M5Stack Unit ChainBus Protocol								All packets start with 0xAA 0x55 and end with 0x55 0xAA				V1 (Version) 2025/11/13
指令集	Byte 指令格式	0	1	2	3	4	5	6	7	8	9	10
		Length_low	Length_high	Index	Cmd	Data1	Data2	Data3	Data4	Data5	Data6	Data7
I2C初始化	指令包格式	0x04	0x00	Index_id	0x10	I2c_speed	CRC					
	应答包格式	0x04	0x00	Index_id	0x10	Operation_status	CRC					
	指令详解	（1）功能说明：I2C初始化。 （2）输入参数：Index_id（设备下标ID）、I2c_speed（I2C速度） （3）返回参数：Operation_status （4）指令代码：0x10 注1：Operation_status 操作状态 0：操作失败 1：操作成功 注2：I2c_speed 0：100KHZ 1：400KHZ										
I2C读	指令包格式	0x05	0x00	Index_id	0x11	I2c_addr	length	CRC				
	应答包格式	(0x04+length) /(0x04)	0x00	Index_id	0x11	Operation_status	Data1	Data2	Data3	...	CRC	
	指令详解	（1）功能说明：I2C读。 （2）输入参数：Index_id（设备下标ID）、I2c_addr（I2C地址）、length（要读取的长度，最大64） （3）返回参数：Operation_status（操作状态），Data（读取的数据） （4）指令代码：0x11 注1：Operation_status 操作状态 0：操作失败 1：操作成功 2：模式不匹配（需要先配置I2C） 注2：操作成功才会返回要读取的数据										
I2C写	指令包格式	0x05+length	0x00	Index_id	0x12	I2c_addr	length	Data1	Data2	Data3	...	CRC
	应答包格式	0x04	0x00	Index_id	0x12	Operation_status	CRC					
	指令详解	（1）功能说明：I2C写。 （2）输入参数：Index_id（设备下标ID）、I2c_addr（I2C地址），length(要写入的数据长度，最大64)，Data要写入的数据 （3）返回参数：Operation_status（操作状态） （4）指令代码：0x12 注1：Operation_status 操作状态 0：操作失败 1：操作成功 2：模式不匹配（需要先配置I2C）										
I2C指定地址读	指令包格式	0x08	0x00	Index_id	0x13	I2c_addr	Reg_addr_length	Reg_addr_low	Reg_addr_high	length	CRC	
	应答包格式	0x04+length	0x00	Index_id	0x13	Operation_status	Data1	Data2	Data3	...	CRC	
	指令详解	（1）功能说明：I2C指定地址读。 （2）输入参数：Index_id（设备下标ID）、I2c_addr（I2C地址）、Reg_addr_length（I2C设备寄存器长度）、Reg_addr（寄存器地址），length（要读取的长度，最大64） （3）返回参数：Operation_status（操作状态），Data（读取的数据） （4）指令代码：0x13 注1：Operation_status 操作状态 0：操作失败 1：操作成功 2：模式不匹配（需要先配置I2C） 注2：Reg_addr_length要控制的传感器设备地址大小 1：8位地址 2：16位地址 注3：Reg_addr = (uint16_t) Reg_addr_high<< 8 | Reg_addr_low（如果i2c设备寄存器地址是8位，让高八位默认0即可）										
I2C指定地址写	指令包格式	0x08+length	0x00	Index_id	0x14	I2c_addr	Reg_addr_length	Reg_addr_low	Reg_addr_high	length	Data...	CRC
	应答包格式	0x04	0x00	Index_id	0x14	Operation_status	CRC					
	指令详解	（1）功能说明：I2C指定地址写。 （2）输入参数：Index_id（设备下标ID）、I2c_addr（I2C地址）、Reg_addr_length（I2C设备寄存器长度）、Reg_addr（寄存器地址），length（要写入数据长度，最大64）、Data（要写入的数据） （3）返回参数：Operation_status（操作状态） （4）指令代码：0x14 注1：Operation_status 操作状态 0：操作失败 1：操作成功 2：模式不匹配（需要先配置I2C） 注2：Reg_addr_length要控制的传感器设备地址大小 1：8位地址 2：16位地址 注3：Reg_addr = (uint16_t) Reg_addr_high<< 8 | Reg_addr_low（如果i2c设备寄存器地址是8位，让高八位默认0即可）										
获取连接设备I2C地址	指令包格式	0x03	0x00	Index_id	0x15	CRC						
	应答包格式	0x05+ I2c_addr_num	0x00	Index_id	0x15	Operation_status	I2c_addr_num	Addr1	Addr2	Addr3	...	CRC
	指令详解	（1）功能说明：获取连接设备的I2C地址。 （2）输入参数：Index_id（设备下标ID） （3）返回参数：I2c_addr_num（I2c地址个数）、Addr(i2c地址) （4）指令代码：0x15 注1：Operation_status 操作状态 0：操作失败 1：操作成功 2：模式不匹配（需要先配置I2C）										
设置RGB值	指令包格式	0x05+Num*3	0x00	Index_id	0x20	Index	Num	R	G	B	...	CRC
	应答包格式	0x04	0x00	Index_id	0x20	Operation_status	CRC					
	指令详解	（1）功能说明：设置RGB值。 （2）输入参数：Index_id（设备下标ID）、Index（RGB灯起始坐标）、Num（要设置的RGB数量）、RGB值 （3）返回参数：Operation_status （4）指令代码：0x20 注1：Operation_status 操作状态 0：操作失败 1：操作成功 注2：Index是RGB灯起始坐标，从0开始，Uart设备只有一个RGB灯，所以Index = 0，Num = 1										

获取RGB值	指令包格式	0x05	0x00	Index_id	0x21	Index	Num	CRC				
	应答包格式	0x04+Num*3/0x04	0x00	Index_id	0x21	Operation_status	R	G	B	...	CRC	
	指令详解	（1）功能说明：获取RGB值。 （2）输入参数：Index_id（设备下标ID）、Index（RGB灯起始坐标）、Num（要读取的RGB灯数量） （3）返回参数：RGB值 （4）指令代码：0x21 注1：Operation_status 操作状态 0：操作失败 1：操作成功 注2：Index是RGB灯起始坐标，从0开始，Uart设备只有一个RGB灯，所以Index = 0，Num = 1 注3：操作失败返回数据包不包含RGB值										
设置RGB灯亮度	指令包格式	0x05	0x00	Index_id	0x22	Light	Save_to_flash	CRC				
	应答包格式	0x04	0x00	Index_id	0x22	Operation_status	CRC					
	指令详解	（1）功能说明：设置RGB灯亮度 （2）输入参数：Index_id（设备下标ID）、Light（RGB亮度 0~100 默认40）、Save_to_flash（是否保存到内部Flash） （3）返回参数：Operation_status （4）指令代码：0x22 注1：Operation_status 操作状态 0：操作失败 1：操作成功 注2：Save_to_flash 是否保存内部Flash 0：不保存 1：保存 注3：设置成功即刻生效 注4：保存至内部 Flash 需要擦除页面，过程较为耗时，频繁操作会影响设备寿命。如需要频繁调整亮度，请避免每次都保存至内部 Flash。										
获取RGB灯亮度	指令包格式	0x03	0x00	Index_id	0x23	CRC						
	应答包格式	0x04	0x00	Index_id	0x23	Light	CRC					
	指令详解	（1）功能说明：获取RGB灯亮度。 （2）输入参数： Index_id（设备下标ID） （3）返回参数： Light（RGB亮度） （4）指令代码： 0x23										
GPIO输出模式配置	指令包格式	0x06	0x00	Index_id	0x30	GPIO	GPIO_mode	GPIO_up/down	CRC			
	应答包格式	0x04	0x00	Index_id	0x30	Operation_status	CRC					
	指令详解	（1）功能说明：输出模式配置。 （2）输入参数：Index_id（设备下标ID）、GPIO(表示引脚)、GPIO_mode:（表示GPIO模式）、GPIO_up/down（ 表示上/下拉） （3）返回参数：Operation_status （4）指令代码：0x30 注1：Operation_status 操作状态 0：操作失败 1：操作成功 注2： GPIO 1: 表示GPIO1 2: 表示GPIO2 注3：GPIO_mode 0: 表示推挽 1: 表示开漏 注4：GPIO_up/down 0: 表示上拉 1: 表示下拉 2: 表示不上拉不下拉										
GPIO设置输出电平	指令包格式	0x05	0x00	Index_id	0x31	GPIO	GPIO_level	CRC				
	应答包格式	0x04	0x00	Index_id	0x31	Operation_status	CRC					
	指令详解	（1）功能说明：设置输出电平。 （2）输入参数：Index_id（设备下标ID）、GPIO(表示引脚)、GPIO_level（ 引脚输出电平） （3）返回参数：Operation_status （4）指令代码：0x31 注1：Operation_status 操作状态 0：操作失败 1：操作成功 2：模式不匹配（输出模式下才可以设置） 注2： GPIO 1: 表示GPIO1 2: 表示GPIO2 注3：GPIO_level 0: 表示低电平 1: 表示高电平 注4：只能在输出模式下设置输出电平。										
GPIO获取输出电平	指令包格式	0x04	0x00	Index_id	0x32	GPIO	CRC					
	应答包格式	0x05	0x00	Index_id	0x32	Operation_status	GPIO_level	CRC				
	指令详解	（1）功能说明：获取输出电平。 （2）输入参数：Index_id（设备下标ID）、GPIO(表示引脚) （3）返回参数：Operation_status、GPIO_level（ 引脚输出电平） （4）指令代码：0x32 注1：Operation_status 操作状态 0：操作失败 1：操作成功 注2： GPIO 1: 表示GPIO1 2: 表示GPIO2 注3：GPIO_level 0: 表示低电平 1: 表示高电平										
GPIO输入模式配置	指令包格式	0x05	0x00	Index_id	0x40	GPIO	GPIO_up/down	CRC				
	应答包格式	0x04	0x00	Index_id	0x40	Operation_status	CRC					
	指令详解	（1）功能说明：输入模式配置。 （2）输入参数：Index_id（设备下标ID）、GPIO(表示引脚)、GPIO_up/down（ 表示上/下拉） （3）返回参数：Operation_status （4）指令代码：0x40 注1：Operation_status 操作状态 0：操作失败 1：操作成功 注2： GPIO 1: 表示GPIO1 2: 表示GPIO2 注3：GPIO_up/down 0: 表示上拉 1: 表示下拉 2: 表示不上拉不下拉										

GPIO读取输入引脚电平	指令包格式	0x04	0x00	Index_id	0x41	GPIO	CRC					
	应答包格式	0x05	0x00	Index_id	0x41	Operation_status	GPIO_level_status	CRC				
	指令详解	（1）功能说明：读取输入引脚电平。 （2）输入参数：Index_id（设备下标ID）、GPIO(表示引脚) （3）返回参数：Operation_status、GPIO_level_status （4）指令代码：0x41 注1：Operation_status 操作状态 0：操作失败 1：操作成功 注2：GPIO_level_status 操作状态 0：低电平 1：高电平 注3：GPIO 1: 表示GPIO1 2: 表示GPIO2										
GPIO外部中断模式配置	指令包格式	0x06	0x00	Index_id	0x50	GPIO	GPIO_up/down	Trigger_mode	CRC			
	应答包格式	0x04	0x00	Index_id	0x50	Operation_status	CRC					
	指令详解	（1）功能说明：外部中断模式配置。 （2）输入参数：Index_id（设备下标ID）、GPIO(表示引脚))、GPIO_up/down（表示是否需要上/下拉）、Trigger_mode（触发方式） （3）返回参数：Operation_status （4）指令代码：0x50 注1：Operation_status 操作状态 0：操作失败 1：操作成功 注2：GPIO_up/down 0: 表示上拉 1: 表示下拉 2: 表示不上拉不下拉 注3：Trigger_mode 0: 上升沿 1: 下降沿 2: 上升/下降沿										
ADC模式配置	指令包格式	0x04	0x00	Index_id	0x60	GPIO	CRC					
	应答包格式	0x04	0x00	Index_id	0x60	Operation_status	CRC					
	指令详解	（1）功能说明：ADC模式配置。 （2）输入参数：Index_id（设备下标ID）、GPIO （3）返回参数：Operation_status （4）指令代码：0x60 注1：Operation_status 操作状态 0：操作失败 1：操作成功 注2：GPIO 1: 表示GPIO1 2: 表示GPIO2										
ADC读取	指令包格式	0x04	0x00	Index_id	0x61	GPIO	CRC					
	应答包格式	0x06	0x00	Index_id	0x61	Operation_status	ADC_value_low	ADC_value_high	CRC			
	指令详解	（1）功能说明：读取采集数据。 （2）输入参数：Index_id（设备下标ID）、GPIO （3）返回参数：Operation_status 、ADC_value（ADC采集值） （4）指令代码：0x61 注1：Operation_status 操作状态 0：操作失败 1：操作成功 2：模式不匹配（未配置ADC模式，请先配置） 注2：ADC_value = (uint16_t)((ADC_value_high << 8) | ADC_value_low)（0～4095，参考电压3.3V） 注3：GPIO 1: 表示GPIO1 2: 表示GPIO2										
工作状态查询	指令包格式	0x03	0x00	Index_id	0x70	CRC						
	应答包格式	0x05	0x00	Index_id	0x70	GPIO1_work_status	GPIO2_work_status	CRC				
	指令详解	（1）功能说明：工作状态查询。 （2）输入参数：Index_id（设备下标ID） （3）返回参数：GPIO_work_status（引脚工作状态） （4）指令代码：0x70 注1：GPIO_work_status 0：表示未配置工作状态 1：输出工作状态 2：输入工作状态 3：表示NVIC工作状态 4：ADC工作状态 5：I2C工作状态										
GPIO外部中断返回	指令包格式	None										
	应答包格式	0x05	0x00	Index_id	0xE0	Trigger_mode	GPIO	CRC				
	指令详解	（1）功能说明：中断模式配置返回主动。 （2）输入参数：Index_id（设备下标ID）、GPIO(表示引脚))、Trigger_mode:（触发方式） （3）返回参数：GPIO、Trigger_mode （4）指令代码：0xE0 注1：GPIO 1: 表示GPIO1 2: 表示GPIO2 注2：Trigger_mode 0: 上升沿 1: 下降沿										
查询设备唯一UID	指令包格式	0x04	0x00	Index_id	0xF8	UID_Type	CRC					
	应答包格式	0x04+4/12	0x00	Index_id	0xF8	Operation_status	UID（多字节）	CRC				
	指令详解	（1）功能说明：查询设备唯一UID。 （2）输入参数：Index_id（设备下标ID）、UID_Type （3）返回参数：Operation_status、UID （4）指令代码：0xF8 注1：Operation_status 操作状态 0：操作失败 1：操作成功 注2：UID_Type UID的类型 0：4byte UID 1：12byte UID										

查询升级程序版本号	指令包格式	0x03	0x00	Index_id	0xF9	CRC						
	应答包格式	0x04	0x00	Index_id	0xF9	Bootloader_version	CRC					
	指令详解	（1）功能说明：查询升级程序版本号。 （2）输入参数：Index_id（设备下标ID） （3）返回参数：Bootloader_version （4）指令代码：0xF9										
查询设备软件版本号	指令包格式	0x03	0x00	Index_id	0xFA	CRC						
	应答包格式	0x04	0x00	Index_id	0xFA	Firmware_version	CRC					
	指令详解	（1）功能说明：查询设备软件版本号。 （2）输入参数：Index_id（设备下标ID） （3）返回参数：Firmware_version （4）指令代码：0xFA										
查询设备类型	指令包格式	0x03	0x00	Index_id	0xFB	CRC						
	应答包格式	0x05	0x00	Index_id	0xFB	Device_type_low	Device_type_high	CRC				
	指令详解	（1）功能说明：查询设备类型。 （2）输入参数：Index_id（设备下标ID） （3）返回参数：Device_type （4）指令代码：0xFB 注1：Device_type = (uint16_t)((Device_type_high << 8) | Device_type_low) 注2：Unit ChainBus的设备类型码是0x0006										
枚举请求	指令包格式	None										
	应答包格式	0x03	0x00	0xFF	0xFC	CRC						
	指令详解	（1）功能说明：枚举请求，chain链路变更末端设备发送、以及设备上电发送，通知主机更新链路设备状态。 （2）输入参数：none （3）返回参数：none （4）指令代码：0xFC										
心跳包	指令包格式	0x03	0x00	0xFF	0xFD	CRC						
	应答包格式	0x03	0x00	0xFF	0xFD	CRC						
	指令详解	（1）功能说明：心跳包，chain设备之间定时通信，可以自发现自己是不是末端设备，主机也可以通过心跳包来判断是否有chain设备连接。 （2）输入参数：none （3）返回参数：none （4）指令代码：0xFD										
枚举	指令包格式	0x04	0x00	0xFF	0xFE	Send_num	CRC					
	应答包格式	0x04	0x00	0xFF	0xFE	Receive_num	CRC					
	指令详解	（1）功能说明：枚举获取链路设备连接个数。 （2）输入参数：Send_num（默认0，用于记录设备个数） （3）返回参数：Receive_num（数值代表设备个数） （4）指令代码：0xFE										

注1：数据包最大长度是 256 byte、串口
注2：数据长度是Index_id到CRC，包括Index_id以及CRC不包括数据长度本身
注3：CRC 计算时，需要排除包头、包尾、长度和 CRC 字段本身，仅对其余数据求和
注4：串口通讯波特率115200，8位数据位，1停止位，无校验位

```
uint8_t calculateCRC(const uint8_t *buffer, uint16_t size)
{
    uint8_t crc8 = 0;
    for (uint8_t i = 4; i < (size - 3); i++) {
        crc8 += buffer[i];
    }
    return crc8;
}
```