树莓派Python开发第8课： PWM实验

原创

下家山 于 2020-10-12 15:16:38 发布 4457 收藏 67

分类专栏： 树莓派Python开发 文章标签： 树莓派python pwm开发 树莓派python脉宽调制 树莓派Python呼吸灯 树莓派Python软件pwm

版权声明：本文为博主原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接和本声明。

本文链接：https://blog.csdn.net/qq_27320195/article/details/109026574

版权

树莓派Python开发 专栏收录该内容

9 篇文章 20 订阅

订阅专栏

什么是PWM

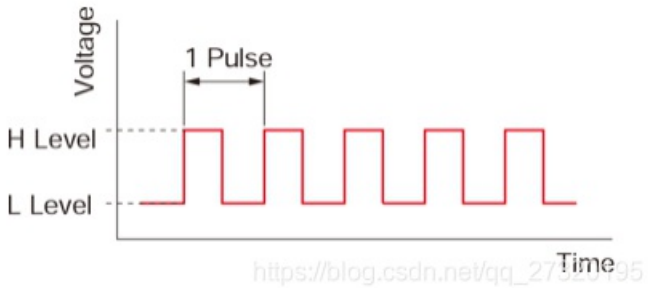
PWM是Pulse Width Modulation (PWM)的缩写，意思是脉冲宽度调制，那什么是脉冲呢？

脉冲

脉冲是一种信号，我们平常听到“脉搏”这个词，我们这里的脉冲和这个脉搏其实很相似，脉搏是跳动的，有规律的，脉冲也是这样，是有规律的，这个规律在编程领域叫频率，或者叫周期。如果要给脉冲下一个定义的话：

脉冲是一种电压反复在高和低(H和L)之间改变的电信号。

就像下面这张图：



什么是脉宽调制

脉宽很好理解，就是脉冲的宽度，高电平表示有脉冲信号，低电平表示没有脉冲信号，这个宽度就是高电平的持续时间，或者叫长度也行。

那么，调制呢，就是这个高电平的持续时间是可以调整的，可以通过程序改变，动态的改变，当然也可以通过硬件电路实现调整，比如我们用得比较多的旋钮，可以把台灯调到最亮，或者最暗。

树莓派如何实现PWM

有了上面的知识铺垫，就可以来理解树莓派的实现原理了。
首先，树莓派有一个专门类叫pwm，这个类的使用非常简单。
我们可以先通过help查看帮助，就像这样：

```
>>> import RPi.GPIO
>>> help(RPi.GPIO.PWM)
Help on class PWM in module RPi.GPIO:

class PWM(__builtin__.object)
 |  Pulse Width Modulation class
 |
 |  Methods defined here:
 |
 |  ChangeDutyCycle(...)
 |      Change the duty cycle
 |      dutycycle - between 0.0 and 100.0
 |
 |  ChangeFrequency(...)
 |      Change the frequency
 |      frequency - frequency in Hz (freq > 1.0)
 |
 |  __init__(...)
 |      x.__init__(...) initializes x; see help(type(x)) for signature
 |
 |  start(...)
 |      Start software PWM
 |      dutycycle - the duty cycle (0.0 to 100.0)
 |
 |  stop(...)
 |      Stop software PWM
 |
 |  ----------------------------------------------------------------------
 |  Data and other attributes defined here:
 |
 |  __new__ = <built-in method __new__ of type object>
 |      T.__new__(S, ...) -> a new object with type S, a subtype of T
(END)
```

https://blog.csdn.net/qq_27320195

但是，真正使用，你看这个帮助还是看不到实际有价值的内容，这就是用Python控制树莓派的一些弊端，因为它很多的理论和原理是基于C语言的，所以一般开发我们要结合C，不清楚的去查看C的代码，这里并非说用python不好，其实有些地方用python非常简单，比如I2C，SPI这些比较复杂的，如果有人用Python整合好了，直接安装模块就可以使用，没必要再去造轮子。
所以呢，有我帮大家总结出来了，大家收藏我这篇博客就行了。
步骤是这样的：

step1: 生产pwm类对象

PR = gpio.PWM(18,50)
上面这条语句是调用gpio模块中的PWM类，生成一个对象PR，PWM是类名，也是构造函数，传进去的两个参数，第一个是引脚编号，第二个参数是PWM的频率，50代表50HZ。

step2: 启动pwm

PR.start(0) #这个函数是启动pwm功能，里面有一个参数，这个参数如果填0，代表脉冲宽度为0，也就是没有脉冲，在led灯上表现就是不亮，如果是电机，表现就是轮子不转。

step3: 改变脉冲宽度

PR.ChangeDutyCycle(dc) #改变脉冲宽度是通过ChangeDutyCycle函数，这个函数你可以传进去你要设置的脉冲宽度值，这个值只能是0~100.

step4: 停止pwm

PR.stop() #停止pwm
当你要停止pwm的时候，可以调用stop函数。不需要传参数。

实验方法

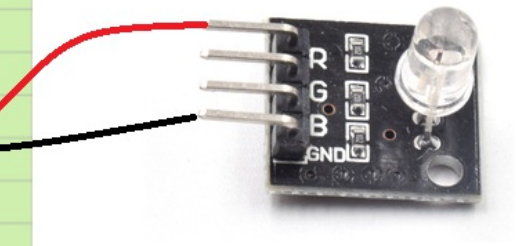
我们只需要接一个led灯到18号引脚，然后配置为输出功能，接着就按照上面的步骤就可以编程了。G和B不需要连接。

树莓派 40Pin 引脚对照表

wiringPi 编码	BCM 编码	功能名	物理引脚 BOARD编码		功能名	BCM 编码	wiringPi 编码
		3.3V	1	2	5V		
8	2	SDA.1	3	4	5V		
9	3	SCL.1	5	6	GND		
7	4	GPIO.7	7	8	TXD	14	15
		GND	9	10	RXD	15	16
0	17	GPIO.0	11	12	GPIO.1	18	1
2	27	GPIO.2	13	14	GND		
3	22	GPIO.3	15	16	GPIO.4	23	4
		3.3V	17	18	GPIO.5	24	5
12	10	MOSI	19	20	GND		
13	9	MISO	21	22	GPIO.6	25	6
14	11	SCLK	23	24	CE0	8	10
		GND	25	26	CE1	7	11
30	0	SDA.0	27	28	SCL.0	1	31
21	5	GPIO.21	29	30	GND		
22	6	GPIO.22	31	32	GPIO.26	12	26
23	13	GPIO.23	33	34	GND		
24	19	GPIO.24	35	36	GPIO.27	16	27
25	26	GPIO.25	37	38	GPIO.28	20	28
		GND	39	40	GPIO.29	21	29

表格由树莓派实验室绘制 http://shumeipai.nxez.com

NXEZ.COM



https://blog.csdn.net/qq_27320195

实现呼吸灯

#coding:utf-8 #设置中文编码模式，支持中文注释

导入模块
import RPi.GPIO as gpio #导入树莓派模块RPi.GPIO
import time #导入时间模块，方便后面做延时
R = 18 #设置BCM模式第18号引脚作为led灯控制引脚
gpio.setwarnings(False) #屏蔽警告

设置BCCM编码模式
gpio.setmode(gpio.BCM) #设置引脚模式BCM

设置引脚功能
gpio.setup(R,gpio.OUT) #设置引脚方向为输出 ，Python中，PWM功能是软件模拟的，就是通过控制高电平宽度来实现
#所以，引脚必须设置输出
PR = gpio.PWM(R,50) #这种PWM频率为50Hz
PR.start(0) #设置启动脉冲宽度，0表示开始没有脉冲，你可以把这个参数想象成起步的速度是多少码，
#是20码起步，还是0速度
4）实现呼吸灯

while 1:
for dc in range(0,100,5): #python中调试pwm，规定pwm宽度范围是0~100，5是每次循环累加的步长
PR.ChangeDutyCycle(dc) #PWM真正调制的地方就在这里
time.sleep(0.5) #不用太快，否则人眼看不到变化
time.sleep(1)
for dc in range(100,0,-5): #由亮到暗的过程
PR.ChangeDutyCycle(dc)
time.sleep(0.5)
完整代码：

```
pi@xiajiashan:~/pi-python-blog$ cat pwm_led.py
#!/usr/bin/python
#coding:utf-8
import RPi.GPIO as gpio
import time
R = 18
gpio.setwarnings(False)#屏蔽警告
gpio.setmode(gpio.BCM)#设置引脚模式BCM
gpio.setup(R,gpio.OUT)#设置引脚11方向输出
PR = gpio.PWM(R,50)
PR.start(0)
try:                    #try和except是一对，通过它来捕捉在执行while 1的时候，是否按下了ctrl+c按键，crtl+c是终止程序
   while 1:
       for dc in range(0,100,5):
           PR.ChangeDutyCycle(dc)
           time.sleep(0.5)
       time.sleep(1)
       for dc in range(100,0,-5):
           PR.ChangeDutyCycle(dc)
           time.sleep(0.5)
except KeyboardInterrupt:      #如果按下了ctrl+c进行异常处理，让pwm停止，释放引脚
    PR.stop()
    gpio.cleanup()
pi@xiajiashan:~/pi-python-blog$
```

直接运行(python无需编译)

```
pi@xiajiashan:~/pi-python-blog$ python pwm_led.py
```

效果请看视频教程
https://edu.csdn.net/course/play/28043/383541?spm=1002.2001.3001.4143

软件PWM

Python中的PWM功能是通过软件模拟的，软件模拟的意思就是它的脉冲宽度是我们程序打高电平来实现的，不是cpu硬件做到的，C语言中有一个引脚（BCM的第18脚）可以实现硬逻辑，但是python里面不可以。
python中的PWM是通过C语言的函数实现的，python只不过是做了一个整合调用而已，其实整个的树莓派中，Python编程都是用的C语言函数库，其实这也没什么，大家不要说python没有技术含量之类的话，本来就不需要造轮子，但是你要理解核心，需要看懂C语言中的代码。
具体软件PWM模拟的原理，请大家参考我C语言中的讲解。
软件PWM内部实现原理

如何通过软件PWM实现三色灯的呼吸功能

因为上面原理性的东西已经介绍得很详细了，这里就只给大家源码，下面有注释

```
pi@xiajiashan:~/pi-python-blog$ cat pwm_rgb.py
#coding:utf-8
import RPi.GPIO as gpio
import time
R = 16  #设置红色灯接到第16脚(BCM编码模式)
G = 20  #设置绿色灯接到第20脚(BCM编码模式)
B = 21  #设置蓝色灯接到第21脚(BCM编码模式)
gpio.setwarnings(False)  #去掉警告信息
gpio.setmode(gpio.BCM)  #设置引脚为BCM编码模式
gpio.setup(R,gpio.OUT)  #PWM功能控制引脚，需要改引脚设置为输出
gpio.setup(G,gpio.OUT)  #
gpio.setup(B,gpio.OUT)  #
PWMR = gpio.PWM(R,50)  #设置R为PWM功能，并且设置频率为50HZ，也就是PWM周期是20ms
PWMG = gpio.PWM(G,50)  #设置G为PWM功能，并且设置频率为50HZ，也就是PWM周期是20ms
PWMB = gpio.PWM(B,50)  #设置B为PWM功能，并且设置频率为50HZ，也就是PWM周期是20ms
PWMR.start(0)  #启动R 的PWM
PWMG.start(0)  #启动G 的PWM
PWMB.start(0)  #启动B 的PWM
try:  #捕捉ctrl+c按键
   while 1:
       #R慢慢变亮
       for dc in range(0,100,5): #定义循环变量dc从0~100，每次增加5
           PWMR.ChangeDutyCycle(dc)  #调制PWM周期
           time.sleep(0.5)
       time.sleep(2)
       #R慢慢变暗
       for dc in range(100,0,-5): #循环变量从100~0，每次递减5
           PWMR.ChangeDutyCycle(dc)  #调制PWM周期
           time.sleep(0.5)
       #G慢慢变亮
       for dc in range(0,100,5): #定义循环变量dc从0~100，每次增加5
           PWMG.ChangeDutyCycle(dc)  #调制PWM周期
           time.sleep(0.5)
       time.sleep(2)
       #R慢慢变暗
       for dc in range(100,0,-5): #循环变量从100~0，每次递减5
           PWMG.ChangeDutyCycle(dc)  #调制PWM周期
           time.sleep(0.5)
       #B慢慢变亮
       for dc in range(0,100,5): #定义循环变量dc从0~100，每次增加5
           PWMB.ChangeDutyCycle(dc)  #调制PWM周期
           time.sleep(0.5)
       time.sleep(2)
       #R慢慢变暗
       for dc in range(100,0,-5): #循环变量从100~0，每次递减5
           PWMB.ChangeDutyCycle(dc)  #调制PWM周期
           time.sleep(0.5)
except KeyboardInterrupt:
   PWMB.stop()    #捕捉到ctrl+c信号后停止pwm
   gpio.cleanup() #释放引脚资源
pi@xiajiashan:~/pi-python-blog$
```

END

想通过视频学习的可以购买我的视频教程：

https://edu.csdn.net/course/play/28043/383541?spm=1002.2001.3001.4143

希望本文能帮到各位，如果怕忘了就收藏吧！

第一课：什么是树莓派

第二课：树莓派能做什么

第三课：购买您的第一个树莓派

第四课：如何安装树莓派系统

第五课：树莓派Python编程手册

第六课：树莓派Python Led实验

第七课：树莓派Python 按键实验

第八课：树莓派Python开发 PWM实验

视频链接