Wi-Fi 断电快联串口协议

协议生成时间：2025年05月22日 00:17

产品信息

产品名称：RA4M2_IOT

产品ID(PID)：63pnfirmrslxtur8

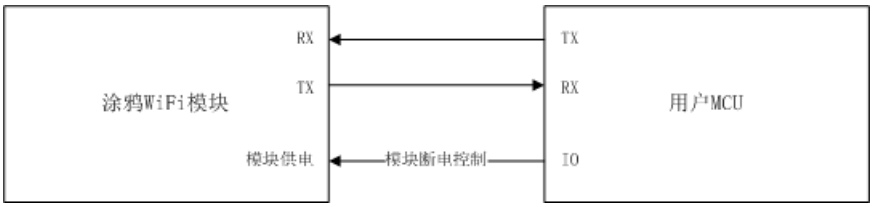
产品功能：

dpID	功能名称	数据传输类型	数据类型	功能属性	备注
3	电池电量	只上报	value	数值范围：0-100，间距：1，单位：%	
8	当前温度	只上报	value	数值范围：-400-2000，间距：1，单位：null	
10	震动状态	只上报	enum	枚举范围：normal,vibration,drop,tilt	

类型说明：

布尔型(bool)：非真即假的二值型变量，如开关功能；
数值型(value)：可线性调节数值型的功能，如温度调节20-40℃；
枚举值(enum)：自定义的有限集合值，如档位的高、中、低；
故障型(fault)：专门用于上报和统计故障的功能点，支持多故障，数据只上报。故障值，是按照对应的字节位来标识的，依次为bit0-bitx。从左到右依次对应，如E1(bit0)、E2(bit1)、F1(bit3)。多个字节位代表多个故障可同时发生，最多支持30位；
字符串型(string)：以字符串形式传输的功能点；
透传型(raw)：以二进制形式透传的功能点。

涂鸦 Wi-Fi 串口通用协议为涂鸦定制的 Wi-Fi 模块串口通用协议，主要用于涂鸦 Wi-Fi 模块与其它 MCU 串口直连做串口通信，其架构框图如下图所示。



实际开发中注意点：

- 整套协议适用于使用 Wi-Fi 模块且没法使用外电的设备。开发 MCU 程序的时候，断电管理尤为重要。在可以完成功能的前提下，尽量减少Wi-Fi 模块的上电时间，这是节约设备功耗的关键。对于数据上传，我们给出了大体的控制流程图。开发者可以根据自己设备的特性来选择协议中需要用到的功能。开发过程中，开发者可以根据自己的需要实时调整相关控制逻辑。
- 很多日常设备即使不开启 Wi-Fi 功能，也可以正常使用。用户拿到设备时，可能由于种种原因而不使用 Wi-Fi 功能。为了防止这种情况下给 Wi-Fi 模块上电而造成电能损耗，我们在设备端可以设计一个物理按键或者相关选项。只有当用户主动打开这个按键或者选项，MCU 才在每次数据变动时给 Wi-Fi 模块上电传输相关数据。

串口通信约定

- 波特率：9600
- 数据位：8
- 奇偶校验：无

- 停止位：1
- 数据流控：无

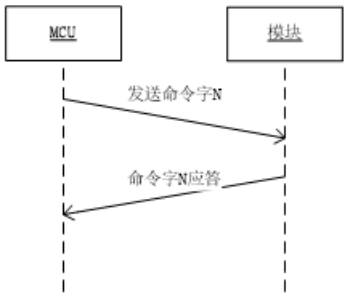
MCU：用户控制板控制芯片，与涂鸦模块通过串口对接。协议设计上，所有包的交互为全双工通信设计。

帧格式说明

字段	长度（byte）	说明
帧头	2	固定为 0x55aa
版本	1	升级扩展用
命令字	1	具体帧类型
数据长度	2	大端
数据	N	/
校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

说明：

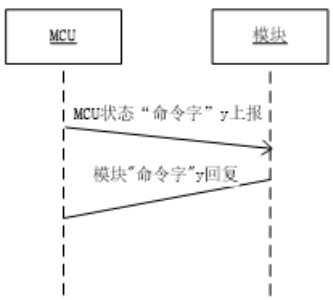
- 所有大于 1 个字节的数据均采用大端模式传输。
- 协议中所有举例说明数据都是十六进制数据。
- Wi-Fi 模块主动发起的通信发送包超时时间为 1S，重发机制会重发三包。
- 一般情况下，采用同命令字一发一收同步机制，即一方发出命令，另一方应答。若发送 方超时未收到正确的响应包，则超时传输，如下图所示：



说明：具体通信方式以“协议详述”章节中为准

- MCU 状态上报则采用同步模式，MCU 状态上报“命令字”为 y，如下所示：

  统计数据上报：



状态数据单元

说明：

- datapoint 命令/状态数据单元如下所示：

数据段	长度（byte）	说明
dpid	1	datapoint 序号
type	1	对应开放平台上某 datapoint 具体的数据类型，详细信息见如下type字段说明
len	2	长度对应 value 的字节数，详细信息见如下type字段说明
value	1/2/4/N	hex 表示，大于 1 字节采用大端传输

type字段说明：

type	类型	对应长度len（字节）	说明
0x00	raw	N	对应于 raw 型 datapoint（模块透传）
0x01	bool	1	范围：0x00/0x01
0x02	value	4	对应 int 类型，大端表示
0x03	string	N	对应于具体字符串
0x04	enum	1	枚举类型，范围 0-255
0x05	bitmap	1/2/4	长度大于 1 字节时，大端表示

- datapoint 命令/状态数据单元除”raw”类型外，其他类型均属于“obj”型 datapoint
- “状态数据”可含多个 datapoint“命令数据单元”

协议详述

查询产品信息

说明：

- product id：对应涂鸦开发者平台 PID(产品标识)，由涂鸦云开发者平台生成，用于云端记录产品相关信息。
- 产品信息由 product id （pid）和 MCU 软件版本构成。
- MCU 软件版本号格式定义：采用点分十进制形式，”x.x.x”（0<=x<=99），x 为十进制数。

模块发送：

字段	长度（byte）	说明
帧头	2	0x55aa
版本	1	0x00
命令字	1	0x01
数据长度	2	0x0000
数据	0	无
校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

模块发送：

55 aa 00 01 00 00 00

MCU 返回：

字段	长度（byte）	说明
帧头	2	0x55aa
版本	1	0x00
命令字	1	0x01
数据长度	2	N
数据	N	{“p”:”vHXEcqntLpkAl***”, “v”:”1.0.0”}
校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

例：{“p”:”vHXEcqntLpkAl***”,“v”:”1.0.0”}

p 表示产品 ID 为 vHXEcqntLpkAl***，参数为用户在涂鸦官网创建的产品 PID

v 表示 MCU 版本为 1.0.0

MCU 返回例子信息:

55 aa 00 01 00 24 7b 22 70 22 3a 22 76 48 58 45 63 71 6e 74 4c 70 6b 41 6c 4f 73 79 22 2c 22 76 22 3a 22 31 2e 30 2e 30 22 7d bf

报告设备联网状态

设备联网状态	描述	状态值
状态 1	smartconfig 配置状态	0x00
状态 2	AP 配置状态	0x01
状态 3	Wi-Fi 已配置但未连上路由器	0x02
状态 4	Wi-Fi 已配置且连上路由器	0x03
状态 5	已连上路由器且连接到云端	0x04

说明：

- 当模块的 Wi-Fi 状态发生变化，则会主动下发 Wi-Fi 状态至 MCU。
- 状态包用于 MCU 设备得知模块目前的状态，两种配置状态需要 MCU 做相关的配网显示。
- 原则上我们要求用户做两种配网方式，第一种配网方式存在少部分路由 兼容性问题，第二种配网方式是确保设备可以联网成功的保障。

模块发送：

字段	长度（byte）	说明
帧头	2	0x55aa
版本	1	0x00
命令字	1	0x02

数据长度	2	0x0001
数据	1	指示 Wi-Fi 工作状态： 0x00:状态 1 0x01:状态 2 0x02:状态 3 0x03:状态 4 0x04:状态 5
校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

模块发送：

55 aa 00 02 00 01 04 06（已连上路由器且连接到云端）

MCU 返回：

字段	**长度（byte）**	**说明**
帧头	2	0x55aa
版本	1	0x00
命令字	1	0x02
数据长度	2	0x0000
数据	0	无
校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

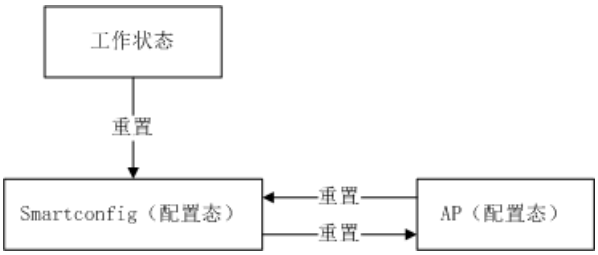
MCU 返回：

55 aa 00 02 00 00 01

重置Wi-Fi

说明：

- 重置 Wi-Fi 状态转化如下图所示：



MCU 发送：

字段	**长度（byte）**	**说明**
帧头	2	0x55aa
版本	1	0x00
命令字	1	0x03
数据长度	2	0x0000
数据	0	无

校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

MCU 发送：

55 aa 00 03 00 00 02

模块返回：

字段	长度（byte）	说明
帧头	2	0x55aa
版本	1	0x00
命令字	1	0x03
数据长度	2	0x0000
数据	0	无
校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

模块返回：

55 aa 00 03 00 00 02

重置 Wi-Fi-选择配置模式

说明：

- 相对于“重置Wi-Fi”而言，此帧提供 MCU 根据自身需求选择性地设置重置Wi-Fi 后的配置方式。
- MCU 接入用户可选择性的实现该协议。

MCU 发送：

字段	长度（byte）	说明
帧头	2	0x55aa
版本	1	0x00
命令字	1	0x04
数据长度	2	0x0001
数据	1	0x00：进入 smartconifg 配置 模式 0x01：进入 AP 配置模式
校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

MCU 发送控制模块进入 AP 配置模式:

55 aa 00 04 00 01 01 05

模块返回：

字段	长度（byte）	说明
帧头	2	0x55aa

版本	1	0x00
命令字	1	0x04
数据长度	2	0x0000
数据	0	无
校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

模块返回：

55 aa 00 04 00 00 03

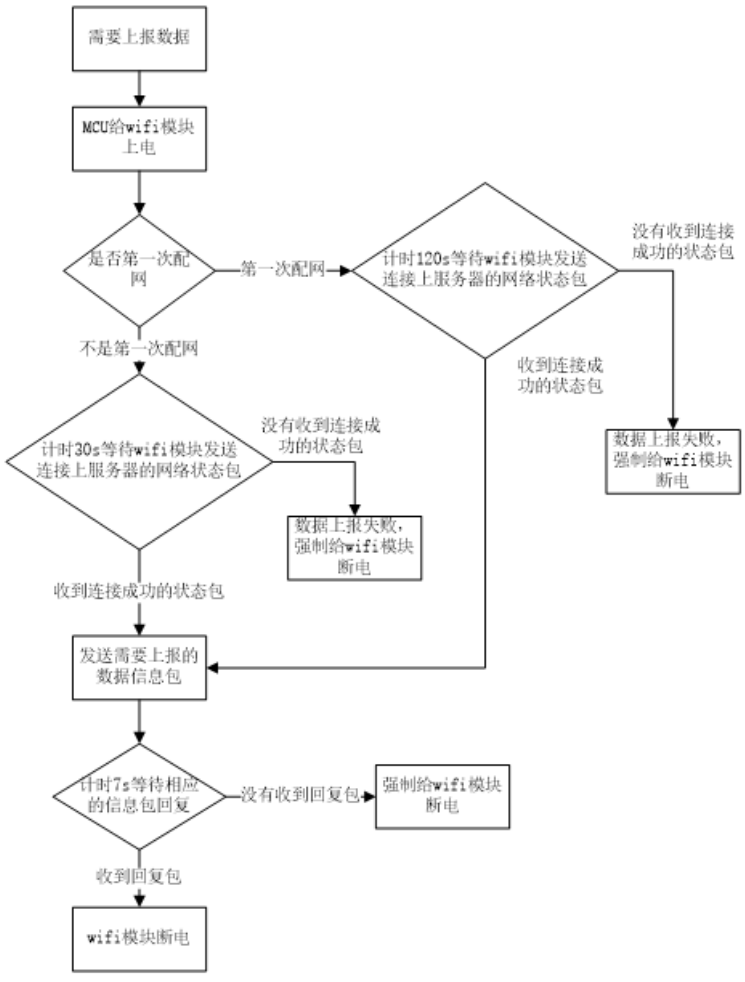
实时状态上报

使用场景：对于类似报警器要求实时推送的设备，可以通过本条命令上报相关状态数据。

说明：当 MCU 需要上报实时状态的数据时，可通过该条协议上报数据。

- 状态数据会直接上报到云端，故设备联网状态必须为已连接到云端，否则数据上报失败 。
- 该条指令为同步指令，MCU 数据上报后需等待模块返回上报结果，等待超时的时间为 7 秒。
- 需要等待模块发送连接上服务器的网络状态包，MCU再发送需要发送的统计数据包，这里的数据上传没有储存功能。MCU等待一段时间后还没收到连接上服务器的网络状态包，也强制给设备断电。等待时间受网络环境和是否是第一次配网影响：如果是第一次配网，从WiFi已配置（状态3）到连接到云端（状态5）时间较长，原因是设备需要激活、初始化等配置，网络差的情况下会更长，建议等待120秒；如果不是第一次配网（已配网），考虑到网络差的环境下，等待时间会变长，所以建议等待30秒。
- 模块可以支持多个数据单元上报，和单个数据单元上报，用户根据自己的需要来选择发送组包的方式。具体数据包可以参考下面的例子。

大致操作的流程图：



MCU 发送：

字段	长度（byte）	说明
帧头	2	0x55aa
版本	1	0x00
命令字	1	0x05
数据长度	2	取决于“状态数据单元”类 型以及个数
数据	N	一个或多个组合“状态数据单元”组
校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

模块返回：

字段	长度（byte）	说明
帧头	2	0x55aa
版本	1	0x00
命令字	1	0x05
数据长度	2	0x0001
数据	1	0x00 成功 0x01 失败
校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

“单个状态数据单元”上报例子：

DP 点 109 bool 型变量，数值为 1

55 aa 00 05 00 05 6d 01 00 01 01 79

“多个状态数据单元”上报例子：

DP 点 109 bool 型变量，数值为 1

DP 点 102 string 型变量，“201804121507”（具体传输对应 ASCII 值）

55 aa 00 05 00 15 6d 01 00 01 01 66 03 00 0c 32 30 31 38 30 34 31 32 31 35 30 37 5d

记录型状态上报(带记录储存功能)

使用场景：对于类似门锁等记录型的设备，包含多个 DP 点数据需要服务端作为整条记录处 理，在短暂断网的情况下本条命令会保存下无法上报成功的数据，本条命令可以 满足记录型设备的上报需求。

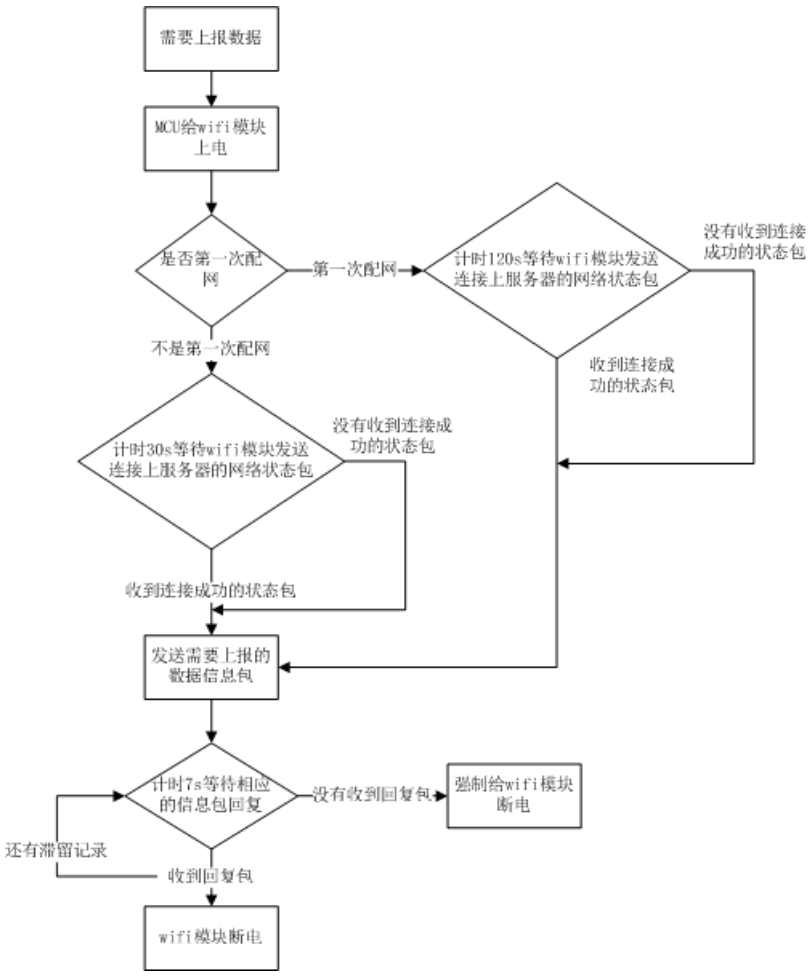
说明：当 MCU 需要记录型数据是由多个 DP 点组合的整条数据，需要整体上报，便可以通 过这条命令往 Wi-Fi 模块发送，并且在有数据上报，而此时设备断网了的时候，这条数据会储存下来。当下次有数据上报的时候，模块再上传这条数据，之后并把之前的储存数据上报成功。

- 当有滞留记录每次上报成功一条，模块就会主动发送一条 09 命令字数据为 01 的回复包。MCU 可以根据这个回复做超时断电处理！当发送一条记录后，强制等待 7s 。如果没有收到模块任何回复，则认为模块异常也要断电模块.
- 网络正常的情况下，模块上电后，正常需要4S左右连接上服务器。每当有记录产生给设备上电后，可以等WIFI模块发送协议文档所描述的的设备联网状态包。等待时间受网络环境和是

否是第一次配网影响：如果是第一次配网，从WiFi已配置（状态3）到连接到云端（状态5）时间较长，原因是设备需要激活、初始化等配置，网络差的情况下会更长，建议等待120秒；如果不是第一次配网（已配网），考虑到网络差的环境下，等待时间会变长，所以建议等待30秒。当等待时间已到还没收到模块连接上服务器的包，也将该条指令上报，MCU数据上报后需等待模块返回上报结果。

- 单次记录上报的数据区域(多个状态数据单元)最大长度为 80，根据实际的 DP点的数据，最后组合实际储存的长度会有所变化。当没有网络的状态下，超过限制长度，Wi-Fi 模块会返回记录发送失败的结果。
- 最多可以储存20 条历史记录。当历史记录超过 20 条，从最早储存的记录开始覆盖， 如此循环覆盖。
- 当 Wi-Fi 模块接收到一条数据成功推送，或者当没有网络状态下，记录成功存储进 flash 中，也会当做推送成功（00）。当有网络时推送成功一条，还有滞留记录的话，会返回 01。其他的情况均返回推送失败 02。
- 时间数据：当我们需要服务器显示上传数据的时间，以本地上传的时间为准，MCU 在发送的时候把本地时间数据带上，把本地时间协议中的 Data[0]时间标志位设置为 1，并按协议要求传输相关数据。当 Wi-Fi 模块检测到时间标志位为 1，便会以 MCU 传输的时间为准（**注意保证时间数据正确**）

上报大致操作流程图：



MCU 发送：

字段	**长度（byte）**	**说明**
帧头	2	0x55aa
版本	1	0x00
命令字	1	0x08
数据长度	2	取决于“状态数据单元”类 型以及个数类型以及个数

数据	7	数据长度为 7 字节: Data[0]为这条数据传输是否带本地时间标志位: 0 表示这条数据不带本地时间，后面的时间模块认为数据无效，不处理 1 表示后面的时间数据有效 Data[1] 为 年份，0x00 表示 2000 年 Data[2]为月份，从 1 开始到 12 结束 Data[3]为日期，从 1 开始到 31 结束 Data[4]为时钟，从 0 开始到 23 结束 Data[5]为分钟，从 0 开始到 59 结束 Data[6]为秒钟，从 0 开始到 15 结束
	N	一个或多个组合“状态数据单元”组
校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

模块返回：

字段	**长度（byte）**	**说明**
帧头	2	0x55aa
版本	1	0x00
命令字	1	0x08
数据长度	2	0x0001
数据	1	0x00 上报成功 0x01 当前记录上报成功，并 且还有滞留记录需要上报 0x02 上报失败
校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

“单个状态数据单元”上报例子： DP 点 109 bool 型变量，数值为 1

55 aa 00 08 00 0c 01 12 04 13 0d 03 1d 6d 01 00 01 01 da//本地时间为准

55 aa 00 08 00 0c 00 12 04 13 0d 04 14 6d 01 00 01 01 d1//服务端时间为准

“多个状态数据单元”上报例子：

DP 点 109 bool 型变量，数值为 1

DP 点 102 string 型变量，“201804121507”（具体传输对应 ASCII 值）

55 aa 00 08 00 1c 00 12 04 13 0d 06 04 6d 01 00 01 01 66 03 00 0c 32 30 31 38 30 34 31 32 31 35 30 37 a7 //服务端时间为准

55 aa 00 08 00 1c 01 12 04 13 0d 08 2e 6d 01 00 01 01 66 03 00 0c 32 30 31 38 30 3431 32 31 35 30 37 d4 //本地时间为准

模块命令下发

说明:

“命令下发”为异步处理协议，MCU 收到相关控制包，确认接收到回复下发包后，完成相关的控制动作，MCU 状态反馈通过 MCU 的状态上报来实现。

模块发送：

字段	**长度（byte）**	**说明**
帧头	2	0x55aa
版本	1	0x00

命令字	1	0x09
数据长度	2	取决于“命令数据单元”类 型以及个数
数据	N	“状态数据单元”组
校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

模块下发控制命令：（系统开关对应 3 号 DP,使用 bool 型变量，开机数值为 1）

55 aa 00 09 00 05 03 01 00 01 01 13

MCU 回复：

字段	**长度（byte）**	**说明**
帧头	2	0x55aa
版本	1	0x00
命令字	1	0x09
数据长度	2	0x0000
数据	0	无
校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

MCU 确认回复:

55 aa 03 09 00 00 0b

获取本地时间

说明:

1. 需要等待设备连接上网络（也就是收到 设备联网状态 命令 状态 5）后开始发送获取本地时间的数据包
2. 在设备连接上服务器后，在网络情况非常差的情况下，有可能会获取时间数据失败。对于依赖时间的设备（例如门锁），如果本地时间没有校准过，当获取时间数据失败时，需要做一个间隔 3s 的获取时间值，确保时间数据获取成功。

MCU 发送：

字段	**长度（byte）**	**说明**
帧头	2	0x55aa
版本	1	0x00
命令字	1	0x06
数据长度	2	0x0000
数据	0	无
校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

MCU 获取本地时间：

55 aa 00 06 00 00 05

模块返回：

字段	长度（byte）	说明
帧头	2	0x55aa
版本	1	0x00
命令字	1	0x06
数据长度	2	0x0008
数据	Data	数据长度为 8 字节: Data[0]为是否获取时间成功 标志，为 0 表示失败，为 1 表示成功 Data[1] 为 年 份， 0x00 表示 2000 年 Data[2]为月份，从 1 开始到 12 结束 Data[3]为日期，从 1 开始到 31 结束 Data[4]为时钟，从 0 开始到 23 结束 Data[5]为分钟，从 0 开始到 59 结束 Data[6]为秒钟，从 0 开始到 59 结束 Data[7]为星期，从 1 开始到 7 结束，1 代表星期一
校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

模块返回本地时间数据：

55 aa 00 06 00 08 01 12 09 11 10 09 05 01 59

本地时间:18年9月17日16时9分5秒星期一

- 如设备在国内激活使用，则当地时间为北京时间(东 8 区)
- 如果设备在国外激活使用，则当地时间为设备所处时区时间

Wi-Fi 功能性测试

说明：

- 扫描指定的 SSID: tuya_mdev_test,返回扫描结果和信号强度百分比。
- 产测指令需要在 Wi-Fi 上电工作完成初始化流程以后发送（回复查询产品信息的数据包），否则导致产测失败或无结果。
- 本条命令多用于设备量产时的产品整机测试。
- 产测路由器密码设置不做要求（随意设置），路由需要为 2.4G 信号的路由器。

MCU 发送：

字段	长度（byte）	说明
帧头	2	0x55aa
版本	1	0x00
命令字	1	0x07
数据长度	2	0x0000
数据	Data	无
校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

MCU 触发模块功能性测试:

55 aa 00 07 00 00 06

模块返回：

字段	长度（byte）	说明
帧头	2	0x55aa
版本	1	0x00
命令字	1	0x07
数据长度	2	0x0002
数据	2	数据长度为 2 字节： Data[0]: 0x00 失败, 0x01 成功; 当 Data[0] 为 0x01 成功 ： Data[1] 表示信号强度 (0-100，0 信号最差，100 信号最强) 当 Data[0]为 0x00 失败： Data[1]为 0x00 表示未扫描到 指定的 ssid Data[1]为 0x01 表示模块未烧 录授权 key
校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

模块返回测试结果:

55 aa 00 07 00 02 01 50 59

Wi-Fi 功能性测试成功，信号强度为 80。

固件升级

请求 Wi-Fi 模块固件升级

说明：

Wi-Fi 模块的电源通断是由 MCU 去控制的，当 MCU 需要去升级 Wi-Fi 模块的固件，便可以通过下面的命令去拉去最新固件。

MCU 主板根据 Wi-Fi 模块的回复包决定是否需要给 Wi-Fi 模块断电。当 MCU 发送 0a 命令等待 5S 没有收到相关回复便把 Wi-Fi 模块断电。当模块回复正在更新固件，MCU 也需要起一个定时，当处于正在更新固件超过 60S 没有收到固件升级成功，也强制认为固件升级失败给 Wi-Fi 模块断电。

MCU 发送：

字段	长度（byte）	说明
帧头	2	0x55aa
版本	1	0x00
命令字	1	0x0a
数据长度	2	0x0000
数据	0	无
校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

MCU 请求 Wi-Fi 固件升级:

55 aa 00 0a 00 00 09

模块返回：

字段	长度（byte）	说明
帧头	2	0x55aa
版本	1	0x00
命令字	1	0x0a
数据长度	2	0x0001
数据	1	0x00(开始检测固件更新）不可断电 0x01（已经是最新固件） 断电 0x02(正在更新固件）不可断电 0x03(固件更新成功）断电 0x04(固件更新失败）断电
校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

模块升级状态返回：

55 aa 00 0a 00 01 00 0a （收到升级请求包立马返回该数据）

55 aa 00 0a 00 01 01 0b （拉取服务端信息，返回没有固件需要更新）

请求 MCU 固件升级

说明：

- 当 MCU 收到模块返回固件更新完成的信息之后，说明服务端 MCU 的升级文件已经全部拉取完成并且完成了串口数据传输。注意 MCU 在收到完整升级文件后，确认拉取文件没有问题，这个时候模块会重新发送查询信息，MCU 需要返回相关产品信息和在升级服务器后台填写一致的新的软件版本号。
- 目前支持最大为 480K 的 MCU 文件的 OTA（Over-the-Air Technology）。
- 用户在涂鸦官网配置 MCU 升级时，配置 MCU 升级方式为 App 静默升级

MCU 发送：

字段	长度（byte）	说明
帧头	2	0x55aa
版本	1	0x00
命令字	1	0x0c
数据长度	2	0x0000
数据	0	无
校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

MCU 请求 MCU 固件升级：

55 aa 00 0c 00 00 0b

模块返回：

字段	长度（byte）	说明
帧头	2	0x55aa

版本	1	0x00
命令字	1	0x0c
数据长度	2	0x0001
数据	1	0x00（开始检测固件更新）不可断电 0x01（已是最新固件）断电 0x02（正在更新固件）不可断电 0x03（固件更新完成）断电 0x04（固件更新失败）断电
校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

模块升级状态返回:

55 aa 00 0c 00 01 00 0c （收到升级请求包立马返回）

55 aa 00 0c 00 01 01 0d （拉取服务端信息返回没有固件需要更新）

MCU 升级包大小通知

说明：

当 MCU 收到升级启动包，可以知道需要升级的文件大小。此时模块开始去服务端拉取 MCU 的升级包，之后发送升级包传输的相关数据包。

模块发送：

字段	**长度（byte）**	**说明**
帧头	2	0x55aa
版本	1	0x00
命令字	1	0x0d
数据长度	2	0x0004
数据	4	固件包字节数，unsigned int， 大端
校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

模块发送文件包大小：

55 aa 00 0d 00 04 00 00 68 00 78（固件包长度 26624，即 26KB）

MCU 返回：

字段	**长度（byte）**	**说明**
帧头	2	0x55aa
版本	1	0x00
命令字	1	0x0d
数据长度	2	0x0000
数据	0	无
校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

MCU 确认包回复：

55 aa 00 0d 00 00 0c

升级包传输

说明：

- 升级包传输数据格式：包偏移（unsigned short） + 包数据。
- MCU 若收到该帧数据长度为 4 且包偏移>=固件大小，则包传输结束 。

模块发送：

字段	**长度（byte）**	**说明**
帧头	2	0x55aa
版本	1	0x00
命令字	1	0x0e
数据长度	2	0x0004+数据包长度
数据	N	前四字节，固定为包偏移， 后面为数据包内容
校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

模块发送文件数据：

若要升级的文件大小 530Byte，(最后一包数据可不回复)

1. 第一包数据，包偏移为 0x00000000，数据包长度为 256：

55aa 00 0e 0104 00000000 xx...xx XX

2. 第二包数据，包偏移为 0x00000100，数据包长度为 256：

55aa 00 0e 0104 00000100 xx...xx XX

3. 第三包数据，包偏移为 0x00000200，数据包长度为 18：

55aa 00 0e 0016 00000200 xx...xx XX

4. 最后一包，包偏移为 0x00000212，数据包长度为 0：

55aa 00 0e 0004 00000212 xx...xx XX

MCU 返回：

字段	**长度（byte）**	**说明**
帧头	2	0x55aa
版本	1	0x00
命令字	1	0x0e
数据长度	2	0x0000
数据	0	无

校验和	1	从帧头开始按字节求和得出的 结果对 256 求余

MCU 每包数据包确认：

55 aa 00 0e 00 00 0d

通讯协议-功能协议

通讯协议(产品功能部分)指令收发表

ID	功能名称		帧头版本	命令字	数据长度	dpID	数据类型	功能长度	功能指令	校验
3	电池电量	MCU上报	0x55aa 0x00	0x05	0x00 0x08	0x03	0x02	0x00 0x04	0x0-0x64	校验和
8	当前温度	MCU上报	0x55aa 0x00	0x05	0x00 0x08	0x08	0x02	0x00 0x04	0xfffffe70-0x7d0	校验和
10	震动状态	MCU上报	0x55aa 0x00	0x05	0x00 0x05	0x0a	0x04	0x00 0x01	normal:0x00 vibration:0x01 drop:0x02 tilt:0x03	校验和