

M5Stack Chain Mono Protocol							All packets start with 0xAA 0x55 and end with 0x55 0xAA						V1.0 (Version)
													2026/1/19
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	
设置显示模式	指令包格式	0x04	0x00	Index_id	0x10	Mode	CRC						
	应答包格式	0x04	0x00	Index_id	0x10	Operation_status	CRC						
	指令详解	(1) 功能说明: 设置显示模式 (2) 输入参数: Index_id (设备下标ID)、Mode (显示模式) (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: Mode 显示模式 (默认0) 0: 像素点控制模式 (直接设置像素显示) 1: 滚动字符串模式 (滚动显示字符串)											
获取显示模式	指令包格式	0x03	0x00	Index_id	0x11	CRC							
	应答包格式	0x04	0x00	Index_id	0x11	Mode	CRC						
	指令详解	(1) 功能说明: 获取显示模式 (2) 输入参数: Index_id (设备下标ID) (3) 返回参数: Mode (显示模式) (4) 指令代码: 0x11 注1: Mode 显示模式 (默认0) 0: 像素点控制模式 (直接设置像素显示) 1: 滚动字符串模式 (滚动显示字符串)											
设置像素	指令包格式	0x04 + Num	0x00	Index_id	0x30	Num	Draw_pixel0	Draw_pixel1	Draw_pixel2	Draw_pixel3	...	CRC	
	应答包格式	0x04	0x00	Index_id	0x30	Operation_status	CRC						
	指令详解	(1) 功能说明: 设置像素。 (2) 输入参数: Index_id (设备下标ID)、Num (要设置的像素数量)、Draw_pixel (像素点值设置, 包含XY坐标) (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 2: 模式不匹配 (需要配置为像素点控制模式) 注2: Num范围: 1-64 注3: Draw_pixel bit [7]: 保留 bit [6]: 状态 (0=灭, 1=亮) bit [5:3]: X坐标 (0-7) bit [2:0]: Y坐标 (0-7)											
全屏显示缓存刷新	指令包格式	0x0B	0x00	Index_id	0x31	Display_buffer0	Display_buffer1	Display_buffer2	Display_buffer3	...	Display_buffer7	CRC	
	应答包格式	0x04	0x00	Index_id	0x31	Operation_status	CRC						
	指令详解	(1) 功能说明: 全屏显示缓存刷新 (2) 输入参数: Index_id (设备下标ID)、Display_buffer (全屏显示缓存) (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 (需要配置为像素点控制模式) 注2: Display_buffer Display_buffer0 → Y: 第0行数据 Display_buffer1 → Y: 第1行数据 ... Bit7→Bit0 对应 X坐标 0-7, 高位在前 0=灭, 1=亮											
获取像素值	指令包格式	0x04 + Num	0x00	Index_id	0x32	Num	Coordinate_xy0	Coordinate_xy1	Coordinate_xy2	Coordinate_xy3	...	CRC	
	应答包格式	0x03 + Num	0x00	Index_id	0x32	Pixel_state0	Pixel_state1	Pixel_state2	Pixel_state3	...	CRC		
	指令详解	(1) 功能说明: 获取像素值。 (2) 输入参数: Index_id (设备下标ID)、Num (要获取的像素数量)、Coordinate_xy (像素坐标) (3) 返回参数: Pixel_state (像素点值) (4) 指令代码: 0x32 注1: Coordinate_xy bit [7:6]: 保留 bit [5:3]: X坐标 (0-7) bit [2:0]: Y坐标 (0-7) 注2: Pixel_state 0: 灭 1: 亮											
获取全屏显示缓存	指令包格式	0x03	0x00	Index_id	0x33	CRC							
	应答包格式	0x0B	0x00	Index_id	0x33	Display_buffer0	Display_buffer1	Display_buffer2	Display_buffer3	...	Display_buffer7	CRC	
	指令详解	(1) 获取整个屏幕刷新状态: 获取全屏显示缓存 (2) 输入参数: Index_id (设备下标ID) (3) 返回参数: Display_buffer (全屏显示缓存) (4) 指令代码: 0x33 注1: Display_buffer Display_buffer0 → Y: 第0行数据 Display_buffer1 → Y: 第1行数据 ... Bit7→Bit0 对应 X坐标0-7, 高位在前 0=灭, 1=亮											
显示字符	指令包格式	0x05	0x00	Index_id	0x34	Char	Offset	CRC					
	应答包格式	0x04	0x00	Index_id	0x34	Operation_status	CRC						
	指令详解	(1) 功能说明: 显示字符 (2) 输入参数: Index_id (设备下标ID)、Char (要显示字符的ASCII码)、Offset (显示的位置偏移) (3) 返回参数: Operation_status (4) 指令代码: 0x34 注1: Operation_status 操作状态 0: 操作失败 1: 操作成功 2: 模式不匹配 (需要配置为像素点控制模式) 注2: Char 要显示字符的ASCII码, 支持ASCII在32-127之间的英文字母, 数字和符号, 字体尺寸为5x7 注3: Offset [7:4]: x轴偏移, 0-7 [3:0]: y轴偏移, 0-7											

显示滚动字符串	指令包格式	0x07+String_length	0x00	Index_id	0x40	Scroll_mode	Scroll_interval_low	Scroll_interval_high	String_length	char1	...	CRC
	应答包格式	0x04	0x00	Index_id	0x40	Operation_status	CRC					
	指令详解	<p>(1) 功能说明: 显示滚动字符串</p> <p>(2) 输入参数: Index_id (设备下标ID)、Scroll_mode (滚动模式)、Scroll_interval (滚动速度)、String_length(字符串长度)、char(字符)</p> <p>(3) 返回参数: Operation_status</p> <p>(4) 指令代码: 0x40</p> <p>注1: Operation_status 操作状态</p> <p>0: 操作失败</p> <p>1: 操作成功</p> <p>2: 模式不匹配 (需要配置滚动字符串模式)</p> <p>注2: Scroll_mode</p> <p>[7:4]: 滚动方向 0: 向右滚动, 1: 向左滚动, 2: 向上滚动, 3: 向下滚动</p> <p>[3:0]: 滚动模式 0: 播放一次停止, 1: 循环滚动, 2: 来回反弹</p> <p>注3: Scroll_interval</p> <p>Scroll_interval移动速度单位为ms/pixel, 每个Scroll_interval时间移动一个像素</p> <p>Scroll_interval = (uint16_t)((Scroll_interval_high << 8) Scroll_interval_low) 范围 0~65535, 单位ms/pixel</p> <p>注4: String_length</p> <p>滚动字符串的长度</p> <p>注5: char</p> <p>显示的字符, 支持ASCII在32~127之间的英文字母, 数字和符号, 字体尺寸为5x7</p>										
获取滚动字符串	指令包格式	0x03	0x00	Index_id	0x41	CRC						
	应答包格式	0x07+String_length	0x00	Index_id	0x41	Scroll_mode	Scroll_interval_low	Scroll_interval_high	String_length	Char1	...	CRC
	指令详解	<p>(1) 功能说明: 获取滚动字符串</p> <p>(2) 输入参数: Index_id (设备下标ID)</p> <p>(3) 返回参数: Scroll_mode (滚动模式)、Scroll_interval_low、Scroll_interval_high (滚动速度)、String_length(字符串长度)、char(字符)</p> <p>(4) 指令代码: 0x41</p> <p>注1: Scroll_mode</p> <p>[7:4]: 滚动方向 0: 向右滚动, 1: 向左滚动, 2: 向上滚动, 3: 向下滚动</p> <p>[3:0]: 滚动模式 0: 播放一次停止, 1: 循环滚动, 2: 来回反弹</p> <p>注2: Scroll_interval</p> <p>Scroll_interval移动速度单位为ms/pixel, 每个Scroll_interval时间移动一个像素</p> <p>Scroll_interval = (uint16_t)((Scroll_interval_high << 8) Scroll_interval_low) 范围 0~65535, 单位ms/pixel</p> <p>注3: String_length</p> <p>滚动字符串的长度</p> <p>注4: Char</p> <p>显示的字符, 支持ASCII在32~127之间的英文字母, 数字和符号, 字体尺寸为5x7</p>										
设置滚动字符串状态	指令包格式	0x04	0x00	Index_id	0x42	Scroll_state	CRC					
	应答包格式	0x04	0x00	Index_id	0x42	Operation_status	CRC					
	指令详解	<p>(1) 功能说明: 设置滚动字符串状态</p> <p>(2) 输入参数: Index_id (设备下标ID), Scroll_state (滚动状态)</p> <p>(3) 返回参数: Operation_status</p> <p>(4) 指令代码: 0x42</p> <p>注1: Operation_status 操作状态</p> <p>0: 操作失败</p> <p>1: 操作成功</p> <p>2: 模式不匹配 (需要配置滚动字符串模式)</p> <p>注2: Scroll_state</p> <p>0: 开始/继续滚动</p> <p>1: 暂停滚动 (画面保持当前字符静止显示)</p> <p>2: 停止并清除 (清除显示的字符, 下次设置开始将从头开始)</p>										
获取滚动字符串状态	指令包格式	0x03	0x00	Index_id	0x43	CRC						
	应答包格式	0x04	0x00	Index_id	0x43	Scroll_state	CRC					
	指令详解	<p>(1) 功能说明: 设置滚动字符串状态</p> <p>(2) 输入参数: Index_id (设备下标ID)</p> <p>(3) 返回参数: Scroll_state (滚动状态)</p> <p>(4) 指令代码: 0x43</p> <p>注1: Scroll_state</p> <p>0: 正在滚动</p> <p>1: 暂停状态</p> <p>2: 空闲/停止状态</p>										
设置屏幕旋转角度	指令包格式	0x05	0x00	Index_id	0xE0	Screen_Rotation	Save_to_flash	CRC				
	应答包格式	0x04	0x00	Index_id	0xE0	Operation_status	CRC					
	指令详解	<p>(1) 功能说明: 设置屏幕旋转角度</p> <p>(2) 输入参数: Index_id (设备下标ID)、Screen_Rotation(屏幕旋转角度)、Save_to_flash (是否保存到内部Flash)</p> <p>(3) 返回参数: Operation_status</p> <p>(4) 指令代码: 0xE0</p> <p>注1: Operation_status 操作状态</p> <p>0: 操作失败</p> <p>1: 操作成功</p> <p>注2: Screen_Rotation</p> <p>0: 默认显示角度</p> <p>1: 顺时针旋转90°</p> <p>2: 顺时针旋转180°</p> <p>3: 顺时针旋转270° (逆时针90°)</p> <p>注3: Save_to_flash 是否保存到内部Flash</p> <p>0: 不保存</p> <p>1: 保存</p> <p>注4: 设置成功即刻生效</p> <p>注5: 保存至内部 Flash 需要擦除页面, 过程较为耗时 (约20ms), 期间会关闭串口中断, 且频繁操作会影响设备寿命。</p>										
获取屏幕旋转角度	指令包格式	0x03	0x00	Index_id	0xE1	CRC						
	应答包格式	0x04	0x00	Index_id	0xE1	Screen_Rotation	CRC					
	指令详解	<p>(1) 功能说明: 获取屏幕旋转角度</p> <p>(2) 输入参数: Index_id (设备下标ID)</p> <p>(3) 返回参数: Screen_Rotation(屏幕旋转角度)</p> <p>(4) 指令代码: 0xE1</p> <p>注1: Screen_Rotation</p> <p>0: 默认显示角度</p> <p>1: 顺时针旋转90°</p> <p>2: 顺时针旋转180°</p> <p>3: 顺时针旋转270° (逆时针90°)</p>										

设置屏幕亮度等级	指令包格式	0x05	0x00	Index_id	0xE2	Brightness_level	Save_to_flash	CRC											
	应答包格式	0x04	0x00	Index_id	0xE2	Operation_status	CRC												
	指令详解	<p>(1) 功能说明: 设置屏幕亮度等级。 (2) 输入参数: Index_id (设备下标ID)、Brightness_level(屏幕亮度等级)、Save_to_flash (是否保存到内部Flash) (3) 返回参数: Operation_status (4) 指令代码: 0xE2 注1: Operation_status 操作状态 0: 操作失败 1: 操作成功 注2: Brightness_level 0-7: 7个亮度等级 默认7 注3: Save_to_flash 是否保存内部Flash 0: 不保存 1: 保存 注4: 设置成功即刻生效 注5: 设置成功内部 Flash 需要擦除页面, 过程较为耗时 (约20ms), 期间会关闭串口中断, 且频繁操作会影响设备寿命。</p>																	
获取屏幕亮度等级	指令包格式	0x03	0x00	Index_id	0xE3	CRC													
	应答包格式	0x04	0x00	Index_id	0xE3	Brightness_level	CRC												
	指令详解	<p>(1) 功能说明: 获取屏幕旋转角度。 (2) 输入参数: Index_id (设备下标ID) (3) 返回参数: Brightness_level(屏幕亮度等级) (4) 指令代码: 0xE3 注1: Brightness_level 0-7: 7个亮度等级 默认7</p>																	
清屏	指令包格式	0x03	0x00	Index_id	0xE4	CRC													
	应答包格式	0x04	0x00	Index_id	0xE4	Operation_status	CRC												
	指令详解	<p>(1) 功能说明: 清屏。 (2) 输入参数: Index_id (设备下标ID) (3) 返回参数: Operation_status (4) 指令代码: 0xE4 注1: Operation_status 操作状态 0: 操作失败 1: 操作成功</p>																	
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	<p>(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</p>																	
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	<p>(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</p>																	
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	<p>(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</p>																	
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	<p>(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: Mono的设备类型码是0x000D</p>																	
枚举请求	指令包格式	None																	
	应答包格式	0x03	0x00	0xFF	0xFC	CRC													
	指令详解	<p>(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</p>																	
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	<p>(1) 功能说明: 心跳包, chain设备之间定时通信, 可以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</p>																	
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	<p>(1) 功能说明: 枚举获取下级设备的个数。 (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</p>																	

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;
}
```