

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 - 751, 文章 - 0, 评论 - 317, 阅读 - 183万

导航

[博客园](#)
[首页](#)
[新随笔](#)
[联系](#)
[订阅](#)
[管理](#)

公告

渡我不渡她 -
Not available

00:00 / 03:41

- 渡我不渡她
- 小镇姑娘
- PDD洪荒之力

加入QQ群

昵称：杨奉武
 园龄：5年10个月
 粉丝：637
 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

Air724UG学习开发(2)
 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(15)
 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 SDK开发: 外设篇-系统任务(消息队列,通知)

<p>
 <iframe name="ifd" src="https://mnifdv.cn/resource/cnblogs/Learn8266ForSDK"
 frameborder="0" scrolling="auto" width="100%" height="1500">
 </iframe>
 </p>

ESP8266:SDK开发(源码见资料源码)

开发板购买链接:[开发板购买链接](#)

资料源

码:<https://github.com/yangfengwu45/learn-esp8266-sdk.git>

开发软

件:https://mnifdv.cn/resource/cnblogs/Learn8266ForSDK/AiThinkerIDE_V0.5

点击加入群聊【ESP8266开发交流群】： 加入QQ群

- 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。

- 1.01-准备工作-硬件说明
- 1.02-整体运行测试-APP使用SmartConfig配网绑定ESP8266,并通过MQTT远程通信控制,采集DHT11温湿度数据
- 2.01 开发环境搭建(RTOS 2.2.0)(建议只参考这篇文章搭建即可,教程以NONOS版本为主!)
- 2.01 开发环境搭建(NONOS 2.2.0)
- 2.02-外设篇-GPIO输出高低电平
- 2.03-外设篇-GPIO输入检测
- 2.04-外设篇-GPIO中断检测
- 2.05-外设篇-定时器,延时
- 2.05-外设篇-系统任务(消息队列,通知)
- 2.06-外设篇-串口
- 2.07-外设篇-PWM,呼吸灯(RTOS 2.2.0)
- 2.08-外设篇-SPI(RTOS 2.2.0)
- 2.09-外设篇-温湿度传感器-DHT11
- 2.11-外设篇-时钟芯片DS1302使用和拓展知识time.h的使用
- 2.12-外设篇-内存分布说明及Flash读写
- 3.01-网络篇-8266TCP服务器(LWIP,RAW模式,PCB控制块)(RTOS 2.2.0)
- 3.02-网络篇-8266TCP服务器(espconn实现)(NONOS 2.2.0)
- 3.03-网络篇-8266连接路由器(实现局域网网络通信控制)
- 3.04-网络篇-TCP客户端(espconn)(NONOS 2.2.0)
- 3.10-网络篇-UDP通信 - 微信小程序篇-微信小程序通过UDP实现和ESP8266局域网通信控制

ESP8266 LUA开发基础入门篇
备份(22)
ESP8266 SDK开发(33)
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)
更多

最新评论

1. Re:(一)Lua脚本语言入门
楼主可以分享一下这本电子
书吗?
--戢思
2. Re:学习C语言-学习指针
学到了学到了,很清晰的思
路,给博主赞赞赞
--*夏日么么茶

阅读排行榜

1. ESP8266使用详解(AT,LUA,
SDK)(172847)
2. 1-安装MQTT服务器(Windo
ws),并连接测试(99168)
3. ESP8266刷AT固件与node
mcu固件(64823)
4. 用ESP8266+android,制作
自己的WIFI小车(ESP8266篇)
(64354)
5. 有人WIFI模块使用详解(385
48)

[序通过UDP实现和ESP8266局域网通信控制](#)

-
- [4.01-自建MQTT服务器篇-安装MQTT服
务器,ESP8266连接MQTT服务器实现通信控制](#)
- [4.02-自建MQTT服务器篇-ESP8266配网
SmartConfig](#)
- [4.03-自建MQTT服务器篇-APP使用SmartConfig
配网绑定ESP8266,并通过MQTT远程通信控制](#)
- [4.05-自建MQTT服务器篇-编写微信小程序连接
MQTT服务器程序](#)
-
-
- [4.10 阿里云物联网平台篇-测试MQTT调试助手和
ESP8266连接阿里云物联网平台](#)
- [4.11-阿里云物联网平台篇-ESP8266连接阿里云
物联网平台使用自定义Topic实现自定义数据的上
报和数据下发](#)
- [4.12-阿里云物联网平台篇-ESP8266连接阿里云
物联网平台使用物理模型Topic实现温湿度数据显
示](#)
- [4.13-阿里云物联网平台篇-阿里云物联网平台加入
规则引擎\(云产品流转\),让MQTT设备之间实现通信](#)
- [4.14-阿里云物联网平台篇-Android和ESP8266连](#)

说明

咱先使用一下这个功能,然后再说明使用这个功能的应用场合

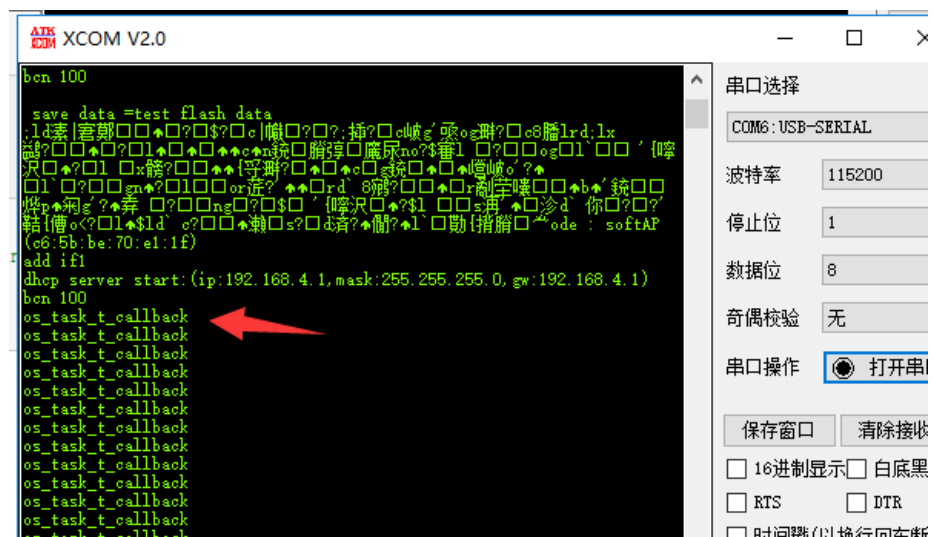
写上以下程序

```
user_main.c  uartc  user_interface.h  ets_sys.h  Makefile  Makefile  Makefile  Makefile
25 #include "user_websvr.h"
30 #if ESP_PLATFORM
31 #include "user_esp_platform.h"
32 #endif
33 #include "driver/uart.h"
34
35 #define os_event_t_buff_len 255 /*消息队列长度;最大255*/
36 os_event_t os_event_t_buff[os_event_t_buff_len]; /*存储消息的数组*/
37 #define TaskPrio 2 /*任务等级(0,1,2),2是最高等级*/
38
39 os_timer_t os_timer_one; /*定义软件定时器结构体变量*/
40
41 uint32_t priv_param_start_sec;
42 #user_rf_cal_sector_set(void)
43
44 void ICACHE_FLASH_ATTR
45 user_rf_pre_init(void){}
46
47 //定时器回调函数
48 void os_timer_one_function(void *parg){
49     //把消息插入队列(sig=0;par=0)
50     system_os_post(TaskPrio, 0, 0);
51 }
52
53 /* 系统有空余时间的时候会取出消息队列里面的消息
54 如果有消息则会调用其回调函数 */
55 void os_task_t_callback(os_event_t *events){
56     if(events->sig == 0 && events->par == 0){
57         os_printf("os_task_t_callback\r\n");
58     }
59 }
60
61 #FunctionName::user_init
62 void ICACHE_FLASH_ATTR user_init(void){
63     uart_init_2(BIT_RATE_115200,BIT_RATE_115200);
64     //os_task_t_callback:任务回调函数 //TaskPrio:任务等级(0,1,2),2是最高等级
65     //os_event_t_buff:消息队列记录的数组 //os_event_t_buff_len:消息队列长度
66     system_os_task(os_task_t_callback, TaskPrio, os_event_t_buff, os_event_t_buff_len);
67
68     //配置定时器
69     os_timer_setfn(&os_timer_one,os_timer_one_function,NULL); //os_timer_one:定时器结构体变量 os_t
70     //使能定时器
71     os_timer_arm(&os_timer_one,1000,1); //os_timer_one:定时器变量 1:1s进一次 1:循环
72 }
```

10. android客户端+eps8266+单片机+路由器之远程控制系统(31338)

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)

会看到程序每隔1S打印 os task t callback



关于ESP8266的系统任务

ESP8266内部可以运行消息队列任务.总共可以创建3个消息队列任务

创建任务:

调用创建消息队列任务函数的时候设置好任务的回调函数, 任务的消息等级, 缓存队列消息的数组.

下面是固定的形式哈,就是这样子写.

```
user_main.c  uart.c  user_interface.h  ets_sys.h  Makefile  Makefile  Makefile  Makefile
33 #include "driver/uart.h"
34
35 #define os_event_t_buff_len 255 /*消息队列长度;最大255*/
36 os_event_t os_event_t_buff[os_event_t_buff_len]; //存储消息的数组
37 #define TaskPrio 2 //任务等级(0,1,2),2是最高等级
38
39 os_timer_t os_timer_one; //定义软件定时器结构体变量
40
41 uint32_t priv_param_start_sec;
43 #user_rf_cal_sector_set(void)
93 #void ICACHE_FLASH_ATTR
94 user_rf_pre_init(void){}
95
96 //定时器回调函数
97 void os_timer_one_function(void *pang){
98     //把消息插入队列(sig=0;par=0)
99     system_os_post(TaskPrio, 0, 0);
100 }
101 /* 系统有空余时间的时候会取出消息队列里面的消息
102 如果有消息则会调用其回调函数 */
103 void os_task_t_callback(os_event_t *events){
104     if(events->sig == 0 && events->par == 0){
105         os_printf("os_task_t_callback\r\n");
106     }
107 }
108 /*
109  * FunctionName: user_init
110  * Description: entry of user application, init user function here
111  * Parameters: none
112  * Returns: none
113  */
114 void ICACHE_FLASH_ATTR user_init(void){
115     uart_init_2(BIT_RATE_115200, BIT_RATE_115200);
116     //os_task_t_callback:任务回调函数 //TaskPrio:任务等级(0,1,2),2是最高等级
117     //os_event_t_buff:消息队列记录的数组 //os_event_t_buff_len:消息队列长度
118     system_os_task(os_task_t_callback, TaskPrio, os_event_t_buff, os_event_t_buff_len);
119
120     //配置定时器
121     os_timer_setfn(&os_timer_one, os_timer_one_function, NULL); //os_timer_one:定时器结构体变量 os_timer
122     //使能定时器
123     os_timer_arm(&os_timer_one, 1000, 1); //os_timer_one:定时器变量 1:1s进一次 1:循环
```

把消息插入消息队列:

第一个参数 TaskPrio填写的是任务消息等级;

后面的两个参数是 0- 4294967296之间的数

后面是把0,0插入了消息队列.第一个0设置的是消息队列sig值; 第二个0设置的是消息队列par的值.

```

user_main.c  os_type.h  ets_sys.h
35 #define os_event_t_buff_len 255 /*消息队列长度;最大255*/
36 os_event_t ... os_event_t_buff[os_event_t_buff_len]; //存储消息的数
37 #define TaskPrio 2 //任务等级(0,1,2),2是最高等级
38
39 os_timer_t os_timer_one; //定义软件定时器结构体变量
40
41 uint32 priv_param_start_sec;
43 user_rf_cal_sector_set(void)
93 void ICACHE_FLASH_ATTR
94 user_rf_pre_init(void){}
95
96 //定时器回调函数
97 void os_timer_one_function(void *parg){
98     //把消息插入队列(sig=0;par=0)
99     system_os_post(TaskPrio, 0, 0);
100 }
101 /* 系统有空余时间的时候会取出消息队列里面的消息
102 如果有消息则会调用其回调函数 */
103 void os_task_t_callback(os_event_t *events){
104     if(events->sig == 0 && events->par == 0){
105         os_printf("os_task_t_callback\r\n");
106     }
107 }

```

系统有空的时候就从消息队列里面取出数据,然后调用回调函数:

```

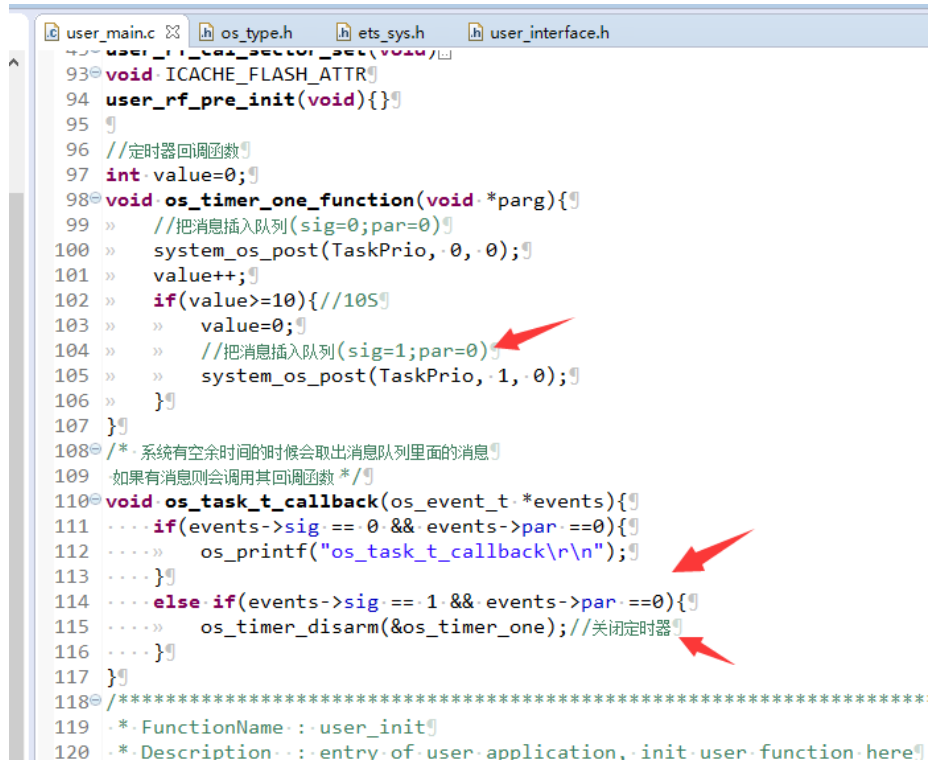
user_main.c  os_type.h  ets_sys.h  user_interface.h
35 #define os_event_t_buff_len 255 /*消息队列长度;最大255*/
36 os_event_t ... os_event_t_buff[os_event_t_buff_len]; //存储消息
37 #define TaskPrio 2 //任务等级(0,1,2),2是最高等级
38
39 os_timer_t os_timer_one; //定义软件定时器结构体变量
40
41 uint32 priv_param_start_sec;
43 user_rf_cal_sector_set(void)
93 void ICACHE_FLASH_ATTR
94 user_rf_pre_init(void){}
95
96 //定时器回调函数
97 void os_timer_one_function(void *parg){
98     //把消息插入队列(sig=0;par=0)
99     system_os_post(TaskPrio, 0, 0);
100 }
101 /* 系统有空余时间的时候会取出消息队列里面的消息
102 如果有消息则会调用其回调函数 */
103 void os_task_t_callback(os_event_t *events){
104     if(events->sig == 0 && events->par == 0){
105         os_printf("os_task_t_callback\r\n");
106     }
107 }
108 /* *****
109 *. FunctionName : user_init

```

假设需要在适当的时候发送个通知关闭定时器

system_os_post可以写到任意地方,我下面只是演示,就放到了定时器里面

假设咱设置sig为1;par为1的时候用来关闭定时器



```
user_main.c  os_type.h  ets_sys.h  user_interface.h
93 void ICACHE_FLASH_ATTR
94 user_rf_pre_init(void){}
95
96 //定时器回调函数
97 int value=0;
98 void os_timer_one_function(void *parg){
99     //把消息插入队列(sig=0;par=0)
100     system_os_post(TaskPrio, 0, 0);
101     value++;
102     if(value>=10){//10S
103         value=0;
104         //把消息插入队列(sig=1;par=0)
105         system_os_post(TaskPrio, 1, 0);
106     }
107 }
108 /* 系统有空余时间的时候会取出消息队列里面的消息
109 如果有消息则会调用其回调函数 */
110 void os_task_t_callback(os_event_t *events){
111     if(events->sig == 0 && events->par == 0){
112         os_printf("os_task_t_callback\r\n");
113     }
114     else if(events->sig == 1 && events->par == 0){
115         os_timer_disarm(&os_timer_one); //关闭定时器
116     }
117 }
118 /*****
119  * FunctionName:: user_init
120  * Description:: entry of user application, init user function here
121  */
```

The screenshot shows a code editor with four tabs: user_main.c, os_type.h, ets_sys.h, and user_interface.h. The code in user_main.c includes a timer callback function os_timer_one_function and a task callback function os_task_t_callback. Red arrows point to the following lines: line 104 (sig=1;par=0), line 115 (os_timer_disarm), and line 112 (os_printf).

在网络通信的时候,官方也是建议使用系统消息队列的形式关闭网络连接

- 调用 `espconn_accept` 侦听 TCP 连接;
- TCP连接建立成功后, 在连接成功的回调函数 (`espconn_connect_callback`) 中, 注册接收数据的回调函数, 发送数据成功的回调函数和断开连接的回调函数。
 - (调用 `espconn_regist_rcvcb`, `espconn_regist_sentcb` 和 `espconn_regist_disconcb`)

3. espconn callback

注册函数	回调函数	说明
<code>espconn_regist_connectcb</code>	<code>espconn_connect_callback</code>	TCP 连接建立成功
<code>espconn_regist_reconcb</code>	<code>espconn_reconnect_callback</code>	TCP 连接发生异常而断开
<code>espconn_regist_sentcb</code>	<code>espconn_sent_callback</code>	TCP 或 UDP 数据发送完成
<code>espconn_regist_rcvcb</code>	<code>espconn_rcv_callback</code>	TCP 或 UDP 数据接收
<code>espconn_regist_write_finish</code>	<code>espconn_write_finish_callback</code>	数据成功写入 TCP 数据缓存
<code>espconn_regist_disconcb</code>	<code>espconn_disconnect_callback</code>	TCP 连接正常断开

注意

- 回调函数中传入的指针 `arg`, 对应网络连接的结构体 `espconn` 指针。该指针为 SDK 内部维护的指针, 不同回调传入的指针地址可能不一样, 请勿依此判断网络连接。可根据 `espconn` 结构体中的 `remote_ip`, `remote_port` 判断多连接中的不同网络传输。
- 如果 `espconn_connect` (或者 `espconn_secure_connect`)失败, 返回非零值, 连接未建立, 不会进入任何 `espconn` callback。
- 请勿在 `espconn` 任何回调中调用 `espconn_disconnect` (或者 `espconn_secure_disconnect`) 断开连接。如有需要, 可以在 `espconn` 回调中使用触发任务的方式 (`system_os_task` 和 `system_os_post`) 调用 `espconn_disconnect` (或者 `espconn_secure_disconnect`) 断开连接。

使用任务代替定时器

定时器只能定时在5ms和100us的级别;而且经过测试,频繁的定时器会影响到网络信号的传输.

3.1. 软件定时器

以下软件定时器接口位于 `/ESP8266_NONOS_SDK/include/osapi.h`。请注意，以下接口使用的定时器由软件实现，定时器的函数在任务中被执行。因为任务可能被中断，或者被其他高优先级的任务延迟，因此以下 `os_timer` 系列的接口并不能保证定时器精确执行。

如果需要精确的定时，例如，周期性操作某 GPIO，请使用硬件中断定时器，具体可参考 `hw_timer.c`，硬件定时器的执行函数在中断里被执行。

注意：

- 对于同一个 timer，`os_timer_arm` 或 `os_timer_arm_us` 不能重复调用，必须先 `os_timer_disarm`
- `os_timer_setfn` 必须在 timer 未使能的情况下调用，在 `os_timer_arm` 或 `os_timer_arm_us` 之前或者 `os_timer_disarm` 之后

1. `os_timer_arm`

功能：

使能毫秒级定时器

函数定义：

```
void os_timer_arm (
    os_timer_t *ptimer,
    uint32_t milliseconds,
    bool repeat_flag
)
```

参数：

`os_timer_t *ptimer` : 定时器结构

`uint32_t milliseconds` : 定时时间，单位：毫秒

如未调用 `system_timer_reinit`，可支持范围 5 ~ 0x68D7A3

如调用了 `system_timer_reinit`，可支持范围 100 ~ 0x689D0

`bool repeat_flag` : 定时器是否重复

返回：

无

如果需要轮训,可以使用下面的方式

```
user_main.c | os_type.h | ets_sys.h | user_interface.h
32 #endif
33 #include "driver/uart.h"
34
35 #define os_event_t_buff_len 255 /*消息队列长度;最大255*/
36 os_event_t... os_event_t_buff[os_event_t_buff_len]; //存储消息的数组
37 #define TaskPrio 2 //任务等级(0,1,2),2是最高等级
38
39 os_timer_t os_timer_one; //定义软件定时器结构体变量
40
41 uint32_t priv_param_start_sec;
42 #user_rf_cal_sector_set(void)
43 #void ICACHE_FLASH_ATTR
44 user_rf_pre_init(void){}
45
46 /* 系统有空余时间的会取出消息队列里面的消息
47 如果有消息则会调用其回调函数 */
48 void os_task_t_callback(os_event_t *events){
49     if(events->sig == 0 && events->par == 0){
50         system_os_post(TaskPrio, 0, 0);
51     }
52 }
53
54 /* FunctionName: user_init
55  * Description: entry of user application, init user function here
56  * Parameters: none
57  * Returns: none
58  */
59
60 void ICACHE_FLASH_ATTR user_init(void){
61     uart_init_2(BIT_RATE_115200,BIT_RATE_115200);
62     //os_task_t_callback:任务回调函数 //TaskPrio:任务等级(0,1,2),2是最高等级
63     //os_event_t_buff:消息队列记录的数组 //os_event_t_buff_len:消息队列长度
64     system_os_task(os_task_t_callback, TaskPrio, os_event_t_buff, os_event_t_buff_len);
65     system_os_post(TaskPrio, 0, 0); //把消息插入队列
66 }
67
68
```


轮训一次任务的时间基本上是固定的,可以使用变量累加来执行定时操作

```
user_main.c | os_type.h | ets_sys.h | user_interface.h
38
39 os_timer_t os_timer_one; //定义软件定时器结构体变量
40
41 uint32 priv_param_start_sec;
43 user_rf_cal_sector_set(void)
93 void ICACHE_FLASH_ATTR
94 user_rf_pre_init(void){}
95
96 /* 系统有空余时间的时候会取出消息队列里面的消息
97 如果有消息则会调用其回调函数 */
98 int value=0;
99 void os_task_t_callback(os_event_t*events){
100     if(events->sig == 0 && events->par == 0){
101         system_os_post(TaskPrio, 0, 0);
102     }
103
104     value++;
105     if(value>300000){ //大约1S
106         value=0;
107         os_printf("os_task_t_callback\r\n");
108     }
109 }
110 /*****
```

停止轮训

```
user_main.c | os_type.h | ets_sys.h | user_interface.h
38
39 os_timer_t os_timer_one; //定义软件定时器结构体变量
40
41 uint32 priv_param_start_sec;
43 user_rf_cal_sector_set(void)
93 void ICACHE_FLASH_ATTR
94 user_rf_pre_init(void){}
95
96 /* 系统有空余时间的时候会取出消息队列里面的消息
97 如果有消息则会调用其回调函数 */
98 int value=0;
99 void os_task_t_callback(os_event_t*events){
100     if(events->sig == 0 && events->par == 0){
101         if(1){ //不再把消息插入队列即可结束轮训,根据项目的实际情况修改
102             system_os_post(TaskPrio, 0, 0);
103         }
104     }
105
106     value++;
107     if(value>300000){ //大约1S
108         value=0;
109         os_printf("os_task_t_callback\r\n");
110     }
111 }
112 /*****
```

分类: ESP8266 SDK开发

好文要顶

关注我

收藏该文





杨奉武
关注 - 1
粉丝 - 637

« 上一篇：[2-Air724UG\(4G全网通GPRS\)开发-下载AT指令固件](#)

posted on 2021-08-14 04:56 杨奉武 阅读(1) 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】[百度智能云2021普惠上云节：新用户首购云服务器低至0.7折](#)
【推荐】[阿里云云大使特惠：新用户购ECS服务器1核2G最低价87元/年](#)
【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：

- [C# 10 完整特性介绍](#)
- [不是技术也能看懂云原生](#)
- [记一次接口慢查排查](#)
- [一个故事看懂HTTPS](#)
- [人人都能看懂系列：分布式系统改造方案之数据篇](#)



最新新闻：

- [你还抢购华为吗？门店可能没有存货了](#)
- [字节新消费版图大起底：投资自营双管齐下](#)
- [上市破发、资金受困，理想“勇争第一”空成口号](#)
- [百度二季度财报点评：以更高维的ESG识别其价值](#)
- [锂电专利战争：欧美、日韩围剿，中国换道超车](#)
- » [更多新闻...](#)

历史上的今天：

2019-08-14 16-网页,网站,微信公众号基础入门(网页版MQTT,页面控件位置调整入门)

Powered by:

博客园

Copyright © 2021 杨奉武

Powered by .NET 5.0 on Kubernetes



单片机,物联网,上位机,...

扫一扫二维码，入群聊。