HCRTOS SDK pwm userguide

1. 目录

2. 文档履历

版本号	日期	制/修订人	制/修订记录
1.0	2023.04.10	邱浩佳	新增说明文档
2.0	2023.04.23	邱浩佳	支持在初始化的时候设置占空比和周期
3.0	2023.04.25	邱浩佳	取消在初始化时设置占空比和周期，以及在不同阶段避免多次初始化
4.0	2023.08.14	邱浩佳	新增极性设置
5.0	2023.09.09	邱浩佳	新增可配置pwm范围选择

3. 概述

3.1 编写目的

指导和介绍pwm的使用和开发

3.2 读者对象

软件开发工程师和技术支持工程师。

4. 模块介绍

- rtos的pwm模块区分15xx系列和16xx系列的芯片，15xx系列的芯片有3路pwm通道；16xx系列的芯片有6路pwm通道
- 当前驱动下，pwm的最大clk为27M，27M时钟下最大频率的周期为37ns。
- 支持在驱动初始化时设置占空比和周期。

4.1 设备树配置

以pwm0为例子：

```
pwm@0 {
        pinmux-active = <PINPAD_L23 1>;
        devpath = "/dev/pwm0";
        polarity = <1>;
        status = "okay";
};
```

pinmux-active：将要设置的引脚PINPAD_L23配置成pwm模式，并不是配置成pwm的值都写为1，具体的引脚配置成pwm请查看具体配置根据所使用引脚参考components/kernel/source/include/uapi/hcuapi/pinmux/hc15xx_pinmux.h和components/kernel/source/include/uapi/hcuapi/pinmux/hc16xx_pinmux.h。

devpath：生成的pwm节点的路径。

polarity：初始化完pwm后通道的极性。0：低电平；1：高电平；

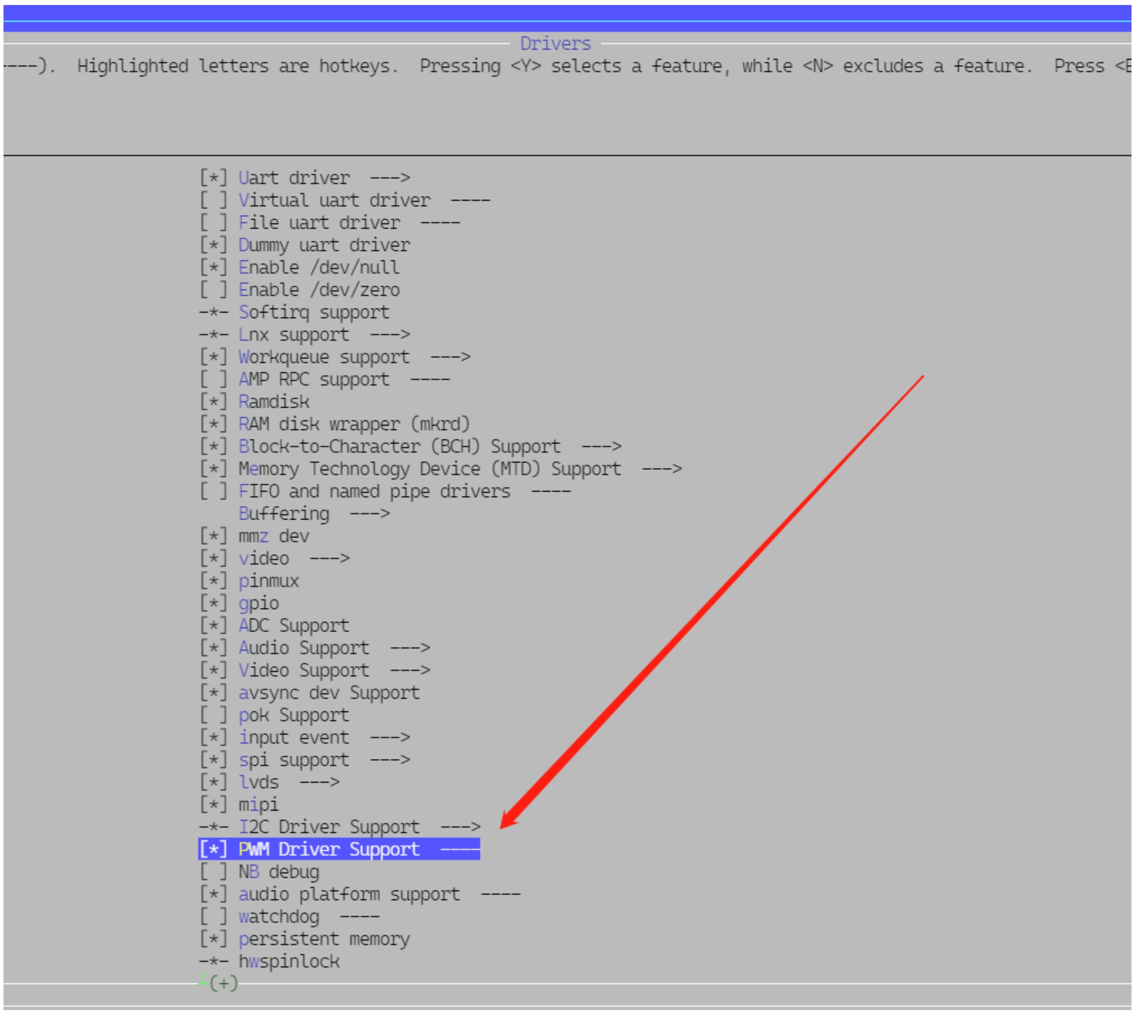
status：状态为okay代表启用；disabled代表不启用。

pwm的占空比和频率均通过代码设置，请参考sample code。

4.2 menuconfig配置

在SDK根目录输入make menuconfig，安装下面路径进行选中pwm

```
Location:
  -> Components
    -> kernel (BR2_PACKAGE_KERNEL [=y])
      -> Drivers
```



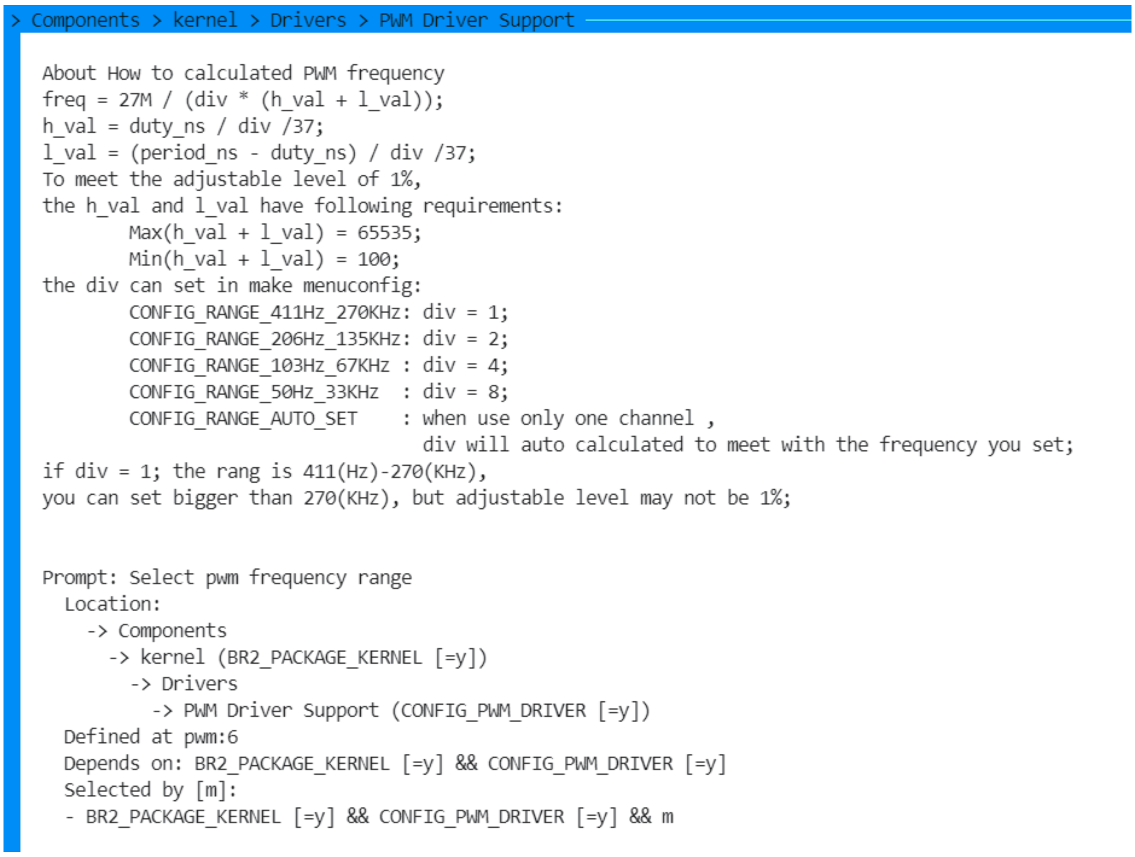
```
                                   Drivers
---).  Highlighted letters are hotkeys.  Pressing <Y> selects a feature, while <N> excludes a feature.  Press <E

                   [*] Uart driver  --->
                   [ ] Virtual uart driver  ----
                   [ ] File uart driver  ----
                   [*] Dummy uart driver
                   [*] Enable /dev/null
                   [ ] Enable /dev/zero
                   -*- Softirq support
                   -*- Lnx support  --->
                   [*] Workqueue support  --->
                   [ ] AMP RPC support  ----
                   [*] Ramdisk
                   [*] RAM disk wrapper (mkrd)
                   [*] Block-to-Character (BCH) Support  --->
                   [*] Memory Technology Device (MTD) Support  --->
                   [ ] FIFO and named pipe drivers  ----
                       Buffering  --->
                   [*] mmz dev
                   [*] video  --->
                   [*] pinmux
                   [*] gpio
                   [*] ADC Support
                   [*] Audio Support  --->
                   [*] Video Support  --->
                   [*] avsync dev Support
                   [ ] pok Support
                   [*] input event  --->
                   [*] spi support  --->
                   [*] lvds  --->
                   [*] mipi
                   -*- I2C Driver Support  --->
                   [*] PWM Driver Support  ----
                   [ ] NB debug
                   [*] audio platform support  ----
                   [ ] watchdog  ----
                   [*] persistent memory
                   -*- hwspinlock
                   (+)
```

4.2.1 pwm频率范围选择

为了满足pwm拥有0-100%，每个可调节level为1%的需求，可以设置pwm范围内的值，但也可以设置超过范围最大值的频率，只是可调节的level不确定，根据所配置的频率变化。

当只使用一个通道时，可以配置CONFIG_RANGE_AUTO_SET，分频系数会自动计算。，但超过该分频系数的范围时，可调节level不确定。

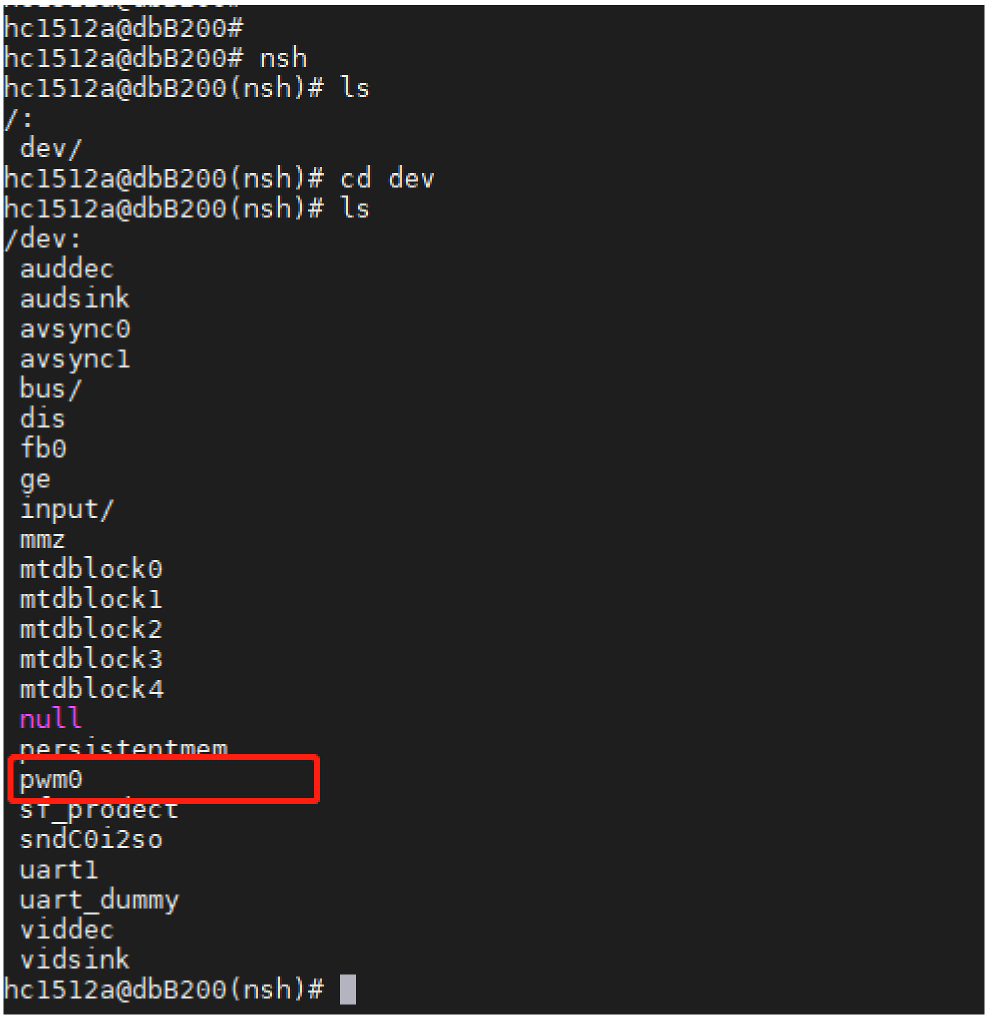


```
> Components > kernel > Drivers > PWM Driver Support

About How to calculated PWM frequency
freq = 27M / (div * (h_val + l_val));
h_val = duty_ns / div /37;
l_val = (period_ns - duty_ns) / div /37;
To meet the adjustable level of 1%,
the h_val and l_val have following requirements:
        Max(h_val + l_val) = 65535;
        Min(h_val + l_val) = 100;
the div can set in make menuconfig:
        CONFIG_RANGE_411Hz_270KHz: div = 1;
        CONFIG_RANGE_206Hz_135KHz: div = 2;
        CONFIG_RANGE_103Hz_67KHz : div = 4;
        CONFIG_RANGE_50Hz_33KHz  : div = 8;
        CONFIG_RANGE_AUTO_SET    : when use only one channel ,
                                   div will auto calculated to meet with the frequency you set;
if div = 1; the rang is 411(Hz)-270(KHz),
you can set bigger than 270(KHz), but adjustable level may not be 1%;


Prompt: Select pwm frequency range
  Location:
    -> Components
      -> kernel (BR2_PACKAGE_KERNEL [=y])
        -> Drivers
          -> PWM Driver Support (CONFIG_PWM_DRIVER [=y])
  Defined at pwm:6
  Depends on: BR2_PACKAGE_KERNEL [=y] && CONFIG_PWM_DRIVER [=y]
  Selected by [m]:
  - BR2_PACKAGE_KERNEL [=y] && CONFIG_PWM_DRIVER [=y] && m
```

配置完成后输入make kernel-rebuild all，进行编译和烧录，在串口控制台输入下面命令既可以查看pwm节点。

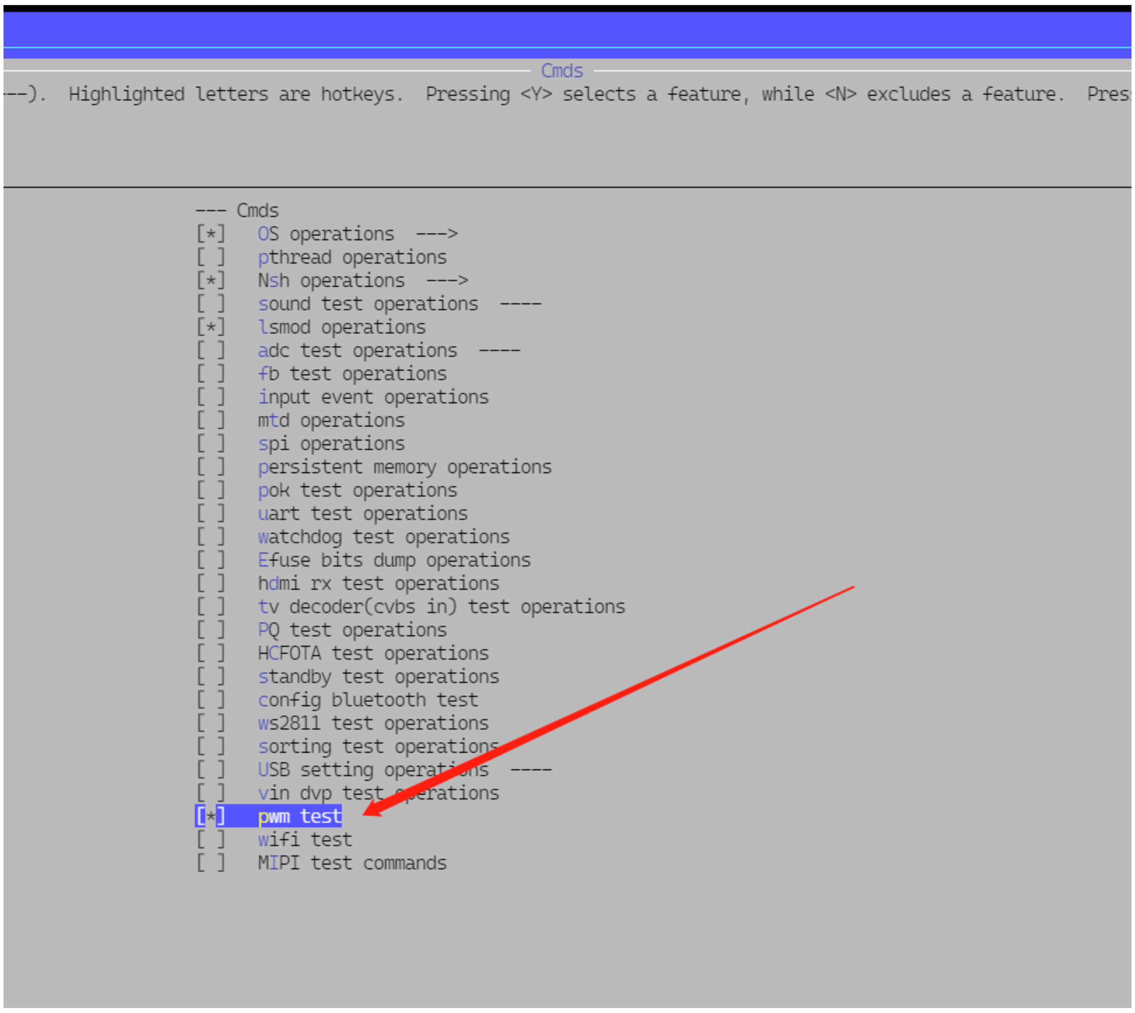


```
hc1512a@dbB200#
hc1512a@dbB200# nsh
hc1512a@dbB200(nsh)# ls
/:
 dev/
hc1512a@dbB200(nsh)# cd dev
hc1512a@dbB200(nsh)# ls
/dev:
 auddec
 audsink
 avsync0
 avsync1
 bus/
 dis
 fb0
 ge
 input/
 mmz
 mtdblock0
 mtdblock1
 mtdblock2
 mtdblock3
 mtdblock4
 null
 persistentmem
 pwm0
 sf_prodect
 sndC0i2so
 uart1
 uart_dummy
 viddec
 vidsink
hc1512a@dbB200(nsh)#
```

4.3 测试命令

在SDK根目录输入make menuconfig，根据下面路径选中pwm_test命令。

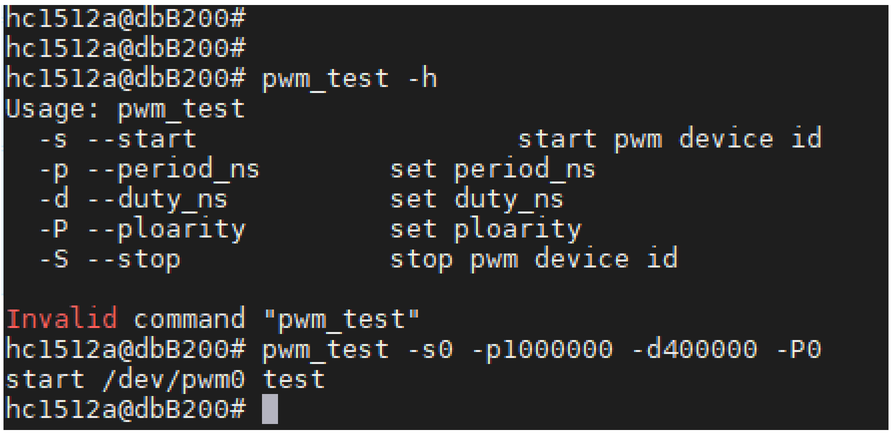
```
Location:
  -> Components
    -> Cmds (BR2_PACKAGE_CMDS [=y])
```



```
                                  Cmds
--).  Highlighted letters are hotkeys.  Pressing <Y> selects a feature, while <N> excludes a feature.  Pres

              --- Cmds
              [*]   OS operations  --->
              [ ]   pthread operations
              [*]   Nsh operations  --->
              [ ]   sound test operations  ----
              [*]   lsmod operations
              [ ]   adc test operations  ----
              [ ]   fb test operations
              [ ]   input event operations
              [ ]   mtd operations
              [ ]   spi operations
              [ ]   persistent memory operations
              [ ]   pok test operations
              [ ]   uart test operations
              [ ]   watchdog test operations
              [ ]   Efuse bits dump operations
              [ ]   hdmi rx test operations
              [ ]   tv decoder(cvbs in) test operations
              [ ]   PQ test operations
              [ ]   HCFOTA test operations
              [ ]   standby test operations
              [ ]   config bluetooth test
              [ ]   ws2811 test operations
              [ ]   sorting test operations
              [ ]   USB setting operations  ----
              [ ]   vin dvp test operations
              [*]   pwm test
              [ ]   wifi test
              [ ]   MIPI test commands
```

配置完成后，输入make cmds-rebuild all，进行编译和烧录，在串口控制台输入下面命令即可以使用测试命令。



```
hc1512a@dbB200#
hc1512a@dbB200#
hc1512a@dbB200# pwm_test -h
Usage: pwm_test
  -s --start                      start pwm device id
  -p --period_ns          set period_ns
  -d --duty_ns            set duty_ns
  -P --ploarity           set ploarity
  -S --stop               stop pwm device id

Invalid command "pwm_test"
hc1512a@dbB200# pwm_test -s0 -p1000000 -d400000 -P0
start /dev/pwm0 test
hc1512a@dbB200#
```

5. 模块接口说明

5.1 接口函数

本模块提供ioctl函数接口供应用层使用。

- 设置pwm的参数，包括占空比、周期和极性。**注意占空比和周期的单位是ns。**

```
info.polarity = polarity;
info.period_ns = period_ns;
info.duty_ns = duty_ns;
ioctl(fd, PWMIOC_SETCHARACTERISTICS, (unsigned long)&info);
```

- 获取pwm的参数，包括占空比、周期和极性。**注意占空比和周期的单位是ns。**

```
info.polarity = polarity;
info.period_ns = period_ns;
info.duty_ns = duty_ns;
ioctl(fd, PWMIOC_GETCHARACTERISTICS, (unsigned long)&info);
```

- 开启pwm和关闭pwm。

```
ioctl(fd, PWMIOC_START);
ioctl(fd, PWMIOC_STOP);
```

6. 模块测试用例与Sample Code

代码存放路径为：components/cmds/source/pwm/pwm_test.c。**注意占空比和周期的单位是ns。**

```
#include <stdio.h>
#include <unistd.h>
#include <fcntl.h>
#include <poll.h>
#include <signal.h>
#include <stdlib.h>
#include <string.h>
#include <getopt.h>
#include <sys/ioctl.h>
#include <hcuapi/pwm.h>
#include <kernel/lib/console.h>

static void print_usage(const char *prog)
{
        printf("Usage: %s \n", prog);
        puts("  -s --start          start pwm device id\n"
         "  -p --period_ns      set period_ns\n"
         "  -d --duty_ns        set duty_ns\n"
         "  -P --ploarity       set ploarity\n"
         "  -S --stop       stop pwm device id\n");
}

static int pwm_test(int argc, char *argv[])
{
```

```c
    int fd;
    int id = 0;
    bool stop_t = false;
    char path[64] = { 0 };
    struct pwm_info_s info = { 0 };
    uint32_t period_ns, duty_ns, polarity;
    period_ns = 1000000;
    duty_ns = 500000;
    polarity = 0;

    opterr = 0;
    optind = 0;

    while (1) {
        static const struct option lopts[] = {
            { "start",          1, 0, 's' },
            { "period_ns",      1, 0, 'p' },
            { "duty_ns",        1, 0, 'd' },
            { "stop",           1, 0, 'S' },
            { "polarity",       1, 0, 'P' },
            { "NULL",        0, 0, 0 },
        };

        int c;

        c = getopt_long(argc, argv, "s:p:d:S:P:", lopts, NULL);

        if (c == -1) {
            break;
        }

        switch(c) {
        case 's':
            id = atoi(optarg);
            break;
        case 'p':
            period_ns = atoi(optarg);
            break;
        case 'P':
            polarity = atoi(optarg);
            break;
        case 'd':
            duty_ns = atoi(optarg);
            break;
        case 'S':
            stop_t = true;
            id = atoi(optarg);
            break;
        default:
            print_usage(argv[0]);
            return -1;
        }
    }

    sprintf(path, "/dev/pwm%d", id);
    fd = open(path, O_RDWR);
    if (fd < 0) {
        printf("%s open fail\n", path);
```

```
            return 0;
        }

        if (stop_t == true) {
            printf("stop %s test\n", path);
            ioctl(fd, PWMIOC_STOP);
            close(fd);
            return 0;
        }

        info.polarity = polarity;
        info.period_ns = period_ns;
        info.duty_ns = duty_ns;
        ioctl(fd, PWMIOC_SETCHARACTERISTICS, (unsigned long)&info);

        ioctl(fd, PWMIOC_START);
        close(fd);
        printf("start %s test\n", path);

        return 0;
}

CONSOLE_CMD(pwm_test, NULL, pwm_test, CONSOLE_CMD_MODE_SELF, "pwm test
operations")
```

7. 模块调试方法

可以用示波器或逻辑分析仪抓取波形查看。

8. 常见问题

暂无。