SUNXI PWM

模块使用说明

1.0

2018.12.18

文档履历

版本号	日期	制/修订人	内容描述
1.0	2018.12.18		version 1.0

目录

1. 概述

1.1 编写目的

介绍 pwm 使用方法。

1.2 适用范围

软件：aw linux-3.10/4.4/4.9 内核。

1.3 相关人员

pwm 驱动、及应用层的开发/维护人员。

1.4 术语与缩略语

术语/缩略语	解释说明
Sunxi	指Allwinner的一系列SOC硬件平台。
PWM	Pulse Width Modulation
PWM对	电机等硬件需要两路脉冲信号来控制其正常运转，一般两路极性相反，频率、占空比参数相同的PWM构成一个PWM对
PWM死区控制时间	大功率电机、变频器等由大功率管、IGBT等元件组成H桥或3相桥，每个桥的上半桥和下半桥是觉对不能导通的，在pwm信号驱动这些元件时，往往会由于没有延迟而造成未关断某路半桥这样会造成功率元件的损坏，在PWM中加入死区时间的控制即是让上半桥关断后，自动插入一个时间，延迟后再打开下半桥。

2. 相关配置

2.1 menuconfig 配置说明

在命令行中进入内核根目录，执行 make ARCH=arm menuconfig 或者 make ARCH=arm64 menuconfig(64 位) 进入配置主界面。并按以下步骤操作：

Device Drivers->Pulse-Width Modulation(PWM) Support->SUNXI PWM SELECT->后面
如果是1.0版本选sunxi pwm support
如果是增强版选sunxi enhance pwm support

```
[*] USB support  --->
<*> MMC/SD/SDIO card support  --->
< > Sony MemoryStick card support  --->
[ ] LED Support  --->
<*> Switch class support  --->
[ ] Accessibility support  --->
[ ] EDAC (Error Detection And Correction) reporting  --->
[*] Real Time Clock  --->
[*] DMA Engine support  --->
[ ] Auxiliary Display support  --->
< > Userspace I/O drivers  --->
[ ] Virtualization drivers  --->
    Virtio drivers  --->
    Microsoft Hyper-V guest support  --->
[*] Staging drivers  --->
    Common Clock Framework  --->
    Hardware Spinlock drivers  --->
    TIMER: Select the soc timer version. (SUNXI TIMER)  --->
[ ] Support for ARM architected timer event stream generation
[ ] Mailbox Hardware Support  --->
[ ] IOMMU Hardware Support  --->
    Remoteproc drivers  --->
    Rpmsg drivers  --->
[*] Generic Dynamic Voltage and Frequency Scaling (DVFS) support  --->
< > External Connector Class (extcon) support  --->
[ ] Memory Controller drivers  --->
< > Industrial I/O support  --->
[*] Pulse-Width Modulation (PWM) Support  --->
< > IndustryPack bus support  --->
[ ] Reset Controller Support  --->
    SOC (System On Chip) specific Drivers  --->
```

图 1: pwm 配置图

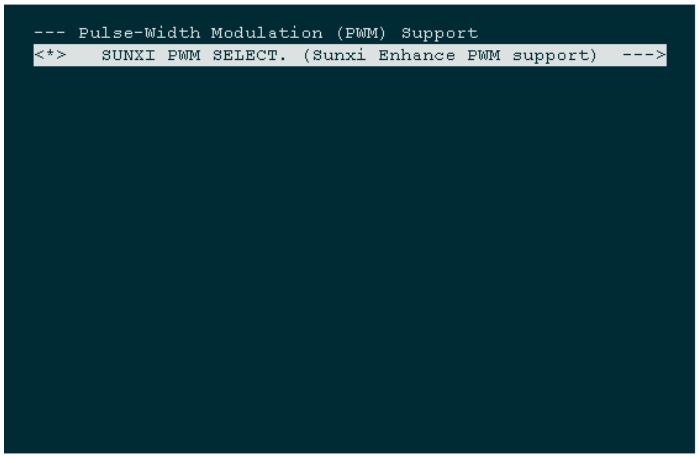

图 2: pwm 配置图

2.2 dts 配置

```
pwm: pwm@0300a000 {
        compatible = "allwinner,sunxi-pwm";
        reg = <0x0 0x0300a000 0x0 0x310>;
        clocks = <&clk_pwm>;
        pwm-number = <3>;
        pwm-base = <0x0>;
        pwms = <&pwm0>, <&pwm1>, <&pwm3>;
    };
    s_pwm: s_pwm@0x07020c00 {
        compatible = "allwinner,sunxi-s_pwm";
        reg = <0x0 0x07020c00 0x0 0x3c>;
        pwm-number = <1>;
        pwm-base = <0x10>;
        pwms = <&spwm0>, <&spwm1>, <&spwm2>;
```

```
        };
pwm-number: 当前chip上一共要使用多少个pwm的数量
pwm-base : 片选的基地址,spwm从16开始
pwms: 当前配置的pwm个体
```

3. 源码位置

3.1 头文件位置

include/linux/pwm.h

3.2 源码位置

drivers/pwm/sunxi-pwm-new.c

4. 使用说明

使用时:
- pwm_request，申请pwm句柄
- pwm_config， 配置pwm period & duty，注意单位是ns。
- pwm_set_polarity， 设置pwm的极性。
- pwm_enable，使能pwm。

不使用时:
- pwm_disable，关闭pwm
- pwm_release，释放pwm句柄

4.9内核,由于pwm core层加了一些限制,必须要设置config之后设置极性才有用,所以调用顺序要搞好.

5. 接口说明

5.1 pwm_request

原型：

```
struct pwm_device *pwm_request(int pwm_id, const char *label);
```

功能：申请 pwm。
pwm_id：pwm 的索引号，从 0 开始
Lable: 标签名，建议传入设备名，或者与用途相关的名字，方便管理。
成功返回 pwm 句柄，如果失败，则返回 NULL。

5.2 pwm_release

原型：

```
void pwm_free(struct pwm_device *pwm);
```

功能：释放 pwm。
pwm：pwm 句柄
无返回值。

5.3 pwm_config

原型：

```
int pwm_config(struct pwm_device *pwm, int duty_ns, int period_ns);
```

功能：配置 pwm 的周期以及占空比。
pwm: pwm 句柄。
duty_ns: 有效区域时间, duty_ns / period_ns = 占空比。
period_ns: pwm 的周期时间，单位为 ns。
成功则返回 0，失败则返回错误码。

5.4 pwm_set_polarity

原型：

```
int pwm_set_polarity(struct pwm_device *pwm, enum pwm_polarity polarity);
```

功能：配置 pwm 的极性, 高电平有效还是低电平有效。
pwm: pwm 句柄。
polarity: pwm 极性，PWM_POLARITY_NORMAL 为正常，高有效，PWM_POLARITY_INVERSED 为反转，即低有效。
period_ns: pwm 的周期时间，单位为 ns。
成功则返回 0，失败则返回错误码。

5.5 pwm_enable

原型：

```
void pwm_enable(struct pwm_device *pwm);
```

功能：使能 pwm。
pwm：pwm 句柄
成功则返回 0，失败则返回错误码。

5.6 pwm_disable

原型：

```
void pwm_disable(struct pwm_device *pwm);
```

功能：关闭 pwm。
pwm：pwm 句柄
成功则返回 0，失败则返回错误码。

6. demo

```
PWM_FREQ = 50000;
pwm_test_info.channel = 0;
pwm_test_info.polarity =1;
pwm_test_info.dev = disp_sys_pwm_request(pwm_test_info.channel);
period_ns = 1000*1000*1000 / PWM_FREQ;
backlight_bright = 250;
duty_ns = (backlight_bright * period_ns) / 256;
pwm_test_info.duty_ns = duty_ns;
pwm_test_info.period_ns = period_ns;
disp_sys_pwm_config(pwm_test_info.dev, duty_ns, period_ns);
disp_sys_pwm_set_polarity(pwm_test_info.dev, pwm_test_info.polarity);
ret = disp_sys_pwm_enable(pwm_test_info.dev);
```

7. Declaration

This document is the original work and copyrighted property of Allwinner Technology (“Allwinner”). Reproduction in whole or in part must obtain the written approval of Allwinner and give clear acknowledgement to the copyright owner. The information furnished by Allwinner is believed to be accurate and reliable. Allwinner reserves the right to make changes in circuit design and/or specifications at any time without notice. Allwinner does not assume any responsibility and liability for its use. Nor for any infringements of patents or other rights of the third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of Allwinner. This datasheet neither states nor implies warranty of any kind, including fitness for any particular application.tates nor implies warranty of any kind, including fitness for any particular application.