AiLink WIFIBLE 系列噪音计应用手册

版本：V1.0

更新日期：2022 年 9 月 30 日

本产品的应用说明书如有变更，恕不另行通知。

深圳市易连物联网有限公司保留在不另行通知的情况下，对其中所包含的材料进行更改的权利，同时由于信任所引用的材料所造成的损害（包括结果性损害），包括但不限于印刷上的错误和其他与此出版物相关的错误，易连物联网将不承担责任。

修改记录

文档版本	作者	审核人	发布日期	修改说明	审核	批准
V1.0	LYX	Lxl，hjb	2022/9/30	1.初版		

目录

1 概述

1.1 本文档适用于深圳市易连物联网 WIFIBLE 模块(WM 模块)接入 ailink APP。

1.2 本文档适用于噪音计设备的 MCU 端开发工程师使用。

1.3 本文档讲详细介绍硬件对接、固件对接。

1.4 文档会保持更新，以官网链接为最新版本。

2 说明

2.1 我们提供标准化的连接模块、app、云平台帮助客户的噪音计设备快速实现智能化，并提供 sdk、云平台配置、增值服务和技术支持帮忙客户差异化、个性化。

2.2 我们提供的模块具有功耗低、认证齐全、APP 功能强大体验好等特点。扫描下面二维码下载 APP。

2.3 支持 MCU 配置模块（VID、PID）实现 APP 连接产品时型号自定义、图标自定义等个性化设计。

2.4 传统噪音计加上 WIFIBLE 模块后,即可实现产品智能化升级:

2.4.1 传统款的噪音计检测只能提供当前测量数据，无法进行数据对比分析，无法判断当前噪音状态，智能款可以通过手机 APP 或者大数据平台，对数据进行判断分析；

2.4.2 数据预警判断：当 APP 连接上设备后读取当前设备数据值，根据预置的数据判断标准或者是通过网络查询，对异常数据进行预警,提示用户；

2.4.3 数据分析：APP 将设备所有检测的数据，汇总成历史数据，用户可以导出图表，对数据进行分析，可以清晰的了解数据的变化过程；

3 模块版本

本文档支持的固件版本：

WM07H1S4.0.0xxx

4 模块选型

- 针对于噪音计产品的功能,主要的对比点及差异.

参数 / 型号	WM05	WM07	WM09	WM16
WiFi 协议	11b	11b	11b/g/n	11b/g/n
主频	176MHz	176MHz	196MHz	196MHz
flash	2MByte	1MByte	2MByte	2MByte
最大发射功率	10db	10db	21db	21db
功耗(深度睡眠)	22uA	20uA	2.7uA	2.6uA
功耗（闲时工作）	7.2mA	12.3mA	35mA	34.7mA
功耗（峰值）	200mA	225mA	310mA	310mA
BLE 版本	5.0	5.0	5.0	5.0

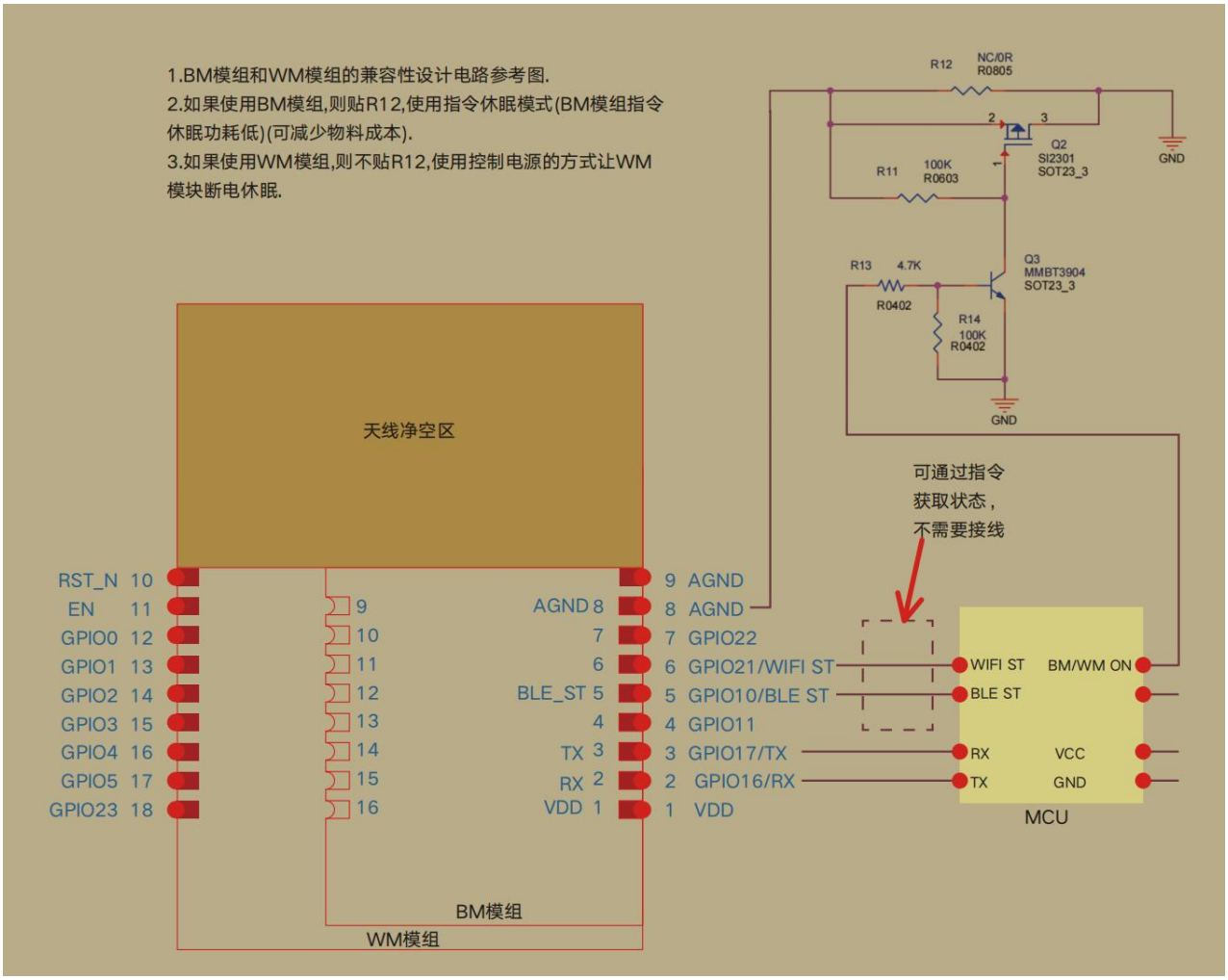
5 硬件参考设计

5.1 串口 UART

波特率 9600 ，1 位开始位，8 位数据位，1 位停止位，无奇偶校验位。

5.2 硬件连接

5.2.1 参考电路：



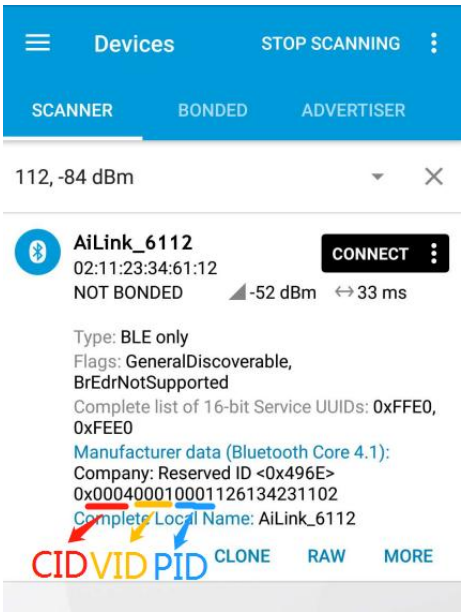
5.2.2 说明

5.2.2.1 产品智能化升级,不仅可以通过接入BLE模块,也可以通过wifi模块.我司提供了这两种方案的技术支持.因此在产品设计阶段,建议 PCB 可以进行兼容式设计.以后 BLE 或者 WiFi 方案相互替换时,则不需再重新设计 PCB 版.

6 蓝牙接口（默认）

6.1 蓝牙名称：AiLink_xxxx

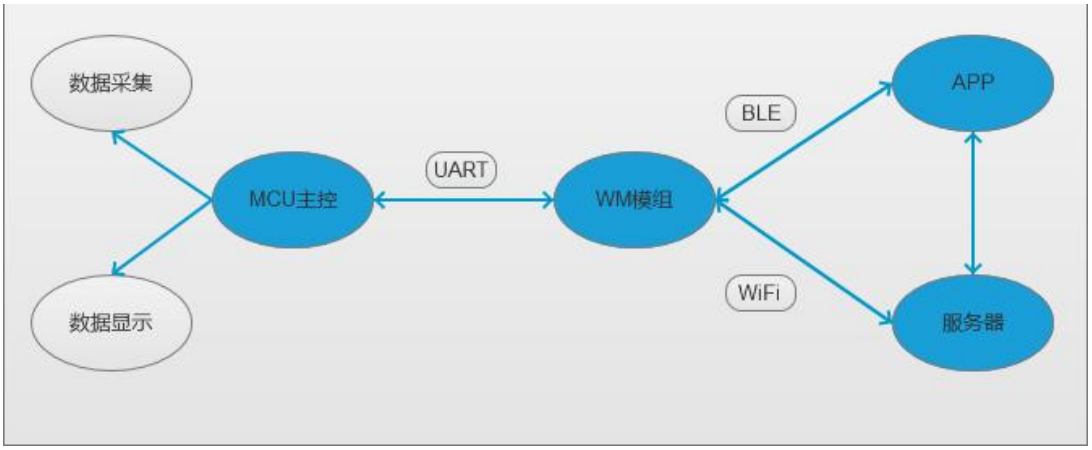
注：xxxx 为 Mac 地址后 4 个字符


Devices
STOP SCANNING
SCANNER
BONDED
ADVERTISER
112, -84 dBm
AiLink_6112
02:11:23:34:61:12
CONNECT
NOT BONDED
-52 dBm
33 ms
Type: BLE only
Flags: GeneralDiscoverable, BrEdrNotSupported
Complete list of 16-bit Service UUIDs: 0xFFE0, 0xFEE0
Manufacturer data (Bluetooth Core 4.1):
Company: Reserved ID <0x496E>
0x0004000100011261342311O2
Complete Local Name: AiLink_6112
CID
VID
PID
CLONE
RAW
MORE

7 流程及软件协议

7.1 产品定义

7.1.1 产品形态



7.1.2 说明:

1. 本产品包含三部分:
   (1) MCU 主控 , WM 传输模块 , APP/服务器
   - MCU 主控:
     MCU 主控主要负责噪音数据的采集,数据的显示,以及通过 UART 连接把数据传输到 WM 模块.
   - WM 模块:
     WM 模块主要作为 APP/服务器和 MCU 的沟通桥梁,把接收到的噪音数据传输到 APP/服务器上.
   - APP/服务器:
     APP 通过与 WM 模块连接,与主机绑定,获取噪音数据信息,设置功能等.

2. WM 模块数据传输原则:
   (1) 在 APP 通过蓝牙连接设备时,数据直接通过蓝牙传输数据.
   (2) APP 没有连接到设备时,设备数据通过 WiFi 传输到 APP 上.

7.2 产品工作流程

1. 设备上电,给 WM 模块通电
2. WM 模块上电成功后,会主动返回状态(CMD=0x26 指令).MCU 主控收到状态后,证明可以进行 UART 读写操作.
3. MCU 通过 UART 设置产品的 CID,VID,PID (必须设,否则 AILink APP 无法找到设备). AiLink CID VID PID 获取介绍：http://doc.elinkthings.com/web/#/40?page_id=144
4. 打开 APP,搜索设备,绑定设备.
5. APP 连接上设备后,会发起绑定指令,请求用户按按键确认, 设备绑定后,设备上发的数据即可在 APP 上显示.
6. 打开配网界面,对设备进行配网.
7. 模块返回联网成功指令(CMD = 0x26 指令)后，MCU 需要发送使能 MQTT 连接指令(CMD = 0xC5 指令)。
8. WM 模块没有通过蓝牙连接 APP 时,WM 模块会通过 WiFi 通过 mqtt 协议传输数据到 APP 上.
9. MCU 传输数据到 APP 上.
10. APP 进行各项的参数设置.
11. WM 状态指令查看模块是否连接上路由器,mqtt 状态指令查看模块是否连接到服务器.
12. 当设备关机时,断开设备供电.

7.3 工作流程图

7.4 基础交互指令

- MCU 和模块通信时，由于数据是需要通过 WiFi 的 mqtt 协议传输到服务器的，因此 MCU 可能通过模块返回结果判断是否已上发到服务器
- 当数据状态出现变化时，例如数据变化，或者认为操作，或者报警等，才需要更新给模块.
- 数据上发时，需要分帧传输.

Byte	Value	Description	
0	0xA7	包头	
1	0x00	产品类型（CID）高字节	
2	0x50	产品类型（CID）低字节	
3		Payload 长度(payload 部分的字节数量)	
4			Payload
5~n			
n+1	SUM（1~n）	（1~n）校验和(累加和，取低八位)	
n+2	0x7A	包尾	

检验和是指 byte1 + byte2 + … +byte n 的和，取低位 1 byte。

7.4.1 通信 TLV 的格式声明.

列表:

Type	Length(后面 Value 的 byte 数)	Value(小端序)
0x80:协议版本	1	0x01
0x01:频率计权 A/C		
0x02:测量总范围		
0x03:测量等级切换		
0x04:Max/Min 模式		
0x05:时间加权(Fast/Slow)		
0x06:数值保持(hold)		
0x07:报警		
0x08:背光		
0x09:噪音值.		
0x0A:历史		
0x0B:供电		
0x0C:设备绑定		

功能声明		
Type	Length(后面 Value 的 byte 数)	Value(小端序)
0x80:协议版本	1	Feature(1byte): 0x01 备注:该功能必须声明

0x01:频率计权 A/C	1	Feature(1byte):
		{
		Bit0: 1:支持 A 权. 0:不支持 A 权
		Bit1: 1:支持 C 权. 0:不支持 C 权
		其他:缺省,为 0
		}

0x02:测量总范围	5	Feature(1byte):
		{
		Bit0-Bit1:小数点位数
		其他:缺省,为 0
		}
		Data(4byte):
		{
		Min(2bytes):{噪音值最小值,单位 db}
		Max(2bytes):{噪音值最大值,单位 db}
		}
		备注:
		1. 如果设备不支持分等级测试,测界面的量程就是该值.
		2. 如果支持分等级测试,则界面量程以等级量程为准.

0x03:测量等级切换	1	Feature(1byte):
		{
		Bit0: 1:支持测量等级切换. 0:不支持测量等级切换
		其他:缺省,为 0
		}
		备注:设备端分多少个等级,APP 上不做区分,在 APP 上切换等级时,APP 下发+-等级给设备即可. 设备端需要返回切换后的量程.

0x04:Max/Min 模式	1	Feature(1byte): { Bit0: 1:支持. 0:不支持 其他:缺省,为 0

		}

0x05:时间加权(Fast/Slow)	1	Feature(1byte):
		{
		Bit0: 1:支持 Fast. 0:不支持 Fast
		Bit1: 1:支持 Slow. 0:不支持 Slow
		其他:缺省,为 0
		}

0x06:数值保持(hold)	1	Feature(1byte):
		{
		Bit0: 1:支持. 0:不支持
		其他:缺省,为 0
		}

0x07:报警功能	5	Feature(1byte):
		{
		Bit0:1:支持报警. 0:不支持报警
		其他:缺省,为 0
		}
		Data(4byte):
		{
		Min(2bytes):{最小值,单位:秒}
		Max(2bytes):{最大值,单位:秒}
		}

0x08:背光	1	Feature(1byte):
		{
		Bit0: 1:支持. 0:不支持
		其他:缺省
		}

0x09:噪音值	1	Feature(1byte):
		{
		Bit0-1 :小数点位数
		其他:缺省
		}

0x0A:历史	1	Feature(1byte):
		{
		Bit0: 1:支持. 0:不支持. (设备端是否支持保存历史数据)
		其他:缺省
		}

0x0B:供电	1	Feature(1byte):
		{
		0x01:锂电池供电
		0x02:干电池供电
		0x03:市电供电
		}

设备返回状态信息		
Type	Length(后面 Value 的 byte 数)	Value(小端序)

0x01:频率计权 A/C	1	Data(1byte):
		{
		0x01:使用 A 权
		0x02:使用 C 权
		}

0x02:测量总范围	--	--

0x03:测量等级及范围	5	Data(5byte):
		{
		LvL(1byte):{当前的等级}
		Min(2bytes):{量程最小值}
		Max(2bytes):{量程最大值}
		}
		备注:单位和前面功能声明的保持一致.

0x04:Max/Min 模式	1	Data(1byte):
		{
		0x00:正常模式
		0x01:最小值测量模式 Min
		0x02:最大值测量模式 Max
		}

0x05:时间加权(Fast/Slow)	1	Data(1byte):
		{
		0x01:Fast 模式
		0x02:Slow 模式
		}

0x06:数值保持(hold)	3	Feature(1byte):
		{
		0x00:正常模式
		0x01:hold 模式
		}
		Data(2byte):
		{
		hold 的值(2bytes,小数点与声明一致，正常模式时,该值无效,为 0
		}

0x07:报警	2	Feature(1byte):
		{
		Bit0: 报警状态:
		0=报警功能未打开.
		1=报警功能已打开.
		其他:缺省
		}
		Data: (1byte)
		{
		Bit0: 噪音过大报警(0:未报警,1:报警)
		其他:缺省
		}

0x08:背光	1	Data(1byte):
		{
		0x00:未打开
		0x01:已打开
		}

0x09:噪音值	3	Feature(1byte):
		{
		0x00:数值有效
		0x01:数值低于量程,显示 under
		0x01:数值高于量程,显示 over
		}
		Data(2byte):
		{
		噪音值
		}
		备注:数值不在量程内时,噪音值无效,显示上一次的值

0x0A:历史	--	--
0x0B:供电	2	Feature(1byte):

		{
		Bit0:充电状态:0=未充电. 1=在充电.
		Bit1:电压状态: 0=电压正常. 1=低电
		}
		Data:电量百分比(1byte)

0x0C:	--	--

APP 参数设置/操作		
Type	Length(后面 Value 的 byte 数)	Value(小端序)

0x01:频率计权 A/C	1	Op(1byte):
		{
		0x01:使用 A 权
		0x02:使用 C 权
		}

0x02:测量总范围	--	--

0x03:测量等级及范围	6	Op(1byte):
		{
		0x01:等级+
		0x02:等级-
		}
		Data(5bytes):
		{
		LvL(1byte):{当前的等级}
		Min(2bytes):{量程最小值}
		Max(2bytes):{量程最大值}
		}
		备注:
		APP 下发时,只发 op 的内容,其他值为 0.
		MCU 回复时,需要把当前对应的内容全部返回.

0x04:Max/Min 模式	1	Op(1byte):
		{
		0x00:正常模式
		0x01:最小值模式 Min
		0x02:最大值模式 Max
		}

0x05:时间加权(Fast/Slow)	1	Op(1byte): { 0x01:Fast 模式 0x02:Slow 模式

		}

0x06:数值保持(hold)	3	Op(1byte):
		{
		0x00:正常模式
		0x01:hold 模式
		}
		Data(2bytes):
		{
		噪音值(正常模式时,该值无效)
		}
		备注:
		1.在 APP 端按 hold 数据时,噪音值为 APP 端界面的值,APP 界面值优先级高.
		2.在设备端按 hold 数据时,噪音值为设备端界面的值,设备端界面值优先级高.

0x07:报警	3	Op(1byte):
		{
		Bit0:0=关闭报警 . 1=打开报警
		Bit1:0=none . 1=关闭当前报警(关闭当前报警,指当前触发了报警,之后可手动关闭当次报警,但是并不会关闭报警功能. 设备端/APP 可以做延时一段时间后,若还在报警值,可选择再次报警)
		}
		Data(2byte):
		{
		报警值:(小数点位数与前面的申明保持一致)
		}

0x08:背光	1	Op(1byte):
		{
		0x00:关闭
		0x01:打开
		}

0x09:噪音值	--	--

0x0A:历史		**获取历史记录:**
		APP 发送:
		Op(1byte)
		{
		0x02:获取设备端历史
		}
		Data(4byte)
		{
		Unix 时间戳:该时间戳到当前时间里产生的历史(之前同步过的历史,则不会重新同步)
		}
		
		设备端返回:
		Op(1byte)
		{
		0x02:获取设备端历史
		}
		Data:
		{
		1.历史总量(4byte): 该值只从 APP 下发的时间戳到当前时间戳里,设备产生历史总量(即未同步的总量)
		2.本次上发条数(4byte):即 history 里面含有多少组
		3.History(n byte):
		{
		//第一组
		时间戳:4byte
		噪音值:2byte
		状态:1byte
		{
		Bit0-bit1:1=A 权. 2=C 权
		Bit2: 0=未报警, 1=当时已报警
		Bit3-bit4: 1=噪音值低于量程
		2=噪音值高于量程
		其他:缺省
		}
		//第二组 (可进行多组拼接)
		.....
		}
		}
		
		清除历史记录:
		APP 发送:

		Op(1byte)
		{
		0x02:获取设备端历史
		}
		设备返回:
		Op(1byte)
		{
		0x00:成功
		0x01:失败
		}
		备注:
		APP 端获取历史记录流程:
		1. 如果未获取过历史记录，发送时间戳为 0;
		2. 设备返回记录数据，APP 收到第一次返回的数据，记录总条数 Total，以及接收到的条数 Receive;
		3. APP 收到模块返回的数据后，根据返回的数据中总条数和本次发送条数，判断是否需要继续获取历史记录，如果需要继续获取，根据返回数据中的最新的时间戳，继续以此时间戳发送获取历史记录指令;
		4. APP 再次收到返回数据，根据返回的本次记录条数，累加 Receive 的值，这样可以根据 Total 和 Receive 计算获取记录的进度;
		5. 需要加入超时机制，防止 APP 不能收到设备回复数据或 BM 模块不能收到 APP 数据的情况，在 APP 发送获取指令后，超时 5 秒后未收到设备模块的回复，重新发送获取指令;

0x0B:供电		
0x0C:设备绑定	1	Data(1byte):(APP 首次连接设备绑定时,会通过指令请求用户确认) 0x01:APP 发起,请求用户按键通过绑定 0x02:MCU 返回,MCU 等待用户按键 0x03:MCU 返回,用户已按按键 0x04:MCU 返回,用户超时(30s)没按按键 0x05:APP 发起,APP 取消绑定.

7.4.2 APP 获取设备功能列表

- APP 为了保证 APP 的功能和设备端的功能同步,APP 连接设备时,需要获取设备支持的功能.
- 若设备不支持功能,则相对应的功能和指令则不需支持及设计.

数据格式：

Byte	Default	Description	
0	0xA7	包头	
1~2	0x0050	产品类型：噪音计	
3		Payload 长度	
4	0x01	CMD：获取设备支持的功能列表	Payload
5	0x00	保留	
6	SUM	校验和	
7	0x7A	包尾	

MCU 主控回应设备功能列表：

Byte	Default	Description	
0	0xA7	包头	
1~2	0x0050	产品类型：噪音计	
3		Payload 长度	
4	0x02	CMD：设备返回功能	Payload
		Group:总组数(一组指令只能发 200byte,当设备指令过长时,需要分包发送)	
	TYPE	具体的内容参考”功能说明”TLV ,只需填写支持的功能 TLV	
	Length		
	Value		
	TYPE		
	Length		
	Value		
	...	...	
	SUM	校验和	
	0x7A	包尾	

7.4.3 APP 获取设备状态

- APP 连接上设备后，需获取设备的状态,MCU 收到指令后需要返回.
- 当设备状态发生改变时,也需要通过该指令返回.
- APP 获取时,MCU 端需要全部支持的 TLV 返回. 当 MCU 主动上发时,只需要发改变了的 TLV 即可.
- 该指令的数据不会保存到服务器.

数据格式：

Byte	Default	Description	
0	0xA7	包头	
1~2	0x0050	产品类型：噪音计	
3		Payload 长度	
4	0x03	CMD：获取设备状态	Payload
5	0x00	保留	
6	SUM	校验和	
7	0x7A	包尾	

MCU 主控回应设备状态：

Byte	Default	Description	
0	0xA7	包头	
1~2	0x0050	产品类型：噪音计	
3		Payload 长度	
4	0x04	CMD：设备返回功能	Payload
		Group:总组数(一组指令只能发 200byte,当设备指令过长时,需要分包发送)	
	TYPE	具体的内容参考”设备返回状态信息”TLV	
	Length		
	Value		
	TYPE		
	Length		
	Value		
	...	...	
	SUM	校验和	
	0x7A	包尾	

7.4.4 APP 设置/获取参数

- APP 在界面上进行功能获取/设置时,通过该指令设置
- MCU收到获取指令时,需返回所有的参数,收到设置指令时,只需返回设置项的TLV内容即可

APP 设置：

Byte	Default	Description	
0	0xA7	包头	
1~2	0x0050	产品类型：噪音计	
3		Payload 长度	
4	0x05	CMD：设置/获取参数功能	Payload
		Op: 0x01:获取(当指令为获取时,指令里不需要包含 TLV,MCU 返回所有的参数即可) 0x02:设置	
		Group:总组数(一组指令只能发 200byte,当设备指令过长时,需要分包发送)	
	TYPE	具体的内容参考"APP 参数设置"TLV	
	Length		
	Value		
	TYPE		
	Length		
	Value		
	...	...	
	SUM	校验和	
	0x7A	包尾	

MCU 返回:

Byte	Default	Description	
0	0xA7	包头	
1~2	0x0050	产品类型：噪音计	
3		Payload 长度	
4	0x06	CMD：MCU 返回参数功能	Payload
		Group:总组数(一组指令只能发 200byte,当设备指令过长时,需要分包发送)	
	TYPE	具体的内容参考"APP 参数设置"TLV	
	Length		
	Value		
	TYPE		
	Length		

	Value		
	...	...	
	SUM	校验和	
	0x7A	包尾	

7.4.5 MCU 上发保存数据

- 该指令的数据,会通过 HTTP 协议上发到服务器. 服务器会保存该数据当做设备的历史数据.
- 设备端定时 10 分钟发一组数据,若当前值与上一次上发值完全一致,则不需要上发数据.
- 当设备端数据出现报警状态时,则需立即上发数据,上发成功后,等 10 分钟后再上发数据即可(若出现其他报警,则可立即上发,不需再等 10 分钟).

MCU 发送：

Byte	Default	Description	
0	0xA7	包头	
1~2	0x0050	产品类型：噪音计	
3		Payload 长度	
4	0xF1	CMD：上发保存数据	Payload
		Group:总组数(一组指令只能发 **255byte**,当设备指令过长时,需要分包发送)	
	TYPE	0x0A	
	Length(n bytes)	n	

	Value	Value:History(n byte):
		{
		//第一组
		时间戳:4byte
		噪音值:2byte
		状态:1byte
		{
		Bit0-bit1:1=A 权. 2=C 权
		Bit2: 0=未报警, 1=当时已报警
		Bit3-bit4: 1=噪音值低于量程
		2=噪音值高于量程
		其他:缺省
		}
		//第二组 (可进行多组拼接)
		.....
		}	

	TYPE	0x0B:电量(当前)	
	Length	2	

	Value	Feature(1byte):
		{
		Bit0:充电状态:0=未充电. 1=在充电.
		Bit1:电压状态: 0=电压正常. 1=低电
		}	

		Data:电量百分比(1byte)	
	SUM	校验和	
	0x7A	包尾	

WM 模块返回:

Byte	Default	Description	
0	0xA7	包头	
1~2	0x0050	产品类型： 0x0050	
3	0x02	Payload 长度	
4	0xF1	Type：WM 回复数据上传结果	Payload
5		结果： 0：上传成功 1：上传失败	
6	SUM（1~5）	(1~5)校验和	
7	0x7A	包尾	

8 模块通用指令集

指令格式

Byte	Value	Description
0	0xA6	包头
1		Payload 长度（最大 16byte）
2 ~n		Payload
n+1	SUM（1~n）	(1~n)校验和
n+2	0x6A	包尾（注：n+2 不能超过 20）byte1 + byte2 + …+byte n 的和，取低位 1 byte。

设置指令里，数据的 Byte 数不能超过 20

8.1 APP 同步时间到 MCU（CMD：0x37、0x38）

- 对于某些设备，具有时间功能的，此时，可利用此指令进行数据的同步。

- **APP 下发时间。**

Byte	Default	Description	
0	0xA6	包头	
1		Payload 长度（最大 15byte）	
2	0x37	CMD：APP 同步时间	Payload
3~9		时间：7 个 byte 年（当前年份-2000） 月 日 时 分 秒 星期（1~7 1=周一 ~ 7=周日）	
10	SUM（1~n）	(1~n)校验和	
11	0x6A	包尾	

- **MCU 返回同步时间结果**

Byte	Value	Description	
0	0xA6	包头	
1	0x02	Payload 长度	
2	0x37	CMD：MCU 返回时间同步结果	Payload
3		结果值： 0：成功	

		1：失败 2：不支持	
4	Sum	(1~3)校验和	
5	0x6A	包尾	

- **MCU 请求时间**

设备有时间功能，且在与 APP 连接状态时，可以请求时间更新，APP 收到该请求，会下发时间同步。

Byte	Value	Description	
0	0xA6	包头	
1	0x02	Payload 长度	
2	0x38	CMD：MCU 请求 APP 下发时间	Payload
3		Value 0x01	
4	Sum	(1~3)校验和	
5	0x6A	包尾	

8.2 设置模块立即进入休眠 CMD = 0x19

- 当模块进入睡眠后, 模块的 Tx 输出高,Rx 作为输入唤醒口.
- MCU 的 Tx 输出高,Rx 作为输入口.
- 通过指令进入的休眠,模块都会有一定的电流,若想完全让模块停止工作.可通过电路断开模块的供电.(可参考硬件参考电路设计)

模块接收：

Byte	Value	Description	
0	0xA6	包头	
1	Len	Payload 长度	
2	0x19	CMD：设置进入睡眠	Payload
3	0x01	Value：0x01	
4		休眠模式： 0x02：BLE 关闭，WIFI 关闭，串口关闭（掉电模式）。 0x01：BLE 正常工作，WIFI 正常工作，串口关闭（正常睡眠）。 0x00：BLE 关闭，WIFI 关闭，串口关闭，系统内部 rc 时钟保持	
5		保留位	
6		保留位	
7	Sum（1~6）	校验和	
8	0x6A	包尾	

模块响应：

Byte	Value	Description	
0	0xA6	包头	
1	Len	Payload 长度	
2	0x19	CMD：回复设置进入睡眠的结果	Payload
3		结果值： 0x00：成功（成功后 100ms 后进入睡眠） 0x01：失败 0x02：不支持	
4	Sum（1~3）	校验和	
5	0x6A	包尾	

➢ 举例：设置进入掉电模式

发送：A6 05 19 01 00 00 00 1f 6A

8.3 设置模块自动修眠时间 CMD = 0x17

➢ 设置自动睡眠时间,当模块超时接收不到数据后,便可进入自动进入休眠模式.

模块接收：

Byte	Value	Description		
0	0xA6	包头		
1	0x08	Payload 长度		
2	0x17	CMD：设置自动睡眠时间		Payload
3		自动睡眠标志位： 0：不开启自动休眠 1：开启自动休眠		
4		自动睡眠时间的最高字节	单位 ：s 范围： 5 ~ 0xffffffff/100 （建议设为：60s）	
5		自动睡眠时间的次高字节		
6		自动睡眠时间的次低字节		
7		自动睡眠时间的最低字节		
8		休眠模式： 0x00：BLE 关闭，WIFI 关闭，串口关闭（掉电模式）。 0x01：BLE 正常工作，WIFI 正常工作，串口关闭（正常睡眠）。		
9	0xFF	保留位		
10	0xFF	保留位		
11	Sum（1~10）	校验和		
12	0x6A	包尾		

模块响应：

Byte	Value	Description	
0	0xA6	包头	
1	0x02	Payload 长度	
2	0x17	CMD：回复设置自动睡眠时间结果	Payload
3		结果值： 0x00：成功 0x01：失败 0x02：不支持	
4	Sum（1~3）	校验和	
5	0x6A	包尾	

➢ 举例：设置 20S 自动进入掉电模式

发送：A6 09 17 01 00 00 00 14 00 FF FF 33 6A

8.4 设置、读取 CID、VID、PID（CMD：0x1D、0x1E）

➢ CID 为产品类型 ID，请按照协议透传产品类型设置（必须设）
➢ VID 为设备厂家 ID，请联系我司分配（必须设）
➢ PID 为产品型号 ID，厂商自己分配，建议根据产品型号分配唯一值（必须设）
➢ 以上三个值默认为 0，不代表任何产品（调试阶段先设置 CID）
➢ ailnk CID VID PID 获取介绍：http://doc.elinkthings.com/web/#/40?page_id=144

MCU/APP 设置 ID：

Byte	Value	Description	
0	0xA6	包头	
1		Payload 长度	
2	0x1D	CMD：设置 ID	Payload
3	0x07	设置 ID 标志位	
4		CID：产品类型 ID 的高字节	
5		CID：产品类型 ID 的低字节	
6		VID：厂商 ID 的高字节	
7		VID：厂商 ID 的低字节	
8		PID：产品 ID 的高字节	
9		PID：产品 ID 的低字节	
10	Sum	(1~9)校验和	
11	0x6A	包尾	

BM 回复设置结果：

Byte	Value	Description	
0	0xA6	包头	
1	Len	Payload 长度	
2	0x1D	CMD：回复设置 ID 结果	Payload
3		结果值： 0：成功 1：失败 2：不支持	
4	Sum	(1~3)校验和	
5	0x6A	包尾	

MCU/APP 获取 ID：

Byte	Value	Description	
0	0xA6	包头	
1	0x01	Payload 长度	
2	0x1E	CMD：获取 ID 设置值	Payload
3	0x1F	(1~2)校验和	
4	0x6A	包尾	

BM 返回 ID 值：

Byte	Value	Description	
0	0xA6	包头	
1	Len	Payload 长度	
2	0x1E	CMD：返回 ID	Payload
3		设置 ID 标志位 Bit0 ： 0 ： 不设置 CID。 1：设置 CID Bit1 ： 0 ： 不设置 VID。 1：设置 VID Bit2： 0 ： 不设置 PID。 1：设置 PID	
4		CID：产品类型 ID 的高字节	
5		CID：产品类型 ID 的低字节	
6		VID：厂商 ID 的高字节	
7		VID：厂商 ID 的低字节	
8		PID：产品 ID 的高字节	
9		PID：产品 ID 的低字节	
10	Sum	(1~9)校验和	
11	0x6A	包尾	

8.5 设置模块重启（CMD：0x21）

设置重启模块：

Byte	Value	Description	
0	0xA6	包头	
1	0x02	Payload 长度	
2	0x21	CMD：设置模块重启	Payload
3	0x01	Value： 0x01	
4	Sum	(1~3)校验和	
5	0x6A	包尾	

BM 回复设置结果：

Byte	Value	Description	
0	0xA6	包头	
1	Len	Payload 长度	
2	0x21	CMD：回复设置模块重启结果	Payload
3		结果值： 0：成功（成功后，100ms 后模块重启） 1：失败 2：不支持	
4	Sum	(1~3)校验和	
5	0x6A	包尾	

8.6 设置恢复出厂设置（CMD：0x22）

设置恢复出厂设置：

Byte	Value	Description	
0	0xA6	包头	
1	0x02	Payload 长度	
2	0x22	CMD：设置恢复出厂设置	Payload
3	0x01	Value： 0x01	
4	Sum	(1~3)校验和	
5	0x6A	包尾	

BM 回复设置结果：

Byte	Value	Description	
0	0xA6	包头	
1	0x02	Payload 长度	
2	0x22	CMD：回复设置模块重启结果	Payload
3		结果值： 0：成功（成功后，100ms 后恢复出厂设置） 1：失败 2：不支持	
4	Sum	(1~3)校验和	
5	0x6A	包尾	

8.7 获取 WM 模块状态(CMD = 0x26,0xAB)

- MCU/APP 可通过指令获取模块的状态
- 当模块状态变化时,会主动返回状态指令

获取模块状态：

Byte	Value	Description	
0	0xA6	包头	
1		Payload 长度	
2	0x26	CMD：获取状态	Payload
3	Sum	(1~2)校验和	
4	0x6A	包尾	

WM 返回模块状态：

Byte	Value	Description	
0	0xA6	包头	
1	0x03	Payload 长度	
2	0x26	CMD：返回模块状态	Payload
3		模块状态： bit0-bit3 表示 BLE 状态： 0：无连接 1：已连接 2：配对完成 Bit4-bit7 表示 wifi 状态： **0：未配置 AP；** **1：连接 AP 失败，连接时密码错误、AP 信号不好、主动断开都会是这个状态；** 2：与服务器通讯失败； 3：成功连接上 AP； 4：正在连接 AP；	

4		工作状态： 0：唤醒 1：进入休眠 2：模块准备就绪	
5	Sum（1~5）	校验和	
6	0x6A	包尾	

WM 返回 WiFi 连接失败原因:

当 WM 连接 AP 失败时，WM 返回 WiFi 连接失败原因。其他状态不回应此条指令。

Byte	Value	Description	
0	0xA6	包头	
1	0x03	Payload 长度	
2	0xAB	CMD：返回 WiFi 连接失败原因	Payload
3		原因值： 0x00：未知原因 0x01：AP 信号差 0x02：密码错误 0x03：获取不到 IP	
4	0x00	保留位	
5	Sum（1~4）	校验和	
6	0x6A	包尾	

8.8 请求 Unix 时间 CMD = 0x44

- MCU 可指令获取 Unix 时间.

模块接收：

Byte	Value	Description	
0	0xA6	包头	
1	0x01	Payload 长度	
2	0x44	CMD：请求 Unix 时间	Payload
4	0x45	（1~2）校验和	
5	0x6A	包尾	

模块响应：

按“设置 Unix 时间 CMD = 45”指令回复

8.9 设置 Unix 时间 CMD = 0x45

模块接收：

Byte	Value	Description	
0	0xA6	包头	
1	0x05	Payload 长度	
2	0x45	CMD：设置 Unix 时间	Payload
3 ~6		0x00000000：未获取到准确值（失败） 其他：Unix 时间，4byte 小端序（零时区）	
7	Sum（1~6）	校验和	
8	0x6A	包尾	

模块响应：

Byte	Value	Description	
0	0xA6	包头	
1	0x02	Payload 长度	
2	0x45	CMD：回复设置 Unix 时间结果	Payload
3		结果值： 0x00：成功 0x01：失败 0x02：不支持	
4	Sum（1~3）	校验和	
5	0x6A	包尾	

8.10 查询/使能 mqtt 连接状态 CMD = 0xC5

模块接收：

Byte	Value	Description	
0	0xA6	包头	
1	0x05	Payload 长度	
2	0xC5	CMD：查询 mqtt 连接状态	Payload
3		0x05	
4		0x01	
5		0x03:使能 MQTT 连接 AILINK 服务器 0x04:查询	
6	Sum（1~5）	校验和	
7	0x6A	包尾	

模块响应：

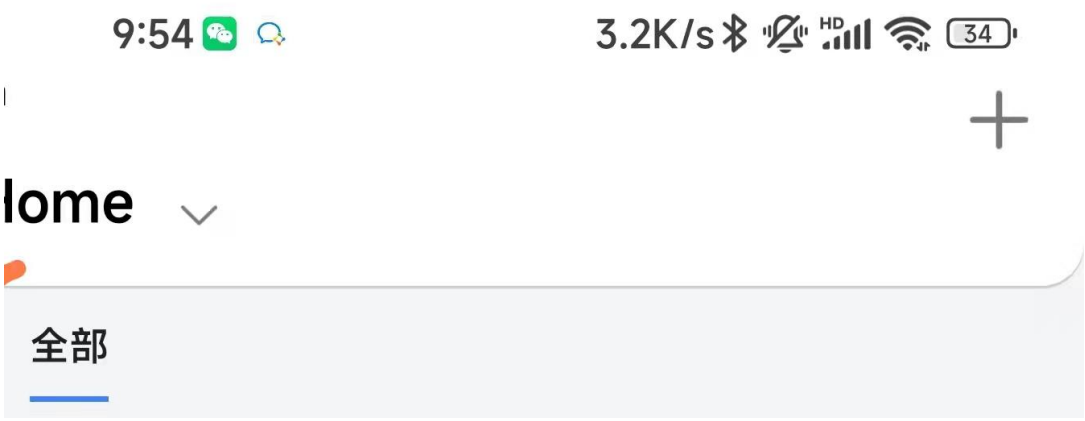
Byte	Value	Description	
0	0xA6	包头	
1	0x05	Payload 长度	
2	0xC5	CMD：查询 mqtt 连接状态	Payload
3		0x05	
4		0x01	
5		0x00：连接 0x01：连接断开 0x02：重新连接 在建立 MQTT 连接后，如果模块和服务端出现断连后，模块会尝试重新连接服务端，在连接状态发生变化时，模块端也会主动发送相应的状态给 MCU 端	
6	Sum（1~5）	校验和	
7	0x6A	包尾	

9 举例说明

10 测试指导

10.1 连接测试

1. 给设备上电
2. 打开 Ailink APP
3. 点击右上角的”+”添加设备



4. 选择”手动添加”,找对应类型的设备添加.或者点击”附近设备”.

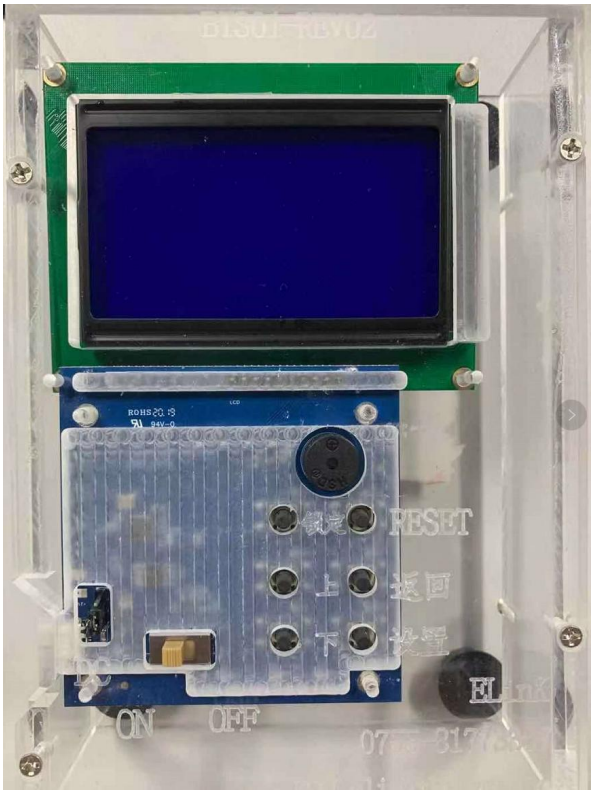
5. 选择设备进行绑定.

10.2 功能测试

具体的功能测试,请到官网链接下载产品的测试用例.测试完成后,通过对接窗口提交与我司审核.

11 生产测试指导

我们有生产使用的测试盒（BTS02），能够高效、快速、批量辅助生产测试。批量时，联系我司购买即可。



12 联系我们

深圳市易连物联网有限公司

地址：深圳市宝安区西乡街道银田工业区侨鸿盛文化创意园写字楼 A 栋五层 502 室

Tel: +（86）0755-81773367

市场部邮箱：marketing@elinkthings.com

FAE 邮箱：hw@elinkthings.com

官网： www.elinkthings.com