机智云
Gizwits

开发指南

编制人	AndyGao	审核人	Sean	批准人	
产品名称		产品编号		文档编号	
会签日期			版本	0.2	

修改记录：

修改时间	修改记录	修改人	版本	备注
20160909	初建	AndyGao	V0.1	
20161030	增加模组产测、绑定、获取 NTP 网络时间接口	TerryLi	V0.2	

目录：

1 文件介绍

```
.
├── app
│   ├── driver                                  //基础驱动层,包括按键驱动
│   │   └── hal_key.c
│   ├── gen_misc.bat
│   ├── gen_misc.sh                             //编译工具脚本,执行./gen_misc.sh
│   ├── Gizwits                                 //机智云协议层,包括通信协议和用户事件处理
│   │   ├── gizwits_product.c
│   │   ├── gizwits_product.h
│   │   ├── gizwits_protocol.c
│   │   └── gizwits_protocol.h
│   ├── include
│   │   └── driver
│   │       └── hal_key.h
│   └── user
│       └── user_main.c                         //APP层,user_init()为入口函数,模组初始化,task创建等
├── bin
│   └── upgrade
│       └── user1.4096.new.6.bin                //编译生成的目标文件,烧录使用
├── ESP8266 SoC 开发指南-V0.1.pdf               //开发指南
├── include
│   └── gagent_external.h
├── ld
├── lib
│   └── libgagent.a                             //gagent封装库文件
└── tools
```

1. gizwits_product.c
   该文件为产品相关处理函数，如 gizwitsEventProcess()。
2. gizwits_product.h
3. 该文件为 gizwits_product.c 的头文件，如 HARDWARE_VERSION、SOFTWARE_VERSION。
4. gizwits_protocol.c
   该文件为 SDK API 接口函数定义文件。
5. gizwits_protocol.h
   该文件为 gizwits_protocol.c 对应头文件，相关 API 的接口声明均在此文件中。
6. 其他文件
   a) app/driver/hal_key.c
   按键模块函数，实现了 2 个 key 的长短按键检测功能，使用 demo 见 gizwitsInit()函数。
   b) app/include/driver/hal_key.h
   hal_key.c 模块的头文件，声明相关接口函数。
   c) app/user/user_main.c
   Esp8266 程序入口函数所在文件，入口函数为 void user_init(void)。
   注意：本工程代码只适用于 ESP8266，4M Flash，QIO 方式的芯片。

2 API 介绍

void gizwitsInit(void)

gizwits 协议初始化接口。

用户调用该接口可以完成 Gizwits 协议相关初始化（包括协议相关定时器、串口的初始化）。

void gizwitsSetMode(uint8_t mode)

参数 mode[in]：WIFI_MODE_TYPE_T 枚举值

参数为 WIFI_RESET_MODE，恢复模组出厂配置接口，调用会清空所有配置参数，恢复到出厂默认配置。

参数为 WIFI_SOFTAP_MODE 或 WIFI_AIRLINK_MODE，配置模式切换接口，支持 SoftAP 和 AirLink 模式。参数为 WIFI_SOFTAP_MODE 时配置模组进入 SoftAp 模式，参数为 WIFI_AIRLINK_MODE 配置模组进入 AirLink 模式。

参数为 WIFI_PRODUCTION_TEST，模组进入产测模式。

参数为 WIFI_NINABLE_MODE，模组进入可绑定模式，可绑定时间为 NINABLETIME(gizwits_protocol.h 中声明)，默认为 0，表示模组永久可绑定。

uint32_t gizwitsGetTimeStamp(void)

获取 NTP 时间接口。

用户调用该接口可以获取当前网络时间。

*void gizwitsHandle(dataPoint_t *dataPoint)*

参数 dataPoint[in]:用户设备数据点。

gizwits 数据点更新上报处理，用户调用该接口可以完成设备数据的变化上报。

*int8_t gizwitsEventProcess(eventInfo_t *info, uint8_t *data, uint32_t len)*

参数 info[in]:事件队列

参数 data[in]:协议数据

参数 len [in]:协议数据长度

用户数据处理函数，包括 wifi 状态更新事件和控制事件。

a) Wifi 状态更新事件

WIFI_开头的事件为 wifi 状态更新事件，data 参数仅在 WIFI_RSSI 有效，data 值为 RSSI 值，数据类型为 uint8_t，取值范围 0~7。

b) 控制事件

与数据点相关，本版本生成代码会打印相关事件信息，相关数值也一并打印输出，用户只需要做相应具体处理即可。

注意：SOC_ESP8266 关于浮点型数据日志打印输出尚存在问题，但是不影响数据的正确性，只影响打印输出，若需要请联系机智云工程师。

*int32_t gizwitsPassthroughData(uint8_t * data, uint32_t len)*

参数 data[in]:输入的私有协议数据
参数 len [in]:输入的私有协议数据长度
gizwits 上报透传数据接口，用户调用该接口可以完成私有协议数据的上报。

3 二次开发

配置入网

Esp8266 支持 SoftAp 和 AirLink 两种方式配置入网，相应接口为 gizwitsSetMode()，本版软件采用按键的方式，相关代码参考 user_main.c 文件的 key 相关操作。

另外，可以通过 gizwitsSetMode()接口复位模组，恢复默认出厂设置。

数据采集

该 SOC 代码默认给用户开启了 userTimer 定时器，定时周期为 1s，并且在 userTimerFunc()函数中以伪代码形式提示用户在此处获取需要上报的数据点数据。特别提醒，该定时器周期为 1s，需要针对不同的需求，用户调整数据点数据的采集周期。

事件处理

数据点方式将转换成数据点事件，开发者只需要在 gizwits_product.c 文件的 gizwitsEventProcess()相应事件下作具体处理即可。

透传方式数据下达后会生成 SIG_PASSTHROUGH 信号，数据内容和数据长度见注释提示。

其他

Wifi 状态

参考接口 gizwitsEventProcess()，本版软件已经将 wifi 状态数据转换成了 event，开发者仅关注相应事件即可。