SigmaStar Camera
PWM 使用参考

REVISION HISTORY

Revision No.	Description	Date
{0.1}	• {Initial release}	{07/28/2018}
{0.2}	• {release in formal format }	{12/17/2019}

TABLE OF CONTENTS

1. PWM 參數

- Duty_cycle :

  占空比。

  - 例：Echo 25 > duty_cycle 表示占空比是 25%。

- Period :

  频率，Frequency。

  - 例：Echo 2000 > period
  - 表示 2K HZ 的 frequency 的 pwm 波。

- Polarity :

  极性。

  - 如果是 normal。那么 duty_cycle=25%，表示高电平占的比例是 25%。
  - 如果是 inverse，那么就反之。

- Enable/disable :

  使能。

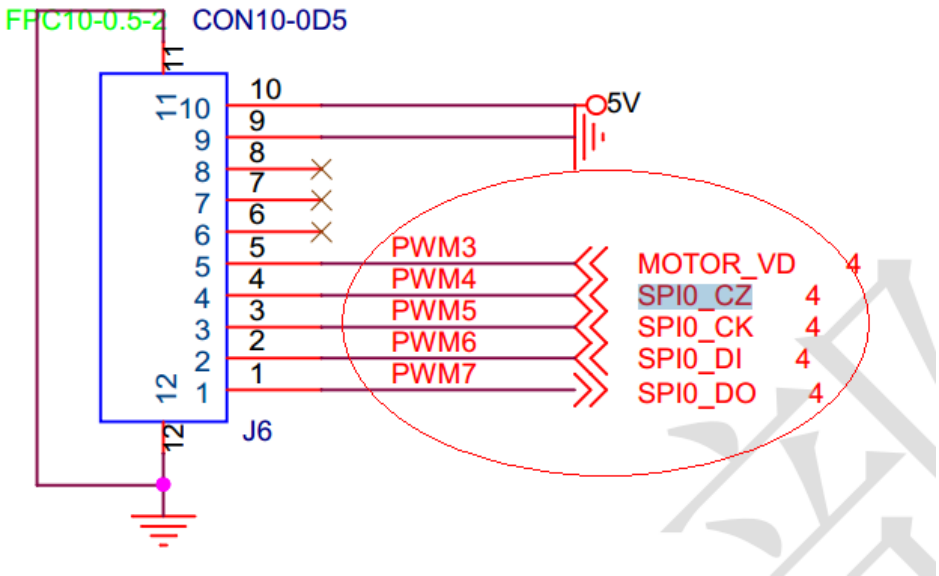
2. KERNEL 配置

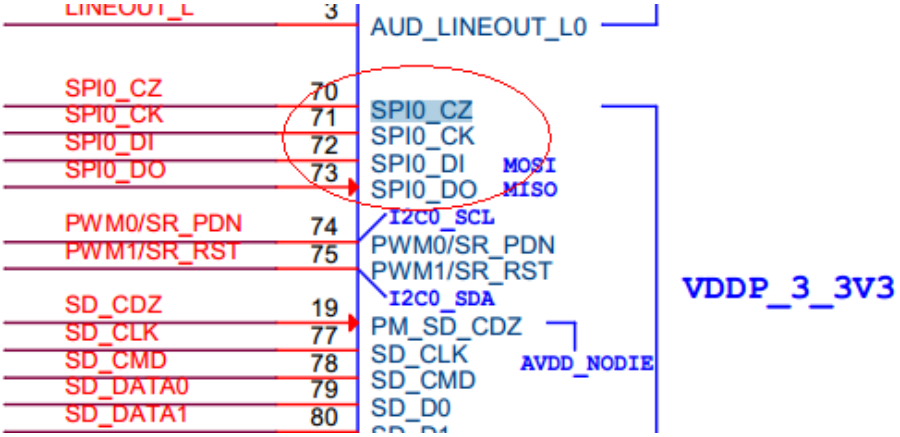
2.1. DTS 的配置

```
pwm {
    compatible = "sstar,infinity-pwm";
    reg = <0x1F003400 0x600>;
    clocks = <&CLK_xtali_12m>;
    npwm = <11>;
    pad-ctrl = <PAD_PWM0 PAD_PWM1 PAD_UNKNOWN PAD_UNKNOWN PAD_UNKNOWN PAD_UNKNOWN PAD_UNKNOWN PAD_UNKNOWN PAD_UN
    status = "ok";
};
```

npwm: 11; 表示 pwm 有 11 组;
Pad-ctrl: 由于是共享 gpio 的。所以当 gpio 被当作 pwm 使用的时候，需要做切 pad 动作。
上述： PAD_PWM0 / PAD_PWM1 表示只使能 PWM0/PWM1。

2.2. 客户硬件举例



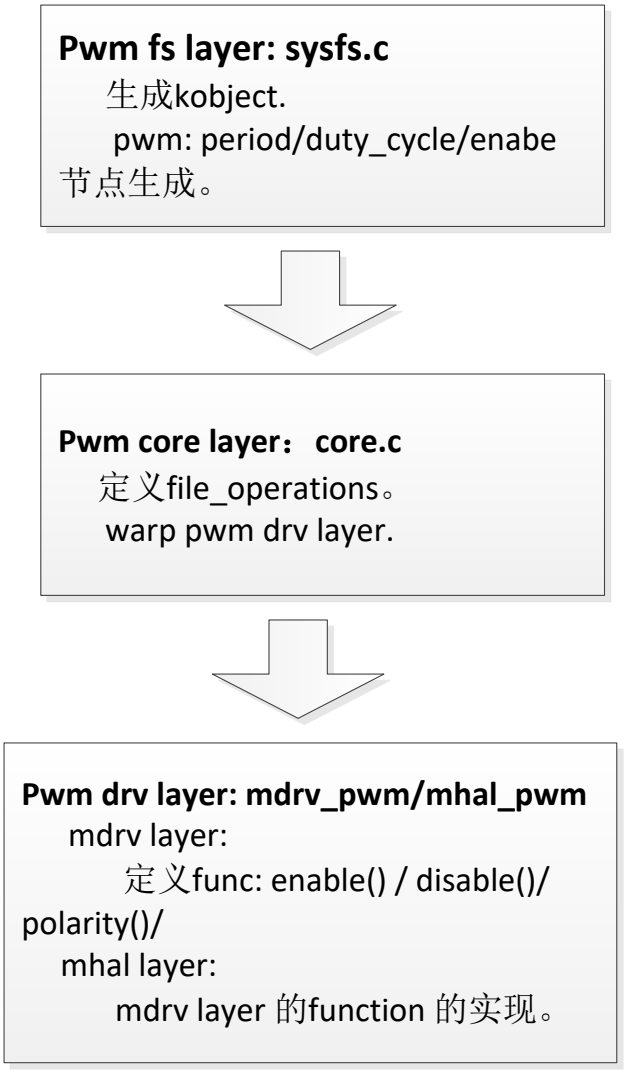


通过查看： drivers\sstar\include\ \gpio.h

```
91 #define PAD_PM_SPI_CZ                70
92 #define PAD_PM_SPI_CK                71
93 #define PAD_PM_SPI_DI                72
94 #define PAD_PM_SPI_DO                73
95 #define PAD_PM_SPI_WPZ               74
96 #define PAD_PM_SPI_HLD               75
97 #define PAD_PM_LED0                  76
drivers/sstar/include/ /gpio.h
```

所以，只需要再配置好 dts 就可以了。

3. PWM 架构



4. USER MODE 对 PWM 之控制：

4.1. 在 User mode Console 下控制 PWM

1. Export PWM number (例如 USB PAD_PWM0)

   Command:

   cd /sys/class/pwm/pwmchip0

   echo 0 > export

2. Set period(frequency) / duty_cycle / polarity / enable

   Command:

   cd pwm0

   echo xxxx > period

   In our driver implementation, xxxx indicates output frequency

   ex: echo 2000 > period will generate 2KHz waveform

   echo xx > duty_cycle

   ex: echo 25 > duty_cycle will generate 25% duty_cycle

   echo inversed > polarity

   Inverse output waveform, default is normal

   echo 1 > enable

   Enable output waveform

   对应 user 层代码：

   即：

   Open 一个节点；

   Write 节点；

4.2. 在 User mode Console 下控制 Motor

1. Motor hierarchy
   Group 0
      PWM 0
      PWM 1
      PWM 2
      PWM 3
   Group 1
      PWM 4
      PWM 5
      PWM 6
      PWM 7
   Group 2
      PWM 8
      PWM 9
      PWM 10

2. Cd 馬達控制路徑

   Command:
   cd /sys/devices/virtual/mstar/motor
3. Set mode/period(frequency) / Begin/End / round number/enable/hold/stop

   - mode

   Command:
   echo PWM_ID enable > group_mode

   ex : echo 0 1 > group_mode # 設定 PWM0 為馬達模式

   ex : echo 0 0 > group_mode # 取消 PWM0 為馬達模式

   - period

   Command:
   echo PWM_ID period > group_period
   In our driver implementation, xxxx indicates output frequency
   ex: echo 0 2000 > group_period # PWM0 will generate 2KHz waveform

   - begin

   Command:
   echo PWM_ID begin > group_begin

ex: echo 0 100 > group_begin # PWM0 will generate duty_cycle starting from 100/1000 of the period

- end

Command:

echo PWM_ID end > group_end

ex: echo 0 250 > group_end # PWM0 will generate duty_cycle ending at 250/1000 of the period

- Round mode

Command:

echo GROUP_ID round > group_round

ex: echo 0 10000 > group_round # Group 0 will generate 10000 period of waveform.

If need to **continue** set a new round after last round completed, set new arguments before the end of the round, than it will continue create a new round.

ex: echo 0 10000 > group_round
(During 10000 rounds)
echo 0 2000 > group_period
echo 0 100 > group_begin
echo 0 250 > group_end
echo 0 20000 > group_round #

Group 0 will generate 10000 period of waveform first, and continue generate 20000 period of waveform after that.

- enable

Command:

echo GROUP_ID enable > group_enable

ex: echo 0 1 > group_enable # Group 0 start generating the waveform
ex: echo 0 0 > group_enable # Group 0 stop generating the waveform

- Hold mode

Command:

echo GROUP_ID > group_hold

ex: echo 0 > group_hold # Group 0 hold the last complete waveform

After enable, if set new arguments **before set group_hold**, it will generate new waveform rather than hold the last complete waveform.

Ex:

echo 0 1 > group_enable
echo 0 2000 > group_period
echo 0 100 > group_begin
echo 0 250 > group_end
echo 0 > group_hold

- stop

Command:

echo GROUP_ID > group_stop

ex: echo 0 > group_stop # Group 0 immediately stop the waveform