QUECTEL

Build a smarter world

移远通信

嵌入式软件开发课程



软件部-蒙政威

01 GPIO的工作原理

Part One

电路结构与输入输出模式

Build a smarter world

GPIO的电路结构

GPIO：

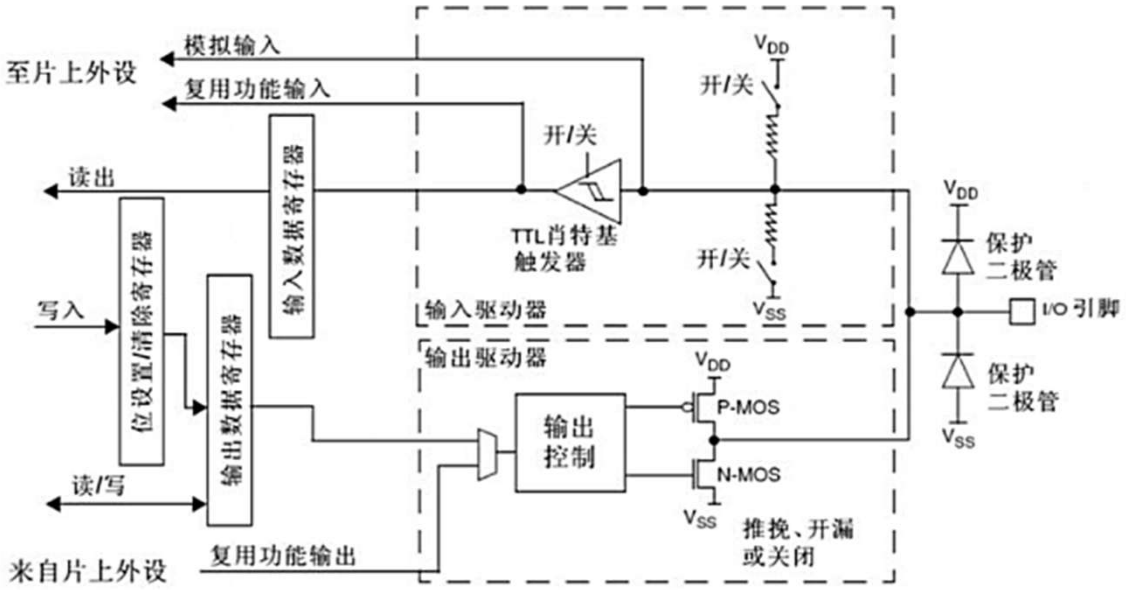
General-purpose input/output

输入模式：

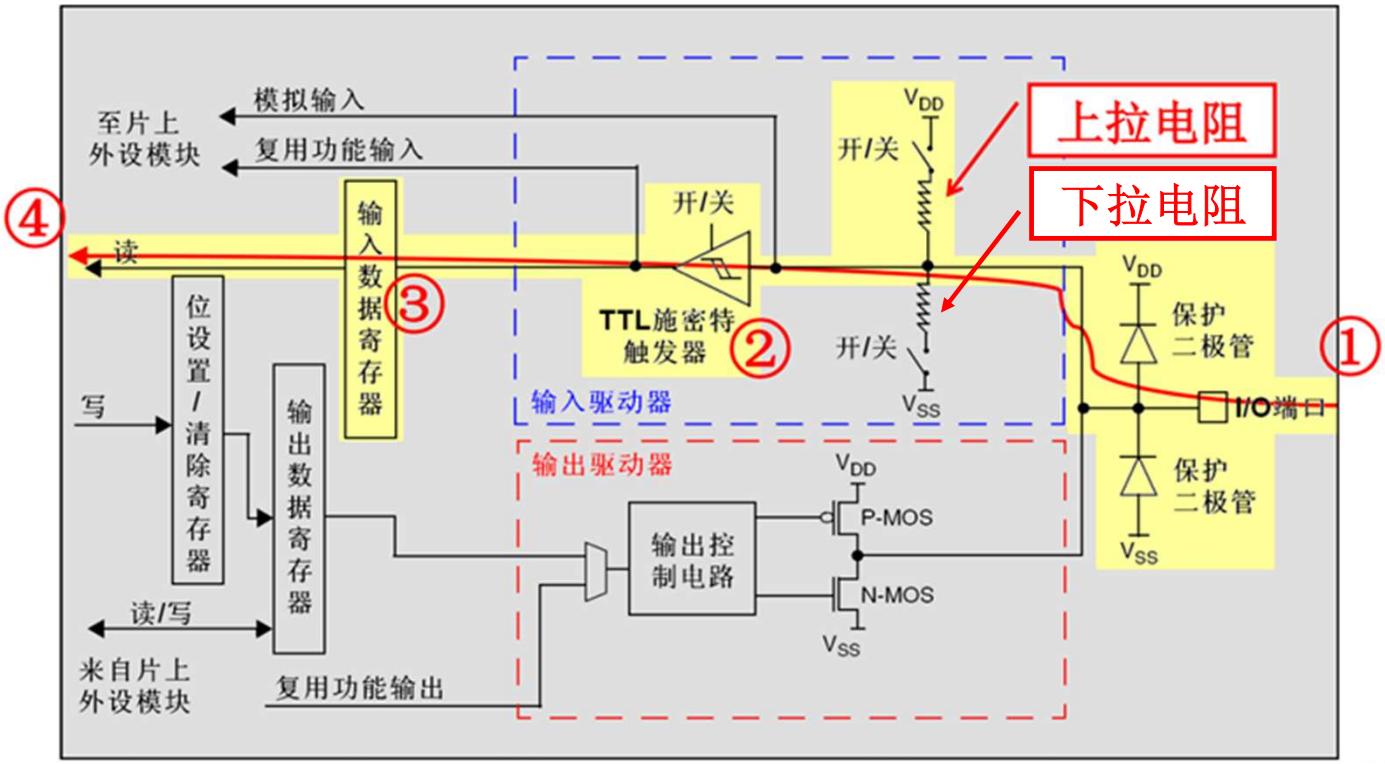
- 上拉输入
- 下拉输入
- 浮空输入
- 模拟输入

输出模式：

- 推挽输出
- 开漏输出



GPIO输入模式——上拉输入与下拉输入



上拉电阻

集成在芯片内部，当IO口悬空时，可读取到高电平。

下拉电阻

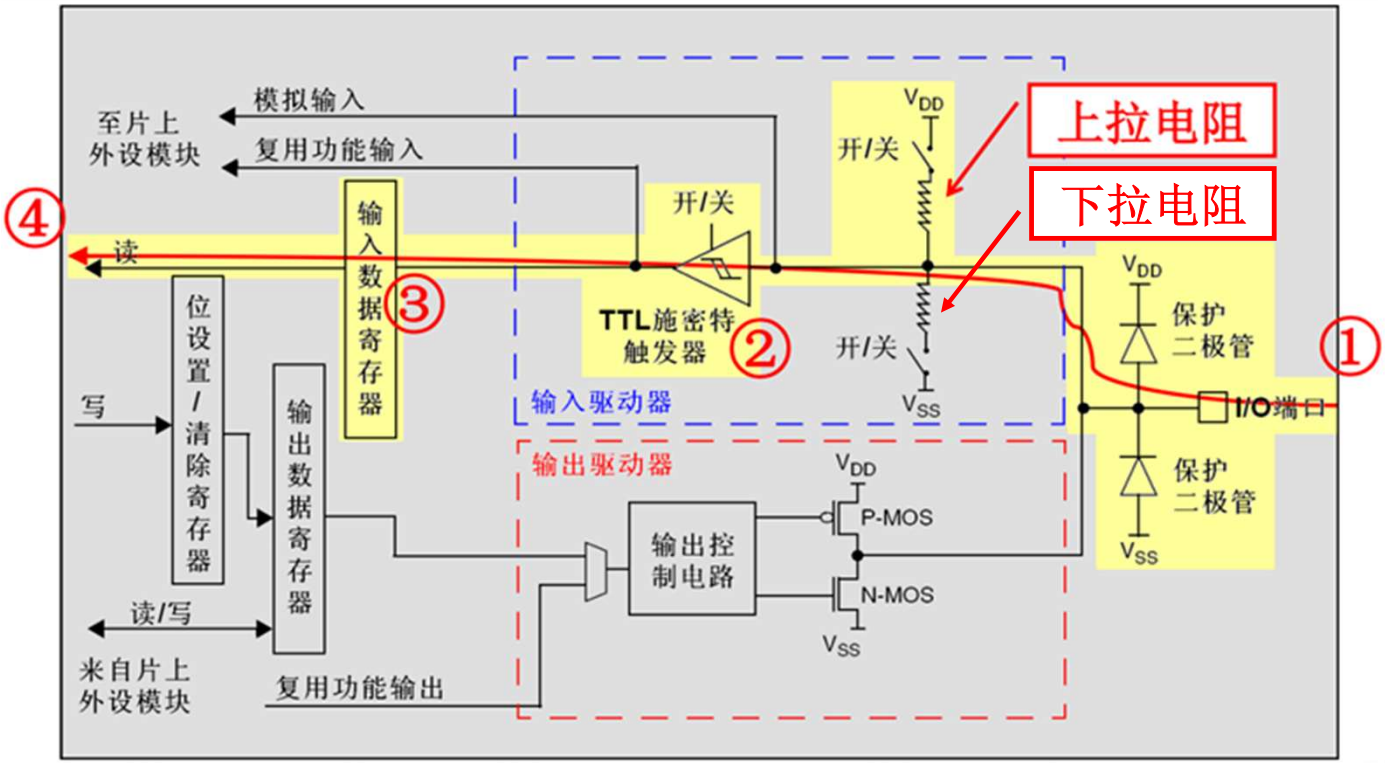
集成在芯片内部，当IO口悬空时，可读取到低电平。

上拉还是下拉，由软件配置寄存器来确定。

应用场景

按键扫描

GPIO输入模式——上拉输入与下拉输入



钳位二极管

假设上下两个二极管为D1和D2，二极管的导通电压为V_d，IO口电压标记为V_{io}：

当$V_{io} > V_{DD} + V_d$，D1导通，D2截止，V_{io}被限制在$V_{DD} + V_d$；

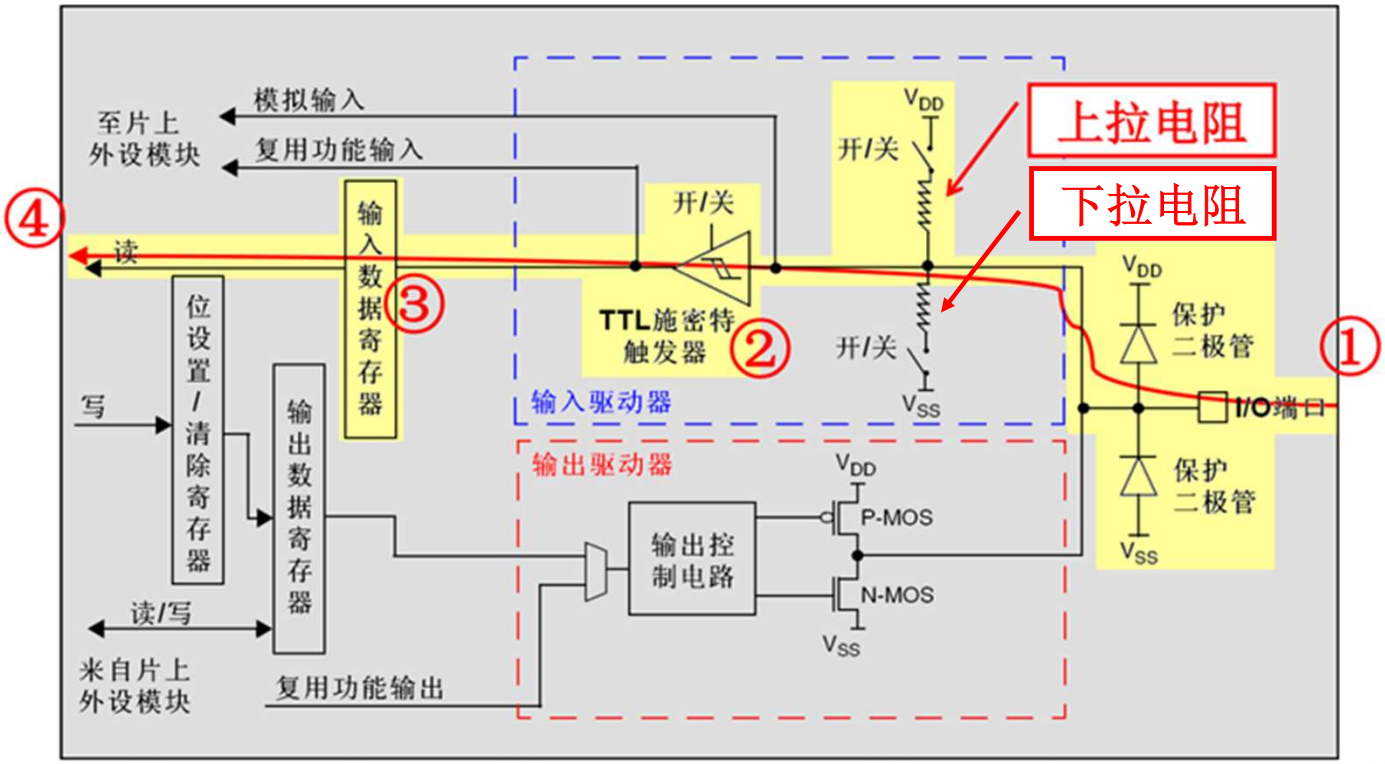
当$V_{io} < V_{ss} - V_d$，D1截止，D2导通，V_{io}被限制在$V_{ss} - V_d$；

当$V_{ss} - V_d \le V_{io} \le V_{DD} + V_d$，D1、D2均截止，$V_{io}$就是当前值。

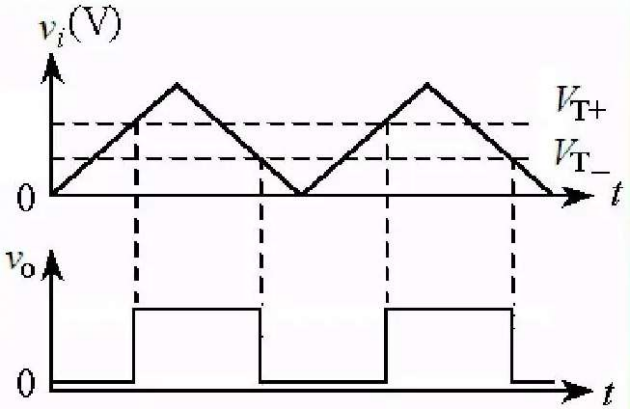
因此，无论V_{io}的电压过高还是过低，由于钳位二极管的作用，其值始终在$V_{ss} - V_d$和$V_{DD} + V_d$之间。

但是，当V_{io}值太高或太低时，会导致二极管过流，烧坏器件。实际使用时仍然要确保V_{io}电压值在合适的范围内。

GPIO输入模式——上拉输入与下拉输入



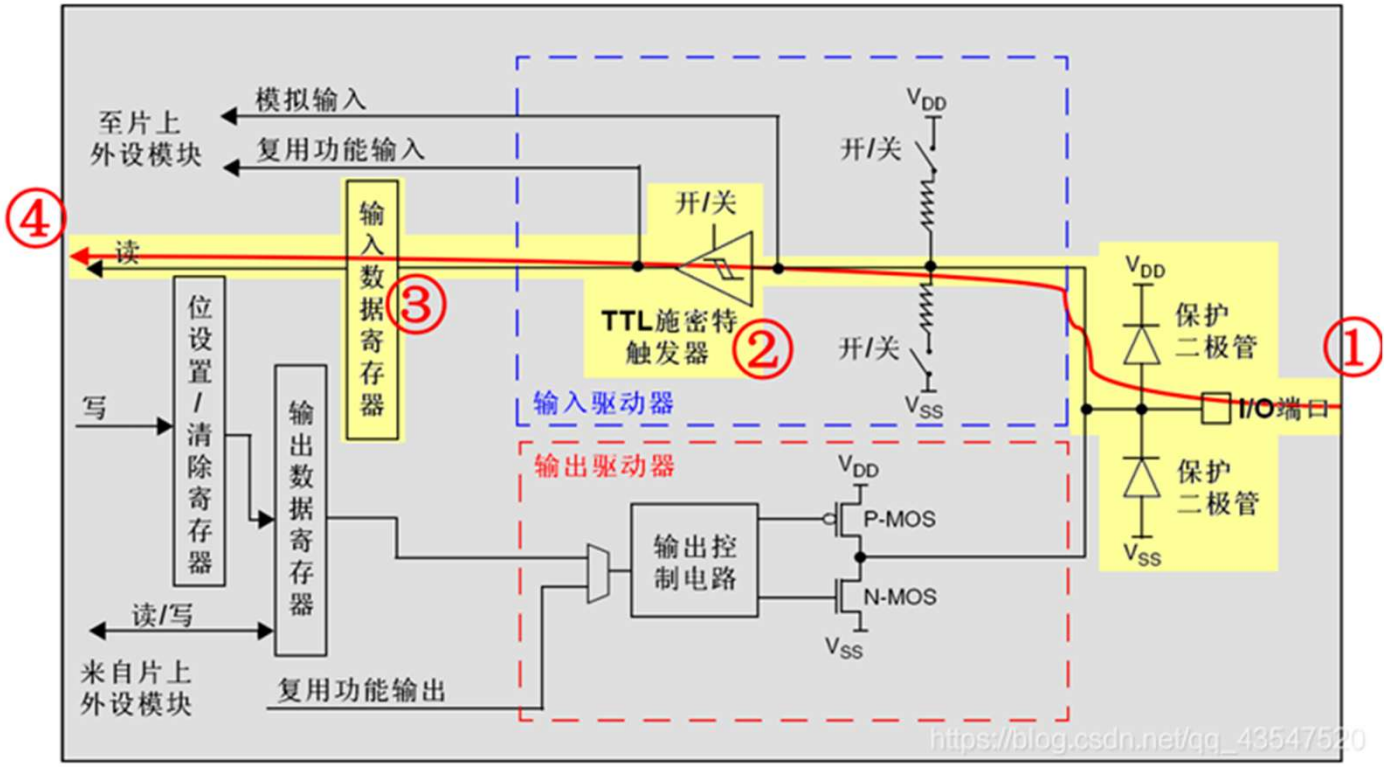
施密特触发器



当V_i大于V_{T+}时，V_o输出为高电平；
当V_i小于V_{T-}时，V_o输出为低电平。

目的：将IO口输入的电压转化为规则的数字信号，有效防止中间电压状态及电压的波动对IO电平读取的影响。

GPIO输入模式——浮空输入

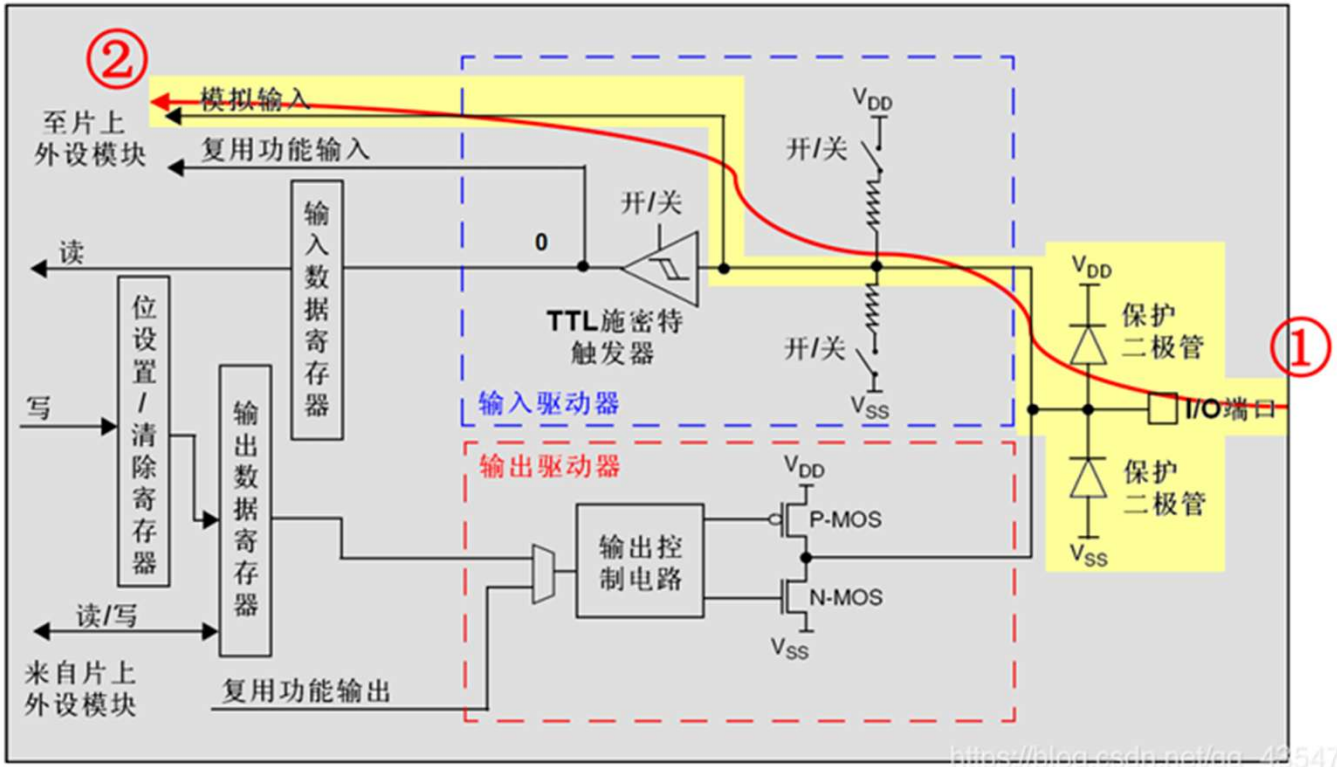


浮空输入

无上下拉电阻；

IO口的电平状态完全由外部输入决定。

GPIO输入模式——模拟输入



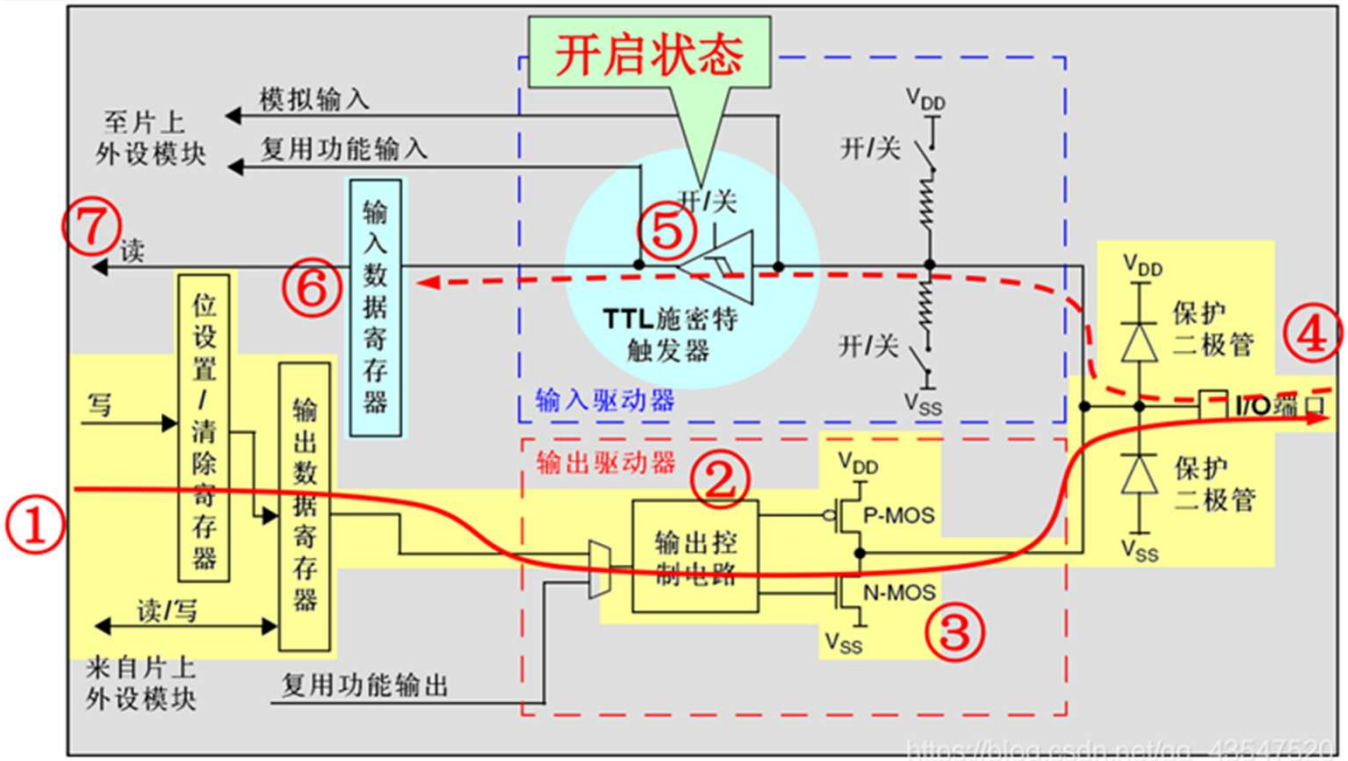
模拟输入

无上下拉电阻，电信号不经过施密特触发器，直接进入模拟输入通道，一般用于ADC数据采集。

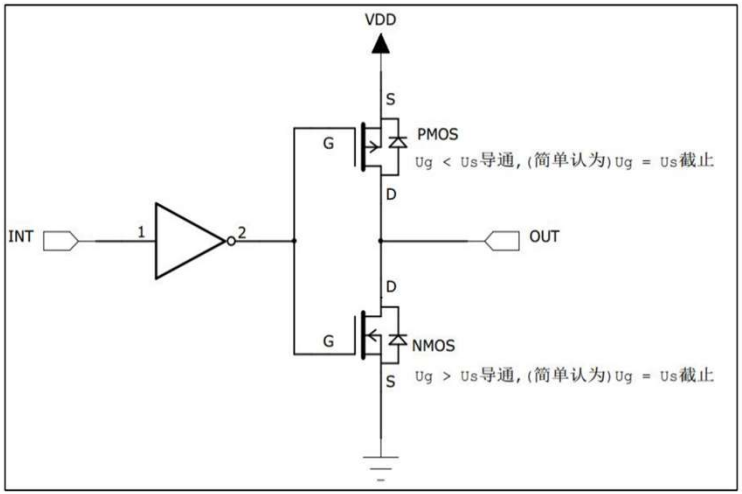
GPIO输入模式

输入模式	特点
上拉输入	应用场景：按键检测 优点：不易受到干扰 缺点：有一定的驱动能力，会增大芯片内部功耗
下拉输入	应用场景：按键检测、低功耗应用 优点：静态功耗低 缺点：驱动能力弱，影响高速应用
浮空输入	应用场景：传感器采集、专用外设接口 优点：无额外负载，采集高速信号更稳定 缺点：可靠性低，必须确保任何时候都有明确的驱动源
模拟输入	应用场景：ADC采集 优点：能够捕捉信号的细微变化，提供高分辨率数据 缺点：抗干扰能力弱，依赖外部电路设计

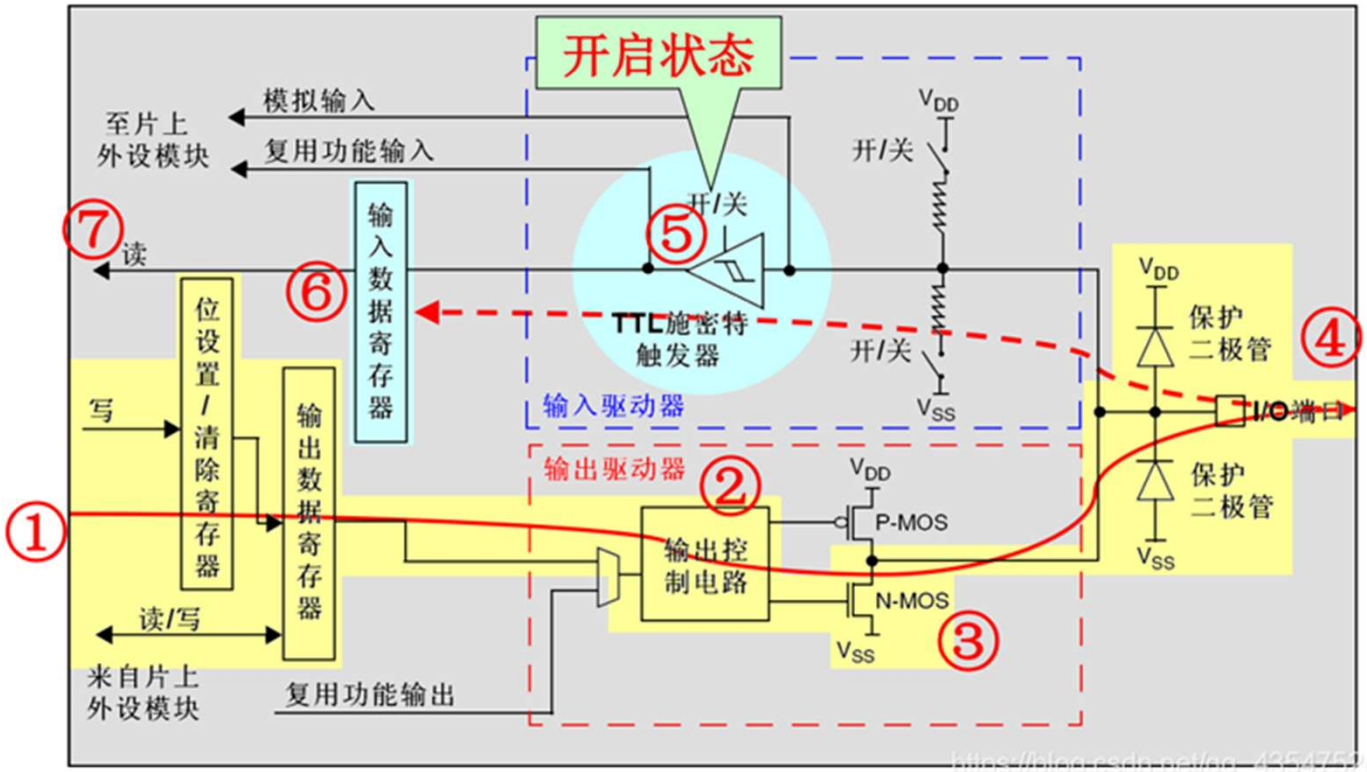
GPIO输出模式——推挽输出



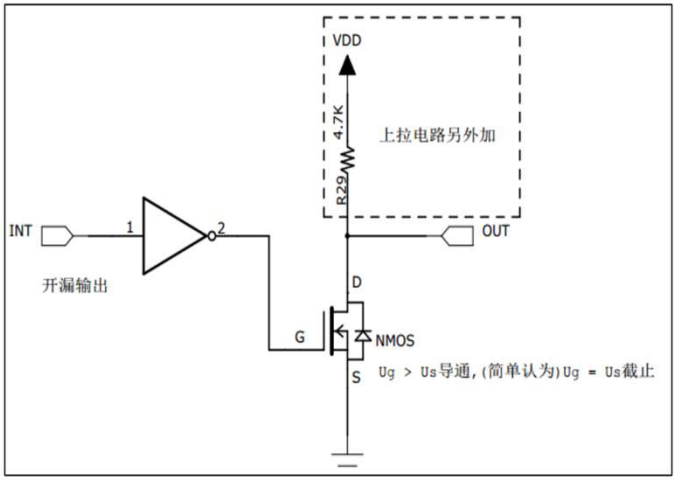
可输出高低电平
可读取引脚电平



GPIO输出模式——开漏输出



可输出低电平
需外接上拉电阻，才能输出高电平
可读取引脚电平



GPIO输出模式

输出模式	特点
推挽输出	应用场景：外设控制、通信接口 优点：驱动能力强、抗干扰能力强 缺点：会增大芯片内部功耗、电压域限制
开漏输出	应用场景：IIC、CAN总线 优点：总线兼容性高，不受电压域限制 缺点：驱动能力弱，依赖外部硬件设计

02 GPIO的应用

Part Two

应用场景、LED灯开关控制实验

Build a smarter world

GPIO应用场景

S

开关控制

如LED控制、电机驱动、蜂鸣器驱动等
（推挽、开漏输出）

E

外部中断

如按键检测、外设中断检测（如网卡）等
（上拉、下拉输入）

A

AD采样

ADC的模拟量输入检测
（模拟输入）

B

总线接口

各类通信总线的物理引脚，如UART、IIC、SPI等
（浮空输入、推挽输出）

GPIO实验——LED灯开关控制

驱动原理：

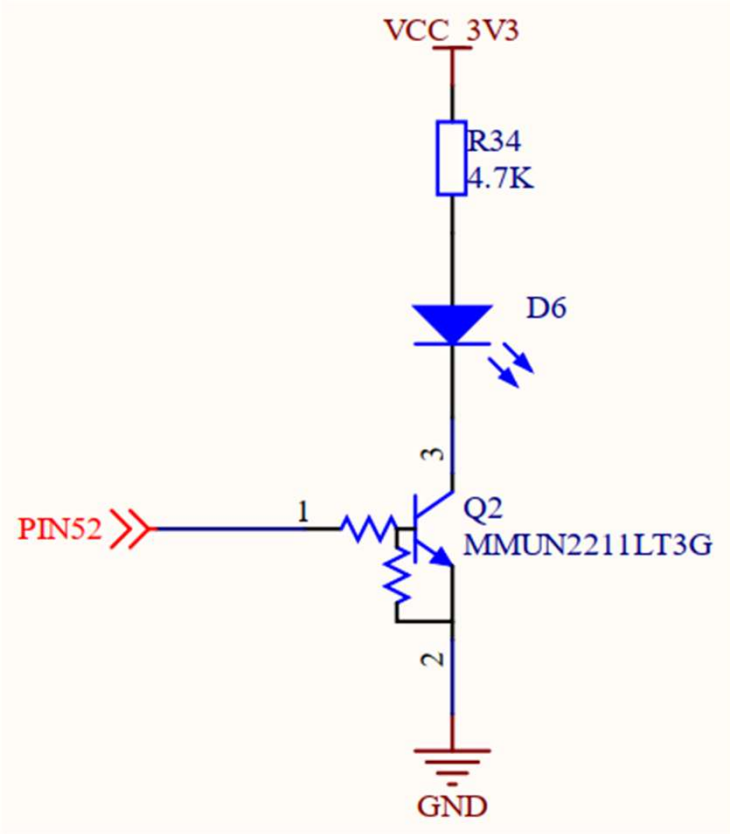
1. 使用Pin52引脚驱动

2. 当Pin52输出高电平时，NPN三极管Q2导通，Q2的集电极为低电平，发光二极管D6变亮

3. 当Pin52输出低电平时，NPN三极管Q2截止，无法通过电流，发光二极管D6熄灭

GPIO操作API：

[class Pin - 控制I/O引脚 - QuecPython]



GPIO实验——LED灯开关控制

软件设计：

1. **用户接口定义**

`LED(pin)`

- 功能：创建LED对象
- 返回：LED对象
- LED：LED类
- pin：控制LED的gpio引脚编号

`LED.on()`

- 功能：打开LED灯

`LED.off()`

- 功能：关闭LED灯

`LED.toggle()`

- 功能：翻转LED灯状态

```python
from machine import Pin

class LED():
    def __init__(self, gpio):
        self.gpio = Pin(gpio, Pin.OUT, Pin.PULL_DISABLE, 0)

    def on(self):
        self.gpio.write(1)

    def off(self):
        self.gpio.write(0)

    def toggle(self):
        if self.gpio.read() == 1:
            self.gpio.write(0)
        else:
            self.gpio.write(1)

if __name__ == "__main__":
    from utime import sleep
    led = LED(Pin.GPIO28)
    while True:
        led.toggle()
        sleep(1)
```

03 外部中断的工作原理

Part Three

外部中断触发条件

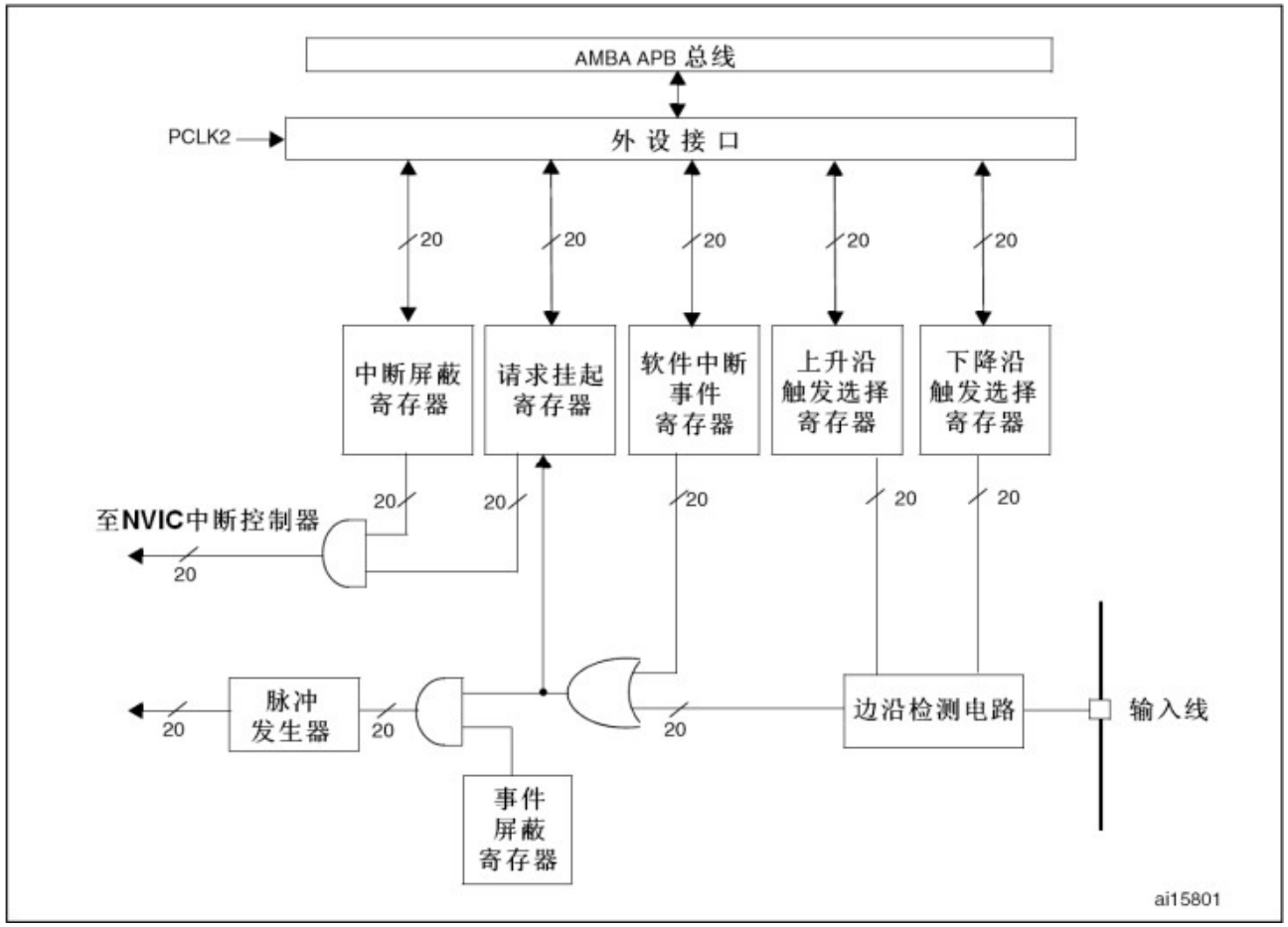
Build a smarter world

外部中断框图

外部中断：
External Interrupt

触发条件：
上升沿触发
下降沿触发
双边沿触发

输入模式：
- 上拉输入
- 下拉输入



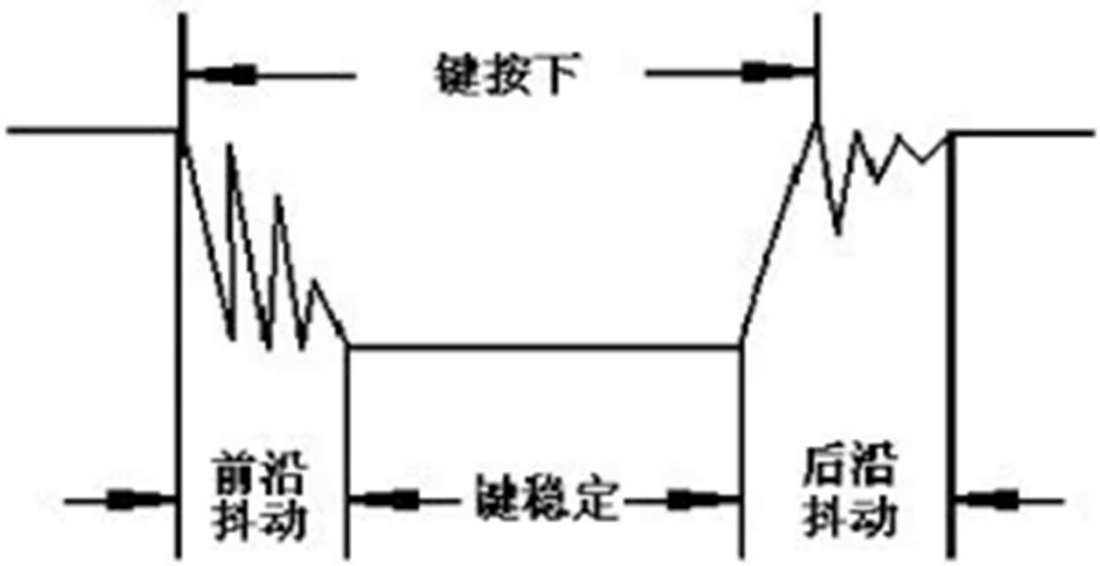
外部中断消抖

抖动产生：

按键的抖动对于人类来说是感觉不到的，但对单片机来说，则是完全可以感应到的，而且还是一个很“漫长”的过程，抖动时间的长短由按键的机械特性决定，一般为5ms ~ 10ms。这是一个很重要的时间参数，在很多场合都要用到。

消抖：

在检测到有按键按下时，不是立即认定此键已被按下，而是执行一个10ms左右（具体时间应视所使用的按键进行调整）的延时程序后，再确认该键电平是否仍然保持闭合状态电平，若仍然保持，则确认该键真正被按下。



04 外部中断的应用

Part Four

按键扫描

Build a smarter world

外部中断应用场景

M

机械开关事件通知

如独立按键和矩阵键盘扫描

S

传感器事件通知

如三轴加速度传感器、某些照度传感器等

N

网卡事件通知

如DM9051、w5500网卡等

B

存储器读写事件通知

外接Flash、RAM芯片的读写等

外部中断实验——按键扫描

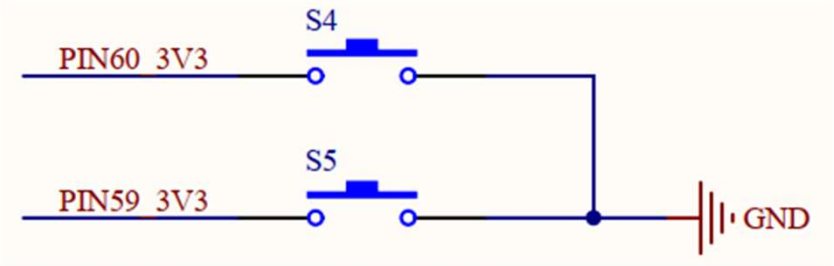
驱动原理：

1. 两个按键一端分别接入Pin59和Pin60两个GPIO，另一端均接地

2. 按键按下时，IO口电平为低电平，因此需要将GPIO设置为上拉输入模式，当按键不被按下时，读取的状态为高电平

3. 如果需要检测按下的事件，需要配置为下降沿触发；如果需要检测松开的事件，需要配置为上升沿触发；如果二者均需要检测，配置为双边沿触发

4. 按键按下和松开时，会产生机械抖动，需要做消抖处理



外部中断操作API：

class ExtInt - 外部中断 - QuecPython

外部中断实验——按键扫描

软件设计：

1. **用户接口定义**

```
Key(pin, level_on_pressed, cared_event, event_cb)
```

- Key：按键类
- pin：按键的gpio引脚编号
- level_on_pressed：按键按下时的电平，0或1。
- cared_event：关心的按键事件，包括按下和释放.
- event_cb：按键事件回调函数，所有的按键事件通过该cb通知用户。

要求：

✓ 软件接口能兼容不同的独立按键检测电路。

✓ 用户只需创建按键对象，并实现事件回调函数即可。

2. **设计思路**

✓ 通过外部中断检测按键是否按下。

✓ 当电路设计为按键按下是低电平时，外部中断设置为上拉输入，此时：

- 若用户关心按键按下的事件，设置外部中断的触发模式为下降沿触发；
- 若用户关心按键释放的事件，设置外部中断的触发模式为上升沿触发；
- 若用户同时关心按键按下和释放的事件，设置外部中断的触发模式为双边沿触发。

✓ 当电路设计为按键按下是高电平时，外部中断设置为下拉输入，此时：

- 若用户关心按键按下的事件，设置外部中断的触发模式为上升沿触发；
- 若用户关心按键释放的事件，设置外部中断的触发模式为下降沿触发；
- 若用户同时关心按键按下和释放的事件，设置外部中断的触发模式为双边沿触发。

✓ 在外部中断的回调函数中做消抖处理，并根据用户关心的按键事件和当前按键状态，将用户关心并且已发生的事件发送到消息队列。

✓ 在一个专门的线程中，等待队列中的消息，有消息到来，则调用用户注册的事件回调函数，将事件通知给上层应用。

外部中断实验——按键扫描

```
from machine import ExtInt
from machine import Pin
from utime import sleep_ms
from queue import Queue
import _thread

key_evt_queue = Queue(8)

def key_evt_thread_entry():···

_thread.start_new_thread(key_evt_thread_entry, ())

class Key():

    class Event():···

    class Error(Exception):···

    def __init__(self, pin, level_on_pressed, cared_event, event_cb):···

    def exit_cb(self, args):···
```

```
if __name__ == '__main__':
    k1 = Pin.GPIO13
    k2 = Pin.GPIO12

    def event_cb(pin, event):
        if pin == k1:
            if event == Key.Event.PRESSED:
                print("k1 pressed")
            else:
                print("k1 released")
        if pin == k2:
            if event == Key.Event.PRESSED:
                print("k2 pressed")
            else:
                print("k2 released")

    Key(k1, 0, Key.Event.PRESSED | Key.Event.RELEASED, event_cb)
    Key(k2, 0, Key.Event.PRESSED | Key.Event.RELEASED, event_cb)
```

课程总结

1. GPIO的输入输出模式：上拉、下拉、浮空、模拟、推挽、开漏
2. LED灯的驱动原理和代码实现
3. 外部中断的触发方式：上升沿、下降沿、双边沿
4. 按键扫描的驱动原理和代码实现

Thank you

获取更多产品信息, 请访问: www.quectel.com.cn
全国热线: 400 960 7678

移远全球中心
上海市松江区泗泾镇外婆泾路8号
联系电话：021 5108 6236　销售支持：sales@quectel.com
技术支持：support@quectel.com　电子邮件：info@quectel.com

QUECTEL

移远微信公众号