

优秀不够，你是否无可替代

知识从未如此性感。烂程序员关心的是代码,好程序员关心的是数据结构和它们之间的关系 --QQ群: 607064330 --本人
QQ:946029359 --淘宝 <https://shop411638453.taobao.com/>

随笔 - 743, 文章 - 0, 评论 - 315, 阅读 - 181万

导航

博客园
首页
新随笔
联系
订阅 
管理

公告



昵称：杨奉武
园龄：5年9个月
粉丝：629
关注：1

搜索

我的标签

8266(88)
MQTT(50)
GPRS(33)
SDK(29)
Air202(28)
云服务器(21)
ESP8266(21)
Lua(18)
小程序(17)
STM32(16)
更多

随笔分类

Android(22)
Android 开发(8)
C# 开发(4)
CH395Q学习开发(17)
CH579M学习开发(7)
ESP32学习开发(10)
ESP8266 AT指令开发(基于
STC89C52单片机)(3)
ESP8266 AT指令开发(基于
STM32)(1)
ESP8266 AT指令开发基础入
门篇备份(12)
ESP8266 LUA脚本语言开发
(13)
ESP8266 LUA开发基础入门篇
备份(22)

102-ESP32学习开发(SDK)-GPIO

<p><iframe name="ifd" src="https://mnifdv.cn/resource/cnblogs/LearnESP32" frameborder="0" scrolling="auto" width="100%" height="1500"></iframe></p>

开源ESP32开发(源码见资料源码)

测试板链接:[ESP32测试板链接](#)

资料源码:<https://github.com/yangfengwu45/learn-esp32.git>

【点击加入乐鑫WiFi模组开发交流群】(群号
822685419)https://jq.qq.com/?_wv=1027&k=fXgd3UOo

python虚拟机: [python-3.8.4-amd64.exe](#)

ESP-IDF工具安装器: [esp-idf-tools-setup-2.3.exe](#)

- [基础开源教程:ESP32开发\(arduino\)](#)
- [基础开源教程:ESP8266:LUA脚本开发](#)
- [基础开源教程:ESP8266 AT指令开发\(基于51单片机\)](#)
- [基础开源教程:Android学习开发](#)
- [基础开源教程:C#学习开发](#)
- [基础开源教程:微信小程序开发入门篇](#)
需要搭配的Android, C#等基础教程如上, 各个教程正在整理。

- [000-ESP32开发板使用说明](#)
- [ESP32_SDK开发](#)
- [001-开发环境搭建\(Windows+VSCode\)](#)
- [002-测试网络摄像头\(OV2640\),实现远程视频监控\(花生壳http映射\)](#)
- [003-学习ESP32资料说明](#)
- [004-新建工程模板和创建新的文件](#)
- [005-ESP32学习开发\(SDK\)-新建工程补充-通过官方示例创建工程](#)
- -----基本外设-----
- [101-ESP32学习开发\(SDK\)-ESP32管脚说明](#)

ESP8266 SDK开发(32)
ESP8266 SDK开发基础入门篇
备份(30)
GPRS Air202 LUA开发(11)
HC32F460(华大) +
BC260Y(NB-IOT) 物联网开发
(5)
NB-IOT Air302 AT指令和LUA
脚本语言开发(25)
PLC(三菱PLC)基础入门篇(2)
STM32+Air724UG(4G模组)
物联网开发(43)
STM32+BC26/260Y物联网开
发(37)
STM32+CH395Q(以太网)物
联网开发(21)
STM32+ESP8266(ZLESP8266/
物联网开发(1)
STM32+ESP8266+AIR202/30:
远程升级方案(16)
STM32+ESP8266+AIR202/30:
终端管理方案(6)
STM32+ESP8266+Air302物
联网开发(64)
STM32+W5500+AIR202/302
基本控制方案(25)
STM32+W5500+AIR202/302
远程升级方案(6)
UCOSii操作系统(1)
W5500 学习开发(8)
编程语言C#(11)
编程语言Lua脚本语言基础入
门篇(6)
编程语言Python(1)
单片机(LPC1778)LPC1778(2)
单片机(MSP430)开发基础入门
篇(4)
单片机(STC89C51)单片机开发
板学习入门篇(3)
单片机(STM32)基础入门篇(3)
单片机(STM32)综合应用系列
(16)
电路模块使用说明(11)
感想(6)
软件安装使用: MQTT(8)
软件安装使用: OpenResty(6)
更多

最新评论

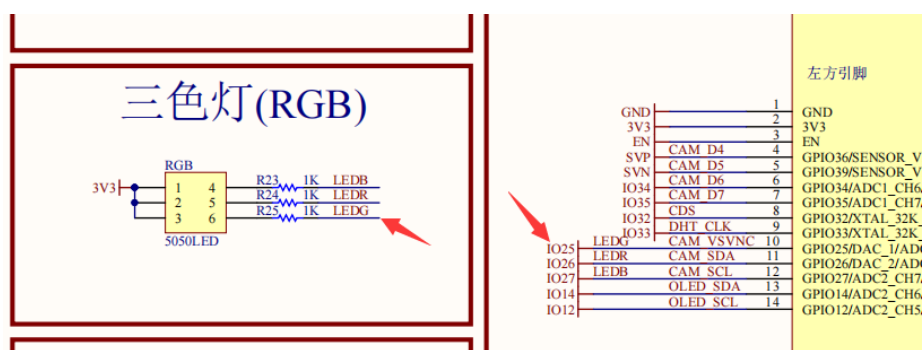
1. Re:单片机模块化程序: 看看是不是你想要的按键处理视频不见了
--伊森亨特
2. Re:C#开发: 通信篇-TCP客户端
感谢分享，直接就用上了
--Zfen

阅读排行榜

1. ESP8266使用详解(AT,LUA, SDK)(172693)
2. 1-安装MQTT服务器(Windows),并连接测试(98591)
3. ESP8266刷AT固件与node mcu固件(64580)
4. 用ESP8266+android,制作自己的WIFI小车(ESP8266篇)(64060)
5. 有人WIFI模块使用详解(38474)

控制GPIO25输出高低电平

1.原理图



2.参考官方例程

LearnESP32 > esp-if > examples > peripherals >		
名称	↑	消
adc		2
adc2		2
gpio		2
i2c		2
i2s		2
i2s_adc_dac		2
ledc		2
mcpwm		2
oled		2
.		~

6. (一)基于阿里云的MQTT远程控制(Android 连接MQTT服务器,ESP8266连接MQTT服务器实现远程通信控制----简单的连接通信)(35888)
7. 关于TCP和MQTT之间的转换(33124)
8. C#中public与private与static(32280)
9. android 之TCP客户端编程(31854)
10. android服务端+eps8266+单片机+路由器之远程控制系统(31298)

推荐排行榜

1. C#委托+回调详解(9)
2. 用ESP8266+android,制作自己的WIFI小车(ESP8266篇)(8)
3. 用ESP8266+android,制作自己的WIFI小车(Android 软件)(6)
4. ESP8266使用详解(AT,LUA,SDK)(6)
5. 关于TCP和MQTT之间的转换(5)

3.程序

```
main > C gpio_example_main.c > ...
1
2  #include <stdio.h>
3  #include <string.h>
4  #include <stdlib.h>
5  #include "freertos/FreeRTOS.h"
6  #include "freertos/task.h"
7  #include "freertos/queue.h"
8  #include "driver/gpio.h"
9
10
11 #define gpio_pin 25
12
13 void app_main(void)
14 {
15     //gpio配置结构体
16     gpio_config_t io_conf;
17     //禁止中断
18     io_conf.intr_type = GPIO_PIN_INTR_DISABLE;
19     //输出模式
20     io_conf.mode = GPIO_MODE_OUTPUT;
21     //配置要设置的引脚
22     io_conf.pin_bit_mask = (unsigned long long)1<<gpio_pin;
23     //禁止下拉
24     io_conf.pull_down_en = 0;
25     //禁止上拉
26     io_conf.pull_up_en = 0;
27     //配置gpio(不设置上下拉默认输出低电平)
28     gpio_config(&io_conf);
29
30     while(1) {
31         gpio_set_level(gpio_pin, 0); //设置引脚输出低电平
32         vTaskDelay(3000 / portTICK_RATE_MS); //延时约3S
33         gpio_set_level(gpio_pin, 1); //设置引脚输出高电平
34         vTaskDelay(3000 / portTICK_RATE_MS); //延时约3S
35     }
36 }
37
```



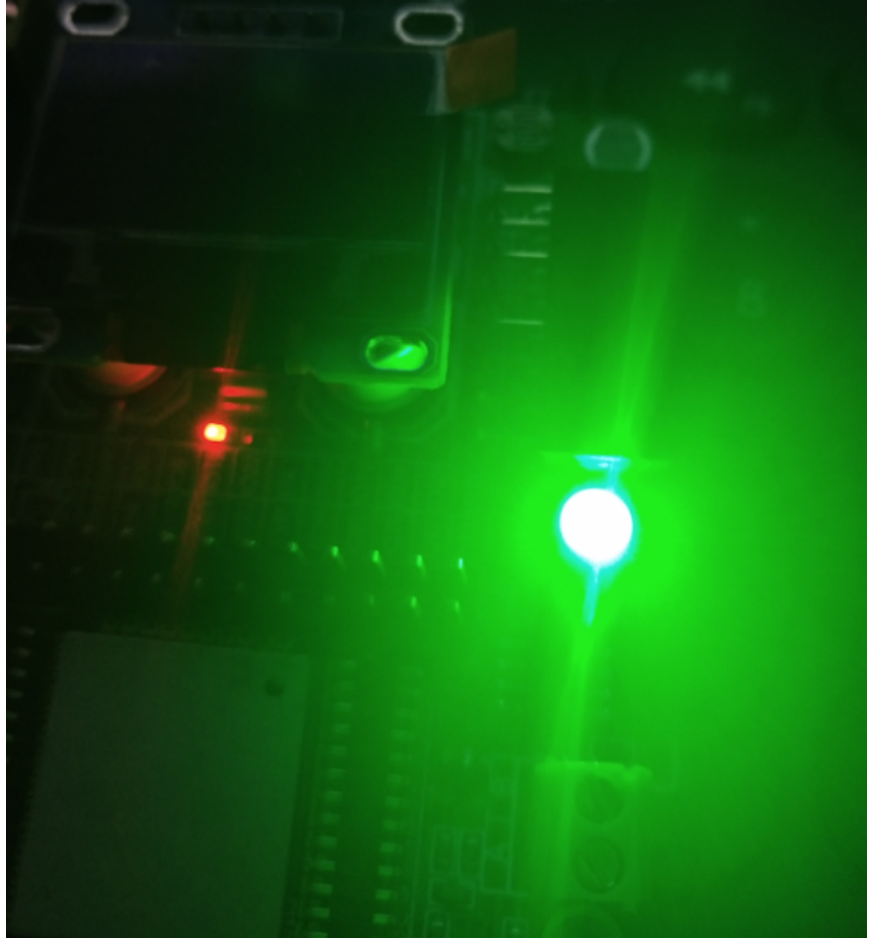
```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include "freertos/FreeRTOS.h"
#include "freertos/task.h"
#include "freertos/queue.h"
#include "driver/gpio.h"

#define gpio_pin 25

void app_main(void)
{
    //gpio配置结构体
    gpio_config_t io_conf;
    //禁止中断
    io_conf.intr_type = GPIO_PIN_INTR_DISABLE;
    //输出模式
    io_conf.mode = GPIO_MODE_OUTPUT;
    //配置要设置的引脚
    io_conf.pin_bit_mask = (unsigned long long)1<<gpio_pin;
    //禁止下拉
    io_conf.pull_down_en = 0;
    //禁止上拉
    io_conf.pull_up_en = 0;
```

```
//配置gpio(不设置上下拉默认输出低电平)
gpio_config(&io_conf);

while(1) {
    gpio_set_level(gpio_pin, 0); //设置引脚输出低电平
    vTaskDelay(3000 / portTICK_RATE_MS); //延时约3S
    gpio_set_level(gpio_pin, 1); //设置引脚输出高电平
    vTaskDelay(3000 / portTICK_RATE_MS); //延时约3S
}
```



控制GPIO25 和 GPIO26 输出高低电平

```
C gpio_example_main.c U C sdkconfig.h U C gpio_types.h
main > C gpio_example_main.c > ...
6 #include "freertos/task.h"
7 #include "freertos/queue.h"
8 #include "driver/gpio.h"
9
10
11 #define gpio_pin 25
12 #define gpio_pin1 26
13
14 void app_main(void)
15 {
16     //gpio配置结构体
17     gpio_config_t io_conf;
18     //禁止中断
19     io_conf.intr_type = GPIO_PIN_INTR_DISABLE;
20     //输出模式
21     io_conf.mode = GPIO_MODE_OUTPUT;
22     //配置要设置的引脚
23     io_conf.pin_bit_mask = (((unsigned long long)1<<gpio_pin) | ((unsigned long long)1<<gpio_pin1));
24     //禁止下拉
25     io_conf.pull_down_en = 0;
26     //禁止上拉
27     io_conf.pull_up_en = 0;
28     //配置gpio(不设置上下拉默认输出低电平)
29     gpio_config(&io_conf);
30
31     while(1) {
32         gpio_set_level(gpio_pin, 0); //设置引脚输出低电平
33         gpio_set_level(gpio_pin1, 0); //设置引脚输出低电平
34
35         vTaskDelay(3000 / portTICK_RATE_MS); //延时约3s
36
37         gpio_set_level(gpio_pin, 1); //设置引脚输出高电平
38         gpio_set_level(gpio_pin1, 1); //设置引脚输出高电平
39
40         vTaskDelay(3000 / portTICK_RATE_MS); //延时约3s
41     }
42 }
43
```



```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include "freertos/FreeRTOS.h"
#include "freertos/task.h"
#include "freertos/queue.h"
#include "driver/gpio.h"

#define gpio_pin 25
#define gpio_pin1 26

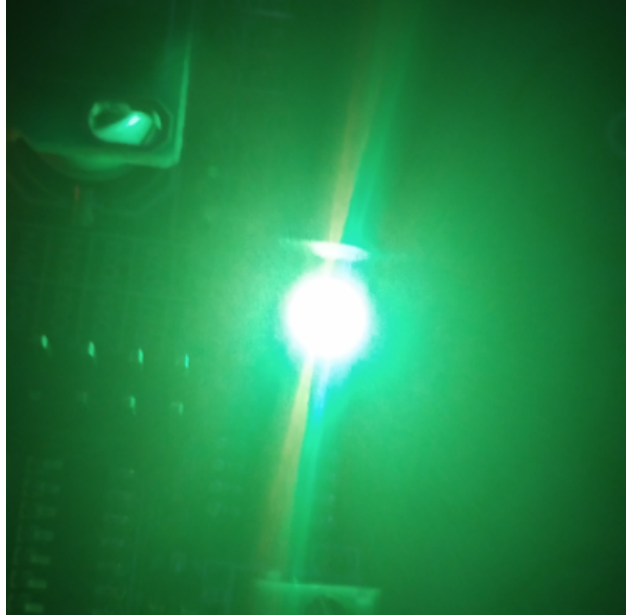
void app_main(void)
{
    //gpio配置结构体
    gpio_config_t io_conf;
    //禁止中断
    io_conf.intr_type = GPIO_PIN_INTR_DISABLE;
    //输出模式
    io_conf.mode = GPIO_MODE_OUTPUT;
    //配置要设置的引脚
    io_conf.pin_bit_mask = (((unsigned long long)1<<gpio_pin) | ((unsigned lo
    //禁止下拉
    io_conf.pull_down_en = 0;
    //禁止上拉
    io_conf.pull_up_en = 0;
    //配置gpio(不设置上下拉默认输出低电平)
    gpio_config(&io_conf);

    while(1) {
        gpio_set_level(gpio_pin, 0); //设置引脚输出低电平
        gpio_set_level(gpio_pin1, 0); //设置引脚输出低电平
```

```
vTaskDelay(3000 / portTICK_RATE_MS); //延时约3S

gpio_set_level(gpio_pin, 1); //设置引脚输出高电平
gpio_set_level(gpio_pin1, 1); //设置引脚输出高电平

vTaskDelay(3000 / portTICK_RATE_MS); //延时约3S
}
}
```



补充:

配置gpio还有一个参数 driver

GPIO_DRIVE_CAP_0 弱 weak

GPIO_DRIVE_CAP_1 强

GPIO_DRIVE_CAP_2 默认值

GPIO_DRIVE_CAP_DEFAULT 默认值

GPIO_DRIVE_CAP_3 最强

```
io_conf.driver = GPIO_DRIVE_CAP_3;
```

提示

GPIO的模式

GPIO_MODE_INPUT 输入

GPIO_MODE_OUTPUT 输出

GPIO_MODE_OUTPUT_OD 开漏输出

GPIO_MODE_INPUT_OUTPUT_OD 开漏输入输出

GPIO_MODE_INPUT_OUTPUT 输入输出(如果想让模块即做输入检测又做输出控制,需要设置这个模式)

```
92     GPIO_INTR_MAX,  
93 } gpio_int_type_t;  
94  
95 typedef enum {  
96     GPIO_MODE_DISABLE = GPIO_MODE_DEF_DISABLE,  
97     GPIO_MODE_INPUT = GPIO_MODE_DEF_INPUT,  
98     GPIO_MODE_OUTPUT = GPIO_MODE_DEF_OUTPUT,  
99     GPIO_MODE_OUTPUT_OD = ((GPIO_MODE_DEF_OUTPUT) | (GPIO_MODE_DEF_OD)),  
100     GPIO_MODE_INPUT_OUTPUT_OD = ((GPIO_MODE_DEF_INPUT) | (GPIO_MODE_DEF_OUTPUT) | (GPIO_MODE_DEF_OD)),  
101     GPIO_MODE_INPUT_OUTPUT = ((GPIO_MODE_DEF_INPUT) | (GPIO_MODE_DEF_OUTPUT)),  
102 } gpio_mode_t;  
103  
104 typedef enum {  
105     GPIO_PULLUP_DISABLE = 0x0, /* Disable GPIO pull-up resistor */
```

配置GPIO0作为输入输出模式,检测引脚输出状态

```
C gpio_example_main.c U x C gpio.c C gpio_types.h  
main > C gpio_example_main.c > app_main(void)  
1  
2 #include <stdio.h>  
3 #include <string.h>  
4 #include <stdlib.h>  
5 #include "freertos/FreeRTOS.h"  
6 #include "freertos/task.h"  
7 #include "freertos/queue.h"  
8 #include "driver/gpio.h"  
9  
10  
11 #define gpio_pin 0  
12  
13 void app_main(void)  
14 {  
15     //gpio配置结构体  
16     gpio_config_t io_conf;  
17     //禁止中断  
18     io_conf.intr_type = GPIO_PIN_INTR_DISABLE;  
19     //输入输出模式  
20     io_conf.mode = GPIO_MODE_INPUT_OUTPUT;  
21     //配置要设置的引脚  
22     io_conf.pin_bit_mask = (unsigned long long)1<<gpio_pin;  
23     //禁止下拉  
24     io_conf.pull_down_en = 0;  
25     //禁止上拉  
26     io_conf.pull_up_en = 0;  
27     //配置gpio(不设置上下拉默认输出低电平)  
28     gpio_config(&io_conf);  
29  
30     while(1) {  
31         gpio_set_level(gpio_pin, 0); //设置引脚输出低电平  
32  
33         printf("获取引脚状态=%d\r\n", gpio_get_level(gpio_pin));  
34  
35         vTaskDelay(3000 / portTICK_RATE_MS); //延时约3s  
36  
37         gpio_set_level(gpio_pin, 1); //设置引脚输出高电平  
38         printf("获取引脚状态=%d\r\n", gpio_get_level(gpio_pin));  
39  
40         vTaskDelay(3000 / portTICK_RATE_MS); //延时约3s  
41     }  
42 }  
43  
44
```



```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include "freertos/FreeRTOS.h"
#include "freertos/task.h"
#include "freertos/queue.h"
#include "driver/gpio.h"

#define gpio_pin 0

void app_main(void)
{
    //gpio配置结构体
    gpio_config_t io_conf;
    //禁止中断
    io_conf.intr_type = GPIO_PIN_INTR_DISABLE;
    //输入输出模式
    io_conf.mode = GPIO_MODE_INPUT_OUTPUT;
    //配置要设置的引脚
    io_conf.pin_bit_mask = (unsigned long long)1<<gpio_pin;
    //禁止下拉
    io_conf.pull_down_en = 0;
    //禁止上拉
    io_conf.pull_up_en = 0;
    //配置gpio(不设置上下拉默认输出低电平)
    gpio_config(&io_conf);

    while(1) {
        gpio_set_level(gpio_pin, 0); //设置引脚输出低电平

        printf("获取引脚状态=%d\r\n", gpio_get_level(gpio_pin));

        vTaskDelay(3000 / portTICK_RATE_MS); //延时约3s

        gpio_set_level(gpio_pin, 1); //设置引脚输出高电平
        printf("获取引脚状态=%d\r\n", gpio_get_level(gpio_pin));

        vTaskDelay(3000 / portTICK_RATE_MS); //延时约3s
    }
}
```




```
获取引脚状态=0  
获取引脚状态=1  
获取引脚状态=0  
获取引脚状态=1  
获取引脚状态=0  
获取引脚状态=1  
获取引脚状态=0  
获取引脚状态=1  
获取引脚状态=0  
获取引脚状态=1  
获取引脚状态=0  
获取引脚状态=1  
获取引脚状态=0  
获取引脚状态=1
```

配置GPIO0下降沿中断

1,中断类型

GPIO_INTR_DISABLE 禁用GPIO中断

GPIO_INTR_POSEDGE GPIO中断类型：上升沿

GPIO_INTR_NEGEDGE 下降沿

GPIO_INTR_ANYEDGE 上升沿和下降沿

GPIO_INTR_LOW_LEVEL 输入低电平触发

GPIO_INTR_HIGH_LEVEL 输入高电平触发

2,程序

```

C gpio_example_main.c U X C esp_intr_alloc.h H\...\esp32\include C intr_alloc.c C esp_intr_alloc.h H
main > C gpio_example_main.c > app_main(void)
2  #include <stdio.h>
3  #include <string.h>
4  #include <stdlib.h>
5  #include "freertos/FreeRTOS.h"
6  #include "freertos/task.h"
7  #include "freertos/queue.h"
8  #include "driver/gpio.h"
9
10
11 #define gpio_pin 0
12
13 static xQueueHandle gpio_evt_queue = NULL;
14
15 /*gpio中断回调函数*/
16 static void IRAM_ATTR gpio_isr_handler(void* arg)
17 {
18     uint32_t gpio_num = (uint32_t) arg;
19     //把消息存储到消息队列
20     xQueueSendFromISR(gpio_evt_queue, &gpio_num, NULL);
21 }
22
23 /*任务函数*/
24 static void gpio_task_example(void* arg)
25 {
26     uint32_t io_num;
27     for(;;) {
28         /*消息队列里面有消息*/
29         if(xQueueReceive(gpio_evt_queue, &io_num, portMAX_DELAY)) {
30             printf("GPIO[%d] intr, val: %d\n", io_num, gpio_get_level(io_num));
31         }
32     }
33 }
34
35
36 void app_main(void)
37 {
38     //gpio配置结构体
39     gpio_config_t io_conf;
40     //下降沿中断
41     io_conf.intr_type = GPIO_INTR_NEGEDGE;
42     //输入模式
43     io_conf.mode = GPIO_MODE_INPUT;
44     //配置要设置的引脚
45     io_conf.pin_bit_mask = (unsigned long long)1<<gpio_pin;

```

```

C gpio_example_main.c U x C esp_intr_alloc.h HA...\esp32\include C intr_alloc.c C esp_intr_alloc.h HA...\esp32s2\... C
main > C gpio_example_main.c > app_main(void)
33
34
35
36 void app_main(void)
37 {
38     //gpio配置结构体
39     gpio_config_t io_conf;
40     //下降沿中断
41     io_conf.intr_type = GPIO_INTR_NEGEDGE;
42     //输入模式
43     io_conf.mode = GPIO_MODE_INPUT;
44     //配置要设置的引脚
45     io_conf.pin_bit_mask = (unsigned long long)1<<gpio_pin;
46     //禁止下拉
47     io_conf.pull_down_en = 0;
48     //上拉
49     io_conf.pull_up_en = 1;
50     //配置gpio
51     gpio_config(&io_conf);
52
53
54     /*中断里面不能写printf,写了会死机; 所以就任务+消息队列打印*/
55     //创建消息队列
56     gpio_evt_queue = xQueueCreate(10, sizeof(uint32_t));
57     //创建任务
58     xTaskCreate(gpio_task_example, "gpio_task_example", 2048, NULL, 10, NULL);
59
60
61     //设置中断优先级(1-8级),如果参数写0,则内部自动从1-3级中分配一个优先级; 7级优先级为最高;
62     //如果设置为8,则此中断是共享中断,即多个外设都可触发这个中断(处理起来应该会很麻烦,到时候真的使用到再说)
63     gpio_install_isr_service(ESP_INTR_FLAG_LEVEL1);
64     //移除中断
65     gpio_isr_handler_remove(gpio_pin);
66     //添加中断
67     gpio_isr_handler_add(gpio_pin, gpio_isr_handler, (void*) gpio_pin);
68
69
70     while(1) {
71         //必须加延时,任务不能没有延时,否则导致任务无法切换.
72         vTaskDelay(1000 / portTICK_RATE_MS);
73     }
74
75
76

```



```

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include "freertos/FreeRTOS.h"
#include "freertos/task.h"
#include "freertos/queue.h"
#include "driver/gpio.h"

#define gpio_pin 0

static xQueueHandle gpio_evt_queue = NULL;

/*gpio中断回调函数*/
static void IRAM_ATTR gpio_isr_handler(void* arg)
{
    uint32_t gpio_num = (uint32_t) arg;
    //把消息存储到消息队列
    xQueueSendFromISR(gpio_evt_queue, &gpio_num, NULL);
}

/*任务函数*/
static void gpio_task_example(void* arg)
{
    uint32_t io_num;
    for(;;) {
        /*消息队列里面有消息*/

```

```

        if(xQueueReceive(gpio_evt_queue, &io_num, portMAX_DELAY)) {
            printf("GPIO[%d] intr, val: %d\n", io_num, gpio_get_level(io_num))
        }
    }
}

void app_main(void)
{
    //gpio配置结构体
    gpio_config_t io_conf;
    //下降沿中断
    io_conf.intr_type = GPIO_INTR_NEGEDGE;
    //输入模式
    io_conf.mode = GPIO_MODE_INPUT;
    //配置要设置的引脚
    io_conf.pin_bit_mask = (unsigned long long)1<<gpio_pin;
    //禁止下拉
    io_conf.pull_down_en = 0;
    //上拉
    io_conf.pull_up_en = 1;
    //配置gpio
    gpio_config(&io_conf);

    /*中断里面不能写printf,写了会死机; 所以就用任务+消息队列打印*/
    //创建消息队列
    gpio_evt_queue = xQueueCreate(10, sizeof(uint32_t));
    //创建任务
    xTaskCreate(gpio_task_example, "gpio_task_example", 2048, NULL, 10, NULL)

    //设置中断优先级(1-8级),如果参数写0,则内部自动从1-3级中分配一个优先级; 7级优先级为最高
    //如果设置为8,则此中断是共享中断,即多个外设都可触发这个中断(处理起来应该会很麻烦,到时候
    gpio_install_isr_service(ESP_INTR_FLAG_LEVEL1);
    //移除中断
    gpio_isr_handler_remove(gpio_pin);
    //添加中断
    gpio_isr_handler_add(gpio_pin, gpio_isr_handler, (void*) gpio_pin);

    while(1) {
        //必须加延时,任务不能没有延时,否则导致任务无法切换.
        vTaskDelay(1000 / portTICK_RATE_MS);
    }
}

```

3,其实主要的就几句话

```

35
36 void app_main(void)
37 {
38     //gpio配置结构体
39     gpio_config_t io_conf;
40     //下降沿中断
41     io_conf.intr_type = GPIO_INTR_NEGEDGE;
42     //输入模式
43     io_conf.mode = GPIO_MODE_INPUT;
44     //配置要设置的引脚
45     io_conf.pin_bit_mask = (unsigned long long)1<<gpio_pin;
46     //禁止下拉
47     io_conf.pull_down_en = 0;
48     //上拉
49     io_conf.pull_up_en = 1;
50     //配置gpio
51     gpio_config(&io_conf);
52
53
54     /*中断里面不能写printf,写了会死机; 所以就任务+消息队列打印*/
55     //创建消息队列
56     gpio_evt_queue = xQueueCreate(10, sizeof(uint32_t));
57     //创建任务
58     xTaskCreate(gpio_task_example, "gpio_task_example", 2048, NULL, 10, NULL);
59
60
61     //设置中断优先级(1-8级),如果参数写0,则内部自动从1-3级中分配一个优先级; 7级优先级为最高;
62     //如果设置为8,则此中断是共享中断,即多个外设都可触发这个中断(处理起来应该会很麻烦,到时候真的使用到再说)
63     gpio_install_isr_service(ESP_INTR_FLAG_LEVEL1);
64     //移除中断
65     gpio_isr_handler_remove(gpio_pin);
66     //添加中断
67     gpio_isr_handler_add(gpio_pin, gpio_isr_handler, (void*) gpio_pin);
68
69

```

```

C gpio_example_main.c U X C esp_intr_alloc.h H\...\esp32\include C intr_alloc.c C es
main > C gpio_example_main.c > app_main(void)
4 #include <stdio.h>
5 #include "freertos/FreeRTOS.h"
6 #include "freertos/task.h"
7 #include "freertos/queue.h"
8 #include "driver/gpio.h"
9
10
11 #define gpio_pin 0
12
13 static xQueueHandle gpio_evt_queue = NULL;
14
15 /*gpio中断回调函数*/
16 static void IRAM_ATTR gpio_isr_handler(void* arg)
17 {
18     uint32_t gpio_num = (uint32_t) arg;
19     //把消息存到消息队列
20     xQueueSendFromISR(gpio_evt_queue, &gpio_num, NULL);
21 }
22
23 /*任务函数*/

```

4,动作 按键 (GPIO0)



```
I (0) gpio_start: Starting scheduler on pin 0
I (307) gpio: GPIO[0]| InputEn: 1| OutputEn: 0
GPIO[0] intr, val: 0
GPIO[0] intr, val: 1
GPIO[0] intr, val: 0
GPIO[0] intr, val: 0
GPIO[0] intr, val: 0
GPIO[0] intr, val: 0
GPIO[0] intr, val: 0
GPIO[0] intr, val: 0
GPIO[0] intr, val: 0
GPIO[0] intr, val: 0
GPIO[0] intr, val: 0
GPIO[0] intr, val: 0
GPIO[0] intr, val: 1
GPIO[0] intr, val: 0
GPIO[0] intr, val: 1
GPIO[0] intr, val: 0
```

5.提示

按理说下降沿中断,读取到的引脚电平只有0才对,但是呢之所有1,是因为

```
9
10
11 #define gpio_pin 0
12
13 static xQueueHandle gpio_evt_queue = NULL;
14
15 /*gpio中断回调函数*/
16 static void IRAM_ATTR gpio_isr_handler(void* arg)
17 {
18     uint32_t gpio_num = (uint32_t) arg;
19     //把消息存储到消息队列
20     xQueueSendFromISR(gpio_evt_queue, &gpio_num, NULL);
21 }
22
23 /*任务函数*/
24 static void gpio_task_example(void* arg)
25 {
26     uint32_t io_num;
27     for(;;) {
28         /*消息队列里面有消息*/
29         if(xQueueReceive(gpio_evt_queue, &io_num, portMAX_DELAY)) {
30             printf("GPIO[%d] intr, val: %d\n", io_num, gpio_get_level(io_num));
31         }
32     }
33 }
34
35
```

进中断的时候确实是下降沿进来的
但是呢,读取引脚电平是在一个任务中
读取的,这时候读取的时候可能按键已经抖动
成高电平了,所以也可能读取1

分类: [ESP32学习开发](#)

好文要顶

关注我

收藏该文



杨奉武

关注 - 1

粉丝 - 629

0

0

« 上一篇 : [005-ESP32学习开发\(SDK\)-新建工程补充-通过官方示例创建工程](#)

posted on 2021-08-02 04:10 杨奉武 阅读(0) 评论(0) 编辑 收藏 举报

[刷新评论](#) [刷新页面](#) [返回顶部](#)

发表评论

编辑 预览

B

支持 Markdown

自动补全

提交评论 退出

[Ctrl+Enter快捷键提交]

【推荐】大型组态、工控、仿真、CAD\GIS 50万行VC++源码免费下载!

编辑推荐：

- 聊聊【向上管理】中的“尺度”
- 一个故事看懂进程间通信技术
- 记一次 .NET 某云采购平台API 挂死分析
- 利用 PGO 提升 .NET 程序性能
- 我给鸿星尔克写了一个720°看鞋展厅

最新新闻：

- 特斯拉前CTO另立门户：搞旧电池回收，获7亿美元融资
- 新能源车和燃油车相争，4S店却先倒下了？
- 比微信支付更安全！今日起 北京地铁可用数字人民币买票充值
- 去年夏天，英特尔为何“崩盘”了，5年后能反超吗
- 天体物理学家在追寻“层次分明”的黑洞
- » 更多新闻...



单片机,物联网,上位机,...

扫一扫二维码，入群聊。