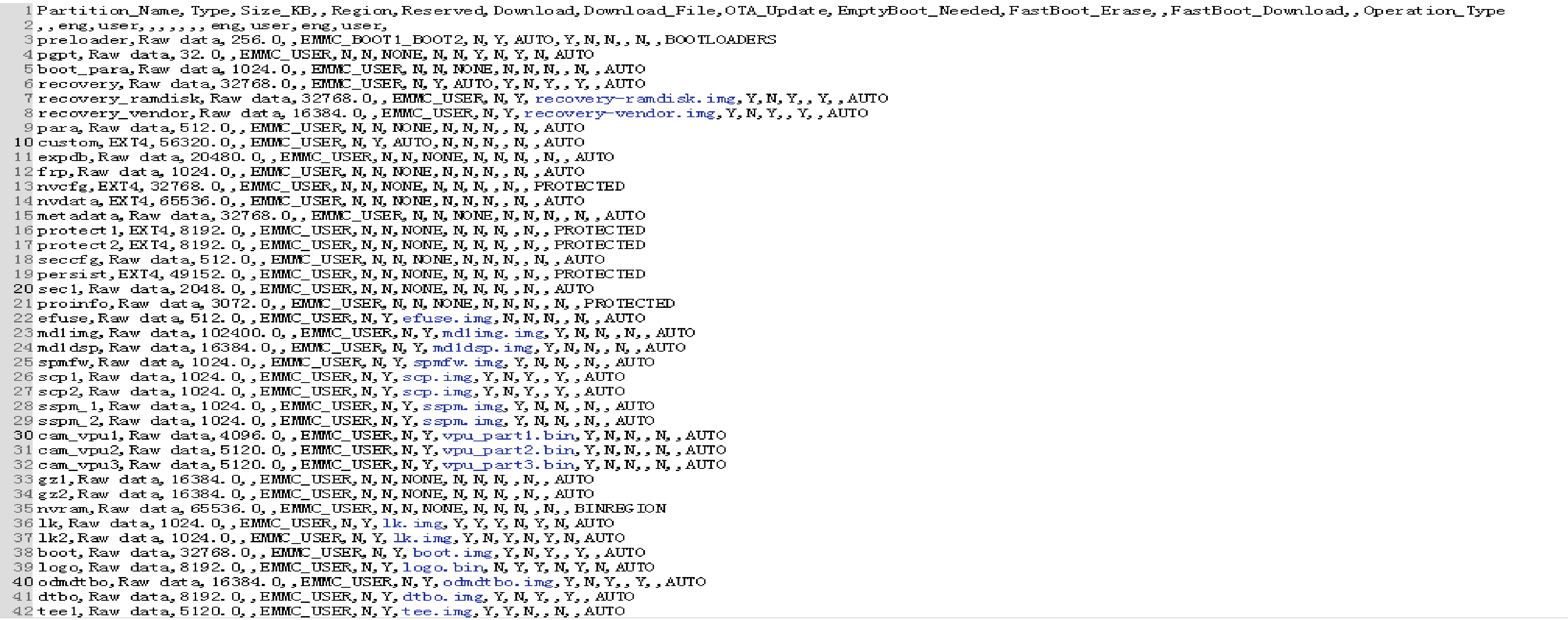
MTxxxx_Android_scatter.txt

ota_scatter.txt

Input Parameters **Processing** **Output Files**

Partition Layout Customization - for Platform

- How to customize for platform
  - Modify colume “**Size_KB**” in **partition_table_MTxxxx.csv**



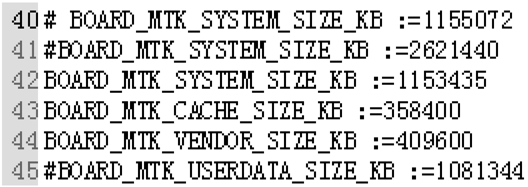
```
Partition_Name,Type,Size_KB,,Region,Reserved,Download,Download_File,OTA_Update,EmptyBoot_Needed,FastBoot_Erase,,FastBoot_Download,,Operation_Type
,,eng,user,,,,,,eng,user,eng,user,
preloader,Raw data,256.0,,EMMC_BOOT1_BOOT2,N,Y,AUTO,Y,N,N,,N,,BOOTLOADERS
pgpt,Raw data,32.0,,EMMC_USER,N,N,NONE,N,N,Y,N,Y,N,AUTO
boot_para,Raw data,1024.0,,EMMC_USER,N,N,NONE,N,N,N,,N,,AUTO
recovery,Raw data,32768.0,,EMMC_USER,N,Y,AUTO,Y,N,Y,,Y,,AUTO
recovery_ramdisk,Raw data,32768.0,,EMMC_USER,N,Y,recovery-ramdisk.img,Y,N,Y,,Y,,AUTO
recovery_vendor,Raw data,16384.0,,EMMC_USER,N,Y,recovery-vendor.img,Y,N,Y,,Y,,AUTO
para,Raw data,512.0,,EMMC_USER,N,N,NONE,N,N,N,,N,,AUTO
custom,EXT4,56320.0,,EMMC_USER,N,Y,AUTO,N,N,N,,N,,AUTO
expdb,Raw data,20480.0,,EMMC_USER,N,N,NONE,N,N,N,,N,,AUTO
frp,Raw data,1024.0,,EMMC_USER,N,N,NONE,N,N,N,,N,,AUTO
nvcfg,EXT4,32768.0,,EMMC_USER,N,N,NONE,N,N,N,,N,,PROTECTED
nvdata,EXT4,65536.0,,EMMC_USER,N,N,NONE,N,N,N,,N,,AUTO
metadata,Raw data,32768.0,,EMMC_USER,N,N,NONE,N,N,N,,N,,AUTO
protect1,EXT4,8192.0,,EMMC_USER,N,N,NONE,N,N,N,,N,,PROTECTED
protect2,EXT4,8192.0,,EMMC_USER,N,N,NONE,N,N,N,,N,,PROTECTED
seccfg,Raw data,512.0,,EMMC_USER,N,N,NONE,N,N,N,,N,,AUTO
persist,EXT4,49152.0,,EMMC_USER,N,N,NONE,N,N,N,,N,,PROTECTED
sec1,Raw data,2048.0,,EMMC_USER,N,N,NONE,N,N,N,,N,,AUTO
proinfo,Raw data,3072.0,,EMMC_USER,N,N,NONE,N,N,N,,N,,PROTECTED
efuse,Raw data,512.0,,EMMC_USER,N,Y,efuse.img,N,N,N,,N,,AUTO
md1img,Raw data,102400.0,,EMMC_USER,N,Y,md1img.img,Y,N,N,,N,,AUTO
md1dsp,Raw data,16384.0,,EMMC_USER,N,Y,md1dsp.img,Y,N,N,,N,,AUTO
spmfw,Raw data,1024.0,,EMMC_USER,N,Y,spmfw.img,Y,N,N,,N,,AUTO
scp1,Raw data,1024.0,,EMMC_USER,N,Y,scp.img,Y,N,Y,,Y,,AUTO
scp2,Raw data,1024.0,,EMMC_USER,N,Y,scp.img,Y,N,Y,,Y,,AUTO
sspm_1,Raw data,1024.0,,EMMC_USER,N,Y,sspm.img,Y,N,N,,N,,AUTO
sspm_2,Raw data,1024.0,,EMMC_USER,N,Y,sspm.img,Y,N,N,,N,,AUTO
cam_vpu1,Raw data,4096.0,,EMMC_USER,N,Y,vpu_part1.bin,Y,N,N,,N,,AUTO
cam_vpu2,Raw data,5120.0,,EMMC_USER,N,Y,vpu_part2.bin,Y,N,N,,N,,AUTO
cam_vpu3,Raw data,5120.0,,EMMC_USER,N,Y,vpu_part3.bin,Y,N,N,,N,,AUTO
gz1,Raw data,16384.0,,EMMC_USER,N,N,NONE,N,N,N,,N,,AUTO
gz2,Raw data,16384.0,,EMMC_USER,N,N,NONE,N,N,N,,N,,AUTO
nvram,Raw data,65536.0,,EMMC_USER,N,N,NONE,N,N,N,,N,,BINREGION
lk,Raw data,1024.0,,EMMC_USER,N,Y,lk.img,Y,Y,Y,N,Y,N,AUTO
lk2,Raw data,1024.0,,EMMC_USER,N,Y,lk.img,Y,N,Y,N,Y,N,AUTO
boot,Raw data,32768.0,,EMMC_USER,N,Y,boot.img,Y,N,Y,,Y,,AUTO
logo,Raw data,8192.0,,EMMC_USER,N,Y,logo.bin,N,Y,Y,N,Y,N,AUTO
odmdtbo,Raw data,16384.0,,EMMC_USER,N,Y,odmdtbo.img,Y,N,Y,,Y,,AUTO
dtbo,Raw data,8192.0,,EMMC_USER,N,Y,dtbo.img,Y,N,Y,,Y,,AUTO
tee1,Raw data,5120.0,,EMMC_USER,N,Y,tee.img,Y,Y,N,,N,,AUTO
```

Partition Layout Customization - for Project

- How to customize for project
  - Modify **BoardConfig.mk**
    - Base Project

      alps/device/${COMPANY}/${BASE_PROJECT}/BoardConfig.mk
    - Flavor Project

      alps/device/${COMPANY}/${BASE_PROJECT}_{$FLAVOR}/BoardConfig.mk
  - Example modification in **BoardConfig.mk**

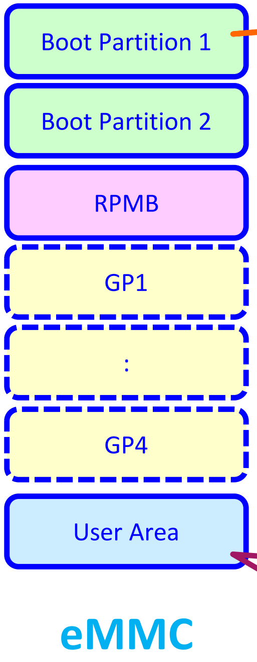


```
# BOARD_MTK_SYSTEM_SIZE_KB :=1155072
#BOARD_MTK_SYSTEM_SIZE_KB :=2621440
BOARD_MTK_SYSTEM_SIZE_KB :=1153435
BOARD_MTK_CACHE_SIZE_KB :=358400
BOARD_MTK_VENDOR_SIZE_KB :=409600
#BOARD_MTK_USERDATA_SIZE_KB :=1081344
```

- How to make modification valid
  - Recommend
    - **Build command**: make -j24 2>&1 | tee build.log

eMMC Device Layout

- Partition Mapping



Partition Table

Partition_Name	Type
preloader	Raw data
pgpt	Raw data
proinfo	Raw data
nvram	Raw data
protect1	EXT4
protect2	EXT4
lk	Raw data
para	Raw data
boot	Raw data
recovery	Raw data
logo	Raw data
expdb	Raw data
seccfg	Raw data
oemkeystore	Raw data
secro	EXT4
keystore	Raw data
tee1	Raw data
tee2	Raw data
persist	EXT4
custom	EXT4
frp	Raw data
nvdata	EXT4
metadata	Raw data
system	EXT4
cache	EXT4
userdata	EXT4
intsd	FAT
otp	Raw data
flashinfo	Raw data
sgpt	Raw data

Software Package Download

- **Download Agent**
  - The agent on target to perform the download procedure upon tool request
- **Scatter File**
  - Location：**alps/out/target/product/$PROJECT/$PLATFORM_Android_scatter.txt**
  - Describe the start address of each partition to download
  - The PLATFORM name is embedded into scatter file name, and tool will check if platform matches devices while handshake

CONFIDENTIAL B

LCM

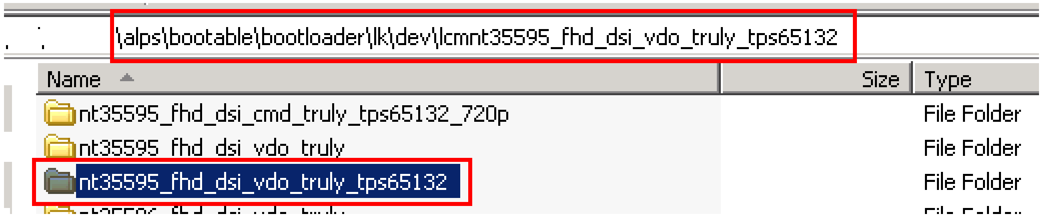
P Display Driver Location

Common	<kernel>/driver/misc/mediatek/video/common
	mtkfb_fence.h, font_8x16.c, mtkfb_dummy.c
	<kernel>/driver/misc/mediatek/video/include
	disp_session.h, disp_svp.h, mtkfb.h, mtkfb_info.h, mtkfb_vsync.h
Dispsys	<kernel>/driver/misc/mediatek/video/<platform>/dispsys
	ddp_ovl.c/h, ddp_rdma.c/h, ddp_reg.h, ddp_color_format.c/h, ddp_debug.c/h, ddp_dpi.c/h, ddp_drv.c/h, ddp_dsi.c/h, ddp_dump.c/h, ddp_hal.h, ddp_info,c/h, ddp_irq.c/h, ddp_irq.h, ddp_log.h, ddp_manager.c/h, ddp_matrix_para.h, ddp_met.c/h, ddp_mmp.c/h, ddp_path.c/h, ddp_wdma_ex.c/h, disp_event.h, display_recorder.c/h
Videox	<kernel>/driver/misc/mediatek/video/<platform>/videox
	disp_drv_platform.h, debug.c/h, disp_assert_layer.c, disp_assert_layer_priv.h, disp_drv_ddp.h, disp_drv_log.h, disp_dts_gpio.c/h, disp_helper.c/h, disp_lcm.c/h, disp_utils.c/h, fbconfig_kdebug.c, mtk_disp_mgr.c/h, mtkfb_console.c/h, mtkfb_fence.c/h, mtk_ovl.c/h, primary_display.c/h,mtkfb.c
LCM	<kernel>/driver/misc/mediatek/lcm
	lcm_common.c,lcm_gpio.c,lcm_i2c.c,lcm_pmic.c,mt65xx_lcm_list.c/h

LK LCM CONFIGURATION

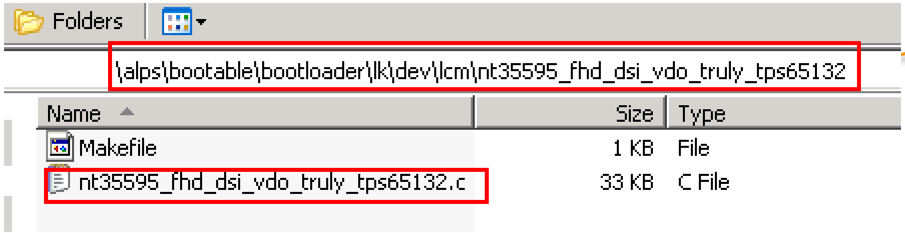
LK LCM Configuration (1/5)

- Step 1: Add your <lcm driver>
  - Add your <lcm driver> into the following path:
    - alps\vendor\mediatek\proprietary\bootable\bootloader\lk\dev\lcm
  - Take <nt35595_fhd_dsi_vdo_truly_tps65132> for example:



\alps\bootable\bootloader\lk\dev\lcmnt35595_fhd_dsi_vdo_truly_tps65132

Name	Size	Type
nt35595_fhd_dsi_cmd_truly_tps65132_720p		File Folder
nt35595_fhd_dsi_vdo_truly		File Folder
nt35595_fhd_dsi_vdo_truly_tps65132		File Folder



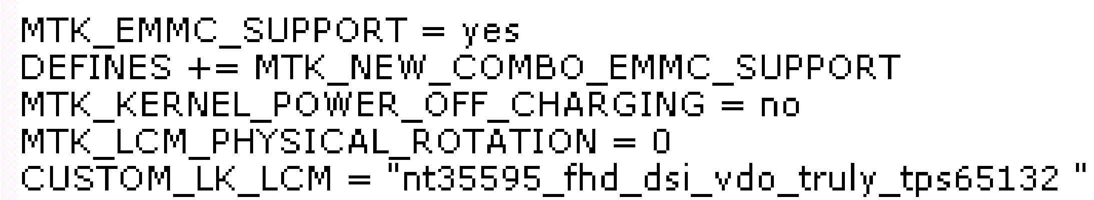
\alps\bootable\bootloader\lk\dev\lcm\nt35595_fhd_dsi_vdo_truly_tps65132

Name	Size	Type
Makefile	1 KB	File
nt35595_fhd_dsi_vdo_truly_tps65132.c	33 KB	C File

Add your <lcm driver>

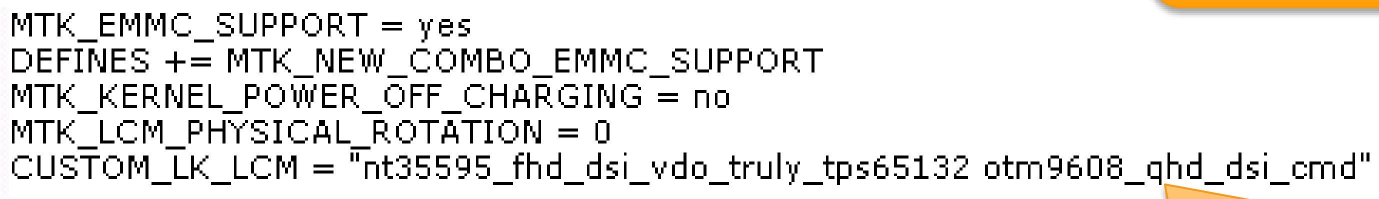
LK LCM Configuration (2/5)

- Step 2: Add your <lcm config> in <project> makefile
  - Add your <lcm confing> in <project>.mk alps\vendor\mediatek\proprietary\bootable\bootloader\lk\project\<project>.mk
  - Take <nt35595_fhd_dsi_vdo_truly_tps65132> for example:



```
MTK_EMMC_SUPPORT = yes
DEFINES += MTK_NEW_COMBO_EMMC_SUPPORT
MTK_KERNEL_POWER_OFF_CHARGING = no
MTK_LCM_PHYSICAL_ROTATION = 0
CUSTOM_LK_LCM = "nt35595_fhd_dsi_vdo_truly_tps65132 "
```

If the case is single LCM, add your <lcm> in CUSTOM_LK_LCM

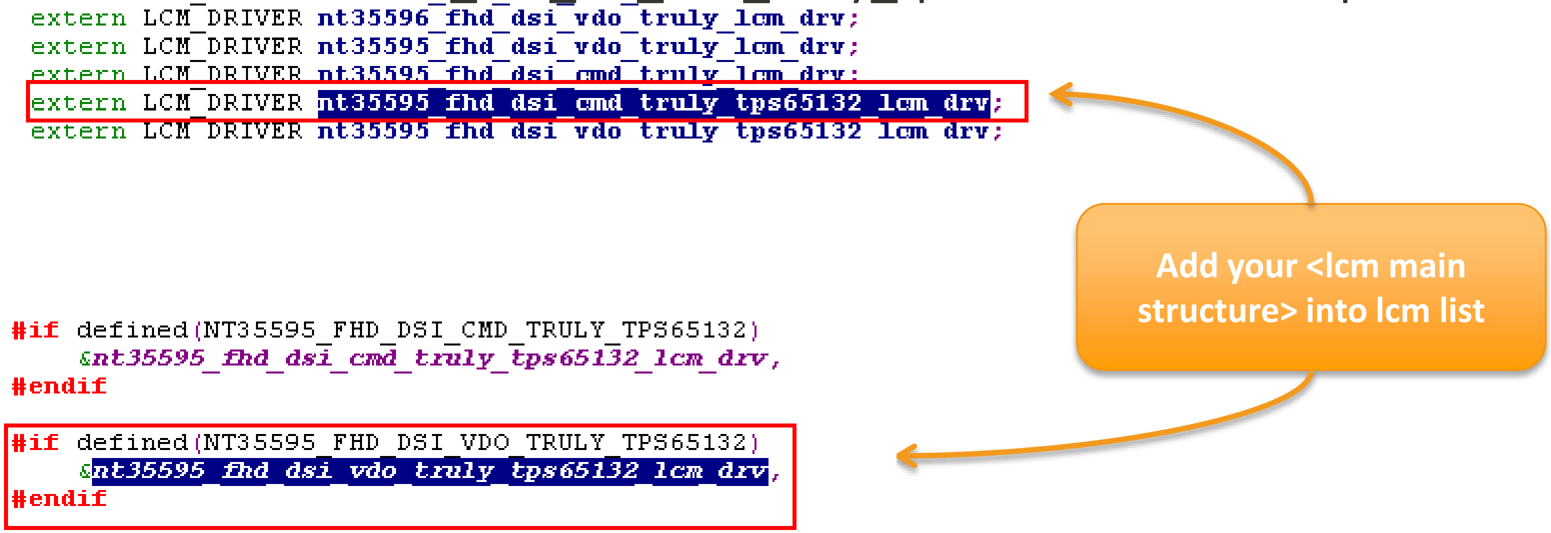


```
MTK_EMMC_SUPPORT = yes
DEFINES += MTK_NEW_COMBO_EMMC_SUPPORT
MTK_KERNEL_POWER_OFF_CHARGING = no
MTK_LCM_PHYSICAL_ROTATION = 0
CUSTOM_LK_LCM = "nt35595_fhd_dsi_vdo_truly_tps65132 otm9608_qhd_dsi_cmd"
```

If the case is multiple LCMs, add your <lcms> in CUSTOM_LK_LCM, and simply separated by space key

LK LCM Configuration (3/5)

- Step 3: Add your <lcm main structure> into lcm list
  - Add your <lcm main structure> into lcm list in alps\vendor\mediatek\proprietary\bootable\bootloader\lk\dev\lcm \mt65xx_lcm_list.c
  - Take < nt35595_fhd_dsi_vdo_truly_tps65132 > for example:



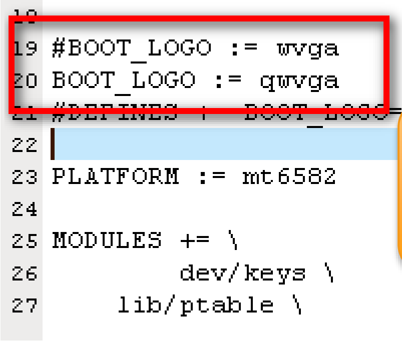
```
extern LCM_DRIVER nt35596_fhd_dsi_vdo_truly_lcm_drv;
extern LCM_DRIVER nt35595_fhd_dsi_vdo_truly_lcm_drv;
extern LCM_DRIVER nt35595_fhd_dsi_cmd_truly_lcm_drv;
extern LCM_DRIVER nt35595_fhd_dsi_cmd_truly_tps65132_lcm_drv;
extern LCM_DRIVER nt35595_fhd_dsi_vdo_truly_tps65132_lcm_drv;
```

```
#if defined(NT35595_FHD_DSI_CMD_TRULY_TPS65132)
    &nt35595_fhd_dsi_cmd_truly_tps65132_lcm_drv,
#endif

#if defined(NT35595_FHD_DSI_VDO_TRULY_TPS65132)
    &nt35595_fhd_dsi_vdo_truly_tps65132_lcm_drv,
#endif
```

Add your <lcm main structure> into lcm list

LK LCM Configuration (4/5)

- Step 4: Switch logo if LCM resolution is different.
  - Modify define marco of BOOT_LOGO in alps\vendor\mediatek\proprietary\bootable\bootloader\lk\project\<project>.mk
  - Take <nt35510_dsi_cmd_6572_qvga> for example:



```

#BOOT_LOGO := wvga
BOOT_LOGO := qwvga
#DEFINES += BOOT_LOGO=

PLATFORM := mt6582

MODULES += \
        dev/keys \
    lib/ptable \
```

Switch to LCM resolution (QVGA)

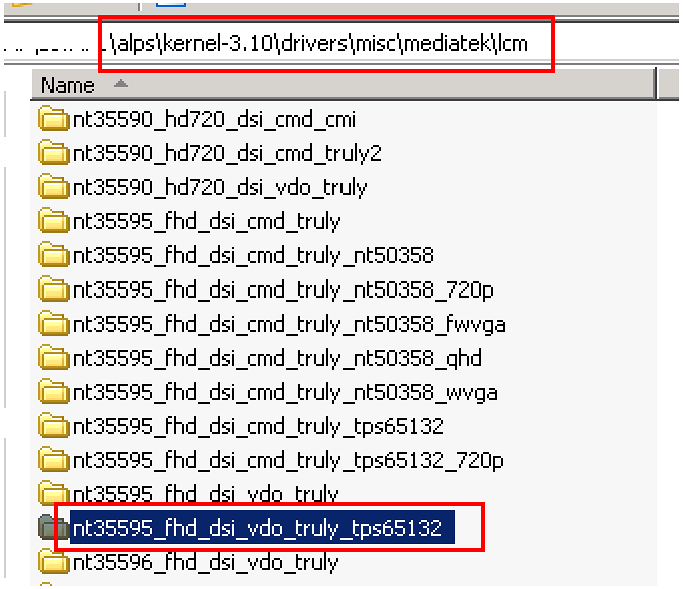
LK LCM Configuration (5/5)

- Step 5: Rebuild lk
  - Rebuild lk and re-download lk.bin.

KERNEL LCM CONFIGURATION

Kernel LCM Configuration (1/5)

- Step 1: Add your <lcm driver>
  - Add your <lcm driver> into the following path:
    - alps\kernel-4.9\drivers\misc\mediatek\lcm\
  - Take <nt35595_fhd_dsi_vdo_truly_tps65132> for example:

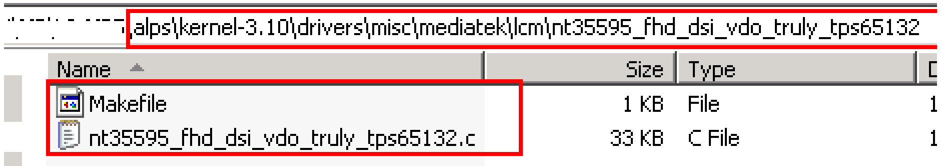


\alps\kernel-3.10\drivers\misc\mediatek\lcm

Name

- nt35590_hd720_dsi_cmd_cmi
- nt35590_hd720_dsi_cmd_truly2
- nt35590_hd720_dsi_vdo_truly
- nt35595_fhd_dsi_cmd_truly
- nt35595_fhd_dsi_cmd_truly_nt50358
- nt35595_fhd_dsi_cmd_truly_nt50358_720p
- nt35595_fhd_dsi_cmd_truly_nt50358_fwvga
- nt35595_fhd_dsi_cmd_truly_nt50358_qhd
- nt35595_fhd_dsi_cmd_truly_nt50358_wvga
- nt35595_fhd_dsi_cmd_truly_tps65132
- nt35595_fhd_dsi_cmd_truly_tps65132_720p
- nt35595_fhd_dsi_vdo_truly
- nt35595_fhd_dsi_vdo_truly_tps65132
- nt35596_fhd_dsi_vdo_truly

Add your <lcm driver>

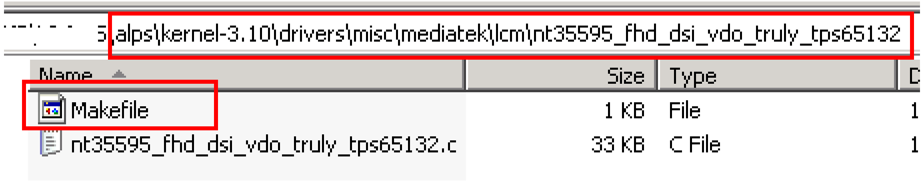


alps\kernel-3.10\drivers\misc\mediatek\lcm\nt35595_fhd_dsi_vdo_truly_tps65132

Name	Size	Type	D
Makefile	1 KB	File	1
nt35595_fhd_dsi_vdo_truly_tps65132.c	33 KB	C File	1

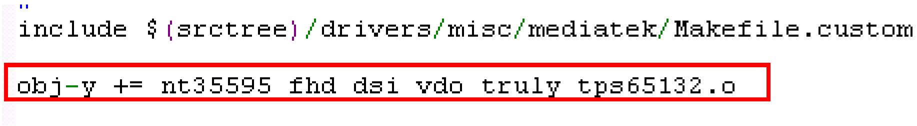
Kernel LCM Configuration (2/5)

- Step 2: Link your <lcm object>
  - Link your compiled <lcm object> in alps\kernel-4.9\drivers\misc\mediatek\lcm\<lcm>\Makefile
  - Take < nt35595_fhd_dsi_vdo_truly_tps65132 > for example:



5\alps\kernel-3.10\drivers\misc\mediatek\lcm\nt35595_fhd_dsi_vdo_truly_tps65132

Name	Size	Type	D
Makefile	1 KB	File	1
nt35595_fhd_dsi_vdo_truly_tps65132.c	33 KB	C File	1

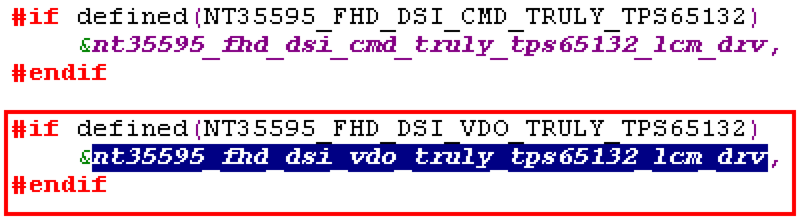


```
include $(srctree)/drivers/misc/mediatek/Makefile.custom

obj-y += nt35595 fhd dsi vdo truly tps65132.o
```

Link your compiled object

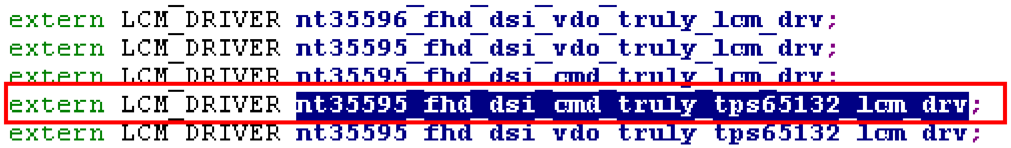
Kernel LCM Configuration (3/5)

- Step 3: Add your <lcm main structure> into lcm list
  - Add your <lcm main structure> into lcm list in alps\kernel-4.9\drivers\misc\mediatek\lcm\mt65xx_lcm_list.c
  - Take <nt35595_fhd_dsi_vdo_truly_tps65132> for example:



```
#if defined(NT35595_FHD_DSI_CMD_TRULY_TPS65132)
    &nt35595_fhd_dsi_cmd_truly_tps65132_lcm_drv,
#endif

#if defined(NT35595_FHD_DSI_VDO_TRULY_TPS65132)
    &nt35595_fhd_dsi_vdo_truly_tps65132_lcm_drv,
#endif
```

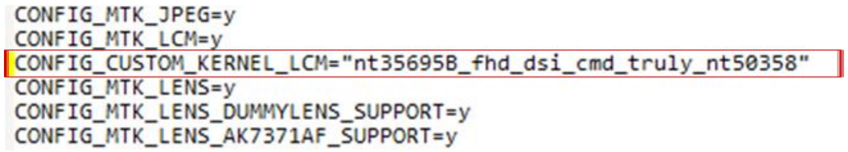
  - alps\kernel-4.9\drivers\misc\mediatek\lcm\inc\mt65xx_lcm_list.h



```
extern LCM_DRIVER nt35596_fhd_dsi_vdo_truly_lcm_drv;
extern LCM_DRIVER nt35595_fhd_dsi_vdo_truly_lcm_drv;
extern LCM_DRIVER nt35595_fhd_dsi_cmd_truly_lcm_drv;
extern LCM_DRIVER nt35595_fhd_dsi_cmd_truly_tps65132_lcm_drv;
extern LCM_DRIVER nt35595_fhd_dsi_vdo_truly_tps65132_lcm_drv;
```

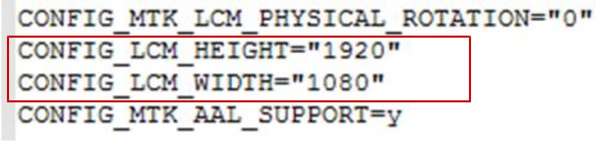
Kernel LCM Configuration (4/5)

- Step 4: Add your <lcm config> in <project> deconfig and modify LCM width and height
- Add your <lcm config> in <project> deconfig in alps\kernel-4.9\arch\<armxx>\configs\<project>_defconfig and <project>_debug_defconfig
  - Take <nt35595B_fhd_dsi_cmd_truly_nt50358> for example:



```
CONFIG_MTK_JPEG=y
CONFIG_MTK_LCM=y
CONFIG_CUSTOM_KERNEL_LCM="nt35695B_fhd_dsi_cmd_truly_nt50358"
CONFIG_MTK_LENS=y
CONFIG_MTK_LENS_DUMMYLENS_SUPPORT=y
CONFIG_MTK_LENS_AK7371AF_SUPPORT=y
```

  - Modify the LCM width according to the new resolution



```
CONFIG_MTK_LCM_PHYSICAL_ROTATION="0"
CONFIG_LCM_HEIGHT="1920"
CONFIG_LCM_WIDTH="1080"
CONFIG_MTK_AAL_SUPPORT=y
```

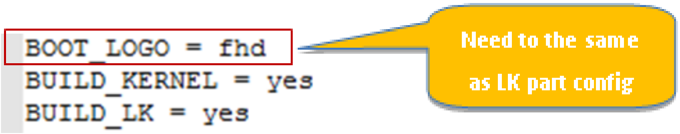
Kernel LCM Configuration (5/5)

- Step 5: Rebuild kernel and bootimage
  - Return to alps folder in console.
  - Rebuild kernel and bootimage, and re-download boot.img

DEVICE LCM CONFIGURATION

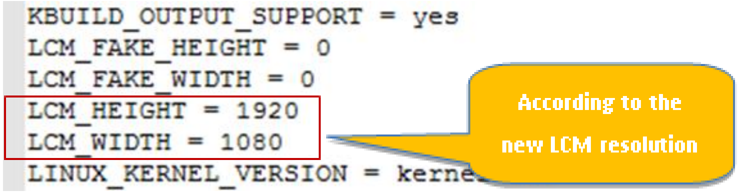
Device LCM Configuration (1/1)

- Step 1: Switch logo modify LCM width and height if LCM resolution is different
- alps\device\<mediatekprojects>\<project>\ProjectConfig.mk
  - Take <nt35595B_fhd_dsi_cmd_truly_nt50358> for example:



```
BOOT_LOGO = fhd
BUILD_KERNEL = yes
BUILD_LK = yes
```

Need to the same as LK part config

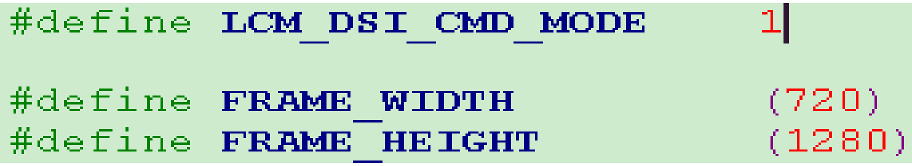


```
KBUILD_OUTPUT_SUPPORT = yes
LCM_FAKE_HEIGHT = 0
LCM_FAKE_WIDTH = 0
LCM_HEIGHT = 1920
LCM_WIDTH = 1080
LINUX_KERNEL_VERSION = kerne
```

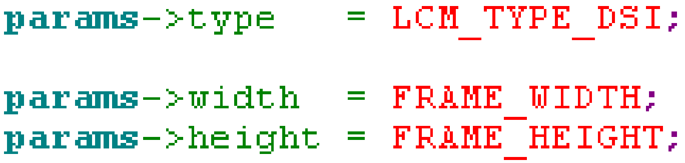
According to the new LCM resolution

Step1：implement driver（1/6）

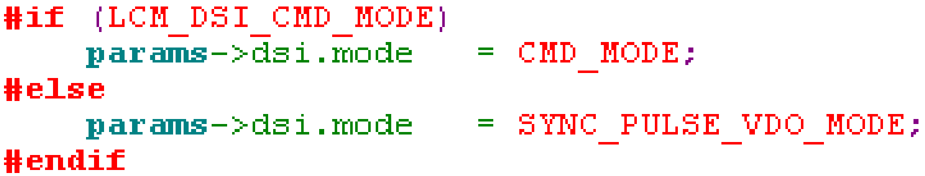
- Fill the LCM parameters
  - Configure the basic information according to the HW connection, LCM type, DSI mode , LCM size and PLL.



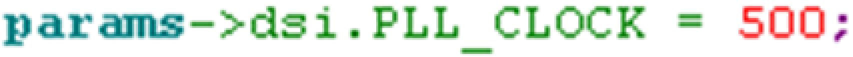
```
#define LCM_DSI_CMD_MODE        1

#define FRAME_WIDTH             (720)
#define FRAME_HEIGHT            (1280)
```



```
params->type   = LCM_TYPE_DSI;

params->width  = FRAME_WIDTH;
params->height = FRAME_HEIGHT;
```



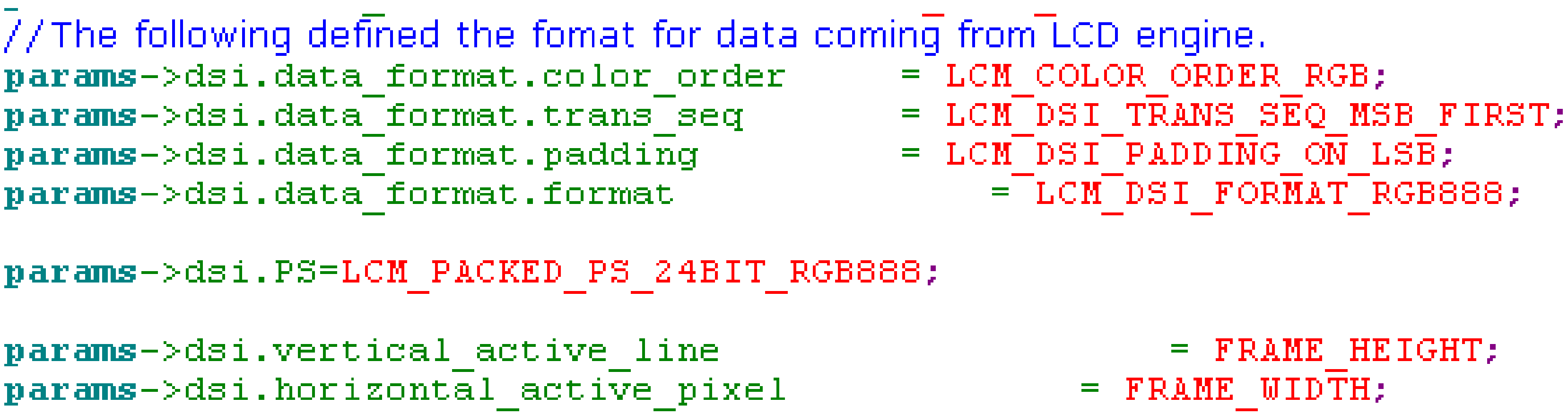
```
#if (LCM_DSI_CMD_MODE)
    params->dsi.mode   = CMD_MODE;
#else
    params->dsi.mode   = SYNC_PULSE_VDO_MODE;
#endif
```



```
params->dsi.PLL_CLOCK = 500;
```

Step1：implement driver（1/6）

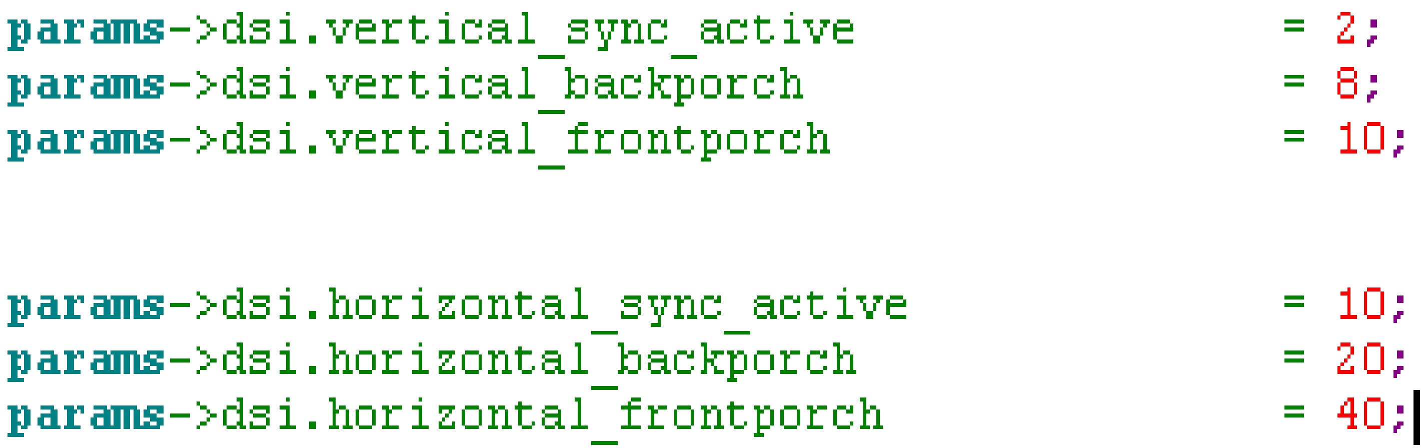
- Fill the LCM parameters
  - Configure vertical line , horizontal pixel and related data format setting.



```
//The following defined the fomat for data coming from LCD engine.
params->dsi.data_format.color_order      = LCM_COLOR_ORDER_RGB;
params->dsi.data_format.trans_seq        = LCM_DSI_TRANS_SEQ_MSB_FIRST;
params->dsi.data_format.padding          = LCM_DSI_PADDING_ON_LSB;
params->dsi.data_format.format              = LCM_DSI_FORMAT_RGB888;

params->dsi.PS=LCM_PACKED_PS_24BIT_RGB888;

params->dsi.vertical_active_line                    = FRAME_HEIGHT;
params->dsi.horizontal_active_pixel             = FRAME_WIDTH;
```

Step1： implement driver（3/6）

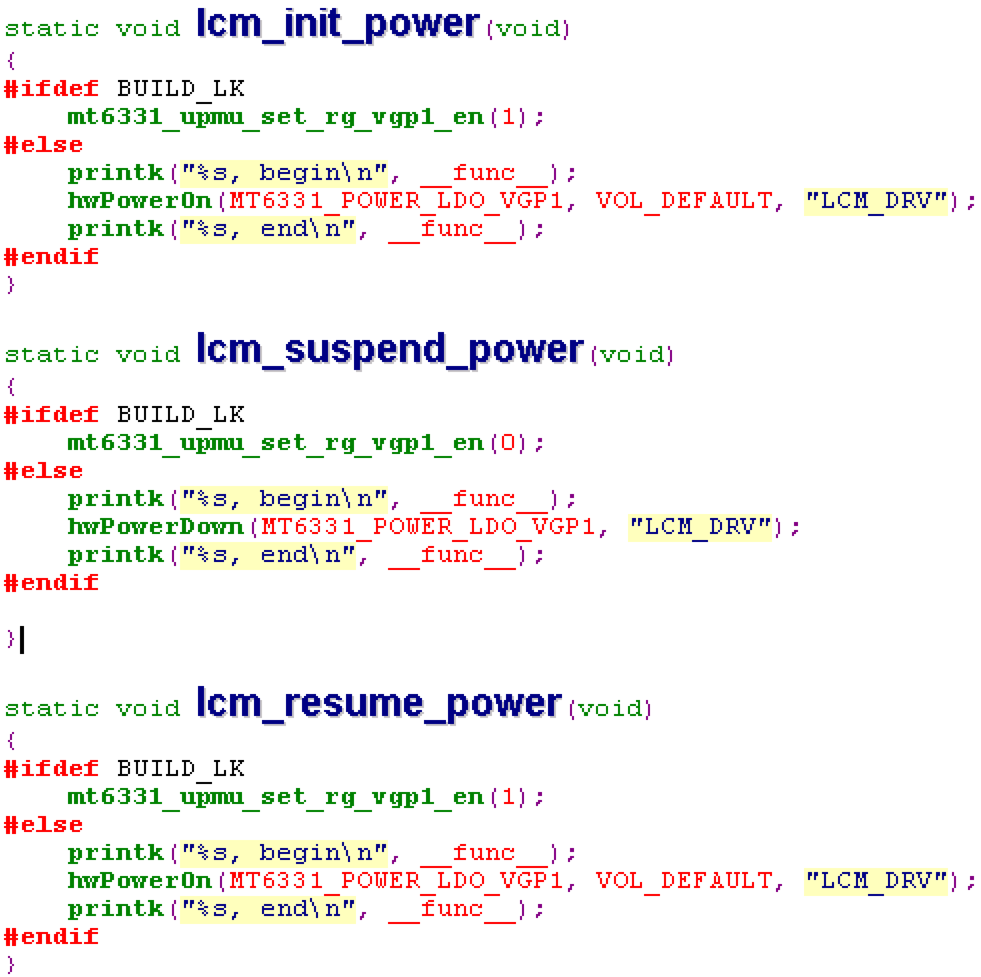
- Fill the LCM parameters
  - Configure video mode timing if params->dsi.mode is not CMD_MODE



```
params->dsi.vertical_sync_active                = 2;
params->dsi.vertical_backporch                  = 8;
params->dsi.vertical_frontporch                 = 10;

params->dsi.horizontal_sync_active              = 10;
params->dsi.horizontal_backporch                = 20;
params->dsi.horizontal_frontporch               = 40;
```

Step2：implement driver（4/6）

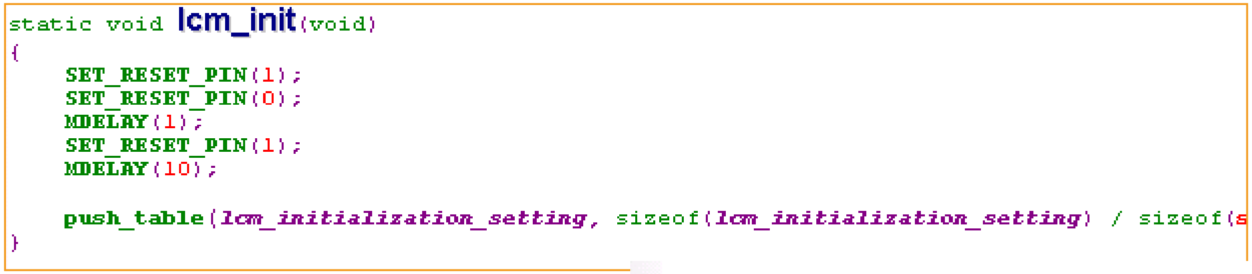
- Implement power on/off
  - Please moving power control into these API from lcm_init. Otherwise , adaptive lcm driver will fail



```
static void lcm_init_power(void)
{
#ifdef BUILD_LK
    mt6331_upmu_set_rg_vgp1_en(1);
#else
    printk("%s, begin\n", __func__);
    hwPowerOn(MT6331_POWER_LDO_VGP1, VOL_DEFAULT, "LCM_DRV");
    printk("%s, end\n", __func__);
#endif
}

static void lcm_suspend_power(void)
{
#ifdef BUILD_LK
    mt6331_upmu_set_rg_vgp1_en(0);
#else
    printk("%s, begin\n", __func__);
    hwPowerDown(MT6331_POWER_LDO_VGP1, "LCM_DRV");
    printk("%s, end\n", __func__);
#endif

}

static void lcm_resume_power(void)
{
#ifdef BUILD_LK
    mt6331_upmu_set_rg_vgp1_en(1);
#else
    printk("%s, begin\n", __func__);
    hwPowerOn(MT6331_POWER_LDO_VGP1, VOL_DEFAULT, "LCM_DRV");
    printk("%s, end\n", __func__);
#endif
}
```

Step2：implement driver（5/6）

- Implement LCM init function
  - According the init process specified in LCM datasheet, pull down/up the reset pin, delay and set LCM init register settings

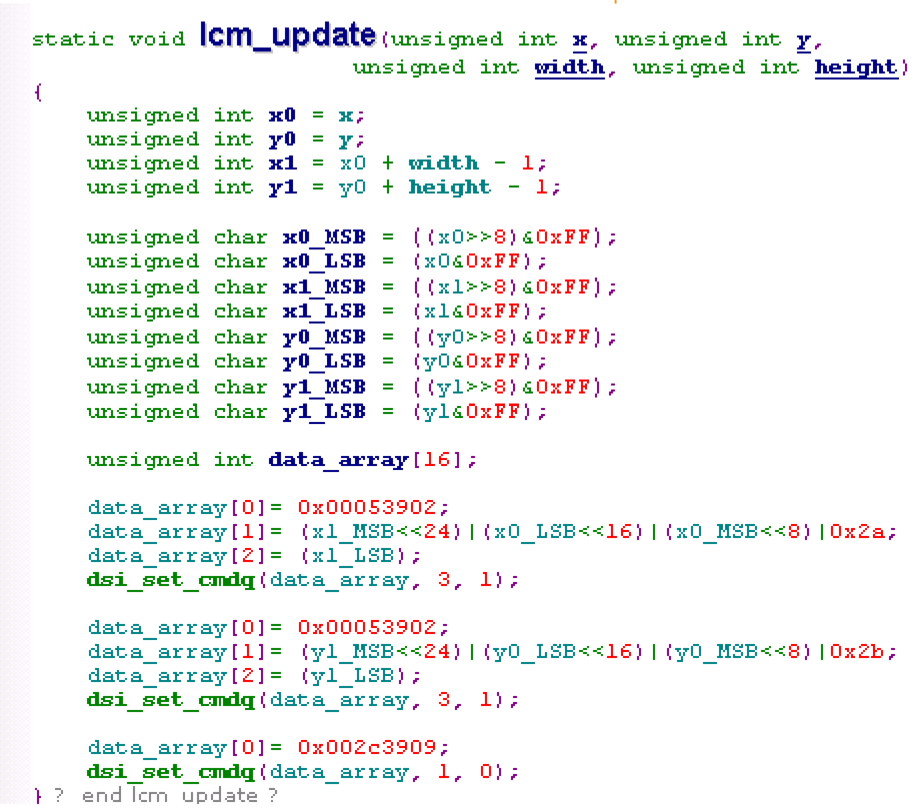


```
static void lcm_init(void)
{
    SET_RESET_PIN(1);
    SET_RESET_PIN(0);
    MDELAY(1);
    SET_RESET_PIN(1);
    MDELAY(10);

    push_table(lcm_initialization_setting, sizeof(lcm_initialization_setting) / sizeof(s
}
```

- Implement LCM update function

  (only for command mode)
  - (Push_table和dsi_set_cmdq

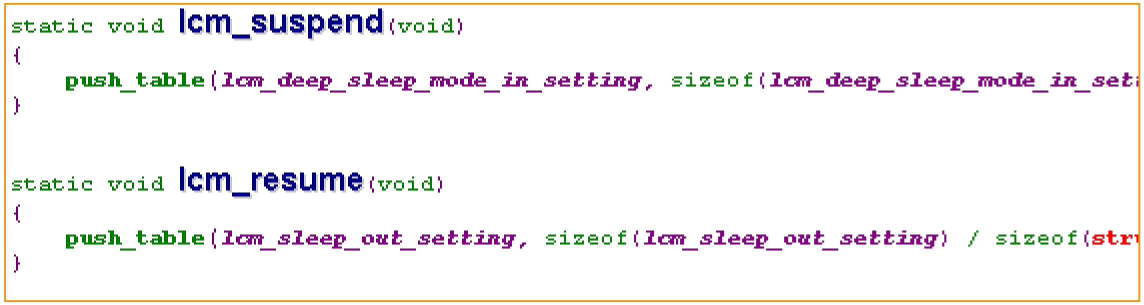
  Please refer to command queue)



```
static void lcm_update(unsigned int x, unsigned int y,
                       unsigned int width, unsigned int height)
{
    unsigned int x0 = x;
    unsigned int y0 = y;
    unsigned int x1 = x0 + width - 1;
    unsigned int y1 = y0 + height - 1;

    unsigned char x0_MSB = ((x0>>8)&0xFF);
    unsigned char x0_LSB = (x0&0xFF);
    unsigned char x1_MSB = ((x1>>8)&0xFF);
    unsigned char x1_LSB = (x1&0xFF);
    unsigned char y0_MSB = ((y0>>8)&0xFF);
    unsigned char y0_LSB = (y0&0xFF);
    unsigned char y1_MSB = ((y1>>8)&0xFF);
    unsigned char y1_LSB = (y1&0xFF);

    unsigned int data_array[16];

    data_array[0]= 0x00053902;
    data_array[1]= (x1_MSB<<24)|(x0_LSB<<16)|(x0_MSB<<8)|0x2a;
    data_array[2]= (x1_LSB);
    dsi_set_cmdq(data_array, 3, 1);

    data_array[0]= 0x00053902;
    data_array[1]= (y1_MSB<<24)|(y0_LSB<<16)|(y0_MSB<<8)|0x2b;
    data_array[2]= (y1_LSB);
    dsi_set_cmdq(data_array, 3, 1);

    data_array[0]= 0x002c3909;
    dsi_set_cmdq(data_array, 1, 0);
} ? end lcm_update ?
```

Step2： implement driver（6/6）

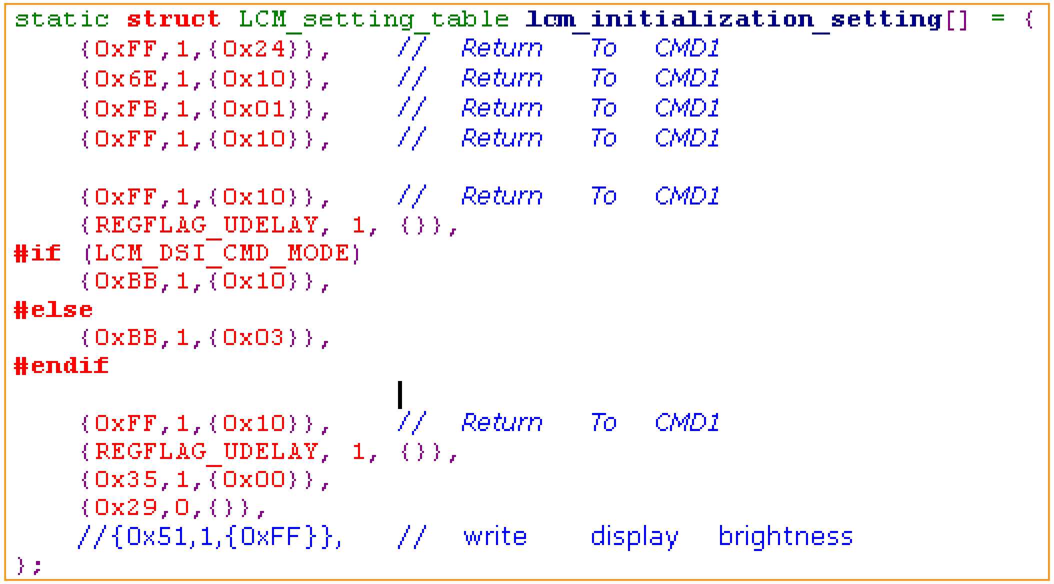
- Implement LCM suspend/resume functions



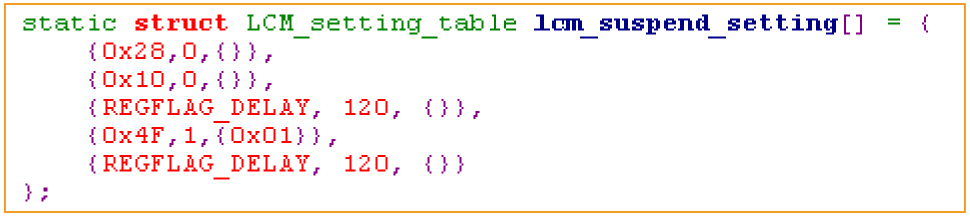
```
static void lcm_suspend(void)
{
    push_table(lcm_deep_sleep_mode_in_setting, sizeof(lcm_deep_sleep_mode_in_sett
}


static void lcm_resume(void)
{
    push_table(lcm_sleep_out_setting, sizeof(lcm_sleep_out_setting) / sizeof(stru
}
```

Step3： Fill in the initialization parameters

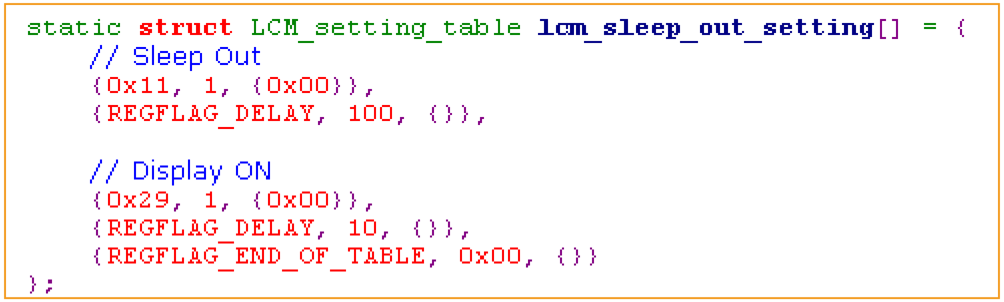
- Get the initial code from LCM FAE



```
static struct LCM_setting_table lcm_initialization_setting[] = {
    {0xFF,1,{0x24}},     //  Return  To  CMD1
    {0x6E,1,{0x10}},     //  Return  To  CMD1
    {0xFB,1,{0x01}},     //  Return  To  CMD1
    {0xFF,1,{0x10}},     //  Return  To  CMD1

    {0xFF,1,{0x10}},     //  Return  To  CMD1
    {REGFLAG_UDELAY, 1, {}},
#if (LCM_DSI_CMD_MODE)
    {0xBB,1,{0x10}},
#else
    {0xBB,1,{0x03}},
#endif

    {0xFF,1,{0x10}},     //  Return  To  CMD1
    {REGFLAG_UDELAY, 1, {}},
    {0x35,1,{0x00}},
    {0x29,0,{}},
    //{0x51,1,{0xFF}},   //  write   display  brightness
};
```



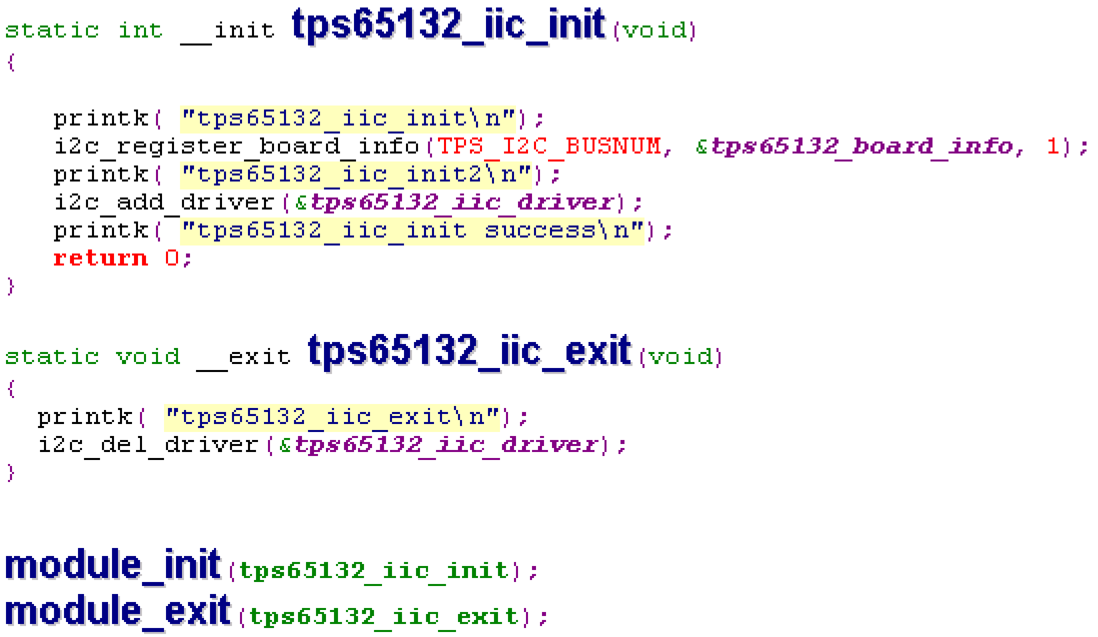
```
static struct LCM_setting_table lcm_suspend_setting[] = {
    {0x28,0,{}},
    {0x10,0,{}},
    {REGFLAG_DELAY, 120, {}},
    {0x4F,1,{0x01}},
    {REGFLAG_DELAY, 120, {}}
};
```



```
static struct LCM_setting_table lcm_sleep_out_setting[] = {
    // Sleep Out
    {0x11, 1, {0x00}},
    {REGFLAG_DELAY, 100, {}},

    // Display ON
    {0x29, 1, {0x00}},
    {REGFLAG_DELAY, 10, {}},
    {REGFLAG_END_OF_TABLE, 0x00, {}}
};
```

Step4: Control the gate IC for LCM power

- **If your project do not use Gate IC or do not controlled by I2C, please pass this step.**
- We will register a I2C client in LCM driver



```
static int __init tps65132_iic_init(void)
{

   printk( "tps65132_iic_init\n");
   i2c_register_board_info(TPS_I2C_BUSNUM, &tps65132_board_info, 1);
   printk( "tps65132_iic_init2\n");
   i2c_add_driver(&tps65132_iic_driver);
   printk( "tps65132_iic_init success\n");
   return 0;
}

static void __exit tps65132_iic_exit(void)
{
  printk( "tps65132_iic_exit\n");
  i2c_del_driver(&tps65132_iic_driver);
}


module_init(tps65132_iic_init);
module_exit(tps65132_iic_exit);
```

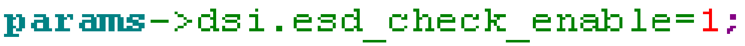
More information please refer to the demo driver in your codebase

Ex: nt35595_fhd_dsi_vdo_truly_nt50358.c

- **If your project use the same gate IC for LCM adapting. Please refer to the FAQ on the MOL: FAQ12444**

Step5： Features Customization (ESD Check 1/2)

- ESD Check
  - Enable:

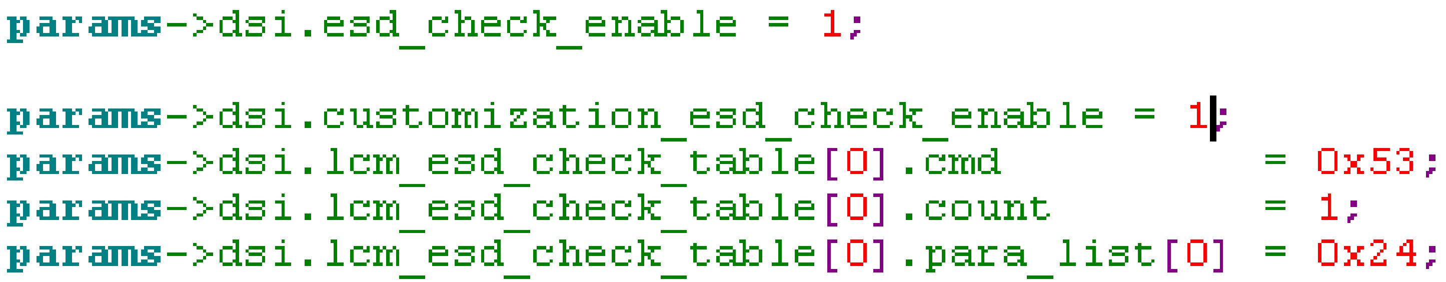


```
params->dsi.esd_check_enable=1;
```

- **Command mode**
  - Please give priority to the use of **polling TE method**
    - Set in LCM driver initial code to let LCM IC TE out
  - You can also do ESD by **reading the LCM IC register**
    - Refer to next page
- **Video mode**
  - Please do ESD by **reading the LCM IC register and polling TE method**

Step5： Features Customization (ESD Check 2/2)

- If do ESD by reading LCM IC register, you can customize as follow in LCM driver



```
params->dsi.esd_check_enable = 1;

params->dsi.customization_esd_check_enable = 1;
params->dsi.lcm_esd_check_table[0].cmd          = 0x53;
params->dsi.lcm_esd_check_table[0].count        = 1;
params->dsi.lcm_esd_check_table[0].para_list[0] = 0x24;
```

- Cmd: the register you will read
- Count: how many parameters will be read back
- Para_list: the right value should been read back

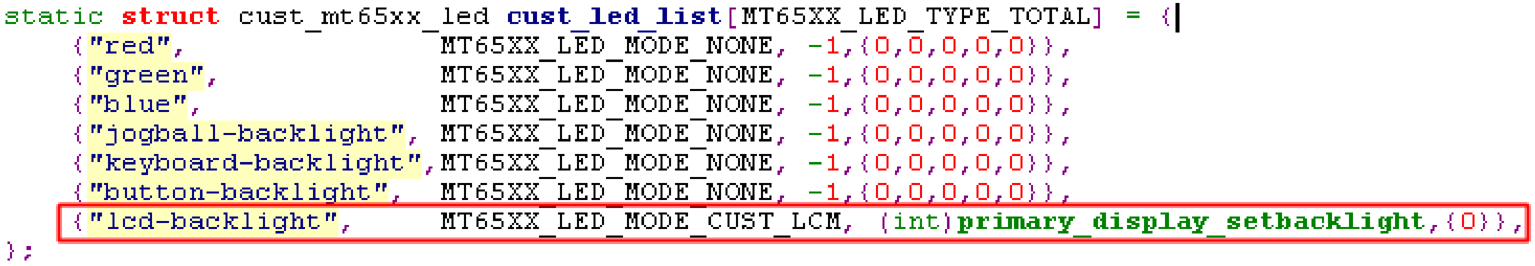
- If the read-back value unequal to the para_list , display system will do recovery

Step5： Features Customization (LCM CABC)

- Customization: (if your project do not use LCM CABC, please pass this step)
  - /kernel-4.9\arch\armxx\boot\dts\mediatek\<project>.dts

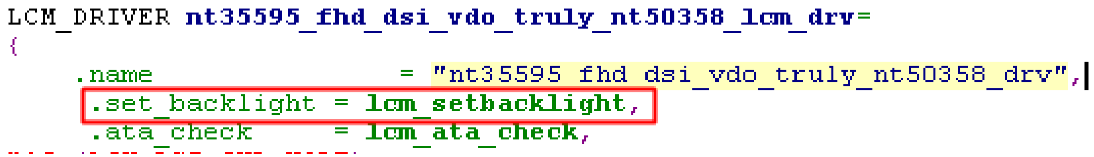
```
led6:led@6 {
        compatible = "mediatek,lcd-backlight";
        led_mode = <5>;    //5 change to 4
        data = <1>;
        pwm_config = <0 3 0 0 0>;
  };
```

— /vendor/mediatek/proprietary/bootable/bootloader/lk/target/[project]/cust_leds.c

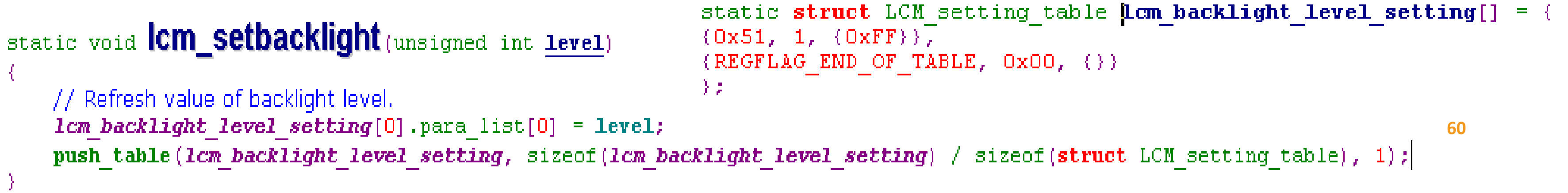


```
static struct cust_mt65xx_led cust_led_list[MT65XX_LED_TYPE_TOTAL] = {
    {"red",                 MT65XX_LED_MODE_NONE, -1,{0,0,0,0,0}},
    {"green",               MT65XX_LED_MODE_NONE, -1,{0,0,0,0,0}},
    {"blue",                MT65XX_LED_MODE_NONE, -1,{0,0,0,0,0}},
    {"jogball-backlight",   MT65XX_LED_MODE_NONE, -1,{0,0,0,0,0}},
    {"keyboard-backlight",MT65XX_LED_MODE_NONE, -1,{0,0,0,0,0}},
    {"button-backlight",    MT65XX_LED_MODE_NONE, -1,{0,0,0,0,0}},
    {"lcd-backlight",       MT65XX_LED_MODE_CUST_LCM, (int)primary_display_setbacklight,{0}},
};
```

- Lk and kernel driver file:



```
LCM_DRIVER nt35595_fhd_dsi_vdo_truly_nt50358_lcm_drv=
{
    .name                = "nt35595_fhd_dsi_vdo_truly_nt50358_drv",
     .set_backlight = lcm_setbacklight,
     .ata_check      = lcm_ata_check,
```

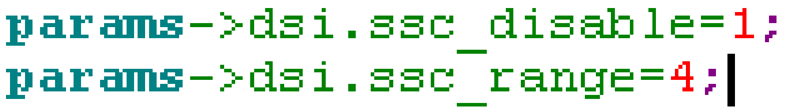


```
static void lcm_setbacklight(unsigned int level)
{
    // Refresh value of backlight level.
    lcm_backlight_level_setting[0].para_list[0] = level;
    push_table(lcm_backlight_level_setting, sizeof(lcm_backlight_level_setting) / sizeof(struct LCM_setting_table), 1);
}
```

```
static struct LCM_setting_table lcm_backlight_level_setting[] = {
{0x51, 1, {0xFF}},
{REGFLAG_END_OF_TABLE, 0x00, {}}
};
```

Step5： Features Customization(Spread Spectrum)

- MIPI Clock Spread Spectrum



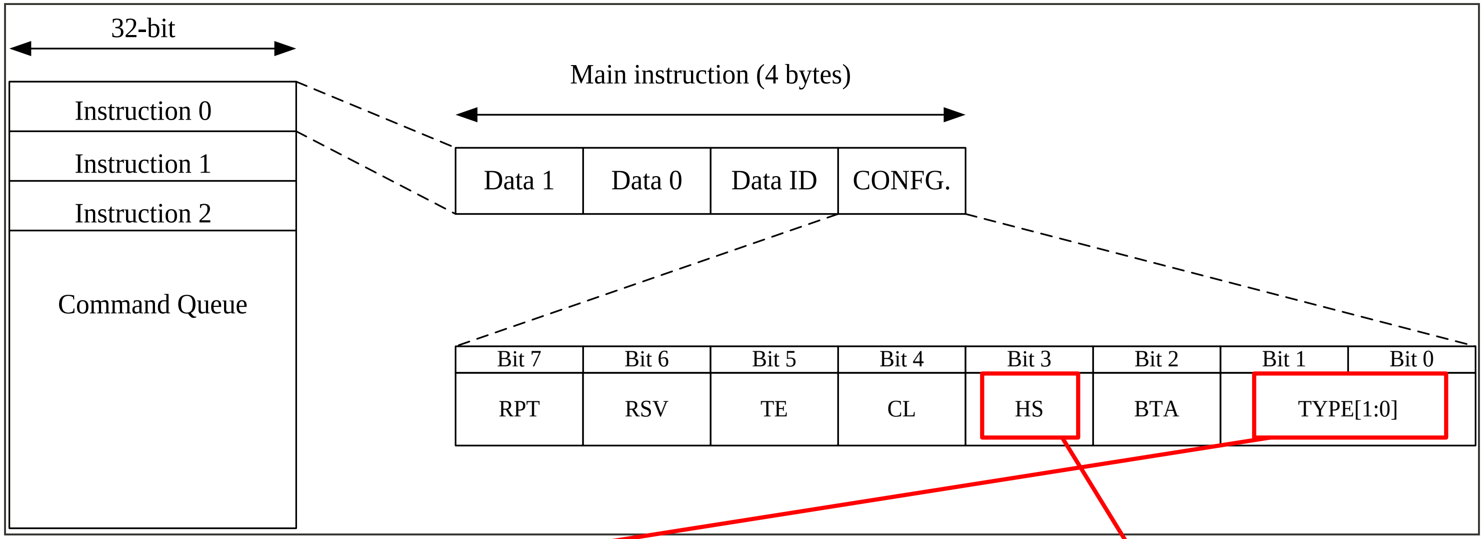
```
params->dsi.ssc_disable=1;
params->dsi.ssc_range=4;
```

- ssc_disable:
  - 1 disable SSC
  - 0 enable SSC
  - default enable SSC
- ssc_range:
  - range 1~8
  - default 5

COMMAND QUEUE

DSI Command Queue(1/2)

- Two dedicated command queues with 32-bit wide and 32-entry depth for each.

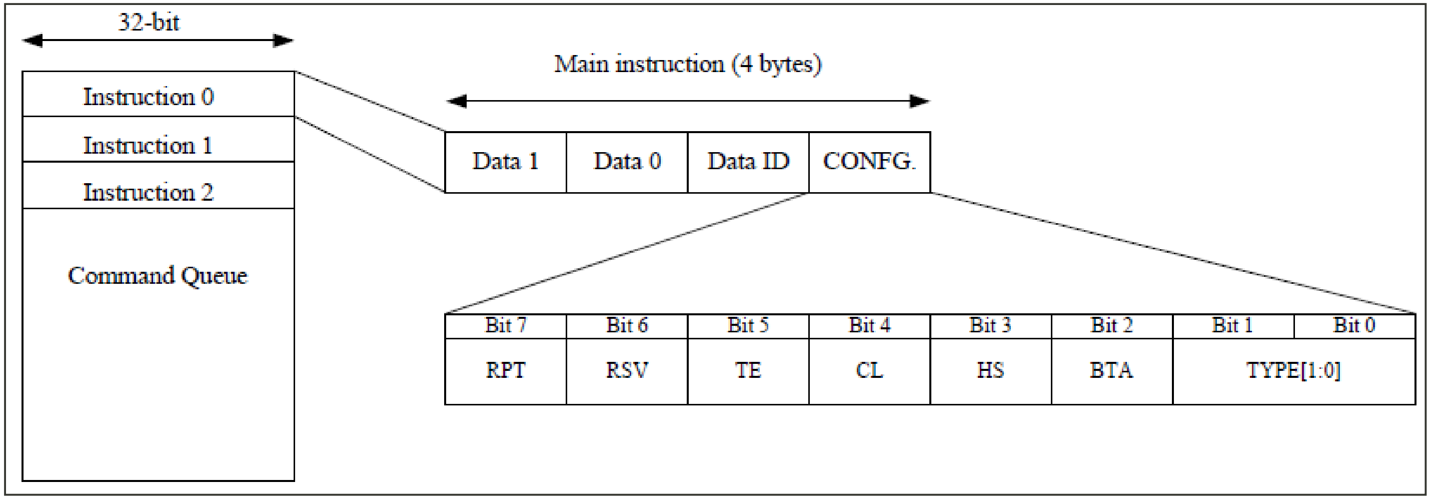


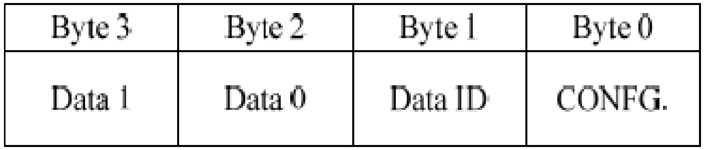
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RPT	RSV	TE	CL	HS	BTA	TYPE[1:0]	

- Type[1:0]
  - 2’b00: (**short packet**) Read/write command
  - 2’b01: (**long packet**) Frame buffer write command (from LCD)
  - 2’b10: (**long packet**) Generic long packet write command (from command queue)
  - 2’b11: (**short packet**) Frame buffer read command

HS=1->HS mode
HS=0->LP mode

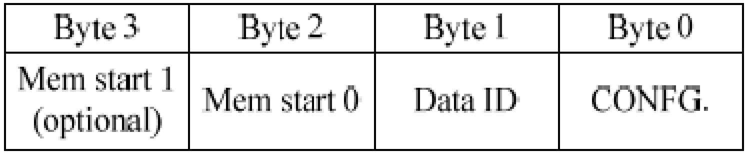
DSI Command Queue(2/2)





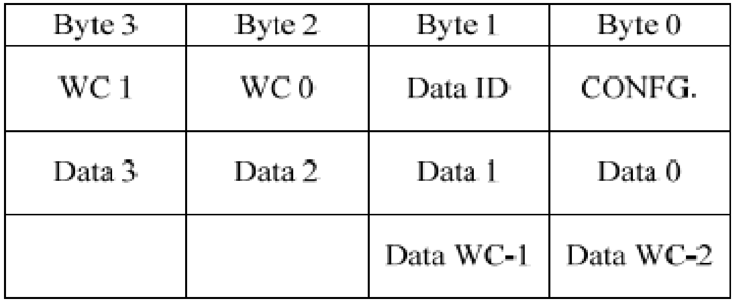
Byte 3	Byte 2	Byte 1	Byte 0
Data 1	Data 0	Data ID	CONFG.

Fig. 5-8: Type-0 instruction format



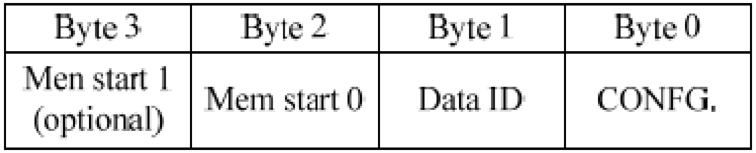
Byte 3	Byte 2	Byte 1	Byte 0
Mem start 1 (optional)	Mem start 0	Data ID	CONFG.

Fig. 5-9: Type-1 instruction format



Byte 3	Byte 2	Byte 1	Byte 0
WC 1	WC 0	Data ID	CONFG.
Data 3	Data 2	Data 1	Data 0
		Data WC-1	Data WC-2

Fig. 5-10: Type-2 instruction format

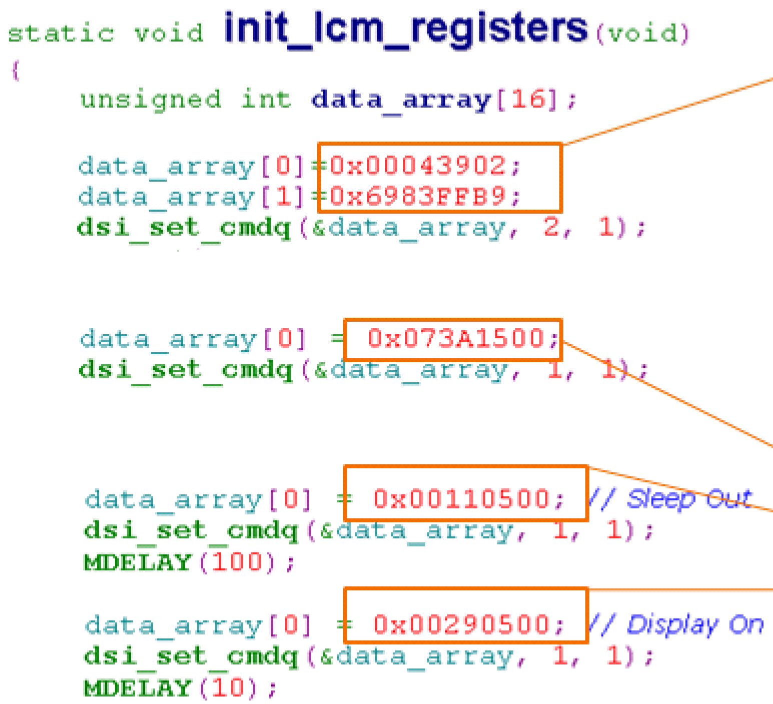


Byte 3	Byte 2	Byte 1	Byte 0
Men start 1 (optional)	Mem start 0	Data ID	CONFG.

Fig. 5-11: Type-3 instruction format

dsi_set_cmdq(*pdata,queue_size,force_update)

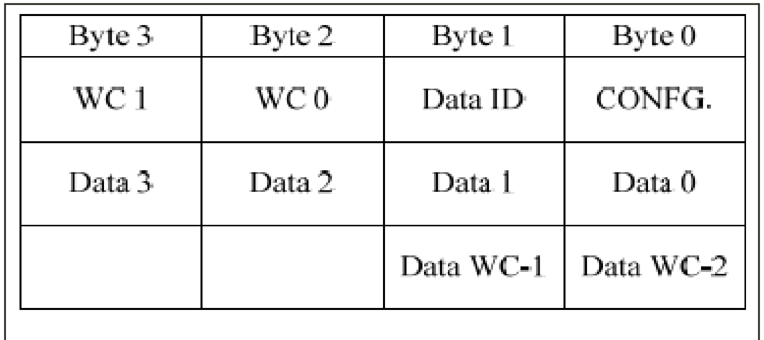
Type 0 & Type 2



```
static void init_lcm_registers(void)
{
    unsigned int data_array[16];

    data_array[0]=0x00043902;
    data_array[1]=0x6983FFB9;
    dsi_set_cmdq(&data_array, 2, 1);


    data_array[0] = 0x073A1500;
    dsi_set_cmdq(&data_array, 1, 1);


    data_array[0] = 0x00110500; // Sleep Out
    dsi_set_cmdq(&data_array, 1, 1);
    MDELAY(100);

    data_array[0] = 0x00290500; // Display On
    dsi_set_cmdq(&data_array, 1, 1);
    MDELAY(10);
```

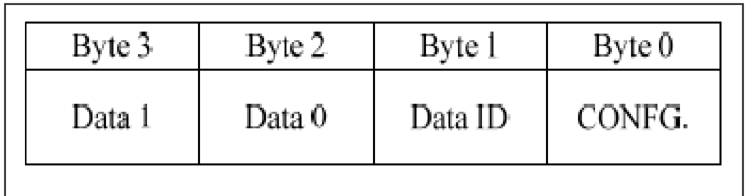
1.data_array[0]=0x00043902
0x02→type 2 Generic Long Write
0x39→DI=0x39,DCS long write command(long packet)
0x0004→WC=4,4Byte data
2.data_array[1]=0x6983FFB9写LCM Register
→command: 0xB9
params:0xFF, 0X83, 0X69

1.data_array[0]=0x073A1500
0x00→type 0 Short Packet Write
0x15→DI=0x15,DCS short write, 1 params
0x05→DI=0X05,DCS short write, no params
0x073A→Data0,Data1 = 0x3A,0X07
→DCS command=0x3A写LCM Register
params=0x07



Byte 3	Byte 2	Byte 1	Byte 0
WC 1	WC 0	Data ID	CONFG.
Data 3	Data 2	Data 1	Data 0
		Data WC-1	Data WC-2

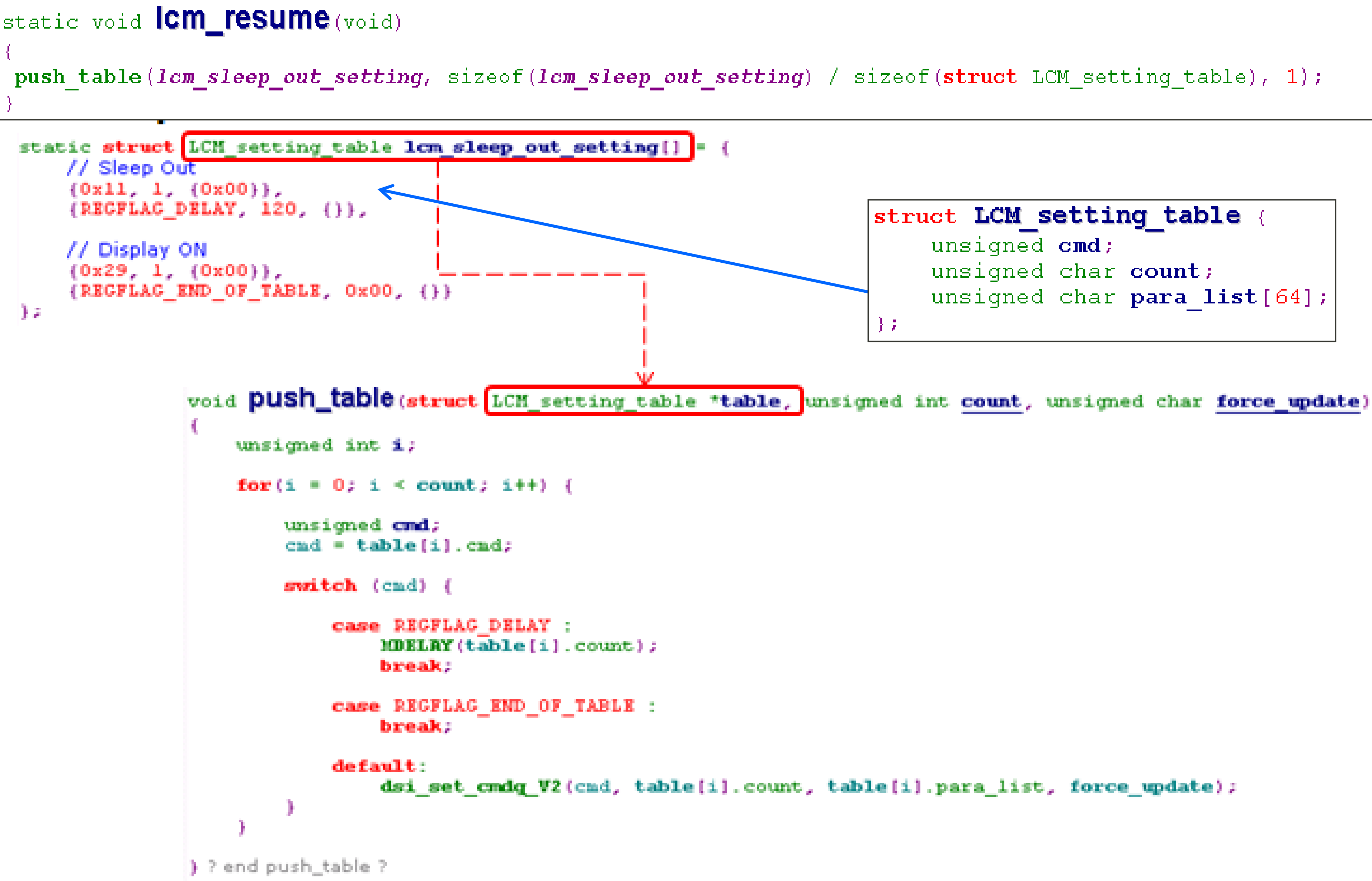
Fig. 5-10: Type-2 instruction format



Byte 3	Byte 2	Byte 1	Byte 0
Data 1	Data 0	Data ID	CONFG.

Fig. 5-8: Type-0 instruction format

dsi_set_cmdq_V2(cmd, count, *para_list, force_update)



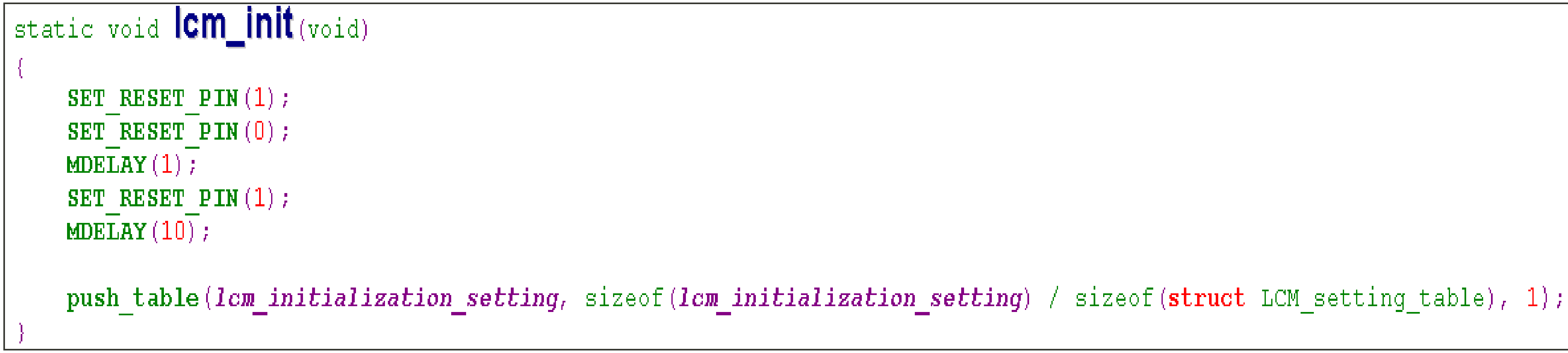
```
static void lcm_resume(void)
{
 push_table(lcm_sleep_out_setting, sizeof(lcm_sleep_out_setting) / sizeof(struct LCM_setting_table), 1);
}
```

```
static struct LCM_setting_table lcm_sleep_out_setting[] = {
    // Sleep Out
    {0x11, 1, {0x00}},
    {REGFLAG_DELAY, 120, {}},

    // Display ON
    {0x29, 1, {0x00}},
    {REGFLAG_END_OF_TABLE, 0x00, {}}
};
```

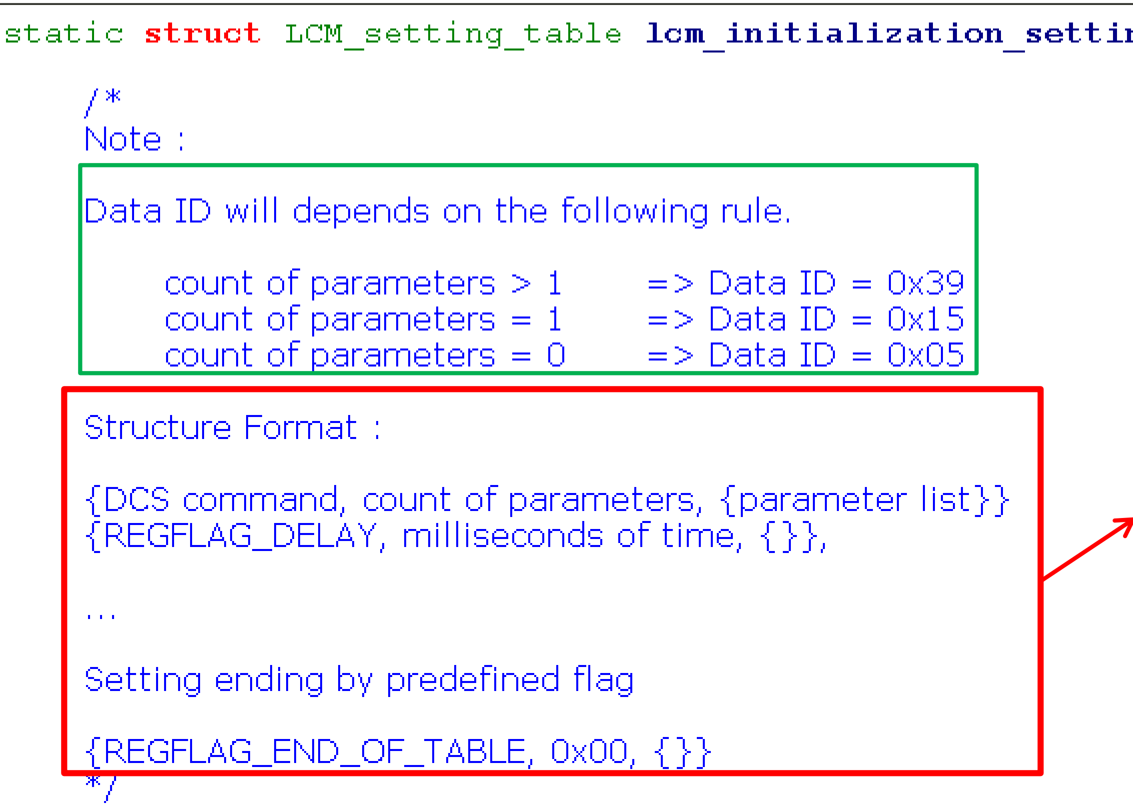
```
struct LCM_setting_table {
    unsigned cmd;
    unsigned char count;
    unsigned char para_list[64];
};
```

```
void push_table(struct LCM_setting_table *table, unsigned int count, unsigned char force_update)
{
    unsigned int i;

    for(i = 0; i < count; i++) {

        unsigned cmd;
        cmd = table[i].cmd;

        switch (cmd) {

            case REGFLAG_DELAY :
                MDELAY(table[i].count);
                break;

            case REGFLAG_END_OF_TABLE :
                break;

            default:
                dsi_set_cmdq_V2(cmd, table[i].count, table[i].para_list, force_update);
        }
    }

} ? end push_table ?
```

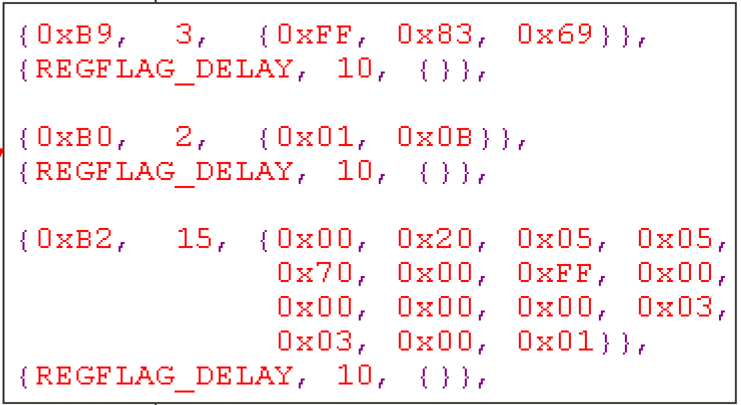
dsi_set_cmdq_V2(cmd, count, *para_list, force_update)



```
static void lcm_init(void)
{
    SET_RESET_PIN(1);
    SET_RESET_PIN(0);
    MDELAY(1);
    SET_RESET_PIN(1);
    MDELAY(10);

    push_table(lcm_initialization_setting, sizeof(lcm_initialization_setting) / sizeof(struct LCM_setting_table), 1);
}
```



```
static struct LCM_setting_table lcm_initialization_setting[] = {

    /*
    Note :

    Data ID will depends on the following rule.

          count of parameters > 1       => Data ID = 0x39
          count of parameters = 1       => Data ID = 0x15
          count of parameters = 0       => Data ID = 0x05

    Structure Format :

    {DCS command, count of parameters, {parameter list}}
    {REGFLAG_DELAY, milliseconds of time, {}},

    ...

    Setting ending by predefined flag

    {REGFLAG_END_OF_TABLE, 0x00, {}}
    */
```

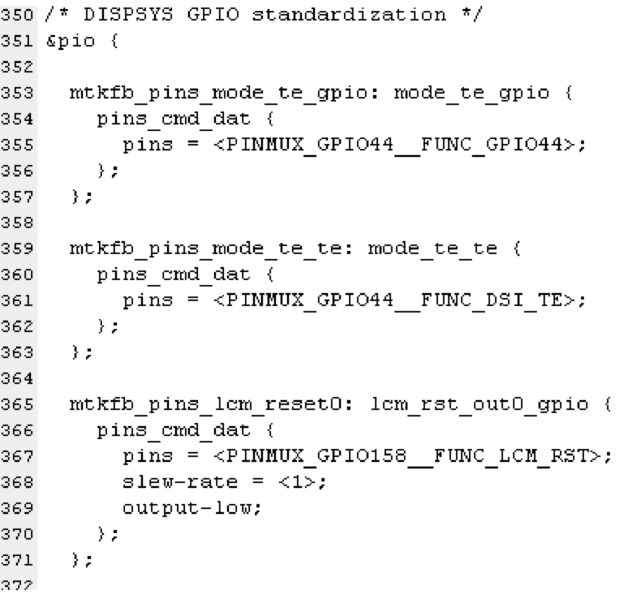


```
{0xB9,   3,   {0xFF,  0x83,  0x69}},
{REGFLAG_DELAY,  10,  {}},

{0xB0,   2,   {0x01,  0x0B}},
{REGFLAG_DELAY,  10,  {}},

{0xB2,   15,  {0x00,  0x20,  0x05,  0x05,
               0x70,  0x00,  0xFF,  0x00,
               0x00,  0x00,  0x00,  0x03,
               0x03,  0x00,  0x01}},
{REGFLAG_DELAY,  10,  {}},
```

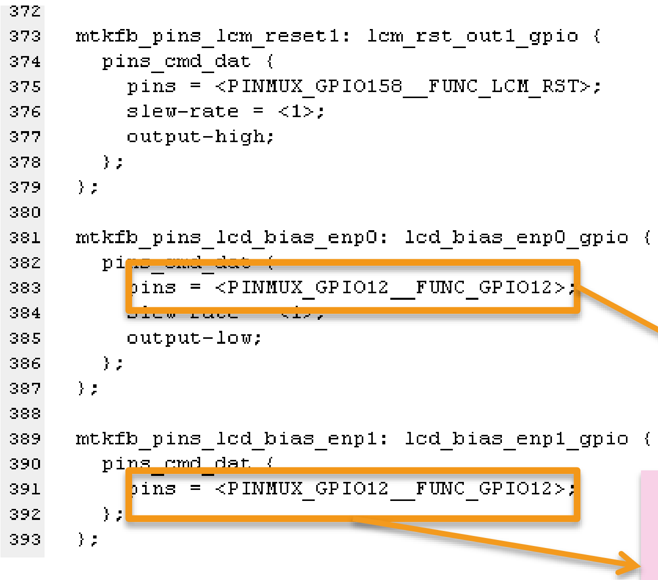
GPIO Kernel Standard Usage

- Display gpio DTS
  - Every item will represent a gpio mode

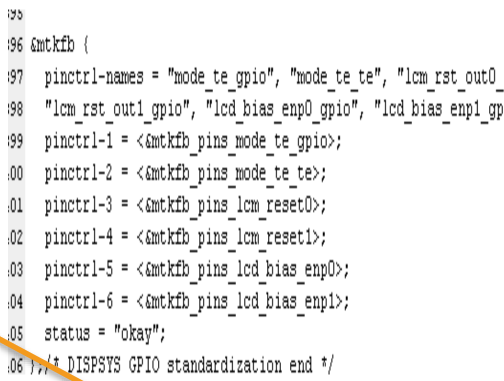
(alps\<kernel>\arch\<arm>\boot\dts\<project>.dts)



```
/* DISPSYS GPIO standardization */
&pio {

  mtkfb_pins_mode_te_gpio: mode_te_gpio {
    pins_cmd_dat {
      pins = <PINMUX_GPIO44__FUNC_GPIO44>;
    };
  };

  mtkfb_pins_mode_te_te: mode_te_te {
    pins_cmd_dat {
      pins = <PINMUX_GPIO44__FUNC_DSI_TE>;
    };
  };

  mtkfb_pins_lcm_reset0: lcm_rst_out0_gpio {
    pins_cmd_dat {
      pins = <PINMUX_GPIO158__FUNC_LCM_RST>;
      slew-rate = <1>;
      output-low;
    };
  };
```



```

  mtkfb_pins_lcm_reset1: lcm_rst_out1_gpio {
    pins_cmd_dat {
      pins = <PINMUX_GPIO158__FUNC_LCM_RST>;
      slew-rate = <1>;
      output-high;
    };
  };

  mtkfb_pins_lcd_bias_enp0: lcd_bias_enp0_gpio {
    pins_cmd_dat {
      pins = <PINMUX_GPIO12__FUNC_GPIO12>;
      slew-rate = <1>;
      output-low;
    };
  };

  mtkfb_pins_lcd_bias_enp1: lcd_bias_enp1_gpio {
    pins_cmd_dat {
      pins = <PINMUX_GPIO12__FUNC_GPIO12>;
    };
  };
```



```

&mtkfb {
  pinctrl-names = "mode_te_gpio", "mode_te_te", "lcm_rst_out0_
  "lcm_rst_out1_gpio", "lcd_bias_enp0_gpio", "lcd_bias_enp1_gp
  pinctrl-1 = <&mtkfb_pins_mode_te_gpio>;
  pinctrl-2 = <&mtkfb_pins_mode_te_te>;
  pinctrl-3 = <&mtkfb_pins_lcm_reset0>;
  pinctrl-4 = <&mtkfb_pins_lcm_reset1>;
  pinctrl-5 = <&mtkfb_pins_lcd_bias_enp0>;
  pinctrl-6 = <&mtkfb_pins_lcd_bias_enp1>;
  status = "okay";
};/* DISPSYS GPIO standardization end */
```

If not use GPIO12, modify this

GPIO Kernel Standard Usage

- how to use in LCM Driver

```
#define  set_gpio_lcd_enp(cmd)\
        lcm_uitl.set_gpio_lcd_enp_bias(cmd)
```

Set GPIO output high

```
set_gpio_lcd_enp(1);
```

Set GPIO output low

```
set_gpio_lcd_enp(0);
```

CONFIDENTIAL B

GPIO

Outline

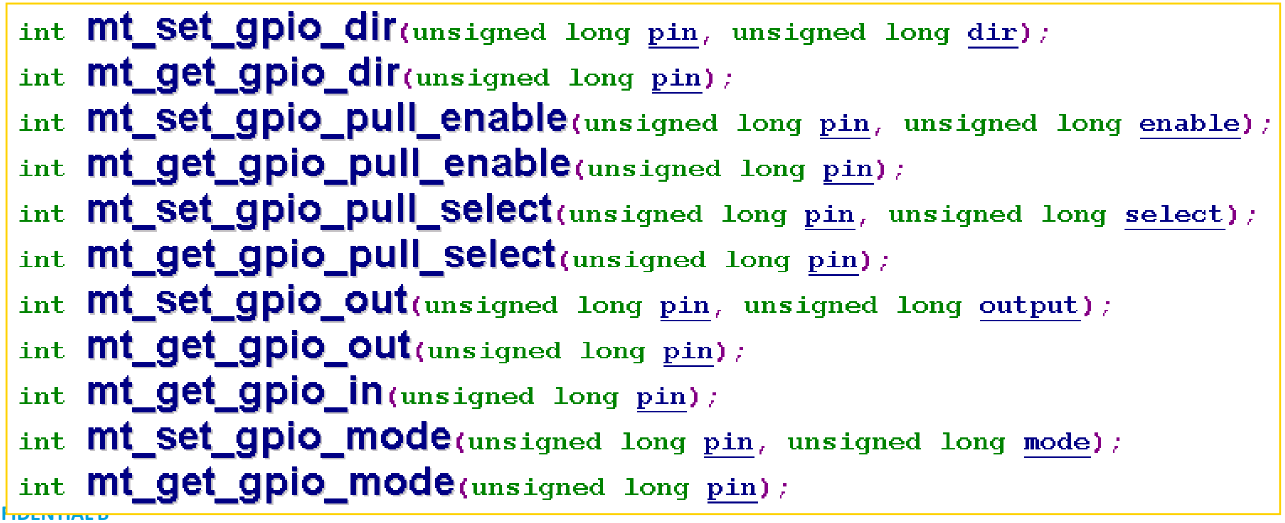
- Pinctrl Node Format
- Pinctrl Usage sample
- GPIOlib API usage
- Check

Note

- The guide just for kernel code, preloader and lk code don't need modify.
- Your code must don't use the old gpio API and the old head file.

```
#include <cust_gpio_usage.h>
#include <mach/mt_gpio.h>
```

✗

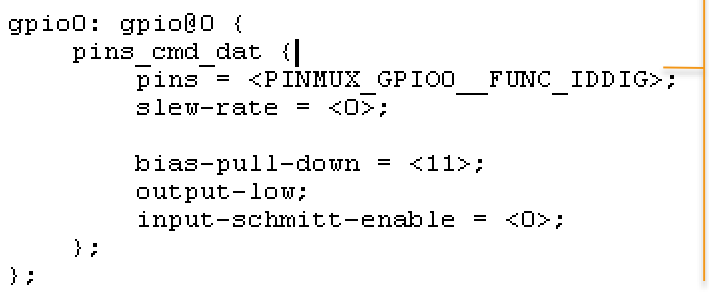


```
int mt_set_gpio_dir(unsigned long pin, unsigned long dir);
int mt_get_gpio_dir(unsigned long pin);
int mt_set_gpio_pull_enable(unsigned long pin, unsigned long enable);
int mt_get_gpio_pull_enable(unsigned long pin);
int mt_set_gpio_pull_select(unsigned long pin, unsigned long select);
int mt_get_gpio_pull_select(unsigned long pin);
int mt_set_gpio_out(unsigned long pin, unsigned long output);
int mt_get_gpio_out(unsigned long pin);
int mt_get_gpio_in(unsigned long pin);
int mt_set_gpio_mode(unsigned long pin, unsigned long mode);
int mt_get_gpio_mode(unsigned long pin);
```

✗

Pinctrl Node Format

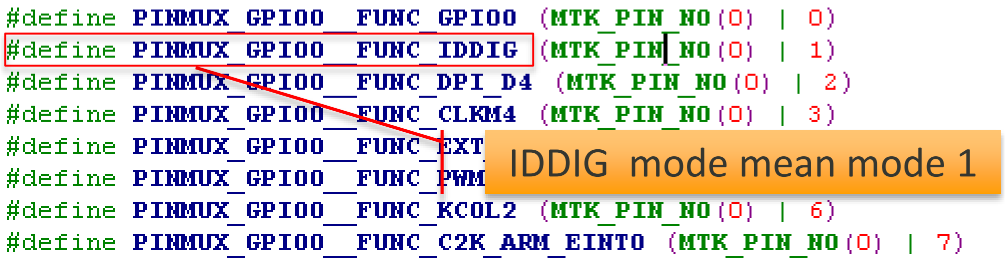
- Set GPIO mode



```
gpio0: gpio@0 {
    pins_cmd_dat {
        pins = <PINMUX_GPIO0__FUNC_IDDIG>;
        slew-rate = <0>;

        bias-pull-down = <11>;
        output-low;
        input-schmitt-enable = <0>;
    };
};
```

gpio pin number and mode, you can find this Macro in kernel-4.4\arch\armxx\boot\dts\include\dt-bingdings\pinctrl\mt67xx-pinfunc.h
this means GPIO0 set to IDDIG mode

kernel-4.4\arch\armxx\boot\dts\include\dt-bingdings\pinctrl\mt67xx-pinfunc.h

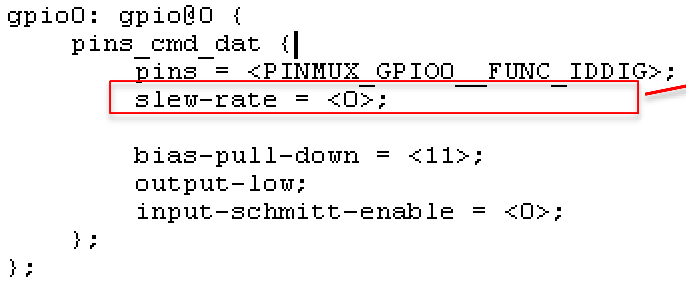


```
#define PINMUX_GPIO0__FUNC_GPIO0 (MTK_PIN_NO(0) | 0)
#define PINMUX_GPIO0__FUNC_IDDIG (MTK_PIN_NO(0) | 1)
#define PINMUX_GPIO0__FUNC_DPI_D4 (MTK_PIN_NO(0) | 2)
#define PINMUX_GPIO0__FUNC_CLKM4 (MTK_PIN_NO(0) | 3)
#define PINMUX_GPIO0__FUNC_EXT
#define PINMUX_GPIO0__FUNC_PWM
#define PINMUX_GPIO0__FUNC_KCOL2 (MTK_PIN_NO(0) | 6)
#define PINMUX_GPIO0__FUNC_C2K_ARM_EINT0 (MTK_PIN_NO(0) | 7)
```

IDDIG mode mean mode 1

Pinctrl Node Format

- Set GPIO direction

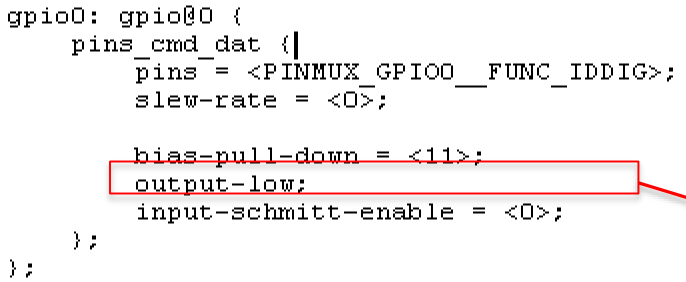


```
gpio0: gpio@0 {
    pins_cmd_dat {
        pins = <PINMUX_GPIO0__FUNC_IDDIG>;
        slew-rate = <0>;

        bias-pull-down = <11>;
        output-low;
        input-schmitt-enable = <0>;
    };
};
```

Means direction in, if you want to set direction out, you can set slew-rate = <1>
(0: input; 1: output)

Pinctrl Node Format

- Set GPIO output value



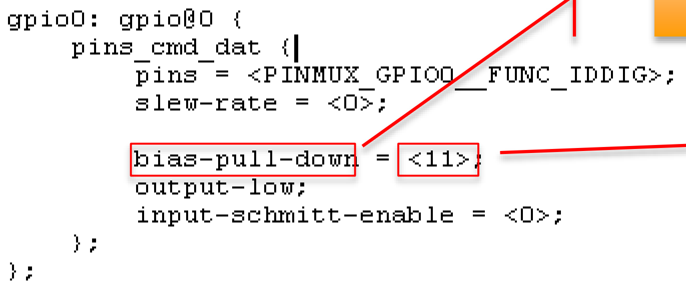
```
gpio0: gpio@0 {
    pins_cmd_dat {
        pins = <PINMUX_GPIO0__FUNC_IDDIG>;
        slew-rate = <0>;

        bias-pull-down = <11>;
        output-low;
        input-schmitt-enable = <0>;
    };
};
```

Means set gpio output low, if you want to set gpio output high, you can write "output-high" in this line
Note: only set gpio direction out , this "output-high" or
"output-low" are effective.

Pinctrl Node Format

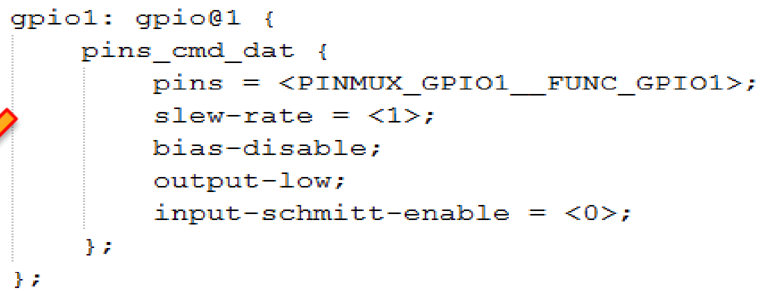
- Set GPIO pull enable/disable

Means set gpio pull enable and pull down

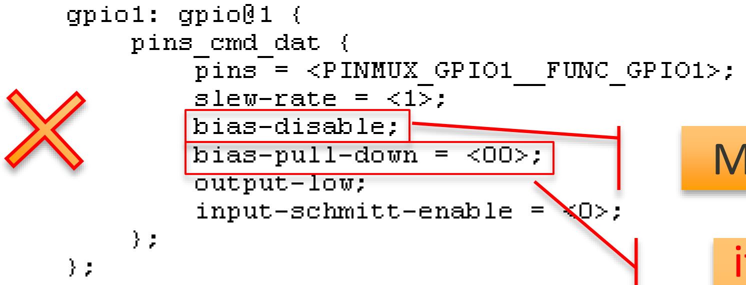


```
gpio0: gpio@0 {
    pins_cmd_dat {
        pins = <PINMUX_GPIO0__FUNC_IDDIG>;
        slew-rate = <0>;

        bias-pull-down = <11>;
        output-low;
        input-schmitt-enable = <0>;
    };
};
```

reserved , you can add 00 now



```
gpio1: gpio@1 {
    pins_cmd_dat {
        pins = <PINMUX_GPIO1__FUNC_GPIO1>;
        slew-rate = <1>;
        bias-disable;
        output-low;
        input-schmitt-enable = <0>;
    };
};
```



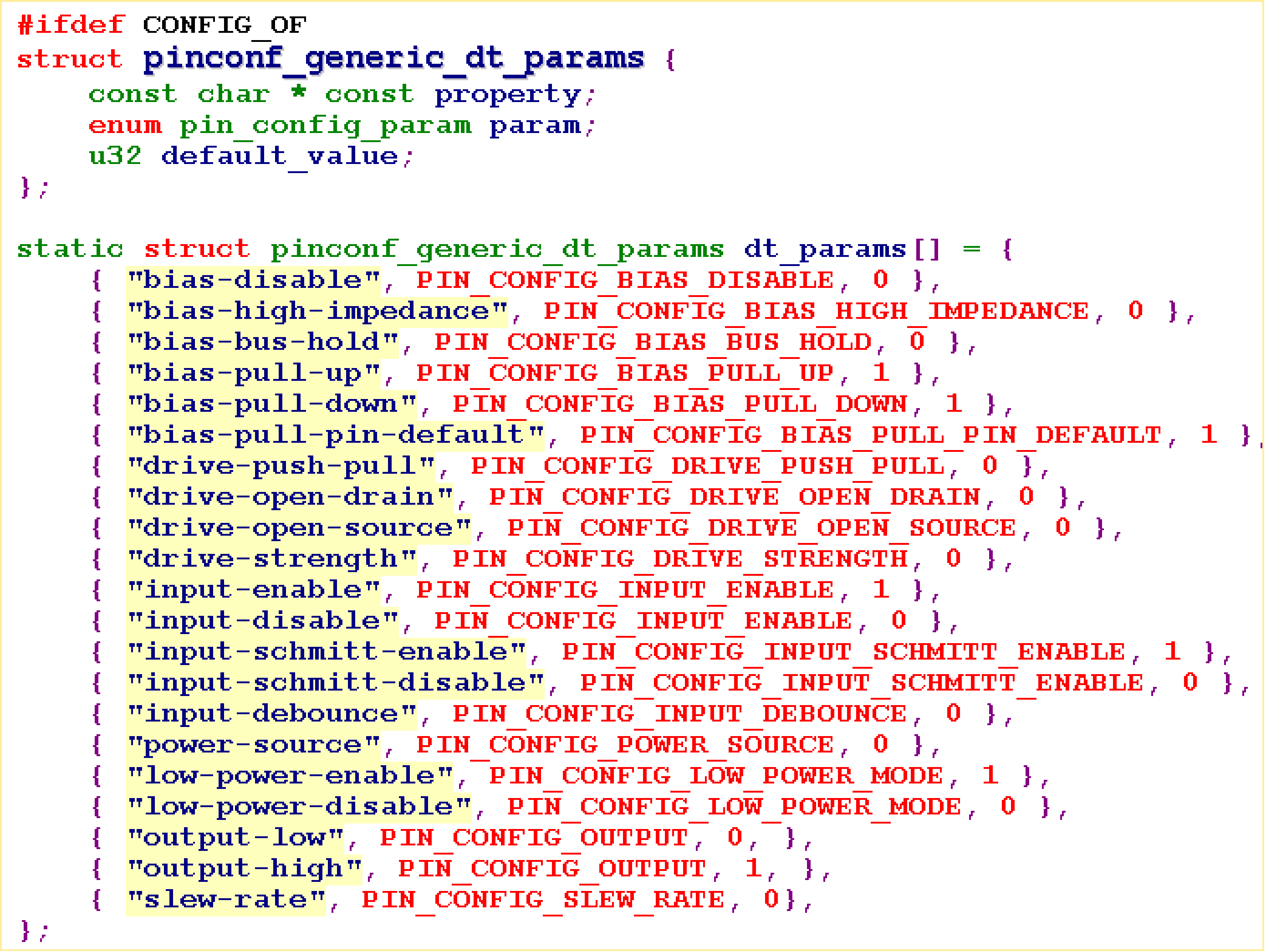
```
gpio1: gpio@1 {
    pins_cmd_dat {
        pins = <PINMUX_GPIO1__FUNC_GPIO1>;
        slew-rate = <1>;
        bias-disable;
        bias-pull-down = <00>;
        output-low;
        input-schmitt-enable = <0>;
    };
};
```

Means set GPIO pull disable

if you set pull disable , you must remove this setting. Because this cmd will set GPIO pull enable

Pinctrl Node Format

- Which parameters can be configured?
  - The parameters is defined in file: kernel-4.4\drivers\pinctrl\ pinconf-generic.c

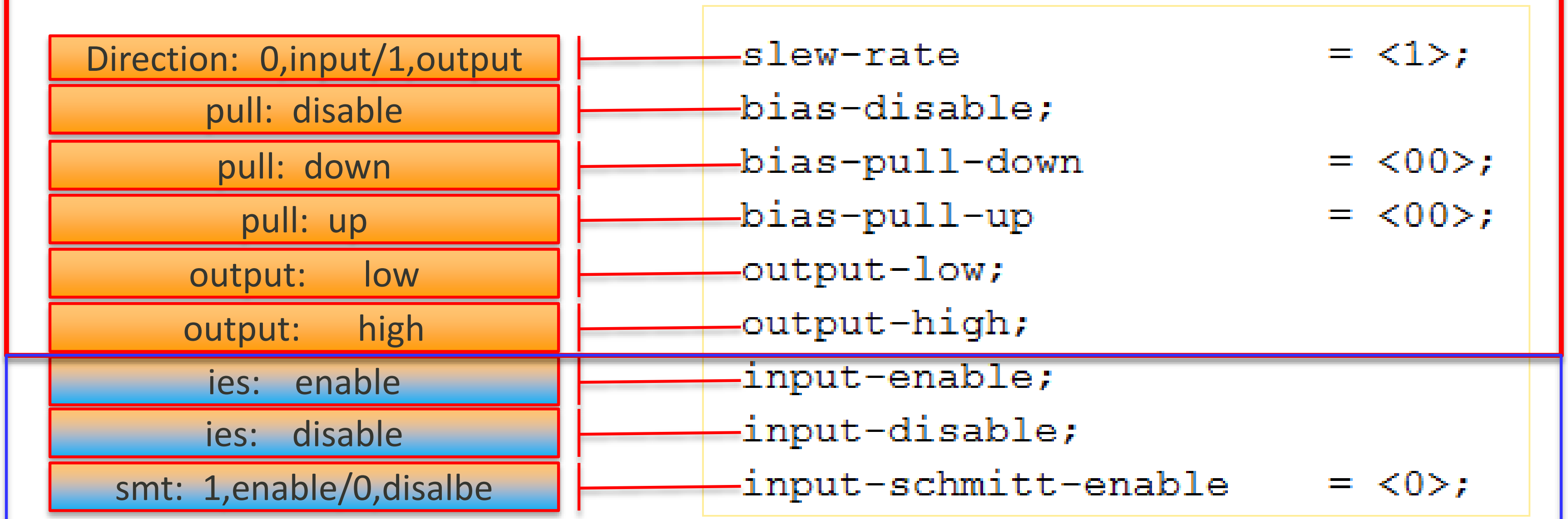


```
#ifdef CONFIG_OF
struct pinconf_generic_dt_params {
	const char * const property;
	enum pin_config_param param;
	u32 default_value;
};

static struct pinconf_generic_dt_params dt_params[] = {
	{ "bias-disable", PIN_CONFIG_BIAS_DISABLE, 0 },
	{ "bias-high-impedance", PIN_CONFIG_BIAS_HIGH_IMPEDANCE, 0 },
	{ "bias-bus-hold", PIN_CONFIG_BIAS_BUS_HOLD, 0 },
	{ "bias-pull-up", PIN_CONFIG_BIAS_PULL_UP, 1 },
	{ "bias-pull-down", PIN_CONFIG_BIAS_PULL_DOWN, 1 },
	{ "bias-pull-pin-default", PIN_CONFIG_BIAS_PULL_PIN_DEFAULT, 1 },
	{ "drive-push-pull", PIN_CONFIG_DRIVE_PUSH_PULL, 0 },
	{ "drive-open-drain", PIN_CONFIG_DRIVE_OPEN_DRAIN, 0 },
	{ "drive-open-source", PIN_CONFIG_DRIVE_OPEN_SOURCE, 0 },
	{ "drive-strength", PIN_CONFIG_DRIVE_STRENGTH, 0 },
	{ "input-enable", PIN_CONFIG_INPUT_ENABLE, 1 },
	{ "input-disable", PIN_CONFIG_INPUT_ENABLE, 0 },
	{ "input-schmitt-enable", PIN_CONFIG_INPUT_SCHMITT_ENABLE, 1 },
	{ "input-schmitt-disable", PIN_CONFIG_INPUT_SCHMITT_ENABLE, 0 },
	{ "input-debounce", PIN_CONFIG_INPUT_DEBOUNCE, 0 },
	{ "power-source", PIN_CONFIG_POWER_SOURCE, 0 },
	{ "low-power-enable", PIN_CONFIG_LOW_POWER_MODE, 1 },
	{ "low-power-disable", PIN_CONFIG_LOW_POWER_MODE, 0 },
	{ "output-low", PIN_CONFIG_OUTPUT, 0, },
	{ "output-high", PIN_CONFIG_OUTPUT, 1, },
	{ "slew-rate", PIN_CONFIG_SLEW_RATE, 0},
};
```

Pinctrl Node Format

- Which parameters can be configured?
  - The parameters can be configured in mt67xx platform as follow.

Generally, we just need configure these parameters.



```
slew-rate                     = <1>;
bias-disable;
bias-pull-down                = <00>;
bias-pull-up                  = <00>;
output-low;
output-high;
input-enable;
input-disable;
input-schmitt-enable          = <0>;
```

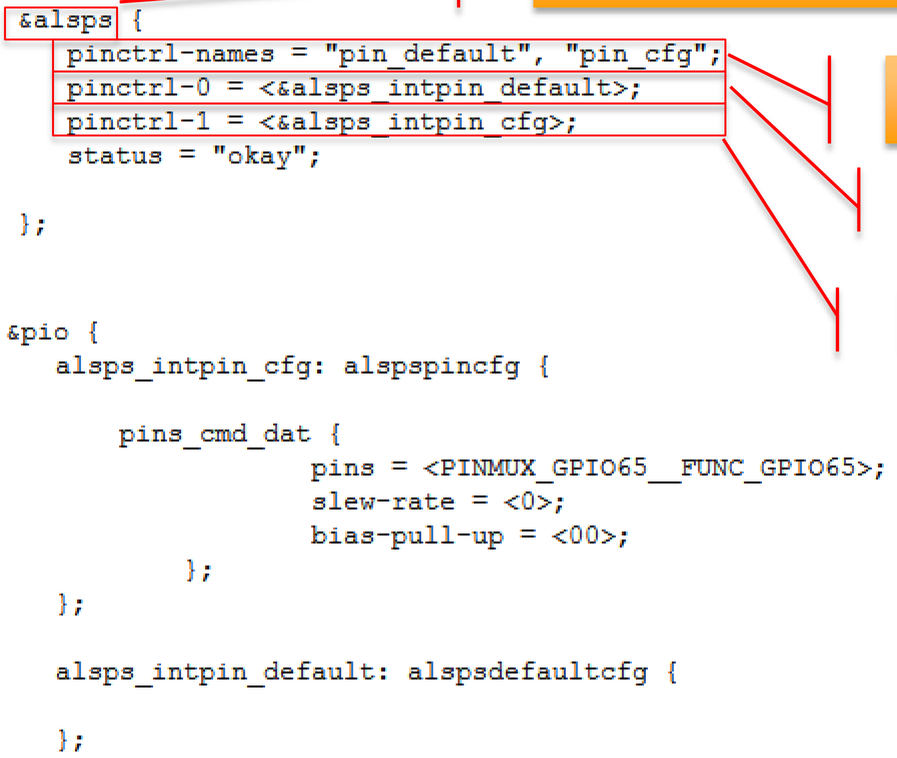
If you don’t have special requirement, please don’t configure these parameters. (input-enable, input-disable, input-schmitt-enable)

Pinctrl Usage sample

alsps use GPIO pinctrl

- Step1 -- write your dts

$(project).dts



```
&alsps {
    pinctrl-names = "pin_default", "pin_cfg";
    pinctrl-0 = <&alsps_intpin_default>;
    pinctrl-1 = <&alsps_intpin_cfg>;
    status = "okay";

};


&pio {
   alsps_intpin_cfg: alspspincfg {

       pins_cmd_dat {
                 pins = <PINMUX_GPIO65__FUNC_GPIO65>;
                 slew-rate = <0>;
                 bias-pull-up = <00>;
           };
   };

   alsps_intpin_default: alspsdefaultcfg {

   };
```

attach to mt67xx.dtsi alsps node

add alsps pinctrl name

add alsps default configs

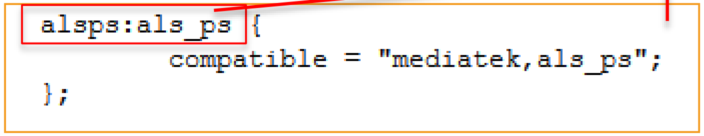
add alsps configure gpio for eint

Pinctrl Usage sample

alsps use GPIO pinctrl

- Step2 -- write your dtsi

mt67xx.dtsi



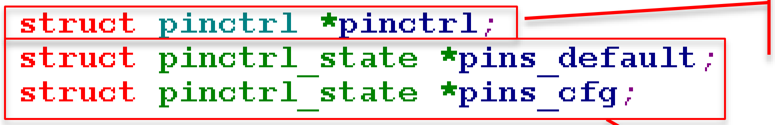
```
alsps:als_ps {
            compatible = "mediatek,als_ps";
};
```

add alsps node at mt67xx.dtsi to attach

Pinctrl Usage sample

alsps use GPIO pinctrl

- Step3 driver coding



```
struct pinctrl *pinctrl;
struct pinctrl_state *pins_default;
struct pinctrl_state *pins_cfg;
```

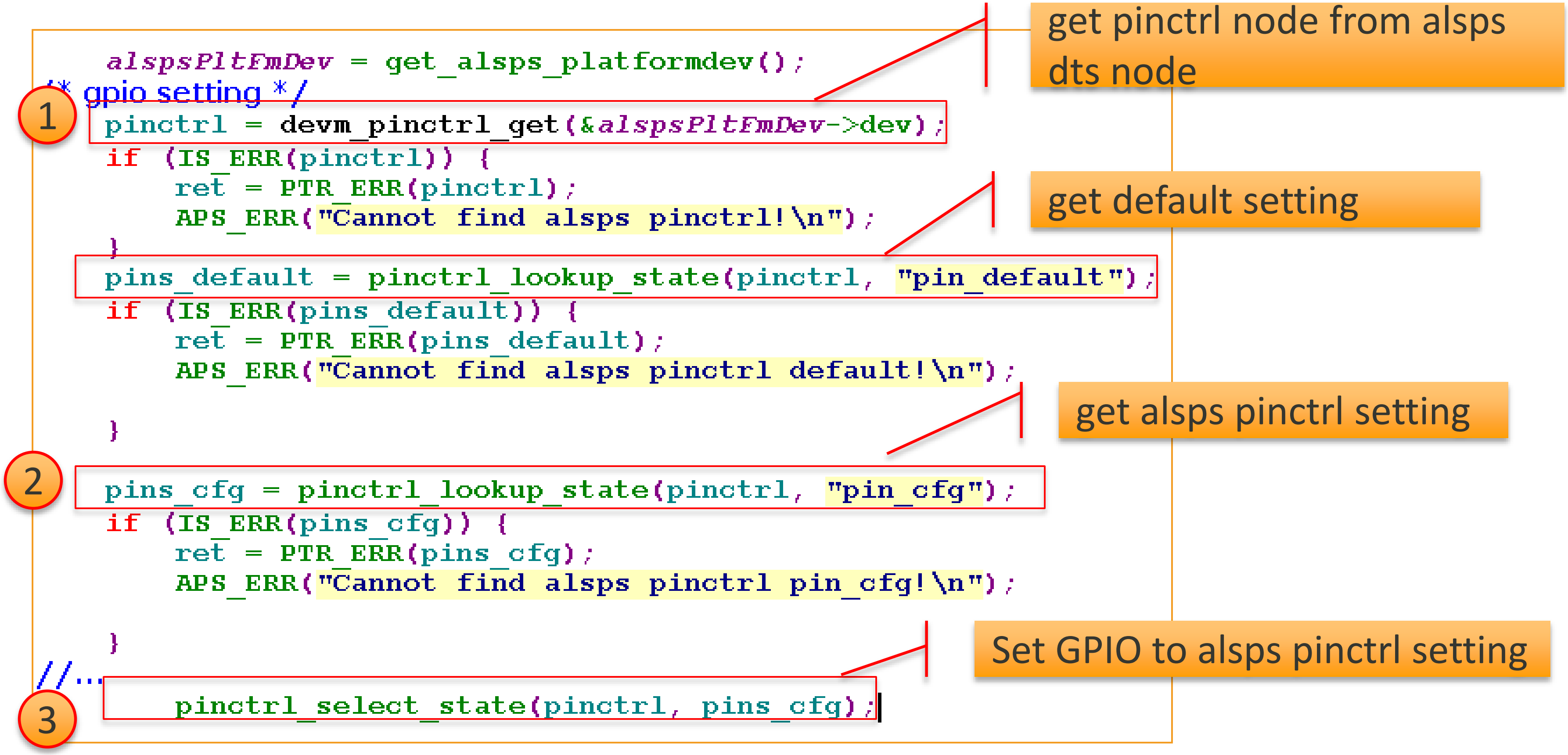
define pinctrl node pointer

define pinctrl select node pointer

Pinctrl Usage sample

alsps use GPIO pinctrl

- Step3 driver coding



```
    alspsPltFmDev = get_alsps_platformdev();
/* gpio setting */
    pinctrl = devm_pinctrl_get(&alspsPltFmDev->dev);
    if (IS_ERR(pinctrl)) {
        ret = PTR_ERR(pinctrl);
        APS_ERR("Cannot find alsps pinctrl!\n");
    }
    pins_default = pinctrl_lookup_state(pinctrl, "pin_default");
    if (IS_ERR(pins_default)) {
        ret = PTR_ERR(pins_default);
        APS_ERR("Cannot find alsps pinctrl default!\n");

    }

    pins_cfg = pinctrl_lookup_state(pinctrl, "pin_cfg");
    if (IS_ERR(pins_cfg)) {
        ret = PTR_ERR(pins_cfg);
        APS_ERR("Cannot find alsps pinctrl pin_cfg!\n");

    }
//...
        pinctrl_select_state(pinctrl, pins_cfg);
```

get pinctrl node from alsps dts node

get default setting

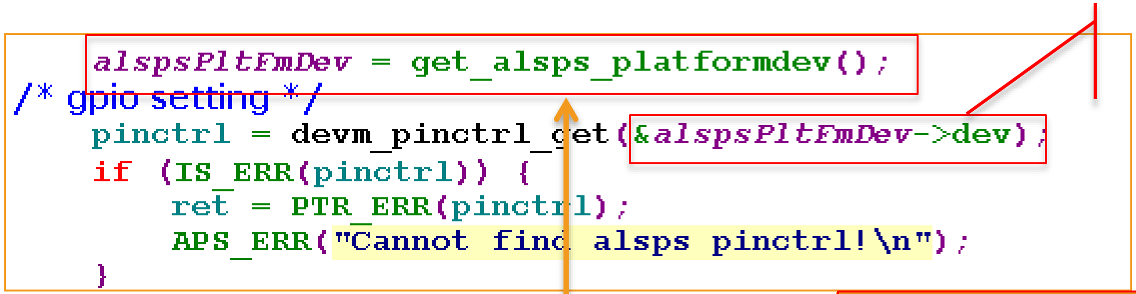
get alsps pinctrl setting

Set GPIO to alsps pinctrl setting

Pinctrl Usage sample

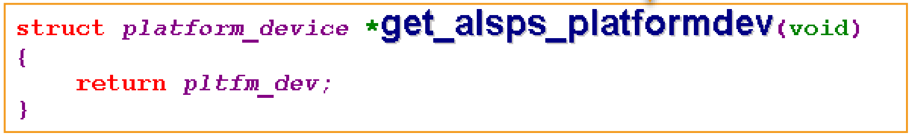
alsps use GPIO pinctrl

- API usage detail



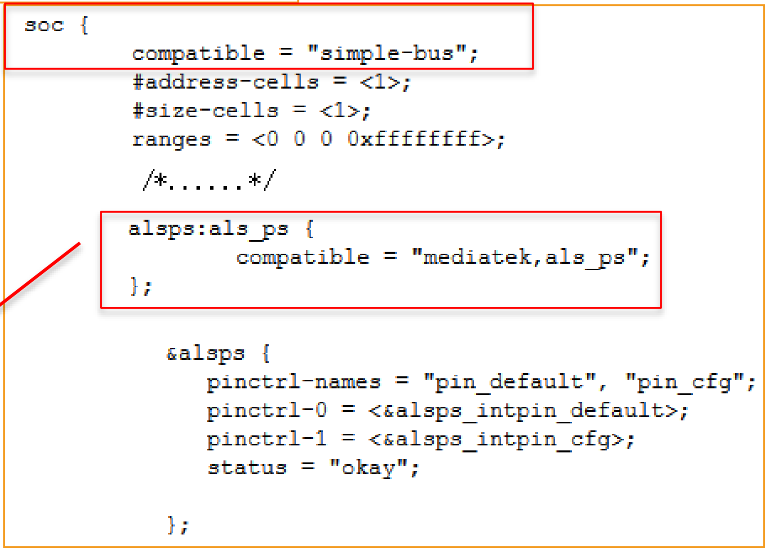
```
    alspsPltFmDev = get_alsps_platformdev();
/* gpio setting */
    pinctrl = devm_pinctrl_get(&alspsPltFmDev->dev);
    if (IS_ERR(pinctrl)) {
        ret = PTR_ERR(pinctrl);
        APS_ERR("Cannot find alsps pinctrl!\n");
    }
```

The pinctrl dev must match with the dts node.



```
struct platform_device *get_alsps_platformdev(void)
{
    return pltfm_dev;
}
```

Usually, pinctrl dev is created as platform device

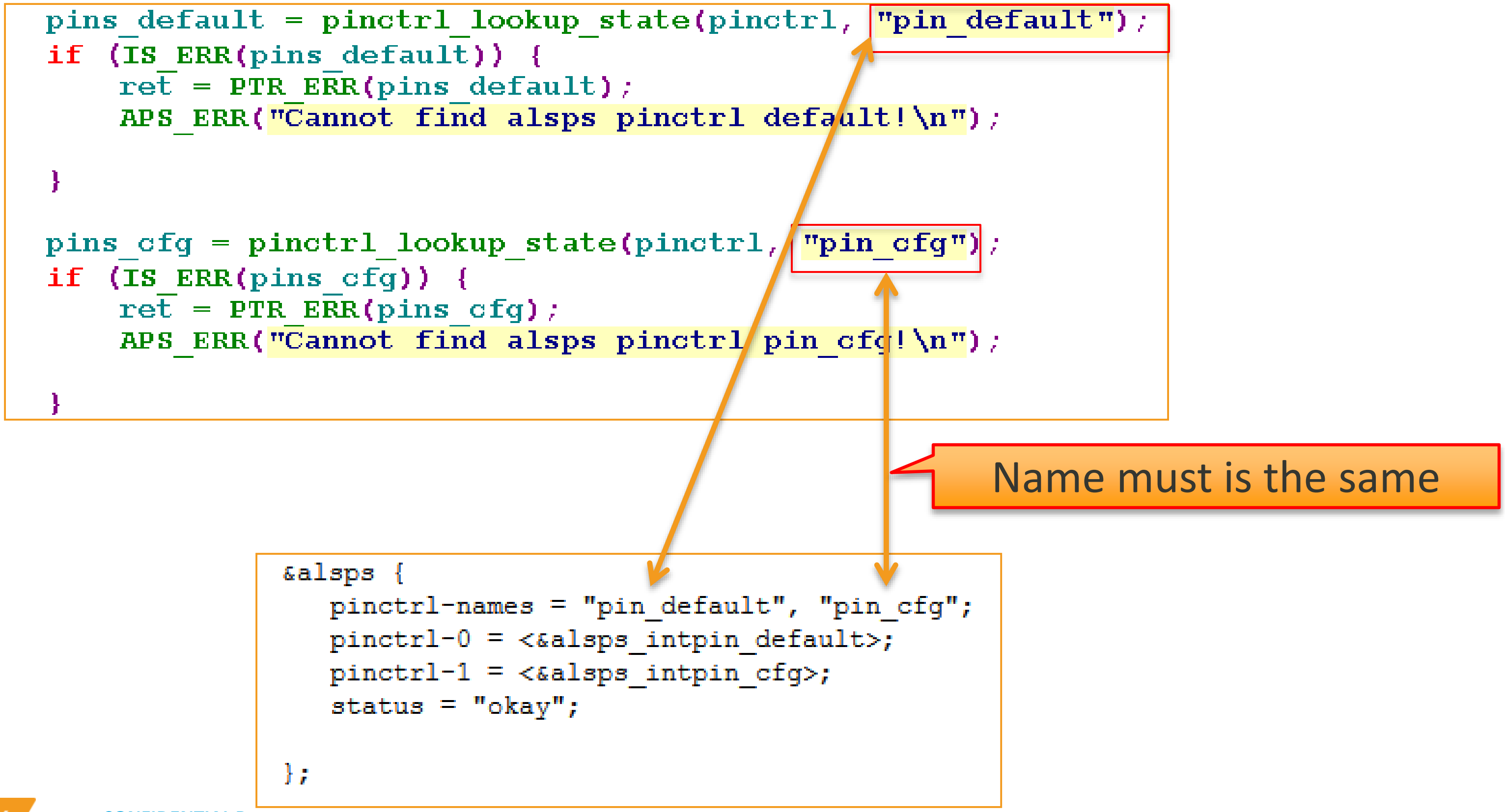


```
soc {
        compatible = "simple-bus";
        #address-cells = <1>;
        #size-cells = <1>;
        ranges = <0 0 0 0xffffffff>;

         /*......*/

        alsps:als_ps {
                compatible = "mediatek,als_ps";
        };

           &alsps {
              pinctrl-names = "pin_default", "pin_cfg";
              pinctrl-0 = <&alsps_intpin_default>;
              pinctrl-1 = <&alsps_intpin_cfg>;
              status = "okay";

           };
```

Pinctrl Usage sample

alsps use GPIO pinctrl

- API usage detail



```
pins_default = pinctrl_lookup_state(pinctrl, "pin_default");
if (IS_ERR(pins_default)) {
    ret = PTR_ERR(pins_default);
    APS_ERR("Cannot find alsps pinctrl default!\n");

}

pins_cfg = pinctrl_lookup_state(pinctrl, "pin_cfg");
if (IS_ERR(pins_cfg)) {
    ret = PTR_ERR(pins_cfg);
    APS_ERR("Cannot find alsps pinctrl pin_cfg!\n");

}
```

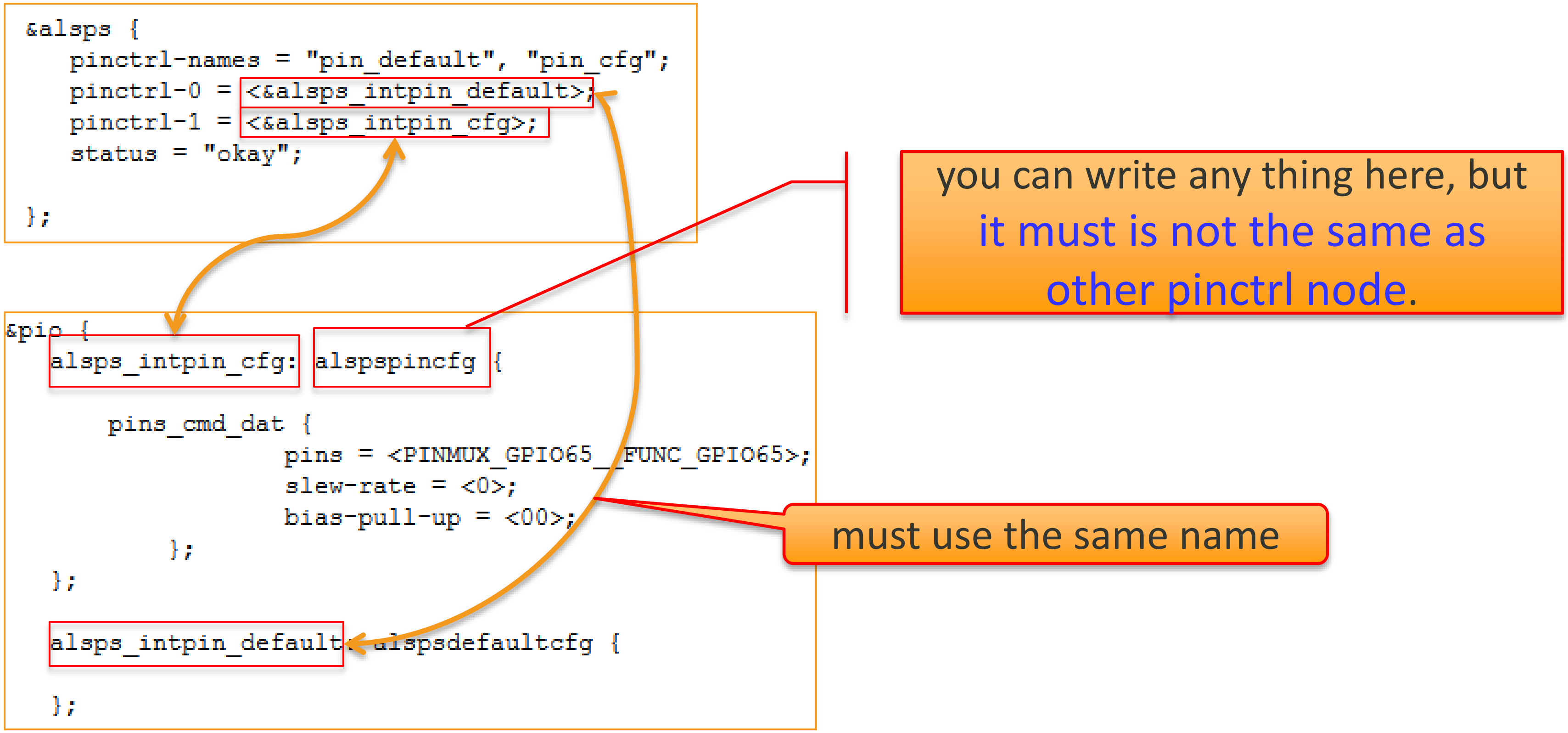
Name must is the same

```
&alsps {
    pinctrl-names = "pin_default", "pin_cfg";
    pinctrl-0 = <&alsps_intpin_default>;
    pinctrl-1 = <&alsps_intpin_cfg>;
    status = "okay";

};
```

Pinctrl Usage sample

alsps use GPIO pinctrl

- DTS format detail



```
&alsps {
    pinctrl-names = "pin_default", "pin_cfg";
    pinctrl-0 = <&alsps_intpin_default>;
    pinctrl-1 = <&alsps_intpin_cfg>;
    status = "okay";

};
```

```
&pio {
    alsps_intpin_cfg: alspspincfg {

        pins_cmd_dat {
                    pins = <PINMUX_GPIO65__FUNC_GPIO65>;
                    slew-rate = <0>;
                    bias-pull-up = <00>;
            };
    };

    alsps_intpin_default: alspsdefaultcfg {

    };
```

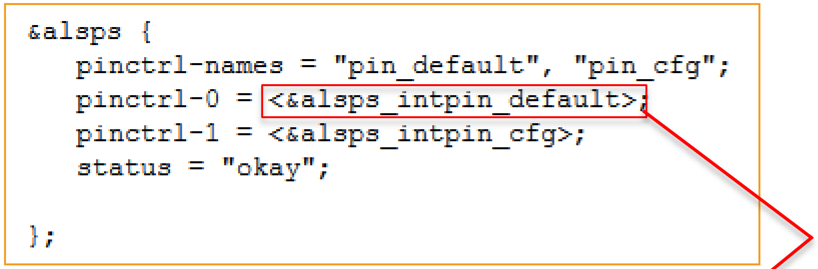
you can write any thing here, but it must is not the same as other pinctrl node.

must use the same name

Pinctrl Usage sample

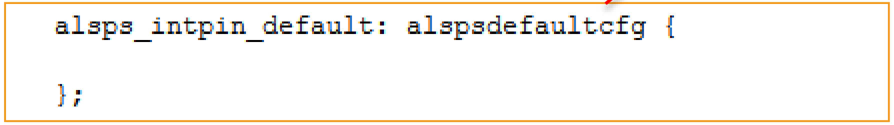
alsps use GPIO pinctrl

- DTS format detail



```
&alsps {
   pinctrl-names = "pin_default", "pin_cfg";
   pinctrl-0 = <&alsps_intpin_default>;
   pinctrl-1 = <&alsps_intpin_cfg>;
   status = "okay";

};
```

you can configure nothing in default ,but must write default in dts

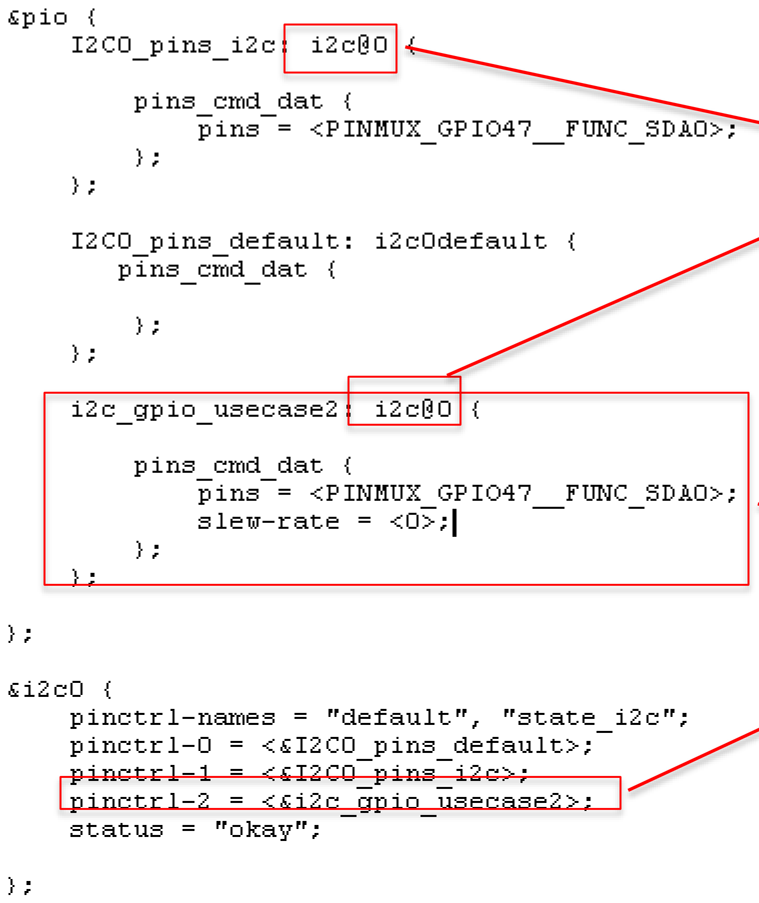


```
  alsps_intpin_default: alspsdefaultcfg {

  };
```

Pinctrl Usage sample

alsps use GPIO pinctrl

- DTS format detail



```
&pio {
    I2C0_pins_i2c: i2c@0 {

        pins_cmd_dat {
            pins = <PINMUX_GPIO47__FUNC_SDA0>;
        };
    };

    I2C0_pins_default: i2c0default {
       pins_cmd_dat {

        };
    };

    i2c_gpio_usecase2: i2c@0 {

        pins_cmd_dat {
            pins = <PINMUX_GPIO47__FUNC_SDA0>;
            slew-rate = <0>;
        };
    };

};

&i2c0 {
    pinctrl-names = "default", "state_i2c";
    pinctrl-0 = <&I2C0_pins_default>;
    pinctrl-1 = <&I2C0_pins_i2c>;
    pinctrl-2 = <&i2c_gpio_usecase2>;
    status = "okay";

};
```

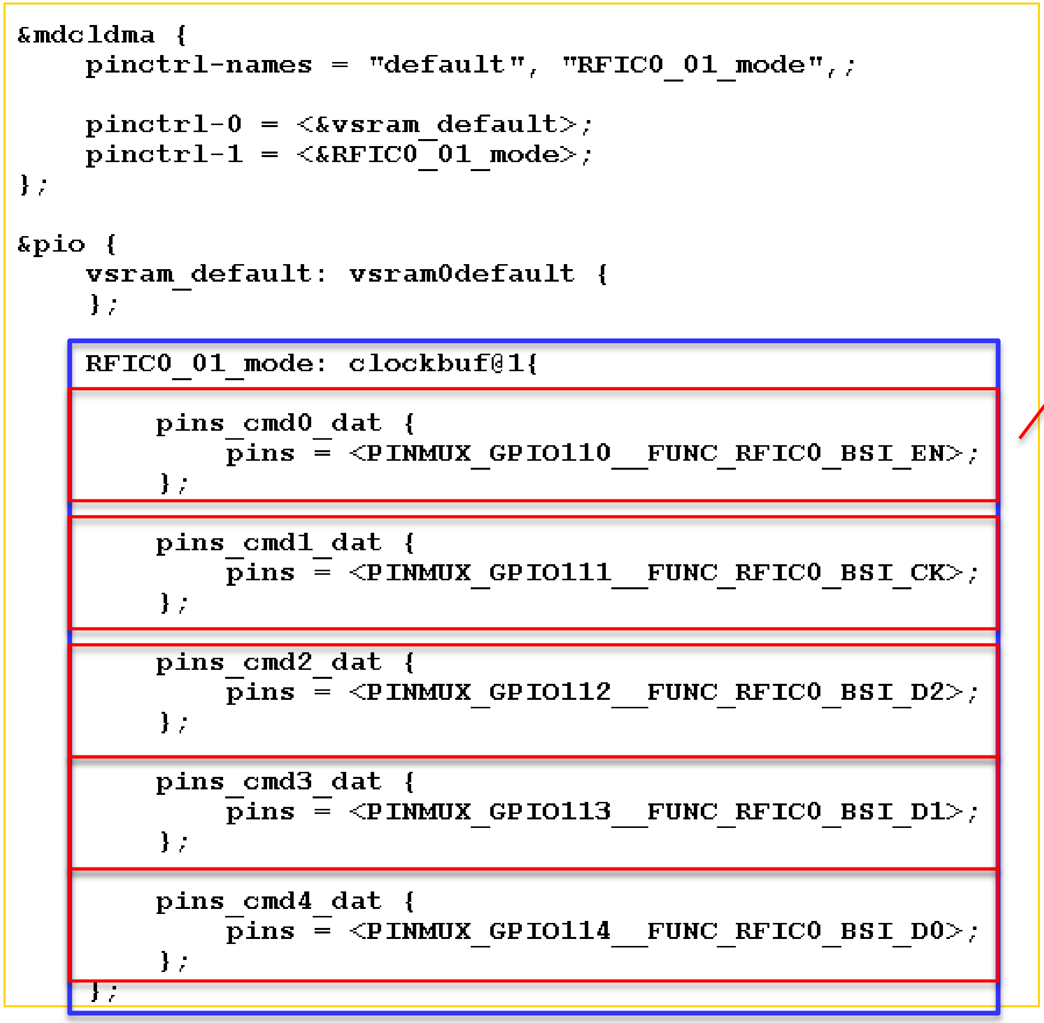
ERROR : you can not use the same name

you can add many configs you wanted to add

Pinctrl Usage sample

alsps use GPIO pinctrl

- DTS format detail

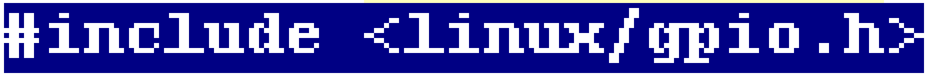


```
&mdcldma {
    pinctrl-names = "default", "RFIC0_01_mode",;

    pinctrl-0 = <&vsram_default>;
    pinctrl-1 = <&RFIC0_01_mode>;
};

&pio {
    vsram_default: vsram0default {
    };

    RFIC0_01_mode: clockbuf@1{

        pins_cmd0_dat {
            pins = <PINMUX_GPIO110__FUNC_RFIC0_BSI_EN>;
        };

        pins_cmd1_dat {
            pins = <PINMUX_GPIO111__FUNC_RFIC0_BSI_CK>;
        };

        pins_cmd2_dat {
            pins = <PINMUX_GPIO112__FUNC_RFIC0_BSI_D2>;
        };

        pins_cmd3_dat {
            pins = <PINMUX_GPIO113__FUNC_RFIC0_BSI_D1>;
        };

        pins_cmd4_dat {
            pins = <PINMUX_GPIO114__FUNC_RFIC0_BSI_D0>;
        };
    };
```

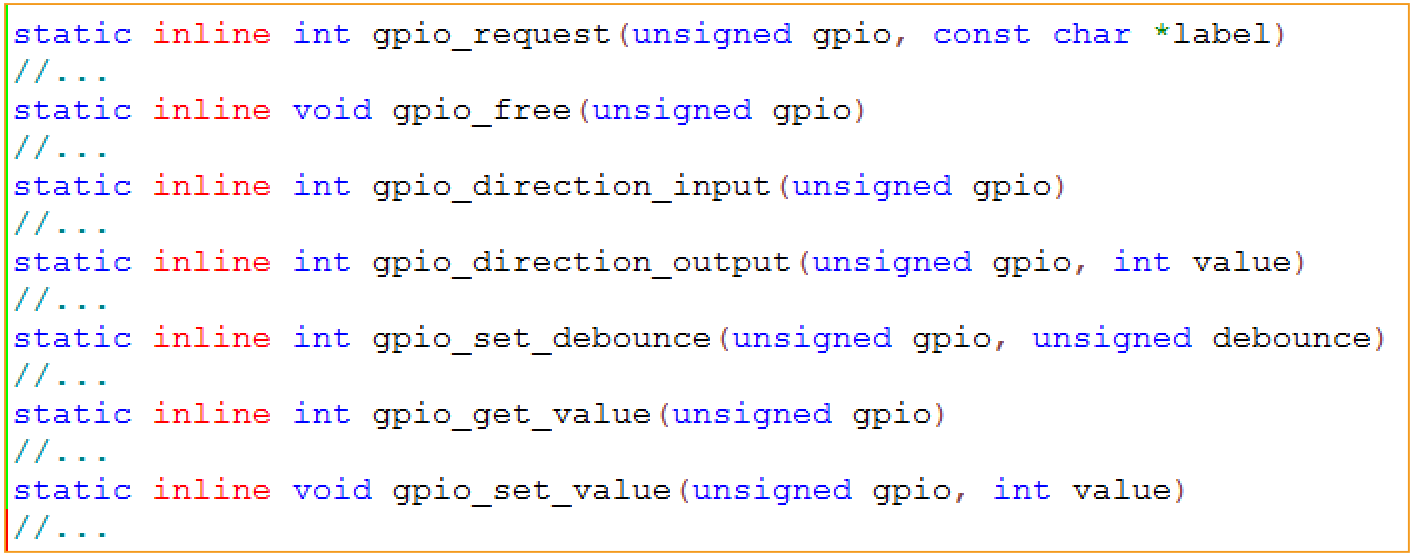
If you want configure multi pins at once, you can write the dts like this.

GPIOlib API usage

- Head file
  - If you want use linux gpio API, you must use the head file: linux/gpio.h



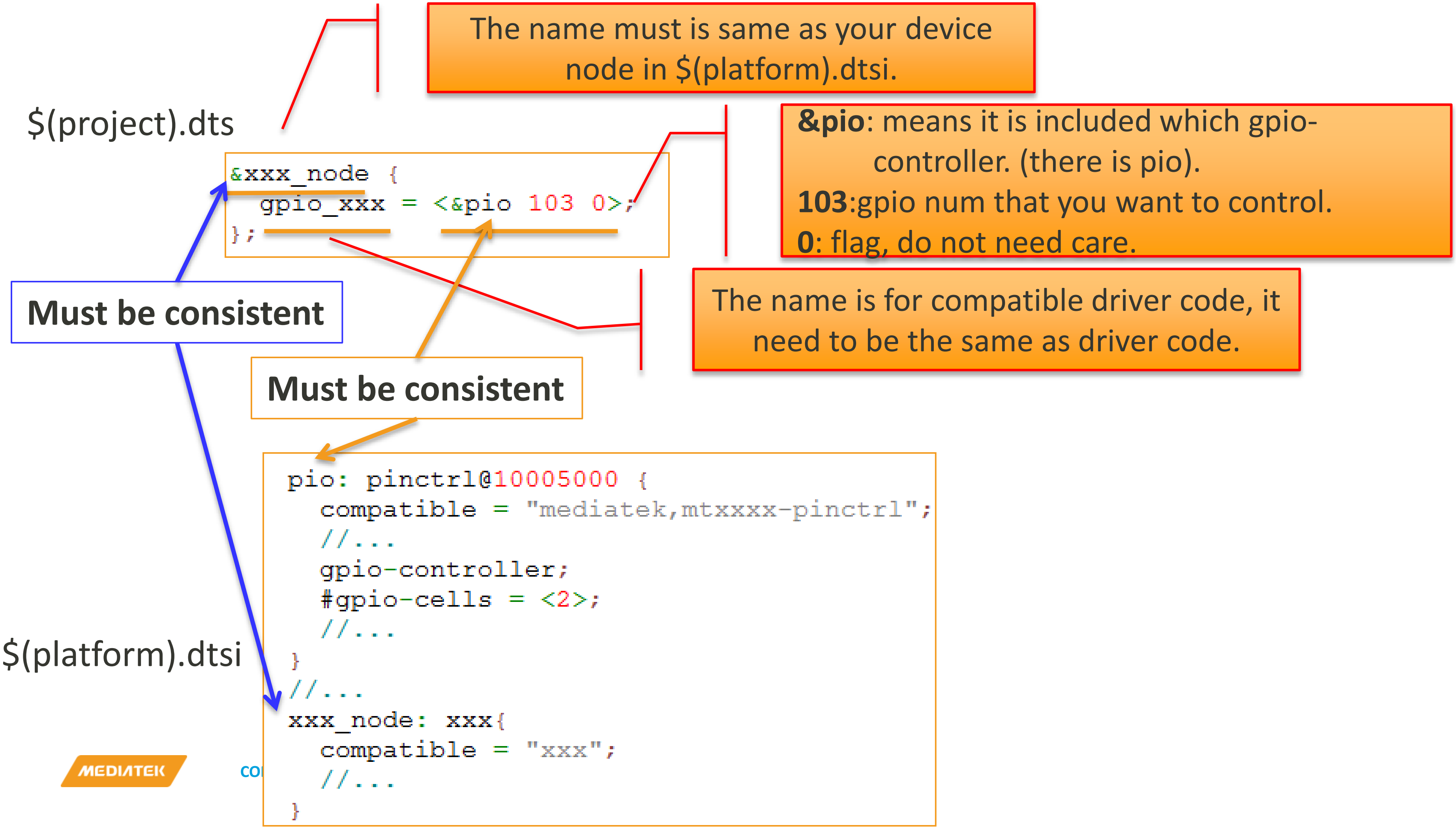
```
#include <linux/gpio.h>
```



```
static inline int gpio_request(unsigned gpio, const char *label)
//...
static inline void gpio_free(unsigned gpio)
//...
static inline int gpio_direction_input(unsigned gpio)
//...
static inline int gpio_direction_output(unsigned gpio, int value)
//...
static inline int gpio_set_debounce(unsigned gpio, unsigned debounce)
//...
static inline int gpio_get_value(unsigned gpio)
//...
static inline void gpio_set_value(unsigned gpio, int value)
//...
```

GPIOlib API usage

- For example: Get the pin state
  - Step 1: configure the pin num that you want to control in $(project).dts



$(project).dts

```
&xxx_node {
  gpio_xxx = <&pio 103 0>;
};
```

The name must is same as your device node in $(platform).dtsi.

&pio: means it is included which gpio-controller. (there is pio).
103:gpio num that you want to control.
0: flag, do not need care.

The name is for compatible driver code, it need to be the same as driver code.

Must be consistent

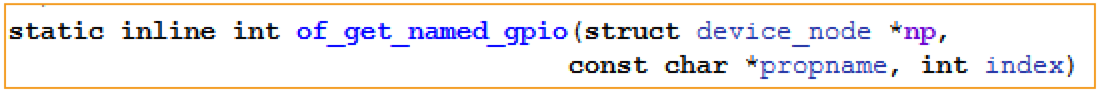
Must be consistent

$(platform).dtsi

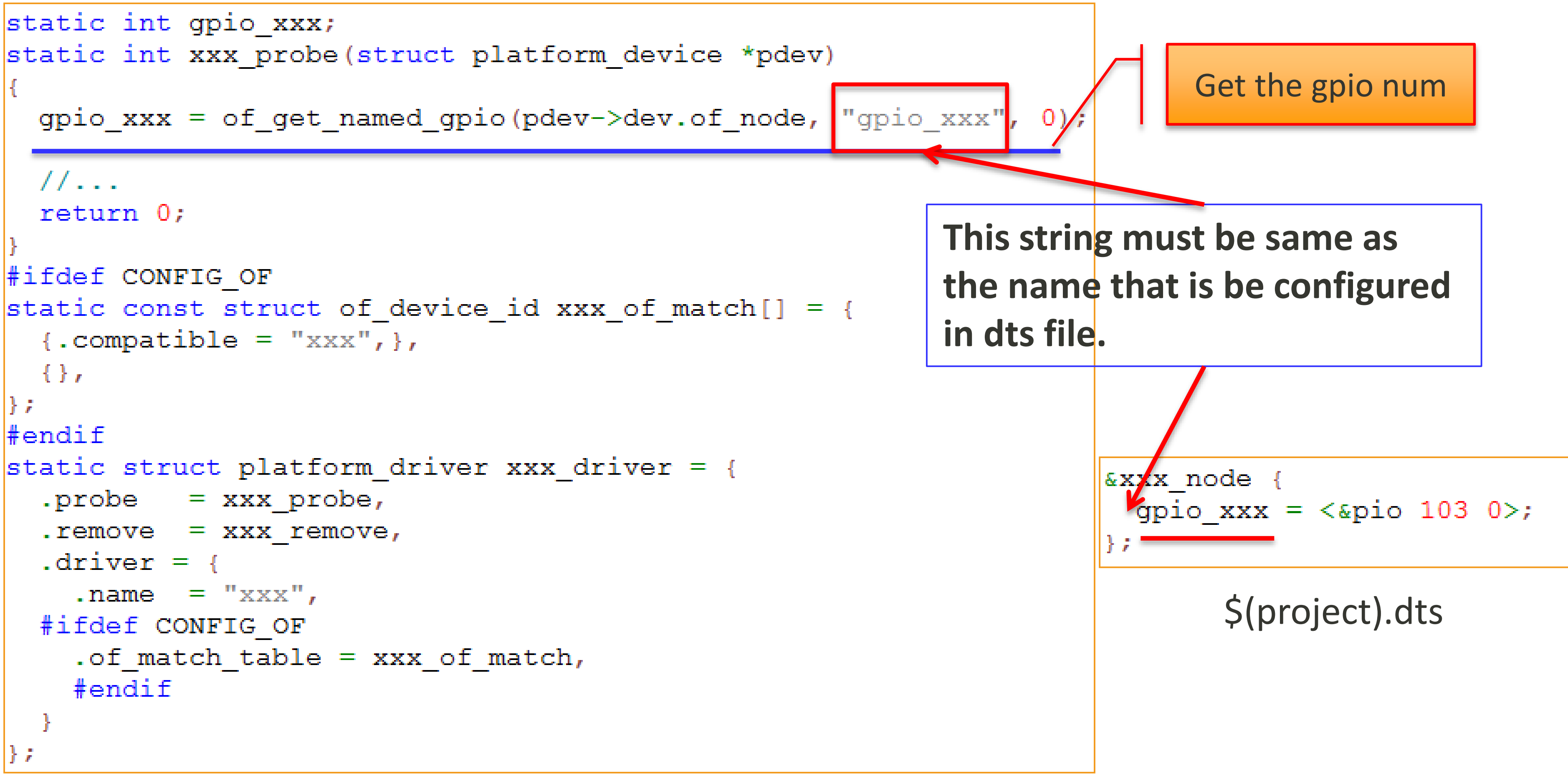
```
pio: pinctrl@10005000 {
  compatible = "mediatek,mtxxxx-pinctrl";
  //...
  gpio-controller;
  #gpio-cells = <2>;
  //...
}
//...
xxx_node: xxx{
  compatible = "xxx";
  //...
}
```

GPIOlib API usage

- For example: Get the pin state
  - Step 2: find the gpio num that you configure in driver code. Use the API:



```
static inline int of_get_named_gpio(struct device_node *np,
                                    const char *propname, int index)
```



```
static int gpio_xxx;
static int xxx_probe(struct platform_device *pdev)
{
  gpio_xxx = of_get_named_gpio(pdev->dev.of_node, "gpio_xxx", 0);

  //...
  return 0;
}
#ifdef CONFIG_OF
static const struct of_device_id xxx_of_match[] = {
  {.compatible = "xxx",},
  {},
};
#endif
static struct platform_driver xxx_driver = {
  .probe    = xxx_probe,
  .remove   = xxx_remove,
  .driver = {
    .name   = "xxx",
  #ifdef CONFIG_OF
    .of_match_table = xxx_of_match,
    #endif
  }
};
```

Get the gpio num

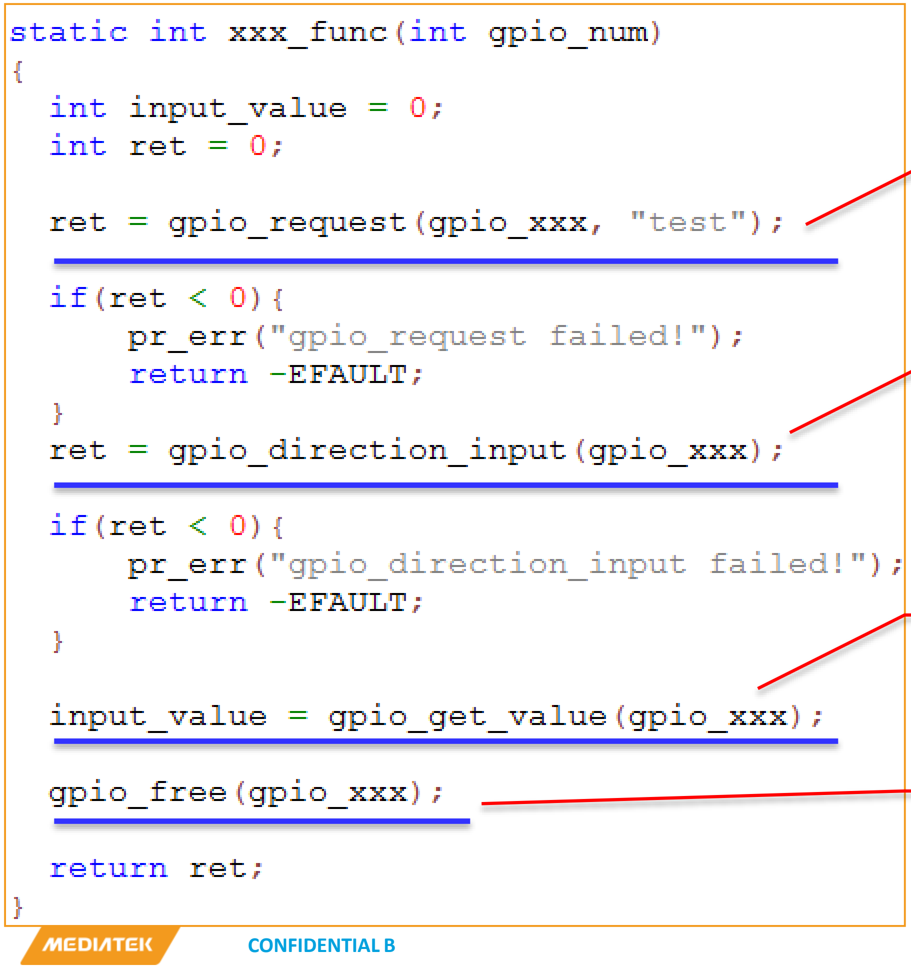
This string must be same as the name that is be configured in dts file.

```
&xxx_node {
  gpio_xxx = <&pio 103 0>;
};
```

$(project).dts

GPIOlib API usage

- For example: Get the pin state
  - Step 3: use the gpiolib API to control the pin:



```
static int xxx_func(int gpio_num)
{
   int input_value = 0;
   int ret = 0;

   ret = gpio_request(gpio_xxx, "test");

   if(ret < 0){
       pr_err("gpio_request failed!");
       return -EFAULT;
   }
   ret = gpio_direction_input(gpio_xxx);

   if(ret < 0){
       pr_err("gpio_direction_input failed!");
       return -EFAULT;
   }

   input_value = gpio_get_value(gpio_xxx);

   gpio_free(gpio_xxx);

   return ret;
}
```

1. Request this gpio.
You must request gpio before use it.

2. Set this gpio direction is input.

3. Get this gpio input value.

4. Free this gpio,
when you do not use it.

Check

- Check pinctrl node
  - You can check the pinctrl node with decompile the dtb file.
  - dtb file path:

    alps\out\target\product\$(proj)\obj\KERNEL_OBJ\arch\arm(xx)\boot\dts\mediatek\$(proj).dtb
  - dtc(decompile tool) path:

    alps\out\target\product\$(proj)\obj\KERNEL_OBJ\scripts\dtc\dtc
  - Command format:

    ```
    dtc –I dtb –O dts –o <dts_file> <dtb_file>
    ```
  - Example:

    ```
    ./dtc –I dtb –O dts –o k35v1_64_op01.dts k35v1_64_op01.dtb
    ```

Check

- Check pin state
  - You can check the pin state with adb command:

```
adb shell “cat /sys/devices/platform/XX.pinctrl/mt_gpio”
```

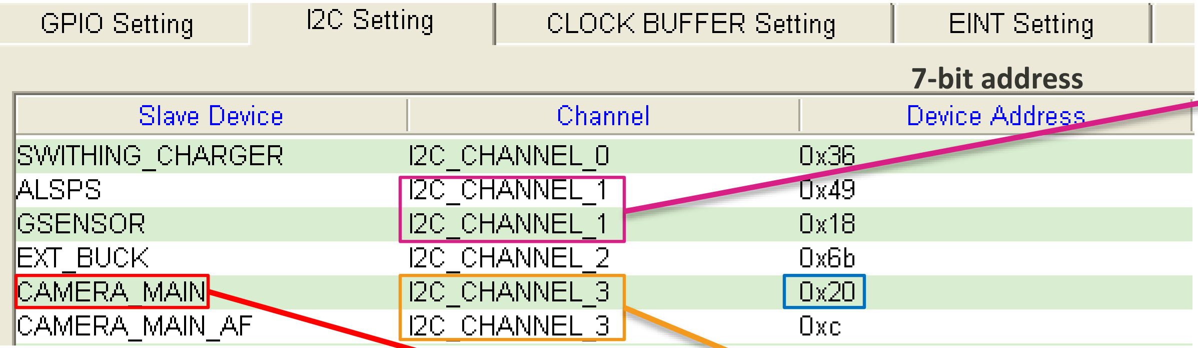
I2C

Outline

- cust_i2c.dtsi
- I2C device driver Modification

cust_i2c.dtsi

- Generation of cust_i2c.dtsi
  - Configure the device’s i2c info in codegen.dws – “I2C Setting”.
  - cust_i2c.dtsi will be generated after build/GenCode:



7-bit address

Slave Device	Channel	Device Address
SWITHING_CHARGER	I2C_CHANNEL_0	0x36
ALSPS	I2C_CHANNEL_1	0x49
GSENSOR	I2C_CHANNEL_1	0x18
EXT_BUCK	I2C_CHANNEL_2	0x6b
CAMERA_MAIN	I2C_CHANNEL_3	0x20
CAMERA_MAIN_AF	I2C_CHANNEL_3	0xc

=>

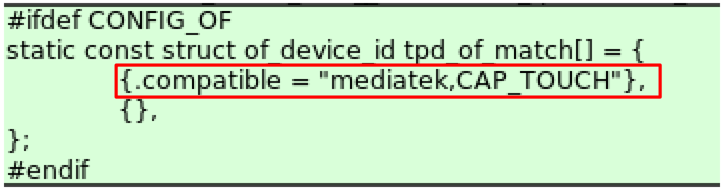
cust_i2c.dtsi

```
&i2c0 {
        SWITHING_CHARGER@36 {
                compatible = "mediatek,SWITHING_CHARGER";
                reg = <0x36>;
        };
};

&i2c1 {
        ALSPS@49 {
                compatible = "mediatek,ALSPS";
                reg = <0x49>;
        };

        GSENSOR@18 {
                compatible = "mediatek,GSENSOR";
                reg = <0x18>;
        };
};

&i2c2 {
        EXT_BUCK@6b {
                compatible = "mediatek,EXT_BUCK";
                reg = <0x6b>;
        };
};

&i2c3 {
        CAMERA_MAIN@20 {
                compatible = "mediatek,CAMERA_MAIN";
                reg = <0x20>;
        };

        CAMERA_MAIN_AF@c {
                compatible = "mediatek,CAMERA_MAIN_AF";
                reg = <0xc>;
        };
};
```

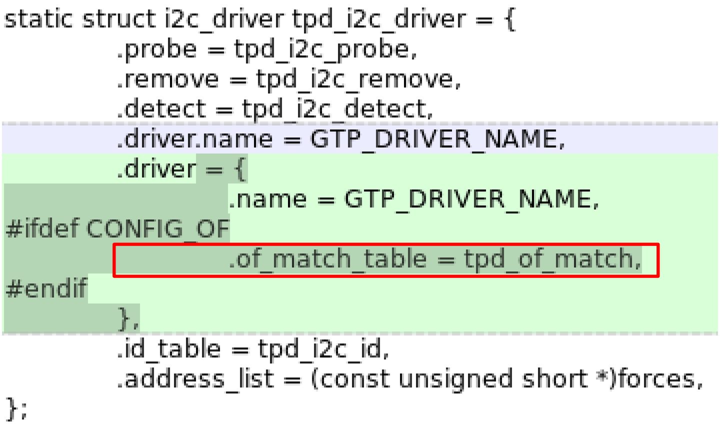
I2C device driver Modification (1/2)

- **Add of_match_id table** in driver code
  - Compatible name must be the same with that defined in codegen.dws and cust_i2c.dtsi

Example (gt1x_tpd.c):

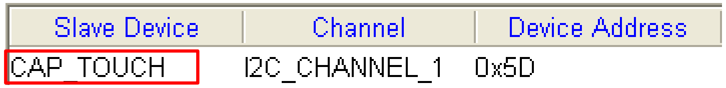


```
#ifdef CONFIG_OF
static const struct of_device_id tpd_of_match[] = {
            {.compatible = "mediatek,CAP_TOUCH"},
            {},
};
#endif
```



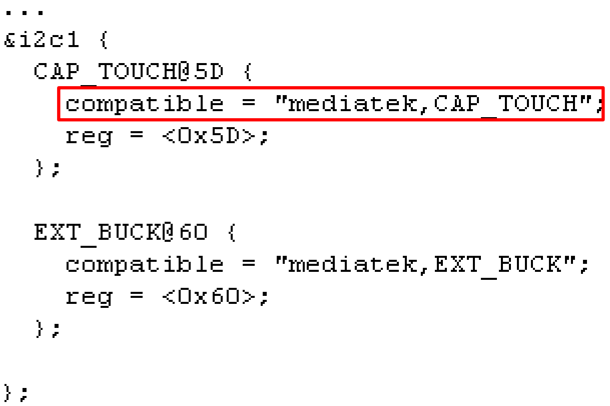
```
static struct i2c_driver tpd_i2c_driver = {
            .probe = tpd_i2c_probe,
            .remove = tpd_i2c_remove,
            .detect = tpd_i2c_detect,
            .driver.name = GTP_DRIVER_NAME,
            .driver = {
                        .name = GTP_DRIVER_NAME,
#ifdef CONFIG_OF
                        .of_match_table = tpd_of_match,
#endif
            },
            .id_table = tpd_i2c_id,
            .address_list = (const unsigned short *)forces,
};
```

codegen.dws



Slave Device	Channel	Device Address
CAP_TOUCH	I2C_CHANNEL_1	0x5D

cust_i2c.dtsi

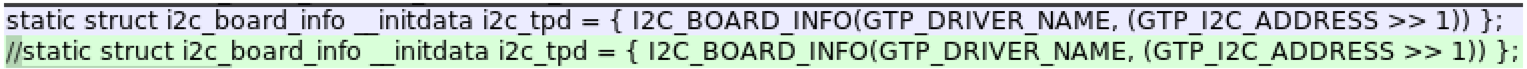


```
...
&i2c1 {
  CAP_TOUCH@5D {
    compatible = "mediatek,CAP_TOUCH";
    reg = <0x5D>;
  };

  EXT_BUCK@60 {
    compatible = "mediatek,EXT_BUCK";
    reg = <0x60>;
  };

};
```

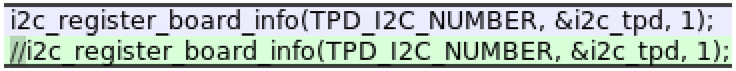
I2C device driver Modification (1/2)

- **Remove** the usage of *#include <cust_i2c.h>*
- **Remove** *i2c_register_board_info()*
- **Remove** *i2c_board_info* information

Example (gt1x_tpd.c):



```
static struct i2c_board_info __initdata i2c_tpd = { I2C_BOARD_INFO(GTP_DRIVER_NAME, (GTP_I2C_ADDRESS >> 1)) };
//static struct i2c_board_info __initdata i2c_tpd = { I2C_BOARD_INFO(GTP_DRIVER_NAME, (GTP_I2C_ADDRESS >> 1)) };
```



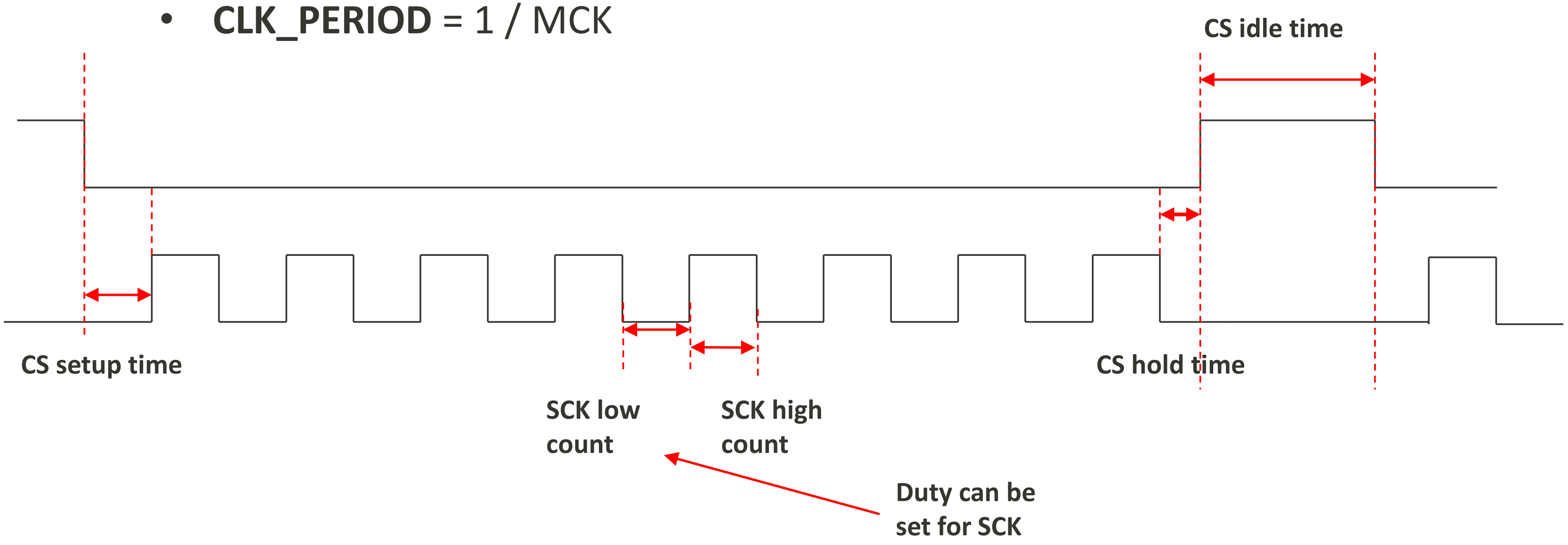
```
i2c_register_board_info(TPD_I2C_NUMBER, &i2c_tpd, 1);
//i2c_register_board_info(TPD_I2C_NUMBER, &i2c_tpd, 1);
```

CONFIDENTIAL B

SPI

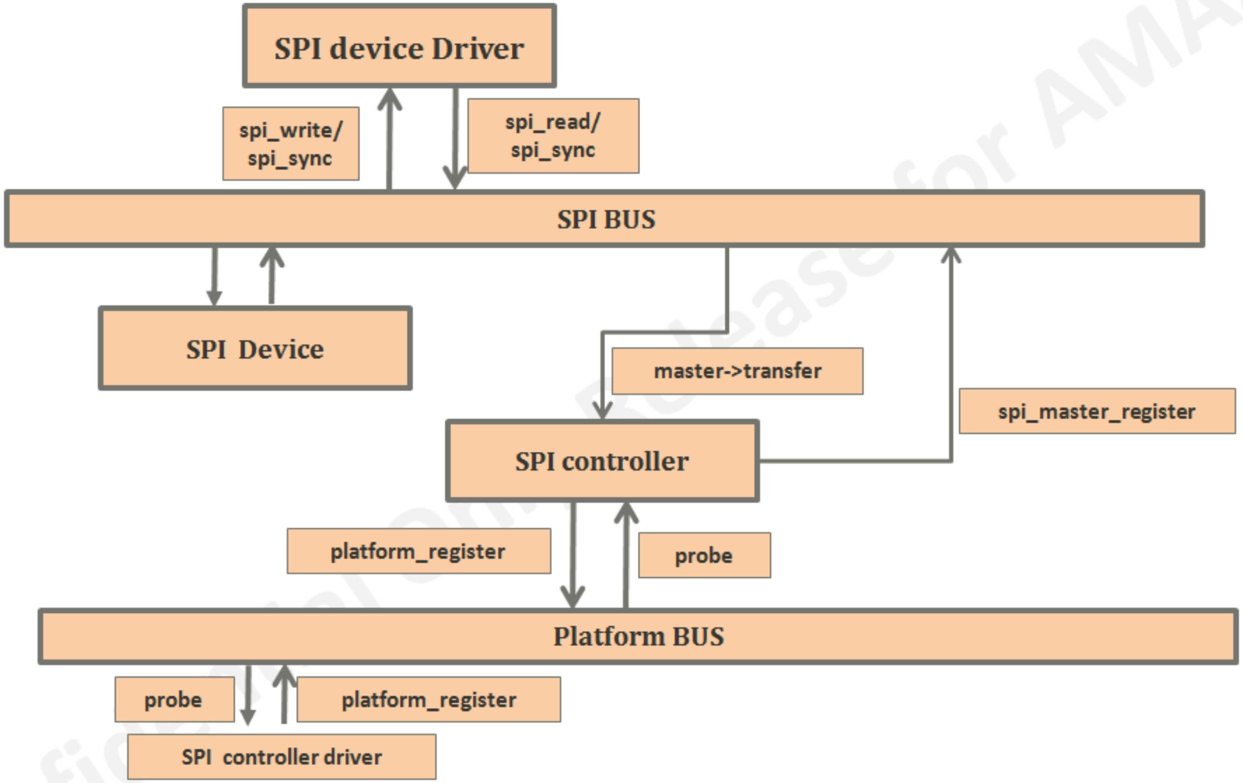
HW Introduce

- Hardware SPEC -- Timing
  - Configurable CS(chip select) idle time, CS setup hold and idle time
  - SCK high time and low time can be programmed
  - The SCK(SPI serial clock) frequency is decided by MCK(SPI master clock)
    - MCK/(SCK_LOW_COUNT + SCK_HIGH_COUNT)
  - **CLK_PERIOD** = 1 / MCK

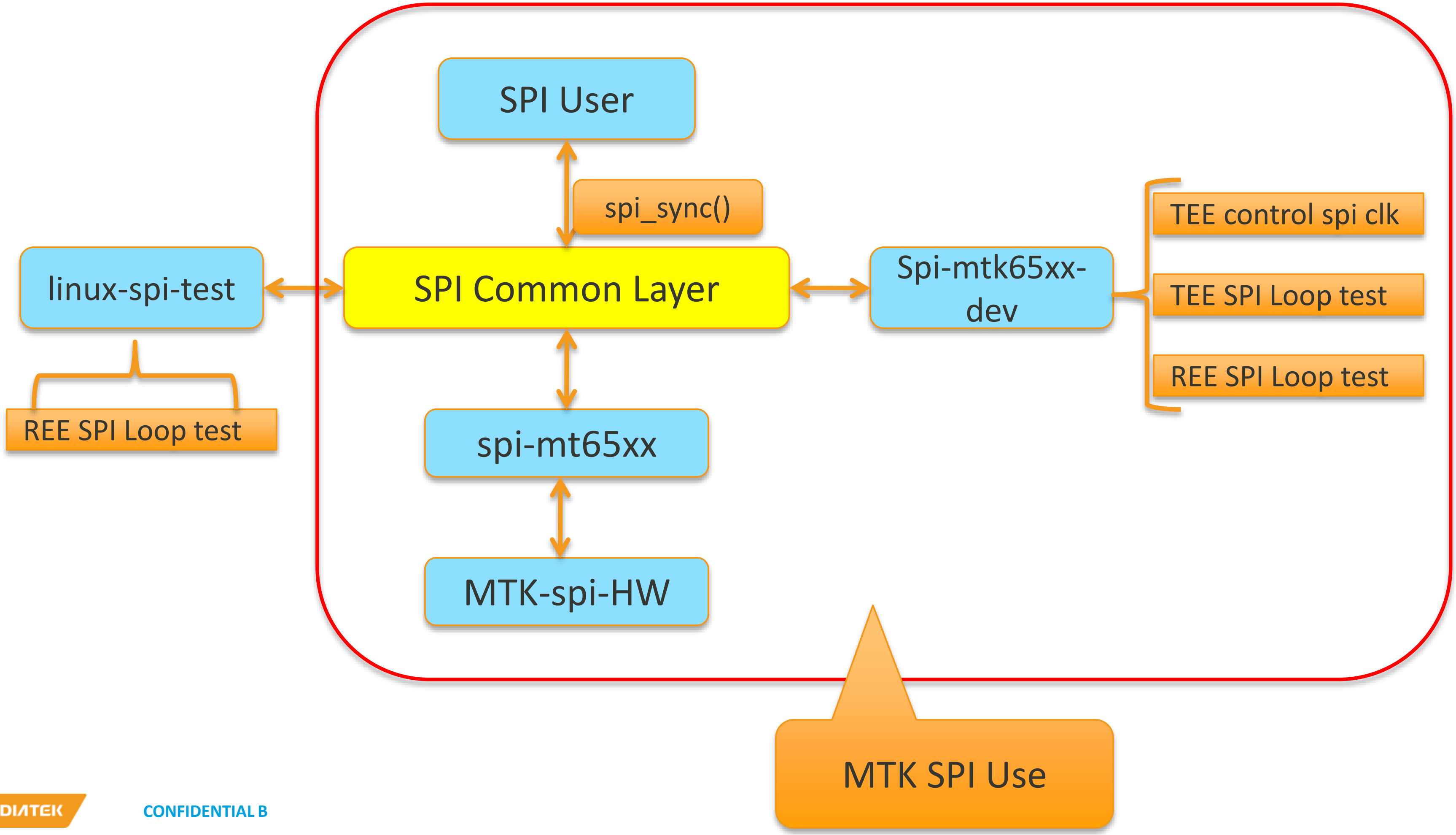


Driver – Control follow

- Description

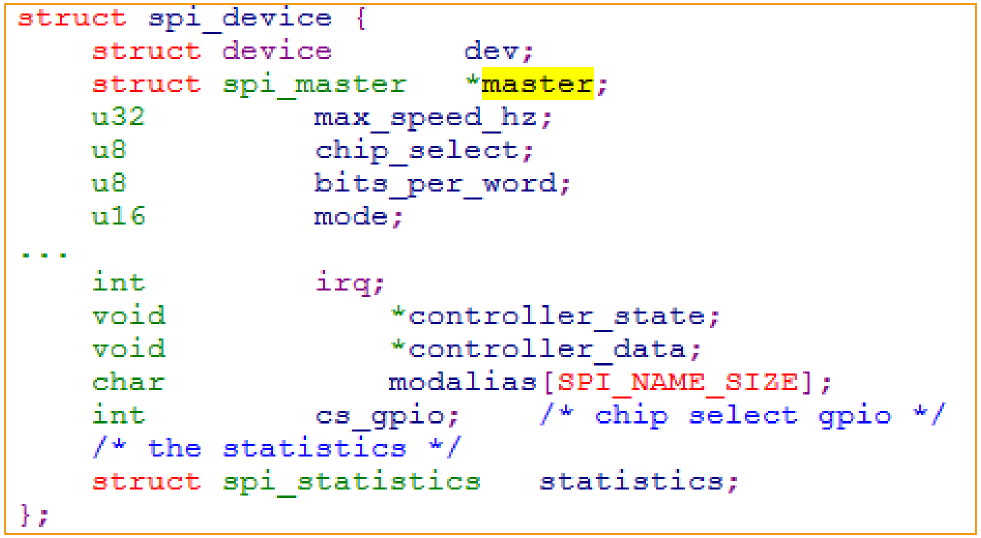


Driver – Control follow



Driver – Data Structure

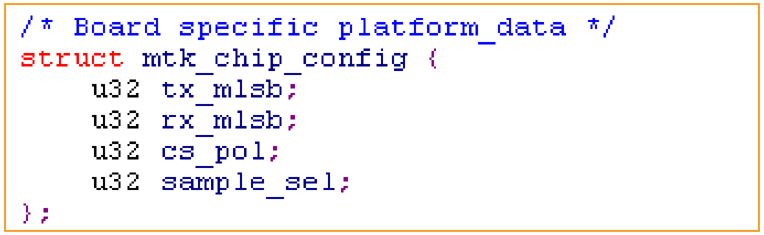
- Spi_device: Used to descript a slave device—defined in spi.h
  - max_speed_hz: it can be defined in dts, spi_transfer.speed_hz can override this, the max speed the device use.
  - mode : The spi mode defines how data is clocked out and in.
  - bits_per_word
  - controller_data: this struct will be descript in next page.



```
struct spi_device {
    struct device           dev;
    struct spi_master       *master;
    u32             max_speed_hz;
    u8              chip_select;
    u8              bits_per_word;
    u16             mode;
...
    int             irq;
    void                *controller_state;
    void                *controller_data;
    char                modalias[SPI_NAME_SIZE];
    int             cs_gpio;    /* chip select gpio */
    /* the statistics */
    struct spi_statistics   statistics;
};
```

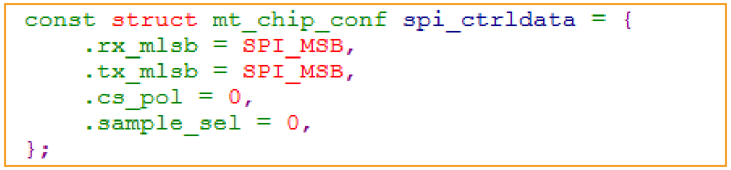
Driver – Data Structure

- mtk_chip_config – defined in spi-mt65xx.h
  - tx_mlsb & rx_mlsb : data transfer type
  - cs_pol : 0 cs active low, 1 cs active high
  - sample_sel: the control bit of chip select polarity. 0 is active low. 1 is active high

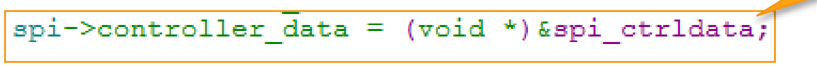


```
/* Board specific platform_data */
struct mtk_chip_config {
    u32 tx_mlsb;
    u32 rx_mlsb;
    u32 cs_pol;
    u32 sample_sel;
};
```

  - Example: device can set mt_chip_conf by itself, otherwise it will use default setting in spi-mt65xx.c

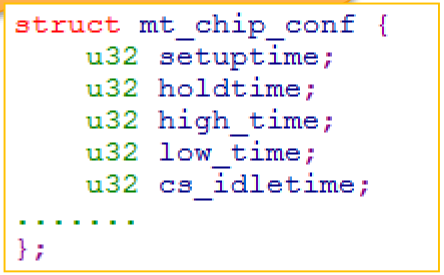


```
const struct mt_chip_conf spi_ctrldata = {
    .rx_mlsb = SPI_MSB,
    .tx_mlsb = SPI_MSB,
    .cs_pol = 0,
    .sample_sel = 0,
};
```



```
spi->controller_data = (void *)&spi_ctrldata;
```

Do not use struct mt_chip_conf{} any more

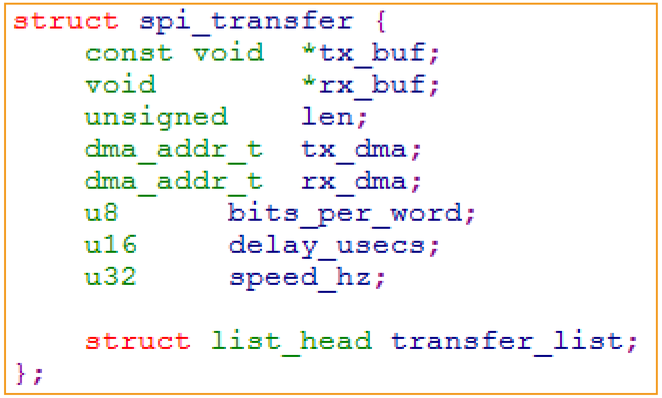


```
struct mt_chip_conf {
    u32 setuptime;
    u32 holdtime;
    u32 high_time;
    u32 low_time;
    u32 cs_idletime;
.......
};
```

Driver – Data Structure

- spi_transfer : a read/write buffer pair—defined in spi.h
  - tx_buf : data to be written.
  - rx_buf: data to be read.
  - len : size of rx or tx buffers, rx buf len

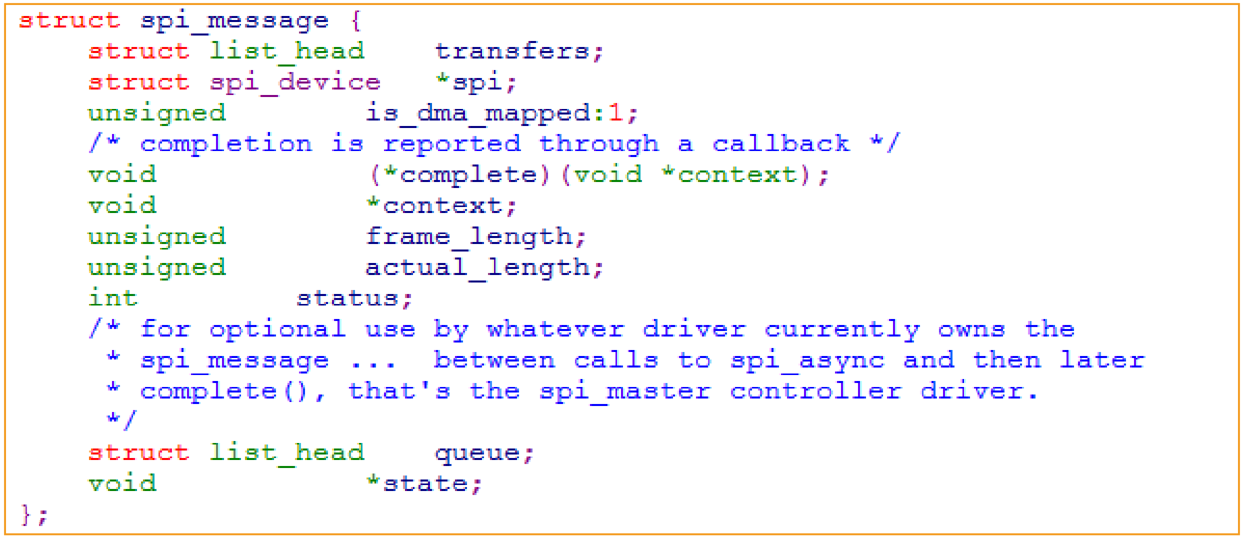
    and tx_buf need to be same.
  - tx_dma: DMA address of tx_buf.
  - rx_dma: DMA address of rx_buf.
    - @spi_message.is_dma_mapped.
  - bits_per_word: select a bits_per_word other than the device default.
  - delay_usecs: microseconds delay between two transfers.
  - speed_hz: Select a speed other than the device default for this transfer, if you want to use 500kzh, you need set speed_hz = 500000.



```
struct spi_transfer {
    const void  *tx_buf;
    void        *rx_buf;
    unsigned    len;
    dma_addr_t  tx_dma;
    dma_addr_t  rx_dma;
    u8      bits_per_word;
    u16     delay_usecs;
    u32     speed_hz;

    struct list_head transfer_list;
};
```

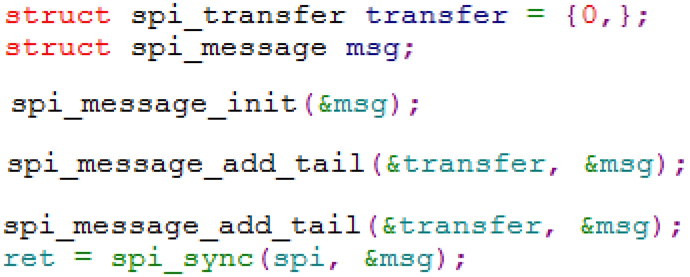
Driver – Data Structure

- spi_message—defined in spi.h



```
struct spi_message {
    struct list_head    transfers;
    struct spi_device   *spi;
    unsigned            is_dma_mapped:1;
    /* completion is reported through a callback */
    void                (*complete)(void *context);
    void                *context;
    unsigned            frame_length;
    unsigned            actual_length;
    int             status;
    /* for optional use by whatever driver currently owns the
     * spi_message ...  between calls to spi_async and then later
     * complete(), that's the spi_master controller driver.
     */
    struct list_head    queue;
    void                *state;
};
```

- Example



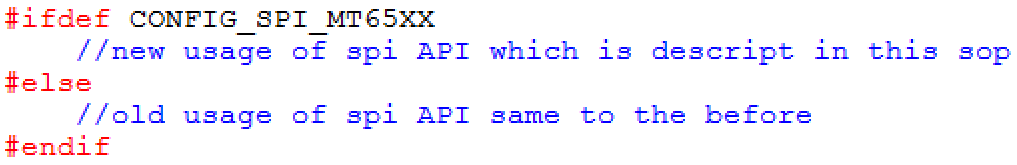
```
struct spi_transfer transfer = {0,};
struct spi_message msg;

spi_message_init(&msg);

spi_message_add_tail(&transfer, &msg);

spi_message_add_tail(&transfer, &msg);
ret = spi_sync(spi, &msg);
```

Driver – sync and async

- spi_sync()
  - thread sleep untill transfer finished
- spi_async()
  - when spi controller finishs truansfer, callback function complete() is called in interrupt.

Usage Example

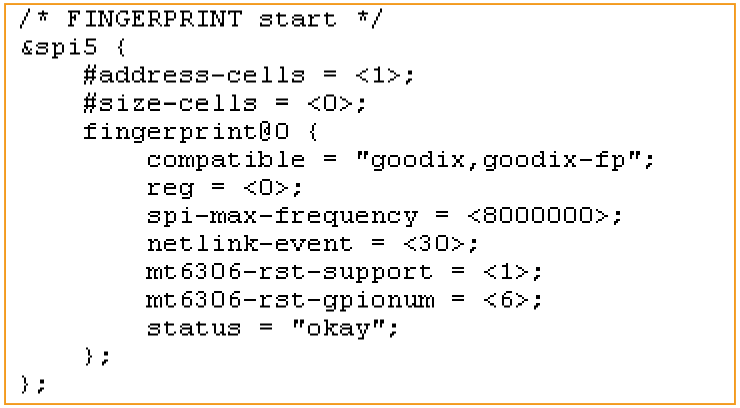
- Option
  - Please use CONFIG_SPI_MT65XX to indentify your usage of spi api in your common code.
  - Example



```
#ifdef CONFIG_SPI_MT65XX
    //new usage of spi API which is descript in this sop
#else
    //old usage of spi API same to the before
#endif
```

Usage Example

- Add device to spi bus
  - it defined in arch/arm64/boot/dts/mediatek/project.dts

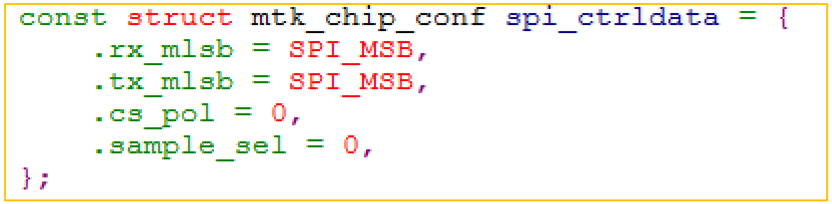


```
/* FINGERPRINT start */
&spi5 {
    #address-cells = <1>;
    #size-cells = <0>;
    fingerprint@0 {
        compatible = "goodix,goodix-fp";
        reg = <0>;
        spi-max-frequency = <8000000>;
        netlink-event = <30>;
        mt6306-rst-support = <1>;
        mt6306-rst-gpionum = <6>;
        status = "okay";
    };
};
```

  - Slave device dts node should be included in &spi node. In this way, fingerprint driver can be binding on spi master.

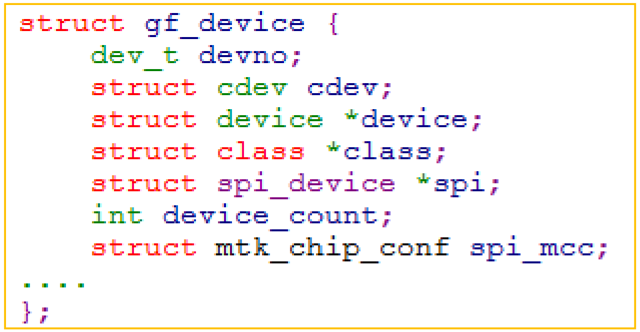
Usage Example

- Probe init
  - Set up spi_device & mtk_chip_config
  - You can set mtk_chip_config if you want, or you can use default defined in spi-mt65xx.c.



```
const struct mtk_chip_conf spi_ctrldata = {
    .rx_mlsb = SPI_MSB,
    .tx_mlsb = SPI_MSB,
    .cs_pol = 0,
    .sample_sel = 0,
};
```

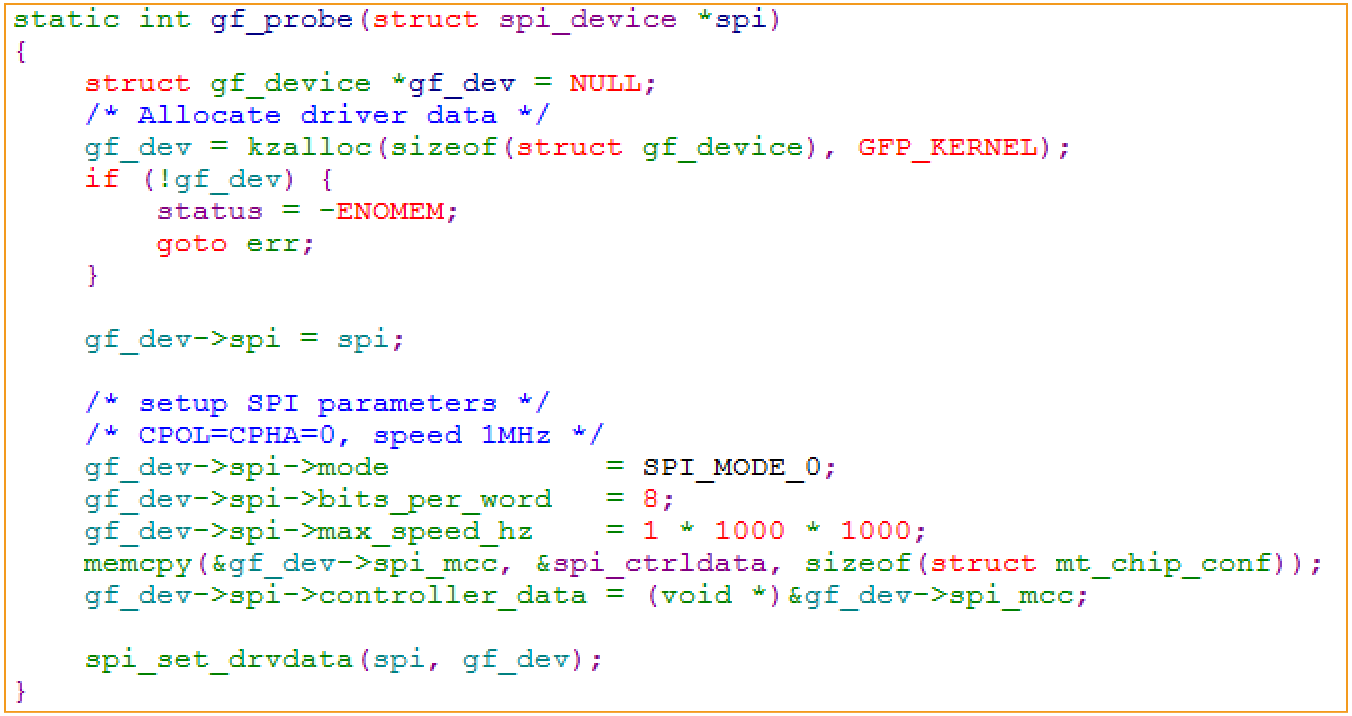
  - You can add spi_device to you driver like the follow way



```
struct gf_device {
    dev_t devno;
    struct cdev cdev;
    struct device *device;
    struct class *class;
    struct spi_device *spi;
    int device_count;
    struct mtk_chip_conf spi_mcc;
....
};
```

Usage Example

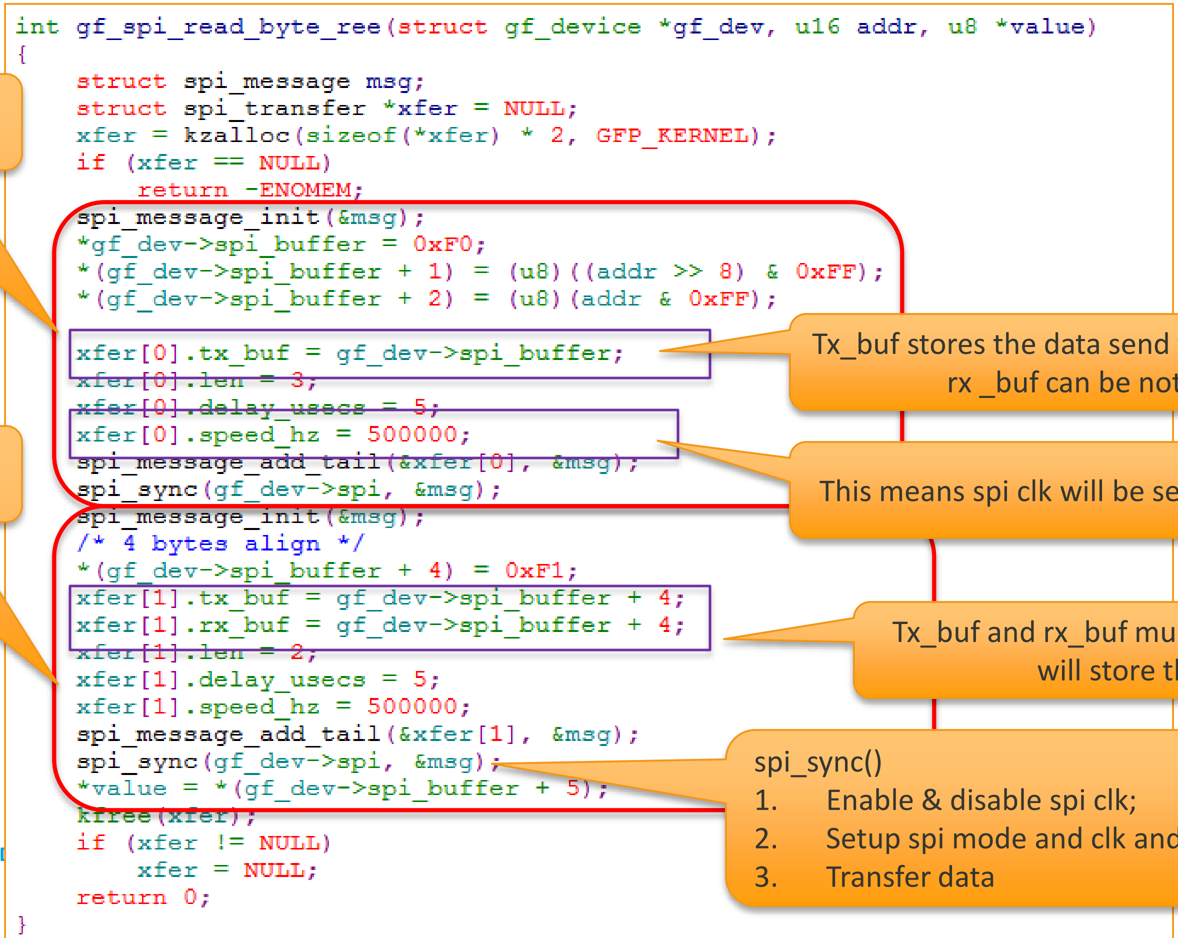
- Probe init
  - Set up spi_device & mtk_chip_config
  - You do not need to control spi clk and call spi_setup() any more. Please save the data in your device(Just like gf_device do), and when you call spi_sync() it will help to set all spi parameter to spi master.



```
static int gf_probe(struct spi_device *spi)
{
    struct gf_device *gf_dev = NULL;
    /* Allocate driver data */
    gf_dev = kzalloc(sizeof(struct gf_device), GFP_KERNEL);
    if (!gf_dev) {
        status = -ENOMEM;
        goto err;
    }

    gf_dev->spi = spi;

    /* setup SPI parameters */
    /* CPOL=CPHA=0, speed 1MHz */
    gf_dev->spi->mode            = SPI_MODE_0;
    gf_dev->spi->bits_per_word   = 8;
    gf_dev->spi->max_speed_hz    = 1 * 1000 * 1000;
    memcpy(&gf_dev->spi_mcc, &spi_ctrldata, sizeof(struct mt_chip_conf));
    gf_dev->spi->controller_data = (void *)&gf_dev->spi_mcc;

    spi_set_drvdata(spi, gf_dev);
}
```

Usage Example

- Use Spi to R.W data
  - In kernel side, slave device no need to enable clk before call spi_sync(), and use speed_zh to set spi clk rate



```
int gf_spi_read_byte_ree(struct gf_device *gf_dev, u16 addr, u8 *value)
{
    struct spi_message msg;
    struct spi_transfer *xfer = NULL;
    xfer = kzalloc(sizeof(*xfer) * 2, GFP_KERNEL);
    if (xfer == NULL)
        return -ENOMEM;
    spi_message_init(&msg);
    *gf_dev->spi_buffer = 0xF0;
    *(gf_dev->spi_buffer + 1) = (u8)((addr >> 8) & 0xFF);
    *(gf_dev->spi_buffer + 2) = (u8)(addr & 0xFF);

    xfer[0].tx_buf = gf_dev->spi_buffer;
    xfer[0].len = 3;
    xfer[0].delay_usecs = 5;
    xfer[0].speed_hz = 500000;
    spi_message_add_tail(&xfer[0], &msg);
    spi_sync(gf_dev->spi, &msg);
    spi_message_init(&msg);
    /* 4 bytes align */
    *(gf_dev->spi_buffer + 4) = 0xF1;
    xfer[1].tx_buf = gf_dev->spi_buffer + 4;
    xfer[1].rx_buf = gf_dev->spi_buffer + 4;
    xfer[1].len = 2;
    xfer[1].delay_usecs = 5;
    xfer[1].speed_hz = 500000;
    spi_message_add_tail(&xfer[1], &msg);
    spi_sync(gf_dev->spi, &msg);
    *value = *(gf_dev->spi_buffer + 5);
    kfree(xfer);
    if (xfer != NULL)
        xfer = NULL;
    return 0;
}
```

First, send read cmd to slave device

Tx_buf stores the data send to slave device, rx _buf can be not set.

This means spi clk will be set to 500000hz.

Second, use spi to read data from slave device

Tx_buf and rx_buf must same len, rx_buf will store the data.

spi_sync()
1. Enable & disable spi clk;
2. Setup spi mode and clk and other parameters;
3. Transfer data

Usage Example

- Use DMA or fifo?
  - It no need to specified use DMA or FIFO to transfer data the spi driver will auto to detect by len.
  - If len < 32byte, it will use fifo to transfer the data, otherwise it will use dma to send the data.

Test

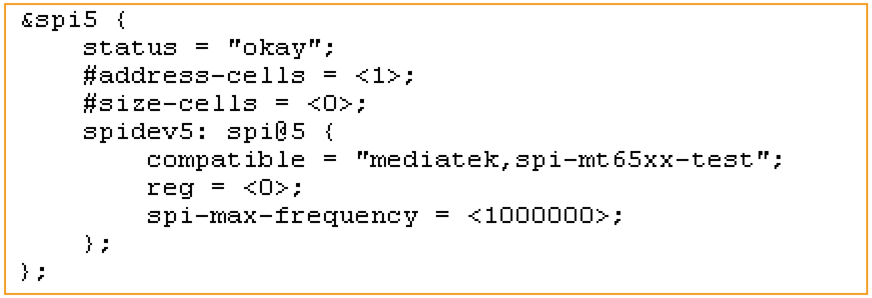
Do spi loopback to test spi hw/driver.

Test flow:

1. This is default enable.

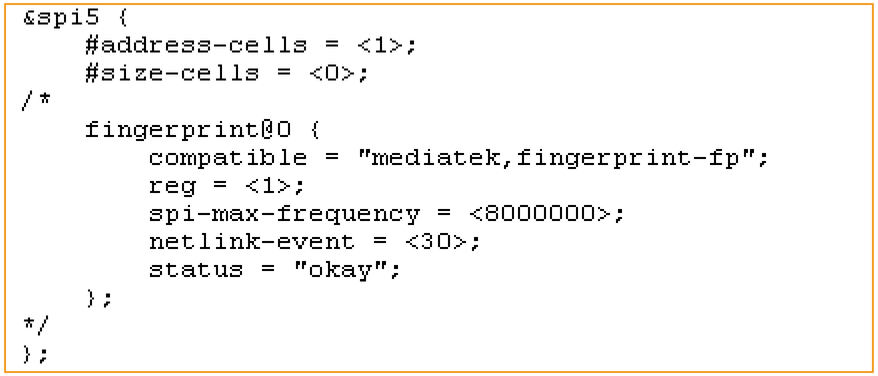
enable CONFIG_SPI_MT65XX=y

2. Add spid-mt65xx-test node to spi bus.



```
&spi5 {
    status = "okay";
    #address-cells = <1>;
    #size-cells = <0>;
    spidev5: spi@5 {
        compatible = "mediatek,spi-mt65xx-test";
        reg = <0>;
        spi-max-frequency = <1000000>;
    };
};
```

3. Unmark other spi5's slave device, for example, fingerprint is spi5's device.



```
&spi5 {
    #address-cells = <1>;
    #size-cells = <0>;
/*
    fingerprint@0 {
        compatible = "mediatek,fingerprint-fp";
        reg = <1>;
        spi-max-frequency = <8000000>;
        netlink-event = <30>;
        status = "okay";
    };
*/
};
```

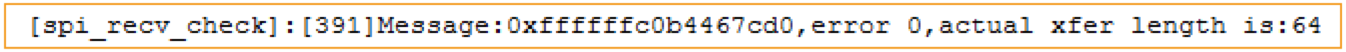
Test

4. Rebuild kernel and bootimage：
5. REE Loop Test: Go to debug node

   adb shell

   cd /sys/devices/platform/111d0000.spi5/spi_master/spi32759/spi32759.0
6. echo –func len=64 > spi_msg   (REE loop test use)

   The following log shows loop test success



```
[spi_recv_check]:[391]Message:0xffffffc0b4467cd0,error 0,actual xfer length is:64
```

7. TEE Loop test: First do the 1-5 step

   modify TEE spi

   \vendor\mediatek\proprietary\trustzone\trustonic\source\trustlet\spi\platform\mt6797\\Drspi\Locals\Code\drspi_Api.c 下，

   打开define macro，进行UT
   - FIFO_TEST
   - DMA_TEST

   adb cmd

   echo send5 >spi   (5 means spi5)

CONFIDENTIAL B

EINT

NO 1： 照常配置dws中的eint，对应会在cust_eint.dtsi生成如下节点，比如：

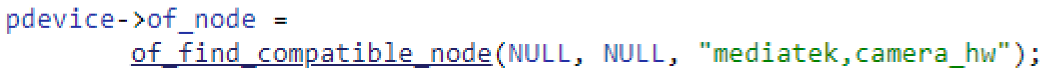
(新增node至mt6765.dtsi)

```
accdet： accdet {
        compatible = "mediatek, pmic-accdet";
};
```

(cust.dtsi)

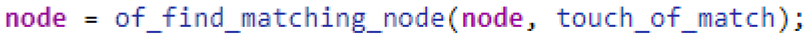
```
&accdet {
        interrupt-parent = <&pio>;
        interrupts = <7 IRQ_TYPE_LEVEL_LOW 7 0>;
        deb-gpios = <&pio 7 0>;
        debounce = <256000>;
        status = “okay”;
};
```

NO 2： 在各个模块的driver code中用of_find_compatible_node or of_find_matching_node去获取device tree中的信息



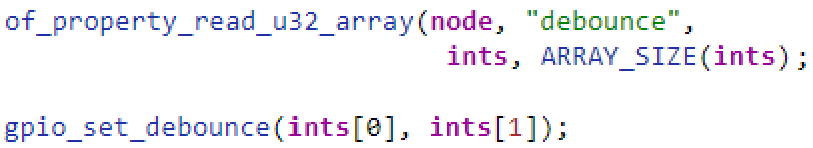
```
pdevice->of_node =
        of_find_compatible_node(NULL, NULL, "mediatek,camera_hw");
```

or



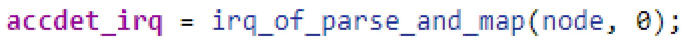
```
node = of_find_matching_node(node, touch_of_match);
```

NO3: 是否有获取当前DCT中设定的GPIO number和Debounce，并调用对应接口设定debounce time



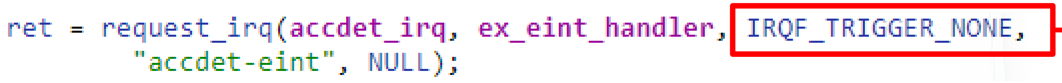
```
of_property_read_u32_array(node, "debounce",
                           ints, ARRAY_SIZE(ints));

gpio_set_debounce(ints[0], ints[1]);
```

NO 4： 是否有获取virtual irq number：



```
accdet_irq = irq_of_parse_and_map(node, 0);
```

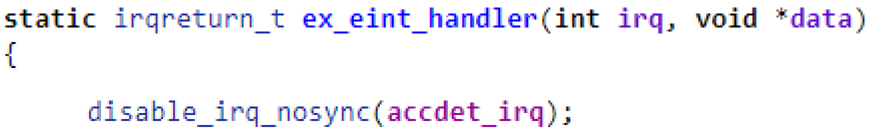
NO 5： 是否有调用request_irq注册中断



```
ret = request_irq(accdet_irq, ex_eint_handler, IRQF_TRIGGER_NONE,
        "accdet-eint", NULL);
```

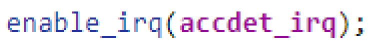
这里设置成none，dws中配置才会生效

NO6: implement non-autounmask EINT



```
static irqreturn_t ex_eint_handler(int irq, void *data)
{

      disable_irq_nosync(accdet_irq);
```

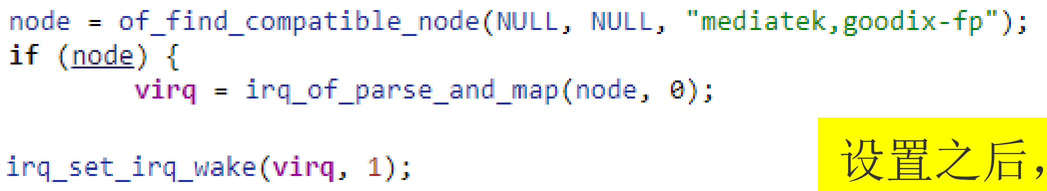
NO7: 使能中断



```
enable_irq(accdet_irq);
```

Attention please!!
Enable(enable_irq) 与 Disable(disable_irq) 必须成对出现，否则会出现下次无法正常Enable 或者Disable的问题

NO 8： 设置中断可以唤醒系统



```
node = of_find_compatible_node(NULL, NULL, "mediatek,goodix-fp");
if (node) {
        virq = irq_of_parse_and_map(node, 0);

irq_set_irq_wake(virq, 1);
```

设置之后，系统休眠之后，此中断的到来可以唤醒系统

Examples(1/2)

```
static int __init eint_example_init(void)
{
        struct device_node *node;

        int irq;
        u32 ints[2] = {0, 0};
        unsigned int gpiopin, debounce;
        /* get gpio pin & debounce time */

        /*
         * kernel standard uses pin control to setup gpio
         * This example doesn't include pin control part.
         */
        node = of_find_compatible_node(NULL, NULL, "mediatek,eint_example");
        if(node) {
                of_property_read_u32_array(node, "debounce", ints, ARRAY_SIZE(ints));
                gpiopin = ints[GPIOPIN];
                debounce = ints[DEBOUNCE];

                printk(KERN_ERR "%s, gpiopin=%d, debounce=%d microsecond\n",
                        __func__, gpiopin, debounce);

                /* get irq # */
                irq = irq_of_parse_and_map(node, 0);
                if(!irq) {
                        printk("can't irq_of_parse_and_map for abc!!\n");
                        return -EINVAL;
                }

                /* set debounce (optional) */
                gpio_set_debounce(gpiopin, debounce);
                /* request irq for eint (either way) */
                if(request_irq(irq, eint_example_isr, IRQF_TRIGGER_NONE, "eint-example", NULL)) {
                        printk(KERN_ERR "EINT EXAMPLE IRQ LINE NOT AVAILABLE!!\n");
                        return -EINVAL;
                }
        }
```

Get eint node

Get gpio number & debouce time

Get irq number

Setup debounce time uses kernel standard function

eint-example.c

Examples(2/2)

```
/* here we use a device named eint_example for instance
 * this sample demonstrate how to implement non-autounmask EINT
 * and remember to call enable_irq(irq) when you want to enable it again!
 */
irqreturn_t eint_example_isr(int irq, void *desc)
{
     /* implement non-autounmask EINT */
     disable_irq_nosync(irq);
     return IRQ_HANDLED;
}
```

implement non-autounmask EINT

Changes of API

- Use irq_of_parse_and_map() to get virtual irq
- Use request_irq() instead of mt_eint_registration() to register ISR
  - interrupt flag should be IRQF_TRIGGER_NONE, since irq_of_parse_and_map() already set trigger type, you can also overwrite the trigger type here)
- Use enable_irq()/disable_irq() for mt_eint_mask()/mt_eint_unmask()
  - use disable_irq_nosync() in irq context instead
- Use irq_set_irq_type() for mt_eint_set_polarity()/mt_eint_set_sens()

IRQ Flags

- #define IRQ_TYPE_NONE 0
- #define IRQ_TYPE_EDGE_RISING 1
- #define IRQ_TYPE_EDGE_FALLING 2
- #define IRQ_TYPE_EDGE_BOTH (IRQ_TYPE_EDGE_FALLING | IRQ_TYPE_EDGE_RISING)
- #define IRQ_TYPE_LEVEL_HIGH 4
- #define IRQ_TYPE_LEVEL_LOW 8

CONFIDENTIAL B

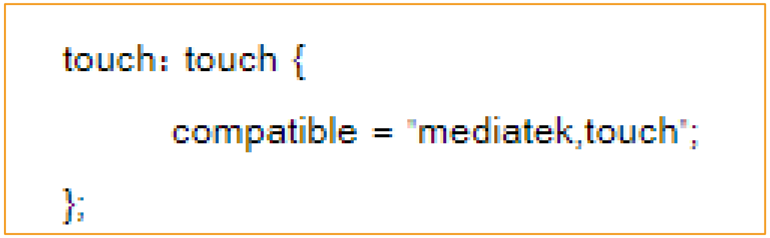
Touch

Outline

- Touch Device Tree Coding
  - Parameter
  - How to use the parameter
  - GPIO pinctrl modify
- Touch project config Coding

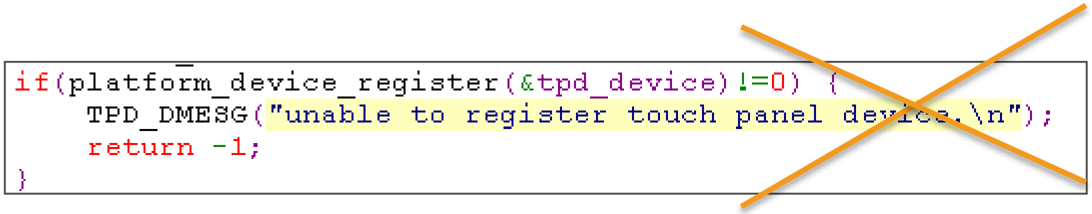
Touch Device Tree Coding

- 1.Add touch node in mtxxxx.dtsi
  - compatible = "mediatek,mt6xxx-touch";



```
touch: touch {
        compatible = "mediatek,touch";
};
```

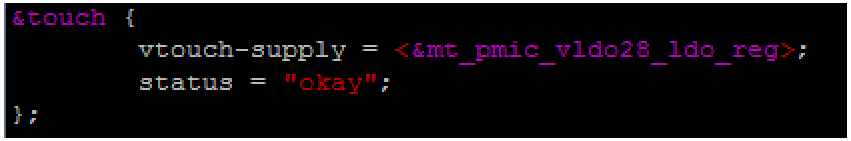
  - Remove platform_device_register()



```
if(platform_device_register(&tpd_device)!=0) {
    TPD_DMESG("unable to register touch panel device.\n");
    return -1;
}
```

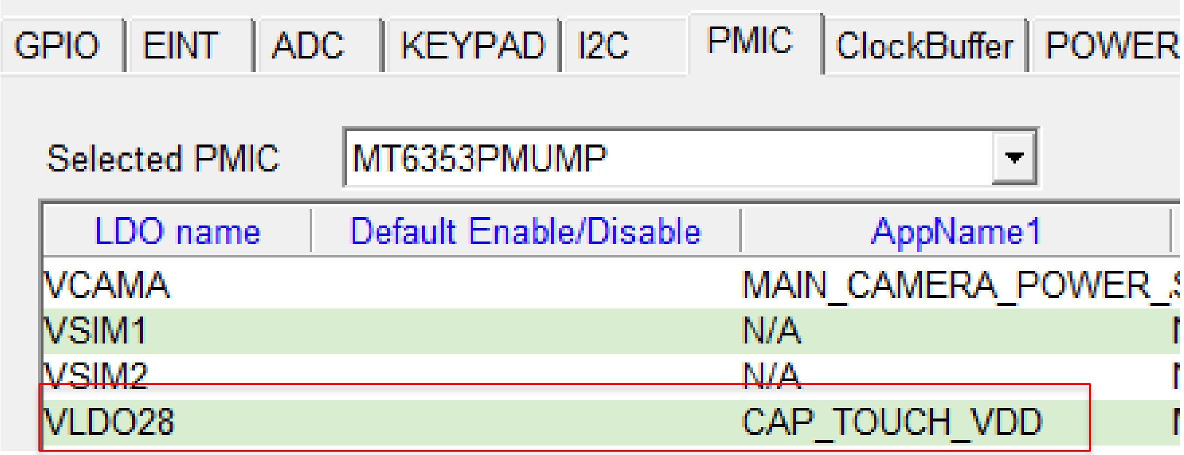
Touch Device Tree Coding

- 2.Touch regulator device tree
  - Defined in cust_pmic.dtsi gen by DCT tool



```
&touch {
        vtouch-supply = <&mt_pmic_vldo28_ldo_reg>;
        status = "okay";
};
```

  - This can be modify in dws

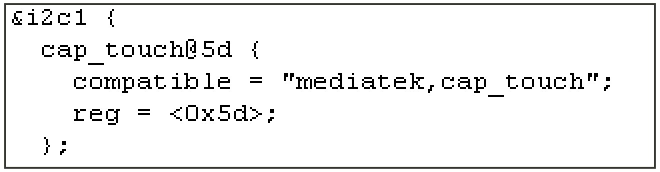


Selected PMIC MT6353PMUMP

LDO name	Default Enable/Disable	AppName1
VCAMA		MAIN_CAMERA_POWER_S
VSIM1		N/A
VSIM2		N/A
VLDO28		CAP_TOUCH_VDD

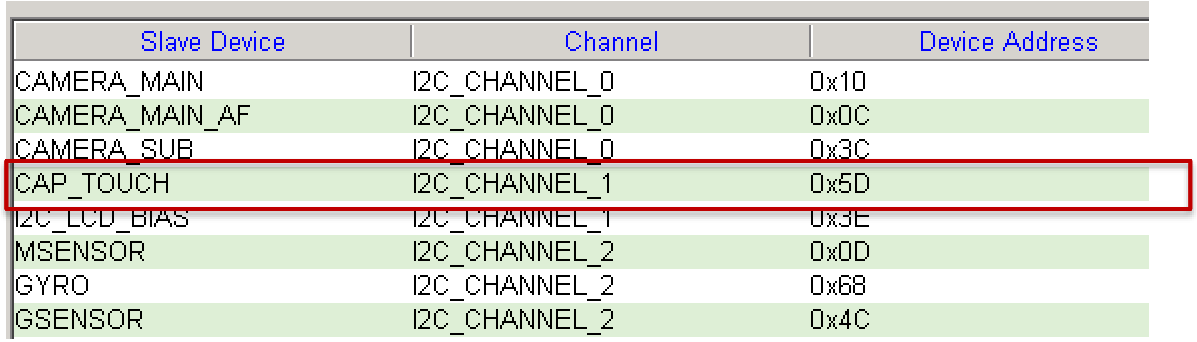
Touch Device Tree Coding

- 3. touch I2C device tree
  - Defined in cust_i2c.dtsi gen by DCT tool



```
&i2c1 {
  cap_touch@5d {
    compatible = "mediatek,cap_touch";
    reg = <0x5d>;
  };
```

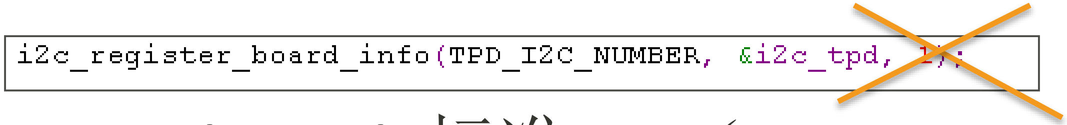
  - This can be modify in dws



Slave Device	Channel	Device Address
CAMERA_MAIN	I2C_CHANNEL_0	0x10
CAMERA_MAIN_AF	I2C_CHANNEL_0	0x0C
CAMERA_SUB	I2C_CHANNEL_0	0x3C
CAP_TOUCH	I2C_CHANNEL_1	0x5D
I2C_LCD_BIAS	I2C_CHANNEL_1	0x3E
MSENSOR	I2C_CHANNEL_2	0x0D
GYRO	I2C_CHANNEL_2	0x68
GSENSOR	I2C_CHANNEL_2	0x4C

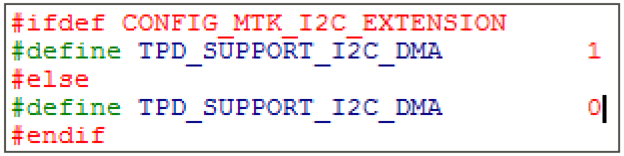
Touch Device Tree Coding

- 3. touch I2C device tree
  - Remove i2c_register_board_info()



```
i2c_register_board_info(TPD_I2C_NUMBER, &i2c_tpd, 1);
```

  - I2c use kernel 标准API（Start From MT6797）
    - You can use a micro CONFIG_MTK_I2C… it defined by I2c, touch no need to config.



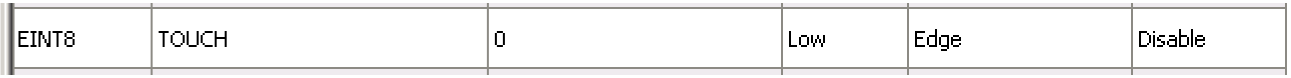
```
#ifdef CONFIG_MTK_I2C_EXTENSION
#define TPD_SUPPORT_I2C_DMA         1
#else
#define TPD_SUPPORT_I2C_DMA         0
#endif
```

    - If the driver only use after 6797, you can remove TPD_SUPPORT_I2C_DMA code
    - I2C auto to use DMA when data bigger than 8 byte

Touch Device tree coding

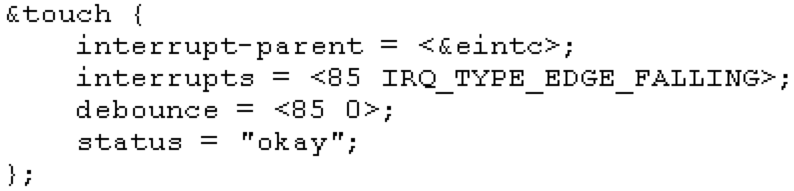
- 4 touch eint
  - Irq DWS setting
    - Eint Var need to select TOUCH(kernel-3.10 is TOUCH_PANEL), so it can be added to compatible = "mediatek,mt6xxx-touch";

      which is defined is chip.dtsi



EINT8	TOUCH	0	Low	Edge	Disable

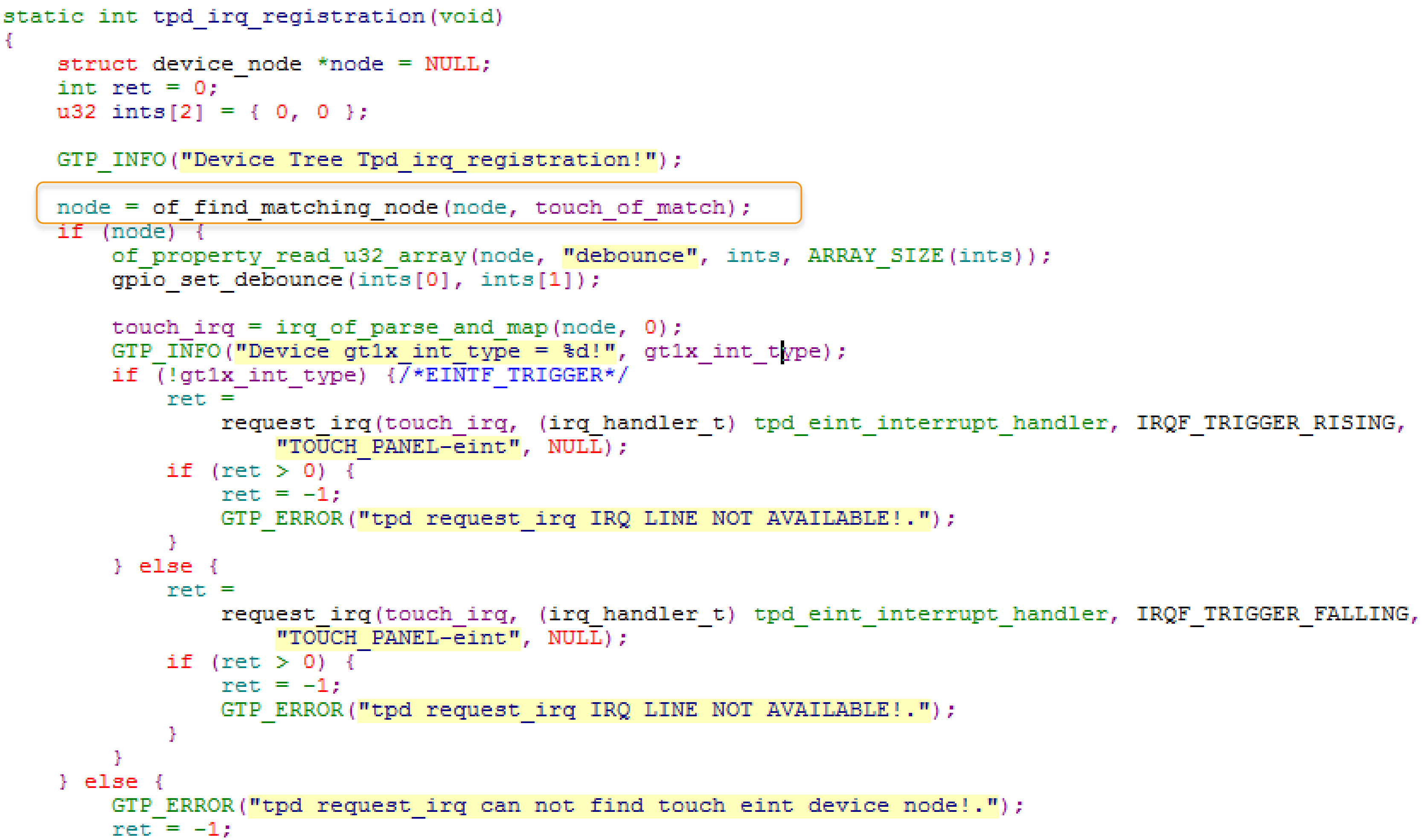
    - Output of touch eint in cust.dtsi



```
&touch {
    interrupt-parent = <&eintc>;
    interrupts = <85 IRQ_TYPE_EDGE_FALLING>;
    debounce = <85 0>;
    status = "okay";
};
```

Touch Device tree coding

- 4 touch eint
  - Touch Device tree coding



```
static int tpd_irq_registration(void)
{
    struct device_node *node = NULL;
    int ret = 0;
    u32 ints[2] = { 0, 0 };

    GTP_INFO("Device Tree Tpd_irq_registration!");

    node = of_find_matching_node(node, touch_of_match);
    if (node) {
        of_property_read_u32_array(node, "debounce", ints, ARRAY_SIZE(ints));
        gpio_set_debounce(ints[0], ints[1]);

        touch_irq = irq_of_parse_and_map(node, 0);
        GTP_INFO("Device gt1x_int_type = %d!", gt1x_int_type);
        if (!gt1x_int_type) {/*EINTF_TRIGGER*/
            ret =
                request_irq(touch_irq, (irq_handler_t) tpd_eint_interrupt_handler, IRQF_TRIGGER_RISING,
                    "TOUCH_PANEL-eint", NULL);
            if (ret > 0) {
                ret = -1;
                GTP_ERROR("tpd request_irq IRQ LINE NOT AVAILABLE!.");
            }
        } else {
            ret =
                request_irq(touch_irq, (irq_handler_t) tpd_eint_interrupt_handler, IRQF_TRIGGER_FALLING,
                    "TOUCH_PANEL-eint", NULL);
            if (ret > 0) {
                ret = -1;
                GTP_ERROR("tpd request_irq IRQ LINE NOT AVAILABLE!.");
            }
        }
    } else {
        GTP_ERROR("tpd request_irq can not find touch eint device node!.");
        ret = -1;
```

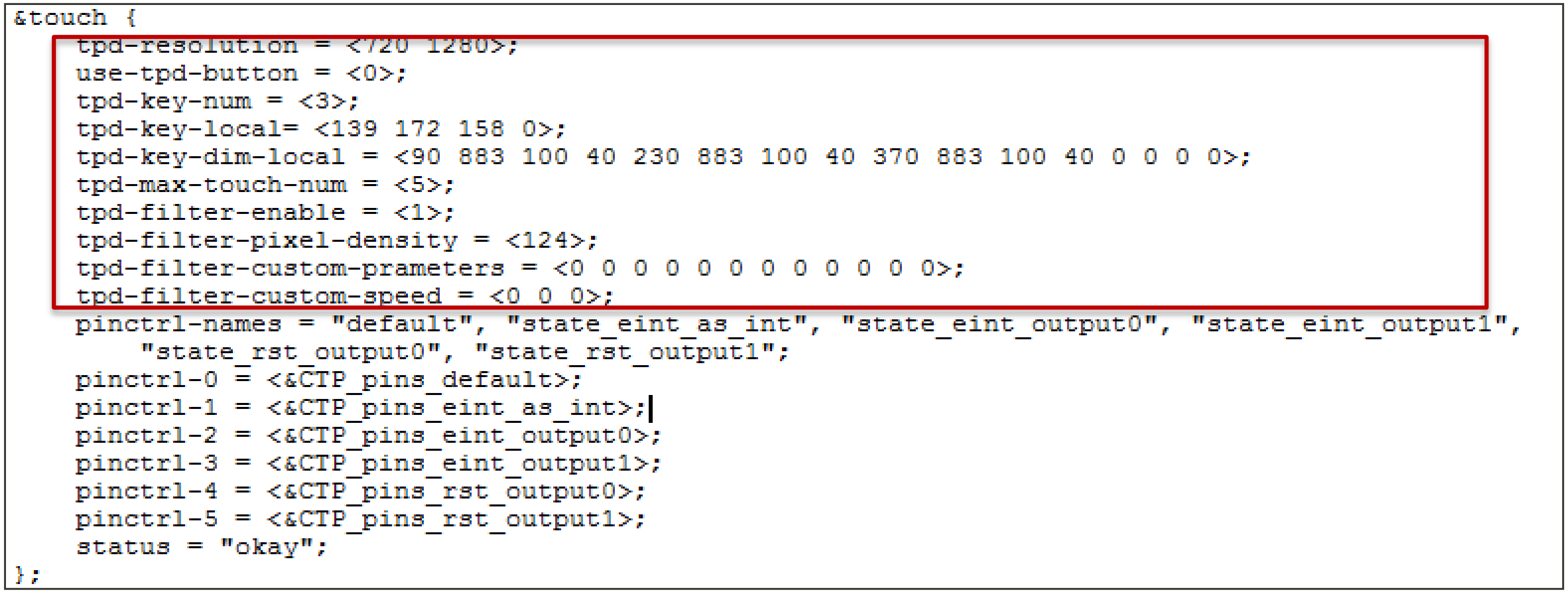
Touch Device tree coding

- 4 touch eint
  - Disable irq and enable irq need to be blance

Touch Irq Issue

Touch Device Tree Coding

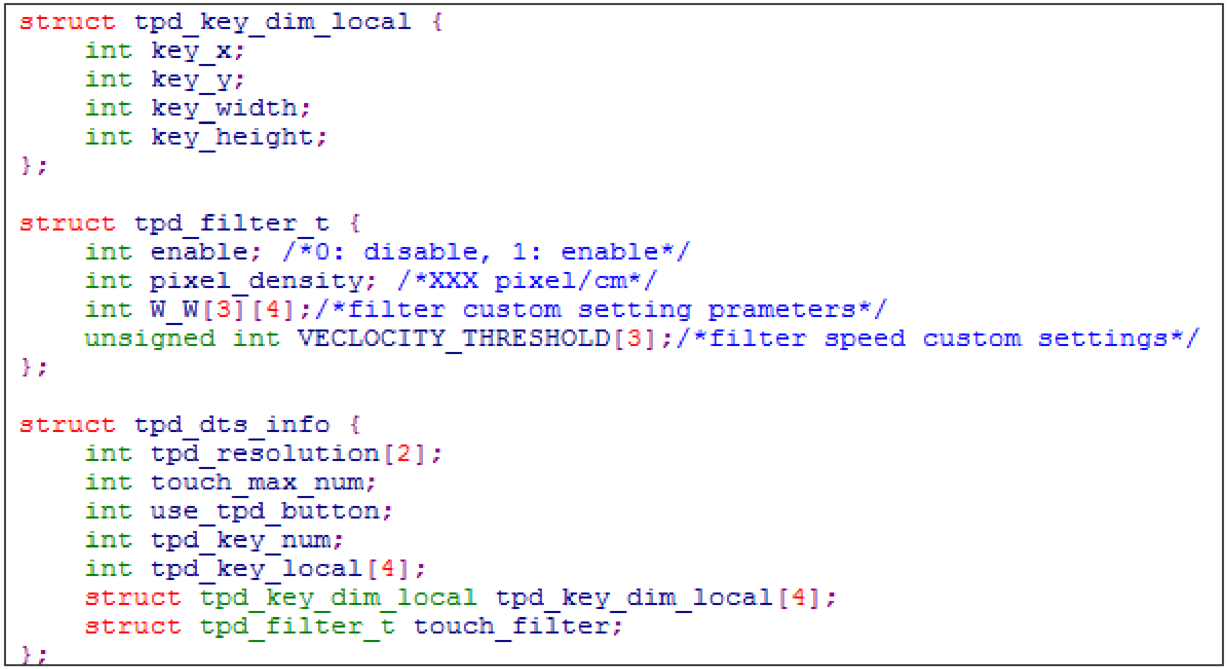
- 5.touch custom device tree
  - Defined in arch/arm/project.dtsi



```
&touch {
    tpd-resolution = <720 1280>;
    use-tpd-button = <0>;
    tpd-key-num = <3>;
    tpd-key-local= <139 172 158 0>;
    tpd-key-dim-local = <90 883 100 40 230 883 100 40 370 883 100 40 0 0 0 0>;
    tpd-max-touch-num = <5>;
    tpd-filter-enable = <1>;
    tpd-filter-pixel-density = <124>;
    tpd-filter-custom-prameters = <0 0 0 0 0 0 0 0 0 0 0 0>;
    tpd-filter-custom-speed = <0 0 0>;
    pinctrl-names = "default", "state_eint_as_int", "state_eint_output0", "state_eint_output1",
        "state_rst_output0", "state_rst_output1";
    pinctrl-0 = <&CTP_pins_default>;
    pinctrl-1 = <&CTP_pins_eint_as_int>;
    pinctrl-2 = <&CTP_pins_eint_output0>;
    pinctrl-3 = <&CTP_pins_eint_output1>;
    pinctrl-4 = <&CTP_pins_rst_output0>;
    pinctrl-5 = <&CTP_pins_rst_output1>;
    status = "okay";
};
```

Touch Device Tree Coding

- 5.touch custom device tree
  - Defined in Kernel4.9/drivers/input/touchscreen/mediatek/tpd.h



```
struct tpd_key_dim_local {
    int key_x;
    int key_y;
    int key_width;
    int key_height;
};

struct tpd_filter_t {
    int enable; /*0: disable, 1: enable*/
    int pixel_density; /*XXX pixel/cm*/
    int W_W[3][4];/*filter custom setting prameters*/
    unsigned int VECLOCITY_THRESHOLD[3];/*filter speed custom settings*/
};

struct tpd_dts_info {
    int tpd_resolution[2];
    int touch_max_num;
    int use_tpd_button;
    int tpd_key_num;
    int tpd_key_local[4];
    struct tpd_key_dim_local tpd_key_dim_local[4];
    struct tpd_filter_t touch_filter;
};
```

Touch Device Tree Coding

- 5.touch custom device tree

Parameter	Introduction	Comments
tpd_resolution[2]	touch panel resolution info for x and y axis	tpd_resolution[0]: LCM resolution of x axis tpd_resolution[1]: LCM resolution of y axis
use_tpd_button	define whether the touch panel use virtual key	1 stands for touch panel use touch virtual key 0 stands for touch panel not use touch virtual key
tpd_key_num	The number of the touch virtual key. you can not set this parameter if use_tpd_button is 0.	The max of the key number is 4.
tpd_key_local[4]	the Linux key value if touch virtual key is used, you can not set this parameter if use_tpd_button is 0.	fill in Linux key code which will use for virtual key on touch panel, layout from left to right corresponding to array value tpd_key_local[0], tpd_key_local[1], tpd_key_local[2], tpd_key_local[3]
tpd_key_dim_local[4] (include 4 parameters tpd_key_dim_local[4].key_x, tpd_key_dim_local[4].key_y, tpd_key_dim_local[4].key_width, tpd_key_dim_local[4].key_high)	the key layout info if touch virtual key is used, you can not set this parameter if use_tpd_button is 0.	every tpd_key_dim_local[i] corresponding to tpd_key_local[i] tpd_key_dim_local[i].key_x: location on x axis of tpd_key_local[i] tpd_key_dim_local[i].key_y: location on y axis of tpd_key_local[i] tpd_key_dim_local[i].key_width: width of tpd_key_local[i] tpd_key_dim_local[i].key_high: height of tpd_key_local[i]

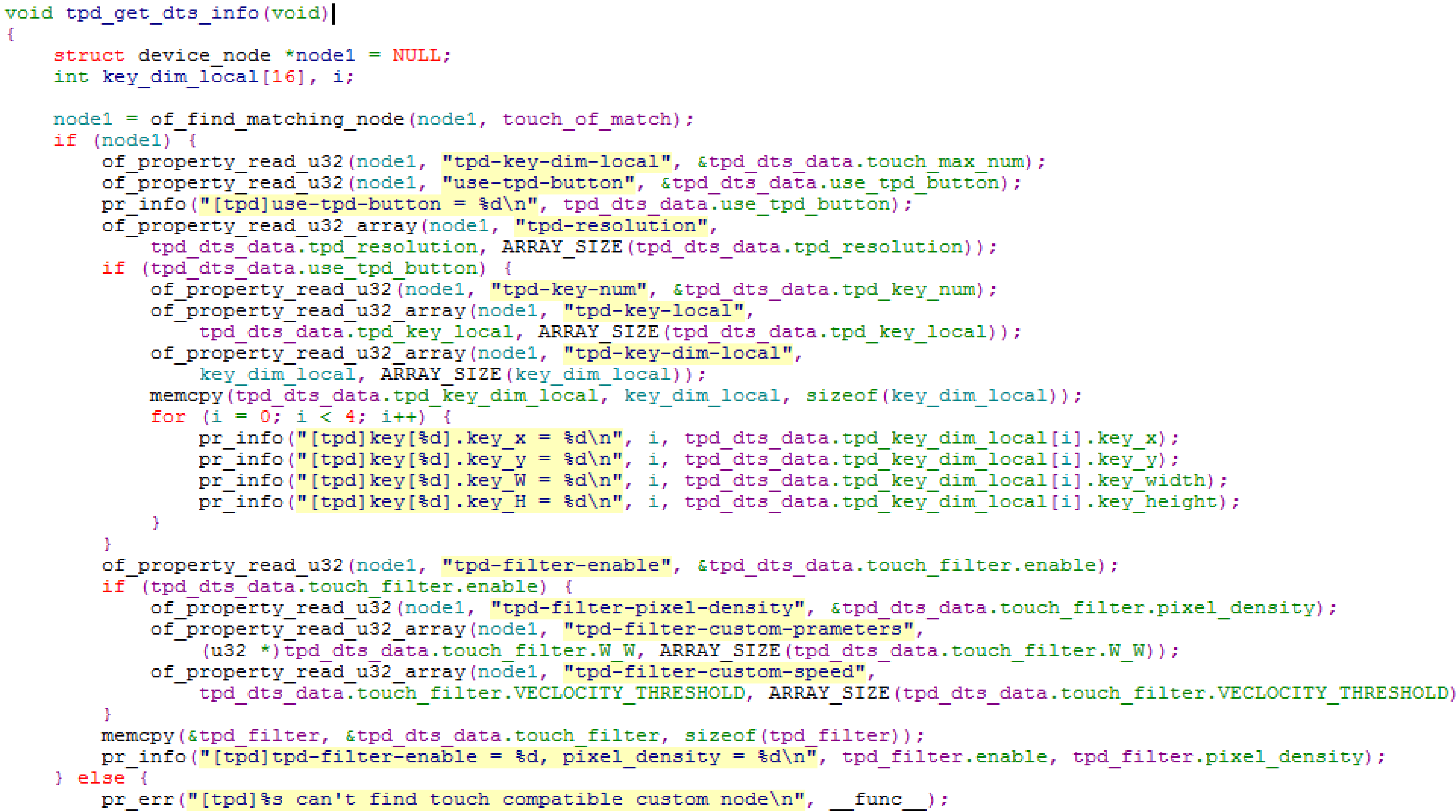
Touch Device Tree Coding

- 5.touch custom device tree

Parameter	Introduction	Comments
touch_max_num	touch panel resolution info for x and y axis	
tpd_filter_t **(include 4 parameters** **enable;** **pixel_density;** **W_W[3][4];** **VECLOCITY_THRESHOLD[3]** **)**	This defined touch filter para	Enable: 1 enable touch filter,0 disable Pixel_density:XXX pixel/cm W_W:filter custom setting prameters VECLOCITY_THRESHOLD:filter speed custom settings

Touch Device Tree Coding

- call tpd_get_dts_info to get touch devices tree



```
void tpd_get_dts_info(void)
{
    struct device_node *node1 = NULL;
    int key_dim_local[16], i;

    node1 = of_find_matching_node(node1, touch_of_match);
    if (node1) {
        of_property_read_u32(node1, "tpd-key-dim-local", &tpd_dts_data.touch_max_num);
        of_property_read_u32(node1, "use-tpd-button", &tpd_dts_data.use_tpd_button);
        pr_info("[tpd]use-tpd-button = %d\n", tpd_dts_data.use_tpd_button);
        of_property_read_u32_array(node1, "tpd-resolution",
            tpd_dts_data.tpd_resolution, ARRAY_SIZE(tpd_dts_data.tpd_resolution));
        if (tpd_dts_data.use_tpd_button) {
            of_property_read_u32(node1, "tpd-key-num", &tpd_dts_data.tpd_key_num);
            of_property_read_u32_array(node1, "tpd-key-local",
                tpd_dts_data.tpd_key_local, ARRAY_SIZE(tpd_dts_data.tpd_key_local));
            of_property_read_u32_array(node1, "tpd-key-dim-local",
                key_dim_local, ARRAY_SIZE(key_dim_local));
            memcpy(tpd_dts_data.tpd_key_dim_local, key_dim_local, sizeof(key_dim_local));
            for (i = 0; i < 4; i++) {
                pr_info("[tpd]key[%d].key_x = %d\n", i, tpd_dts_data.tpd_key_dim_local[i].key_x);
                pr_info("[tpd]key[%d].key_y = %d\n", i, tpd_dts_data.tpd_key_dim_local[i].key_y);
                pr_info("[tpd]key[%d].key_W = %d\n", i, tpd_dts_data.tpd_key_dim_local[i].key_width);
                pr_info("[tpd]key[%d].key_H = %d\n", i, tpd_dts_data.tpd_key_dim_local[i].key_height);
            }
        }
        of_property_read_u32(node1, "tpd-filter-enable", &tpd_dts_data.touch_filter.enable);
        if (tpd_dts_data.touch_filter.enable) {
            of_property_read_u32(node1, "tpd-filter-pixel-density", &tpd_dts_data.touch_filter.pixel_density);
            of_property_read_u32_array(node1, "tpd-filter-custom-prameters",
                (u32 *)tpd_dts_data.touch_filter.W_W, ARRAY_SIZE(tpd_dts_data.touch_filter.W_W));
            of_property_read_u32_array(node1, "tpd-filter-custom-speed",
                tpd_dts_data.touch_filter.VECLOCITY_THRESHOLD, ARRAY_SIZE(tpd_dts_data.touch_filter.VECLOCITY_THRESHOLD)
        }
        memcpy(&tpd_filter, &tpd_dts_data.touch_filter, sizeof(tpd_filter));
        pr_info("[tpd]tpd-filter-enable = %d, pixel_density = %d\n", tpd_filter.enable, tpd_filter.pixel_density);
    } else {
        pr_err("[tpd]%s can't find touch compatible custom node\n", __func__);
```

Touch Device Tree Coding

- GPIO pinctrl modify
  - Remove all mt_set_gpio_xxx() function
  - Replaced by GPIO pinctrl
- Project defined
  - Defined Kernel-4.9/arch/armxx/boot/dts/project.dts.
  - If need to set other GPIO please modify here.

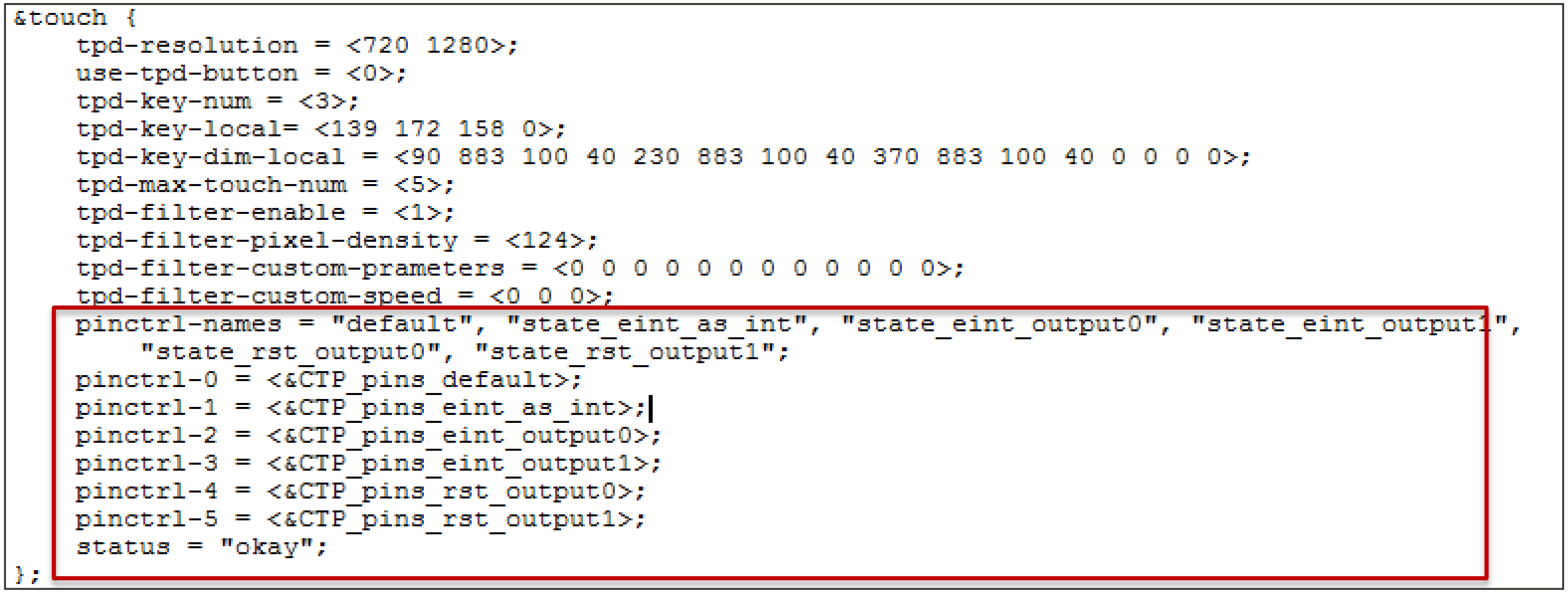
INTERNAL USE

GPIO pinctrl- Kernel Standardization

Touch Device Tree Coding

- First part

Kernel-3.xx/arch/armxx/boot/dts/project.dts

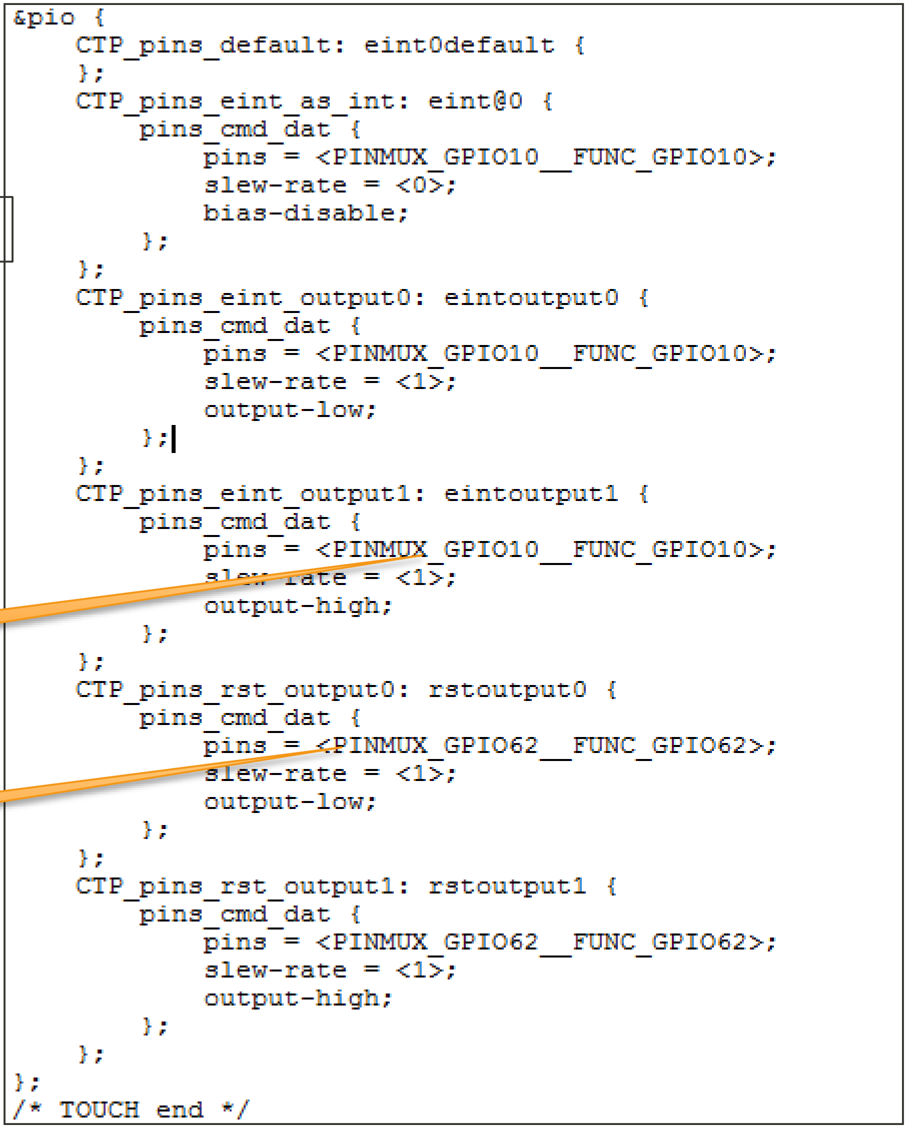


```
&touch {
    tpd-resolution = <720 1280>;
    use-tpd-button = <0>;
    tpd-key-num = <3>;
    tpd-key-local= <139 172 158 0>;
    tpd-key-dim-local = <90 883 100 40 230 883 100 40 370 883 100 40 0 0 0 0>;
    tpd-max-touch-num = <5>;
    tpd-filter-enable = <1>;
    tpd-filter-pixel-density = <124>;
    tpd-filter-custom-prameters = <0 0 0 0 0 0 0 0 0 0 0 0>;
    tpd-filter-custom-speed = <0 0 0>;
    pinctrl-names = "default", "state_eint_as_int", "state_eint_output0", "state_eint_output1",
        "state_rst_output0", "state_rst_output1";
    pinctrl-0 = <&CTP_pins_default>;
    pinctrl-1 = <&CTP_pins_eint_as_int>;
    pinctrl-2 = <&CTP_pins_eint_output0>;
    pinctrl-3 = <&CTP_pins_eint_output1>;
    pinctrl-4 = <&CTP_pins_rst_output0>;
    pinctrl-5 = <&CTP_pins_rst_output1>;
    status = "okay";
};
```

Touch Device Tree Coding

- Second part

Kernel-3.xx/arch/armxx/boot/dts/project.dts



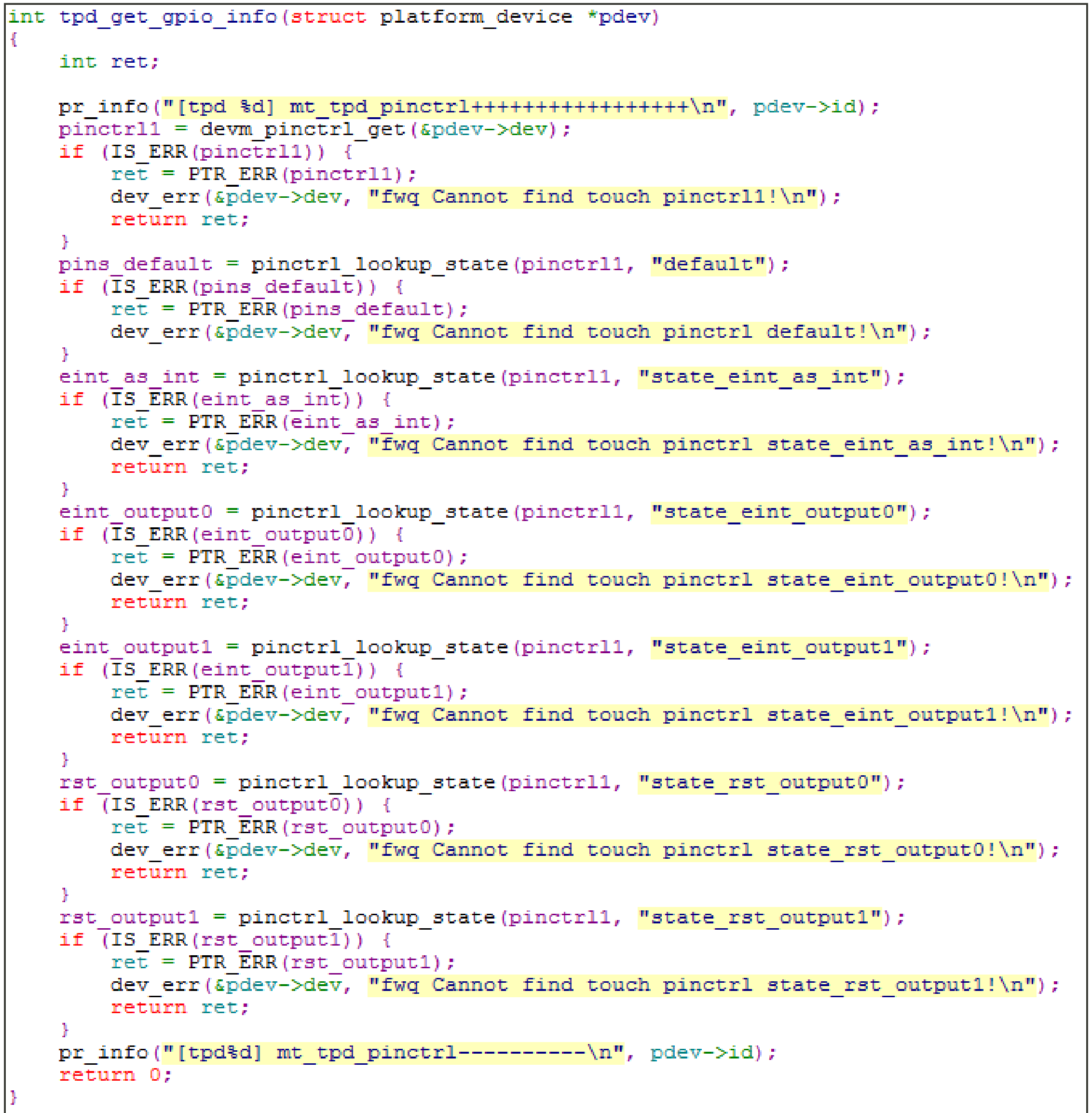
```
&pio {
    CTP_pins_default: eint0default {
    };
    CTP_pins_eint_as_int: eint@0 {
        pins_cmd_dat {
            pins = <PINMUX_GPIO10__FUNC_GPIO10>;
            slew-rate = <0>;
            bias-disable;
        };
    };
    CTP_pins_eint_output0: eintoutput0 {
        pins_cmd_dat {
            pins = <PINMUX_GPIO10__FUNC_GPIO10>;
            slew-rate = <1>;
            output-low;
        };
    };
    CTP_pins_eint_output1: eintoutput1 {
        pins_cmd_dat {
            pins = <PINMUX_GPIO10__FUNC_GPIO10>;
            slew-rate = <1>;
            output-high;
        };
    };
    CTP_pins_rst_output0: rstoutput0 {
        pins_cmd_dat {
            pins = <PINMUX_GPIO62__FUNC_GPIO62>;
            slew-rate = <1>;
            output-low;
        };
    };
    CTP_pins_rst_output1: rstoutput1 {
        pins_cmd_dat {
            pins = <PINMUX_GPIO62__FUNC_GPIO62>;
            slew-rate = <1>;
            output-high;
        };
    };
};
/* TOUCH end */
```

EINT Pin

Reset Pin

Touch Device Tree Coding

- How to use?
  - probe



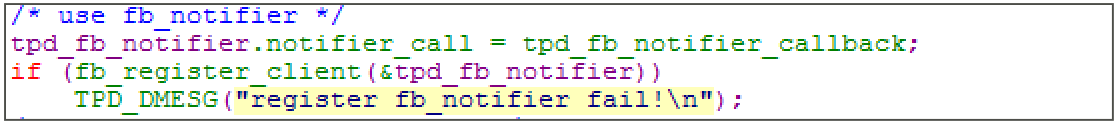
```
int tpd_get_gpio_info(struct platform_device *pdev)
{
    int ret;

    pr_info("[tpd %d] mt_tpd_pinctrl++++++++++++++++++\n", pdev->id);
    pinctrl1 = devm_pinctrl_get(&pdev->dev);
    if (IS_ERR(pinctrl1)) {
        ret = PTR_ERR(pinctrl1);
        dev_err(&pdev->dev, "fwq Cannot find touch pinctrl1!\n");
        return ret;
    }
    pins_default = pinctrl_lookup_state(pinctrl1, "default");
    if (IS_ERR(pins_default)) {
        ret = PTR_ERR(pins_default);
        dev_err(&pdev->dev, "fwq Cannot find touch pinctrl default!\n");
    }
    eint_as_int = pinctrl_lookup_state(pinctrl1, "state_eint_as_int");
    if (IS_ERR(eint_as_int)) {
        ret = PTR_ERR(eint_as_int);
        dev_err(&pdev->dev, "fwq Cannot find touch pinctrl state_eint_as_int!\n");
        return ret;
    }
    eint_output0 = pinctrl_lookup_state(pinctrl1, "state_eint_output0");
    if (IS_ERR(eint_output0)) {
        ret = PTR_ERR(eint_output0);
        dev_err(&pdev->dev, "fwq Cannot find touch pinctrl state_eint_output0!\n");
        return ret;
    }
    eint_output1 = pinctrl_lookup_state(pinctrl1, "state_eint_output1");
    if (IS_ERR(eint_output1)) {
        ret = PTR_ERR(eint_output1);
        dev_err(&pdev->dev, "fwq Cannot find touch pinctrl state_eint_output1!\n");
        return ret;
    }
    rst_output0 = pinctrl_lookup_state(pinctrl1, "state_rst_output0");
    if (IS_ERR(rst_output0)) {
        ret = PTR_ERR(rst_output0);
        dev_err(&pdev->dev, "fwq Cannot find touch pinctrl state_rst_output0!\n");
        return ret;
    }
    rst_output1 = pinctrl_lookup_state(pinctrl1, "state_rst_output1");
    if (IS_ERR(rst_output1)) {
        ret = PTR_ERR(rst_output1);
        dev_err(&pdev->dev, "fwq Cannot find touch pinctrl state_rst_output1!\n");
        return ret;
    }
    pr_info("[tpd%d] mt_tpd_pinctrl----------\n", pdev->id);
    return 0;
}
```

Touch Suspend/Resume modify

- In kernel 3.10 we use early suspend and late resume, but in kernel-4.9 we use kernel API.
- Remove all CONFIG_HAS_EARLYSUSPEND, this will never be defined in kernel-4.9.
- We touch use kernel fb_notify to suspend and suspend.
- All common code is in mtk_tpd.c

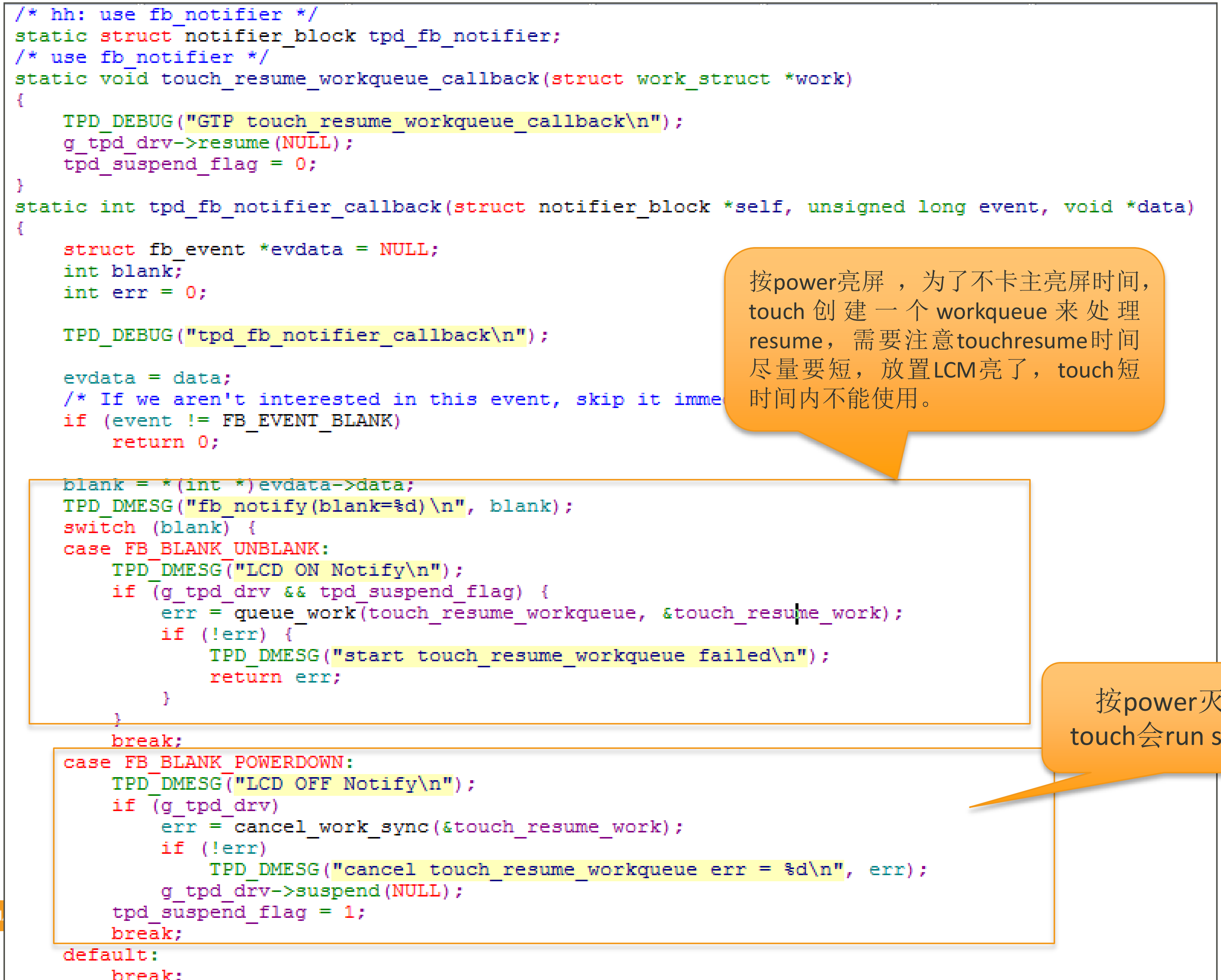
Touch Suspend/Resume modify

- Mtk_tpd.c
  - Register tpd_fb_notifier_callback
    - In tpd_probe, it will be called when LCM on and off



```
/* use fb_notifier */
tpd_fb_notifier.notifier_call = tpd_fb_notifier_callback;
if (fb_register_client(&tpd_fb_notifier))
    TPD_DMESG("register fb_notifier fail!\n");
```

Touch Suspend/Resume modify



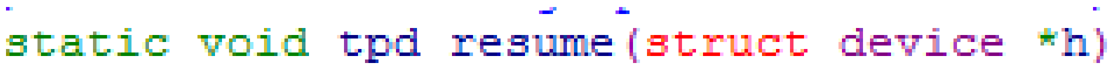
```
/* hh: use fb_notifier */
static struct notifier_block tpd_fb_notifier;
/* use fb_notifier */
static void touch_resume_workqueue_callback(struct work_struct *work)
{
    TPD_DEBUG("GTP touch_resume_workqueue_callback\n");
    g_tpd_drv->resume(NULL);
    tpd_suspend_flag = 0;
}
static int tpd_fb_notifier_callback(struct notifier_block *self, unsigned long event, void *data)
{
    struct fb_event *evdata = NULL;
    int blank;
    int err = 0;

    TPD_DEBUG("tpd_fb_notifier_callback\n");

    evdata = data;
    /* If we aren't interested in this event, skip it imme
    if (event != FB_EVENT_BLANK)
        return 0;

    blank = *(int *)evdata->data;
    TPD_DMESG("fb_notify(blank=%d)\n", blank);
    switch (blank) {
    case FB_BLANK_UNBLANK:
        TPD_DMESG("LCD ON Notify\n");
        if (g_tpd_drv && tpd_suspend_flag) {
            err = queue_work(touch_resume_workqueue, &touch_resume_work);
            if (!err) {
                TPD_DMESG("start touch_resume_workqueue failed\n");
                return err;
            }
        }
        break;
    case FB_BLANK_POWERDOWN:
        TPD_DMESG("LCD OFF Notify\n");
        if (g_tpd_drv)
            err = cancel_work_sync(&touch_resume_work);
            if (!err)
                TPD_DMESG("cancel touch_resume_workqueue err = %d\n", err);
            g_tpd_drv->suspend(NULL);
        tpd_suspend_flag = 1;
        break;
    default:
        break;
```

按power亮屏 ，为了不卡主亮屏时间，touch 创建一个 workqueue 来处理 resume， 需要注意touchresume时间尽量要短，放置LCM亮了，touch短时间内不能使用。

按power灭屏，touch会run suspend

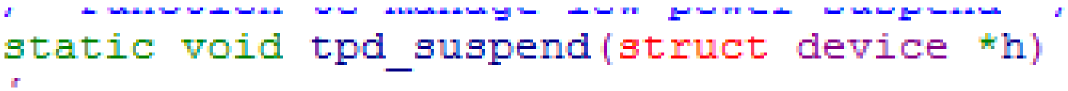
Touch Suspend/Resume modify

- Resume



```
static void tpd resume(struct device *h)
```

- Suspend



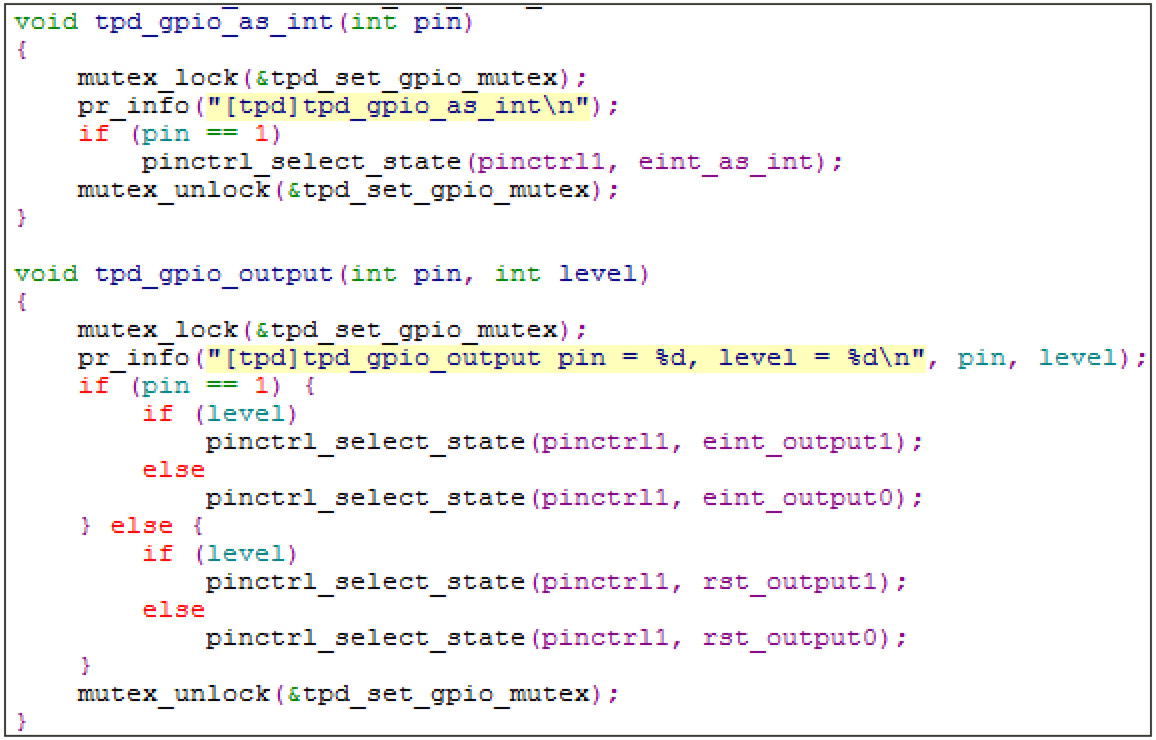
```
/*   Function to manage low power suspend  */
static void tpd_suspend(struct device *h)
```

Touch Device Tree Coding

- Notice
  - Use MT6735 platform as example
  - Use GT1151 touch as example

Touch Device Tree Coding

- How to use?
  - Set GPIO



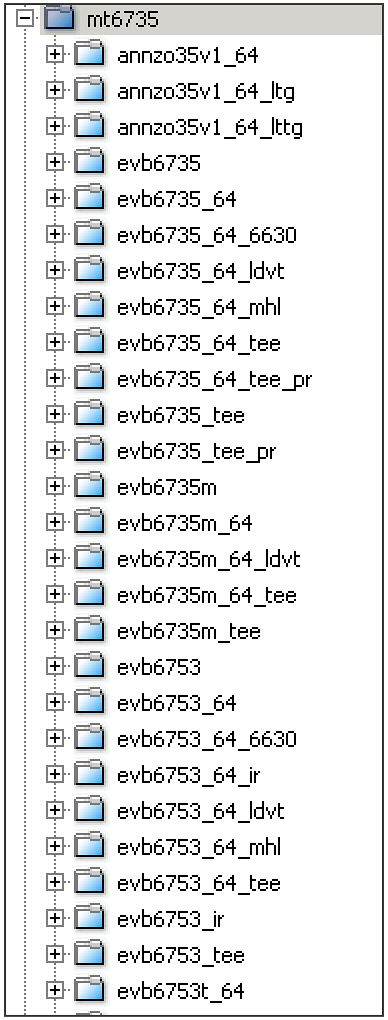
```
void tpd_gpio_as_int(int pin)
{
    mutex_lock(&tpd_set_gpio_mutex);
    pr_info("[tpd]tpd_gpio_as_int\n");
    if (pin == 1)
        pinctrl_select_state(pinctrl1, eint_as_int);
    mutex_unlock(&tpd_set_gpio_mutex);
}

void tpd_gpio_output(int pin, int level)
{
    mutex_lock(&tpd_set_gpio_mutex);
    pr_info("[tpd]tpd_gpio_output pin = %d, level = %d\n", pin, level);
    if (pin == 1) {
        if (level)
            pinctrl_select_state(pinctrl1, eint_output1);
        else
            pinctrl_select_state(pinctrl1, eint_output0);
    } else {
        if (level)
            pinctrl_select_state(pinctrl1, rst_output1);
        else
            pinctrl_select_state(pinctrl1, rst_output0);
    }
    mutex_unlock(&tpd_set_gpio_mutex);
}
```

Touch project config Coding

- Before Git kernel-3.18
  - Project custom folder in mach
  - Use 6735 as example

alps/kernel-3.18/drivers/misc/mediatek/mach/mt6735/

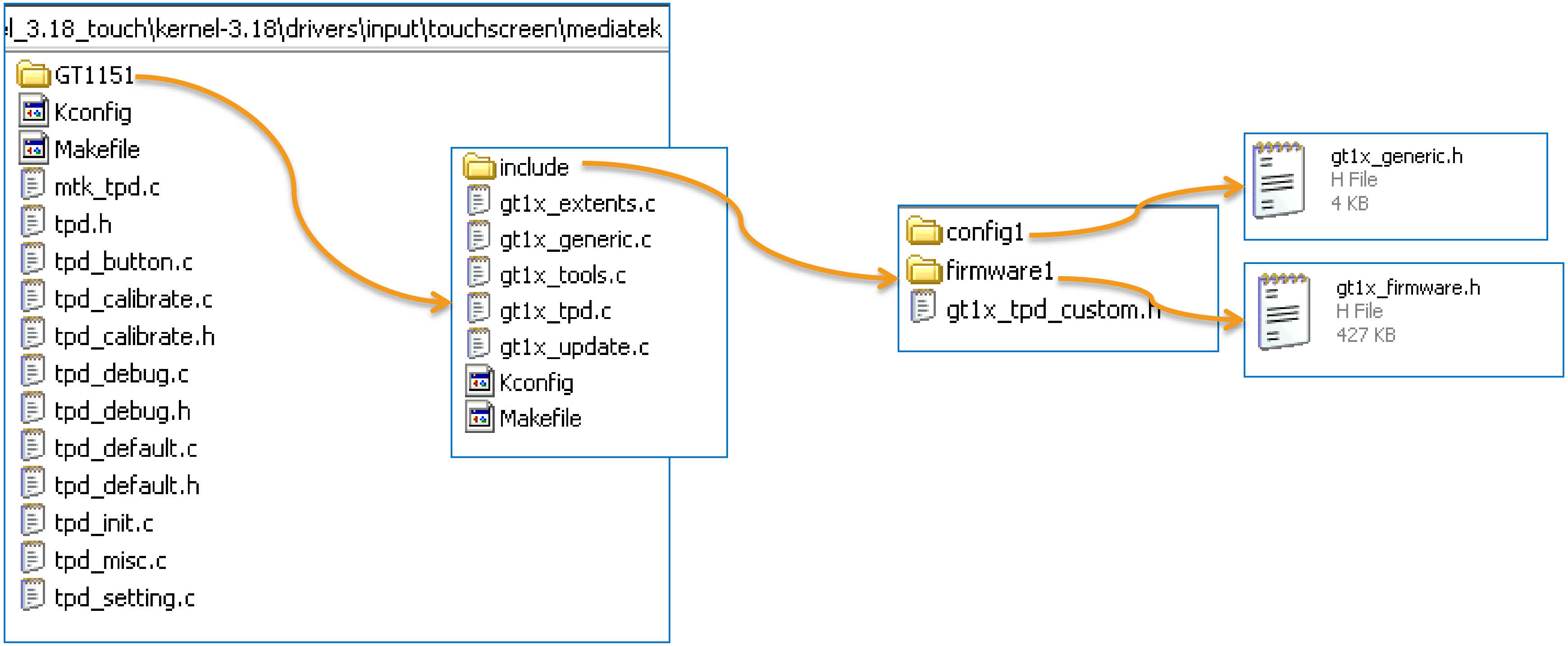
  - Git kernel-3.18 will remove mach folder and no custom config folder



```
mt6735
  annzo35v1_64
  annzo35v1_64_ltg
  annzo35v1_64_lttg
  evb6735
  evb6735_64
  evb6735_64_6630
  evb6735_64_ldvt
  evb6735_64_mhl
  evb6735_64_tee
  evb6735_64_tee_pr
  evb6735_tee
  evb6735_tee_pr
  evb6735m
  evb6735m_64
  evb6735m_64_ldvt
  evb6735m_64_tee
  evb6735m_tee
  evb6753
  evb6753_64
  evb6753_64_6630
  evb6753_64_ir
  evb6753_64_ldvt
  evb6753_64_mhl
  evb6753_64_tee
  evb6753_ir
  evb6753_tee
  evb6753t_64
```

Touch project config Coding

- Git kernel-3.18:Remove project custom folder
  - Touch driver结构如下



l_3.18_touch\kernel-3.18\drivers\input\touchscreen\mediatek

- GT1151
- Kconfig
- Makefile
- mtk_tpd.c
- tpd.h
- tpd_button.c
- tpd_calibrate.c
- tpd_calibrate.h
- tpd_debug.c
- tpd_debug.h
- tpd_default.c
- tpd_default.h
- tpd_init.c
- tpd_misc.c
- tpd_setting.c

GT1151:

- include
- gt1x_extents.c
- gt1x_generic.c
- gt1x_tools.c
- gt1x_tpd.c
- gt1x_update.c
- Kconfig
- Makefile

include:

- config1
- firmware1
- gt1x_tpd_custom.h

config1:

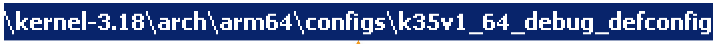
gt1x_generic.h
H File
4 KB

firmware1:

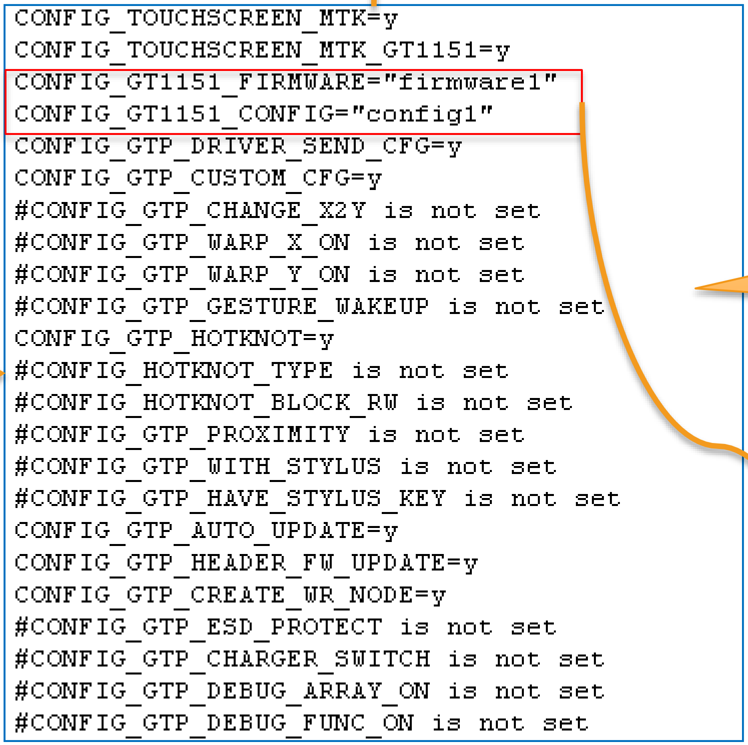
gt1x_firmware.h
H File
427 KB

Touch project config Coding

- 所有custom config定义在project_config中

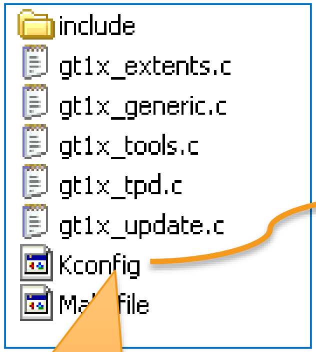


```
\kernel-3.18\arch\arm64\configs\k35v1_64_debug_defconfig
```



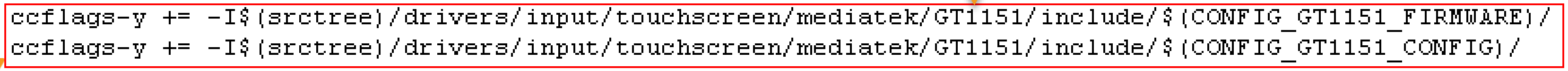
```
CONFIG_TOUCHSCREEN_MTK=y
CONFIG_TOUCHSCREEN_MTK_GT1151=y
CONFIG_GT1151_FIRMWARE="firmware1"
CONFIG_GT1151_CONFIG="config1"
CONFIG_GTP_DRIVER_SEND_CFG=y
CONFIG_GTP_CUSTOM_CFG=y
#CONFIG_GTP_CHANGE_X2Y is not set
#CONFIG_GTP_WARP_X_ON is not set
#CONFIG_GTP_WARP_Y_ON is not set
#CONFIG_GTP_GESTURE_WAKEUP is not set
CONFIG_GTP_HOTKNOT=y
#CONFIG_HOTKNOT_TYPE is not set
#CONFIG_HOTKNOT_BLOCK_RW is not set
#CONFIG_GTP_PROXIMITY is not set
#CONFIG_GTP_WITH_STYLUS is not set
#CONFIG_GTP_HAVE_STYLUS_KEY is not set
CONFIG_GTP_AUTO_UPDATE=y
CONFIG_GTP_HEADER_FW_UPDATE=y
CONFIG_GTP_CREATE_WR_NODE=y
#CONFIG_GTP_ESD_PROTECT is not set
#CONFIG_GTP_CHARGER_SWITCH is not set
#CONFIG_GTP_DEBUG_ARRAY_ON is not set
#CONFIG_GTP_DEBUG_FUNC_ON is not set
```

与project相关宏define在project.config中



```
include
gt1x_extents.c
gt1x_generic.c
gt1x_tools.c
gt1x_tpd.c
gt1x_update.c
Kconfig
Ma file
```

每个Touch driver下增加Kconfig文件



```
ccflags-y += -I$(srctree)/drivers/input/touchscreen/mediatek/GT1151/include/$(CONFIG_GT1151_FIRMWARE)/
ccflags-y += -I$(srctree)/drivers/input/touchscreen/mediatek/GT1151/include/$(CONFIG_GT1151_CONFIG)/
```

Touch project config Coding

- GIT change ID
  - Icde8249b8134b388a7a88e8aae1593d2f90a535d
- If use another touch IC
  - If still define project config in custom headfile
    - 1 select all project config parameters
    - 2 defined them in project_defconfig(arch/arm/config)
    - 3 modify source code

CONFIDENTIAL B

Keypad

Agenda

- introduction
- Architecture
- Interface
- Customization
- Build

introduction

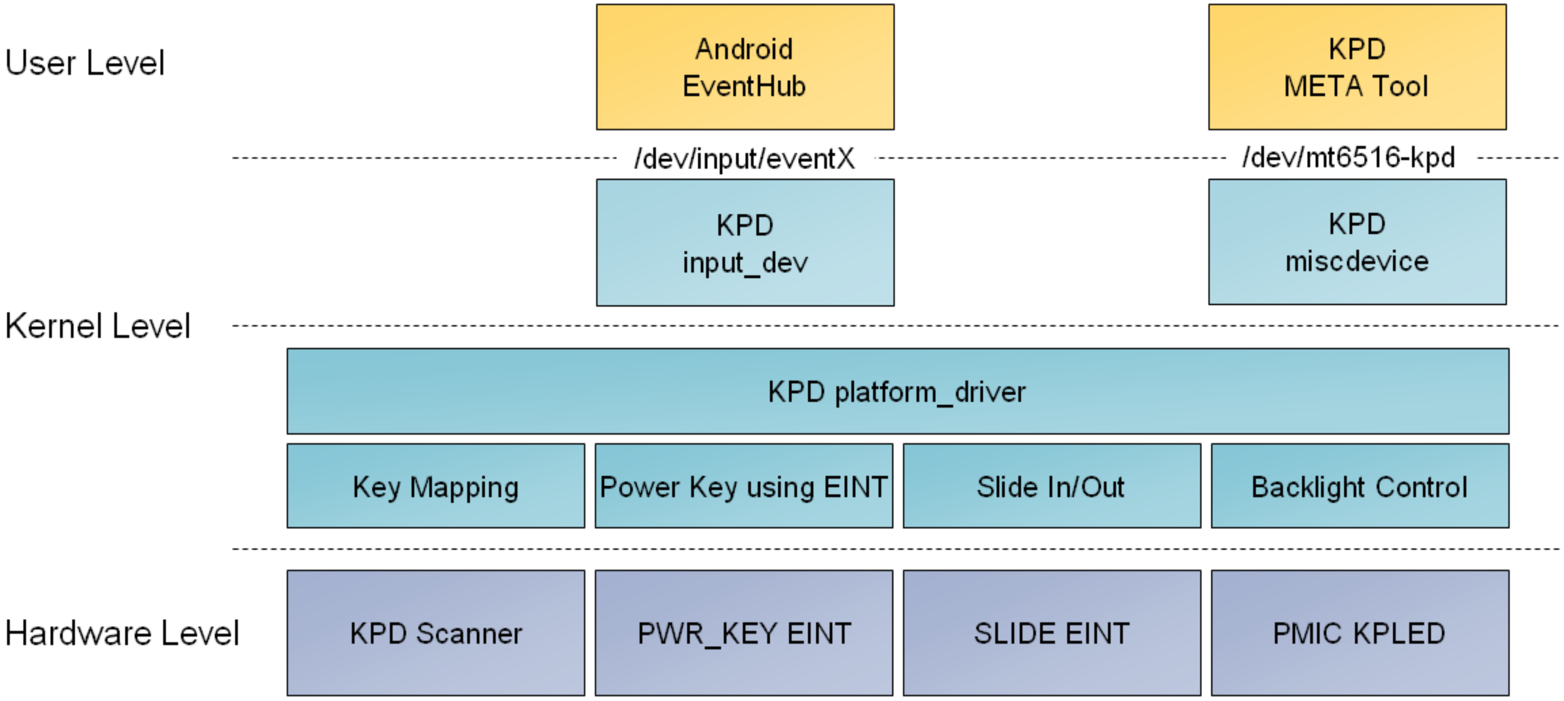
- Report the key event based on a key mapping Table when the key is pressed or released
- Report the key event when the Powerkey is pressed or released.
- Report the Switch key event when Slide QWERTY keypad is slid in or out.
- When the is pressed or released ,there is a keypad interrupt issue.

Agenda

- introduction
- Architecture
- Interface
- Customization
- Build

Architecture

- The architecture of Keypad driver is shown as follows. There are four main tasks in Keypad driver:



Keypad Driver Architecture

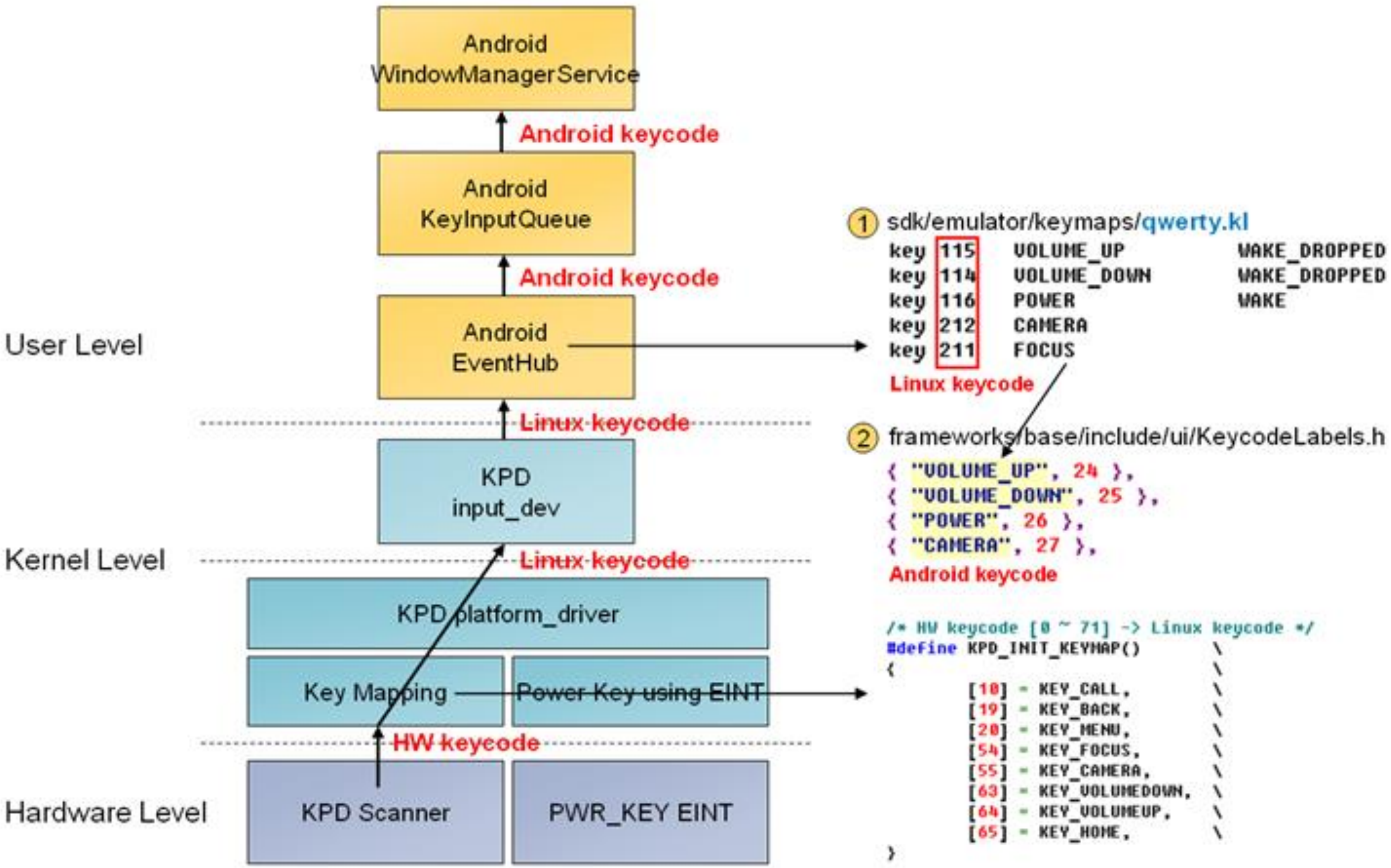
Architecture

- If the key is pressed or released, recognize which key and report the key event based on a key mapping table to Input subsystem through Keypad input device. Use PWR_KEY EINT to detect whether Power key is pressed or released and report the key event to Input subsystem through Keypad input device. (We will use PMIC callback function o detect whether Power key is pressed or released if it is connected to PMIC by hardware layout design.)
- If the device has Slide QWERTY keypad, use SLIDE EINT to detect whether QWERTY keypad is slid in or out and report the switch event to Input subsystem through Keypad input device.
- For META tool, use PMIC KPLED to show the backlight effect. (This feature has been removed since Android 2.3.)

Procedure & Flow

- When the key is pressed or released, Keypad Scanner will issue a interrupt, then Keypad driver recognizes this key's keycode and state from Keypad Scanner registers. This kind of keycode is called HW keycode (0 to 71). The responsibility of Keypad driver is to translate HW keycode to Linux keycode based on a pre-defined key mapping table. After Keypad driver gets the Linux keycode, it will report this Linux keycode to upper layer Input subsystem. And then, Android EventHub can get this Linux keycode from Input subsystem and translates it to Android keycode. So from this layer Android EventHub, the key pressing or releasing is represented as a key event with Android keycode. The keycode translation flow is shown as follows.

Procedure & Flow



① sdk/emulator/keymaps/qwerty.kl

```
key 115   VOLUME_UP              WAKE_DROPPED
key 114   VOLUME_DOWN            WAKE_DROPPED
key 116   POWER                  WAKE
key 212   CAMERA
key 211   FOCUS
```

Linux keycode

② frameworks/base/include/ui/KeycodeLabels.h

```
{ "VOLUME_UP", 24 },
{ "VOLUME_DOWN", 25 },
{ "POWER", 26 },
{ "CAMERA", 27 },
```

Android keycode

```
/* HW keycode [0 ~ 71] -> Linux keycode */
#define KPD_INIT_KEYMAP()               \
{                                       \
        [10] = KEY_CALL,                \
        [19] = KEY_BACK,                \
        [20] = KEY_MENU,                \
        [54] = KEY_FOCUS,               \
        [55] = KEY_CAMERA,              \
        [63] = KEY_VOLUMEDOWN,          \
        [64] = KEY_VOLUMEUP,            \
        [65] = KEY_HOME,                \
}
```

Procedure & Flow

- **Note-1**: The detection of Power key using PWR_KEY EINT or PMIC callback function does not go through Keypad Scanner. In this case, we are not going to get Power key's HW keycode from Keypad Scanner registers.

- **Note-2:** The language support needs to rely on IME (Input Method Engine) not Keypad driver. Keypad driver's main task is to report the key event with the correct "Linux keycode" and state to upper layer when the key is pressed or released.

Agenda

- introduction
- Architecture
- Interface
- Customization
- Build

kpd_dev_ioctl

Prototype		
long kpd_dev_ioctl(struct file *file, unsigned int cmd, unsigned long arg)		
Parameters		
in	file	File pointer
in	cmd	IOCTL command
in/out	arg	The command's argument and it could be input or output
Return Value		
The return value depends on IOCTL command		

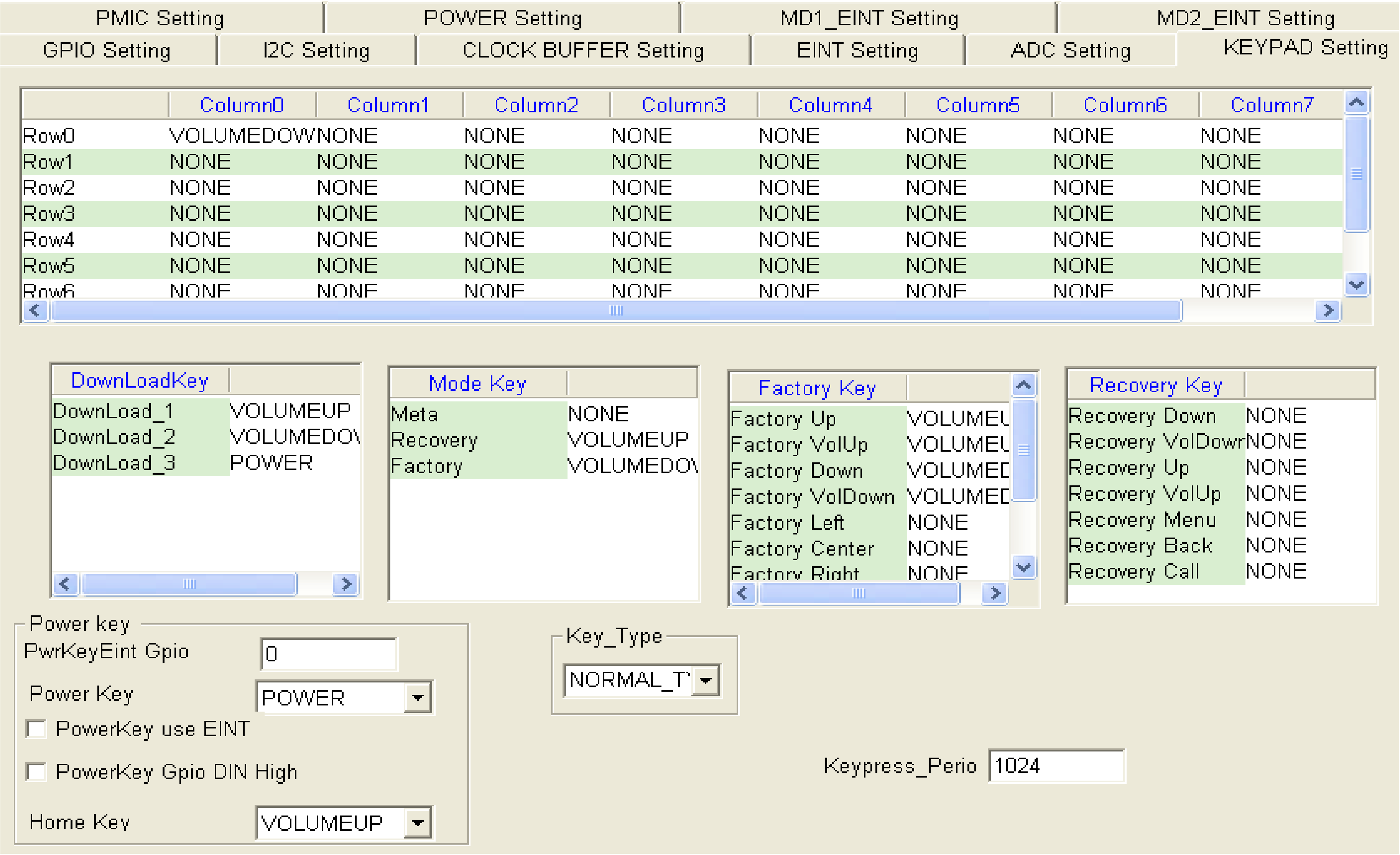
CMD	**SET_KPD_KCOL**
ARG	NULL
Return	0: the operation succeeds; otherwise: the operation fails

Agenda

- introduction
- Architecture
- Interface
- Customization
- Build

DCT (Driver Customization Tool)

DCT is used to customize the key mapping, de-bounce time and Power key setting in Keypad driver.



	Column0	Column1	Column2	Column3	Column4	Column5	Column6	Column7
Row0	VOLUMEDOW	NONE	NONE	NONE	NONE	NONE	NONE	NONE
Row1	NONE	NONE	NONE	NONE	NONE	NONE	NONE	NONE
Row2	NONE	NONE	NONE	NONE	NONE	NONE	NONE	NONE
Row3	NONE	NONE	NONE	NONE	NONE	NONE	NONE	NONE
Row4	NONE	NONE	NONE	NONE	NONE	NONE	NONE	NONE
Row5	NONE	NONE	NONE	NONE	NONE	NONE	NONE	NONE
Row6	NONE	NONE	NONE	NONE	NONE	NONE	NONE	NONE

DownLoadKey	
DownLoad_1	VOLUMEUP
DownLoad_2	VOLUMEDO\
DownLoad_3	POWER

Mode Key	
Meta	NONE
Recovery	VOLUMEUP
Factory	VOLUMEDO\

Factory Key	
Factory Up	VOLUMEU
Factory VolUp	VOLUMEU
Factory Down	VOLUMED
Factory VolDown	VOLUMED
Factory Left	NONE
Factory Center	NONE
Factory Right	NONE

Recovery Key	
Recovery Down	NONE
Recovery VolDowr	NONE
Recovery Up	NONE
Recovery VolUp	NONE
Recovery Menu	NONE
Recovery Back	NONE
Recovery Call	NONE

Power key

PwrKeyEint Gpio: 0

Power Key: POWER

□ PowerKey use EINT

□ PowerKey Gpio DIN High

Home Key: VOLUMEUP

Key_Type: NORMAL_T

Keypress_Perio: 1024

DCT (Driver Customization Tool)

- **Keypad type**

The “Key_Type” field defines whether single keypad or double keypad is used.

- **Key Mapping**

This is corresponding to Keypad matrix. If you want to map MENU key in (row1, column1) (HW keycode is 10) in Keypad matrix, you can choose “MENU” in [Row1, Column1]. DCT will generate “[10] = KEY_MENU,” in cust_kpd.h so that Keypad driver can use this mapping to report the key event when (row1, column1) is pressed or released.

DCT (Driver Customization Tool)

- **De-Bounce Time**
- The number in “Keypress_Period” divided by 32 means the Keypad’s de-bounce time (millisecond). For example, the number 1024 in “Keypress_Period” means the de-bounce time is 1024 / 32 = 32 ms.

- **Power Key Setting**
- If you want to use EINT to detect Power key’s pressing and releasing, you can check the box “PowerKey use EINT” and choose “POWER” or “ENDCALL” in “Power Key definition” so that Keypad driver can use this mapping to report the key event when EINT is issued.

Agenda

- introduction
- Architecture
- Interface
- Customization
- Build

AOSP Structure

File	Description
kernel-4.9/drivers/misc/mediatek/mach/mt6765/<project>/dct/dct/	
cust_kpd.h	The header file generated by DCT from codegen.dws
kernel-4.9/drivers/misc/mediatek/mach/mt6765/<project>/keypad/	
mtk_kpd.h	The header file used to enable/disable some functionality of Keypad driver
kernel-4.9/drivers/misc/mediatek/keypad/	
kpd.c	The core of Keypad driver
kernel-4.9/drivers/misc/mediatek/keypad/mt6765/	
hal_kpd.c	Platform-specific part of keypad driver

Build

Build Option

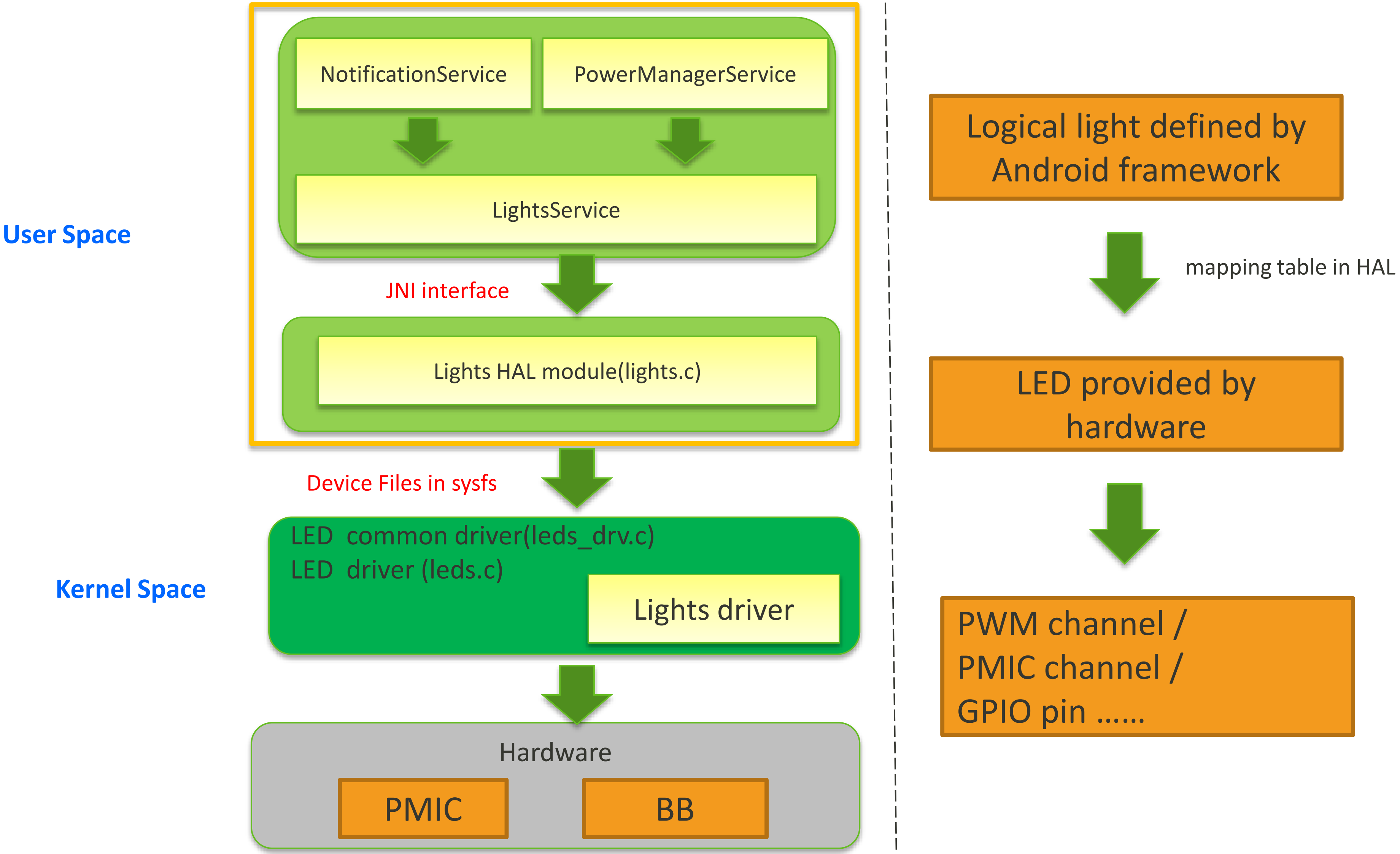
Kernel config: (Defined in kernel-4.9/drivers/misc/mediatek/keypad/Kconfig)

- CONFIG_MTK_KEYPAD
  Set to Y to enable MTK keypad driver.
- CONFIG_ONEKEY_REBOOT_NORMAL_MODE
  Set to Y to set long press reboot by power key on normal mode.
- CONFIG_ONEKEY_REBOOT_OTHER_MODE
  Set to Y to set long press reboot by power key on other boot mode.
- CONFIG_ KPD_PMIC_LPRST_TD
  Long press reset time
  0: 8 second, 1: 11 second, 2: 14 second, 3: 5 second

CONFIDENTIAL B

Lights

Lights system

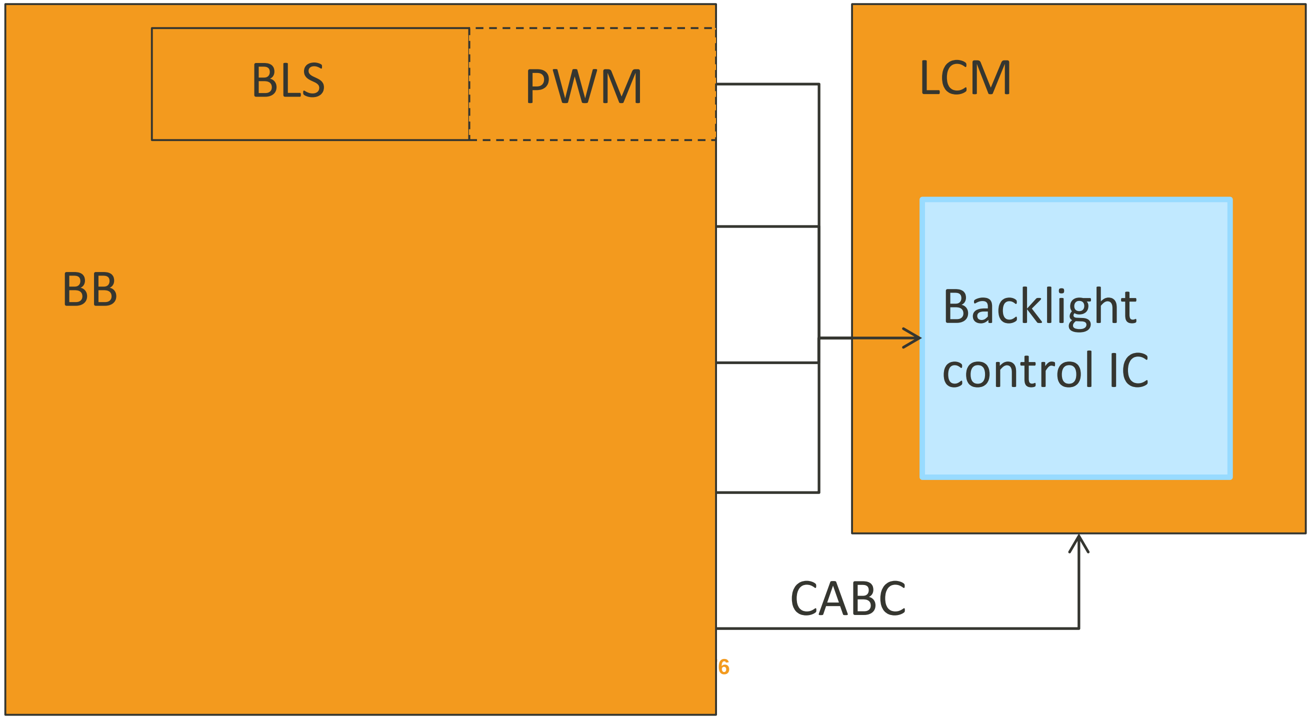


file list

- Hal
  - /vendor/mediatek/proprietary/hardware/liblights/lights.c
- Kernel
  - /kernel-*/drivers/leds/led-class.c
  - /kernel-*/drivers/leds/led-core.c
  - /kernel-*/drivers/leds/led-triggers.c
  - /kernel-*/drivers/leds/trigger/ledtrig-timer.c
- Driver
  - Kernel:
    - /kernel-*/drivers/misc/mediatek/leds/$(platform)/leds.c
    - /kernel-*/drivers/misc/mediatek/leds/mtk_leds_drv.c
  - Lk: /vendor/mediatek/proprietary/bootable/bootloader/lk/platform/$(platform)/mt_leds.c
- Customization
  - Kernel(DTS): /kernel-*/arch/arm(64)/boot/dts/$(project).dts
  - Lk: /vendor/mediatek/proprietary/bootable/bootloader/lk/target/$(project)/cust_leds.c

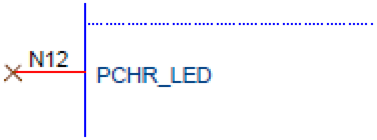
Backlight MODE Overview

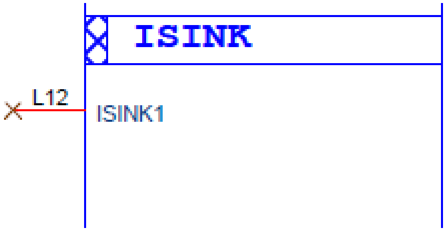
- 1. Control by PWM(pulse-width modulation)
  - Waveform produced directly by BB’s PWM module.
  - Waveform produced by PWM after BLS module.
  - PWM waveform produced by GPIO.
  - PWM waveform produced by backlight control IC ,just need write data to backlight control IC with MIPI(CABC).
- 2. Control by PMIC



Resource for lights

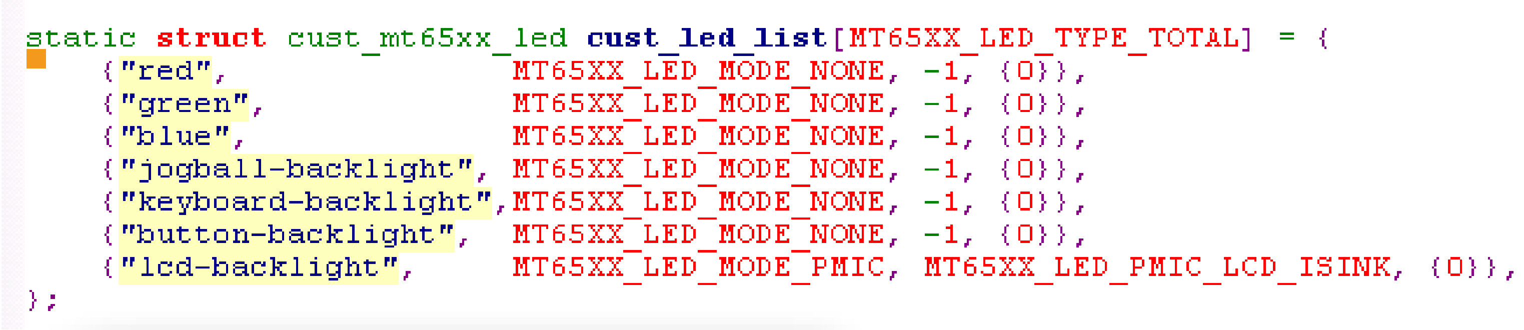
MT6357 pins



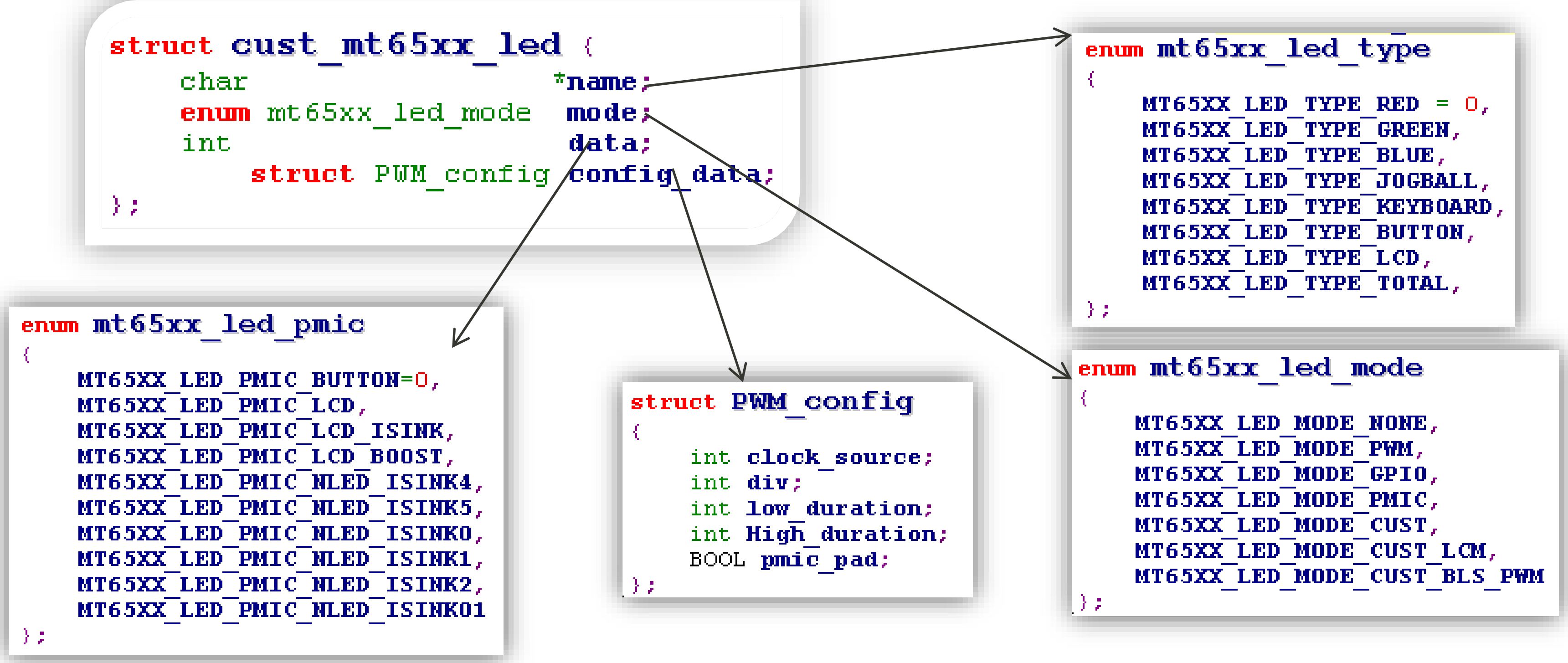


- MT6357
  - 2ISINK channels: ISINK1/PCHR_LED
    - ISINK1 provide 1current sink drivers for general LED indicator application
    - PWM mode, breathe mode , register mode can be set through SPI interface
    - PCHR_LED is default-on is default-on indicator and powered by charger plug-in, and it can set as PWM or breath mode after system power-on.
    - Output Current:
      - PCHR_LED:2/4/6 ma
      - ISINK1:2/4/6/8/10/12 ma
- $(platform)
  - 1 PWM_BL can set 1024 duty in BLS module
  - PWM
    - PWM has more available frequencies and can’t work in sleep mode
    - 2 attributes adjustable for each PWM
      - frequency & duty

Backlight Cust--LK



```
static struct cust_mt65xx_led cust_led_list[MT65XX_LED_TYPE_TOTAL] = {
    {"red",                 MT65XX_LED_MODE_NONE, -1, {0}},
    {"green",               MT65XX_LED_MODE_NONE, -1, {0}},
    {"blue",                MT65XX_LED_MODE_NONE, -1, {0}},
    {"jogball-backlight",   MT65XX_LED_MODE_NONE, -1, {0}},
    {"keyboard-backlight",  MT65XX_LED_MODE_NONE, -1, {0}},
    {"button-backlight",    MT65XX_LED_MODE_NONE, -1, {0}},
    {"lcd-backlight",       MT65XX_LED_MODE_PMIC, MT65XX_LED_PMIC_LCD_ISINK, {0}},
};
```



```
struct cust_mt65xx_led {
    char                      *name;
    enum mt65xx_led_mode       mode;
    int                        data;
        struct PWM_config config_data;
};
```

```
enum mt65xx_led_type
{
    MT65XX_LED_TYPE_RED = 0,
    MT65XX_LED_TYPE_GREEN,
    MT65XX_LED_TYPE_BLUE,
    MT65XX_LED_TYPE_JOGBALL,
    MT65XX_LED_TYPE_KEYBOARD,
    MT65XX_LED_TYPE_BUTTON,
    MT65XX_LED_TYPE_LCD,
    MT65XX_LED_TYPE_TOTAL,
};
```

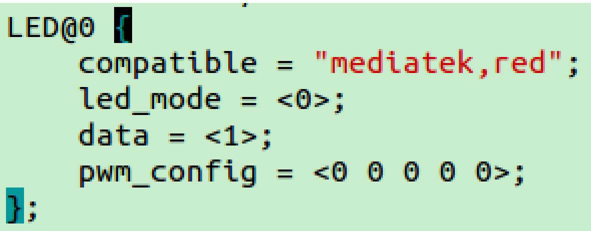
```
enum mt65xx_led_pmic
{
    MT65XX_LED_PMIC_BUTTON=0,
    MT65XX_LED_PMIC_LCD,
    MT65XX_LED_PMIC_LCD_ISINK,
    MT65XX_LED_PMIC_LCD_BOOST,
    MT65XX_LED_PMIC_NLED_ISINK4,
    MT65XX_LED_PMIC_NLED_ISINK5,
    MT65XX_LED_PMIC_NLED_ISINK0,
    MT65XX_LED_PMIC_NLED_ISINK1,
    MT65XX_LED_PMIC_NLED_ISINK2,
    MT65XX_LED_PMIC_NLED_ISINK01
};
```

```
struct PWM_config
{
    int clock_source;
    int div;
    int low_duration;
    int High_duration;
    BOOL pmic_pad;
};
```

```
enum mt65xx_led_mode
{
    MT65XX_LED_MODE_NONE,
    MT65XX_LED_MODE_PWM,
    MT65XX_LED_MODE_GPIO,
    MT65XX_LED_MODE_PMIC,
    MT65XX_LED_MODE_CUST,
    MT65XX_LED_MODE_CUST_LCM,
    MT65XX_LED_MODE_CUST_BLS_PWM
};
```

Customization--kernel

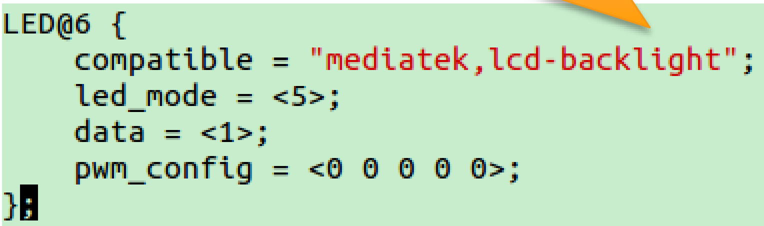
- Lights System Device Tree Customer Parameter
  - M:Kernel4.xx/arch/arm(xx)/boot/dts/$(project).dts



```
LED@0 {
    compatible = "mediatek,red";
    led_mode = <0>;
    data = <1>;
    pwm_config = <0 0 0 0 0>;
};
```

......

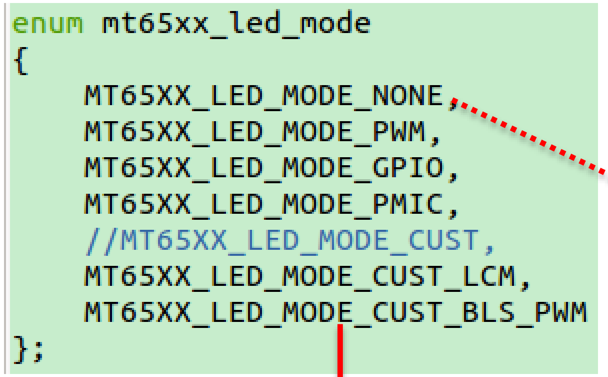
Customer can modify dts to set the parameter what they want



```
LED@6 {
    compatible = "mediatek,lcd-backlight";
    led_mode = <5>;
    data = <1>;
    pwm_config = <0 0 0 0 0>;
};
```

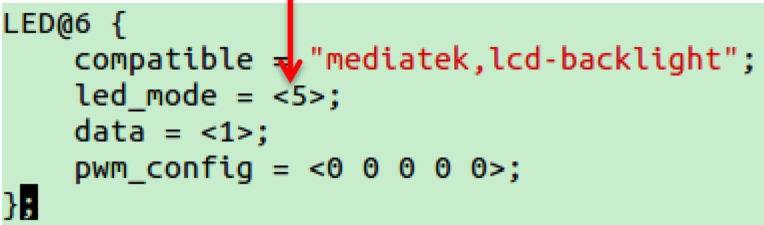
Customization--Set the LED mode in device tree

<cust_leds_def.h>



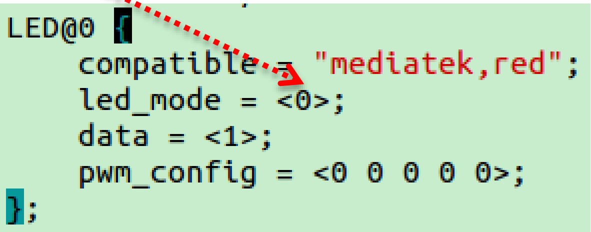
```
enum mt65xx_led_mode
{
    MT65XX_LED_MODE_NONE,
    MT65XX_LED_MODE_PWM,
    MT65XX_LED_MODE_GPIO,
    MT65XX_LED_MODE_PMIC,
    //MT65XX_LED_MODE_CUST,
    MT65XX_LED_MODE_CUST_LCM,
    MT65XX_LED_MODE_CUST_BLS_PWM
};
```

- Led_mode is determining the **HW type** , such as the PWM, PMIC , BLS_PWM etc.



```
LED@6 {
    compatible = "mediatek,lcd-backlight";
    led_mode = <5>;
    data = <1>;
    pwm_config = <0 0 0 0 0>;
};
```

<$(project).dts>



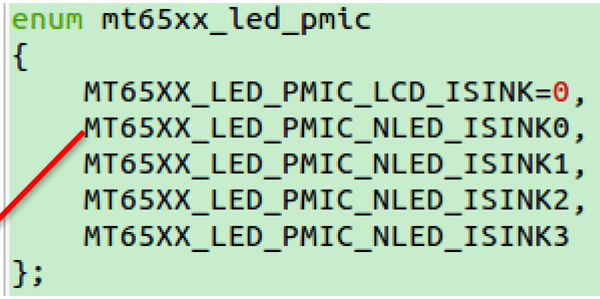
```
LED@0 {
    compatible = "mediatek,red";
    led_mode = <0>;
    data = <1>;
    pwm_config = <0 0 0 0 0>;
};
```

<$(project).dts>

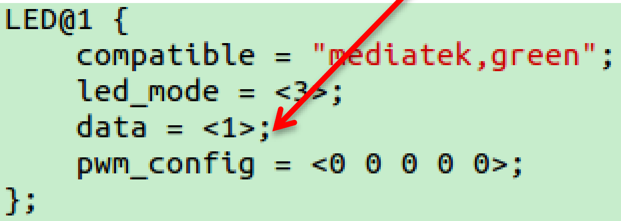
Customize the LED pin Num(1/2)

- If your led mode is MT65XX_LED_MODE_PMIC, you can use the data field to choose the pin num, such as ISINK0

<cust_leds_def.h>



```
enum mt65xx_led_pmic
{
    MT65XX_LED_PMIC_LCD_ISINK=0,
    MT65XX_LED_PMIC_NLED_ISINK0,
    MT65XX_LED_PMIC_NLED_ISINK1,
    MT65XX_LED_PMIC_NLED_ISINK2,
    MT65XX_LED_PMIC_NLED_ISINK3
};
```



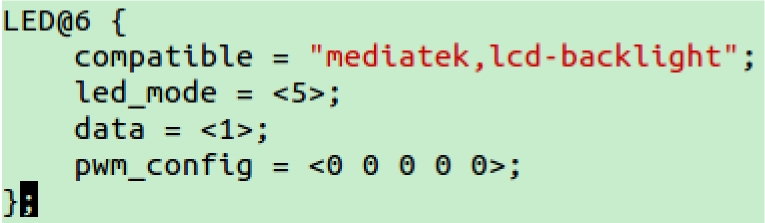
```
LED@1 {
    compatible = "mediatek,green";
    led_mode = <3>;
    data = <1>;
    pwm_config = <0 0 0 0 0>;
};
```

<$(project).dts>

Customiz the LED pin Num(2/2)

- But， if you hw type is MT65XX_LED_MODE_CUST_BLS_PWM or MT65XX_LED_MODE_CUST_LCM , the data region is **not used** any more, the function point will be assigned in driver dynamically.

Android M



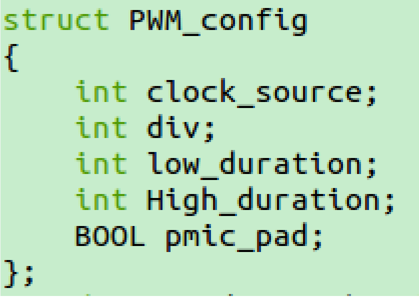
```
LED@6 {
    compatible = "mediatek,lcd-backlight";
    led_mode = <5>;
    data = <1>;
    pwm_config = <0 0 0 0 0>;
};
```

<$(project).dts>

Customize the LED frequency

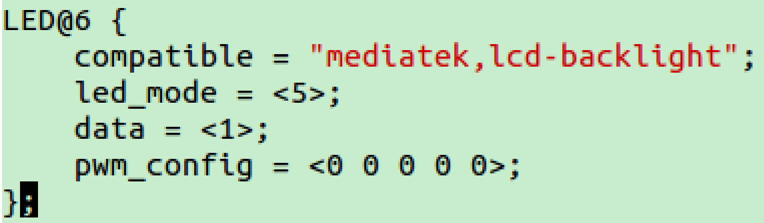
The LED&Backlight frequency setting is the same the the cust_leds.c

<cust_leds_def.h>



```
struct PWM_config
{
    int clock_source;
    int div;
    int low_duration;
    int High_duration;
    BOOL pmic_pad;
};
```

<$(project).dts> Android M



```
LED@6 {
    compatible = "mediatek,lcd-backlight";
    led_mode = <5>;
    data = <1>;
    pwm_config = <0 0 0 0 0>;
};
```

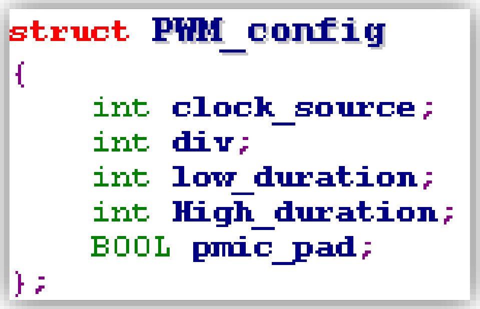
How to Get Device Tree Parameter

PATH: Kernel4.xx/drivers/misc/mediatek/leds/$(platform)/leds.c

Function : struct cust_mt65xx_led ***get_cust_led_dtsi**(void);

BB_PWM cust

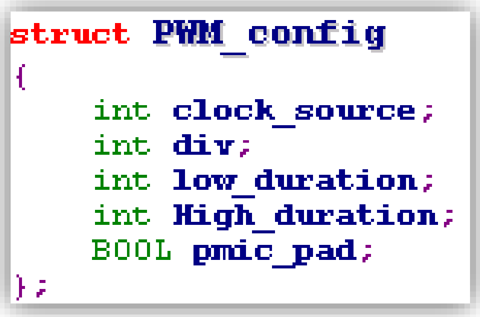
- PWM(pulse-width modulation) means control the wave's duty or wave's counts to control led brightness.
- Config parameters data & config_data for customization.
- If the parameters is 0, it use default parameters.
- PWM config data
  - clock_source: clock source frequency, can be 0/1
  - div: clock division, can be any value within 0~7 (i.e. 1/2^(div) = /1, /2, /4, /8, /16, /32, /64, /128)
  - low_duration: duration of low level. Only for FIFO MODE
  - High_duration: duration of high level. Only for FIFO MODE
- PWM freq
  - PWM freq. = clock source / 2^(div) / 256 for **old mode**
  - PWM freq. = clock source / 2^(div) / [(High_duration+1)(Level')+(low_duration+1)(64 - Level')] for **FIFO mode**



```
struct PWM_config
{
      int clock_source;
      int div;
      int low_duration;
      int High_duration;
      BOOL pmic_pad;
};
```

BLS_PWM cust

- for BLS PWM setting as follow:
  - 1. PWM config data
    - clock_source: clock source frequency, can be 0/1/2/3
    - div: clock division, can be any value within 0~1023
    - low_duration: non-use
    - High_duration: non-use
    - pmic_pad: non-use
  - 2.PWM freq.= clock source / (div + 1) /1024
    - Clock source:
    - 0: PWM_CLK_OLD_MODE_BLOCK
    - 1: PWM_CLK_OLD_MODE_32K



```
struct PWM_config
{
    int clock_source;
    int div;
    int low_duration;
    int High_duration;
    BOOL pmic_pad;
};
```

Debug Skills

- We can get LED’s or backlight’s brightness status by follow files：
  - /sys/class/leds/xxx/brightness
  - Xxx means light’s name， such as: green,blue,red mean green led,blue led and red led. Lcd-backlight means backlight.
  - If the led or backlight is not working properly， we can enter the corresponding folder and execute some command to check the driver is correctly or not：
    - echo brightness_level > brightness →check the brightness is the setting value or not.
    - For LED, brightness level is not zero if only， driver will set led on.
    - If you want LED blink, you can execute the follow command in corresponding folder:
      - echo timer > trigger →this command will create two files: delay_on and delay_off.
      - echo on_time > delay_on →this command use to set led_on time.
      - echo off_time > delay_off →this command use to set led_off time.
      - The unit of led_on and led_off is millisecond.
      - echo none > trigger →this command will delete delay_on file and delay_off file,moreover,it will turn off the led.
    - Backlight not support blink,

CONFIDENTIAL B

Headset

Agenda

- Feature
- Detect Flow
- SW architecture
- SW Customization

Accessory Detection -Feature

- Three accessory detection solutions are provided
  - **Traditional mode (ACC)**
  - Low cost mode without internal Bias
  - Low cost mode with internal Bias
- **Please refer to DCC document for detail:**

  MT6357 Audio_Speech Design Note_V0.1.pptx

Accessory Detection –Feature detail

- /kernel-4.9/drivers/misc/mediatek/accdet/mt6357/accdet.c
- CONFIG_ACCDET_EINT_IRQ

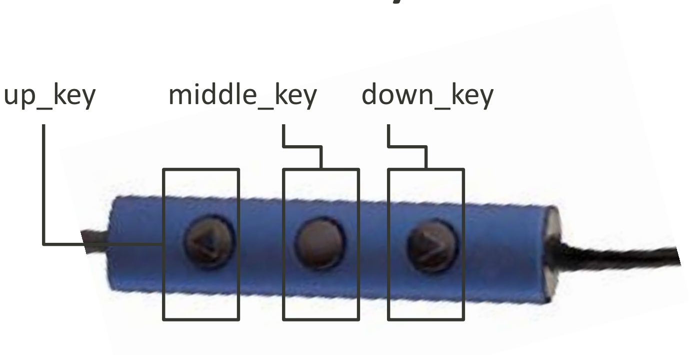
  耳机插拔中断接的PMIC

  - CONFIG_ACCDET_EINT

  耳机插拔中断接的AP

Accessory Detection –Feature detail

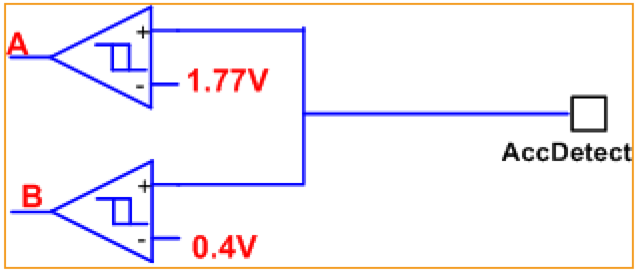
- multi-key function



	Remote button function(multi-key)		
场景	播放音乐	电话通话中	来电话接听前
up_key	上一首	Volume up	Volume up
middle_key	播放/ 暂停	挂断（长按） 切换通话（短按）	挂断（长按） 接听（短按）
down_key	下一首	Volume down	Volume down

Accessory Detection-Detect Flow

- Accdet Hardware Design:
  - Accessory detecting depends on the voltage when 3-pole or 4 pole headset plug in/out, and it uses internal 2-bit comparator to separate what kinds of external components are.



    - If the voltage of AccDet is higher than 1.77V, A=1; or else, A=0;
    - If the voltage of AccDet is higher than 0.4V, B=1; or else, B=0.
    - So AccDet is divided into 3 headset state according to the voltage range:
      - **Plug out state**: 1.77V ≤ Voltage ≤1.9V (**A=1, B=1**);
      - **Mic Bias state**: 0.4V ≤ Voltage<1.77V (**A=0, B=1**);
      - **Hook Switch state**: 0V ≤ Voltage < 0.4V (**A=0, B=0**).

Accessory Detection—State Machine

Step1: plug out→plug in

①: Plug in 4-pole headset;

A=0, B=1 (0.5V ≤ Voltage<1.77V);

⑤ : Plug in 3-pole headset;

A=0, B=0 (Voltage = 0 V);

Step2: plug in→plug out

④: Plug out 4-pole headset.

A=1, B=1 (Voltage = 1.9V).

⑥ : Plug out 3-pole headset.

A=1, B=1 (Voltage = 1.9V).

Step3: press remote button

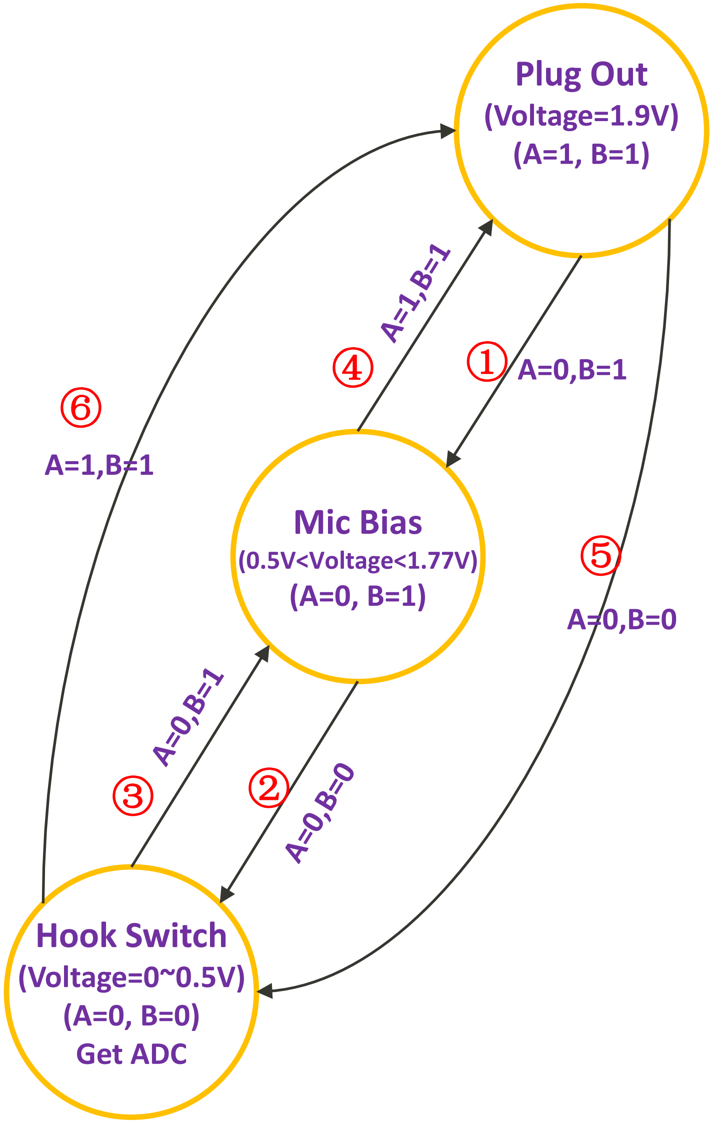
②: Press remote button;

A=0, B=0 (Voltage = 0~0.5 V);

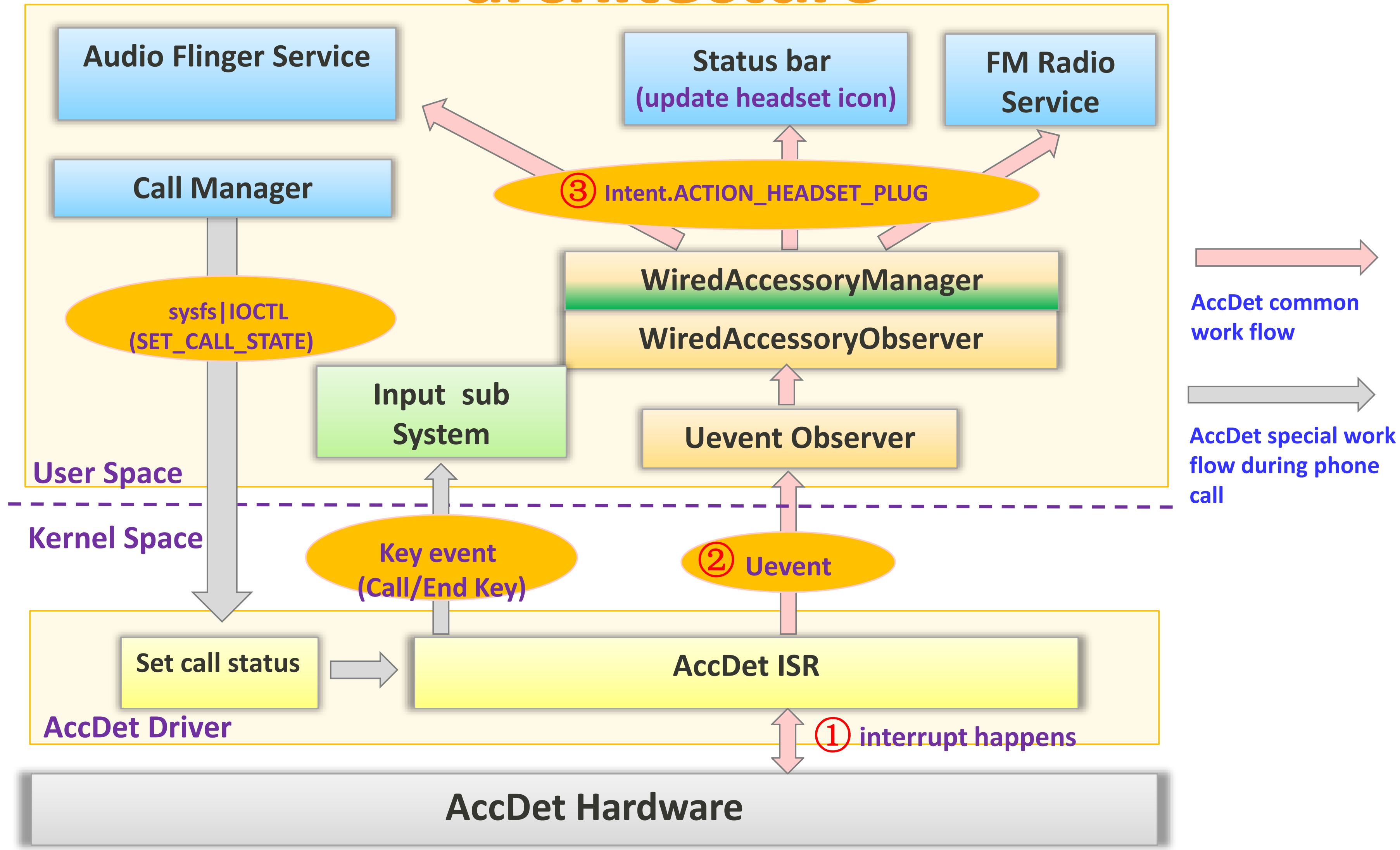
Step4: Release remote button

③: Release remote button;

A=0, B=1 (0.5V ≤ Voltage<1.77V);



Accessory Detection-SW architecture



Accessory Detection-SW architecture file list

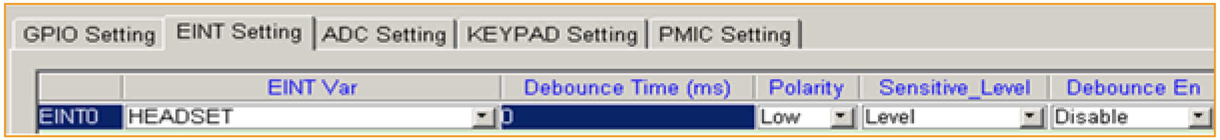
- User Space
  - WiredAccessoryManager.java
  - AudioManager.java
  - PhoneStatusBarPolicy.java
  - FMRadioService.java
  - AudioMTKHeadsetMessager.cpp
  - Ftm_heaset.cpp

- Driver
  - accdet.c
  - accdet_irq.c
  - accdet_drv.c
  - accdet_custom.c
  - accdet_custom_def.h

Accessory Detection-SW Customization

- File list

- Customization item
  - Dts file
  - DCT config



	EINT Var	Debounce Time (ms)	Polarity	Sensitive_Level	Debounce En
EINT0	HEADSET	0	Low	Level	Disable

CONFIDENTIAL B

Sensor&Sensor Hub

参考文件以及路径

- 请参考DCC上面如下路径下的如下文件：
- 路径：
  - SW/3G-4G/Smart Phone/Document Library New/BSP/Drivers
- 文件名：
  - CS6000-AZ6A-PGD-V2.0EN_Sensors_Programming Guide.pdf

CONFIDENTIAL B

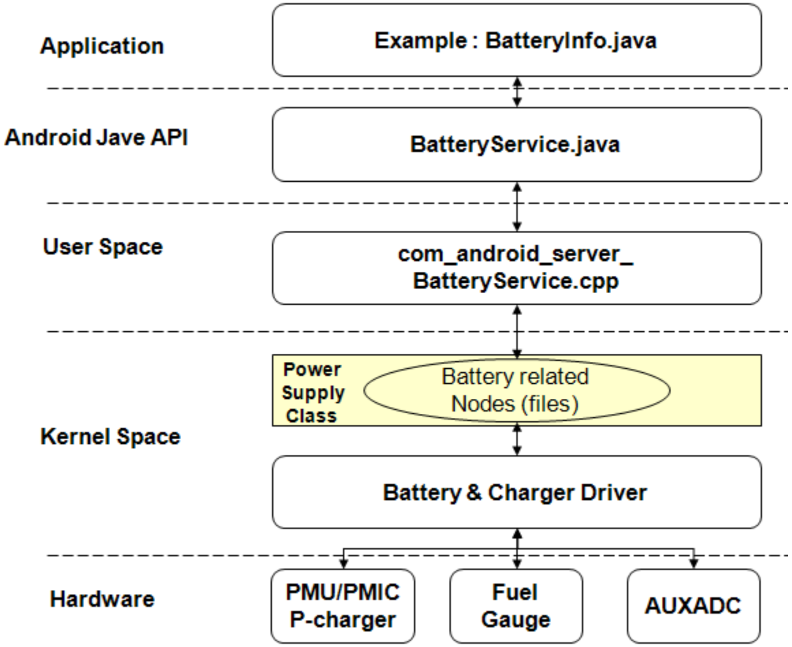
Battery Management

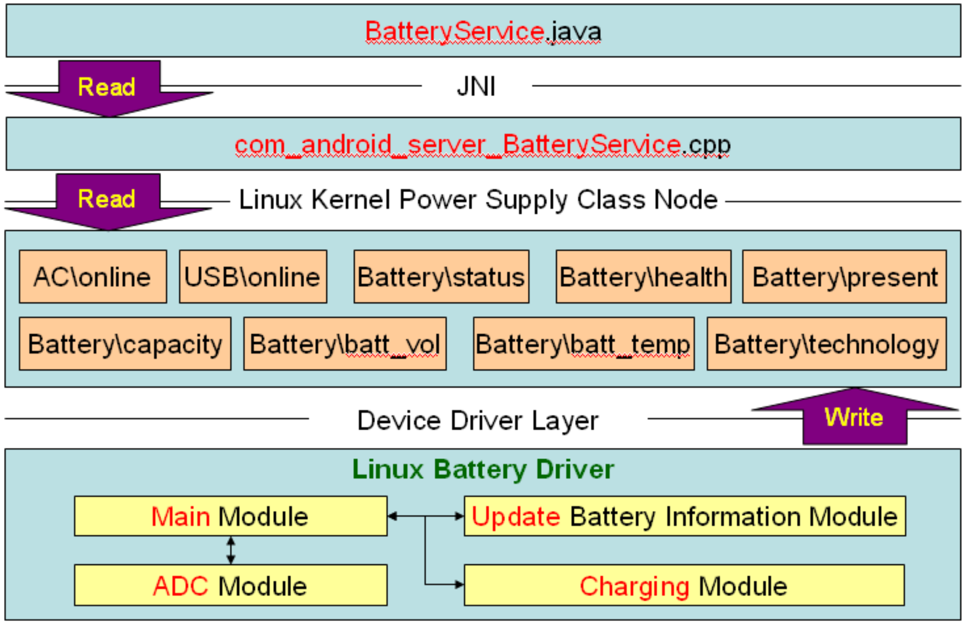
Outline

- **Battery Service**
- Battery Charging Overview
- Kernel Power Off Charging
- Fuel Gauge

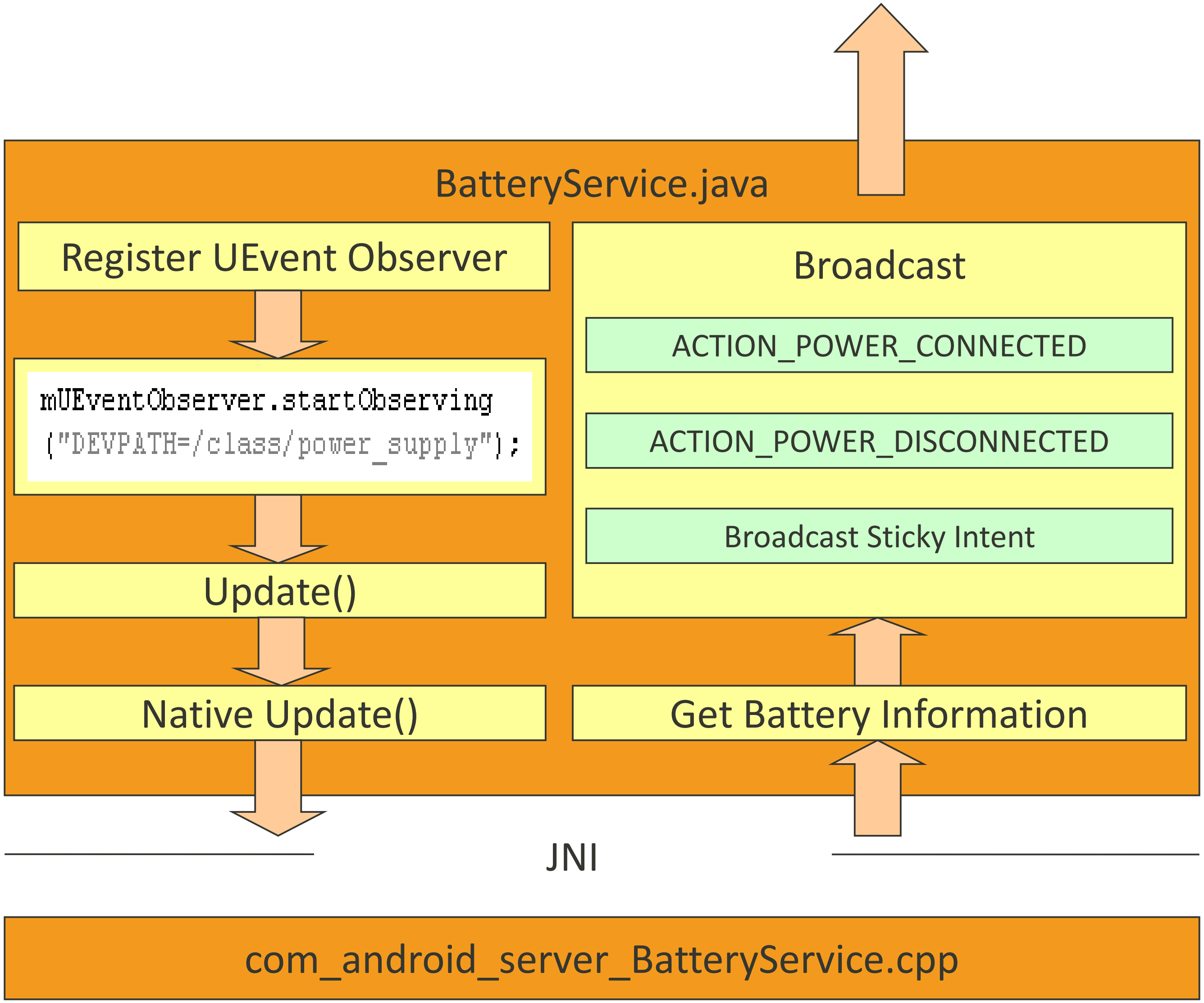
Battery Introduction

- Introduction

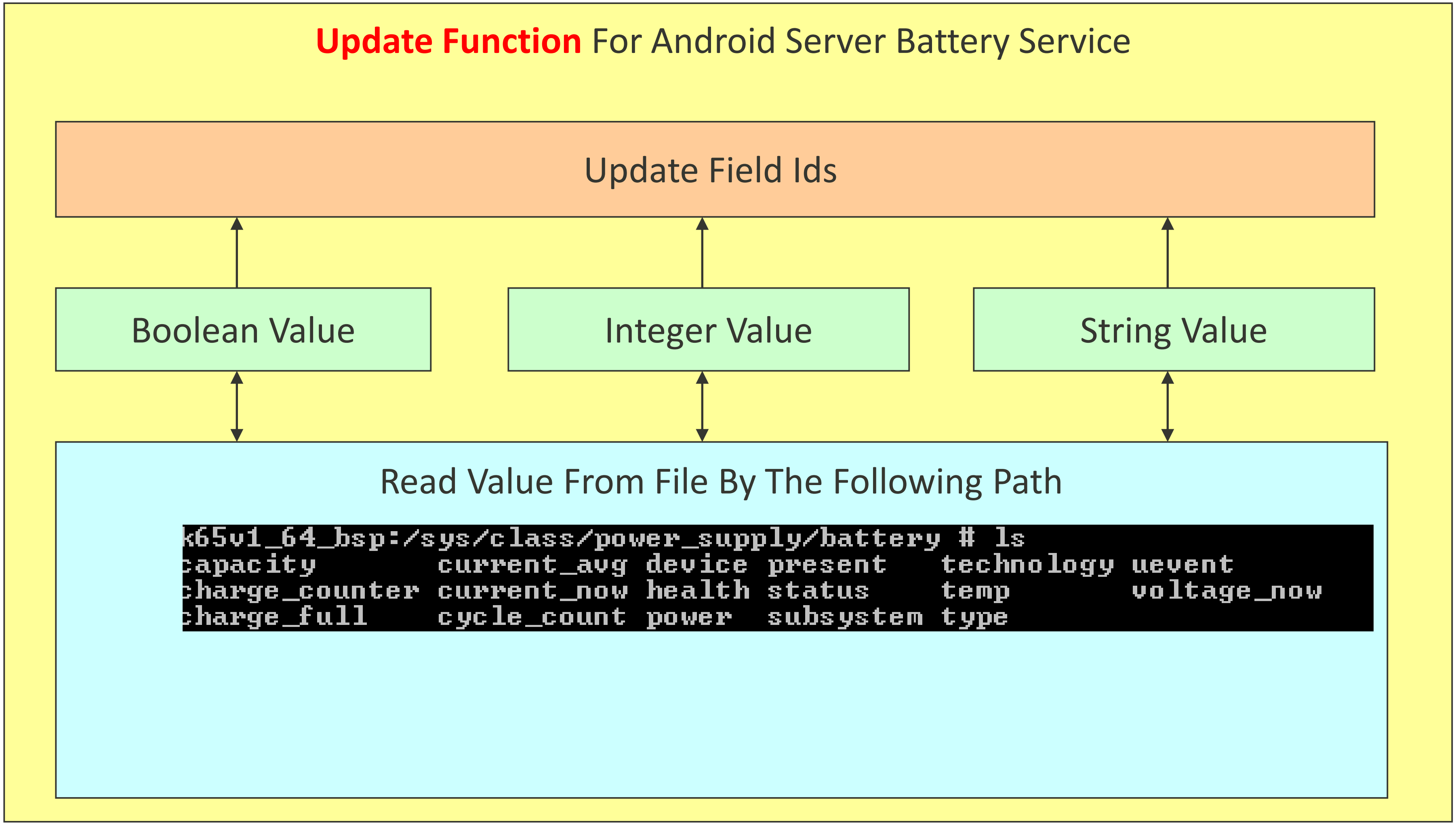




BatteryService.java



Battery Information Update Function



Update Function For Android Server Battery Service

Update Field Ids

Boolean Value

Integer Value

String Value

Read Value From File By The Following Path

```
k65v1_64_bsp:/sys/class/power_supply/battery # ls
capacity       current_avg device present   technology uevent
charge_counter current_now health status    temp       voltage_now
charge_full    cycle_count power  subsystem type
```

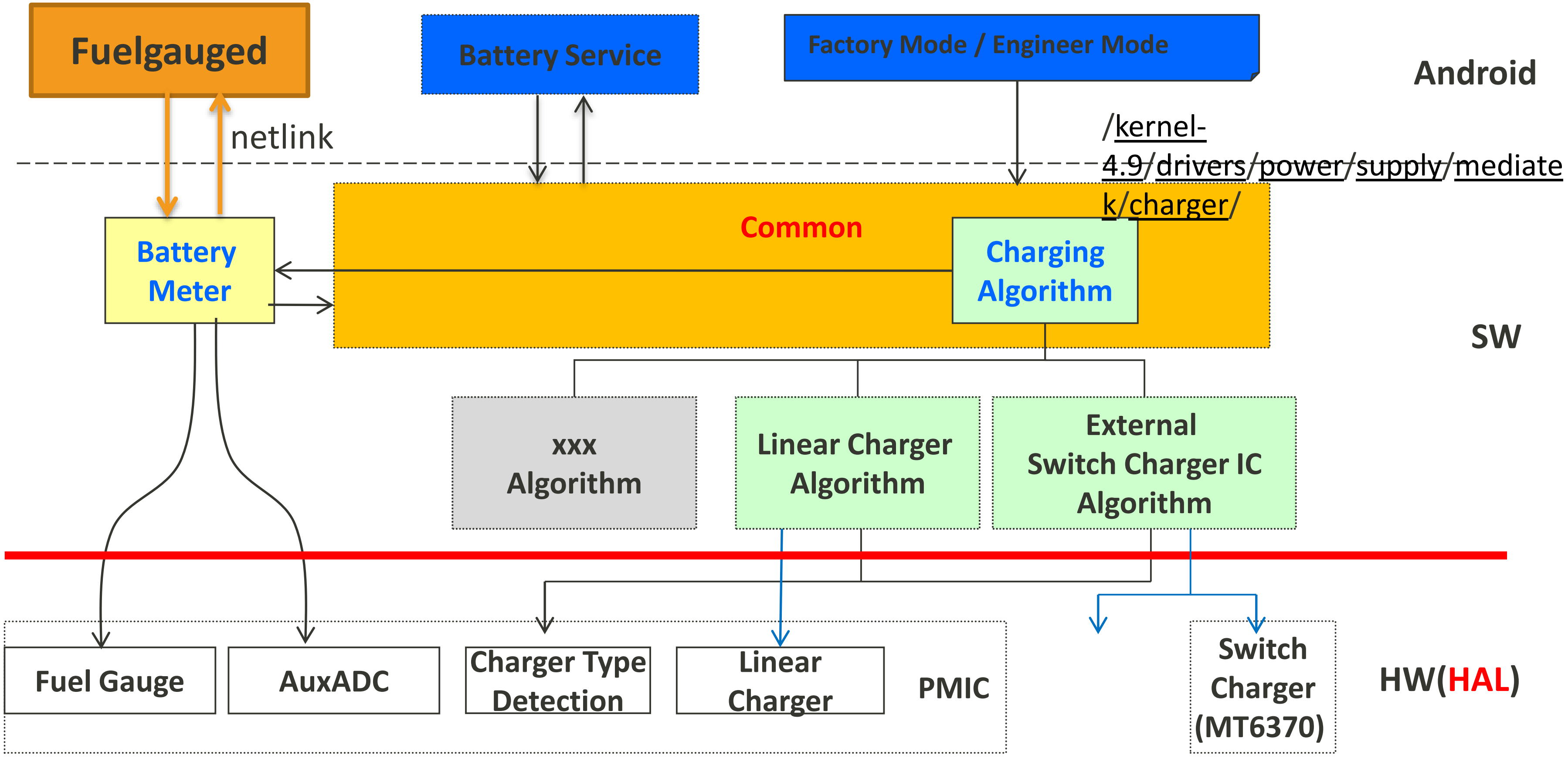
MEDIATEK

CONFIDENTIAL B

Outline

- Battery Service
- **Battery Charging Overview**
- Kernel Power Off Charging
- Fuel Gauge

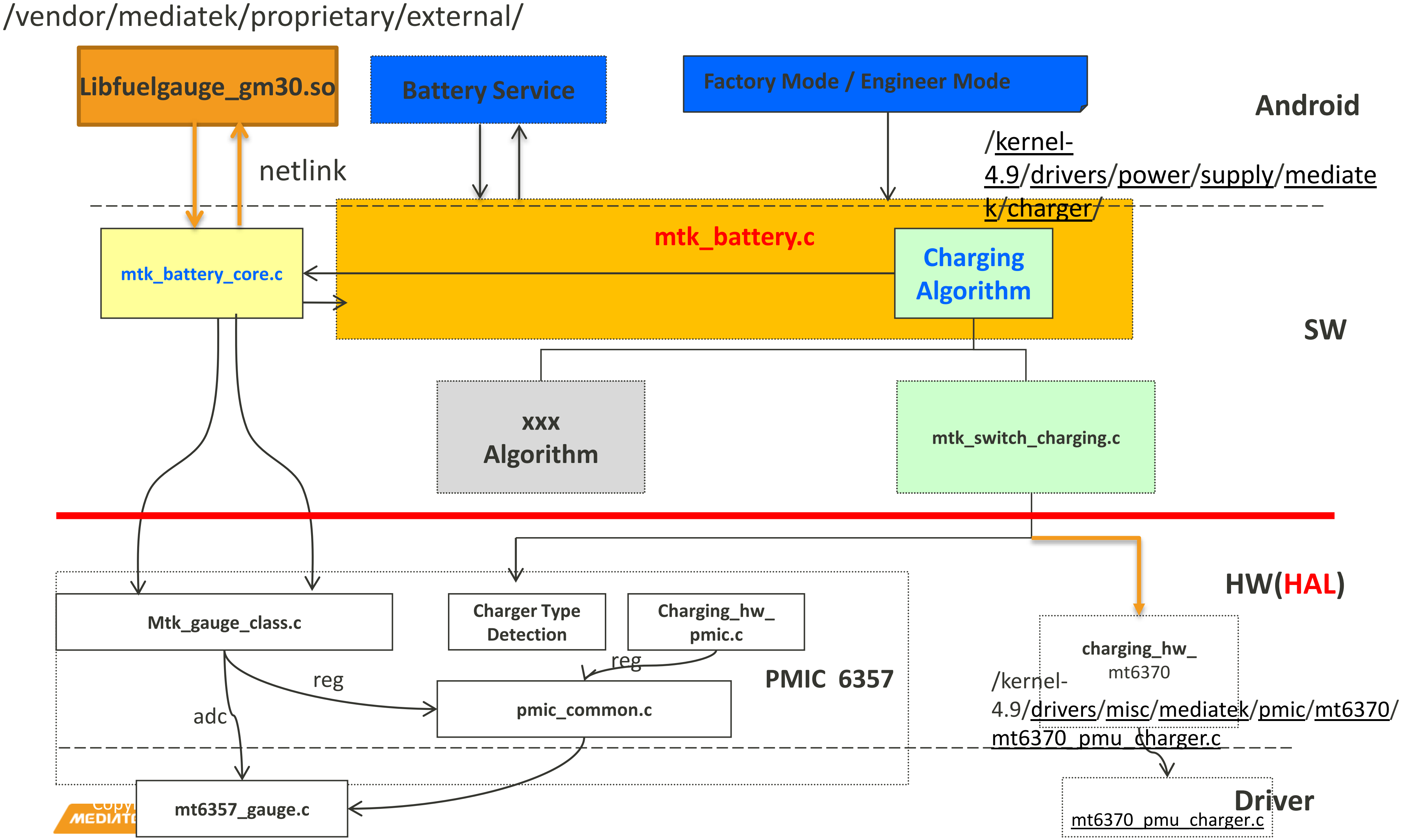
Battery 3.0 SW Architecture



/kernel-4.9/drivers/power/supply/mediatek/charger/

/kernel-4.9/drivers/misc/mediatek/pmic/mt6370/mt6370_pmu_charger.c

Kernel Layer-Battery 3.0 SW Code Architecture



/vendor/mediatek/proprietary/external/

Libfuelgauge_gm30.so

Battery Service

Factory Mode / Engineer Mode

Android

netlink

/kernel-4.9/drivers/power/supply/mediatek/charger/

mtk_battery.c

mtk_battery_core.c

Charging Algorithm

SW

xxx Algorithm

mtk_switch_charging.c

HW(HAL)

Mtk_gauge_class.c

Charger Type Detection

Charging_hw_pmic.c

reg

reg

adc

pmic_common.c

PMIC 6357

charging_hw_mt6370

/kernel-4.9/drivers/misc/mediatek/pmic/mt6370/mt6370_pmu_charger.c

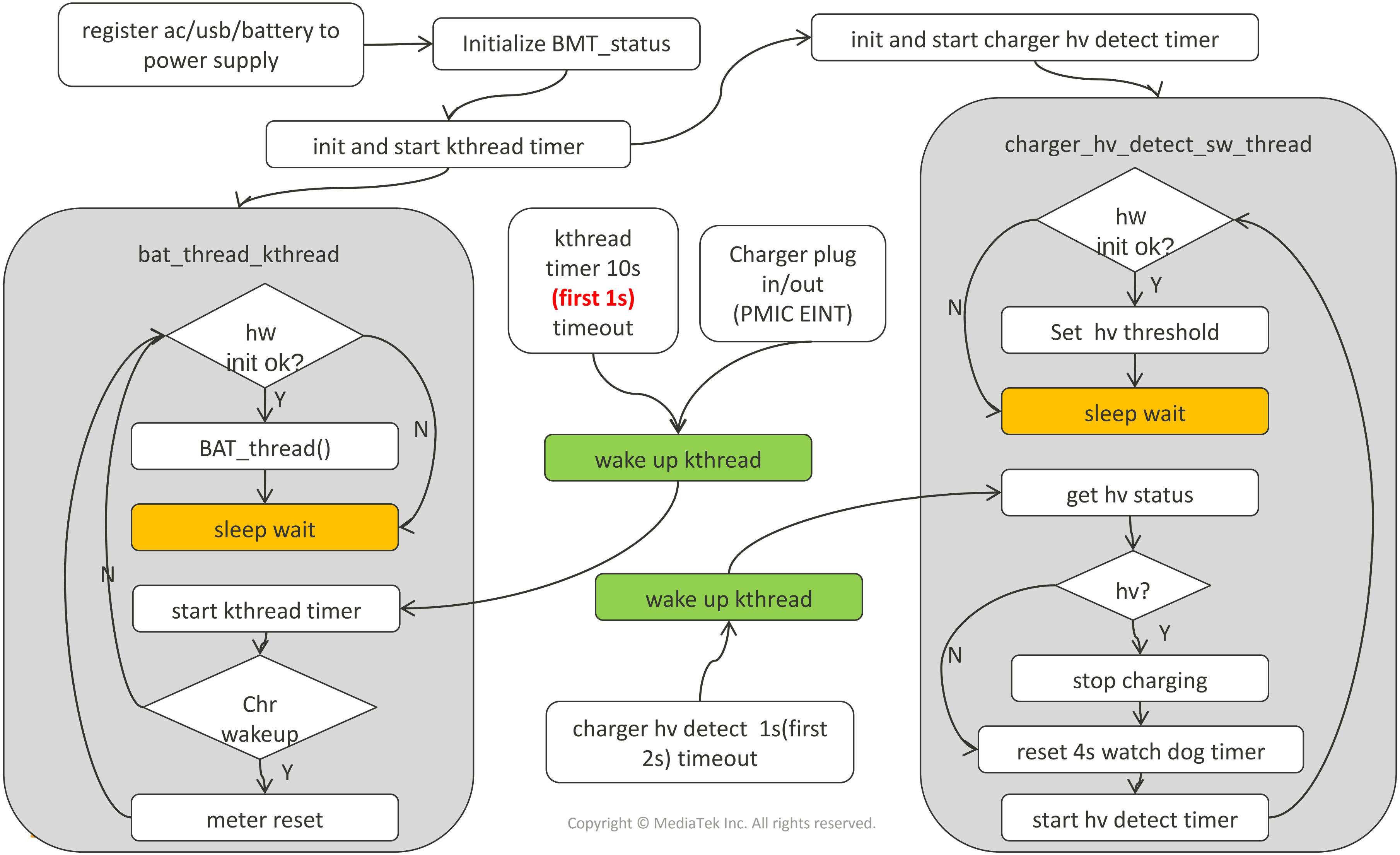
mt6357_gauge.c

Driver

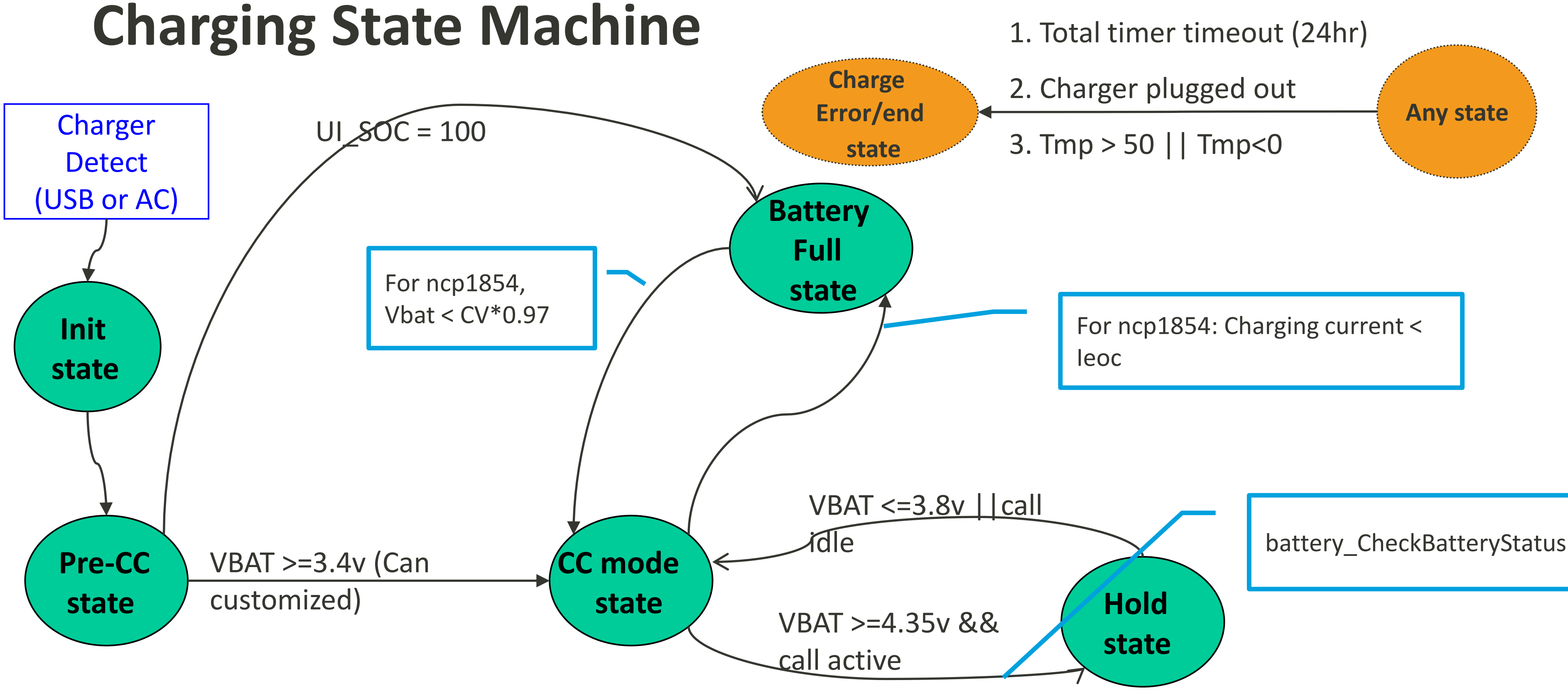
mt6370_pmu_charger.c

Power On Charging(1/x)

Whole Flow



Charging State Machine



1. Total timer timeout (24hr)

2. Charger plugged out

3. Tmp > 50 || Tmp<0

Any state

Charge Error/end state

Charger Detect (USB or AC)

Init state

Pre-CC state

UI_SOC = 100

For ncp1854, Vbat < CV*0.97

Battery Full state

For ncp1854: Charging current < Ieoc

VBAT >=3.4v (Can customized)

CC mode state

VBAT <=3.8v ||call idle

battery_CheckBatteryStatus

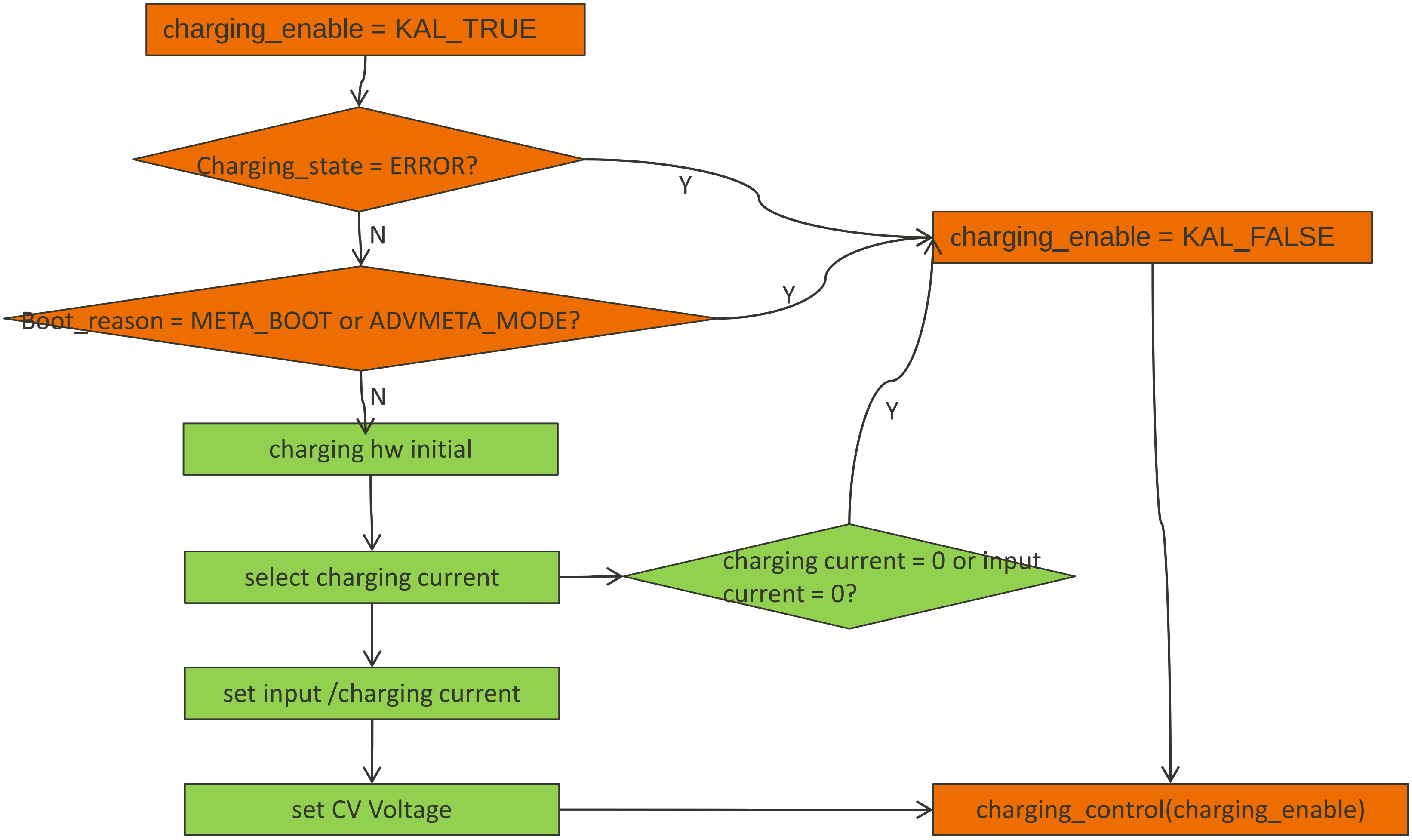
VBAT >=4.35v && call active

Hold state

Init	Pre-CC/CC Mode	TopOff Mode (CV mode)	Battery Full
1. Init the safety timers 2. Check the VBAT, decide to enter CC mode or battery full state	1. Do Charger Protection 2. If (1) fail, stop charging 3. If (2) pass, do charging ➔ Internal P-charger : charging 9s and discharging 1s. (cooling the battery) ➔ External switching charger : always charging Note : Measuring VBA I_charging / V_charger / Bat_temp at charging phase	1. Do Charger Protection 2. If (1) fail, stop charging 3. If (2) pass, do charging algo. : charging 10s Note : Measuring VBAT / I_charging / V_charger / Bat_temp at charging phase	1. Battery percentage always display 100% Note : When recharging, the battery percentage is still 100%.

Sw Flow

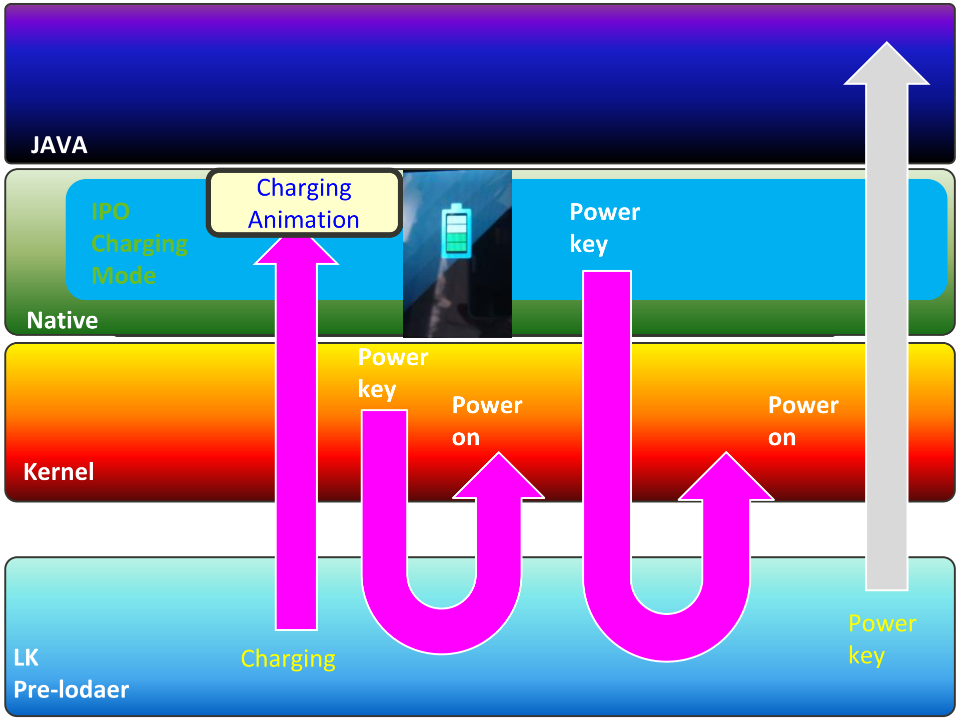
pchr_turn_on_charging()


charging_enable = KAL_TRUE
Charging_state = ERROR?
Y
N
charging_enable = KAL_FALSE
Boot_reason = META_BOOT or ADVMETA_MODE?
Y
N
Y
charging hw initial
select charging current
charging current = 0 or input current = 0?
set input /charging current
set CV Voltage
charging_control(charging_enable)

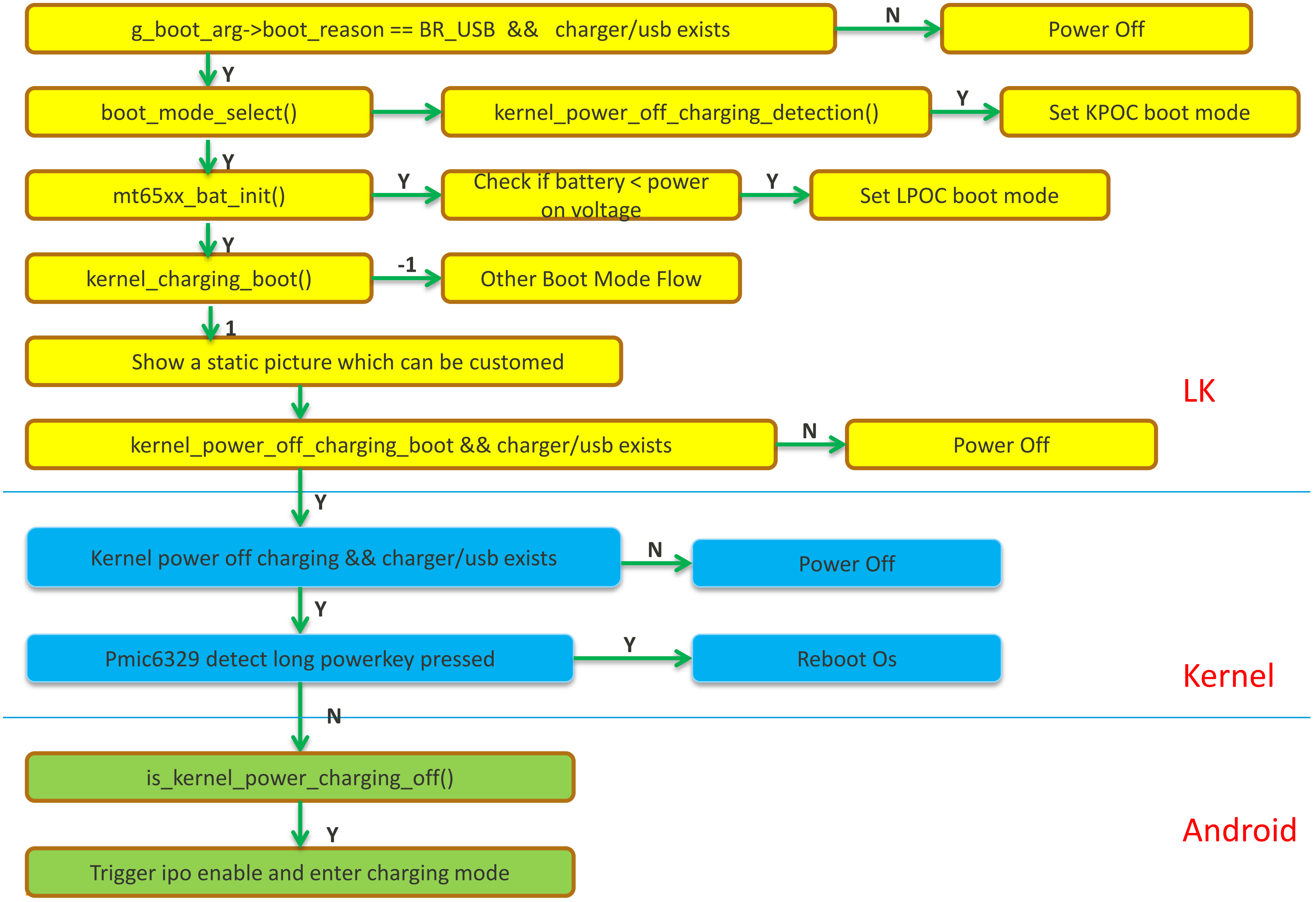
Outline

- Battery Service
- Battery Charging Overview
- **Kernel Power Off Charging**
- Fuel Gauge

MTK Kernel Power Off Charging



Kernel Power Off Charging Flow



g_boot_arg->boot_reason == BR_USB && charger/usb exists

Power Off

boot_mode_select()

kernel_power_off_charging_detection()

Set KPOC boot mode

mt65xx_bat_init()

Check if battery < power on voltage

Set LPOC boot mode

kernel_charging_boot()

Other Boot Mode Flow

Show a static picture which can be customed

kernel_power_off_charging_boot && charger/usb exists

Power Off

LK

Kernel power off charging && charger/usb exists

Power Off

Pmic6329 detect long powerkey pressed

Reboot Os

Kernel

is_kernel_power_charging_off()

Trigger ipo enable and enter charging mode

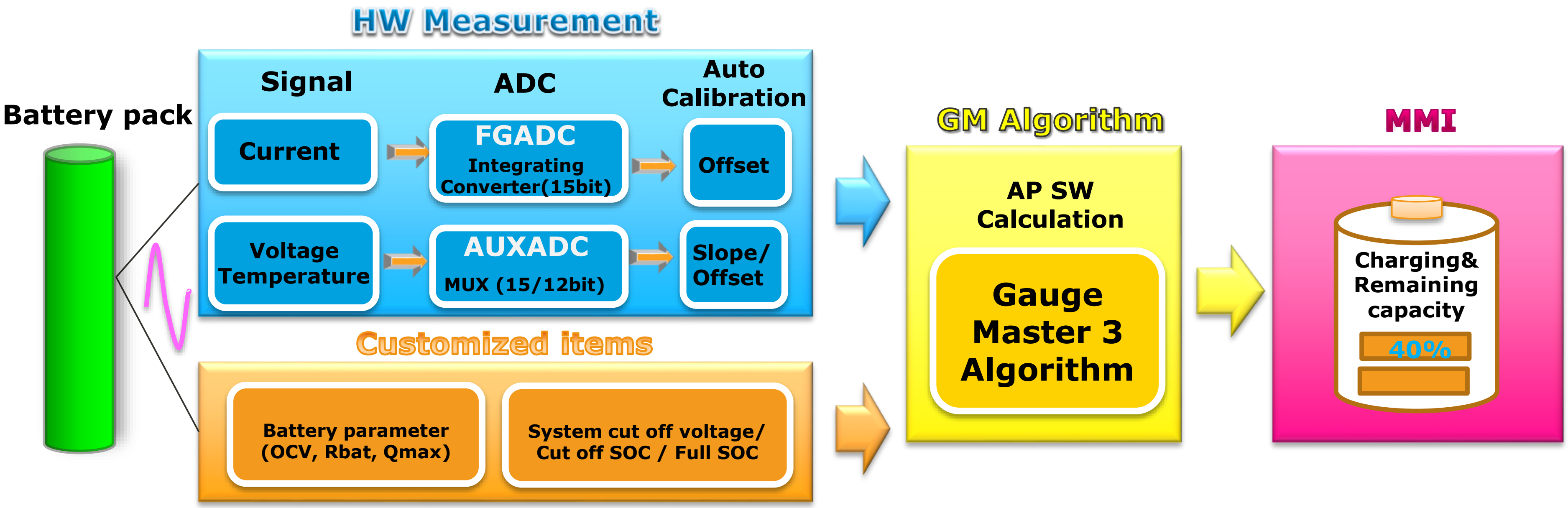
Android

Outline

- Battery Service
- Battery Charging Overview
- Kernel Power Off Charging
- **Fuel Gauge**

Gauge Master 3.0 Introduce

- System-side Li-Ion battery fuel gauge SOC
  - Precise Battery Fuel Gauge
  - Battery current measurement
  - Temperature Reporting



Preloader

Repo: alps/vendor/mediatek/proprietary/bootable/bootloader/preloader

platform/mt6765/src/drivers/battery.c

platform/mt6765/src/drivers/inc/platform.h

platform/mt6765/src/drivers/platform.c

LK

Repo: alps/vendor/mediatek/proprietary/bootable/bootloader/lk

- app/mt_boot/mt_boot.c
- platform/mt6765/include/platform/boot_mode.h
- platform/mt6765/platform.c
- platform/mt6765/rules.mk
- platform/common/power/mtk_battery.h
- platform/common/power/mtk_battery.c
- platform/common/power/rules.mk
- platform/mt6765/include/platform/mt_battery.h
- platform/mt6765/include/platform/mt_pmic_dlpt.h
- platform/mt6765/mt_battery.c
- platform/mt6765/mt_gauge.c
- platform/mt6765/mt_pmic_dlpt.c

kernel

Repo: alps/kernel-4.4

- arch/arm64/configs/k65v1_64_bsp_debug_defconfig

64bit dtsi

- arch/arm64/boot/dts/mediatek/mt6765.dts
- arch/arm64/boot/dts/mediatek/evb6765_64_emmc.dts
- arch/arm64/boot/dts/mediatek/bat_setting/m6765_battery_prop.dtsi
- arch/arm64/boot/dts/mediatek/bat_setting/6765_battery_prop_ext.dtsi
- arch/arm64/boot/dts/mediatek/bat_setting/mt6765_battery_table.dtsi
- arch/arm64/boot/dts/mediatek/bat_setting/mt6765_battery_table_ext.dtsi

32bit dtsi

- arch/arm/boot/dts/mt6765.dts
- arch/arm/boot/dts/mediatek/evb6765_64_emmc.dts
- arch/arm/boot/dts/mediatek/bat_setting/mt6765_battery_prop.dtsi
- arch/arm/boot/dts/mediatek/bat_setting/mt6765_battery_prop_ext.dtsi
- arch/arm/boot/dts/mediatek/bat_setting/m6765_battery_table.dtsi
- arch/arm/boot/dts/mediatek/bat_setting/mt6765_battery_table_ext.dtsi

kernel

Repo: alps/kernel-4.4

Platform header

- drivers/misc/mediatek/include/mt-plat/mtk_battery.h
- drivers/misc/mediatek/include/mt-plat/mt6765/include/mach/mtk_battery_property.h
- drivers/misc/mediatek/include/mt-plat/mt6765/include/mach/mtk_battery_table.h

PMIC HAL

- drivers/misc/mediatek/pmic/mtk_gauge_class.c
- drivers/misc/mediatek/pmic/mtk_gauge_coulomb_service.c
- drivers/misc/mediatek/pmic/mtk_battery_adc_intf.c
- drivers/misc/mediatek/pmic/mt6357/v1/mt6357_gauge.c

Battery Core

- drivers/power/supply/battery/mtk_battery.c
- drivers/power/supply/battery/mtk_battery_internal.h
- drivers/power/supply/battery/mtk_battery_recovery.c
- drivers/power/supply/battery/mtk_battery_recovery.h
- drivers/power/supply/battery/mtk_gauge_time_service.c
- drivers/power/supply/battery/mtk_power_misc.c

selinux

Repo: alps/device/mediatek/sepolicy/basic

device/mediatek/sepolicy/basic/non_plat/file_contexts
device/mediatek/sepolicy/basic/non_plat/fuelgauged.te
device/mediatek/sepolicy/basic/non_plat/fuelgauged_nvram.te

device/mediatek/sepolicy/basic/plat_private/fuelgauged_static.te
device/mediatek/sepolicy/basic/plat_public/fuelgauged_static.te

Gauge Master 2.0 Custom Items

- GM2.0 config CONFIG_MTK_HAFG_20

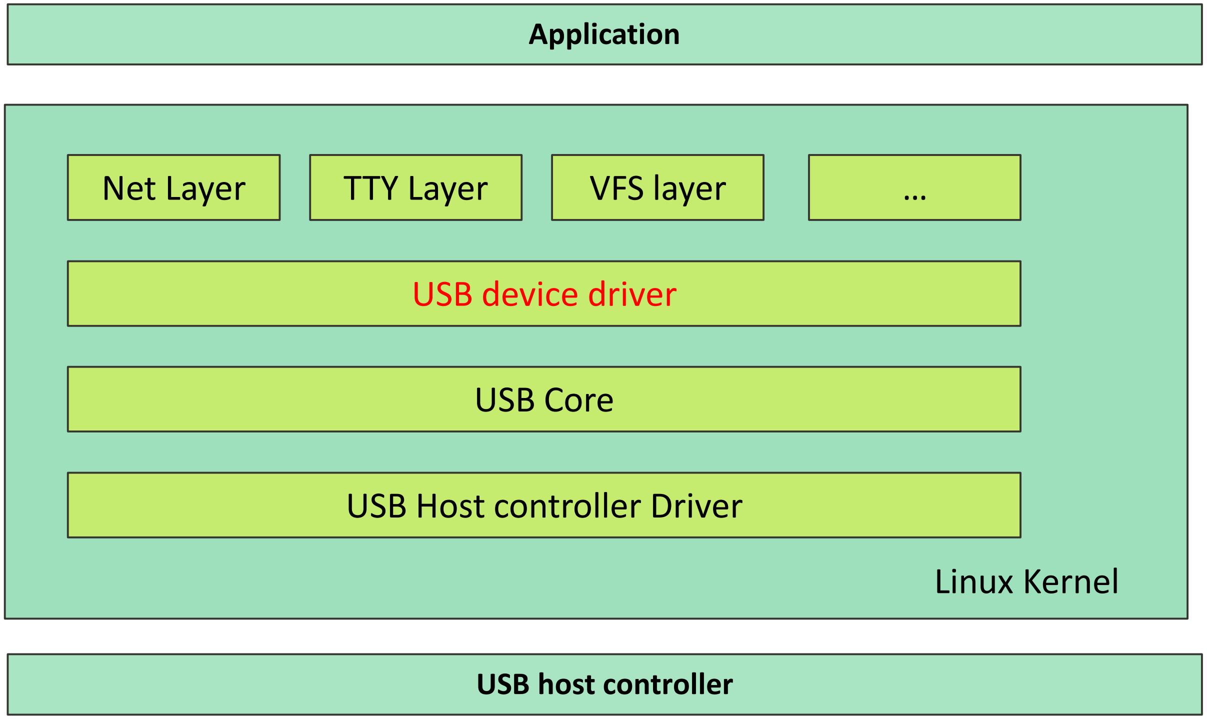
  Alps/kernel-3.18/arch/arm64/configs/XXXX_defconfig

- Other Items please refer to documents on MediaTek On-Line> Quick Start> **Fuel Gauge**

CONFIDENTIAL B

USB OTG

USB Architecture



Application

Net Layer

TTY Layer

VFS layer

...

USB device driver

USB Core

USB Host controller Driver

Linux Kernel

USB host controller

MTK USB Related Macro

- CONFIG_TCPC_CLASS=y
  - Use typec port controller， need DELETE if use Micro B;
  - Code path: drivers/misc/mediatek/typec/tcpc/

- CONFIG_USB_MTK_HDRC=y
  - Has a high speed usb controller based on MTK MUSB IP
  - Code path:drivers/misc/mediatek/usb20/

- CONFIG_MTK_MUSB_QMU_SUPPORT=y
  - Has a QMU capability for USB controller

- CONFIG_USB_MTK_OTG=y
  - Enable USB HOST OTG detection machanism

Code Introduction

- If the device cannot enumerate normally ,please check whether the do_connection_work executed successfully;

- If the OTG host cannot work normally ,please check whether the do_host_work executed successfully;

MTK OTG in MT6762

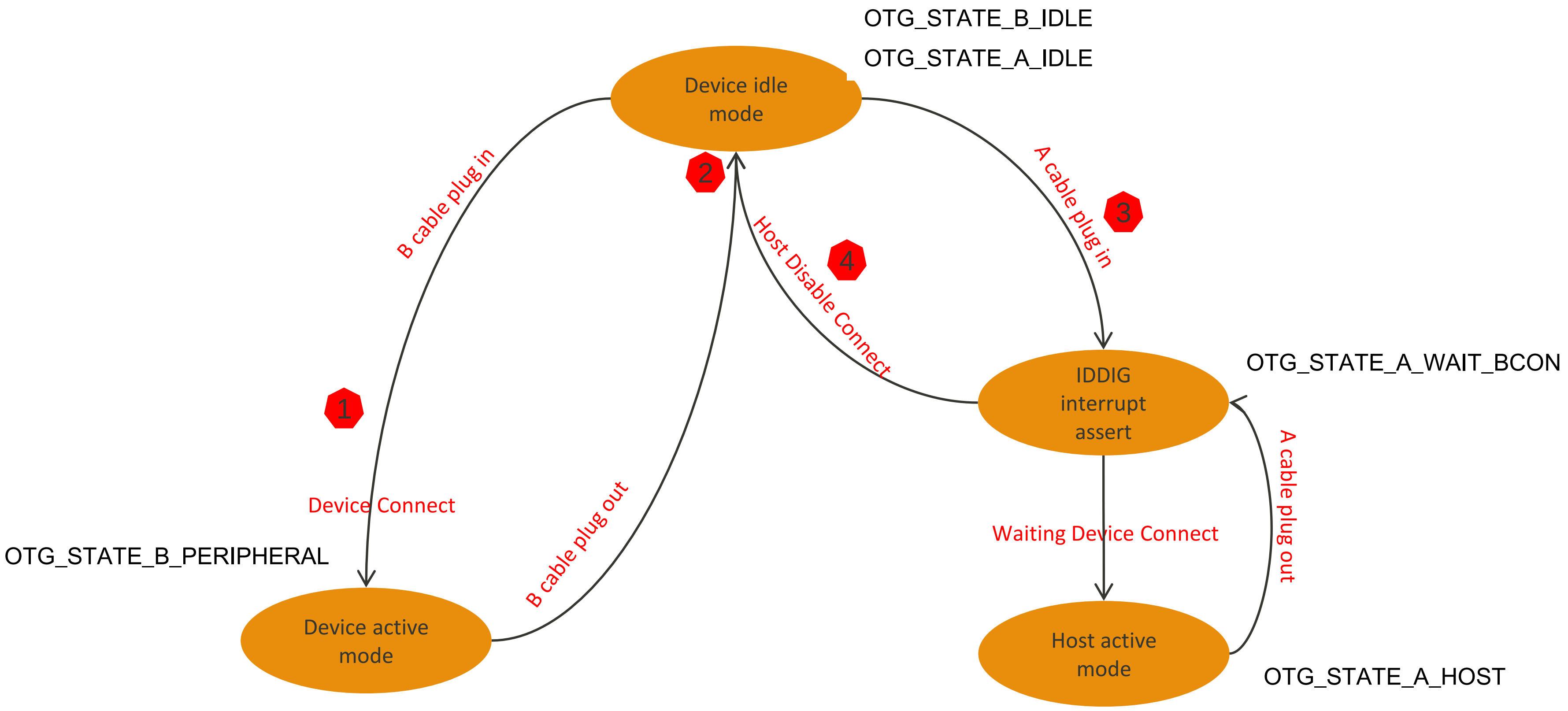
OTG Introduction

- An OTG product is a portable device that uses a single Micro-AB receptacle to operate at times as a USB Targeted Host and at times as a USB peripheral.
- OTG devices must always operate as a standard peripheral when connected to a standard USB host.

- Plug in A-cable, phone can be used as host.
- Plug in B-cable, phone can be used as device.

For more information， please refer to: https://www.usb.org/

State Diagram



1 Will enable soft connect and wait USB host to enumerate-- musb_start()

2 Will disable all EPs, flush all EPs FIFO and clear the soft connect – musb_stop()

3 Will set the session and vbus, enable all interrupt, wait for usb device connect – musb_start()

4 Will clear the session and vbus, disable all EPs, flush all EPs FIFO – musb_stop()

Implemented Functions

- Detect A-cable and B-cable plug in/out .
- Detect other devices connected, such as keyboard, u disk and so on.
- Fully support keyboard, mouse, U-disk and removable disk.
- By default Android JB support PTP device such as Camera.
- If user installs the corresponding application, Android JB will support any standard USB device.

- U-Disk and removable must be FAT file system. And this will recognize just the first partition before Android M and multiple partition on Android M.

Limititions

- PM has not implemented. That means if you plug in A-cable, phone will not suspend until A-cable is plugged out. And on the other hand, after you plug in a USB device, phone will not send suspend/resume signal on USB bus anyway.
- HNP(Host Negotiation Protocol) is not implemented in our product driver. So phone will only work as host after A-cable is plugged in. Beware this makes our product NOT fully compatible with USB OTG specification, but it will not impact daily use.

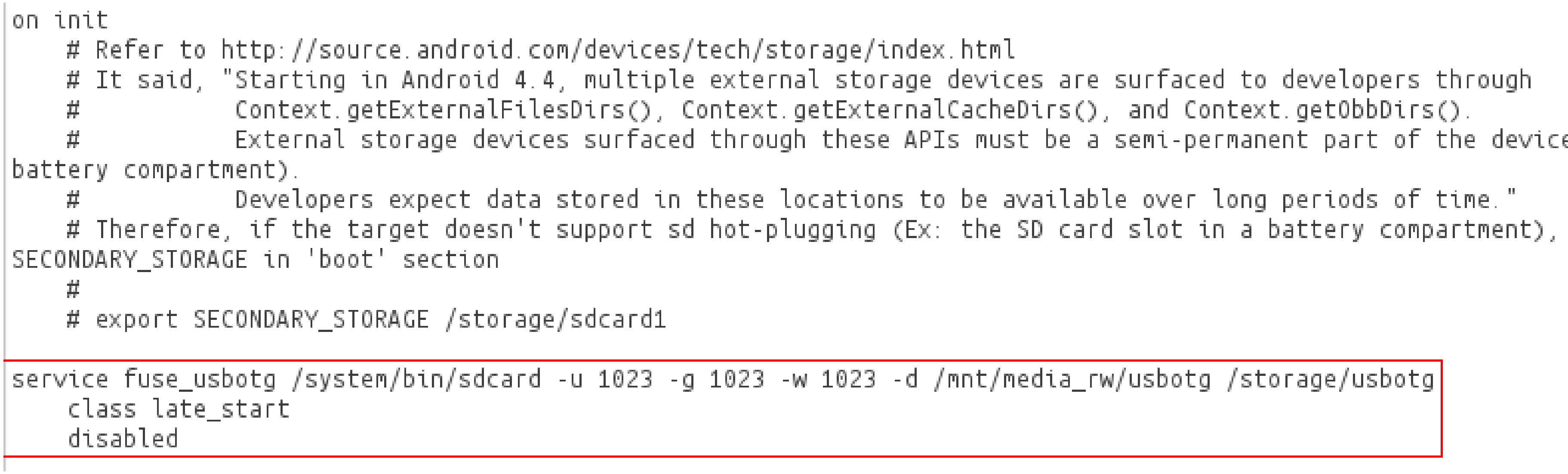
Enable OTG With IDDIG(1/5)

- To enable OTG with IDDIG, the only thing you have to do is set IDDIG pin mode in DWS;

- The following action default enabled ;

Enable OTG With IDDIG(2/5)

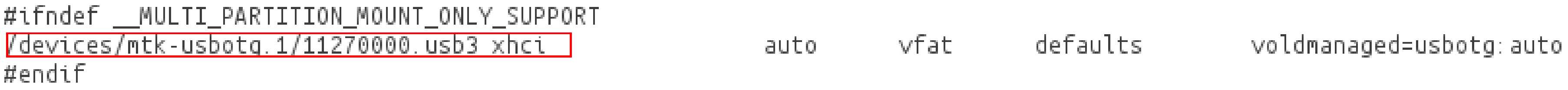
```
on init
    mkdir /mnt/media_rw/usbotg 0700 media_rw media_rw
    mkdir /storage/usbotg 0700 root root
```



```
on init
    # Refer to http://source.android.com/devices/tech/storage/index.html
    # It said, "Starting in Android 4.4, multiple external storage devices are surfaced to developers through
    #           Context.getExternalFilesDirs(), Context.getExternalCacheDirs(), and Context.getObbDirs().
    #           External storage devices surfaced through these APIs must be a semi-permanent part of the devic
battery compartment).
    #           Developers expect data stored in these locations to be available over long periods of time."
    # Therefore, if the target doesn't support sd hot-plugging (Ex: the SD card slot in a battery compartment),
SECONDARY_STORAGE in 'boot' section
    #
    # export SECONDARY_STORAGE /storage/sdcard1

service fuse_usbotg /system/bin/sdcard -u 1023 -g 1023 -w 1023 -d /mnt/media_rw/usbotg /storage/usbotg
    class late_start
    disabled
```

Enable OTG With IDDIG(3/5)

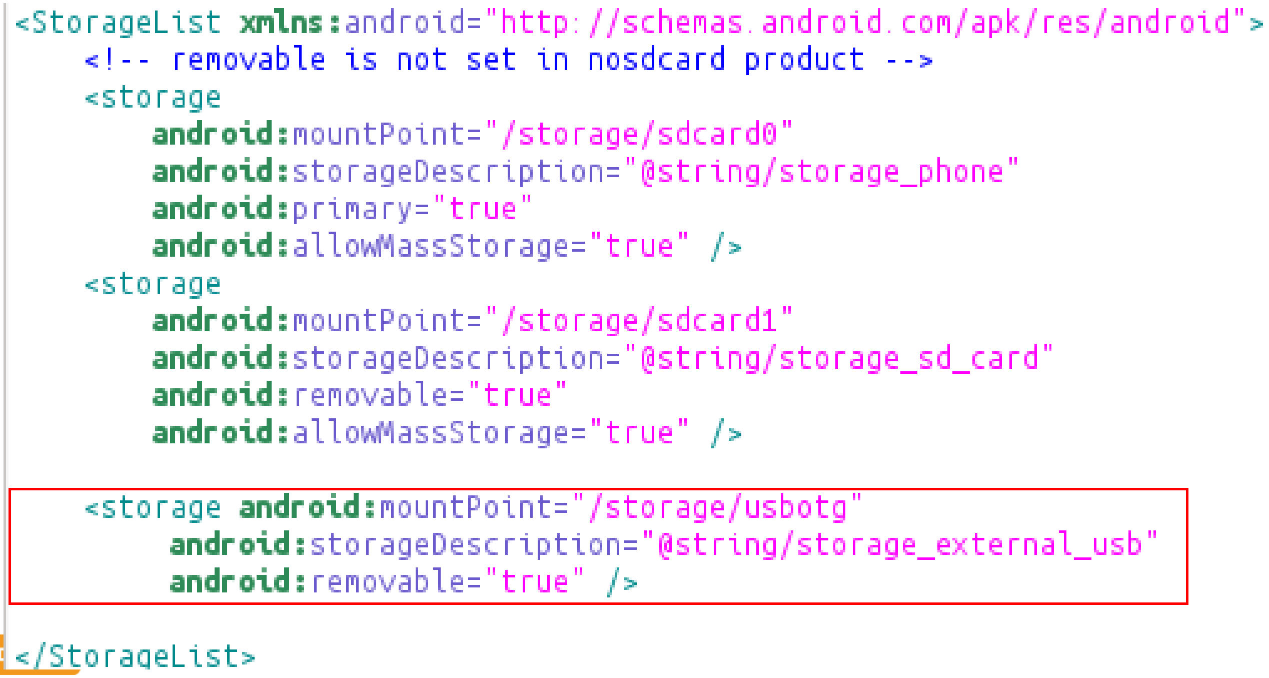
- Vold rule item
  - vendor\mediatek\proprietary\platform\mt6755\external\fstab\fstab.in



```
#ifndef __MULTI_PARTITION_MOUNT_ONLY_SUPPORT
/devices/mtk-usbotg.1/11270000.usb3_xhci                 auto      vfat      defaults        voldmanaged=usbotg:auto
#endif
```

Enable OTG With IDDIG(4/5)

- Storage_list.xml
  - device/mediatek/[project]/overlay/frameworks/base/core/res/res/xml/storage_list.xml



```
<StorageList xmlns:android="http://schemas.android.com/apk/res/android">
    <!-- removable is not set in nosdcard product -->
    <storage
        android:mountPoint="/storage/sdcard0"
        android:storageDescription="@string/storage_phone"
        android:primary="true"
        android:allowMassStorage="true" />
    <storage
        android:mountPoint="/storage/sdcard1"
        android:storageDescription="@string/storage_sd_card"
        android:removable="true"
        android:allowMassStorage="true" />

    <storage android:mountPoint="/storage/usbotg"
         android:storageDescription="@string/storage_external_usb"
         android:removable="true" />

</StorageList>
```

Enable OTG With IDDIG(5/5)

- GPIO dws setting
  - If you want to enable OTG feature, please make sure your hardware supports OTG (ex. VBUS supply and ID pin).
  - if iddig or drv_vbus gpio have changed, please update lk, preloader, kernel by dct tool at the same time.

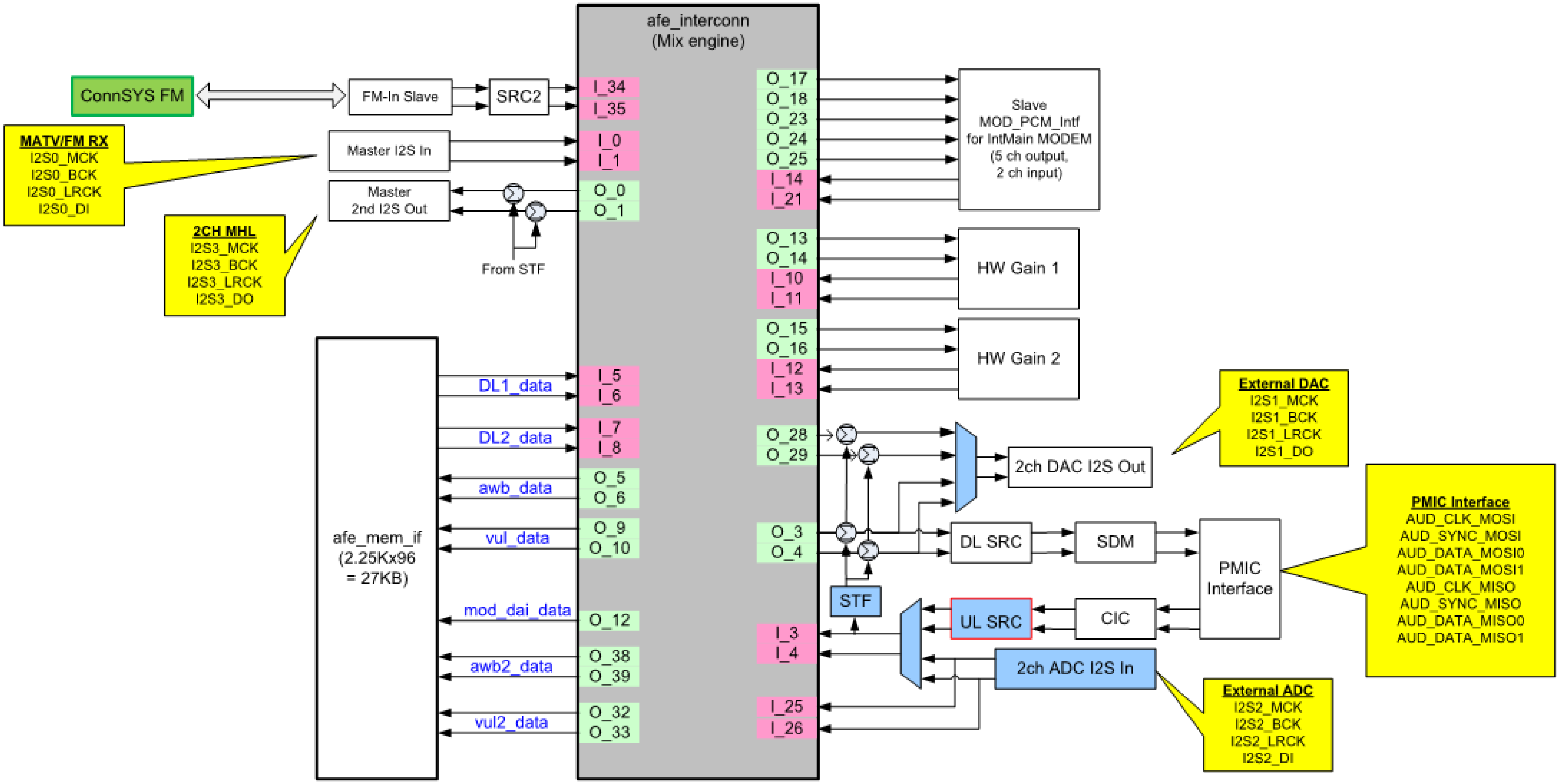
CONFIDENTIAL B

Audio

Outline

- Abbreviation
- Audio AFE Hardware
- Amp Setting
- MIC Setting
- SmartPA Porting

MT6762 AFE InterConnection

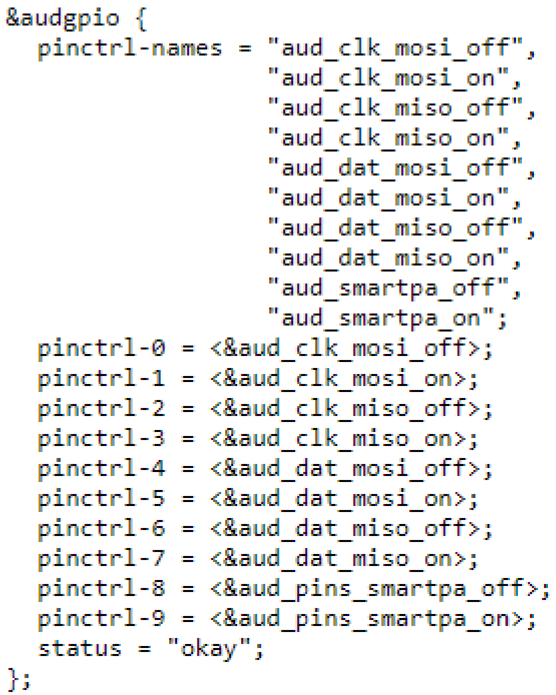


AMP setting

- MT6762 平台搭载的PMIC是MT6357,没有内置PA
- 使用外置PA 的话， 可以接Lineout 或者耳机左右声道（HPL/HPR）
  - 接Lineout
    - ProjectConfig.mk 中MTK_AUDIO_SPEAKER_PATH = int_lo_buf
  - 接(HPL/HPR)
    - ProjectConfig.mk 中MTK_AUDIO_SPEAKER_PATH = int_hp_buf
  - 在mtk-soc-codec-6357.c 中通过pinctrl/gpio 接口 enable/disable PA

AMP setting

- 在DTS 文件中配置使能外置PA 的GPIO， 可以加在audgpio下面
- 在/kernel-4.9/sound/soc/mediatek/common_int/mtk-auddrv-gpio.c 中初始pin ,之后就可以通过pinctrl 的接口enable/disable 外置PA 了
- 部分型号的PA 需要严格时序Delay 脉冲来触发， 如pinctrl_select_state 不能满足时序要求， 可以用gpio_set_value 代替， 配合spin_lock_irqsave()
  spin_unlock_irqrestore() Disable 中断



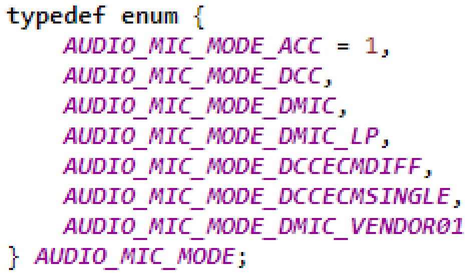
```
&audgpio {
    pinctrl-names = "aud_clk_mosi_off",
                    "aud_clk_mosi_on",
                    "aud_clk_miso_off",
                    "aud_clk_miso_on",
                    "aud_dat_mosi_off",
                    "aud_dat_mosi_on",
                    "aud_dat_miso_off",
                    "aud_dat_miso_on",
                    "aud_smartpa_off",
                    "aud_smartpa_on";
    pinctrl-0 = <&aud_clk_mosi_off>;
    pinctrl-1 = <&aud_clk_mosi_on>;
    pinctrl-2 = <&aud_clk_miso_off>;
    pinctrl-3 = <&aud_clk_miso_on>;
    pinctrl-4 = <&aud_dat_mosi_off>;
    pinctrl-5 = <&aud_dat_mosi_on>;
    pinctrl-6 = <&aud_dat_miso_off>;
    pinctrl-7 = <&aud_dat_miso_on>;
    pinctrl-8 = <&aud_pins_smartpa_off>;
    pinctrl-9 = <&aud_pins_smartpa_on>;
    status = "okay";
};
```

MIC setting(单双MIC)

- 单MIC 项目
  - MTK_AUDIO_NUMBER_OF_MIC = 1
- 双MIC 项目
  - MTK_AUDIO_NUMBER_OF_MIC = 2
- MTK_DUAL_MIC_SUPPORT 这个宏仅在ProjectConfig.mk 中没有定义MTK_AUDIO_NUMBER_OF_MIC的时候才使用
- getNumMicSupport() 用来获取当前project MIC 数量

MIC setting(MIC Mode)

- Digital MIC support
  - MTK_DIGITAL_MIC_SUPPORT = yes
- Mic Mode配置
  - 与HW 工程师确认Project 的Mic mode
  - audio_custom_exp.h 中
  - 设置#define PHONE_MIC_MODE (X)



```
typedef enum {
    AUDIO_MIC_MODE_ACC = 1,
    AUDIO_MIC_MODE_DCC,
    AUDIO_MIC_MODE_DMIC,
    AUDIO_MIC_MODE_DMIC_LP,
    AUDIO_MIC_MODE_DCCECMDIFF,
    AUDIO_MIC_MODE_DCCECMSINGLE,
    AUDIO_MIC_MODE_DMIC_VENDOR01
} AUDIO_MIC_MODE;
```

SmartPA Porting

- 参考SmartPA Framework Porting Guide.pdf
- MT6762 一共四组I2S , I2S0 和 I2S2 输入，I2S1和I2S3 作为输出。SmartPA建议选择I2S0+I2S3,外接I2S MIC 建议选择I2S2
- 其中I2S0 可以做Master/Slave Mode ,其他只能做Master Mode。