everyday genius

eMMC Customization Bring-up SOP

Version: 1.0

Release date: 2019-03-21

Specifications are subject to change without notice.

Document Revision History

Revision	Date	Description
1.0	2019-03-15	Initial Draft

Table of Contents

1. Introduction

The purpose of this document is eMMC boot programming guild, for example eMMC IO driving strength,pull-down/pull-up.

1.1 Purpose

The purpose of this document is eMMC boot programming guild. It will contain four parts, eMMC GPIO setting , eMMC IO strength setting

1.2 Definitions, Acronyms and Abbreviations

N/A

1.3 References

N/A

1.4 Overview

Section 1 is mt8183 supported feature
Section 2 introduce eMMC framework.
Section 3 is eMMC GPIO seting.
Section 4 is how to handle eMMC error during bring-up.

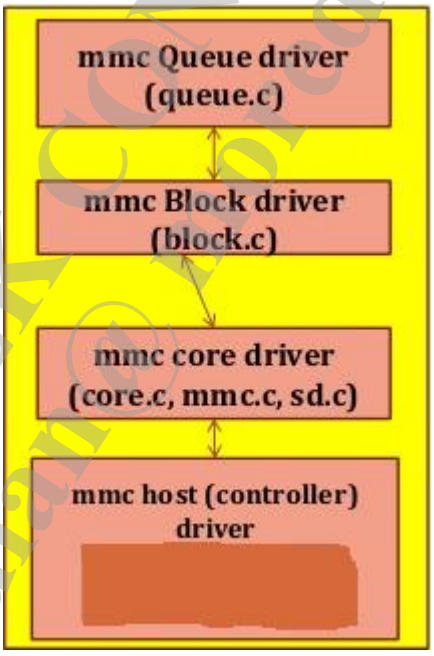
2. eMMC bring-up Contents

2.1 mt8183 support eMMC Feature

Support 1bit/4bit/8bit Bus mode
Support HS200 mode, data rate up to 200*8Mbps
Support HS400 mode, data rate up to 200*8*2MBps
Support eMMC5.1 CQ
Support eMMC Boot up mode

2.2 eMMC Linux Framework

Linux eMMC driver locate in kernel-4.4/drivers/mmc, It contains three directory.
Card/ queue.c block.c
Core/ bus.c core.c mmc.c mmc_ops.c host.c
Host/ mediatek/ComboA/



- mmc queue receives block read/write/erase requests from the generic core block layer.
- mmc queue driver picks up one request from its queue and assign it to mmc block driver.
- mmc block driver analyze the type of request and forwards the request to mmc core driver.
- mmc core driver has the protocol implementation for eMMC device detection, enumeration and data transfers to communicated with the actual hardware device.
- mmc core driver receives the request from block driver, prepares a mmc_request and forwards it to the mmc host driver.
- mmc host driver initiates the transfer to device by programming Hardware controller register.
- Once the request get processed by the hardware controller, an interrupt gets generated.

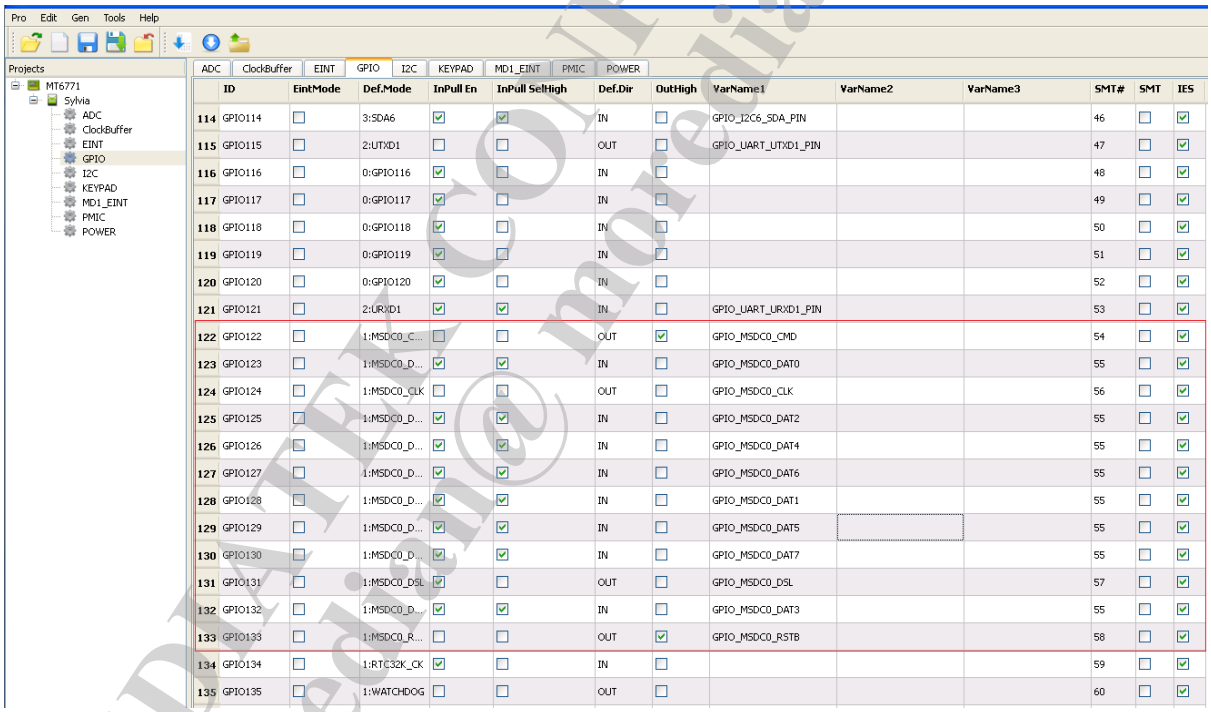
- mmc host driver receives request complete interrupt, analyzes it and pass the response to block driver.
- This process continues for all block requests.

2.3 eMMC GPIO setting

2.3.1 DCT tool and .dws file.

We use DCT tool to set gpio property in .dws file. These properties are GPIO mode, pull-pu/pull-down, direction and so on.
The DCT tool located in alps\vendor\mediatek\proprietary\scripts\dct\DrvGen.exe

There are three .dws files and eMMC gpio setting should be same.
alps\kernel-4.4\drivers\misc\mediatek\dws\mt6771\{Project}.dws
alps\vendor\mediatek\proprietary\bootable\bootloader\preloader\custom\${Project_name}\dct\dct\codegen.dws
alps\vendor\mediatek\proprietary\bootable\bootloader\lk\target\${Project_name}\dct\dct\codegen.dws



	ID	EintMode	Def.Mode	InPull En	InPull SelHigh	Def.Dir	OutHigh	VarName1	VarName2	VarName3	SMT#	SMT	IES
114	GPIO114	☐	3:SDA6	☑	☑	IN	☐	GPIO_I2C6_SDA_PIN			46	☐	☑
115	GPIO115	☐	2:UTXD1	☐	☐	OUT	☐	GPIO_UART_UTXD1_PIN			47	☐	☑
116	GPIO116	☐	0:GPIO116	☑	☐	IN	☐				48	☐	☑
117	GPIO117	☐	0:GPIO117	☑	☐	IN	☐				49	☐	☑
118	GPIO118	☐	0:GPIO118	☑	☐	IN	☐				50	☐	☑
119	GPIO119	☐	0:GPIO119	☑	☐	IN	☐				51	☐	☑
120	GPIO120	☐	0:GPIO120	☑	☐	IN	☐				52	☐	☑
121	GPIO121	☐	2:URXD1	☑	☑	IN	☐	GPIO_UART_URXD1_PIN			53	☐	☑
122	GPIO122	☐	1:MSDC0_C...	☐	☐	OUT	☑	GPIO_MSDC0_CMD			54	☐	☑
123	GPIO123	☐	1:MSDC0_D...	☑	☑	IN	☐	GPIO_MSDC0_DAT0			55	☐	☑
124	GPIO124	☐	1:MSDC0_CLK	☐	☐	OUT	☐	GPIO_MSDC0_CLK			56	☐	☑
125	GPIO125	☐	1:MSDC0_D...	☑	☑	IN	☐	GPIO_MSDC0_DAT2			55	☐	☑
126	GPIO126	☐	1:MSDC0_D...	☑	☑	IN	☐	GPIO_MSDC0_DAT4			55	☐	☑
127	GPIO127	☐	1:MSDC0_D...	☑	☑	IN	☐	GPIO_MSDC0_DAT6			55	☐	☑
128	GPIO128	☐	1:MSDC0_D...	☑	☑	IN	☐	GPIO_MSDC0_DAT1			55	☐	☑
129	GPIO129	☐	1:MSDC0_D...	☑	☑	IN	☐	GPIO_MSDC0_DAT5			55	☐	☑
130	GPIO130	☐	1:MSDC0_D...	☑	☑	IN	☐	GPIO_MSDC0_DAT7			55	☐	☑
131	GPIO131	☐	1:MSDC0_DSL	☑	☐	OUT	☐	GPIO_MSDC0_DSL			57	☐	☑
132	GPIO132	☐	1:MSDC0_D...	☑	☑	IN	☐	GPIO_MSDC0_DAT3			55	☐	☑
133	GPIO133	☐	1:MSDC0_R...	☐	☐	OUT	☑	GPIO_MSDC0_RSTB			58	☐	☑
134	GPIO134	☐	1:RTC32K_CK	☑	☐	IN	☐				59	☐	☑
135	GPIO135	☐	1:WATCHDOG	☐	☐	OUT	☐				60	☐	☑

2.3.2 IO driving strength in .dtsi file.

We can not set GPIO IO driving strength in dws file. IO driving strength is set in cust_mt6771_msdc.dtsi.
It located kernel-4.4/arch/arm64/boot/dts/mediatek/cust_mt6771_msdc.dtsi.

```
68 &pio {
69         msdc0_pins_default: msdc0@default {
70                 pins_cmd {
71                         drive-strength = /bits/ 8 <4>;
72                 };
73                 pins_dat {
74                         drive-strength = /bits/ 8 <4>;
75                 };
76                 pins_clk {
77                         drive-strength = /bits/ 8 <4>;
78                 };
79                 pins_rst {
80                         drive-strength = /bits/ 8 <4>;
81                 };
82                 pins_ds {
83                         drive-strength = /bits/ 8 <4>;
84                 };
85         };
```

2.4 eMMC crc error.

If you meet the issue that board cannot boot-up, due to many eMMC error from uart log. You can disable eMMC HS200 and HS400 mode, only run high-speed mode with lower frequency.
The setting is also in cust_mt6771_msdc.dtsi

```
17 &msdc0 {
18         index = /bits/ 8 <0>;
19         clk_src = /bits/ 8 <MSDC50_CLKSRC_400MHZ>;
20         bus-width = <8>;
21         max-frequency = <200000000>;
22         cap-mmc-highspeed;
23         msdc-sys-suspend;
24         /*
25         mmc-ddr-1_8v;
26         mmc-hs200-1_8v;
27         mmc-hs400-1_8v;
28         */
29         non-removable;
30         pinctl = <&msdc0_pins_default>;
31         register_setting = <&msdc0_register_setting_default>;
32         host_function = /bits/ 8 <MSDC_EMMC>;
33         bootable;
34         status = "okay";
35         vmmc-supply = <&mt_pmic_vemc_ldo_reg>;
36         clocks = <&infracfg_ao INFRACFG_AO_MSDC0_SCK_CG>, <&infracfg_ao INFRACFG_AO_MSDC0_CG>;
37         clock-names = "msdc0-clock", "msdc0-hclock";
38         hw_dvfs = /bits/ 8 <0>;
39 };
```

Then feedback the issue to mediatek. We will analysis it and check root cause.