

DTU - JSON 格式数据协议解释

一、JSON 源码

```
1  {
2    "fota": 1,
3    "uartReadTime": 25,
4    "flow": "",
5    "param_ver": "100",
6    "pwrmod": "normal",
7    "password": "",
8    "netReadTime": 0,
9    "passon": 1,
10   "nolog": "1",
11   "plate": 0,
12   "reg": 0,
13   "convert": 0,
14   "uconf": [
15     [1, "9600", 8, 2, 0, ""],
16     []
17   ],
18   "conf": [
19     ["tcp", "ping", "180", "47.93.118.70", "9009", 1, "", "", "", ""],
20     [],
21     [],
22     [],
23     [],
24     [],
25     []
26   ],
27   "preset": {
28     "number": "13283863803",
29     "delay": "10",
30     "smsword": "update"
31   },
32   "apn": ["", "", ""],
33   "cmds": [
34     [],
35     []
36   ],
37   "pins": [],
38   "gps": {
39     "pio": ["pio3", "pio6", "pio7", "pio13", 0, "1"],
40     "fun": [1, "115200", 0, "10", 0, "json", "0", "|", "10", "1"]
41   },
42   "upprot": ["", "", "", "", "", "", ""],
43   "dwprot": ["", "", "", "", "", "", ""],
44   "warn": {
45     "adc0": [],
46     "adc1": [],
47     "vbatt": [],
48     "gpio": []
49   }
}
```

二、JSON 格式说明

- 这里不做更详细的字段解释，更详细的请参考《iRTU 参考手册V3.2》
- JSON不支持注释，以下用lua的注释符 -- 表示注释
- JSON的key-value序列无顺序

1、最外层：

- {} 是根目录，对应源码的 local dtu = { ... } 变量

2、第一层是“配置保存”指令对应的参数

- 这里本来因为历史原因要兼容最早的版本故没使用数组或JSON对象。

字段	功能	可选值
fota	远程升级开关	0 (关闭) 1 (开启)
uartReadTime	帧超时打包时长	25 (大于5的number)
flow	流控设置	"" (关闭) ("" 或 number)
param_ver	参数版本号	"100" (数字字符串)
pwrmod	功耗模式	"normal" "energy"
password	密码模式	"" (或字符串和字母组合)
netReadTime	往串口发送的打包超时	0 (number)

字段	功能	可选值
passon	传递标志位	0 (非透传) 1 (透传)
nolog	日志标志位	0 (关闭) 1 (打印)
plate	识别码标志位	0 (关闭) 1 (帧头添加IMEI)
reg	注册报文	0不发送,1发送JSON注册报文 { "csq": "rssi", "imei": "imei", "iccid": "iccid", "ver": "Version" }, 2发送HEX报文 "131234512345", 填字符串为用户自定义注册包
convert	报文HEX转换	0 (关闭) 1 (转换)

3、串口配置

- 字段key : uconf
- 字段value : 数组
- 数组下标代表串口 1 - 2

```

1      "uconf": [
2          [
3              1,          -- 串口ID号, 注意要和数组下标对应
4              "9600",    -- 串口波特率
5              8,         -- 串口数据位
6              2,         -- 串口parity
7              0,         -- 串口stopbits
8              ""         -- 485 GPIO 号
9          ],
10         [] -- 关闭串口2配置
11     ],

```

4、通道配置

- 字段key : conf
- 字段value : 数组
- 数组下标代表通道ID 1 - 7
- 数组有字段顺序, 下面举例TCP和阿里云自动注册, 其他同手册字段顺序

```

1      "conf": [
2          [          -- 数组下标代表通道ID号
3              "tcp",  -- 协议 TCP | UDP | MQTT |
4              "ping", -- 心跳包内容
5              "180",  -- 心跳包间隔
6              "47.93.118.70", -- HOST 地址
7              "9009", -- HOST 端口
8              1,     -- 通道透传捆绑的串口ID
9              "",    -- 保持为""
10             "",    -- 保持为""
11             "",    -- 自动采集任务间隔
12             ""     -- SSL 标志位 可选"ssl"
13         ],
14         [], -- 表示通道关闭
15     ]

```

```

16     "aliyun",      -- 公有云配置 如: "aliyun"
17     "auto",       -- 自动注册
18     300,          -- MQTT 保活时长
19     1800,        -- 自动采集任务间隔
20     "cn-shanghai", -- 阿里云地域代码
21     "ProductKey:", -- 阿里云项目key
22     "AccessKey ID:", -- 阿里云AccessKey
23     "Access Key Secret:", -- 阿里云 AccessKey 的 Secret
24     "plus",       -- 阿里云产品版本类型 (阿里云只有高级版了)
25     1,           -- MQTT CleanSession
26     0,           -- MQTT QOS
27     1,           -- MQTT 通道捆绑串口ID
28     "",          -- 订阅主题, 支持多主题参考《iRTU参考手册》
29     "",          -- 发布主题, 支持多主题参考《iRTU参考手册》
30 ],
31 [],
32 [],
33 [],
34 [],
35 ],

```

5、预置信息

- 字段key : preset
- 字段value : 对象
- 对应《iRTU参考手册》用户预定义电话和短信

```

1     "preset": {
2         "number": "13999999999", -- 白名单 允许呼入呼出的电话号码
3         "delay": "10",          -- 振铃时长
4         "smsword": "update"    -- 短信更新参数定义的字符串
5     },

```

6、设置APN

- 字段key : apn
- 字段value : 数组
- 对应《iRTU参考手册》设置APN
- 注意 : 这个指令只适合配置和使用不是同一张卡的场景

```

1     "apn": ["", "", ""],

```

7、自动任务采集

- 字段key : cmds
- 字段value : 数组
- 对应《iRTU参考手册》自动采集任务
- 注意 : 这个指令要配合通道的自动采集任务使用

```

1  "cmds": [
2    [          -- 串口1的自动采集任务
3      "100",   -- 一次下发采集指令之间的间隔(主意和通道配置的采集间隔不是一个意思)
4      "11 22 33 44 55 66 77 88 ", -- 下发给串口1的采集指令(会自动转成数字bin)
5      "11 22 33 44 55 66 77 88 "  -- 同上
6    ],
7    [] -- 串口2的自动采集任务关闭
8  ],

```

8、自定义IRTU预定义的GPIO

- 不必要的情况下不建议修改这个设置
 - 字段key : pins
 - 字段value : 数组
- 未配置的情况 :

```

1  "pins": [],

```

配置情况 :

```

1  "pins": [
2    "pio2", -- 网路指示灯的GPIO
3    "pio4", -- 与服务器脸上后通知GPIO
4    "pio4"  -- 重置DTU参数的GPIO
5  ],

```

9、GPS 部分

- 字段key : gps
- 字段value: 对象
- GPS 的配置略复杂，特别是Air800和Air530的波特率不同，强烈建议看手册和视频后再操作

```

1  "gps": {
2    "pio": ["pio3", "pio6", "pio7", "pio13", 0, "1"],
3    "fun": [1, "115200", 0, "10", 0, "json", "0", "|", "10", "1"]
4  },

```

10、数据流模版

- 字段key : upport
- 字段value: 数组，下标代表1-7 通道
- 字段key: dwprot
- 字段value:数组，下标代表1-7通道
- 数据流模版实现报文协议的转换，比如上行 Modbus 转 JSON，下行 JSON 转 Modbus
- 数据流模版使用 Lua 脚本，故都是字符串。
- 主意：使用JSON Encode 类 API 的时候会自动序列化字符串，如果是自己拼接，请注意序列化字符串。

```

1  "upprot": [ "", "", "", "", "", "", "" ],
2  "dwprot": [ "", "", "", "", "", "", "" ],

```

11、预警配置部分

基本参数 串口参数 网络通道参数 预置信息 GPIO GPS 数据流 **预警** 任务

GPIO触发上报

启用 不启用

第1组 ⊖

GPIO: pio7

触发模式: 下降沿 上升沿

上报消息: GPIO 7 is keypress!

上报通道: 1

上报方式: 互联网 短信 电话

+ 添加

ADC0触发上报

启用 不启用

触发模式: 低于 1000 mv 高于 3000 mv 回差电压 300 mv

上报消息: ADC0 warning!

上报通道: 1

检测频率: 10 (单位秒)

警报间隔: 100 (单位秒)

上报方式: 互联网 短信 电话

ADC1触发上报

启用 不启用

VBATT触发上报

启用 不启用

- 字段key:warn
- 字段value: 对象

```
1  "warn": {
2      "adc0": [
3          1,                -- 启用欠压报警
4          "1000",          -- 欠压下限 1000mV
5          1,                -- 启用过压报警
6          "3000",          -- 过压报警上限 3000mV
7          "300",           -- 报警电压回差（防止临界值反复报警）
8          "ADC0 warning!", -- 报警内容
9          "1",             -- 网路报警上报通道ID
10         "10",             -- 检测频率
11         "100",           -- 上报频率（防止报警没接触一直上报）
12         1,                -- 启用互联网上报
13         1,                -- 启用短信上报
14         0                  -- 不启用电话通知
15     ],
16     "adc1": [],
17     "vbatt": [],
18     "gpio": [
19         [
20             "pio7",       -- 按键GPIO
21             1,            -- 启用按下报警（下降沿触发）
22             0,            -- 不启用松开报警（上升沿触发）
```

```
23     "GPIO 7 is keypres!", -- 报警内容
24     "1",                -- 网路报警上报通道ID
25     1,                  -- 启用网路报警
26     0,                  -- 不启用短信报警
27     0                    -- 不启用电话通知
28     ]
29 ]
30 }
```