前面的文章我们对***WiringPi***软件包做了简单的介绍，今天这篇文章，我们来认识下***WiringPi***的配置函数。

在第一节中我们曾介绍过***WiringPi***对树莓派的引脚进行了封装，可以使用

`$sudo gpio readall`

读出树莓派的引脚定义，如下图：

```
pi@raspberrypi:~ $ gpio readall
 +-----+-----+---------+------+---+---Pi 3B--+---+------+---------+-----+-----+
 | BCM | wPi |   Name  | Mode | V | Physical | V | Mode | Name    | wPi | BCM |
 +-----+-----+---------+------+---+----++----+---+------+---------+-----+-----+
 |     |     |    3.3v |      |   |  1 || 2  |   |      | 5v      |     |     |
 |   2 |   8 |   SDA.1 |   IN | 1 |  3 || 4  |   |      | 5v      |     |     |
 |   3 |   9 |   SCL.1 |   IN | 1 |  5 || 6  |   |      | 0v      |     |     |
 |   4 |   7 | GPIO. 7 |   IN | 1 |  7 || 8  | 0 | IN   | TxD     | 15  | 14  |
 |     |     |      0v |      |   |  9 || 10 | 1 | IN   | RxD     | 16  | 15  |
 |  17 |   0 | GPIO. 0 |   IN | 0 | 11 || 12 | 0 | IN   | GPIO. 1 | 1   | 18  |
 |  27 |   2 | GPIO. 2 |   IN | 0 | 13 || 14 |   |      | 0v      |     |     |
 |  22 |   3 | GPIO. 3 |   IN | 0 | 15 || 16 | 0 | IN   | GPIO. 4 | 4   | 23  |
 |     |     |    3.3v |      |   | 17 || 18 | 0 | IN   | GPIO. 5 | 5   | 24  |
 |  10 |  12 |    MOSI |   IN | 0 | 19 || 20 |   |      | 0v      |     |     |
 |   9 |  13 |    MISO |   IN | 0 | 21 || 22 | 1 | IN   | GPIO. 6 | 6   | 25  |
 |  11 |  14 |    SCLK |   IN | 0 | 23 || 24 | 1 | IN   | CE0     | 10  | 8   |
 |     |     |      0v |      |   | 25 || 26 | 1 | IN   | CE1     | 11  | 7   |
 |   0 |  30 |   SDA.0 |   IN | 1 | 27 || 28 | 1 | IN   | SCL.0   | 31  | 1   |
 |   5 |  21 | GPIO.21 |   IN | 1 | 29 || 30 |   |      | 0v      |     |     |
 |   6 |  22 | GPIO.22 |   IN | 1 | 31 || 32 | 0 | IN   | GPIO.26 | 26  | 12  |
 |  13 |  23 | GPIO.23 |   IN | 0 | 33 || 34 |   |      | 0v      |     |     |
 |  19 |  24 | GPIO.24 |   IN | 0 | 35 || 36 | 0 | IN   | GPIO.27 | 27  | 16  |
 |  26 |  25 | GPIO.25 |   IN | 0 | 37 || 38 | 0 | IN   | GPIO.28 | 28  | 20  |
 |     |     |      0v |      |   | 39 || 40 | 0 | IN   | GPIO.29 | 29  | 21  |
 +-----+-----+---------+------+---+----++----+---+------+---------+-----+-----+
 | BCM | wPi |   Name  | Mode | V | Physical | V | Mode | Name    | wPi | BCM |
 +-----+-----+---------+------+---+---Pi 3B--+---+------+---------+-----+-----+
pi@raspberrypi:~ $
```

可以看到,树莓派的引脚有三种定义方式：

1）按照实际物理位置进行定义（物理编号）；

2）按照Broadcom引脚编号进行定义（BCM编号）；

3）按照类似Arduino的方式进行定义（WiringPi编号）；

同一个引脚采用不同的定义方式其编号是不同的。举个例子，WiringPi编号为0的引脚，在BCM编号中其值为17，在物理编号中其值为11。因此，在实际进行编程时，需要明确采用哪种编号方式，这就要用到***WiringPi***的配置函数。

WiringPi软件包提供了四个配置函数来初始化树莓派的针脚，包括：

- int wiringPiSetup (void) ;
- int wiringPiSetupGpio (void) ;
- int wiringPiSetupPhys (void) ;
- int wiringPiSetupSys (void) ;

1、wiringPiSetup

该函数使用WiringPi编号方式对树莓派引脚进行初始化，没有参数，通常也不需要关心它的返回值；调用该函数需要root权限；

2、wiringPiSetupGpio

该函数使用BCM编号方式对树莓派引脚进行初始化，没有参数，通常也不需要关心它的返回值；调用该函数需要root权限；

3、wiringPiSetupPhys

该函数使用物理编号方式对树莓派引脚进行初始化，没有参数，通常也不需要关心它的返回值；调用该函数需要root权限；

4、wiringPiSetupSys

该函数与wiringPiSetupGpio类似，也是采用BCM编号方式对树莓派引脚进行初始化。所不同的是，该函数并不是访问实际的硬件，而是对*/sys/class/gpio*接口进行操作。该函数可以在没有root权限的情况下对树莓派引脚进行初始化，当前，前提是需要访问的引脚已经被映射到*/sys/class/gpio*下了；

以上就是WiringPi的配置函数，编程时必须调用其中某个对引脚进行初始化。

关于配置函数就先介绍这么多，后续我们会对***WiringPi***的其它函数进行介绍。

欢迎扫描下方的二维码关注我们的微信公众号。


方正智芯
Founder Chip
长按扫码关注我们
方正智芯
公众号：founderchip
官方网站：www.founderchip.com
原创工业智能控制领域（PLC、单片机、通信）的技术分享