

M5Stack Unit Roller485 Protocol(Setting&Control)													All returns start with 0xAA 0x55				V1 (FW Version)	
Type	Command	Bytes											CRC8		note			
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14		
	Cmd ID	Motor ID	Data1 Byte0	Data1 Byte1	Data1 Byte2	Data1 Byte3	Data2 Byte0	Data2 Byte1	Data2 Byte2	Data2 Byte3	Data3 Byte0	Data3 Byte1	Data3 Byte2	Data3 Byte3	CRC8			
Setting	Motor switch	0x00	ID	Output	0	0	0	0	0	0	0	0	0	0	CRC8	Output: 0: Motor off 1: Motor on		
	Return	0x10	ID	Output	0	0	0	0	0	0	0	0	0	0	CRC8			
Setting	Mode setting (can be saved to flash)	0x01	ID	Mode	0	0	0	0	0	0	0	0	0	0	CRC8	Mode: 1: Speed Mode 2: Position Mode 3: Current Mode 4: Encoder Mode		
	Return	0x11	ID	Mode	0	0	0	0	0	0	0	0	0	0	CRC8			
Setting	Remove protection	0x06	ID	0	0	0	0	Release Jam protection	0	0	0	0	0	0	CRC8	Release Jam protection: Send 1 to unprotect		
	Return	0x06	ID	0	0	0	0	Release Jam protection	0	0	0	0	0	0	CRC8			
Setting	Save to flash	0x07	ID	Save to flash	0	0	0	0	0	0	0	0	0	0	CRC8	Save to flash: Send 1 save parameters to flash		
	Return	0x17	ID	Save to flash	0	0	0	0	0	0	0	0	0	0	CRC8			
Setting	Encoder	0x08	ID	Encoder-byte0	Encoder-byte1	Encoder-byte2	Encoder-byte3	0	0	0	0	0	0	0	CRC8	Encoder value(int32_0) Encoder = Encoder-byte0 + Encoder-byte1 * 256 + Encoder-byte2 * 65536 + Encoder-byte3 * 16777216		
	Return	0x18	ID	Encoder-byte0	Encoder-byte1	Encoder-byte2	Encoder-byte3	0	0	0	0	0	0	0	CRC8			
Setting	Button switching mode enable	0x09	ID	Button switching	0	0	0	0	0	0	0	0	0	0	CRC8	0: Off; 1: Press and hold for 5S to switch modes in running mode.		
	Button switching mode enable	0x19	ID	Button switching	0	0	0	0	0	0	0	0	0	0	CRC8			
Setting	RGB (RGB Mode and RGB Brightness can be saved to flash)	0x0A	ID	RGB-B	RGB-G	RGB-R	RGB Mode	RGB Brightness	0	0	0	0	0	0	CRC8	RGB Mode: 0: Sys-default 1: User-define RGB Brightness: 0-100		
	Return	0x1A	ID	RGB-B	RGB-G	RGB-R	RGB Mode	RGB Brightness	0	0	0	0	0	0	CRC8			
Setting	485 BPS (can be saved to flash)	0x0B	ID	485 BPS	0	0	0	0	0	0	0	0	0	0	CRC8	BPS: 0,115200bps; 1, 19200bps; 2, 9600bps		
	Return	0x1B	ID	485 BPS	0	0	0	0	0	0	0	0	0	0	CRC8			
Setting	485 ID (can be saved to flash)	0x0C	ID	485 ID	0	0	0	0	0	0	0	0	0	0	CRC8	485 ID: 0-255		
	Return	0x1C	ID	485 ID	0	0	0	0	0	0	0	0	0	0	CRC8			
Setting	Motor Jam Protection	0x0D	ID	Motor Jam Protection	0	0	0	0	0	0	0	0	0	0	CRC8	Motor Jam Protection: 0:Disable; 1: Enable		
	Return	0x1D	ID	Motor Jam Protection	0	0	0	0	0	0	0	0	0	0	CRC8			
Setting	Motor Position Over Range Protection (can be saved to flash)	0x0E	ID	Motor Position Over Range Protection	0	0	0	0	0	0	0	0	0	0	CRC8	Motor Position Over Range Protection: 0:Disable; 1: Enable		
	Return	0x1E	ID	Motor Position Over Range Protection	0	0	0	0	0	0	0	0	0	0	CRC8			
Speed mode configuration	Speed & Max Current X100 Int	0x20	ID	Speed-byte0	Speed-byte1	Speed-byte2	Speed-byte3	Max Current-byte0	Max Current-byte1	Max Current-byte2	Max Current-byte3	0	0	0	CRC8	Speed Setting(int32_0) = Speed Setting-byte0 + Speed Setting-byte1 * 256 + Speed Setting-byte2 * 65536 + Speed Setting-byte3 * 16777216 Actual Speed Setting = Position Setting/100 Range: -2100000000-2100000000 Max Current = Max Current-byte0 + Max Current-byte1 * 256 + Max Current-byte2 * 65536 + Max Current-byte3 * 16777216 Actual Max Current = Max Current/100 range: -120000-120000		
	Return	0x30	ID	Speed-byte0	Speed-byte1	Speed-byte2	Speed-byte3	Max Current-byte0	Max Current-byte1	Max Current-byte2	Max Current-byte3	0	0	0	CRC8			
Speed mode configuration	Speed PID Int (can be saved to flash)	0x21	ID	P-X100000-byte0	P-X100000-byte1	P-X100000-byte2	P-X100000-byte3	I-X100000000-byte0	I-X100000000-byte1	I-X100000000-byte2	I-X100000000-byte3	D-X1000000-byte0	D-X1000000-byte1	D-X1000000-byte2	D-X1000000-byte3	CRC8	P/I/D(int32_0): PID = PID-byte0 + PID-byte1 * 256 + PID-byte2 * 65536 + PID-byte3 * 16777216 For example: P=0.00001, P setting value=0.00001*1000000=1, P-byte0=1, P-byte1=0, P-byte2=0, P-byte3=0	
	Return	0x31	ID	P-byte0	P-byte1	P-byte2	P-byte3	I-byte0	I-byte1	I-byte2	I-byte3	D-byte0	D-byte1	D-byte2	D-byte3	CRC8		
Position mode configuration	Position & Max Current X100 Int	0x22	ID	Position-byte0	Position-byte1	Position-byte2	Position-byte3	Max Current-byte0	Max Current-byte1	Max Current-byte2	Max Current-byte3	0	0	0	CRC8	Position Setting(int32_0) = Position Setting-byte0 + Position Setting-byte1 * 256 + Position Setting-byte2 * 65536 + Position Setting-byte3 * 16777216 Actual Position Setting = Position Setting/100 range: -2100000000-2100000000 Max Current = Max Current-byte0 + Max Current-byte1 * 256 + Max Current-byte2 * 65536 + Max Current-byte3 * 16777216 Actual Max Current = Max Current/100 range: -120000-120000		
	Return	0x32	ID	Position-byte0	Position-byte1	Position-byte2	Position-byte3	Max Current-byte0	Max Current-byte1	Max Current-byte2	Max Current-byte3	0	0	0	CRC8			
Position mode configuration	Position PID Int (can be saved to flash)	0x23	ID	P-X1000000-byte0	P-X1000000-byte1	P-X1000000-byte2	P-X1000000-byte3	I-X100000000-byte0	I-X100000000-byte1	I-X100000000-byte2	I-X100000000-byte3	D-X1000000-byte0	D-X1000000-byte1	D-X1000000-byte2	D-X1000000-byte3	CRC8	P/I/D(int32_1): PID = PID-byte0 + PID-byte1 * 256 + PID-byte2 * 65536 + PID-byte3 * 16777216 For example: P=0.00001, P setting value=0.00001*1000000=1, P-byte0=1, P-byte1=0, P-byte2=0, P-byte3=0	
	Return	0x33	ID	P-byte0	P-byte1	P-byte2	P-byte3	I-byte0	I-byte1	I-byte2	I-byte3	D-byte0	D-byte1	D-byte2	D-byte3	CRC8		
Current mode configuration	Current X100 Int	0x24	ID	Current-byte0	Current-byte1	Current-byte2	Current-byte3	0	0	0	0	0	0	0	CRC8	Current Setting(int32_0) = Current Setting-byte0 + Current Setting-byte1 * 256 + Current Setting-byte2 * 65536 + Current Setting-byte3 * 16777216 Actual Current Setting = Current Setting/100 range: -120000-120000		
	Return	0x34	ID	Current-byte0	Current-byte1	Current-byte2	Current-byte3	0	0	0	0	0	0	0	CRC8			

```

(1) CRC8:
uint8_t crc8(uint8_t *data, uint8_t len)
{
    uint8_t crc = 0;
    crc = 0x00;
    while(len--)
    {
        crc ^= *data++;
        for(i = 0; i < 8; i++)
        {
            if(crc & 0x01)
            {
                crc = (crc >> 1) ^ 0x8c;
            }
            else crc >>= 1;
        }
    }
    return crc;
}

```

(2) All returned messages begin with 0xAA 0x55, which are not included in the CRC check.

M5Stack Unit Roller485 Protocol(Readback)										All returns start with 0xAA 0x55								V1 (FW Version)	
																		2025/6/5	
																		note	
bytes	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	
command	Cmd ID	Motor ID	Data1 Byte0	Data1 Byte1	Data1 Byte2	Data1 Byte3	Data2 Byte0	Data2 Byte1	Data2 Byte2	Data2 Byte3	Data3 Byte0	Data3 Byte1	Data3 Byte2	Data3 Byte3	Data4	Data5	Data6	CRC8	
Readback 0	0x40	ID	0	CRC8															
Return	0x50	ID	Speed Readback-byte0	Speed Readback-byte1	Speed Readback-byte2	Speed Readback-byte3	Position Readback-byte0	Position Readback-byte1	Position Readback-byte2	Position Readback-byte3	Current Readback-byte0	Current Readback-byte1	Current Readback-byte2	Current Readback-byte3	Mode	Status	Error	CRC8	<p>Speed Readback = Speed Readback-byte0 + Speed Readback-byte1 * 256 + Speed Readback-byte2 * 65536 + Speed Readback-byte3 * 16777216 Actual Speed Readback = Speed Readback/100</p> <p>Position Readback = Position Readback-byte0 + Position Readback-byte1 * 256 + Position Readback-byte2 * 65536 + Position Readback-byte3 * 16777216 Actual Position Readback = Position Readback/100</p> <p>Current Readback = Current Readback-byte0 + Current Readback-byte1 * 256 + Current Readback-byte2 * 65536 + Current Readback-byte3 * 16777216 Actual Current Readback = Current Readback/100</p> <p>Mode 1: Speed Mode 2: Position Mode 3: Current Mode 4: Encoder Mode</p> <p>Status: 0: Standby; 1: Running; 2: Error</p> <p>Error: 1: Overvoltage; 2: Stalled; 4: Over Range</p>
Readback 1	0x41	ID	0	CRC8															
Return	0x51	ID	VIN X100-byte0	VIN X100-byte1	VIN X100-byte2	VIN X100-byte3	Temp-byte0	Temp-byte1	Temp-byte2	Temp-byte3	Encoder Counter-byte0	Encoder Counter-byte1	Encoder Counter-byte2	Encoder Counter-byte3	RGB Mode	RGB Brightness	0	CRC8	<p>VIN X100 = VIN X100-byte0 + VIN X100-byte1 * 256 + VIN X100-byte2 * 65536 + VIN X100-byte3 * 16777216 Actual VIN = VIN X100/100</p> <p>Temp = Temp-byte0 + Temp-byte1 * 256 + Temp-byte2 * 65536 + Temp-byte3 * 16777216</p> <p>Encoder Counter = Encoder Counter-byte0 + Encoder Counter-byte1 * 256 + Encoder Counter-byte2 * 65536 + Encoder Counter-byte3 * 16777216 Can only use in Encoder mode</p> <p>RGB Mode: 0, Sys-default 1, User-define</p> <p>RGB Brightness: 0-100</p>
Readback 2	0x42	ID	0	CRC8															
Return	0x52	ID	Speed-P-X100000-byte0	Speed-P-X100000-byte1	Speed-P-X100000-byte2	Speed-P-X100000-byte3	Speed-I-X100000-byte0	Speed-I-X100000-byte1	Speed-I-X100000-byte2	Speed-I-X100000-byte3	Speed-D-X100000-byte0	Speed-D-X100000-byte1	Speed-D-X100000-byte2	Speed-D-X100000-byte3	RGB-B	RGB-G	RGB-R	CRC8	<p>P/I/D(uint32_t): PID = PID-byte0 + PID-byte1 * 256 + PID-byte2 * 65536 + PID-byte3 * 16777216 For example: P=0.00001, P setting value=0.00001*100000=1, P-byte0=1, P-byte1=0, P-byte2=0, P-byte3=0</p>
Readback 3	0x43	ID	0	CRC8															
Return	0x53	ID	Position-P-X100000-byte0	Position-P-X100000-byte1	Position-P-X100000-byte2	Position-P-X100000-byte3	Position-I-X100000-byte0	Position-I-X100000-byte1	Position-I-X100000-byte2	Position-I-X100000-byte3	Position-D-X100000-byte0	Position-D-X100000-byte1	Position-D-X100000-byte2	Position-D-X100000-byte3	485 ID	485 BPS	Button switching mode enable	CRC8	<p>P/I/D(uint32_t): PID = PID-byte0 + PID-byte1 * 256 + PID-byte2 * 65536 + PID-byte3 * 16777216 For example: P=0.00001, P setting value=0.00001*100000=1, P-byte0=1, P-byte1=0, P-byte2=0, P-byte3=0</p>
<pre> (1) CRC8: uint8_t crc8(uint8_t *data, uint8_t len) { uint8_t crc, i; crc = 0x00; while(len--) { crc ^= *data++; for(i = 0; i < 8; i++) { if(crc & 0x01) { crc = (crc >> 1) ^ 0x8c; } else crc >>= 1; } } return crc; } </pre> <p>(2) 所有返回的消息以0xAA 0x55开头, 0xAA 0x55 不参与CRC校验</p>																			

M5Stack Unit Roller485 Protocol(485->I2C)										All returns start with 0xAA 0x55														V1 (FW Version)				
																								2025/6/3				
																								note				
bytes	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24			
command	Cmd ID	Motor ID	Data1 Byte0	Data1 Byte1	Data1 Byte2	Data1 Byte3	Data2 Byte0	Data2 Byte1	Data2 Byte2	Data2 Byte3	Data3 Byte0	Data3 Byte1	Data3 Byte2	Data3 Byte3	Data4	Data5	Data6	CRC8										
I2C Read	0x60	ID	i2c address	i2c reg address len	i2c reg address bytes0	i2c reg address bytes1	i2c data len	CRC8																	i2c reg address len: 0, 1 byte address; 1, 2 bytes address			
																									i2c reg address: 1. i2c reg address len = 0, i2c reg address = i2c reg address bytes0 2. i2c reg address len = 1, i2c reg address = i2c reg address bytes0 + (i2c reg address bytes1 * 255)			
Return	0x70	ID	is read success	0	i2c data len	0	0	0	data0	data1	data2	data3	data4	data5	data6	data7	data8	data9	data10	data11	data12	data13	data14	data15	CRC8	i2c data len: The maximum number of bytes to be read is 16 is read success: 1, Read successful; 0, Read failed		
I2C Write	0x61	ID	i2c address	i2c reg address len	i2c reg address bytes0	i2c reg address bytes1	i2c data len	0	data0	data1	data2	data3	data4	data5	data6	data7	data8	data9	data10	data11	data12	data13	data14	data15	CRC8	i2c reg address len: 0, 1 byte address; 1, 2 bytes address		
																									i2c reg address: 1. i2c reg address len = 0, i2c reg address = i2c reg address bytes0 2. i2c reg address len = 1, i2c reg address = i2c reg address bytes0 + (i2c reg address bytes1 * 255)			
Return	0x71	ID	is write success	CRC8																				is write success: 1, Write successful; 0, Write failed				
I2C Read Raw	0x62	ID	i2c address	i2c data len	CRC8																							
Return	0x72	ID	is read success	0	i2c data len	0	0	0	data0	data1	data2	data3	data4	data5	data6	data7	data8	data9	data10	data11	data12	data13	data14	data15	CRC8	i2c data len: The maximum number of bytes to be read is 16 is read success: 1, Read successful; 0, Read failed		
I2C Write Raw	0x63	ID	i2c address	i2c data len	is stop bit	0	0	0	data0	data1	data2	data3	data4	data5	data6	data7	data8	data9	data10	data11	data12	data13	data14	data15	CRC8	is stop bit: 0, have no stop bit 1, have stop bit		
Return	0x73	ID	is write success	CRC8																				is write success: 1, Write successful; 0, Write failed				
<pre> (1) CRC8: uint8_t crc8(uint8_t *data, uint8_t len) { uint8_t crc, i; crc = 0x00; while(len--) { crc ^= *data++; for(i = 0; i < 8; i++) { if(crc & 0x01) { crc = (crc >> 1) ^ 0x8c; } else crc >>= 1; } } return crc; } </pre>																												
(2) 所有返回的消息以0xAA 0x55开头, 0xAA 0x55 不参与CRC校验																												