PWM（脉冲调制）可以用于控制电机、调节背光、控制呼吸灯等。Linux内核将pwm驱动分为pwm\_device(设备) 和pwm\_chip(芯片)两个部分。pwm\_chip指输出pwm的控制器，其可以输出单路或者多路pwm，而pwm\_device就是使用pwm\_chip的pwm的设备。

# pwm\_chip

## struct pwm\_chip

pwm\_chip结构描述了一个pwm控制器，也为使用此pwm\_chip的pwm\_device提供了接口函数。pwm\_chip结构定义如下：

struct pwm\_chip {

struct device \*dev;

struct list\_head list;

const struct pwm\_ops \*ops;

int base;

unsigned int npwm;

struct pwm\_device \*pwms;

struct pwm\_device \* (\*of\_xlate)(struct pwm\_chip \*pc,

const struct of\_phandle\_args \*args);

unsigned int of\_pwm\_n\_cells;

bool can\_sleep;

};

成员含义如下：

dev: pwm\_chip对应的设备，一般都是直接指定为platform\_device的device，chip.dev = &pdev->dev

ops: 控制器的操作函数集，必须提供！

base: pwmchip\_add注册时创建的/sys/class/pwm/pwmchipN中的N值，注意指定的话可能会因为冲突而导致注册失败，也可以设置为-1让内核自动分配。

npwm: 支持的pwm路数，等同于支持的pwm\_device个数，内核根据此数目分配相应数目的pwm\_device，必须提供！

pwms: 指向内核分配的个数为npwm的pwm\_device数组，内核自动分配，无需关心。

of\_pwm\_n\_cells：dts节点的cell，一般是2或者3，为3时，dts包含pwm number、pwm period和pwm flag，如pwms = <&pwm 0 5000000 PWM\_POLARITY\_INVERTED>;，为2时，则不包含flag信息。

of\_xlate：请求pwm\_device时执行的回调，用于根据设备树指定信息进行pwm\_device的请求与信息的解析。此函数内核已经提供，of\_pwm\_n\_cells为3时使用of\_pwm\_xlate\_with\_flags，of\_pwm\_n\_cells为2时使用of\_pwm\_simple\_xlate，如果未赋值，则内核默认赋值of\_pwm\_simple\_xlate。

## struct pwm\_ops

struct pwm\_ops为控制pwm\_device时执行的pwm\_chip函数合集，定义如下：

struct pwm\_ops {

int (\*request)(struct pwm\_chip \*chip, struct pwm\_device \*pwm);

void (\*free)(struct pwm\_chip \*chip, struct pwm\_device \*pwm);

int (\*config)(struct pwm\_chip \*chip, struct pwm\_device \*pwm,

int duty\_ns, int period\_ns);

int (\*set\_polarity)(struct pwm\_chip \*chip, struct pwm\_device \*pwm,

enum pwm\_polarity polarity);

int (\*capture)(struct pwm\_chip \*chip, struct pwm\_device \*pwm,

struct pwm\_capture \*result, unsigned long timeout);

int (\*enable)(struct pwm\_chip \*chip, struct pwm\_device \*pwm);

void (\*disable)(struct pwm\_chip \*chip, struct pwm\_device \*pwm);

int (\*apply)(struct pwm\_chip \*chip, struct pwm\_device \*pwm,

struct pwm\_state \*state);

void (\*get\_state)(struct pwm\_chip \*chip, struct pwm\_device \*pwm,

struct pwm\_state \*state);

struct module \*owner;

};

成员含义如下：

request/free: 已弃用

config: 与pwm\_device接口函数中的pwm\_config相对应，配置pwm\_device的频率、占

空比。必项提供。

enable/disable: 与pwm\_device接口函数中的pwm\_enable/pwm\_disable函数相对应，

用于使能/禁止pwm 信号输出。必项提供。

set\_polarity: 与pwm\_device接口函数中的pwm\_set\_polarity函数对应，设置 pwm 信号的极性。可选。

## 操作函数

按照上述结构描述将必须提供的成员完成后就可以向内核注册pwm\_chip了。

int pwmchip\_add(struct pwm\_chip \*chip)

pwm\_chip的注册，此时极性是默认的PWM\_POLARITY\_NORMAL。

int pwmchip\_add\_with\_polarity(struct pwm\_chip \*chip,enum pwm\_polarity polarity)

pwm\_chip的注册，此时极性可选PWM\_POLARITY\_NORMAL或PWM\_POLARITY\_INVERSED

int pwmchip\_remove(struct pwm\_chip \*chip)

pwm\_chip的注销。

# pwm\_device

## struct pwm\_device

pwm\_device描述了一个pwm设备，定义如下：

struct pwm\_device {

const char \*label;//pwm设备的名称

unsigned long flags;//标志

unsigned int hwpwm;//设备的索引号

unsigned int pwm;

struct pwm\_chip \*chip;

void \*chip\_data;

struct pwm\_args args;

struct pwm\_state state;

};

此结构可通过如下函数获取：

struct pwm\_device \*pwm\_get(struct device \*dev, const char \*con\_id)

从指定设备的设备树节点获取pwm\_device。

struct pwm\_device \*of\_pwm\_get(struct device\_node \*np, const char \*con\_id)

直接从设备树节点获取pwm\_device。

void pwm\_put(struct pwm\_device \*pwm)

释放 pwm\_device。

## 参数说明

pwm设备关注4个参数：

频率：输出pwm信号的频率。

占空比：一个周期内PWM信号高低的比率。

极性：PWM信号的极性，决定了是高占空比的信号输出电平高，还是低占空比信号输出电平高。假设一个信号的占空比为100%，如果为正常极性，则输出电平最大，如果为翻转的极性，则输出电平为0。

开关：控制pwm信号是否输出。

## 操作函数

int pwm\_enable(struct pwm\_device \*pwm)

void pwm\_disable(struct pwm\_device \* pwm)

控制pwm信号是否输出。

int pwm\_set\_polarity(struct pwm\_device \*pwm, enum pwm\_polarity polarity)

设置极性，可选参数有 PWM\_POLARITY\_NORMAL(普通)和PWM\_POLARITY\_INVERSED(反转)两种。

int pwm\_config(struct pwm\_device \*pwm, int duty\_ns, int period\_ns)

设置占空比和频率，参数duty\_ns为占空比，period\_ns为频率。

# pwm sysfs使用

cd /sys/class/pwm/pwmchipN/

为已probe的pwmchip创建的目录，N就是pwm\_chip结构中的base。

cat npwm

pwmchip支持的pwm通道数。

echo 0 > export

调出pwm通道供sysfs控制，输入值范围为0~npwm-1，可通过echo 0 > unexport关闭。

cd pwm0/

export之后就会创建pwm通道目录。

echo 1000000 > period

echo 500000 > duty\_cycle

将pwm设置为周期1ms，占空比0.5ms。

echo 1 > enable

使能pwm，可通过echo 0 > enable失能。

# 示例说明

## 设备树节点

pwm: pwm@0 {

compatible = "pwmchip-test";

#pwm-cells = <2>;//此节点必须定义

};

pwm-device {

compatible = "pwmdevice-test";

pwms = <&pwm 0 5000000>;

};