

L 门禁软件 API-websocket 接口说明-v4.0

--对应底层版本 T-L80xx.40.02.001+, 设备服务器版

本标准版本 6.2.9 +

1. 概述

本接口协议是针对运行在局域网内的终端设备，开发人员可以通过 HTTP 协议直接访问软件接口。本协议 API 为 restful 软件架构风格，支持 http 与 websocket 两者协议共存

2.1 接口规范

2.1 接口地址：<http://127.0.0.1:8888/> 接口 IP 是指设备服务器所处的电脑 ip

2.2 返回数据格式：JSON 字符串

2.3 返回字段描述：

字段	描述	类型	必须	详细说明
status	状态码	int	是	0-成功，其他状态见状态码表
msg	状态描述	String	是	状态码的详细描述，比如：成功
data	返回数据	String(json)	否	返回请求的数据，比如：请求人员列表，则 data 即为人员列表的 json 字符串

2.4 接口身份认证为确保数据传输的安全性，在正式传输数据时进行身份验证，暂时使用 basic 方式，默认用户：

Username: admin

Password: admin

备注：所有控制器的设备 id 需要大写，

例如：05D23ABC000013716C

2.2 接口目录

设备层：

序号	URL	备注
1	open_door	远程开门
2	settime	同步时间
3	task_cardpower	下发权限卡
4	get_userinfo	获取指定用户数据
5	get_event	获取刷卡记录
6	get_devinfo	获取设备信息
7	get_doorstate	获取门状态
8	get_devstate	获取设备状态
9	get_devdp	获取门点数
10	get_doorpart	获取门参

11	set_doorpart	设置门参
12	get_pointpart	获取点参
13	set_pointpart	设置点参
14	set_doorstate	设置门状态
15	get_specvertime	获取特殊时间表
16	set_specvertime	设置特殊时间表
17	get_temptime	获取临时时间表
18	set_temptime	设置临时时间表
19	get_doorstatetime	获取门状态时间组
20	set_doorstatetime	设置门状态时间组
21	get_doortimegroup	获取门禁时间组
22	set_doorimegroup	设置门禁时间组
23	factory_restore_setting	出厂还原设置
24	input_status	外联动
25	point_status	设置点状态
26	get_point_status	获取点状态
27	control_type	更换控制类型
28	downtemp	注册控制器
29	update_user	设置用户密码
30	get_user_event	查询卡号对应的人 员
31	all_user_event	查询所有卡号对应

		的人员
32	get_allcontrol	获取所有控制器 ip, sn

2.3 HTML5 Websocket 接口说明

WebSocket 是 HTML5 提供的一种全双工通信协议，使客户端与服务器之间数据交换更加便捷，并双向数据传输。

接口地址: <http://127.0.0.1:8888/websocket>

接口说明: 可直接调用接口地址 websocket client, 来获取实时刷卡数据, 也可提供 weboscket demo 自行编写 websocket client

附录:

1. 门状态表
2. 事件 id 表
3. 控制器状态表
4. 服务器存储事件表

3. 接口说明：分应用层和设备层

3-1 应用层 - pass

3-2 设备层

3.2.1 远程开门

URL: open_door

Method: POST

参数说明:

字段	描述	类型	详细说明
target_ip	目标 ip	String	控制板的 ip
sn	设备 id	String	控制器 id
door	门地址	Int	门号 1-4

返回示例:

```
{  
  "msg": "成功",  
  "status": 0  
}
```

3.2.2 同步时间

URL: settime

Method: POST

参数说明:

字段	描述	类型	详细说明
target_ip	目标 ip	String	控制板的 ip
sn	设备 id	String	控制器 id

返回示例:

```
{  
  "msg": "成功",  
  "status": 0  
}
```

3.2.3 下发权限卡

URL: task_cardpower

Method: POST

参数说明:

字段	描述	类型	详细说明
target_ip	目标 ip	String	控制板的 ip
sn	设备 id	String	控制器 id
id	标识	Int	1-3 ,1 添加权限卡, 2 删除权限卡, 3 修改权限卡, 4-预设临时卡时间
user_id	用户 id	String	最大 8 位数支持 10 进制和 16 进制
pw	用户密码	String	默认值 888888 密码 0-999999
door	门点权限	String	默认值 1111 0-无效 1-有效, 一位代表一个门, 默认 4 个门都生效
door_time	门点时间组	String	默认值 00000000 00-15 两位代表一个门 默认为 4 门的第 0 组
user_type	用户类型	String	0-固定卡, 1-临时卡

<code>valid_time</code>	临时卡有效 时间	String	临时卡才生效, 格式为 2019-01-01 00:00:00
<code>start_time</code>	预设临时卡 开始时间	String	id=4 的情况下, 此参 数才有效, id 不为 4 无需填写此参数, 格式 同上
<code>dep_id</code>	部门 id	String	默认值为 1, 此参数设 置区域管理有效
<code>ladder_control1</code>	梯控权限 1	String	默认值为 0, 值为 0 -FFFFFFFF, 梯控权限 1-32 层
<code>ladder_control2</code>	梯控权限 2	String	默认值为 0, 值为 0-FFFFFFFF, 梯控权限 33-64 层
<code>ladder_control3</code>	梯控权限 3	String	默认值为 0, 值为 0-FFFFFFFF, 梯控权限 65-96 层
<code>ladder_control4</code>	梯控权限 4	String	默认值为 0, 值为 0-FFFFFFFF, 梯控权限 97-128 层

返回示例:

```
{
  "msg": "成功",
  "status": 0
}
```

3.2.4 获取指定用户数据

URL: get_userinfo

Method: POST

参数说明:

字段	描述	类型	详细说明
target_ip	目标 ip	String	控制板的 ip
sn	设备 id	String	控制器 id
id	用户序号	String	0-999999, 默认使用该序号查询
card_id	卡号 id	String	序号 id 为 0, 则使用该卡号 id 查询

返回示例:

```
{
  "data": [
    {
      "T_time": "2020-01-01 00:00:00",
      "card": "C8781384",
      "card_type": "0",
      "door": "1111",
      "pw": "999999",
      "time_group": "00000000"
    }
  ]
}
```

```
    ],  
    "msg": "成功",  
    "status": 0  
}
```

3.2.5 获取刷卡记录

URL: get_event

Method: POST

参数说明: 无

返回示例:

```
{  
  "data": [  
    {  
      "ID": "01400105",  
      "card": "C8781384",  
      "event_type": "100",  
      "ip": "192.168.001.141",  
      "time": "2019-06-18 15:15:53",  
      "Message_type": "1"  
    }  
  ],  
  "msg": "成功",  
  "status": 0  
}
```

3.2.6 获取设备信息

URL: get_devinfo

Method: POST

参数说明:

字段

描述

类型

详细说明

target_ip	目标 ip	String	控制板的 ip
sn	设备 id	String	控制器 id

返回示例:

```
{
  "data": [
    {
      "IP": "192.168.001.141",
      "connect_ip": "192.168.001.023",
      "mac": "00:11:22:a3:14:0d",
      "port": "39101",
      "time": "2019-06-18 15:23:36",
      "Compile_time": "",
      "control_type": "4",

      "version": "T1-L1000.40.01"
    }
  ],
  "msg": "成功",
  "status": 0
}
```

3.2.7 获取门状态

URL: get_doorstate

Method: POST

参数说明:

字段	描述	类型	详细说明
target_ip	目标 ip	String	控制板的 ip
door	门地址	String	门号 1-4
sn	设备 id	String	控制器 id

返回示例:

```
{
  "data": [
    {
      "door": "1",
      "state": "3"
    }
  ],
  "msg": "成功",
  "status": 0
}
```

3.2.8 获取设备状态

URL: get_devstate

Method: POST

参数说明:

字段	描述	类型	详细说明
target_ip	目标 ip	String	控制板的 ip
sn	设备 id	String	控制器 id

返回示例:

```
{
  "data": {
    "is_online": 1
  },
  "msg": "成功",
  "status": 0
}
```

3.2.8 获取门点数

URL: get_devdp

Method: POST

参数说明:

字段	描述	类型	详细说明
target_ip	目标 ip	String	控制板的 ip
sn	设备 id	String	控制器 id

返回示例:

```
{
  "data": {
    "door": 2,
    "point": 5
  },
  "msg": "成功",
  "status": 0
}
```

3.2.9 获取门参

URL: get_doorpart

Method: POST

参数说明:

字段	描述	类型	详细说明
target_ip	目标 ip	String	控制板的 ip
door	门地址	String	门号 1-4
sn	设备 id	String	控制器 id

返回示例:

```
{
  "data": [
    {
      "Acard_num": "0",
      "anti_tail": "0",
      "alarm_time": "30",
      "card_mode": "0",
      "card_num": "1",
      "card_time": "0",
      "coer_code": "9999",
      "d_type": "0",
      "door": "1",
      "door_id": "014001",
      "door_name": "001601",
      "door_time": "5",
      "first_card": "0",
      "relay_time": "5",
      "super_pw": "15963248",
      "two_door": "0",
      "muil_door": "0"
    }
  ],
  "msg": "成功",
  "status": 0
}
```

3.3.0 设置门参

URL: set_doorpart

Method: POST

参数说明:

字段	描述	类型	详细说明
target_ip	目标 ip	String	控制板的 ip
door	门地址	String	门号 1-4
sn	设备 id	String	控制器 id

door_name	门名字	String	默认为 1Door
two_door	双向门标识	String	默认为 0, 0-无效, 1-有效
d_type	读卡器输入类型	String	默认为 0, 0-自动检测, 1-韦根 26, 2-韦根 32
card_time	读卡器扫描时间	String	默认为 0, 范围 0-255
door_time	门开保持时间	String	默认为 5, 范围 1-255
relay_time	辅助继电器动作时间	String	默认为 5, 范围 1-255
alarm_time	门开报警时间	String	默认为 30, 范围 0-255
super_pw	超级密码	String	默认为 15963248, 范围 0-99999999
coer_code	胁迫密码	String	默认为 9999 范围 0-9999
card_num	多卡开门普卡数量	String	默认为 1, 范围 1-64
Acard_num	多卡开门 A 卡数量	String	默认为 0, 范围 1-64
first_card	首卡开门	String	默认为 0, 0-无效, 1-有效

card_mode	卡号处理模式	String	默认为0, 0-设备自动判断, 1-服务器判断
anti_tail	防跟随	String	默认为0, 0-无效, 1-有效
muil_door	多门互锁	String	默认为0, 0-无效, 1-有效

返回示例:

```
{
  "msg": "成功",
  "status": 0
}
```

3.3.1 获取点参

URL: get_pointpart

Method: POST

参数说明:

字段	描述	类型	详细说明
target_ip	目标 ip	String	控制板的 ip
sn	设备 id	String	控制器 id
door	门号	String	门号 1-4
point	点号	String	点号 0-4, 0-出门按钮, 1-门磁, 2-辅助输入,

返回示例:

```
{
  "data": [
    {
      "c_state": "1",
      "door": "1",
      "inner_linkage": "0",
      "inner_linkage_act": "0",
      "inner_linkage_act1": "0",
      "inner_linkage_act2": "0",
      "inner_linkage_doorpoint": "0000",
      "inner_linkage_doorpoint1": "0000",
      "inner_linkage_doorpoint2": "0000",
      "out_linkage": "0",
      "out_linkage_group": "0",
      "p_put": "0",
      "p_vaild": "1",
      "point": "1",
      "point_id": "01400101",
      "point_name": "0101_门磁",
      "s_type": "0"
    }
  ],
  "msg": "成功",
  "status": 0
}
```

3.3.2 设置点参

URL:set_pointpart

Method: POST

参数说明:

字段	描述	类型	详细说明
target_ip	目标 ip	String	控制板的 ip
door	门地址	String	门号 1-4

sn	设备 id	String	控制器 id
point	点地址	String	点号 0-4
point_name	点名称	String	默认 0100_DName
p_vaild	点是否有 效	String	默认 1, 0-无效, 1-无效
c_state	点常态	String	默认 0, 0-常开, 1-常闭
s_type	消息类型	String	默认 0, 1-普通 消息, 2-报警消 息, 0-不产生消 息
p_put	点输出模 式	String	默认 0, 0-连续 输出, 1 跳动输 出
out_linkage	是否有外 联动	String	默认 0, 0-无外 联动, 1-有外联 动
out_linkage_group	外联动组 号	String	默认 0
inner_linkage	内联动点 触发方式	String	默认 2 0-无触发, 1-关

			闭时触发，2-打开时触发，3-短路时触发，4-开路时触发
<code>inner_linkage_act</code>	1-内联点动作	String	默认 2 0-无效, 1-关闭, 2-开启, 3-持续关闭(输出), 4-持续打开(输出), 5-自动
<code>inner_linkage_doorpoint</code>	1-内联门号点号	String	默认 0103, 门号 00-无效, 01-1门, 02-2门; 点号 00-出门按钮, 01-门磁, 02-辅助输入, 03-主输出, 04-辅助输出, 05-读卡器, 06-密码键盘, 07-掉电保护 检测点
<code>inner_linkage_act2</code>	2-内联	String	默认 0, 同上

	点动作		
inner_linkage_doorpoint2	2-内联动 门号点号	String	默认 0000, 同上
inner_linkage_act3	3-内联动 点动作	String	默认 0, 同上
inner_linkage_doorpoint3	3-内联动 门号点号	String	默认 0000, 同上
inner_linkage_act4	4-内联动 点动作	String	默认 0, 同上
inner_linkage_doorpoint4	4-内联动 门号点号	String	默认 0000, 同上

返回示例:

```
{
  "msg": "成功",
  "status": 0
}
```

3.3.3 设置门状态

URL:set_doorstate

Method: POST

参数说明:

字段	描述	类型	详细说明
target_ip	目标 ip	String	控制板的 ip

Method: POST

参数说明:

字段	描述	类型	详细说明
target_ip	目标 ip	String	控制板的 ip
sn	设备 id	String	控制器 id

返回示例:

```
{
  "data":
  [
    {
      "day1": "0",
      "day2": "0",
      "day3": "0",
      "day4": "0",
      "day5": "0",
      "day6": "0",
      "day7": "0"
    }
  ]
},
"msg": "成功",
"status": 0
}
```

3.3.7 设置临时时间表

URL:set_temptime

Method: POST

参数说明:

字段	描述	类型	详细说明
target_ip	目标 ip	String	控制板的 ip
sn	设备 id	String	控制器 id

day	星期	String	默认 0 0 0 0 0 0 0， 顺序为星期日-星期六 0-无效，1 有效
------------	----	--------	--

返回示例：

```
{
  "msg": "成功",
  "status": 0
}
```

3.3.8 获取门状态时间组

URL: get_doorstatetime

Method: POST

参数说明：

字段	描述	类型	详细说明
target_ip	目标 ip	String	控制板的 ip
sn	设备 id	String	控制器 id
door	门地址	String	门号 1-4

返回示例：

```
{
  "data": [
    {
      "S0-S6": "0000000",
      "T0-T1": "00",
      "W0-W1": "0000000",
      "act_time_type1": "07003",
      "act_time_type10": "00000",
    }
  ]
}
```

```

    "act_time_type11": "00000",
    "act_time_type12": "00000",
    "act_time_type13": "00000",
    "act_time_type14": "00000",
    "act_time_type15": "00000",
    "act_time_type16": "00000",
    "act_time_type2": "00000",
    "act_time_type3": "00000",
    "act_time_type4": "00000",
    "act_time_type5": "00000",
    "act_time_type6": "00000",
    "act_time_type7": "00000",
    "act_time_type8": "00000",
    "act_time_type9": "00000",
    "door": "1"
  }
],
"msg": "成功",
"status": 0

```

3.3.9 设置门状态时间组

URL: set_doorstatetime

Method: POST

参数说明:

字段	描述	类型	详细说明
target_ip	目标 ip	String	控制板的 ip
sn	设备 id	String	控制器 id
door	门地址	String	门号 1-4
Temporary_time	临时时间组	String	默认 0 0 共 2 组, 0-无效, 1-有效
Special_time	特殊日期组	String	默认 0 0 0 0 0 0 0 0,

			共 7 组， 0-无效，1-有效
Ordinary_time	普通时间组	String	默认 0 0 0 0 0 0 0， 共 7 组， 顺序周日-周 6 0-无效，1-有效
time_group1	时间计划组 1	String	默认 0700 3，前 4 位代 表时间 7 点，后一位代 表门状态
time_group2	时间计划组 2	String	默认 0000 0，同上
time_group3	时间计划组 3	String	默认 0000 0，同上
time_group4	时间计划组 4	String	默认 0000 0，同上
time_group5	时间计划组 5	String	默认 0000 0，同上
time_group6	时间计划组 6	String	默认 0000 0，同上
time_group7	时间计划组 7	String	默认 0000 0，同上
time_group8	时间计划组 8	String	默认 0000 0，同上
time_group9	时间计划组 9	String	默认 0000 0，同上
time_group10	时间计划组 10	String	默认 0000 0，同上
time_group11	时间计划组 11	String	默认 0000 0，同上

time_group12	时间计划组 12	String	默认 0000 0, 同上
time_group13	时间计划组 13	String	默认 0000 0, 同上
time_group14	时间计划组 14	String	默认 0000 0, 同上
time_group15	时间计划组 15	String	默认 0000 0, 同上
time_group16	时间计划组 16	String	默认 0000 0, 同上

返回示例:

```
{
  "msg": "成功",
  "status": 0
}
```

3.4.0 获取门禁时间组

URL: get_doortimegroup

Method: POST

参数说明:

字段	描述	类型	详细说明
target_ip	目标 ip	String	控制板的 ip
sn	设备 id	String	控制器 id

door	门号	String	门号 1-4
time_group_num	组号	String	组号 0-15

返回示例:

```
{
  "data": [
    {
      "S0-S6": "0000000",
      "T0-T1": "00",
      "W0-W1": "1111111",
      "door": "1",
      "time_group1": "00002359",
      "time_group2": "00000000",
      "time_group3": "00000000",
      "time_group4": "00000000",
      "time_group5": "00000000",
      "time_group6": "00000000",
      "time_group7": "00000000",
      "time_group_num": "0"
    }
  ],
  "msg": "成功",
  "status": 0
}
```

3.4.1 设置门禁时间组

URL: set_doortimegroup

Method: POST

参数说明:

字段	描述	类型	详细说明
target_ip	目标 ip	String	控制板的 ip
sn	设备 id	String	控制器 id
door	门号	String	门号 1-4

group	组号	String	时间组号 0-15
Temporary_time	临时时间组	String	默认 0 0, 共 2 组, 0-无效, 1-有效
Special_time	特殊日期组	String	默认 0 0 0 0 0 0 0 0, 共 7 组, 0-无效, 1-有效
Ordinary_time	普通日期组	String	默认 0 0 0 0 0 0 0 0, 共 7 组, 顺序周日-周 6 0-无效, 1-有效
time_group1	时间计划组 1	String	默认 00002359, 起始 时间和结束时间 00:00-2359
time_group2	时间计划组 2	String	默认 00000000, 同上
time_group3	时间计划组 3	String	默认 00000000, 同上
time_group4	时间计划组 4	String	默认 00000000, 同上
time_group5	时间计划组 5	String	默认 00000000, 同上
time_group6	时间计划组 6	String	默认 00000000, 同上
time_group7	时间计划组 7	String	默认 00000000, 同上

返回示例:

```
{  
  "msg": "成功",  
  "status": 0  
}
```

3.4.2 出厂还原设置

URL: factory_restore_settings

Method: POST

参数说明:

字段	描述	类型	详细说明
target_ip	目标 ip	String	控制板的 ip
sn	设备 id	String	控制器 id
grade	还原等级	String	0-出厂还原保留设备 id, 有效期, 1-在 0 基础上保留网络参数, 不包括

返回示例:

```
{  
  "msg": "成功",  
  "status": 0  
}
```

3.4.3 外联动

URL: outlink_status

Method: POST

参数说明:

字段	描述	类型	详细说明
target_ip	目标 ip	String	控制板的 ip
sn	设备 id	String	控制器 id
door	门地址	String	门号 1-4
point	点地址	String	点号 0-4
status	点状态	String	0-无效, 1-关闭, 2-开启, 3-短路/持续关闭, 4-开路/持续打开, 5-自动

返回示例:

```
{  
  "msg": "成功",  
  "status": 0  
}
```

3.4.4 设置点状态

URL: point_status

Method: POST

参数说明:

字段	描述	类型	详细说明
target_ip	目标 ip	String	控制板的 ip
sn	设备 id	String	控制器 id
door	门地址	String	门号 1-4
point	点地址	String	门点 0-4
status	点状态	String	0-无效, 1-关闭, 2-开启, 3-短路/持续关闭, 4-开路/持续打开, 5-自动

返回示例:

```
{  
  "msg": "成功",
```

```
    "status": 0
}
```

3.4.5 获取点状态

URL: get_point_status

Method: POST

参数说明:

字段	描述	类型	详细说明
target_ip	目标 ip	String	控制板的 ip
sn	设备 id	String	控制器 id
door	门地址	String	门号 1-4
point	点地址	String	门点 0-4, 0: 出门按钮; 1: 门磁; 2 辅助输入; 3: 门锁; 4: 辅助门锁

返回示例:

```
{
  "data": [
    {
      "door": "1",
```

```
        "point": "1",
        "state": "2"
    }
],
"msg": "成功",
"status": 0
}
```

3.4.6 更换控制器类型

URL: control_type

Method: POST

参数说明:

字段	描述	类型	详细说明
target_ip	目标 ip	String	控制板的 ip
sn	设备 id	String	控制器 id
con_type	控制器类型	String	0-1, 0 代表门禁, 1 代表梯控

返回示例:

```
{
  "msg": "成功",
  "status": 0
}
```

3.4.7 上传注册文件

URL: downtemp

Method: POST-FILE

参数说明:

字段	描述	类型	详细说明
file	上传文件	lic 文件	注册控制器

返回示例:

```
{  
  "msg": "成功",  
  "status": 0  
}
```

3.4.8 设置用户密码

URL: update_user

Method: POST

参数说明:

字段	描述	类型	详细说明
username	用户	String	用户长度不超过 10
password	密码	String	密码长度不超过 10

返回示例:

```
{
  "msg": "成功",
  "status": 0
}
```

3.4.9 获取所有在线控制器 ip, sn

URL: get_allcontrol

Method: POST

参数说明: 无

返回示例:

```
{
  "data": {
    "ip_list": [
      [
        "172.24.50.20",
        "05D23ABB0000EE3DD"
      ],
      [
        "172.24.50.10",
        "05D23ABA000061461"
      ]
    ],
    "ip_num": 2
  },
  "msg": "成功",
  "status": 0
}
```

3.5.0 查询卡号对应的人员

URL: get_user_event

Method: POST

参数说明:

字段	描述	类型	详细说明
card_num	卡号	String	卡号

返回示例:

```
{
  "data": [
    {
      "CardID": 2,
      "CardNumber": "951",
      "DeptID": 1,
      "DeptName": "新部门 1",
      "IsUsed": 1,
      "Photo": "20200921110011.png",
      "UserID": 1,
      "UserName": "mcy"
    }
  ],
  "msg": "成功",
  "status": 0
}
```

3.5.1 查询全部卡号对应的人员

URL: all_user_event

Method: POST

参数说明:

无

返回示例:

```
{
  "data": [
    {
      "CardID": 2,
      "CardNumber": "951",
      "DeptID": 1,
      "DeptName": "新部门 1",
      "IsUsed": 1,
      "Photo": "20200921110011.png",
      "UserID": 1,
      "UserName": "mcy"
    },
    {
      "CardID": 3,
      "CardNumber": "9999",
      "DeptID": 1,
      "DeptName": "新部门 1",
      "IsUsed": 1,
      "Photo": "20200921110011.png",
      "UserID": 1,
      "UserName": "mcy"
    },
    {
      "CardID": 4,
      "CardNumber": "666",
      "DeptID": 1,
      "DeptName": "新部门 1",
      "IsUsed": 1,
      "Photo": "20200921110011.png",
      "UserID": 1,
      "UserName": "mcy"
    },
  ],
}
```

```

    {
      "CardID": 7,
      "CardNumber": "99999",
      "DeptID": 1,
      "DeptName": "新部门 1",
      "IsUsed": 1,
      "Photo": "20200921141050.png",
      "UserID": 2,
      "UserName": "对对对"
    },
    {
      "CardID": 8,
      "CardNumber": "74421",
      "DeptID": 1,
      "DeptName": "新部门 1",
      "IsUsed": 1,
      "Photo": "20200921141050.png",
      "UserID": 2,
      "UserName": "对对对"
    },
    {
      "CardID": 9,
      "CardNumber": "331111",
      "DeptID": 1,
      "DeptName": "新部门 1",
      "IsUsed": 1,
      "Photo": "20200921141050.png",
      "UserID": 2,
      "UserName": "对对对"
    }
  ],
  "msg": "成功",
  "status": 0
}

```

附：

特殊日期组，临时时间组，普通日期组：不同日期组在不同时间计划组的计算规则按 2 的幂数。时间计划组 1 的特殊日期组 1 == 2 的 0 次幂以此类推 特殊日期组的值==时间计划组 1 的

特殊日期组 1 +时间计划组 2 的特殊日期组 2 + ...

单独特殊日期组计算规则：s1-s7 按 2 的 0-6 次幂计算，如同
时存在不同日期组 2 者值相加

附：

POST 类型 参数输入全部采用 FORMDATA 格式

附录：

1. 门状态表

ActType	Content
0	无效
1	睡眠
2	常开
3	安全
4	密码
5	APB
6	APB密码
7	自动

2. 事件 id 表

EventType	EventContent
0	系统热复位报警
1	系统冷复位报警
2	外部电源掉电
3	外部电源恢复正常
4	控制箱打开
5	控制箱关闭
16	出门按钮短路
17	出门按钮断路
18	出门按钮开启动作
19	出门按钮关闭动作
32	门(门磁)短路
33	门(门磁)断路
34	门(门磁)被正常打开
35	门(门磁)异常打开
36	门(门磁)正常关闭
37	门(门磁)异常关闭
38	门(门磁)开超时
39	门(门磁)未打开
48	主辅助输入短路
49	主辅助输入断路
50	主辅助输入开启动作
51	主辅助输入关闭动作
64	子辅助输入短路
65	子辅助输入断路
66	子辅助输入开启动作
67	子辅助输入关闭动作
80	有效输出点开启动作
81	有效输出点关闭动作
82	有效输出点持续开启动作
83	有效输出点持续关闭动作
96	非法卡进门
97	非法卡出门(双向门)
98	无效卡进门
99	无效卡出门(双向门)
100	有效卡进门
101	有效卡出门(双向门)

102	有效卡无效时间段进门	
103	有效卡无效时间段出门(双向)	
104	有效卡无效密码进门	
105	有效卡无效密码出门(双向)	
106	有效卡无效门点进门	
107	有效卡无效门点出门(双向)	
108	跟随进门	
109	跟随出门	
110	无效区域进入	
111	无效区域退出	
112	胁迫进门	
113	胁迫出门(双向)	
114	有效超级密码进门	
115	有效超级密码出门(双向)	
116	无效超级密码进门	
117	无效超级密码出门(双向)	
118	休眠模式刷卡进门	
119	休眠模式刷卡出门	
120	休眠模式密码进门	
121	休眠模式密码出门	
122	常开模式刷卡进门	
123	常开模式刷卡出门	
124	常开模式密码进门	
125	常开模式密码出门	
126	随机密码进入	
127	随机密码出门	
128	产生外联动	
129	门状态变化	
130	远程开门	
253	无信息	
254	错误信息	
255	错误信息	

3. 控制器状态表

0-离线，1-上线

4. 服务器存储事件表

SetID	SetContent
1	系统消息
2	错误消息
3	警告消息
4	普通消息

5. 点状态 id 表

PointActType	PointActTypeContent
0	无效
1	关闭
2	开启
3	短路
4	开路
5	自动