

基于串口通信的人脸识别模块通信协议 V1.4

修改记录

版本	时间	修改内容
V1.0	2019.05.31	新建协议
V1.1	2019.06.03	head_fix改为pkt_fix 添加串口 IO 说明
V1.2	2019.06.05	所有数据包增加code及msg
V1.3	2019.06.13	增加翻转屏幕及摄像头接口
V1.4	2019.06.14	增加使用特征值添加用户接口 修复bug 添加初始化添加版本信息

硬件连接

K210	MCU	说明
I09	uart_rx	K210 通信串口 TX
I010	uart_tx	K210 通信串口 RX
I016	record key	short press record, long press clear
I022	relay_act_low	relay output
I023	relay_act_high	relay output

Json 基本格式

```
{"vesrion":$protocol_version,"type":"cmd_type","code":0,"msg":"msg","param":  
{"xx":"xx","xx":xx}}\r\n
```

说明

- 当前 protocol_version 为 1
- 数据包结尾必须加 \r\n
- 数据包中间不允许有 \r\n

cmd_type

cmd_type	说明
----------	----

cmd_type	说明
init	模块发送初始化完成消息
pkt_prase_failed_ret	数据包解析出错信息
set_cfg	设置模块参数
set_cfg_ret	设置模块参数结果
get_cfg	获取模块参数
get_cfg_ret	获取模块参数结果
cal_pic_fea	计算图片人脸特征值
cal_pic_fea_ret	JPEG 图片特征值计算结果
del_user_by_uid	使用 UID 进行用户删除
del_user_by_uid_ret	使用 UID 进行用户删除结果
face_info	识别人脸后输出信息
query_face	查询当前模块存储人脸信息
query_face_ret	查询当前模块存储人脸信息结果
lcd_cam_roate	设置LCD及CAMERA进行旋转
lcd_cam_roate_ret	设置LCD及CAMERA进行旋转结果
add_uer_by_fea	通过特征值添加用户
add_uer_by_fea_ret	通过特征值添加用户结果

以上 cmd 可点击跳转

通信协议

初始化完成

```
{ "version": 1, "type": "init", "code":0,"msg": "init done","version":"v1.4" }
```

说明

模块启动完成之后发送，之后才可进行操作

数据包解析出错信息

```
{"version":1,"type":"pkt_prase_failed_ret","msg":"json prase failed","code":1,"cmd":"unknown"}
```

说明

接收到错误的数据包之后返回，并指出错误的地方，主要用来协助排查错误

设置模块参数

```
{"version":1,"type":"set_cfg","cfg":  
{ "uart_baud":115200,"out_feature":0,"open_delay":1,"pkt_fix":0,"auto_out_feature":  
0,"out_interval_in_ms":500,"fea_gate":70}}
```

说明

uart_baud: 模块通信串口波特率，默认为 115200

out_feature: 识别到人脸之后输出信息是否附带人脸的特征值，默认为 0

open_delay: 输出 IO 打开（翻转）时间，默认为 1

pkt_fix: 通信协议是否加头尾以及 CRC16 校验，默认为 0，不开启将直接传输数据

auto_out_feature: 识别到人脸之后自动输出特征值，不进行比对

out_interval_in_ms: 输出特征值间隔时间 ms,默认 500，设置为 0 不限制，最小限制为 500

fea_gate: 人脸比对阈值

开启 **pkt_fix** 的通信格式

```
0xAA 0x55 HI(LEN) LO(LEN) HI(CRC16) LO(CRC16) ...(data)... 0x55 0xAA
```

可使用 <http://www.ip33.com/crc.html> 选择 CRC-16/X25 $x^{16}+x^{12}+x^5+1$

设置模块参数结果

```
{"version":1,"type":"set_cfg_ret","msg":"save cfg success","code":0}
```

说明

`msg`: 返回信息

`code`: 返回状态码

- 状态码
 - 0: 设置模块参数成功
 - 1: 解析 `Json` 失败, `msg` 中会指出错误地方
 - 2: 保存模块参数失败

如果更改了 `uart_baud` 以及 `pkt_fix` 模块会自动重启使更改生效

获取模块参数

```
{"version":1,"type":"get_cfg"}
```

说明

获取模块当前配置

获取模块参数结果

```
{"version":1,"type":"get_cfg_ret","code":0,"msg":"get_cfg_success","cfg":{"uart_baud":115200,"out_feature":0,"open_delay":1,"pkt_fix":0,"auto_out_feature":0,"out_interval_in_ms":500,"fea_gate":70}}
```

说明

与 `set_cfg` 参数一致, 只是 `type` 不同

计算图片人脸特征值

```
{"version":1,"type":"cal_pic_fea","img":{"size":10152,"auto_add":0,"sha256":"E65083CFEEEE8F377094C2297E8D3691C23AA8BAD33A82B5E4E4981914FFAC74"}}
```

说明

size: Jpeg 图片的大小, 注意, 最大支持**30K**的图片

auto_add: 识别成功后自动添加用户到模块 (默认不添加, 那么返回的**UID**全为**0**)

sha256: Jpeg 图片的 **sha256** 校验值 (**7zip** 以及 **haozip** 都支持计算)

收到以下返回内容 (**code** 为 **1**) 就可以开始发送图片, 必须在**10s**内发送完毕。超时退出接受**jpeg**状态

```
{"version":1,"type":"cal_pic_fea_ret","code":1,"msg":"please start send jpeg image","info":{"uid":"null","feature":"null"}}
```

当 **info** 中的 **code** 为 **1** 表示可以开始发送 **Jpeg** 图片

可使用 **XCOM** 或者其他串口助手的发送文件功能, 载入 **Jpeg** 文件, 选择发送即可

计算图片人脸特征值结果

- 正确返回

```
{"version":1,"type":"cal_pic_fea_ret","code":0,"msg":"get feature success","info":{"uid":"0A6F4FB4000000000000000000000000","feature":"feature base64 encode"}}
```

- 错误返回(举例)

```
{"version":1,"type":"cal_pic_fea_ret","code":2,"msg":"can not find img.sha256","info":{"uid":"null","feature":"null"}}
```

说明

code: 状态码

msg: 返回信息

uid: 人脸在模块中存储的 **UID**, 如果需要删除相应人脸需要此 **UID**

feature: 人脸计算得出的特征值 (**base64 encode**)

在**计算图片人脸特征值**数据包中可选择是否自动添加用户到模块中

- 状态码

- 0: 计算特征值成功
- 1: 可以开始发送 Jpeg 图片
- 2: 解析 Json 出错, msg 中会指出出错地方
- 3: 模块存储已满
- 4: 图片中有多张人脸
- 5: 图片中没有人脸 (可能是方向不对, 或者是人脸太小)
- 6: Jpeg 解码失败 (图片不是 320x240 分辨率)
- 7: Jpeg 文件 sha256 校验失败, 或者等待接受图片超时

删除用户

```
{"version": 1, "type": "del_user_by_uid", "uid": "1BC6EB528C0000000000000000000000"}
```

说明

uid: 需要删除的用户 uid

当 uid 为 FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF (32个F) 时删除所有的用户

如果失败在返回结果有具体说明

删除用户结果

```
{"version": 1, "type": "del_user_by_uid_ret", "code": 3, "msg": "can not find user by uid"}
```

说明

msg: 执行结果

code: 状态码

- 状态码

0: 删除成功

- 1: `Json` 解析出错，缺少关键字
- 2: `flash` 操作失败
- 3: 未能查找到对应 `uid` 的用户

输出人脸信息

```
{"version":1,"type":"face_info","code":0,"msg":"have face","info":  
{"pic":"540A730200000000","total":1,"current":0,"x1":34,"y1":34,"x2":171,"y2":178,  
"score":0,"uid":"null","feature":"feature base64 encode or null"}}
```

说明

`pic`: 当有多个人脸时用来判断是否属于同一张图片

`total`: 本次识别中人脸总数

`current`: 本次识别中第 `N` 张人脸（从 `0` 开始计数）

`x1`: 人脸坐标框左上角 `x`

`y1`: 人脸坐标框左上角 `y`

`x2`: 人脸坐标框右下角 `x`

`y2`: 人脸坐标框右下角 `y`

`score`: 人脸匹配值（直接输出特征值时为 `0`）

`uid`: 人脸对应的 `UID`（直接输出特征值时为 `null`）

`feature`: 人脸特征值，可配置是否输出（不输出时为 `null`）

查询当前模块存储人脸信息

```
{"version":1,"type":"query_face","query":  
{"total":1,"start":0,"end":10,"out_feature":0}}
```

说明

`total`: 为 `1` 查询一共存储多少人脸信息，`start` 和 `end` 以及 `out_feature` 没有意义，为 `0` 表示查询从 `start` 到 `end` 区间的人脸 `UID`

out_feature: 人脸信息是否输出对应特征值，默认不输出（输出特征值的话一次只能查一个）

start: 查询区间起始值

end: 查询区间结束值

查询当前模块存储人脸信息结果

```
{ "version": 1, "type": "query_face_ret", "code": 0, "msg": "query uid ands feature success", "face": { "total": 2, "start": 0, "end": 1, "info": [ { "order": 0, "uid": "22BCD2392900000000000000000000", "feature": "feature_bease64_encode" } ... ] } }
```

说明

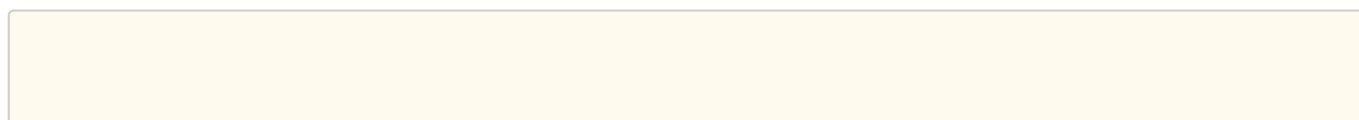
code: 状态码

msg: 返回信息

face:

- **total:** 当前模块中人脸总数
- **start:** 本次查询结果区间起始值（从0开始计数）
- **end:** 本次查询结果区间结束值
- **info:**
 - **order:** 当前人脸的顺序
 - **uid:** 当前人脸的 UID
 - **feature:** 当前人脸的特征值
- 状态码
 - 0: 查询成功
 - 1: Json 解析出错
 - 2: 当查询包中设置 total 为 1 时返回当前模块中存储的人脸数量
 - 3: 从 flash 中读取保存的数据时出错

屏幕及摄像头旋转




```
{"version":1,"type":"lcd_cam_roate","cfg":
{"get_cfg":0,"lcd_flip":0,"lcd_mirror":0,"cam_flip":0,"cam_mirror":0}}
```

说明

`get_cfg`: 获取当前模块的 LCD 及 CMA 参数

`lcd_flip`: lcd显示进行垂直镜像

`lcd_mirror`: lcd显示进行水平镜像

`cam_flip`: camera 进行垂直镜像

`cam_mirror`:camera 进行水平镜像

屏幕及摄像头旋转结果

```
{"version":1,"type":"lcd_cam_roate_ret","code":0,"msg":"set lcd and cam
success","cfg":{"lcd_flip":0,"lcd_mirror":0,"cam_flip":0,"cam_mirror":0}}
```

说明

`code`: 状态码

`msg`: 返回信息

- 状态码
 - 0: 设置成功
 - 1: Json 解析出错
 - 2: 保存配置出错
 - 3: 获取配置成功

通过特征值添加用户

```
{"version":1,"type":"add_uer_by_fea","user":
{"uid":"EDE6E800A200000000000000000000","fea":"feature base64 encode"}}
```

说明

`uid`: 用户的uid

`fea`: 用户人脸特征值

通过特征值添加用户结果

```
{"version":1,"type":"add_uer_by_fea_ret","code":0,"msg":"add user success","uid":"EDE6E800A20000000000000000000000"}
```

说明

code: 状态码

msg: 返回信息

uid: 添加成功的uid,如果失败全为0

- 状态码

0: 添加成功

1: Json 解析出错

2: 保存到flash失败

3: uid已存在

深圳码速SIPPEE