LifeSmart 设备规格属性说明 (v2.27)

版本	修订日期	修订人	修订内容
1.0	2017/06/21	LuLu	
2.0	2017/10/12	Pretty Ju	完善属性说明并重新整理
2.01	2017/12/26	Jon Fan	增加：窗帘，甲醛报警器
2.02	2018/02/10	Jon Fan	增加：智能窗帘电机，修改温控器系列，通用控制器，灯光产品属性说明
2.03	2018/03/30	Jon Fan	增加：开关智控器SL_S，超级碗CoSS版 SL_SPOT，红外模块SL_P_IR
2.04	2018/07/06	Jon Fan	增加：SL_SPWM 可调亮度开关智控器
2.10	2018/09/06	Jon Fan	重新梳理了排版。 增加：OD_WE_QUAN量子灯； SL_SC_CP 气体感应器(燃气)； SL_SC_CH 气体感应器(甲醛)； 白玉/墨玉开关系列； 恒星开关、恒星开关伴侣系列；
2.11	2018/11/12	Jon Fan	增加：通用控制器(HA) 以及120调光开关
2.12	2018/11/22	Jon Fan	增加：九路开关控制器(SL_P_SW)， 智能报警器(CoSS版)(SL_ALM)
2.13	2019/01/22	Jon Fan	修改 环境感应器(TVOC+CO2)描述
2.14	2019/02/12	Jon Fan	增加：摄像头属性 修改：Sensor的电压描述
2.15	2019/03/07	Jon Fan	增加SL_OE_3C计量插座, SL_LI_WW白光智能灯泡, SL_SW_MJ1,SL_SW_MJ2奇点开关模块,
2.16	2019/04/07	Jon Fan	增加SL_P_V2智界窗帘电机智控器
2.16	2019/04/09	Jon Fan	增加 SL_UACCB空调控制面板
2.17	2019/06/01	Jon Fan	增加IO值浮点类型说明 增加调光开关指示灯说明
2.18	2019/07/12	Jon Fan	增加SL_P_A烟雾感应器, SL_SC_CA环境感应器, SL_P_RM 人体存在感应器
2.19	2019/08/30	Jon Fan	增加SL_SW_MJ3
2.20	2019/11/06	Jon Fan	增加SL_LI_GD1调光壁灯，SL_LI_UG1花园地灯
2.21	2019/11/13	Jon Fan	增加SL_CP_VL 温控阀门 增强感应器系列产品的电量/充电电量描述

2.22	2020/10/29	Jon Fan	增加开关、插座类设备dark、bright属性说明; 增加SL_SW_NS[1|2|3] 星玉开关 SL_SW_RC 极星开关(零火版) SL_NATURE 超能面板 SL_P_IR 超级碗(Mini版) DLT电量计量
2.23	2021/02/19	Jon Fan	增加通用控制器以及通用控制器(HA)描述 增加云防DEFED系列产品属性描述
2.24	2021/10/15	Jon Fan	增加艾弗纳KV11新风V_FRESH_P
2.25	2022/01/16	Jon Fan	增加超能面板系列与星玉温控面板
2.26	2022/04/07	Jon Fan	增加SL_SC_CN噪音感应器
2.27	2022/06/30	Pretty Ju	补充V_485_P描述; 更新《1.1 设备模型说明》; 增加《附录 3.5 IO实际值转换及Type定义说明》

目录

1 说明

该文档主要描述LifeSmart管理的设备的规格属性的说明。具体使用请结合**《LifeSmart云平台服务接口》**文档。

1.1 设备模型说明

为了方便阐述，这里我们对LifeSmart的设备模型做个简要的说明，对一些术语作出说明。
我们以EpGet请求获取的数据为例：

```
{
  "name": "Smart Switch",
  "agt": "A3EAAABtAEwQRzM0Njg5NA",
  "me": "2d11",
  "devtype": "SL_SW_IF3",
  "fullCls": "SL_SW_IF3_V2",
  "stat": 1,
  "data": {
    "L1": {"type": 129, "val": 1,"name": "Living", "v": 1, ...},
    "L2": {"type": 128, "val": 0,"name": "Study", "v": 0, ...},
    "L3": {"type": 129, "val": 1,"name": "Kid", "v": 1, ...},
  },
}
```

其中：

- **智慧中心(Agt)**：　*"A3EAAABtAEwQRzM0Njg5NA"*
- **设备(EP)**：　*"2d11"*，它是一个 *SL_SW_IF3* 类型的三联开关
- **devtype**：　对应表格中***Devtype/Cls***，它描述的是设备的类型，规格名称；
- **fullCls**：　包含版本号的完整的设备类型，一般它的值等于*devtype+V[n]*，*V[n]*指明其版本号信息，一般情况下使用*devtype*即可标识设备类型，在需要区分设备不同版本的特性的时候才需要用到*fullCls*，如果设备类型没有版本信息，则*fullCls*可能与*devtype*相同；
- **设备属性*(IO口)*：**　设备的属性，可以用于读取状态，控制行为，*L1*、*L2*、*L3*它们都是设备的*IO*口，当然对于只读的*IO*口例如温度传感器，则只能读取状态，不能控制。它对应表格中***IO idx，IO名称***。后面的**属性值描述**与**下发命令描述**也是对*IO*口的说明。
- **设备属性值*(IO值)*：**　设备的属性值，可以被解析为对应设备该*IO*的状态值。需要注意的是*IO*属性值中的***val***表示设备的原始数据值，实际生活中的值可能为不同精度的浮点值或其他，我们在展示给用户的时候更多的需要使用实际值。一般接口会在*IO*属性值中的***v***表示该设备属性的实际值，若接口为返回则有可能需要在使用时进行手动转换，可参考 **附录3.5 IO实际值转换及Type定义说明**。

2 智慧设备

2.1 插座系列（Outlet Series）

2.1.1 传统插座系列

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

SL_OL
SL_OL_3C
SL_OL_DE
SL_OL_UK
SL_OL_UL
	O	开关	type%2==1,表示打开(忽略val值);
			
			type%2==0,表示关闭(忽略val值);
				RW	打开，则下发:
					type=0x81,val=1;
					
					关闭，则下发:
					type=0x80,val=0;

OD_WE_OT1	P1	开关	type%2==1,表示打开(忽略val值);
			
			type%2==0,表示关闭(忽略val值);
				RW	打开，则下发:
					type=0x81,val=1;
					
					关闭，则下发:
					type=0x80,val=0;

2.1.2 计量插座系列

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

	P1	开关	type%2==1,表示打开(忽略val值);
			
			type%2==0,表示关闭(忽略val值);
				RW	打开，则下发:
					type=0x81,val=1;
					
					关闭，则下发:
					type=0x80,val=0;

	P2	用电量	为累计用电量， val值为为IEEE 754浮点数的32位整数表示， v值为浮点数，单位为度(kwh)。	R	注意：v值可以直接使用，若不存在v值，则需要手动转换。 其值类型为IEEE 754浮点数的32位整数布局。 例如 1024913643 表示的是浮点值：0.03685085，可以依据整数值计算出浮点值。 具体请参考: **附录3.4 IO值浮点类型说明**

SL_OE_3C SL_OE_DE SL_OE_W	P3	功率	为当前负载功率， val值为为IEEE 754浮点数的32位整数表示， v值为浮点数，单位为w。	R	注意：v值可以直接使用，若不存在v值，则需要手动转换。 其值类型为IEEE 754浮点数的32位整数布局。 例如 1024913643 表示的是浮点值：0.03685085，可以依据整数值计算出浮点值。 具体请参考： **附录3.4 IO值浮点类型说明**

	P4	功率门限	功率门限，用电保护，值为整数，超出门限则会断电，单位为w。	RW	使能，则下发：
					type=0x81,val=1;
					不使能，则下发：
					type=0x80,val=0;
					使能并且设置门限，则下发：
					type=207；val=门限值；
					不使能并且设置门限，则下发：
					type=206；val=门限值；

2.2 开关系列（Switch Series）

2.2.1 传统开关系列

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

	L1,L2,L3	开关	type%2==1,表示打开(忽略val值)；
			type%2==0,表示关闭(忽略val值)；
			L1表示第一路开关控制口；
			L2表示第二路开关控制口；
			L3表示第三路开关控制口；
				RW	打开，则下发：
					type=0x81,val=1;
					关闭，则下发：
					type=0x80,val=0;

SL_SF_RC SL_SW_RC	dark	关状时示亮 态指灯 度	type%2=1 表示打开； type%2=0 表示关闭； val表示指示灯亮度值，取值范围：0～1023	RW	开灯： type=0x81；val=1； 关灯： type=0x80；val=0； 开灯并设置亮度值： type=0xff；val=亮度值； 开灯并设置亮度值： type=0xfe；val=亮度值；
	bright	开状时示亮 态指灯 度	type%2=1 表示打开； type%2=0 表示关闭； val表示指示灯亮度值，取值范围：0～1023	RW	开灯： type=0x81；val=1； 关灯： type=0x80；val=0； 开灯并设置亮度值： type=0xff；val=亮度值； 开灯并设置亮度值： type=0xfe；val=亮度值；
SL_SW_IF3 SL_SF_IF3 SL_SW_CP3 SL_SW_RC3	L1,L2,L3	开关	type%2==1,表示打开(忽略val值)； type%2==0,表示关闭(忽略val值)； L1表示第一路开关控制口； L2表示第二路开关控制口； L3表示第三路开关控制口；	RW	打开，则下发： type=0x81,val=1； 关闭，则下发： type=0x80,val=0；
	dark1, dark2, dark3	关状时示亮 态指灯 度	type%2=1 表示打开； type%2=0 表示关闭； val表示指示灯亮度值，定义如下： bit0~bit7:Blue bit8~bit15: Green bit16~bit23:Red bit24~bit31:white （当White>0时，表示动态模式）具体动态值请参考：**附录3.1 动态颜色（DYN）定义**	RW	开灯： type=0x81；val=1； 关灯： type=0x80；val=0； 开灯并设置颜色或动态值： type=0xff；val=颜色或动态值； 关灯并设置颜色值或动态值： type=0xfe；val=颜色或动态值；

	bright1,
	bright2,
	bright3
		开状时示亮
		态指灯
		度
			type%2=1 表示打开;
			type%2=0 表示关闭;
			val表示指示灯亮度值，定义如下:
			bit0~bit7:Blue
			bit8~bit15: Green
			bit16~bit23:Red
			bit24~bit31:white
			（当White>0时，表示动态模式）具体动态值请参考：**附录 3.1 动态颜色（DYN）定义**
				RW	开灯:
					type=0x81; val=1;
					关灯:
					type=0x80; val=0;
					开灯并设置颜色或动态值:
					type=0xff; val=颜色或动态值;
					关灯并设置颜色值或动态值:
					type=0xfe; val=颜色或动态值;

SL_SW_IF2
SL_SF_IF2
SL_SW_CP2
SL_SW_FE2
SL_SW_RC2
	L1,L2	开关	type%2==1,表示打开(忽略val值);
			type%2==0,表示关闭(忽略val值);
			L1表示第一路开关控制口;
			L2表示第二路开关控制口;
				RW	打开，则下发:
					type=0x81,val=1;
					关闭，则下发:
					type=0x80,val=0;

	dark1,
	dark2
		关状时示亮
		态指灯
		度
			type%2=1 表示打开;
			type%2=0 表示关闭;
			val表示指示灯亮度值，定义如下:
			bit0~bit7:Blue
			bit8~bit15: Green
			bit16~bit23:Red
			bit24~bit31:white
			（当White>0时，表示动态模式）具体动态值请参考：**附录 3.1 动态颜色（DYN）定义**
				RW	开灯:
					type=0x81; val=1;
					关灯:
					type=0x80; val=0;
					开灯并设置颜色或动态值:
					type=0xff; val=颜色或动态值;
					关灯并设置颜色值或动态值:
					type=0xfe; val=颜色或动态值;

	bright1,
	bright2
		开状时示亮
		态指灯
		度
			type%2=1 表示打开;
			type%2=0 表示关闭;
			val表示指示灯亮度值，定义如下:
			bit0~bit7:Blue
			bit8~bit15: Green
			bit16~bit23:Red
			bit24~bit31:white
			（当White>0时，表示动态模式）具体动态值请参考：**附录 3.1 动态颜色（DYN）定义**
				RW	开灯:
					type=0x81; val=1;
					关灯:
					type=0x80; val=0;
					开灯并设置颜色或动态值:
					type=0xff; val=颜色或动态值;
					关灯并设置颜色值或动态值:
					type=0xfe; val=颜色或动态值;

SL_SW_IF1
SL_SF_IF1
SL_SW_CP1
SL_SW_FE1
SL_OL_W
SL_SW_RC1
	L1	开关	type%2==1,表示打开(忽略val值)；
			type%2==0,表示关闭(忽略val值)；
			L1表示第一路开关控制口；
				RW	打开，则下发:
					type=0x81,val=1;
					
					关闭，则下发:
					type=0x80,val=0;

	dark	关状态时指示灯亮度	type%2=1 表示打开；
			type%2=0 表示关闭；
			val表示指示灯亮度值，定义如下:
			bit0~bit7:Blue
			bit8~bit15: Green
			bit16~bit23:Red
			bit24~bit31:white
			（当White>0时，表示动态模式）具体动态值请参考：**附录3.1 动态颜色（DYN）定义**
				RW	开灯:
					type=0x81; val=1;
					
					关灯:
					type=0x80; val=0;
					
					开灯并设置颜色或动态值:
					type=0xff; val=颜色或动态值；
					
					关灯并设置颜色值或动态值:
					type=0xfe; val=颜色或动态值；

	bright	开状态时指示灯亮度	type%2=1 表示打开；
			type%2=0 表示关闭；
			val表示指示灯亮度值，定义如下:
			bit0~bit7:Blue
			bit8~bit15: Green
			bit16~bit23:Red
			bit24~bit31:white
			（当White>0时，表示动态模式）具体动态值请参考：**附录3.1 动态颜色（DYN）定义**
				RW	开灯:
					type=0x81; val=1;
					
					关灯:
					type=0x80; val=0;
					
					开灯并设置颜色或动态值:
					type=0xff; val=颜色或动态值；
					
					关灯并设置颜色值或动态值:
					type=0xfe; val=颜色或动态值；

2.2.2 恒星/辰星/极星开关系列

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

SL_SW_ND3
SL_MC_ND3
	P1,P2,P3	开关	type%2==1,表示打开(忽略val值)；
			type%2==0,表示关闭(忽略val值)；
			P1表示第一路开关控制口；
			P2表示第二路开关控制口；
			P3表示第三路开关控制口；
				RW	打开，则下发:
					type=0x81,val=1;
					
					关闭，则下发:
					type=0x80,val=0;

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
	P4	电量	val值表示原始电压值 v值将表示当前剩余电量百分比，值范围[0,100]，它是根据val电压值换算的。	R	

SL_SW_ND2
SL_MC_ND2
	P1,P2	开关	type%2==1,表示打开(忽略val值)；
			type%2==0,表示关闭(忽略val值)；
			P1表示第一路开关控制口；
			P2表示第二路开关控制口；
				RW	打开，则下发:
					type=0x81,val=1;
					关闭，则下发:
					type=0x80,val=0;

	P3	电量	val值表示原始电压值 v值将表示当前剩余电量百分比，值范围[0,100]，它是根据val电压值换算的。	R	

SL_SW_ND1
SL_MC_ND1
	P1	开关	type%2==1,表示打开(忽略val值)；
			type%2==0,表示关闭(忽略val值)；
			P1表示第一路开关控制口；
				RW	打开，则下发:
					type=0x81,val=1;
					关闭，则下发:
					type=0x80,val=0;

	P2	电量	val值表示原始电压值 v值将表示当前剩余电量百分比，值范围[0,100]，它是根据val电压值换算的。	R	

注意：极星开关系列其fullCls的版本号为V2，例如 SL_SW_ND3_V2 指明其为极星3联开关。

2.2.3 开关控制器系列

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

SL_S	P2	开关		RW	打开，则下发:
					type=0x81,val=1;
					关闭，则下发:
					type=0x80,val=0;

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

SL_SPWM	P1	开关	type%2==1表示处于打开状态；
			type%2==0表示处于关闭状态；
			
			val值为亮度值，可调范围：[0,255]
			值越大表示光越亮，0处于最暗，光完全熄灭，255处于最亮。
				RW	打开，则下发：
					type=0x81; val=1;
					
					关闭，则下发：
					type=0x80; val=0;
					
					打开并且设置亮度，则下发：
					type=207; val=亮度值[0,255];
					
					关闭并设置亮度，则下发：
					type=206; val=亮度值[0,255];

SL_P_SW	P1,P2,P3,P4,P5,P6,P7,P8,P9	开关	type%2==1表示处于打开状态；
			type%2==0表示处于关闭状态；
				RW	打开，则下发：
					type=0x81; val=1;
					
					关闭，则下发：
					type=0x80; val=0;

	Note： • SL_P_SW的P1,P2,P3,P4,P5,P6,P7,P8,P9在最新版本，除了支持常规的打开关闭操作，同时也支持开关打开并持续设置的时间后自动关闭的功能，用于模拟遥控器按键按下动作。 设置该功能参数如下： type：0x89 val：其值为32位整数，指示打开持续的时长，单位为100ms。 如：设置P1打开持续1s自动关闭，则下发命令为：type=0x89,val=10 如：设置P1打开持续200ms自动关闭，则下发命令为：type=0x89,val=2 如：设置P1打开持续2.5s自动关闭，则下发命令为：type=0x89,val=25 注意：其设置时长精度为100ms，因此当val=1的时候，其时长可能会小于100ms。 • SL_P_SW的非新版本，下发以上命令不生效该打开延时关闭功能。				

2.2.4 随心开关（CUBE Clicker）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
SL_SC_BB	V	电量	val值表示原始电压值 v值将表示当前剩余电量百分比，值范围[0,100]，它是根据val电压值换算的。	R	

	B	按键状态	val的值定义如下： 0： 未按下按键 1： 按下按键	R	

2.2.5 动态调光开关（Dimmer Motion Controller）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

SL_SW_DM1	P1	开关	type%2==1表示处于打开状态；
			type%2==0表示处于关闭状态；
			
			val值为亮度值，可调范围：[0,255]
			值越大表示光越亮，0处于最暗，光完全熄灭，255处于最亮。
				RW	打开，则下发：
					type=0x81; val=1;
					
					关闭，则下发：
					type=0x80; val=0;
					
					打开并且设置亮度，则下发：
					type=207; val=亮度值[0,255];
					
					关闭并设置亮度，则下发：
					type=206; val=亮度值[0,255];

	P2	指示灯	type%2==1表示处于打开状态；
			type%2==0表示处于关闭状态；
			
			val值为亮度值，它有灯光处于打开状态下的指示灯亮度和灯光处于关闭状态下的指示灯亮度。
			
			bit8-bit15：用于指示灯光处于打开状态下的指示灯亮度
			
			bit0-bit7：用于指示灯光处于关闭状态下的指示灯亮度
			
			每8个bit可调范围：[0,255]
			值越大表示光越亮，0处于最暗，光完全熄灭，255处于最亮。
				RW	打开，则下发：
					type=0x81; val=1;
					
					关闭，则下发：
					type=0x80; val=0;
					
					设置亮度，则下发：
					type=223; val=亮度值[0,65535]；例如设置灯光开启状态亮度为50，灯光关闭状态亮度为25，则下发参数如下：type=223; val=12825
					
					注意： 如果是物理触发控制的(即直接在物理开关上触发)则指示灯会有5秒的响应物理按键触发时间，过了5秒才会反应真实的设置。如果是远端触发(通过API或App)则会立即反应真实的设置。

	P3	移动检测	val值定义如下： 0： 没有检测到移动 1:有检测到移动	R	
	P4	环境光照	val值表示原始光照值(单位：lux)	R	
	P5	调光设置	当前调光设置值	RW	具体参见下方描述

	P6	动态设置	当前动态设置值	RW	具体参见下方描述

- **P5 IO描述**

P5 IO用于控制调光设置。

其值为32bit整数，每位bit定义如下：

- bit0-1：定义调光类型
  - 00：自动模式(Auto)
  - 01：50Hz频率
  - 10：60Hz频率
  - 11：禁用调光(Binary)，调光直接是开关切换
- bit2-3：定义调光速度
  - 00：8x(8倍)
  - 01：4x(4倍)
  - 10：2x(2倍)
  - 11：1x(1倍)
- bit4-5：定义灵敏度
  - 00：高(High)
  - 01：中高(Middle-High)
  - 10：中低(Middle-Low)
  - 11：低(Low)
- ~~bit6-7：旋钮使用开关~~

~~若不使能则开关处于关闭状态下，旋钮操作灯不会开启，开关处于开启状态下，旋钮操作灯不会关闭。~~

~~00：使能~~

~~01：不使能~~

- bit6-7 旋钮使能开关

建议一直使能开关，即值为"00"，否则会影响用户体验。

- bit8-15：物理按键开灯时默认亮度

取值范围[0,255]

为0时，代表缺省亮度，其值等于64。

- bit8：保持物理按键默认亮度

等于1表示 每次灯处于关闭状态，物理按键触发打开灯的亮度都会调整为"物理按键开灯时默认亮度"；

等于0表示 每次灯处于关闭状态，物理按键触发打开灯的亮度将比较设定的亮度值是否小于"物理按键开灯时默认亮度"，若小于则调整为"物理按键开灯时默认亮度"，否则为设定的亮度值。

注意：bit8有特殊意义，所以在设置bit8-bit15的时候，需要考虑合并最低位bit8的值。

- bit16-23：调光最小亮度

取值范围[0,255]

- bit24-31：调光最大亮度

取值范围[0,255]

注意：
调光最小亮度与调光最大亮度是用于当控制的灯具的可调光范围较小时，锁定范围的，例如灯具调光范围，最小只能到达50，最大只能到达150，则可以设置最小调光亮度50，最大调光亮度150。让调光开关在这个范围区间调节亮度。

- **P6 IO描述**
  - bit0-7：动态锁定时间

    动态锁定时间指检查到动态之后，这个时间段内有动态这个状态都不会消失，只有过了动态锁定时间之后动态状态才会消失，若这个时间段内又侦测到动态则重新开始计时。

    取值范围[0,255]，单位为秒
  - ~~bit8-23：光照度门限~~

    ~~取值范围[0,65535]~~
  - bit24：是否使能 检测到动态联动执行开灯
  - bit25：是否使能 动态感应消失时联动执行关灯
  - bit26：联动开灯时是否保持原先亮度

    等于1表示使能保持原先亮度， 联动执行开灯时：

    若灯处于开启状态，则保持当前亮度；

    若灯处于关闭状态，则开启灯，亮度设置参考P5.bit8设置，

    若P5.bit8的值等于1则亮度调整为"物理按键开灯时默认亮度"，

    若P5.bit8的值等于0则将比较设定的亮度值是否小于"物理按键开灯时默认亮度"，若小于则调整为"物理按键开灯时默认亮度"，否则为设定的亮度值。

    等于0表示不使能原先亮度，联动执行开灯时：

    若灯处于开启状态，则参考P5.bit8设置，

    若P5.bit8的值等于1则亮度调整为"物理按键开灯时默认亮度"，

    若P5.bit8的值等于0则将比较设定的亮度值是否小于"物理按键开灯时默认亮度"，若小于则调整为"物理按键开灯时默认亮度"，否则为设定的亮度值。

    若灯处于关闭状态，则开启灯，亮度为"物理按键开灯时默认亮度"。
  - ~~bit27：是否使能 联动类型为检测到动态+光照小于光照门限值~~

    ~~如果bit27等于1则表示执行联动开灯需要同时满足：检测到动态，并且光照小于门限值。~~

    注意：由于关照度与实际使用环境影响很大，例如有没有拉开窗帘等都会有很大的影响，因此不建议使用这个参数的制定联动的，如果要使用则需要根据实际使用环境精确设置。

注意：
执行EpSet命令修改P5，P6 IO口属性的时候，type取该IO口当前的type值；
其nonVolatile属性需要等于1，否则掉电之后配置将会丢失；

2.2.6 奇点开关模块(CUBE Switch Module)

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

SL_SW_MJ3	P1,P2,P3	开关	type%2==1,表示打开(忽略val值);
			type%2==0,表示关闭(忽略val值);
			P1表示第一路开关控制口;
			P2表示第二路开关控制口;
			P3表示第三路开关控制口;
				RW	打开，则下发:
					type=0x81,val=1;
					
					关闭，则下发:
					type=0x80,val=0;

SL_SW_MJ2	P1,P2	开关	type%2==1,表示打开(忽略val值);
			type%2==0,表示关闭(忽略val值);
			P1表示第一路开关控制口;
			P2表示第二路开关控制口;
				RW	打开，则下发:
					type=0x81,val=1;
					
					关闭，则下发:
					type=0x80,val=0;

SL_SW_MJ1	P1	开关	type%2==1,表示打开(忽略val值);
			type%2==0,表示关闭(忽略val值);
			P1表示第一路开关控制口;
				RW	打开，则下发:
					type=0x81,val=1;
					
					关闭，则下发:
					type=0x80,val=0;

2.2.7 随心按键（CUBE Clicker2）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
SL_SC_BB_V2	P1	按键状态	type的只定义如下: type%2==1,表示有按键事件产生; type%2==0,表示按键事件消失; val值指明按键事件类型，只有在type%2==1才有效，val的值定义如下: 1：单击事件 2：双击事件 255：长按事件	R	
	P2	电量	val值表示原始电压值 v值将表示当前剩余电量百分比，值范围[0,100]，它是根据val电压值换算的。	R	

2.2.8 星玉开关（Nature Switch）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

SL_SW_NS3	L1,L2,L3	开关	type%2==1,表示打开(忽略val值)；
			type%2==0,表示关闭(忽略val值)；
			L1表示第一路开关控制口；
			L2表示第二路开关控制口；
			L3表示第三路开关控制口；
				RW	打开，则下发:
					type=0x81,val=1;
					
					关闭，则下发:
					type=0x80,val=0;

	dark1,
	dark2,
	dark3
		关状态时指示灯亮度	type%2=1 表示打开；
			type%2=0 表示关闭；
			val表示指示灯亮度值，定义如下:
			bit0~bit7:Blue
			bit8~bit15: Green
			bit16~bit23:Red
			bit24~bit31:white
			（当White>0时，表示动态模式）具体动态值请参考：**附录3.1 动态颜色（DYN）定义**
				RW	开灯:
					type=0x81; val=1;
					
					关灯:
					type=0x80; val=0;
					
					开灯并设置颜色或动态值:
					type=0xff; val=颜色或动态值;
					
					关灯并设置颜色值或动态值:
					type=0xfe; val=颜色或动态值;

	bright1,
	bright2,
	bright3
		开状态时指示灯亮度	type%2=1 表示打开；
			type%2=0 表示关闭；
			val表示指示灯亮度值，定义如下:
			bit0~bit7:Blue
			bit8~bit15: Green
			bit16~bit23:Red
			bit24~bit31:white
			（当White>0时，表示动态模式）具体动态值请参考：**附录3.1 动态颜色（DYN）定义**
				RW	开灯:
					type=0x81; val=1;
					
					关灯:
					type=0x80; val=0;
					
					开灯并设置颜色或动态值:
					type=0xff; val=颜色或动态值;
					
					关灯并设置颜色值或动态值:
					type=0xfe; val=颜色或动态值;

	L1,L2	开关	type%2==1,表示打开(忽略val值)；
			type%2==0,表示关闭(忽略val值)；
			L1表示第一路开关控制口；
			L2表示第二路开关控制口；
				RW	打开，则下发:
					type=0x81,val=1;
					
					关闭，则下发:
					type=0x80,val=0;

SL_SW_NS2	dark1,
	dark2
		关状时指示灯亮度	type%2=1 表示打开;
			type%2=0 表示关闭;
			val表示指示灯亮度值，定义如下:
			bit0~bit7:Blue
			bit8~bit15: Green
			bit16~bit23:Red
			bit24~bit31:white
			（当White>0时，表示动态模式）具体动态值请参考：**附录3.1 动态颜色（DYN）定义**
				RW	开灯:
					type=0x81; val=1;
					关灯:
					type=0x80; val=0;
					开灯并设置颜色或动态值:
					type=0xff; val=颜色或动态值;
					关灯并设置颜色值或动态值:
					type=0xfe; val=颜色或动态值;

	bright1,
	bright2
		开状时指示灯亮度	type%2=1 表示打开;
			type%2=0 表示关闭;
			val表示指示灯亮度值，定义如下:
			bit0~bit7:Blue
			bit8~bit15: Green
			bit16~bit23:Red
			bit24~bit31:white
			（当White>0时，表示动态模式）具体动态值请参考：**附录3.1 动态颜色（DYN）定义**
				RW	开灯:
					type=0x81; val=1;
					关灯:
					type=0x80; val=0;
					开灯并设置颜色或动态值:
					type=0xff; val=颜色或动态值;
					关灯并设置颜色值或动态值:
					type=0xfe; val=颜色或动态值;

SL_SW_NS1	L1	开关	type%2==1,表示打开(忽略val值);
			type%2==0,表示关闭(忽略val值);
			L1表示第一路开关控制口;
				RW	打开，则下发:
					type=0x81,val=1;
					关闭，则下发:
					type=0x80,val=0;

	dark	关状时指示灯亮度	type%2=1 表示打开;
			type%2=0 表示关闭;
			val表示指示灯亮度值，定义如下:
			bit0~bit7:Blue
			bit8~bit15: Green
			bit16~bit23:Red
			bit24~bit31:white
			（当White>0时，表示动态模式）具体动态值请参考：**附录3.1 动态颜色（DYN）定义**
				RW	开灯:
					type=0x81; val=1;
					关灯:
					type=0x80; val=0;
					开灯并设置颜色或动态值:
					type=0xff; val=颜色或动态值;
					关灯并设置颜色值或动态值:
					type=0xfe; val=颜色或动态值;

	bright	开状态时指示灯亮度	type%2=1 表示打开； type%2=0 表示关闭； val表示指示灯亮度值，定义如下： bit0~bit7:Blue bit8~bit15: Green bit16~bit23:Red bit24~bit31:white （当White>0时，表示动态模式）具体动态值请参考：**附录3.1 动态颜色（DYN）定义**	RW	开灯： type=0x81；val=1； 关灯： type=0x80；val=0； 开灯并设置颜色或动态值： type=0xff；val=颜色或动态值； 关灯并设置颜色值或动态值： type=0xfe；val=颜色或动态值；

2.2.9 极星开关（零火版）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

SL_SW_RC	L1,L2,L3	开关	type%2==1,表示打开(忽略val值)；
			type%2==0,表示关闭(忽略val值)；
			L1表示第一路开关控制口；
			L2表示第二路开关控制口；
			L3表示第三路开关控制口；
				RW	打开，则下发：
					type=0x81,val=1;
					关闭，则下发：
					type=0x80,val=0;

	dark	关状态时指示灯亮度	type%2=1 表示打开； type%2=0 表示关闭； val表示指示灯亮度值，取值范围：0～1023	RW	开灯： type=0x81； val=1； 关灯： type=0x80； val=0； 开灯并设置亮度值： type=0xff；val=亮度值； 关灯并设置亮度值： type=0xfe；val=亮度值；

	bright	开状态时指示灯亮度	type%2=1 表示打开； type%2=0 表示关闭； val表示指示灯亮度值，取值范围：0～1023	RW	开灯： type=0x81； val=1； 关灯： type=0x80； val=0； 开灯并设置亮度值： type=0xff；val=亮度值； 关灯并设置亮度值： type=0xfe；val=亮度值；

2.2.10 超能面板（Nature Mini/MiniS）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

SL_NATURE	P1,P2,P3	开关	type%2==1,表示打开(忽略val值)；
			type%2==0,表示关闭(忽略val值)；
			P1表示第一路开关控制口；
			P2表示第二路开关控制口；
			P3表示第三路开关控制口；
				RW	打开，则下发：
					type=0x81,val=1;
					
					关闭，则下发：
					type=0x80,val=0;

	P4	热敏电阻数组	可以通过公式换算出温度	R	

2.2.11 极星开关（120零火版）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

SL_SW_BS3	P1,P2,P3	开关	type%2==1,表示打开(忽略val值)；
			type%2==0,表示关闭(忽略val值)；
			P1表示第一路开关控制口；
			P2表示第二路开关控制口；
			P3表示第三路开关控制口；
				RW	打开，则下发：
					type=0x81,val=1;
					
					关闭，则下发：
					type=0x80,val=0;

SL_SW_BS2	P1,P2	开关	type%2==1,表示打开(忽略val值)；
			type%2==0,表示关闭(忽略val值)；
			P1表示第一路开关控制口；
			P2表示第二路开关控制口；
				RW	打开，则下发：
					type=0x81,val=1;
					
					关闭，则下发：
					type=0x80,val=0;

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

SL_SW_BS1	P1	开关	type%2==1,表示打开(忽略val值);
			type%2==0,表示关闭(忽略val值);
			P1表示第一路开关控制口;
				RW	打开，则下发:
					type=0x81,val=1;
					
					关闭，则下发:
					type=0x80,val=0;

2.3 窗帘控制系列（Curtain Controller）

2.3.1 窗帘控制开关

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
SL_SW_WIN	OP	打开窗帘(open)	type%2==1 表示打开窗帘；	RW	执行打开窗帘，则下发：type=0x81,val=1;
	ST	停止(stop)	type%2==1 表示停止当前动作；	RW	执行停止窗帘，则下发：type=0x81,val=1;
	CL	关闭窗帘(close)	type%2==1 表示关闭窗帘；	RW	执行关闭窗帘，则下发：type=0x81,val=1;
	dark	关状态时指示灯亮度	type%2=1 表示打开；type%2=0 表示关闭；val表示指示灯亮度值，取值范围：0～1023	RW	开灯：type=0x81；val=1； 关灯：type=0x80；val=0； 开灯并设置亮度值：type=0xff；val=亮度值； 开灯并设置亮度值：type=0xfe；val=亮度值；
	bright	开状态时指示灯亮度	type%2=1 表示打开；type%2=0 表示关闭；val表示指示灯亮度值，取值范围：0～1023	RW	开灯：type=0x81；val=1； 关灯：type=0x80；val=0； 开灯并设置亮度值：type=0xff；val=亮度值； 开灯并设置亮度值：type=0xfe；val=亮度值；
	P1	打开窗帘(open)	type%2==1 表示打开窗帘；	RW	执行打开窗帘，则下发：type=0x81,val=1;
	P2	停止(stop)	type%2==1 表示停止当前动作；	RW	执行停止窗帘，则下发：type=0x81,val=1;
	P3	关闭窗帘(close)	type%2==1 表示关闭窗帘；	RW	执行关闭窗帘，则下发：type=0x81,val=1;

SL_CN_IF	P4	打开窗帘(open)时指示灯的颜色值	type%2=1 表示打开;
			type%2=0 表示关闭;
			val表示指示灯亮度值,定义如下:
			bit0~bit7:Blue
			bit8~bit15: Green
			bit16~bit23:Red
			bit24~bit31:white
			（当White>0时，表示动态模式）具体动态值请参考：**附录 3.1 动态颜色（DYN）定义**
				RW	开灯:
					type=0x81; val=1;
					关灯:
					type=0x80; val=0;
					开灯并设置颜色或动态值:
					type=0xff; val=颜色或动态值;
					关灯并设置颜色值或动态值:
					type=0xfe; val=颜色或动态值;

	P5	停止(stop)时指示灯的颜色值	type%2=1 表示打开;
			type%2=0 表示关闭;
			val表示指示灯亮度值,定义如下:
			bit0~bit7:Blue
			bit8~bit15: Green
			bit16~bit23:Red
			bit24~bit31:white
			（当White>0时，表示动态模式）具体动态值请参考：**附录 3.1 动态颜色（DYN）定义**
				RW	开灯:
					type=0x81; val=1;
					关灯:
					type=0x80; val=0;
					开灯并设置颜色或动态值:
					type=0xff; val=颜色或动态值;
					关灯并设置颜色值或动态值:
					type=0xfe; val=颜色或动态值;

	P6	关闭窗帘(close)时指示灯的颜色址	type%2=1 表示打开;
			type%2=0 表示关闭;
			val表示指示灯亮度值,定义如下:
			bit0~bit7:Blue
			bit8~bit15: Green
			bit16~bit23:Red
			bit24~bit31:white
			（当White>0时，表示动态模式）具体动态值请参考：**附录 3.1 动态颜色（DYN）定义**
				RW	开灯:
					type=0x81; val=1;
					关灯:
					type=0x80; val=0;
					开灯并设置颜色或动态值:
					type=0xff; val=颜色或动态值;
					关灯并设置颜色值或动态值:
					type=0xfe; val=颜色或动态值;

SL_CN_FE	P1	打开窗帘(open)	type%2==1 表示打开窗帘;	RW	执行打开窗帘，则下发: type=0x81,val=1;
	P2	停止(stop)	type%2==1 表示停止当前动作;	RW	执行停止窗帘，则下发: type=0x81,val=1;

	P3	关闭窗帘(close)	type%2==1 表示关闭窗帘;	RW	执行关闭窗帘，则下发: type=0x81,val=1;

2.3.2 DOOYA窗帘电机

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
SL_DOOYA	P1	窗帘状态	type%2==1 表示控制正在运行; type%2==0 表示没有运行; 当正在运行的时候即(type%2==1): val&0x80==0x80表示正在开，否则表示正在关; val&0x7F的值表示窗帘打开的百分比([0,100]); 若val&0x7F大于100则表示获取不到位置信息，只有执行全开或全关之后才能重新获取位置信息。	R	

	P2	窗帘控制		W	完全打开，则下发:
					type=0xCF,val=100;
					
					完全关闭，则下发:
					type=0xCF,val=0;
					
					停止窗帘，则下发:
					type=0xCE,val=0x80;
					
					开到百分比，则下发:
					type=0xCF,val=percent;
					percent取值:
					[0,100];

2.3.3 智界窗帘电机智控器

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
	P2	打开窗帘(open)	type%2==1 表示打开窗帘;	RW	执行打开窗帘，则下发: type=0x81,val=1;
	P4	停止(stop)	type%2==1 表示停止当前动作;	RW	执行停止窗帘，则下发: type=0x81,val=1;

SL_P_V2	P3	关闭窗帘(close)	type%2==1 表示关闭窗帘;	RW	执行关闭窗帘，则下发: type=0x81,val=1;
	P8	电压(V)	val值表示原始电压值 v值将表示当前剩余电量百分比，值范围[0,100]，它是根据val电压值换算的。	R	

2.4 灯光系列（Light Series）

2.4.1 灯光系列

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

SL_CT_RGBW
SL_LI_RGBW
	RGBW	RGBW颜色值	type%2==1表示打开;
			type%2==0表示关闭;
			
			val值为颜色值，大小4个字节，定义如下:
			bit0~bit7:Blue
			bit8~bit15:Green
			bit16~bit23:Red
			bit24~bit31:White
			例如
			红色: 0x00FF0000
			白色: 0xFF000000
				RW	开灯:
					type=0x81; val=1;
					
					关灯:
					type=0x80; val=0;
					
					开灯并设置颜色值:
					type=0xff; val=颜色值;
					
					关灯并设置颜色值:
					type=0xfe; val=颜色值;
					
					注意：如果执行关灯动作则该灯光设备将处于关闭状态；如果执行开灯动作则需要参考DYN的配置，若DYN处于使能状态，则DYN会覆盖住灯光的颜色效果，既DYN的优先级高于RGBW的颜色显示，若只想显示颜色，则请关闭DYN。可以调用EpsSet接口命令一起设置DYN与RGBW两个值。

	DYN	动态颜色值	type%2==1表示打开动态； type%2==0表示关闭动态； val表示动态颜色值 具体动态值请参考：**附录 3.2 动态颜色（DYN）定义**	RW	使能： type=0x81；val=1； 关闭： type=0x80；val=0； 使能并设置动态值： type=0xff；val=动态值； 关闭并设置动态值： type=0xfe；val=动态值； **注意：当前DYN不能单独下发，必须与RGBW一起下发，请使用EpsSet接口命令一起设置DYN与RGBW两个值。如当前灯光设备处于关闭状态，则需要设置RGBW处于打开状态，既RGBW的type=0x81；val=1；然后再设置DYN等于具体指。**
SL_SC_RGB	RGB	RGB颜色值	type%2==1表示打开； type%2==0表示关闭； val值为颜色值，大小4个字节，定义如下： bit0~bit7:Blue bit8~bit15: Green bit16~bit23:Red bit24~bit31:White （当White>=128时，表示动态模式）具体动态值请参考：**附录 3.2 动态颜色（DYN）定义**	RW	开灯： type=0x81；val=1； 关灯： type=0x80；val=0； 开灯并设置颜色或动态值： type=0xff；val=颜色或动态值； 关灯并设置颜色值或动态值： type=0xfe；val=颜色或动态值；

2.4.2 量子灯（Quantum）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

	P1	亮度控制	type%2==1,表示打开(忽略val值)；
			type%2==0,表示关闭(忽略val值)；
			val指示灯光的亮度值范围[0,100]，100亮度最大。
				RW	开灯(打开)，则下发：
					type=0x81,val=1;
					
					关灯(关闭)，则下发：
					type=0x80,val=0;
					
					设置亮度，则下发：
					type=0xcf,val=亮度值
					亮度值=[0,100]

OD_WE_QUAN	P2	颜色控制	val值为颜色值，大小4个字节，定义如下： bit0~bit7:Blue bit8~bit15: Green bit16~bit23:Red bit24~bit31:White （当White>0时，表示动态模式）具体动态值请参考：**附录 3.2 动态颜色（DYN）定义** **附录 3.3 量子灯特殊（DYN）定义**	RW	设置颜色或动态值： type=0xff；val=颜色或动态值；

2.4.3 调光调色控制器/白光智能灯泡（Smart Bulb）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

SL_LI_WW	P1	亮度控制	type%2==1,表示打开（忽略val值）；
			type%2==0,表示关闭（忽略val值）；
			val指示灯光的亮度值范围[0,255]，255亮度最大。
				RW	打开，则下发：
					type=0x81,val=1;
					
					关闭，则下发：
					type=0x80,val=0;
					
					设置亮度，则下发：
					type=0xcf,val=亮度值
					亮度值=[0,255]

	P2	色温控制	val值为色温值，取值范围[0,255]，0表示暖光，255表示冷光	RW	设置色温，则下发： type=0xcf,val=色温值 色温值=[0,255]

2.4.4 门廊壁灯（Porch Wall Lamp）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

SL_LI_GD1	P1	开关	type%2==1表示处于打开状态； type%2==0表示处于关闭状态； val值为亮度值，可调范围：[0,255] 值越大表示光越亮，0处于最暗，光完全熄灭，255处于最亮。	RW	打开，则下发： type=0x81；val=1； 关闭，则下发： type=0x80；val=0； 打开并且设置亮度，则下发： type=207；val=亮度值[0,255]； 关闭并设置亮度，则下发： type=206；val=亮度值[0,255]；
	P2	移动检测	val值定义如下： 0：没有检测到移动 1:有检测到移动	R	
	P3	环境光照	val值表示原始光照值(单位：lux)	R	

2.4.5 花园地灯（Garden Landscape Light）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

SL_LI_UG1	P1	开关/颜色设置	type%2==1表示打开； type%2==0表示关闭； val值为颜色值，大小4个字节，定义如下： bit0~bit7:Blue bit8~bit15:Green bit16~bit23:Red bit24~bit31:White/DYN 例如： 红色：0x00FF0000 白色：0xFF000000 bit24~bit31即可以设置白光又可以设置动态。 • 当其值在[0~100]表示设置的是白光，0表示不显示白光，100表示白光最亮。 **即白光的显示范围是：[0~100]** • 当其值大于等于128表示设置为动态模式， 具体动态值请参考：**附录 3.2 动态颜色（DYN）定义**	RW	开灯： type=0x81；val=1； 关灯： type=0x80；val=0； 开灯并设置颜色值： type=0xff；val=颜色值； 关灯并设置颜色值： type=0xfe；val=颜色值；
	P2	环境光照	val值表示光照值(单位：lux)	R	
	P3	充电指示	val值定义如下： 0： 没有充电 1:正在充电	R	
	P4	电量	val表示原始电压值 v值将表示当前剩余电量百分比，值范围[0,100]，它是根据val电压值换算的。其电池为4.2v锂电池。	R	

2.5 超级碗（SPOT）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
MSL_IRCTL	RGBW	RGBW颜色值	type%2==1表示打开； type%2==0表示关闭； val值为颜色值，大小4个字节，定义如下： bit0~bit7:Blue bit8~bit15:Green bit16~bit23:Red bit24~bit31:White 例如： 红色：0x00FF0000 白色：0xFF000000	RW	开灯： type=0x81；val=1； 关灯： type=0x80；val=0； 开灯并设置颜色值： type=0xff；val=颜色值； 关灯并设置颜色值： type=0xfe；val=颜色值； **注意：如果执行关灯动作则该灯光设备将处于关闭状态；如果执行开灯动作则需要参考DYN的配置，若DYN处于使能状态，则DYN会覆盖住灯光的颜色效果，既DYN的优先级高于RGBW的颜色显示，若只想显示颜色，则请关闭DYN。可以调用EpsSet接口命令一起设置DYN与RGBW两个值。**
	DYN	动态颜色值	type%2==1表示打开动态； type%2==0表示关闭动态； val表示动态颜色值 具体动态值请参考：**附录 3.2 动态颜色（DYN）定义**	RW	使能： type=0x81；val=1； 关闭： type=0x80；val=0； 使能并设置动态值： type=0xff；val=动态值； 关闭并设置动态值： type=0xfe；val=动态值； **注意：当前DYN不能单独下发，必须与RGBW一起下发，请使用EpsSet接口命令一起设置DYN与RGBW两个值。如当前灯光设备处于关闭状态，则需要设置RGBW处于打开状态，既RGBW的type=0x81；val=1；然后再设置DYN等于具体指。**

OD_WE_IRCTL	RGB	RGB颜色值	type%2==1表示打开； type%2==0表示关闭； val值为颜色值，大小4个字节，定义如下： bit0~bit7:Blue bit8~bit15: Green bit16~bit23:Red bit24~bit31:White 例如： 红色：0x00FF0000 白色：0x00FFFFFF （当White>0时，表示动态模式）具体动态值请参考：**附录 3.2 动态颜色（DYN）定义**	RW	开灯： type=0x81；val=1； 关灯： type=0x80；val=0； 开灯并设置颜色或动态值： type=0xff；val=颜色或动态值； 关灯并设置颜色值或动态值： type=0xfe；val=颜色或动态值；
SL_SPOT	RGB	RGB颜色值	type%2==1表示打开； type%2==0表示关闭； val值为颜色值，大小4个字节，定义如下： bit0~bit7:Blue bit8~bit15: Green bit16~bit23:Red bit24~bit31:White （当White>0时，表示动态模式）具体动态值请参考：**附录 3.2 动态颜色（DYN）定义**	RW	开灯： type=0x81；val=1； 关灯： type=0x80；val=0； 开灯并设置颜色或动态值： type=0xff；val=颜色或动态值； 关灯并设置颜色值或动态值： type=0xfe；val=颜色或动态值；
SL_P_IR	**Note**： 红外模块/超级碗(Mini版)没有灯光控制，不需要关注IO属性，红外功能与其它超级碗相同 **Note:** 对于922射频版本的超级碗(Mini版)，其"P2" IO Idx用于指示配对按钮状态。 当 P2.type%2==1指示配对按钮按下； 当 P2.type%2==0指示配对按钮松开； 注意：仅指示配对按钮是否按下，并不指示当前是否处于配对状态!				

2.6 感应器系列（Sensor Series）

2.6.1 门禁感应器（Guard Sensor）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
SL_SC_G	G	当前状态	val值定义如下: 0：打开 1：关闭	R	
	V	电量	val值表示原始电压值 v值将表示当前剩余电量百分比，值范围[0,100]，它是根据val电压值换算的。	R	
SL_SC_BG	G	当前状态	val值定义如下: 0：打开 1：关闭	R	
	V	电量	val值表示原始电压值 v值将表示当前剩余电量百分比，值范围[0,100]，它是根据val电压值换算的。	R	
	B	按键状态	val值定义如下: 0：未按下按键 1：按下按键	R	
	AXS	震动状态	val值定义如下: 0：无震动 非0：震动	R	

2.6.2 动态感应器（Motion Sensor）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
SL_SC_MHW SL_SC_BM	M	移动检测	val值定义如下: 0：没有检测到移动 1:有检测到移动	R	
	V	电量	val表示原始电压值 v值将表示当前剩余电量百分比，值范围[0,100]，它是根据val电压值换算的。	R	

SL_SC_CM	P1	移动检测	val值定义如下: 0：没有检测到移动 1:有检测到移动	R	
	P3	电量	val表示原始电压值 v值将表示当前剩余电量百分比，值范围[0,100]，它是根据val电压值换算的。	R	
	P4	USB供电电压	val表示原始电压值 若val值大于430则表明电已经充满。若设备连接USB，供电在工作，则应该忽略P3电量属性。	R	

2.6.3 环境感应器（Env Sensor）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
SL_SC_THL SL_SC_BE	T	当前环境温度	val值表示原始温度值，它是温度值*10， v值表示实际值 （单位：℃）	R	
	H	当前环境湿度	val值表示原始湿度值，它是湿度值*10， v值表示实际值 （单位：%）	R	
	Z	当前环境光照	val值表示原始光照值， v值表示实际值 （单位：lux）	R	
	V	电量	val值表示原始电压值 v值将表示当前剩余电量百分比，值范围[0,100]，它是根据val电压值换算的。	R	

2.6.4 水浸传感器（Water Flooding Sensor）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
	WA	导电率	val值定义如下: 0：未检测到水； 值越大表示水越多，导电率越高；	R	

SL_SC_WA	V	电量	val值表示原始电压值 v值将表示当前剩余电量百分比，值范围[0,100]，它是根据val电压值换算的。	R	

2.6.5 气体感应器(甲醛)（CH_2O Sensor）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
SL_SC_CH	P1	甲醛浓度	type%2==1表示甲醛浓度值超过告警门限； val值表示甲醛浓度原始值，实际值等于原始值/1000（单位：ug/m3）； v值表示实际值； 甲醛浓度安全区间为：[0,0.086]mg/m3 也即：[0,86]ug/m3； 甲醛浓度告警门限设置请参考P2定义。	R	
	P2	甲醛浓度告警门限	val值越大则灵敏度越低，门限越高（单位：ug/m3）： 我们定义了三档： 不告警：val=5000； 中灵敏： val=100； 高灵敏： val=80； 如果取值不在这三个值之列，可以根据这三档自行去判断。	RW	设置报警器灵敏度，val值越大则灵敏度越低，门限越高（单位：ug/m3）： type=0x8D； 各种级别val取值如下： 不告警：val=5000； 中灵敏：val=100； 高灵敏：val=80；
	P3	警报音	type%2==1指示报警音正在响，反之则没有报警音。 注意：有没有甲醛报警需要查看P1的值，具体参考P1定义。	RW	若要手工触发报警音，则可以下发： type=0x81,val=1 若需手动消除报警音，则可以下发： type=0x80,val=0

2.6.6 气体感应器(燃气)（Gas Sensor）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
SL_SC_CP	P1	燃气浓度	type%2==1表示燃气浓度值超过告警门限，有告警； val为当前燃气浓度值。 燃气浓度告警门限设置请参考P2定义。	R	
	P2	燃气浓度告警门限	val值越大则灵敏度越低，门限越高，我们定义了三档： 低灵敏度：val=150; 中灵敏度： val=120; 高灵敏度： val=90; 如果燃气浓度超过了门限设定，则将出发告警。 如果取值不在这三个值之列，可以根据这三档自行去判断。	RW	设置报警器灵敏度，val值越大则灵敏度越低，门限越高： type=0xDF; 各种级别val取值如下： 低灵敏度：val=120; 中灵敏度：val=120; 高灵敏度：val=90;
	P3	警报音	type%2==1指示报警音正在响，反之则没有报警音。 注意：有没有燃气报警需要查看P1的值，具体参考P1定义。	RW	若要手工触发报警音，则可以下发： type=0x81,val=1 若需手动消除报警音，则可以下发： type=0x80,val=0

2.6.7 环境感应器(TVOC+CO_2)（TVOC+CO_2 Sensor）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
	P1	当前环境温度	val值表示原始温度值，它是温度值*10， v值表示实际值 （单位：℃）	R	
	P2	当前环境湿度	val值表示原始湿度值，它是湿度值*10， v值表示实际值 （单位：%）	R	
	P3	当前CO2浓度值	val值表示CO2浓度值， v值表示实际值 （单位：ppm） 我们定义了以下几档参考： val <= 500：优 val <= 700：良 val <= 1000：中 val > 1000：差	R	

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
SL_SC_CQ	P4	当前TVOC浓度值	val值表示TVOC原始浓度值，它是TVOC浓度值*1000，也即是实际浓度值=原始浓度值/1000， v值表示实际值 （单位：mg/m³） 我们定义了以下几档参考： val < 0.34：优 val < 0.68：良 val <= 1.02：中 val > 1.02：差	R	
	P5	电量	val值表示原始电压值 v值将表示当前剩余电量百分比，值范围[0,100]，它是根据val电压值换算的。	R	
	P6	USB供电电压	val表示原始电压值 若val值大于430则表明电已经充满。若设备连接USB，供电在工作，则应该忽略P3电量属性。	R	

2.6.8 ELIQ电量计量器（ELIQ）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
ELIQ_EM	EPA	平均功率	val值表示平均功率	R	

2.6.9 烟雾感应器（Smoke Sensor）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
SL_P_A	P1	当前是否有烟雾告警	val等于0表示没有烟雾告警，等于1表示有烟雾告警	R	
	P2	电压	val表示原始电压值 • 如果使用9V的电池，则实际电压值=(val/100)*3，注意：其值可能会超过9V，例如9.58V； • 如果外接12V供电，则该值无效。 v值将表示当前剩余电量百分比，值范围[0,100]，它是根据val电压值换算的。	R	

2.6.10 环境感应器(CO_2)（CO_2 Sensor）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
SL_SC_CA	P1	当前环境温度	val值表示原始温度值，它是温度值*10， v值表示实际值 （单位：℃）	R	
	P2	当前环境湿度	val值表示原始湿度值，它是湿度值*10， v值表示实际值 （单位：%）	R	
	P3	当前CO2浓度值	val值表示CO2浓度值， v值表示实际值 （单位：ppm） 我们定义了以下几档参考： val <= 500：优 val <= 700：良 val <= 1000：中 val > 1000：差	R	
	P4	电量	val值表示原始电压值 v值将表示当前剩余电量百分比，值范围[0,100]，它是根据val电压值换算的。	R	
	P5	USB供电电压	val表示原始电压值 若val值大于0则表明供电在工作，否则表明未供电工作。	R	

2.6.11 人体存在感应器（Radar Motion Sensor）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
	P1	移动检测 Motion	val值定义如下： 0：没有检测到移动 非0:有检测到移动	R	

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
SL_P_RM	P2	移动检测参数设置	val值表示移动检测的参数设置，包含动态锁定时间与灵敏度设置。其中： • **bit0-bit7：动态锁定时间** 取值范围为：1-255，具体锁定时间为：配置值*4，单位为秒，例如bit0-bit7配置值为16，则表示动态锁定时间为64秒。 • **bit8-bit25：灵敏度** 灵敏度默认值为4，范围1-255，值越小则越灵敏。	RW	设置 感应器动态锁定时间与灵敏度 type=0xDF; val=需要设置的值;

2.6.12 云防门窗感应器（DEFED Window/Door）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
SL_DF_GG	A	当前状态	type%2==1 表示处于打开状态(忽略val值)； type%2==0 表示处于吸合状态(忽略val值)；	R	
	A2	外部感应器状态	type%2==1 表示处于打开状态(忽略val值)； type%2==0 表示处于吸合状态(忽略val值)； 需要接外部感应器，如果没有接则type值为1。	R	
	T	温度	val值表示原始温度值，它是实际温度值*10， v值表示实际值（单位：℃）	R	
	V	电量	val值表示原始电压值 v值将表示当前剩余电量百分比，值范围[0,100]，它是根据val电压值换算的。 **注意：type%2==1表示低电报警状态**	R	
	TR	防拆状态	type%2==1 则表示触发防拆警报； type%2==0 则表示状态正常；	R	

2.6.13 云防动态感应器（DEFED Motion）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
SL_DF_MM	M	当前状态	type%2==1 表示侦测到人体移动(忽略val值)； type%2==0 表示没有侦测到人体移动(忽略val值)；	R	
	T	温度	val值表示原始温度值，它是实际温度值*10， v值表示实际值（单位：℃）	R	
	V	电量	val值表示原始电压值 v值将表示当前剩余电量百分比，值范围[0,100]，它是根据val电压值换算的。 **注意：type%2==1表示低电报警状态**	R	

	TR	防拆状态	type%2==1 则表示触发防拆警报； type%2==0 则表示状态正常；	R	

2.6.14 云防室内警铃（DEFED Indoor Siren）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
SL_DF_SR	SR	当前状态	type%2==1 表示警铃播放(忽略val值)； type%2==0 表示正常(忽略val值)；	R	
	T	温度	val值表示原始温度值，它是实际温度值*10， v值表示实际值 （单位：℃）	R	
	V	电量	val值表示原始电压值 v值将表示当前剩余电量百分比，值范围[0,100]，它是根据val电压值换算的。 **注意：type%2==1表示低电报警状态**	R	
	TR	防拆状态	type%2==1 则表示触发防拆警报； type%2==0 则表示状态正常；	R	

	P1	播放控制	val为32bit值，描述如下(16进制)：0xAABBCCDD
			AABB表示鸣笛持续时长，单位为0.1秒；
			CC是声音强度(136-255)，255最强，136最弱，请不要设置声音强度低于136，否则会听不到声音；
			DD表示音频序号；
			0：无
			1：信息
			2：告警
				RW	播放，则下发：
					type=0x81, val=1;
					停止，则下发：
					type=0x80, val=0;
					设置值并播放：
					type=0xff；val=需要设置的值；
					设置值并停止：
					type=0xfe；val=需要设置的值；
					设置值并保持当前播放状态：
					type=保持当前值；val=需要设置的值；

2.6.15 云防遥控器（DEFED Key Fob）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
SL_DF_BB	eB1	按键1状态 (为布防图标)	type%2==1 表示按键处于按下状态(忽略val值)； type%2==0 表示按键处于松开状态(忽略val值)；	R	
	eB2	按键2状态 (为撤防图标)	type%2==1 表示按键处于按下状态(忽略val值)； type%2==0 表示按键处于松开状态(忽略val值)；	R	
	eB3	按键3状态 (为警告图标)	type%2==1 表示按键处于按下状态(忽略val值)； type%2==0 表示按键处于松开状态(忽略val值)；	R	
	eB4	按键4状态 (为在家图标)	type%2==1 表示按键处于按下状态(忽略val值)； type%2==0 表示按键处于松开状态(忽略val值)；	R	
	V	电量	val值表示原始电压值 v值将表示当前剩余电量百分比，值范围[0,100]，它是根据val电压值换算的。 **注意：type%2==1表示低电报警状态**	R	

2.6.16 噪音感应器（Noise Sensor）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
	P1	噪音值	type%2==1 表示噪音值大于告警门限； type%2==0 表示噪音值没有超过告警门限； val表示当前噪音值，单位为 分贝	R	

P2	告警门限设置	val为32bit值，描述如下(16进制)：0xAABBCCDD • DD表示告警门限值，数值单位为分贝，取值范围[0,255]； • CC表示采样值1，取值范围[0,255]； • BB表示采样值2，取值范围[0,255]； • CCBB共同作用形成越限率，举例说明，CC=6，BB=10，则表示如果最近采样的10次数据中，其值有6次超过门限DD的设定值，则进入告警； • AA表示警报持续时间，其单位为0.5秒，如AA=20，则表示警报持续时间为10秒。	RW	修改门限值： type=255，val取值参考属性值描述说明 缺省配置如下： DD：70 CC：6 BB：10 AA：20

SL_SC_CN	P3	报警设置	type%2==1 表示处于报警状态；
			type%2==0 表示处于正常状态；
			
			val为32bit值，描述如下(16进制)：0xAABBCCDD
			• AABB表示警报持续时长，单位为0.1秒，等于65535则表示一直持续；
			• CC是声音强度，0表示没有声音，其它值表示有声音，请注意：设置为0将没有声音；
			• DD表示音频模式；
			0：无声音
			1：指示音,每秒响0.2秒
			2：告警音,每秒响0.5秒
			0x7F：测试音,每秒响1秒
			0x80-0xFF：自定义模式
			• 自定义模式8bit中，最高位必须是1，8bit从高到低设定如下：BXXXYYYYY
			
			▸ XXX表示声音模版。1表示响,0表示静音,每位时长是0.1秒；
			▸ YYYYY表示周期时长。取值0-31分别对应1-32个周期时长，单位是0.1秒，在一个周期时长内，前3个单位既0.3秒使用XXX指示的声音模版，剩余的时间则静音。
			
			• 例如：
			▸ B10100111,XXX=101,YYYYY=00111：则周期时长为8即0.8秒，第0.1秒响、第0.3秒响，其它时间则静音。
			
			▸ B11101000,XXX=111,YYYYY=01000：则周期时长为9即0.9秒，第0.1秒响、第0.2秒响，第0.3秒响，其它时间则静音。
			
			▸ B11100000,XXX=111,YYYYY=00000：则周期时长为1即0.1秒，由于周期时长短于3，不够声音模版，则直接使用声音模版第一位值，则会一直响。
				RW	播放，则下发：
					type=0x81, val=1;
					
					停止，则下发：
					type=0x80, val=0;
					
					设置值并播放：
					type=0xff；val=需要设置的值；
					
					设置值并停止：
					type=0xfe；val=需要设置的值；
					
					设置值并保持当前播放状态：
					type=保持当前值；
					val=需要设置的值；

	P4	噪音校正值	取值范围为 [-128~127]	RW	修改校正值： type=207, val取值参考属性值描述说明 注：如果噪音采样有误差，可以配置噪音校正值校正。

2.7 空气净化器（AIR Purifier）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

OD_MFRESH_M8088	O	开关	type%2==1，val值忽略 表示打开；
			type%2==0，val值忽略 表示关闭；
				RW	打开，则下发：
					type=0x81,val=1;
					关闭，则下发：
					type=0x80,val=0;

	RM	运行模式	val值定义如下： 0:auto 1～3:风量1～3 4:风量最大 5:睡眠模式	RW	
	T	温度	val值表示原始温度值，它是温度值*10，v值表示实际值（单位：℃）	R	
	H	湿度	val值表示原始湿度值，它是湿度值*10，v值表示实际值（单位：%）	R	
	PM	PM2.5	val值表示PM2.5值，v值表示实际值（单位：ug/m³）	R	
	FL	滤芯寿命	val值表示滤芯寿命，范围：0～4800（单位：h）	R	
	UV	紫外线	val值表示紫外线指数	R	

2.8 智能门锁（Smart Door Lock）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
	BAT	电量	val表示电量值（单位：%）	R	

SL_LK_LS SL_LK_GTM SL_LK_AG SL_LK_SG SL_LK_YL	ALM	告警信息	val值定义如下： bit0：1为错误报警（输入错误密码或指纹或卡片超过10次就报警） bit1：1为劫持报警（输入防劫持密码或防劫持指纹开锁就报警） bit2：1为防撬报警（锁被撬开） bit3：1为机械钥匙报警（使用机械钥匙开锁） bit4：1为低电压报警（电池电量不足） bit5：1为异动告警 bit6：1为门铃 bit7：1为火警 bit8：1为入侵告警 bit11：1为恢复出厂告警	R	
	EVTLO	实时开锁信息	type%2==1 表示打开； type%2==0 表示关闭； val值定义如下： bit0～11表示用户编号； bit12～15表示开锁方式：（ 0：未定义； 1：密码； 2：指纹； 3：NFC； 4：机械钥匙； 5：远程开锁； 6：一键开启(12V开锁信号开锁)； 7：APP开启； 8：蓝牙开锁； 9：手动开锁； 15：出错） bit16～27表示用户编号； bit28～31表示开锁方式：（同上定义）（注：因有可能存在两种方式同时开启门锁的情况，单开时bit0～15为开锁信息，其他位为0；双开时bit0～15和bit16～31分别为相应的开锁信息）	R	
	EVTOP	操作记录	val的长度有8/24/32bit三种类型，根据type可以获知长度，方法是： (type=0x40+(8-1)*2 or type=0x40+(16-1)*2 or type=0x40+(32-1)*2) val的通用的编码次序是：[1Byte的记录类型] [2Byte的用户id] [1Byte的用户flag] 用户标志flag：bit01=11表示管理员,01表示普通用户,00表示已经删除了	R	

	HISLK	最近一次开锁信息	type%2=1 表示打开； type%2=0 表示关闭； val值定义如下： bit0～11表示用户编号； bit12～15表示开锁方式：（ 0：未定义； 1：密码； 2：指纹； 3：NFC； 4：机械钥匙； 5：远程开锁； 6：一键开启(12V开锁信号开锁)； 7：APP开启） bit16～27表示用户编号； bit28～31表示开锁方式：(同上定义）	R	

2.9 温控器系列（Thermostat Series）

2.9.1 智控器空调面板（Central AIR Board）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
V_AIR_P	O	开关	type%2==1，val值忽略表示打开； type%2==0，val值忽略表示关闭；	RW	打开空调，则下发type=0x81,val=1; 关闭空调，则下发type=0x80,val=0;
	MODE	模式	type==0xCE，val值表示模式，定义如下: 1:Auto 自动; 2:Fan 吹风; 3:Cool 制冷; 4:Heat 制热; 5:Dry 除湿:	RW	下发命令的时候type保持原样， val值定义与属性址描述相同
	F	风速	type==0xCE，val值表示风速，定义如下: val<30:低档; val<65:中档; val>=65:高档	RW	下发命令的时候type保持原样， val值如下: 低档，val=15; 中档，val=45; 高档，val=75;
	tT	目标温度	type==0x88， v值表示实际温度值 val值表示原始温度值，它是温度值*10	RW	下发命令的时候type保持原样，val需要使用原始温度值，即目标温度值*10，例如下发设定温度为25摄氏度，则应该下发val=250
	T	当前温度	type==0x08， v值表示实际温度值 val值表示原始温度值，它是温度值*10	R	

2.9.2 新风系统（Fresh Air System）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

	P1	系统配置	type%2==1 表示打开;
			type%2==0 表示关闭;
			
			val表示模式配置，bit13～bit14定义如下:
			1：自动
			2：手动
			3：定时
				RW	打开，则下发:
					type=0x81, val=1;
					关闭，则下发:
					type=0x80, val=0;
					
					如果需要修改val的配置，则下发:
					type保持当前值，
					val等于要配置的值。

SL_TR_ACIPM	P2	风速	val值定义如下： 0：关闭 1：1档 2：2档 3：3档	RW	type保持当前值， val设置风速。 注意：只有在模式处于手动模式下该参数设置才有效。
	P3	设置VOC	val值减小10倍为真实值， v值表示实际值 （单位：ppm）	RW	设置VOC值(val)时，需要将真实值扩大10倍
	P4	VOC	val值表示原始VOC值，且 val值减小10倍为真实值， v值表示实际值 （单位：ppm）	R	
	P5	PM2.5	val值表示原始PM2.5值， v为实际值 （单位：vg/m³）	R	
	P6	当前温度	val值除以10为真实温度值， v值表示实际值 （单位：℃）	R	

2.9.3 地暖温控器（Thermostat）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

SL_CP_DN	P1	系统配置	该IO的type和val字段说明，
			详见：
			表2-17-1
				RW	打开，则下发：
					type=0x81, val=1;
					关闭，则下发：
					type=0x80, val=0;
					
					如果需要修改val的配置，则下发：
					type保持当前值，
					val等于要配置的值。

	P2	继电器开关	type%2==1 表示打开； type%2==0 表示关闭；	RW	
	P3	目标温度	val值表示原始温度值，真实温度值为原始值减小10倍，精度为0.5， v值表示实际值	RW	如果需要修改val的配置，则下发： type保持当前值， val等于要配置的值。
	P4	室内温度	val值表示原始温度值，真实温度值为原始值减小10倍，精度为0.1， v值表示实际值	R	

	P5	底版温度	val值表示原始温度值，真实温度值为原始值除以10，精度为0.1， v值表示实际值	R	

type说明	最低位代表系统的开机状态 0:OFF 1:ON										
val说明	bit位	31	24~19	18~17	16~15	14~11	10~8	7~3	2	1	0
	设置内容	模式 0:手动模式 1:自动模式	限温值 =val+40	控温模式	时段模式	外置探头差 =(val-10)/5	内置探头差 =(val-10)/5	温度修正 =val/2+5	防冻	停电后来电状态	背光

表2-17-1

2.9.4 风机盘管（Fan Coil Unit）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

SL_CP_AIR	P1	系统配置	该IO的type和val字段说明，详见：
			表2-18-1
				RW	打开，则下发：
					type=0x81, val=1;
					关闭，则下发：
					type=0x80, val=0;
					
					如果需要修改val的配置，则下发：
					type保持当前值，
					val等于要配置的值。

	P2	阀门状态	type值定义如下： 0x80:阀门关 0x81:阀门开	R	
	P3	风速状态	val值定义如下： 0：自动； 1：低速； 2：中速； 3：高速	R	
	P4	目标温度	val值表示原始温度值，真实温度值为原始值减小10倍，精度为0.5， v值表示实际值	RW	
	P5	室内温度	val值表示原始温度值，真实温度值为原始值除以10，精度为0.1， v值表示实际值	R	

Type 说明	最低位代表系统的开机状态 0:OFF 1:ON							
Val 说明	Bit位	16~15	14~13	12~8	7~3	2	1	0
	设置内容	风速： 0: 自动; 1~3: 1~3档;	模式： 0: 制冷; 1: 制热; 2: 通风;	dt0外置探头回差 =val/2	温度修正 =val/2+5	sat 停电后状态	背光	风机 Fn

表2-18-1

2.9.5 空调控制面板（AIR Board）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

SL_UACCB	P1	开关
		(O)
			type%2==1，val值忽略表示打开；
			type%2==0，val值忽略表示关闭；
				RW	打开空调，则下发
					type=0x81,val=1;
					关闭空调，则下发
					type=0x80,val=0;

	P2	模式 (MODE)	type==0xCE，val值表示模式，定义如下： 1:Auto 自动； 2:Fan 吹风； 3:Cool 制冷； 4:Heat 制热； 5:Dry 除湿：	RW	下发命令的时候type保持原样， val值定义与属性址描述相同
	P3	目标温度 (tT)	type==0x88， v值表示实际温度值 val值表示原始温度值，它是温度值*10	RW	下发命令的时候type保持原样，val需要使用原始温度值，即目标温度值*10，例如下发设定温度为25摄氏度，则应该下发 val=250
	P4	风速 (F)	type==0xCE，val值表示风速，定义如下： val<30:低档； val<65:中档； val>=65:高档	RW	下发命令的时候type保持原样， val值如下： 低档，val=15； 中档，val=45； 高档，val=75；
	P6	当前温度 (T)	type==0x08， v值表示实际温度值 val值表示原始温度值，它是温度值*10	R	

2.9.6 温控阀门（Thermostat Valve）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

	P1	开关及系统配置(O)	type%2==1，val值忽略表示打开；
			type%2==0，val值忽略表示关闭；
			
			该IO的type和val字段说明，详见：
			表2-19-1
				RW	打开空调，则下发
					type=0x81,val=1;
					关闭空调，则下发
					type=0x80,val=0;
					
					如果需要修改val的配置，则下发：
					type保持当前值，
					val等于要配置的值。

	P3	目标温度 (tT)	v值表示实际温度值 val值表示原始温度值，它是温度值*10	RW	下发命令的时候type保持原样，val需要使用原始温度值，即目标温度值*10，例如下发设定温度为25摄氏度，则应该下发 val=250

SL_CP_VL	P4	当前温度 (T)	v值表示实际温度值 val值表示原始温度值，它是温度值*10	R	
	P5	告警 (ALM)	val表示告警信息，可参考: bit0:高温保护 bit1:低温保护 bit2:int_sensor bit3:ext_sensor bit4:低电量 bit5:设备掉线	R	
	P6	电量 (V)	val值表示原始电压值 v值将表示当前剩余电量百分比，值范围[0,100]，它是根据val电压值换算的。	R	

type说明	最低位代表系统的开机状态 0:OFF 1:ON		
val说明	bit位	2～1	0
	设置内容	值为0表示手动模式; 值为1表示节能模式; 值为2表示自动模式;	值为0表示关闭儿童锁; 值为1表示开启儿童锁;

表2-19-1

2.9.7 艾弗纳 KV11（Controlador EFFNER KV11）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

	O	开关	type%2==1, val值忽略表示打开;
			type%2==0, val值忽略表示关闭;
				RW	打开空调，则下发
					type=0x81,val=1;
					关闭空调，则下发
					type=0x80,val=0;

	MODE	模式	val值定义如下: 0-1位: 0x01:手动 0x10:定时 2-3位: 0x01:热交换 0x10:防霜冻 其它值则为未知。	RW	type保持当前值， val设置模式。 0-1位: 0x01:手动 0x10:定时 2-3位: 0x01:热交换 0x10:防霜冻

V_FRESH_P	F1	送风机	val值表示风速，定义如下: 0：停止 val<30:低档; val<65:中档; val>=65:高档	RW	下发命令的时候type保持原样， val值如下: 停止，val=0; 低档，val=15; 中档，val=45; 高档，val=75;
	F2	排风机	val值表示风速，定义如下: 0：停止 val<30:低档; val<65:中档; val>=65:高档	RW	下发命令的时候type保持原样， val值如下: 停止，val=0; 低档，val=15; 中档，val=45; 高档，val=75;
	T	当前温度	val值除以10为真实温度值， v值表示实际值（单位：℃）	R	

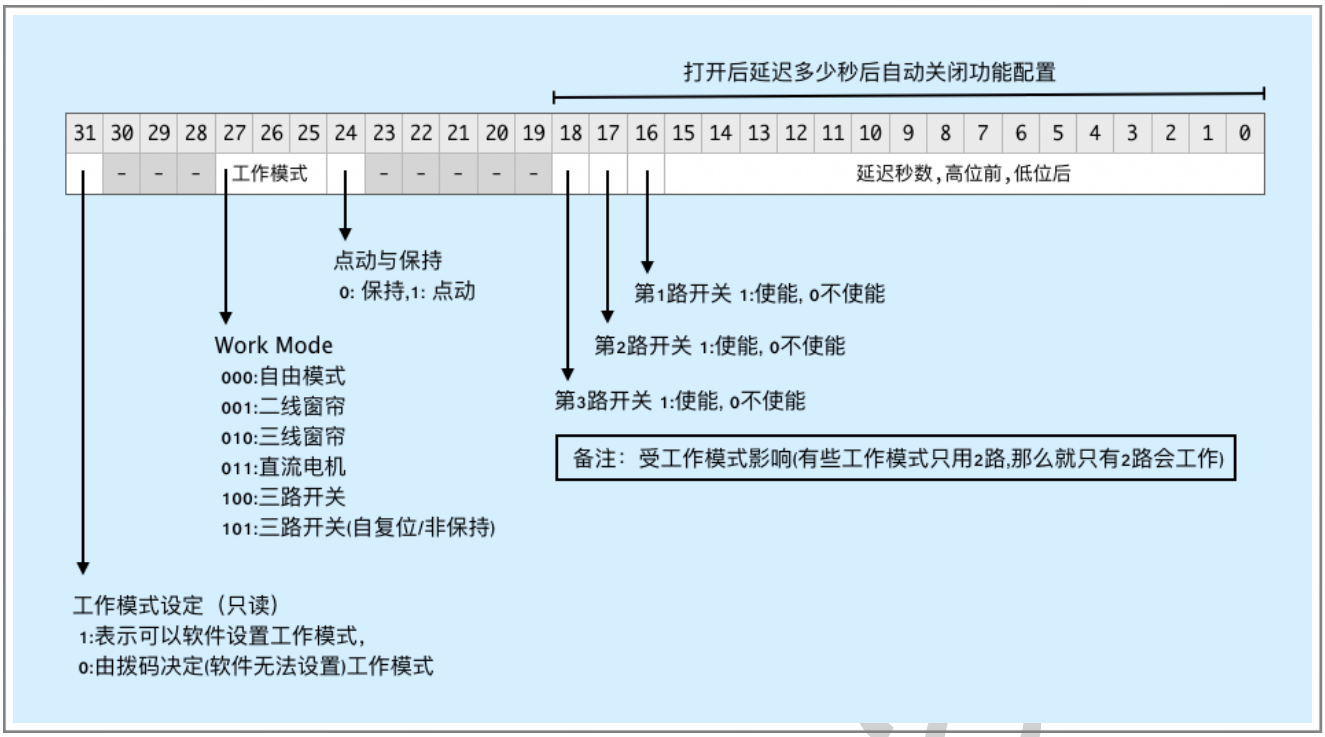
2.10 通用控制器系列（General Controller Series）

2.10.1 通用控制器（General Controller）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
SL_P	P1		详见如下：**Note**	RW	
	P2	Ctrl1	type%2==1 表示打开; type%2==0 表示关闭;	RW	
	P3	Ctrl2	type%2==1 表示打开; type%2==0 表示关闭;	RW	
	P4	Ctrl3	type%2==1 表示打开; type%2==0 表示关闭;	RW	
	P5	Status1	type%2==1 表示有状态触发，仅自由模式有效	R	
	P6	Status2	type%2==1 表示有状态触发，仅自由模式有效	R	
	P7	Status3	type%2==1 表示有状态触发，仅自由模式有效	R	

Note：

- P2,P3,P4的工作模式是由P1来确定的。
  val：P1.val(其值为32位整数)
  31bit：等于1表示可以由软件设置工作模式，等于0表示只能由拨码决定(软件无法设置)工作模式)，它是一个只读属性
- 工作模式：(val >>> 24) & 0xe
  0：自由模式
  2：二线窗帘
  4：三线窗帘
  6：直流电机
  8：三路开关
  10：三路开关(自复位/非保持)
- 点动与保持：(val >>> 24) & 1
  0：保持：比如开关
  1：点动：按下时接通，松开时断开
- 打开后延迟多少秒后自动关闭功能配置：val & 0x7FFFF
  18bit：第3路开关 1:使能，0不使能
  17bit：第2路开关 1:使能，0不使能
  16bit：第1路开关 1:使能，0不使能
  15-0bit：延迟秒数,高位前,低位后
  受工作模式影响(有些工作模式只用2路,那么就只有2路会工作)
- **修改时，需保留未修改的bit位数据**



• 如果工作模式是二线窗帘，三线窗帘，直流电机则：

P2表示打开窗帘；P3表示关闭窗帘；P4表示停止窗帘；

具体含义请参考：**2.3 窗帘控制系列（Curtain Controller）**。

• 如果工作模式是三路开关，三路开关(翘板)则：

P2表示第一路开关；P3表示第二路开关；P4表示第三路开关；

具体含义请参考：**2.2 开关系列（Switch Series）**。

• 如果工作模式是自由模式，则P5，P6，P7表示状态。

可分别接到带有线输出的传感器(如烟感、气感、红外 栅栏、火焰等)设备的不同继电器输出干触点，当检测到有信号输出 ，P5/P6/P7会反应状态。

2.10.2 通用控制器(HA)（HA Interface Adapter）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
	P1		详见如下：**Note**	RW	

	P2	Ctrl1	type%2==1 表示打开;
			type%2==0 表示关闭;
				RW	打开，则下发:
					type=0x81, val=1;
					关闭，则下发:
					type=0x80, val=0;

	P3	Ctrl2	type%2==1 表示打开;
			type%2==0 表示关闭;
				RW	打开，则下发:
					type=0x81, val=1;
					关闭，则下发:
					type=0x80, val=0;

	P4	Ctrl3	type%2==1 表示打开;
			type%2==0 表示关闭;
				RW	打开，则下发:
					type=0x81, val=1;
					关闭，则下发:
					type=0x80, val=0;

SL_JEMA	P5	Status1	type%2==1 表示有状态触发，仅自由模式有效	R	
	P6	Status2	type%2==1 表示有状态触发，仅自由模式有效	R	
	P7	Status3	type%2==1 表示有状态触发，仅自由模式有效	R	

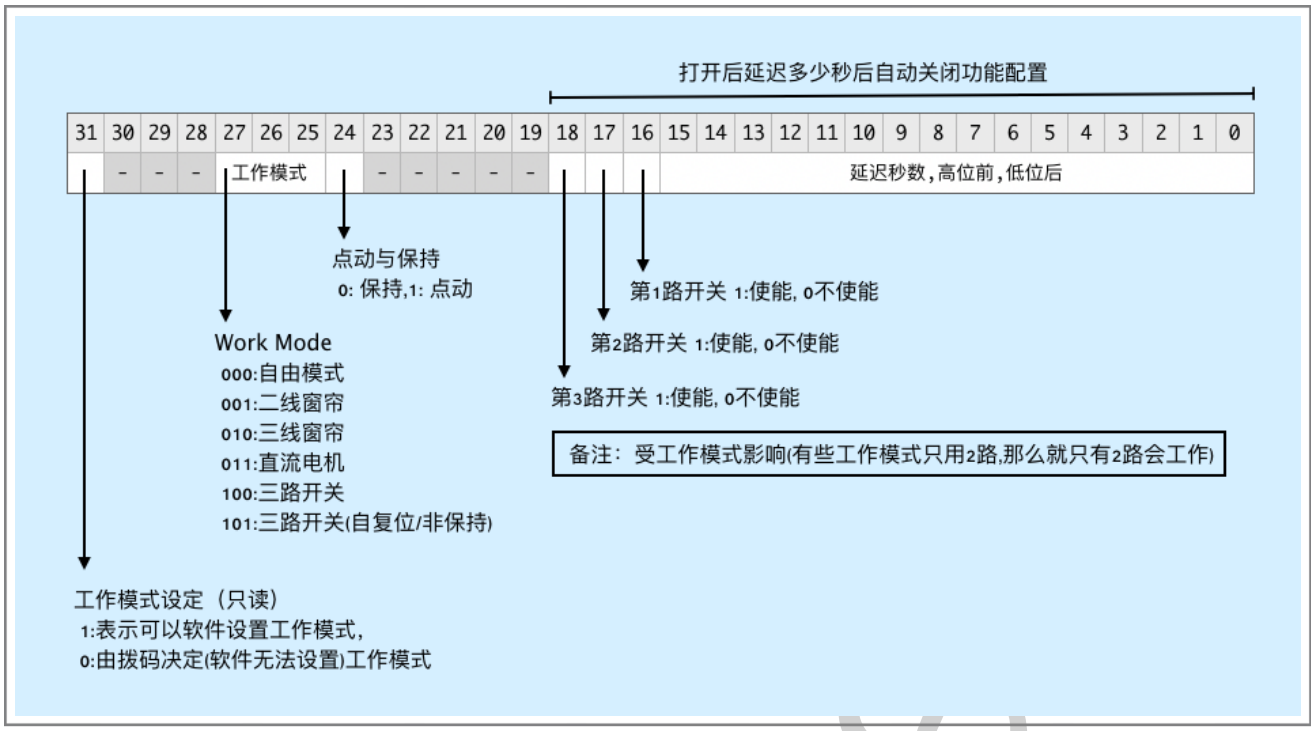
	P8	HA1	type%2==1 表示打开;
			type%2==0 表示关闭;
				RW	打开，则下发:
					type=0x81, val=1;
					关闭，则下发:
					type=0x80, val=0;

	P9	HA2	type%2==1 表示打开;
			type%2==0 表示关闭;
				RW	打开，则下发:
					type=0x81, val=1;
					关闭，则下发:
					type=0x80, val=0;

	P10	HA3	type%2==1 表示打开;
			type%2==0 表示关闭;
				RW	打开，则下发:
					type=0x81, val=1;
					关闭，则下发:
					type=0x80, val=0;

Note:

- P2,P3,P4的工作模式是由P1来确定的。
  val: P1.val(其值为32位整数)
  31bit: 等于1表示可以由软件设置工作模式，通用控制器(HA)其值一直为1，它是一个只读属性
- 工作模式: (val >>> 24) & 0xe
  0. 自由模式
  2. 二线窗帘
  4. 三线窗帘
  6. 直流电机
  8. 三路开关
  10. 三路开关(自复位/非保持)
- 点动与保持: (val >>> 24) & 1
  0. 保持: 比如开关
  1. 点动: 按下时接通，松开时断开
- 打开后延迟多少秒后自动关闭功能配置: val & 0x7FFFF
  18bit: 第3路开关 1:使能, 0不使能
  17bit: 第2路开关 1:使能, 0不使能
  16bit: 第1路开关 1:使能, 0不使能
  15-0bit: 延迟秒数,高位前,低位后
  受工作模式影响(有些工作模式只用2路,那么就只有2路会工作)
- **修改时，需保留未修改的bit位数据**



• 如果工作模式是二线窗帘，三线窗帘，直流电机则：

P2表示打开窗帘；P3表示关闭窗帘；P4表示停止窗帘；

具体含义请参考：**2.3 窗帘控制系列（Curtain Controller）**。

• 如果工作模式是三路开关，三路开关(翘板)则：

P2表示第一路开关；P3表示第二路开关；P4表示第三路开关；

具体含义请参考：**2.2 开关系列（Switch Series）**。

• 如果工作模式是自由模式，则P5，P6，P7表示状态。

可分别接到带有线输出的传感器(如烟感、气感、红外 栅栏、火焰等)设备的不同继电器输出干触点，当检测到有信号输出 ，P5/P6/P7会反应状态。

• P8,P9,P10是HA(Home Automation)开关端口，它们是独立的端口，不受P1工作模式设置的影响。具体命令控制见上表说明。

2.10.3 DLT电量计量

Devtype/Cls	IO idx	IO名称	属性值描述	RW	属性值说明
V_DLT645_P	EE	用电量	为累计用电量， val值为为IEEE 754浮点数的32位整数表示， v值为浮点数，单位为度(kwh)。	R	注意：v值可以直接使用，若不存在v值，则需要手动转换。 其值类型为IEEE 754浮点数的32位整数布局。 例如 1024913643 表示的是浮点值：0.03685085，可以依据整数值计算出浮点值。 具体请参考： **附录3.4 IO值浮点类型说明**

	EP	功率	为当前负载功率， val值为为IEEE 754浮点数的32位整数表示， v值为浮点数，单位为w。	R	注意：v值可以直接使用，若不存在v值，则需要手动转换。 其值类型为IEEE 754浮点数的32位整数布局。 例如 1024913643 表示的是浮点值：0.03685085，可以依据整数值计算出浮点值。 具体请参考： **附录3.4 IO值浮点类型说明**

2.10.4 485控制器

该设备较为特殊，设备的属性集合为实际接入的设备有关，实际的设备属性以接口调用获取的属性为准，其中可能出现的属性可以参考以下描述：

Devtype/Cls	IO idx	IO名称	属性值描述	RW	属性值说明
	P1	值	为当前接入设备的值， val值为为IEEE 754浮点数的32位整数表示， v值为浮点数，单位为具体接入设备当前的单位。 如：接入设备为压力传感器，那么val为当前接入设备的压力值，单位以接入设备的单位设定为准。	R	注意：v值可以直接使用，若不存在v值，则需要手动转换。 其值类型为IEEE 754浮点数的32位整数布局。 例如 1024913643 表示的是浮点值：0.03685085，可以依据整数值计算出浮点值。 具体请参考： **附录3.4 IO值浮点类型说明**

	Lx（x取值为数字）	开关	type%2=1，val值忽略表示打开；
			type%2=0，val值忽略表示关闭；
			（Lx，x为1时，即L1表示第一位开关的IO控制口，多位开关时x可取值为3，L3则表示第三位开关的IO控制口）
				RW	打开，则下发：
					type=0x81,val=1;
					关闭，则下发：
					type=0x80,val=0;

	O	开关	type%2=1，val值忽略表示打开；
			type%2=0，val值忽略表示关闭；
				RW	打开，则下发：
					type=0x81,val=1;
					关闭，则下发：
					type=0x80,val=0;

V_485_P	EE	用电量	为累计用电量， val值为为IEEE 754浮点数的32位整数表示， v值为浮点数，单位为度(kwh)。	R	注意：v值可以直接使用，若不存在v值，则需要手动转换。 其值类型为IEEE 754浮点数的32位整数布局。 例如 1024913643 表示的是浮点值：0.03685085，可以依据整数值计算出浮点值。 具体请参考： **附录3.4 IO值浮点类型说明**
	EP	功率	为当前负载功率， val值为为IEEE 754浮点数的32位整数表示， v值为浮点数，单位为w。	R	注意：v值可以直接使用，若不存在v值，则需要手动转换。 其值类型为IEEE 754浮点数的32位整数布局。 例如 1024913643 表示的是浮点值：0.03685085，可以依据整数值计算出浮点值。 具体请参考： **附录3.4 IO值浮点类型说明**
	T	温度	val值表示原始温度值， v值为实际值 （单位：℃）	R	
	H	湿度	val值表示原始湿度值， v值为实际值 （单位：%）	R	
	PM	PM2.5	val值表示PM2.5值， v值为实际值 （单位：ug/m³）	R	
	PMx	PM10	val值表示PM10值， v值为实际值 （单位：ug/m³）	R	
	COPPM	一氧化碳	val值表示CO浓度值， v值为实际值 （单位：ppm）	R	
	CO2PPM	二氧化碳	val值表示CO2浓度值， v值为实际值 （单位：ppm）	R	
	CH2OPPM	甲醛	val值表示甲醛原始浓度值，v值为实际值 （单位：ppm）	R	

	O2VOL	氧气	val值表示氧气原始浓度值，v值为实际值（单位：vol%）	R	
	NH3PPM	氨气	val值表示氨气原始浓度值，v值为实际值（单位：ppm）	R	
	H2SPPM	硫化氢	val值表示硫化氢原始浓度值，v值为实际值（单位：ppm）	R	
	TVOC	TVOC	val值表示TVOC原始浓度值，v值为实际值（单位：mg/m³）	R	
	PHM	噪音	val值表示噪音原始值，v值为实际值（单位：dB）		
	SMOKE	烟雾	val值表示烟雾原始浓度值，v值为实际值（单位：ppm）	R	

2.11 车库门（Garage Door）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

SL_ETDOOR	P1	灯光控制	type%2==1,表示打开(忽略val值)；
			type%2==0,表示关闭(忽略val值)；
				RW	打开，则下发：
					type=0x81,val=1;
					
					关闭，则下发：
					type=0x80,val=0;

	P2	车库门状态	type%2==1 表示控制正在运行； type%2==0 表示没有运行； 当正在运行的时候即(type%2==1)： val&0x80==0x80表示正在开，否则表示正在关； val&0x7F的值表示车库门打开的百分比([0,100])。	R	

	P3	车库门控制		W	完全打开，则下发：
					type=0xCF,val=100;
					
					完全关闭，则下发：
					type=0xCF,val=0;
					
					停止车库门开合，则下发：
					type=0xCE,val=0x80;
					
					开到百分比，则下发：
					type=0xCF,val=percent;
					percent取值：[0,100];

2.12 智能报警器(CoSS版)（Smart Alarm(CoSS)）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

SL_ALM	P1	播放控制	type%2==1,表示正在播放(忽略val值)；
			type%2==0,表示没有播放(忽略val值)；
			
			val为32bit值，描述如下(16进制)：
			0xAABBCCDD
			AABB表示时间或者循环次数(最高位1表示次数，否则为时间，时间单位为秒)；
			CC是音量(只有16级，使用高4位，若CC值等于0将采用P2 IO定义的音量值，否则将使用CC值做为音量值)；
			DD表示音频序号；
				RW	播放，则下发：
					type=0x81, val=1;
					
					停止，则下发：
					type=0x80, val=0;
					
					设置值并播放：
					type=0xff；val=需要设置的值；
					
					设置值并停止：
					type=0xfe；val=需要设置的值；
					
					设置值并保持当前播放状态：
					type=保持当前值；val=需要设置的值；

	P2	音量控制	type%2==1 表示处于正常模式；
			type%2==0 表示处于静音模式；
			
			val值表示音量值，只有16级，使用高4位。
			即有效位为：
			0x000000F0
				RW	要设置静音，则下发：
					type=0x80, val=0;
					
					要取消静音，则下发：
					type=0x81, val=1;
					
					要设置音量，则下发：
					type=保持当前值，
					val=音量值

2.13 摄像头（Smart Camera）

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述
cam	M	移动检测	val值定义如下： 0： 没有检测到移动 1:有检测到移动	R	
	V	电压	val表示原始电压值 v值将表示当前剩余电量百分比，值范围[0,100]，它是根据val电压值换算的。 注意：当前只有FRAME设备有该属性。	R	

	CFST	摄像头状态	val值定义如下： 按位表示值： • 第0位：表示是否有外接电源，1表示有外接电源，0表示没有； • 第1位：是否为旋转云台，1表示摄像头在旋转云台上，0表示没有； • 第2位：表示是否正在旋转，1表示正在旋转。 注意：当前只有FRAME设备有该属性。	R	

提示：如何查看摄像头的规格?

EpGetAll,EpGet接口返回的 dev_rt 属性将表示该摄像头的具体规格，当前有如下值：

- **LSCAM:LSCAMV1**： FRAME
- **LSCAM:LSICAMEZ1**： 户外摄像头
- **LSCAM:LSICAMEZ2**： 户外摄像头
- **LSCAM:LSLKCAMV1**： 视频门锁摄像头
- **LSCAM:LSICAMGOS1**：高清摄像头

2.14 超能面板系列（NATURE Series）

2.14.1 超级面板PRO(温控)

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

SL_NATURE	P1	开关	type%2==1,表示打开(忽略val值)；
			type%2==0,表示关闭(忽略val值)；
				RW	打开，则下发：
					type=0x81,val=1;
					关闭，则下发：
					type=0x80,val=0;

	P2	阀门状态	阀门1状态 （盘管的冷阀 或者盘管的冷热阀）	R	
	P3	阀门状态	阀门2状态 （盘管的热阀 或者地暖阀）	R	
	P4	T 当前温度	v值表示温度值 val值表示原始温度值，它是温度值*10	R	
	P5	设备种类	val&0xFF 指示设备种类。 • 1： 开关面板 • 2： POE面板 • 3： 温控面板 注意：值必须是3才是温控面板，否则是其它类型的设备。	R	
	P6	CFG 配置	(val>>6)&0x7 指示设备类型 • 0： 新风模式 • 1： 风机盘管（单阀）模式 • 2： 水地暖模式 • 3： 风机盘管+水地暖模式 • 4： 风机盘管（双阀）模式 • 5： 水地暖+新风模式	RW	下发命令的时候type保持原样，或者type值为0xFF，val值为需要设置的值 注意：需要保留其它位，仅仅设置需要设置的3个bit，即 newCfg&0x7<<6
	P7	MODE 模式	• 1： Auto 自动 • 2： Fan 通风 • 3： Cool 制冷 • 4： Heat 制热 • 7： DN 地暖 • 8： DN_Heat 地暖+空调 注意：P6 CFG配置不同，支持的MODE也会不同，具体请参考P7 MODE选项说明。	RW	下发命令的时候type保持原样，或者type值为0xCE，val需要设置的模式值

	P8	tT 目标温度	v值表示温度值 val值表示原始温度值，它是温度值*10	RW	下发命令的时候type保持原样，或者type值为0x88，val需要使用原始温度值，即目标温度值*10，例如下发设定温度为25摄氏度，则应该下发 val=250
	P9	tF 目标风速	val值表示风速，定义如下： • 0：Stop停止 • 0<val<30：Low低档 • 30<=val<65：Medium中档 • 65<=val<100：High高 • 101：Auto自动 注意：P6 CFG配置不同，支持的tF也会不同，具体请参考P9 tF选项说明。	RW	下发命令的时候type保持原样，或者type值为0xCE，val值如下： 0：Stop停止 15：Low低档 45：Medium中档 75：High高档 101：Auto自动
	P10	F 当前风速	val值表示风速，定义如下： • 0：Stop停止 • 0<val<30：Low低档 • 30<=val<65：Medium中档 • 65<=val<100：High高 • 101：Auto自动	R	

提示： 如何区分设备是超能温控面板?

设备的规格是SL_NATURE，并且存在P5属性，其值&0xFF等于3。

说明： P7 MODE选项说明

MODE可以选择的值依赖于CFG配置属性，定义如下：

0 新风模式： MODE不支持
1 风机盘管（单阀）模式： Fan,Cool,Heat
2 水地暖模式： MODE不支持
3 风机盘管+水地暖模式： Fan,Cool,Heat,DN,DN_Heat
4 风机盘管（双阀）模式： Auto,Fan,Cool,Heat,DN
5 水地暖+新风模式： Fan,DN

说明： P9 tF选项说明

tF可以选择的值依赖于CFG配置属性，定义如下：

0 新风模式： Low,Medium,High,Auto,Stop
1 风机盘管（单阀）模式： Low,Medium,High,Auto,Stop
2 水地暖模式： tF不支持
3 风机盘管+水地暖模式： Low,Medium,High,Auto,Stop
4 风机盘管（双阀）模式： Low,Medium,High,Auto,Stop
5 水地暖+新风模式： Low,Medium,High,Auto,Stop

2.14.2 超级面板PRO

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

SL_NATURE	P1, P2, P3	开关	type%2==1,表示打开(忽略val值);
			type%2==0,表示关闭(忽略val值);
			P1表示第一路开关控制口;
			P2表示第二路开关控制口;
			P3表示第三路开关控制口;
				RW	打开，则下发:
					type=0x81,val=1;
					
					关闭，则下发:
					type=0x80,val=0;

	P5	设备种类	val&0xFF 指示设备种类。 • 1：开关面板 • 2：POE面板 • 3：温控面板 注意：值必须是1才是温开关面板，否则是其它类型的设备。	R	

提示： 如何区分设备是超能开关面板?
设备的规格是SL_NATURE，若不存在P5属性，则缺省认为是超能开关面板。若存在P5属性，其值&0xFF等于1则认为是超能开关面板。

2.14.3 超级面板PRO(PoE)

超能面板PRO(PoE)采用PoE供电。

2.14.4 星玉温控面板(Nature Thermostat)

Devtype/Cls	IO idx	IO名称	属性值描述	RW	下发命令描述

	P1	开关	type%2==1,表示打开(忽略val值);
			type%2==0,表示关闭(忽略val值);
				RW	打开，则下发:
					type=0x81,val=1;
					
					关闭，则下发:
					type=0x80,val=0;

	P2	阀门状态	阀门1状态 （盘管的冷阀 或者盘管的冷热阀）	R	
	P3	阀门状态	阀门2状态 （盘管的热阀 或者地暖阀）	R	

SL_FCU	P4	T 当前温度	type==0x08, v值表示温度值 val值表示原始温度值，它是温度值*10	R	
	P6	CFG 配置	(val>>6)&0x7 指示设备类型 • 0：新风模式 • 1：风机盘管（单阀）模式 • 2：水地暖模式 • 3：风机盘管+水地暖模式 • 4：风机盘管（双阀）模式 • 5：水地暖+新风模式	RW	下发命令的时候type保持原样，或者type值为0xFF，val值为需要设置的值 注意：需要保留其它位，仅仅设置需要设置的3个bit，即 newCfg&0x7<<6
	P7	MODE 模式	• 1：Auto 自动 • 2：Fan 通风 • 3：Cool 制冷 • 4：Heat 制热 • 7：DN 地暖 • 8：DN_Heat 地暖+空调 注意：P6 CFG配置不同，支持的MODE也会不同，具体请参考P7 MODE选项说明。	RW	下发命令的时候type保持原样，或者type值为0xCF，val需要设置的模式值
	P8	tT 目标温度	v值表示温度值 val值表示原始温度值，它是温度值*10	RW	下发命令的时候type保持原样，或者type值为0x89，val需要使用原始温度值，即目标温度值*10，例如下发设定温度为25摄氏度，则应该下发 val=250
	P9	tF 目标风速	val值表示风速，定义如下： • 0：Stop停止 • 0<val<30：Low低档 • 30<=val<65：Medium中档 • 65<=val<100：High高 • 101：Auto自动 注意：P6 CFG配置不同，支持的tF也会不同，具体请参考P9 tF选项说明。	RW	下发命令的时候type保持原样，或者type值为0xCF，val值如下： 0：Stop停止 15：Low低档 45：Medium中档 75：High高档 101：Auto自动
	P10	F 当前风速	val值表示风速，定义如下： • 0：Stop停止 • 0<val<30：Low低档 • 30<=val<65：Medium中档 • 65<=val<100：High高 • 101：Auto自动	R	

说明： P7 MODE选项说明

MODE可以选择的值依赖于CFG配置属性，定义如下：

0 新风模式： MODE不支持

1 风机盘管（单阀）模式： Fan,Cool,Heat

2 水地暖模式： MODE不支持
3 风机盘管+水地暖模式： Fan,Cool,Heat,DN,DN_Heat
4 风机盘管（双阀）模式： Auto,Fan,Cool,Heat,DN
5 水地暖+新风模式： Fan,DN

说明： P9 tF选项说明

tF可以选择的值依赖于CFG配置属性，定义如下：

0 新风模式： Low,Medium,High,Auto,Stop
1 风机盘管（单阀）模式： Low,Medium,High,Auto,Stop
2 水地暖模式： tF不支持
3 风机盘管+水地暖模式： Low,Medium,High,Auto,Stop
4 风机盘管（双阀）模式： Low,Medium,High,Auto,Stop
5 水地暖+新风模式： Low,Medium,High,Auto,Stop

3 附录

3.1 智慧设备规格名称

Devtype/Cls ***<<fullCls>>***	**Name**
插座系列	
SL_OL	智慧插座
SL_OL_3C	智慧插座
SL_OL_DE	德标插座
SL_OL_UK	英标插座
SL_OL_UL	美标插座
OD_WE_OT1	Wi-Fi插座
SL_OL_W	入墙插座
SL_OE_3C	计量插座
SL_OE_DE	计量插座德标
开关系列	
SL_SW_IF1	流光开关一键
SL_SW_IF2	流光开关二键
SL_SW_IF3	流光开关三键
SL_SF_IF1	单火流光开关一键
SL_SF_IF2	单火流光开关二键
SL_SF_IF3	单火流光开关三键
SL_SW_CP1	橙朴流光开关一键
SL_SW_CP2	橙朴流光开关二键
SL_SW_CP3	橙朴流光开关三键
SL_SW_FE1	塞纳/格致开关一键
SL_SW_FE2	塞纳/格致开关二键
SL_SW_RC	触摸开关，极星开关(零火版)
SL_SF_RC	单火触摸开关
SL_SW_RC1	白玉/墨玉流光开关一键
SL_SW_RC2	白玉/墨玉流光开关二键

SL_SW_RC3	白玉/墨玉流光开关三键
SL_SW_ND1	恒星/辰星/极星开关一键
SL_SW_ND2	恒星/辰星/极星开关二建
SL_SW_ND3	恒星/辰星/极星开关三键
SL_MC_ND1	恒星/辰星/极星开关伴侣一键
SL_MC_ND2	恒星/辰星/极星开关伴侣二键
SL_MC_ND3	恒星/辰星/极星开关伴侣三键
SL_S	开关智控器
SL_SPWM	可调亮度开关智控器
SL_SC_BB	随心开关
SL_P_SW	九路开关控制器
SL_SW_MJ1	奇点开关模块一键
SL_SW_MJ2	奇点开关模块二键
SL_SW_MJ3	奇点开关模块三键
SL_SW_NS1	视界触摸开关一键
SL_SW_NS2	视界触摸开关二键
SL_SW_NS3	视界触摸开关三键
SL_SW_BS1	极星开关（120零火版）一键
SL_SW_BS2	极星开关（120零火版）二键
SL_SW_BS3	极星开关（120零火版）三键
窗帘控制系列	
SL_CN_IF	流光窗帘控制器
SL_CN_FE	格致/塞纳三键窗帘
SL_SW_WIN	窗帘控制器
SL_DOOYA	窗帘电机
SL_DOOYA/ *<<SL_DOOYA_V2>>*	速接窗帘电机
SL_DOOYA/ *<<SL_DOOYA_V3>>*	卷帘电机
SL_DOOYA/ <<SL_DOOYA_V4>>	卷帘电机(锂电池)
SL_P_V2	智界窗帘电机智控器
灯光系列	
SL_LI_RGBW	胶囊灯泡

SL_CT_RGBW	幻彩灯带
SL_SC_RGB	幻彩灯带（不带白光）
OD_WE_QUAN	量子灯
SL_LI_WW	白光智能灯泡
SL_LI_WW/ <<SL_LI_WW_V2>>	调光调色控制器 SL_LI_WW_V1:智能灯泡(冷暖白) SL_LI_WW_V2:调光调色智控器(0-10V)
SL_LI_GD1	调光壁灯
SL_LI_UG1	花园地灯
超级碗系列	
MSL_IRCTL	超级碗（基础版,蓝牙版）
OD_WE_IRCTL	超级碗（闪联版）
SL_SPOT	超级碗（CoSS版）
SL_P_IR	红外模块
SL_P_IR/ <<SL_P_IR_V2>>	超级碗（Mini版）
感应器系列	
SL_SC_G	门禁感应器
SL_SC_BG	多功能(CUBE)门禁感应器
SL_SC_MHW	动态感应器
SL_SC_CM	动态感应器（7号电池版）
SL_SC_BM	多功能(CUBE)动态感应器
SL_P_RM	人体存在感应器
SL_SC_THL	环境感应器
SL_SC_BE	多功能(CUBE)环境感应器
SL_SC_CQ	环境感应器(CO2+TVOC)
SL_SC_CA	环境感应器(CO2)
SL_SC_WA	水浸感应器
SL_SC_CH	气体感应器(甲醛)
SL_SC_CP	气体感应器(燃气)
ELIQ_EM	ELIQ电量计量器
SL_SC_CV	语音小Q
SL_P_A	烟雾感应器

SL_DF_GG	云防门窗感应器（DEFED Window/Door）
SL_DF_MM	云防动态感应器（DEFED Motion）
SL_DF_SR	云防室内警铃（DEFED Indoor Siren）
SL_DF_BB	云防遥控器（DEFED Key Fob）
SL_SC_CN	噪音感应器(Noise Sensor)
温控器系列	
V_AIR_P	智控器空调面板
SL_CP_DN	地暖温控器
SL_CP_AIR	风机盘管
SL_TR_ACIPM	新风系统
SL_CP_VL	温控阀门
门锁系列	
SL_LK_LS	智能门锁
SL_LK_GTM	智能门锁 耶鲁门锁
SL_LK_AG	智能门锁 西勒奇锁模块
OD_JIUWANLI_LOCK1	九万里门锁
SL_P_BDLK	必达门锁模块
SL_LK_SG	思哥智能门锁
SL_LK_YL	Yale/Gateman门锁模块
摄像头系列	
LSCAM:LSICAMGOS1	高清摄像头系列，可以根据ModelKey区分具体规格类型，可以从EpGetAll/EpGet接口返回设备属性"modelKey"。modelKey定义如下: 0xd2：高清摄像头 0xd4,0xd7,0xd9：1080P高清摄像头 0xda：云视户外摄像头 0xdb：云瞳室内摄像头 0xdc：云瞳室外摄像头
LSCAM:LSICAMEZ2	户外摄像头
LSCAM:LSCAMV1	FRAME摄像头
其它	
SL_P	通用控制器
OD_MFRESH_M8088	空气净化器
LSSSMINIV1	多功能报警器

SL_ETDOOR	车库门
SL_ALM	智能报警器(CoSS版)
V_DLT645_P	DLT电量计量器
V_485_P	485控制器

注：<<fullCls>>是可选的，如果列表里面指明了fullCls，则说明该设备有多个版本形态，不同版本有不同的名称，需要通过fullCls来区分各个版本的定义。

3.2 动态颜色（DYN）定义

效果	val
青草	0x8218cc80
海浪	0x8318cc80
深蓝山脉	0x8418cc80
紫色妖姬	0x8518cc80
树莓	0x8618cc80
橙光	0x8718cc80
秋实	0x8818cc80
冰淇淋	0x8918cc80
高原	0x8020cc80
披萨	0x8120cc80
果汁	0x8a20cc80
温暖小屋	0x8b30cc80
魔力红	0x9318cc80
光斑	0x9518cc80
蓝粉知己	0x9718cc80
晨曦	0x9618cc80
木槿	0x9818cc80
缤纷时代	0x9918cc80
天上人间	0xa318cc80
魅蓝	0xa718cc80
炫红	0xa918cc80

3.3 量子灯特殊（DYN）定义

效果	val
马戏团	0x04810130
北极光	0x04c40600
黑凤梨	0x03bc0190
十里桃花	0x04940800
彩虹糖	0x05bd0690
云起	0x04970400
日出印象	0x01c10a00
马卡龙	0x049a0e00
光盘时代	0x049a0000
动感光波	0x0213a400
圣诞节	0x068B0900
听音变色(第二代量子灯才支持)	0x07bd0990

3.4 IO值浮点类型说明

IEEE 754规范定义的浮点数可以采用32为整数来表示。在获取真实浮点数的时候，可以从32位整数转换出来。它仅适用于32位单精度数字。并且它不会返回负零。
32位整数定义如下：
第 31 位（掩码 0x80000000 选定的位）表示浮点数的符号。
第 30-23 位（掩码 0x7f800000 选定的位）表示指数。
第 22-0 位（掩码 0x007fffff 选定的位）表示浮点数的有效位数（有时也称为尾数）。
如果参数为正无穷大，则结果为 0x7f800000。
如果参数为负无穷大，则结果为 0xff800000。
如果参数为 NaN，则结果是表示实际 NaN 值的整数

假设某个IO数据为 {type:IO_TYPE,val:IO_VAL}

3.4.1 如何判断IO值是否为浮点类型?

只有满足下面条件的IO数据其值IO_VAL才是浮点类型数据

```
(IO_TYPE & 0x7e) == 0x2

例如：  {type:3, val: 1024913643}
```

3.4.2 如何转换为真实的浮点值

假设 {type:3, val: 1024913643}

- **Java代码示例：**

```
float realVal = Float.intBitsToFloat(1024913643)
// 返回 0.03685085
```

- **Python代码示例：**

```
realVal = struct.unpack('!f', struct.pack('!i', 1024913643))[0]
// 返回 0.03685085102915764
```

- **JavaScript代码示例：**

```
var ieee32ToFloat = function(intval) {
  var fval = 0.0;
  var x;//exponent
  var m;//mantissa
  var s;//sign
  s = (intval & 0x80000000)?-1:1;
  x = ((intval >> 23) & 0xFF);
  m = (intval & 0x7FFFFF);
  switch(x) {
      case 0:
          //zero, do nothing, ignore negative zero and subnormals
          break;
      case 0xFF:
          if (m) fval = NaN;
          else if (s > 0) fval = Number.POSITIVE_INFINITY;
          else fval = Number.NEGATIVE_INFINITY;
          break;
      default:
          x -= 127;
          m += 0x800000;
          fval = s * (m / 8388608.0) * Math.pow(2, x);
          break;
  }
  return fval;
}

var realVal = ieee32ToFloat(1024913643)
// 返回: 0.03685085102915764
```

3.5 IO实际值转换及Type定义说明

IoT应用中常见的IO数据有很多种类，描述设备状态的数据如：

* 开关的状态：1个bit位即可表示

* 温湿度：1位小数位的浮点数字
* 彩灯颜色：RGB/RGBW(4字节)
* 一副图像：长度可达上百K字节
* 一段声音：长度可达上百K字节

描述设备规格的数据：

* 输入 / 输出：用来表示数据流向
* 模拟量精度：用来描述AD采样精度(1-32bit的数值)

为了统一的协议层处理，我们定义一种统一的标准来描述这些数据，常用的做法是采用TLV格式（type+length+value)。考虑到射频和串口协议的带宽资源及其有限，MCU的RAM资源也十分紧张，TLV数据在系统数据库中保存和分析的开销也会很重。在CoSS系统中我们设计了一套巧妙的数据表示方式，对于各种形式的IoT数据，采用此方式统一描述，同时有以下优点：

* 表示数据的同时还能描述数据的精度 / 方向
* 同时包含模拟量和开关量信息（例如灯的开关状态和当前的亮度，开关和亮度信息不冲突）
* 对于非大数据块的数据，上层可以用单一列的数值(int64/double)来进行存储和分析

CoSS系统中对于数据，用TYPE+VAL的方式进行表示， TYPE是一个8位(1个字节)的数字， 用来表示数据的类型，VAL是额外的数据信息，VAL的长度由TYPE的值决定。

TYPE定义

bit位	说明	备注
bit7	表示数据流向，输入/输出	1：输出，0：输入 比如设置灯的开关，一般为1输出； 比如动态感应器的状态上报，一般为0输入；
bit6-1	描述数据类型/精度	
bit0	表示开关状态	1：打开，0：关闭

TYPE(bit6-1)定义描述

bit6-bit1值	VAL长度(byte)	用途
001XXX	待定(XXX不能都为1)	
001111	0	无效数据，用来表示异常
000000	0	开关量数据，只需要type的bit0来表示开关状态
1XXXXX	1-4	模拟量数据，XXXXX表示精度，实际精度=XXXXX+1个数目的bit位数，通常用来表示1-32位的二进制数； 如：二进制XXXXX为11111时，实际精度为31+1=32，所以XXXXX为11111时，实际精度为32位二进制数。
0001XX	2	16位整形表示的浮点数，XX是10进制的小数位， XX=0时，精度为1个小数位，.0； XX=1时，精度为2个小数位，.00； XX=2时，精度为3个小数位，.000； XX=3时，精度为4个小数位，.0000；
000001	4	Float32

3.5.1 如何判断IO实际值是否需要手动转换?

假设某个IO数据为 IO: { type:IO_TYPE, val:IO_VAL, v:REAL_VAL? }
一般而言使用接口获取当前IO数据时，后台会自动进行IO实际值的转换，接口返回的IO字段键为"v"，若接口调用获取IO详情中没有包含该字段信息时，UI需要显示该IO的实际值（友好值）时，则需要通过IO_TYPE和IO_VAL进行转换。

某个获取设备信息的API请求回应中关于IO信息片段如下:

```
{
    "cls": "V_485_P",
    "data": {
        "H2SPPM": {
            "type": 10,
            "v": 0.05,
            "val": 5,
            "valts": 1656492389816
        },
        "PM": {
            "type": 94,
            "v": 26,
            "val": 26,
            "valts": 1656492389816
        },
        "TVOC": {
            "type": 10,
            "v": 0.23,
            "val": 23,
            "valts": 1656492389816
        }
    },
    "name": "环境",
    "stat": 1,
    ...
}
```

如上，其中IO TVOC的v后台进行了自动转换则不需要手动进行转换。
只有IO数据中IO.v的值不存在时需要进行转换，我们假设转换的函数名字为"getIoFriendVal"

```
IO: { type:IO_TYPE, val:IO_VAL, v:REAL_VAL }
// IO数据存在实际值 IO.v 字段信息，不需要手动转换


IO: { type:IO_TYPE, val:IO_VAL }
// IO数据不存在实际值 IO.v，需要手动调用转换函数，
// 使得 IO.v = getIoFriendVal(IO_TYPE, IO_VAL)
```

建议在需要使用IO实际值的地方进行类似如下的取值:

```
If (not IO.v) then
     /* getIoFriendVal 在转换时可以存在异常数据，需要进行有效性判定，
      * 具体以getIoFriendVal的非法返回值来进行判定，这里只是伪代码示例
```

```
      */
     IO.v = getIoFriendVal(IO_TYPE, IO_VAL)
     if (not IO.v) then
          IO.v = IO_VAL
     end
end
```

3.4.2 如何转换IO的实际值

假设 `IO: { type:136, val:250 }`
如下示例代码，其中已经包含IO值浮点类型的判定和转换：

- **Java代码示例：**

```
/**
 * 获取该IO值的精度，用于val与friendVal的相互转换
 * 常规的有：10, 100, 1000, 10000。
 */
public static int getIoValPrecision(int ioType) {
    if ((ioType & 0x78) == 0x08) {
        int precision = (ioType & 0x06) / 2;
        if (precision == 0) return 10;
        else if (precision == 1) return 100;
        else if (precision == 2) return 1000;
        else if (precision == 3) return 10000;
    }
    return 1;
}

/**
 * 异常数据返回null
 */
public static Double getIoFriendVal(int ioType, int ioVal) {
    if (ioType == 0x1E) return null;
    if ((ioType & 0x7e) == 0x2) return Float.intBitsToFloat(ioVal);

    int precision = getIoValPrecision(ioType);
    return (ioVal>0x7fff ? (ioVal-0x10000) : ioVal)/(double)precision;
}

IO.v = getIoFriendVal(136, 250)
// 返回 25.00
```

- **Python代码示例：**

```
import struct

# 异常数据返回-1
def get_io_friend_val(io_type, io_val):
    if not isinstance(io_type, int) or not isinstance(io_val, int):
        return -1
    if io_type == 0x1E:
        return -1
    if io_type & 0x7e == 0x2:
        return struct.unpack('!f', struct.pack('!i', io_val))[0]
```

```
    else:
        precision = 1
        if io_type & 0x78 == 0x08:
            precision = (io_type & 0x06) / 2 + 1
            precision = math.pow(10, precision)
        if io_val > 0x7fff:
            io_val = io_val - 0x10000
        return io_val / precision

IO.v = get_io_friend_val(136, 250)
// 返回 25.0
```

- **JavaScript代码示例：**

```
/**
 * 异常数据返回-1 */
function getIoFriendVal(ioType, ioVal) {
  if (typeof ioType !== 'number')
    return -1;
  if (ioType === 0xE1)
    return -1;
  if ((ioType & 0x7e) == 0x2) {
     return ieee32ToFloat(ioVal);
  }
  var precision = 1;
  if ((ioType & 0x78) == 0x08) {
    var p = (ioType & 0x06) / 2;
    if (p == 0) precision = 10;
    else if (p == 1) precision = 100;
    else if (p == 2) precision = 1000;
    else if (p == 3) precision = 10000;
  }

  return ioVal / precision;
}

IO.v = getIoFriendVal(136, 250)
// 返回: 25
```