树莓派开发

13 认识树莓派的GPIO

1、认识GPIO位置

- GPIO（General Purpose Input Output，通用输入/输出）
- 树莓派上的GPIO接口在哪里？它是怎么定义的？看两幅图：

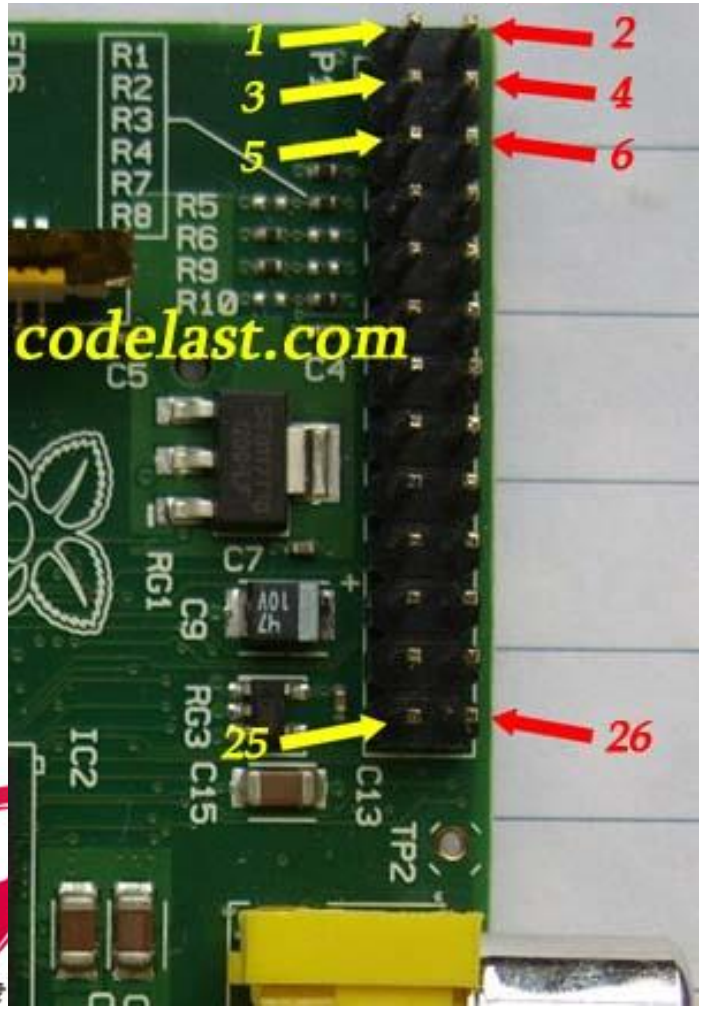
- （1）实物图
  - 这是树莓派的正面图
  - 可见PGIO26针
  - 在方框中，左上角为1、右上脚为2
  - 左排位单数、右排位双数

1、认识GPIO位置

- 将Pi的电路板翻到反面，在26个针脚中，焊点为方形的那个针脚就是“1”，在它旁边同一排（两个针脚一排）的那一个就是“2”，依此类推。

1、认识GPIO位置

- 根据这个判断，得到了Pi的GPIO接口顺序图：



2、树莓派GPIO编号方式

1、物理引脚：

从左到右，从上到下：左边基数，右边偶数：1-40（树莓派版本不同，个数不同，A、B型早期为26个，B+为40个）

2、BCM:

编号侧重CPU寄存器，根据BCM2835的GPIO寄存器编号。

3、wpi：

编号侧重实现逻辑，把扩展GPIO端口从0开始编号，这种编号方便编程。

2、树莓派GPIO编号方式

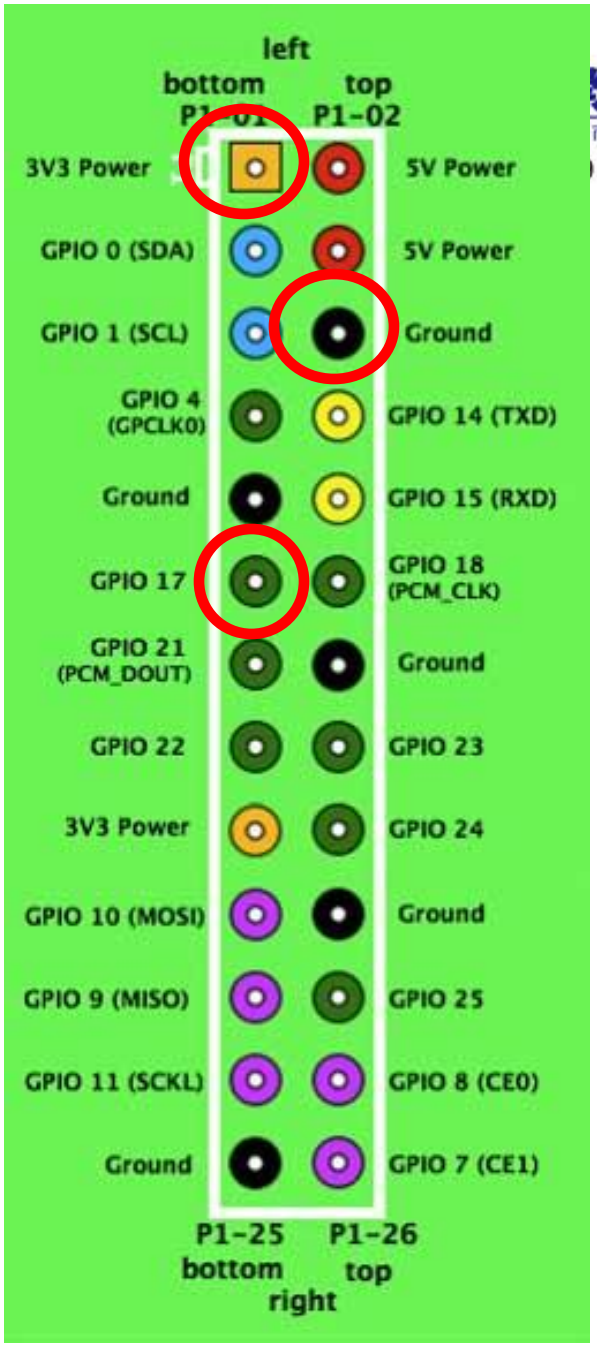
BCM 编码方式　wpi 编码方式　功能名　物理接口　功能名　wpi 编码方式　BCM 编码方式

BCM	wPi	Name	Mode	V	B Plus Physical	V	Mode	Name	wPi	BCM
		3.3v			1 || 2			5v		
2	8	SDA.1	ALT0	1	3 || 4			5V		
3	9	SCL.1	ALT0	1	5 || 6			0v		
4	7	GPIO. 7	IN	1	7 || 8	0	ALT0	TxD	15	14
		0v			9 || 10	1	ALT0	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	1	OUT	GPIO. 5	5	24
10	12	MOSI	ALT0	0	19 || 20			0v		
9	13	MISO	ALT0	1	21 || 22	1	OUT	GPIO. 6	6	25
11	14	SCLK	ALT0	1	23 || 24	1	ALT0	CE0	10	8
		0v			25 || 26	1	ALT0	CE1	11	7
0	30	SDA.0	ALT0	1	27 || 28	1	ALT0	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 B Plus	V	Mode	Name	wPi	BCM

3、认识GPIO接口定义

- 树莓派上的GPIO接口在哪里？它是怎么定义的？看两幅图：

- （2）定义图
  - 这是树莓派PGIO26脚定义图
- 要用到的是：
  - Pin01（Power电源正）
  - Pin06（Ground电源接地）
  - Pin11（GPIO17控制信号）
- 知道了脚的定义，换一个相同定义的脚也没有问题了。



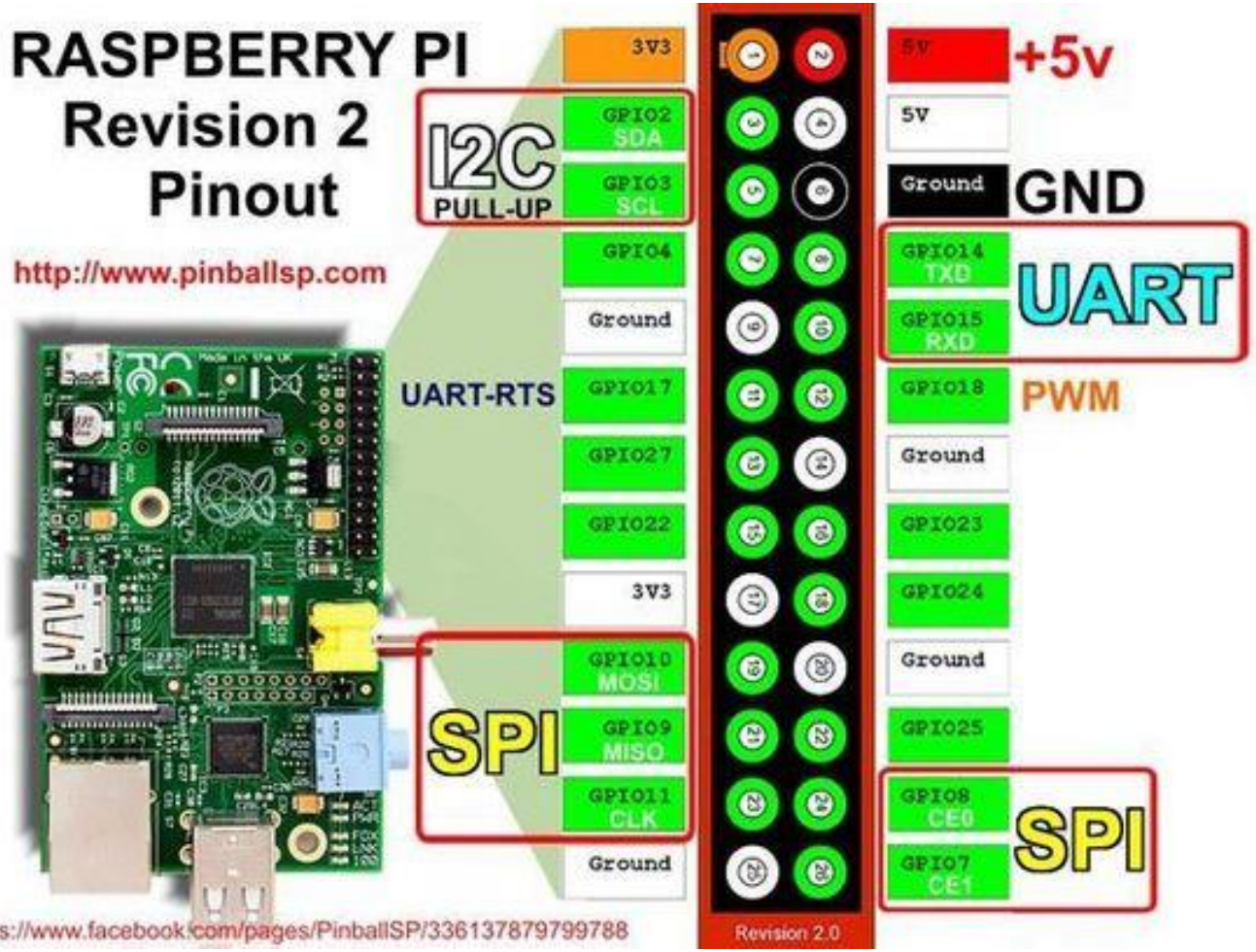
3、认识GPIO接口定义

- 在树莓派的PIN脚编号旁边，还有其他的字母
- 这是这些脚的“专业”名称
- 例如：
  - 3和5是SDA和SCL，它们分别是I2C串行总线的时钟和数据线。I2C用于温度传感器和LCD显示器
  - 8和10用于串行通信的RX和TX（接收发送），另一种串行通信（我们现在用的SPI）则使用11、13、15）。我们现在用11

3.3V	1	2	5V
I2C0 SDA	3	4	DNC
I2C0 SCL	5	6	GROUND
GPIO4	7	8	UART TXD
DNC	9	10	UART RXD
GPIO 17	11	12	GPIO 18
GPIO 21	13	14	DNC
GPIO 22	15	16	GPIO 23
DNC	17	18	GPIO 24
SP10 MOSI	19	20	DNC
SP10 MISO	21	22	GPIO 25
SP10 SCLK	23	24	SP10 CE0 N
DNC	25	26	SP10 CE1 N

3、认识GPIO接口定义

- 这是一张专业分组的接口图



3、认识GPIO接口定义-针脚编号

- 两种接口命名方法对照：

Phyical / Raspberry Pi Name			
3.3v	1	2	5v
SDA0	3	4	DNC
SCL0	5	6	0v
GPIO 7	7	8	TX
DNC	9	10	RX
GPIO 0	11	12	GPIO 1
GPIO 2	13	14	DNC
GPIO 3	15	16	GPIO 4
DNC	17	18	GPIO 5
SPI MOSI	19	20	DNC
SPI MISO	21	22	GPIO 6
SPI SCLK	23	24	SP10 CEO N
DNC	26	26	SP10 CE1 N

Broadcom names			
3.3v	1	2	5v
I2C0 SDA	3	4	DNC
I2C0 SCL	5	6	0v
GPIO 4	7	8	UART TXD
DNC	9	10	UART RXD
GPIO 17	11	12	GPIO 18
GPIO 21	13	14	DNC
GPIO 22	15	16	GPIO 23
DNC	17	18	GPIO 24
SPI MOSI	19	20	DNC
SPI MISO	21	22	GPIO 25
SPI SCLK	23	24	SP10 CEO N
DNC	26	26	SP10 CE1 N

3、认识GPIO接口定义-针脚编号

- 目前有两种方式可以通过 RPi.GPIO 对 Raspberry Pi 上的 IO 针脚进行编号。
- 第一种方式是使用 BOARD 编号系统。
  - 该方式参考 Raspberry Pi 主板上 P1 接线柱的针脚编号。使用该方式的优点是无需考虑主板的修订版本，您硬件始终都是可用的状态。您将无需从新连接线路和更改您的代码。
- 第二种方式是使用 BCM 编号。
  - 这是一种较低层的工作方式 - 该方式参考 Broadcom SOC 的通道编号。使用过程中，您始终要保证主板上的针脚与图表上标注的通道编号相对应。您的脚本可能在 Raspberry Pi 主板进行修订版本更新时无法工作。
- 一般情况下，要指定使用哪种模式，否则会报错！

3、认识GPIO接口定义-针脚编号

□指定所使用的方式（必须指定）：

GPIO.setmode(GPIO.BOARD)

或者

GPIO.setmode(GPIO.BCM)

警告

可能Raspberry Pi 的 GPIO 上同时有多个脚本/循环。因此，如果 RPi.GPIO 检测到某个针脚被设置为其它用途而非默认的状态（默认为输入），您会在尝试配置某脚本时得到警告消息。

禁用该警告消息：

GPIO.setwarnings(False)

3、认识GPIO接口定义-配置通道

- 需要为每个用于输入或输出的针脚配置通道。
- 配置为输入的通道：

GPIO.setup(channel, GPIO.IN)

- 通道编号是基于所使用的编号系统所指定的（BOARD 或 BCM）。
- 配置为输出的通道：

GPIO.setup(channel, GPIO.OUT)

- 通道编号是基于您所使用的编号系统所指定的（BOARD 或 BCM）。
- 还可以指定输出通道的初始值：

GPIO.setup(channel, GPIO.OUT, initial=GPIO.HIGH)

3、认识GPIO接口定义-输入输出

- **输入**，读取 GPIO 针脚的值：

GPIO.input(channel)

- 通道编号是基于您所使用的编号系统所指定的（BOARD 或 BCM）。这将返回 0 / GPIO.LOW / False 或者 1 / GPIO.HIGH / True。

- **输出**，设置 GPIO 针脚的输出状态：

GPIO.output(channel, state)

- 通道编号是基于您所使用的编号系统所指定的（BOARD 或 BCM）。状态可以为 0 / GPIO.LOW / False 或者 1 / GPIO.HIGH / True。

3、认识GPIO接口定义-清理

- 清理
- 在任何程序结束后，请养成清理用过的资源的好习惯。使用 RPi.GPIO 也同样需要这样。
- 恢复所有使用过的通道状态为输入，您可以避免由于短路意外损坏您的 Raspberry Pi 针脚。
- 注意，该操作仅会清理您的脚本使用过的 GPIO 通道。
- 在您的脚本结束后进行清理的命令是：

GPIO.cleanup()

4、验证GPIO接口

- 同时，为了保险，还需要用万用表测量一下刚才的判断：
- 在Raspberry Pi的工作状态下，将万用表的“+”接到“1”口上，万用表的“-”接到“6”口上，在看看万用表的显示是不是大约为正3.3V，如果不是，那么说明我们弄错了针脚顺序，需要回头再检查！

5、常用GPIO开源工程

树莓派内核中已经编译自带了gpio的驱动，我们常通过一些第三方写好的库函数来完成具体的操作，比较常见的操作库函数有3种：

1、python GPIO

【开发语言】——python（类似C语言）

【简单介绍】——树莓派官方资料中推荐且容易上手。python GPIO是一个小型的python库，可以帮助用户完成raspberry相关IO口操作，但是python GPIO库还没有支持SPI、I2C或者1-wire等总线接口。

【官方网站】—— https://code.google.com/p/raspberry-gpio-python/

我们的第一个例子，就是用python GPIO实现的

/

5、常用GPIO开源工程

2、wiringPi

【开发语言】——C语言

【简单介绍】——wiringPi适合那些具有C语言基础，在接触树莓派之前已经接触过单片机或者嵌入式开发的人群。wiringPi的API函数和arduino非常相似，这也使得它广受欢迎。作者给出了大量的说明和示例代码，这些示例代码也包括UART设备，I2C设备和SPI设备等。

【官方网站】—— http://wiringpi.com/

如果有时间的话，我们也用wiringPi，把第一个例子再做一遍。

5、常用GPIO开源工程

3、BCM2835 C Library

【开发语言】——C语言

【简单介绍】BCM2835 C Library可以理解为使用C语言实现的相关底层驱动，BCM2835 C Library的驱动库包括GPIO、SPI和UART等，可以通过学习BCM2835 C Library熟悉BCM2835相关的寄存器操作。如果有机会开发树莓派上的linux驱动，或自主开发python或PHP扩展驱动，可以从BCM2835 C Library找到不少的“灵感”。

【官方网站】——

http://www.airspayce.com/mikem/bcm2835/

最后，这个库也请有兴趣的同学尝试一下吧。