

VSMD104/105_025T
步进电机控制驱动器
Modbus RTU 协议



北京伟恩斯技术有限公司

www.vincetech.com

【序言】

感谢您购买本公司微型步进电机驱动器，本使用说明书将详细介绍该产品的各项功能和操作方法，让您充分感受本产品带给您的方便、快捷和安全。

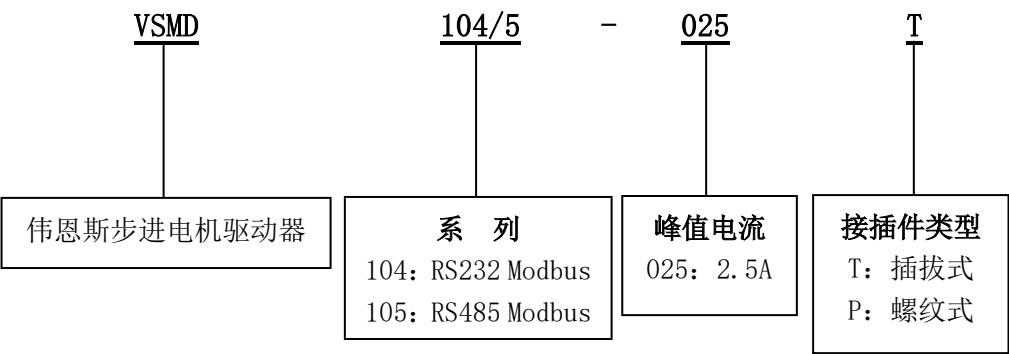
【安全使用说明】

- 使用前请务必仔细阅读本使用说明书，按照说明书要求进行接线，以免损坏产品；
- 请不要将本产品暴露在潮湿过高的地方；
- 请不要将接线端子短路，否则会毁坏产品；
- 如果步进电机额定电流大于 2.5A，请将驱动器电流调整到 2.5A 以下，以免损坏电机；

【联系方式】

北京伟恩斯技术有限公司
地址：北京市昌平区科技园区生命园路 29 号 1 幢 B316
电话：18612497280
邮箱：xu_guoen@vincetech.com
网址：www.vincetech.com

【命名规则】



目录

1.	简介.....	1
2.	接线方式.....	2
2.1.	接线端口描述.....	2
2.2.	LED 灯显示.....	错误！未定义书签。
2.3.	恢复出厂设置.....	4
2.4.	传感器使用.....	4
2.4.1.	NPN 型传感器接线方式.....	4
2.4.2.	PNP 型传感器接线方式.....	5
2.4.3.	机械开关接线方式.....	5
3.	指令和反馈.....	6
3.1.	指令格式.....	6
3.2.	功能码.....	6
3.2.1.	功能码 0x03.....	6
3.2.2.	功能码 0x04.....	7
3.2.3.	功能码 0x06.....	7
3.2.4.	功能码 0x10.....	7
3.3.	指令异常.....	8
3.4.	驱动器状态寄存器（3 区）.....	9
3.4.1.	读取当前速度.....	9
3.4.2.	读取当前位置.....	9
3.4.3.	读取当前状态信息.....	9
3.5.	驱动器控制和设置寄存器（4 区）.....	11
3.5.1.	控制寄存器总览.....	11
3.5.2.	驱动器控制寄存器地址 00.....	12
3.5.3.	目标位置设置寄存器地址 0102.....	12
3.5.4.	速度设置寄存器地址 0304.....	12
3.5.5.	加速度设置寄存器地址 0506.....	13
3.5.6.	减速度设置寄存器地址 0708.....	13
3.5.7.	加速电流设置寄存器地址 0910.....	13
3.5.8.	运行电流设置寄存器地址 1112.....	13
3.5.9.	保持电流设置寄存器地址 1314.....	13
3.5.10.	S1 传感器功能设置寄存器地址 16.....	14
3.5.11.	S2 传感器功能设置寄存器地址 17.....	14
3.5.12.	S3 传感器功能设置寄存器地址 18.....	15
3.5.13.	归零功能设置寄存器地址 24.....	15
3.5.14.	归零用传感器类型设置寄存器地址 25.....	15
3.5.15.	归零用传感器设置寄存器地址 26.....	16
3.5.16.	归零速度设置寄存器地址 2728.....	16
3.5.17.	归零后停止位置设置寄存器地址 2930.....	16

3.5.18.	站点 ID 号设置寄存器地址 31.....	16
3.5.19.	通讯波特率设置寄存器地址 3233.....	17
3.5.20.	细分设置寄存器地址 34.....	17
3.5.21.	上电自动归零设置寄存器地址 39.....	17
3.5.22.	上电自动使能设置寄存器地址 40.....	17
3.5.23.	数据格式设置寄存器地址 41.....	18
4.	归零功能设置.....	19
4.1.1.	归零功能简介.....	19
4.1.2.	归零参数设置.....	20
4.1.3.	归零动作执行.....	20
5.	PC 端控制配置工具.....	20
5.1.	主界面.....	21
5.2.	打开映射文件.....	21
5.3.	串口连接.....	22
5.4.	驱动器状态寄存器.....	22
5.5.	驱动器控制寄存器.....	23
6.	性能指标.....	25
6.1.	电气性能.....	25
6.2.	使用环境.....	25
7.	附件.....	26
7.1.	外形尺寸图.....	26
7.2.	CRC 校验程序.....	27

1. 简介

VSMD104/5_025 系列驱动器，是基于 Modbus RTU 协议的运动控制和电机驱动一体化的电机控制驱动模块。用户无需了解电机驱动控制的底层内容，通过 Modbus RTU 标准总线协议就可以控制电机的电流、速度、加减速、启动停止等。

【基本参数】

- 输入电压：12 ~ 40VDC
- 峰值电流：2.5A
- 微步细分：1/2/4/8/16/32

【通讯】

- 通讯方式：RS232-MODBUS（VSMD104） RS485-MODBUS（VSMD105）
- 通讯速率：2400 ~ 921600
- 通讯参数：8 位数据位，无奇偶校验位，1 位停止位
- 指令格式：标准 MODBUS-RTU 协议
- 反馈格式：标准 MODBUS-RTU 协议

【结构】

- 铸铝外壳：坚固、散热好
- 外形尺寸：42.5mm × 42.5mm × 16.8mm

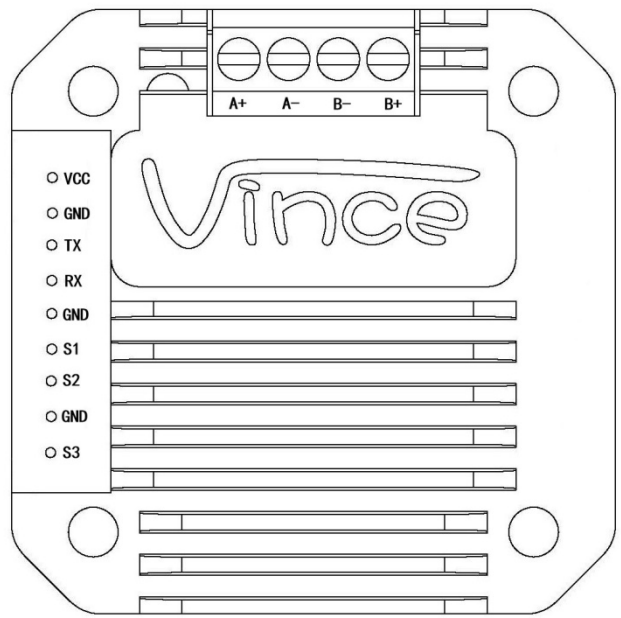
【特点】

- 采用 32 位微处理器控制，运行响应速度更快。
- 32 细分下，电机可达到 1800RPM 的转速。
- 能完成各种复杂运动的控制（平滑加减速、平滑转向等）。
- 可运行在速度模式、位置模式，并能随意互相切换。
- 特有的电流控制模式，能在保持运行平稳的前提下，减小噪声，降低发热量。
- 内置归零功能，根据设置的归零参数，自动完成归零动作，减轻用户工作量。

2. 接线方式

2.1. 接线端口描述

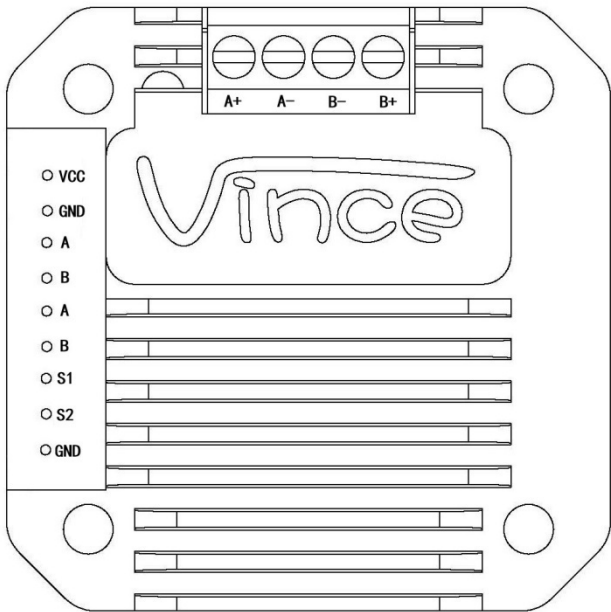
【VSMD104_025T】



接口	说明
A+ A- B- B+	电机接口
VCC	电源正极（12 ~ 40VDC）
GND	电源地
TX	串口 TX
RX	串口 RX
GND	串口地
S1	传感器 1
S2	传感器 2
GND	数字地
S3	传感器 3

※ S1、S2、S3 仅支持 0 ~ 5V 的 TTL 电平。
※ 串口和上位机 232 接口是交叉连接，及 RX 接 TX，TX 接 RX。

【VSMD105_025T】



接口	说明
A+ A- B- B+	电机接口
VCC	电源正极（12 ~ 40VDC）
GND	电源地
A	485 接口 A
B	485 接口 B
A	485 接口 A（级联用）
B	485 接口 B（级联用）
S1	传感器 1（※）
S2	传感器 2（※）
GND	数字地

※ 485 接口最多级联 32 个从设备。
※ S1、S2 传感器接口仅支持 0 ~ 5V 的 TTL 电平。

2.2. 恢复出厂设置

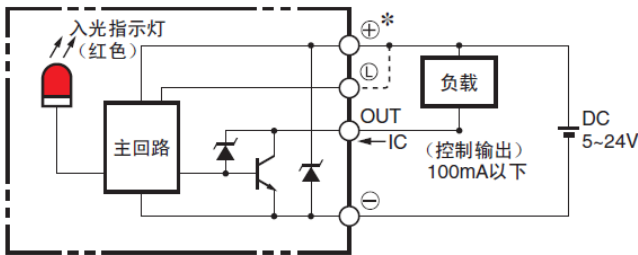
将 S1 和 S2 端子用信号线短接，然后上电，这时 ID 号和波特率就是 1 和 9600（VSMD104_025 没有 ID 号），重新设置 ID 和波特率并保存后，重新上电生效。

2.3. 传感器使用

传感器从性质上来看，主要分为有源和无源的。常用的光电开关，是有源的，微动开关是无源的。如何选择开关，以及如何连接，需要根据实际情况来决定。

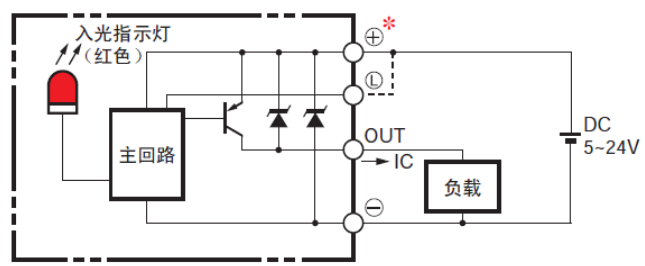
类型	传感器信号（开放状态）	传感器信号（触发状态）
常通/常开	1	0
常闭	0	1

2.3.1. NPN 型传感器接线方式



传感器	接线方式
S1、S2、S3	<p>The diagram shows the wiring for S1, S2, and S3 sensors. The sensor is connected to a VCC (5-24V) supply and a SGND (Signal Ground) terminal. The output terminal (OUT) is connected to the S3 terminal. The control output (IC) is labeled as (控制输出) 100mA以下.</p>

2.3.2. PNP 型传感器接线方式



传感器	接线方式
S1、S2、S3	<p>This diagram shows the specific wiring for sensors S1, S2, and S3. The sensor's 'OUT' pin is connected to 'VCC (5V)', and the 'IC' pin is connected to 'SGND'. The '入光指示灯 (红色)' and '主回路' are also shown within the sensor's internal structure.</p>

2.3.3. 机械开关接线方式

机械开关属于无源传感器，把传感器的 COM 端连接 GND，S1/S2/S3 根据需要（常开，常闭）连接到传感器的 NO/NC 管脚即可。

传感器	接线方式
S1、S2、S3	<p>This diagram shows the wiring for mechanical switch sensors. The 'COM' terminal of the switch is connected to 'SGND', and the 'NO' or 'NC' terminal is connected to 'S3'.</p>

3. 指令和反馈

3.1. 指令格式

VSMD 的指令格式完全兼容标准的 MODBUS-RTU 协议，MODBUS-RTU 基本格式如下：

设备号	功能码	数据	CRC 校验
1 byte	1 byte	n bytes	2 bytes

※ 设备号支持范围为 1 ~ 32，当设备号是 0 时，为广播指令。

※ 数据组织格式为：CDAB。

3.2. 功能码

VSMD 实现了 MODBUS-RTU 的 3、4、6、16 (0x10) 功能码，如下所示：

功能码	说明
0x03	用于读取驱动器设置参数（电流、细分等），四区保持寄存器值
0x04	用于读取驱动器状态（当前速度、位置等），三区状态寄存器值
0x06	用于写入单个寄存器（电流、细分等），写入四区保持寄存器
0x10	用于写入多个寄存器（速度、位置等），写入四区保持寄存器

3.2.1. 功能码 0x03

功能码 3 用于读取保持寄存器的值，及四区存储的驱动器设置参数，地址范围是 0~63，指令格式如下所示：

设备号	功能码	起始地址	数据长度	CRC 校验
1 byte	1 byte	2 bytes	2 bytes	2 bytes

返回值格式如下所示：

设备号	功能码	数据长度	数据 (1~N)	CRC 校验
1 byte	1 byte	2 bytes	N* (2 bytes)	2 bytes

以设备号是 1 为例，读取保持寄存器四区数据指令如下：

发送：01 03 00 00 00 23 04 13

接收：01 03 46 01 01 00 00 00 00 80 00 46 3B 80 00 46 3B 80 00 46 3B 99 9A
 3F 19 CC CD 3E CC 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 00 01 25 80
 00 00 00 05 86 3A

3.2.2. 功能码 0x04

功能码 4 用于读取状态寄存器的值，及三区存储的驱动器状态值，地址范围是 0～19，指令格式如下所示：

设备号	功能码	起始地址	数据长度	CRC 校验
1 byte	1 byte	2 bytes	2 bytes	2 bytes

返回值格式如下所示：

设备号	功能码	字节数	数据 (1～N)	CRC 校验
1 byte	1 byte	1 byte	N* (2 bytes)	2 bytes

以设备号是 1 为例，读取三区状态寄存器指令如下：

发送： 01 04 00 00 00 14 F0 05
接收： 01 04 28 00 00 00 00 EE 72 00 16 21 0F 00 01 53 56 44 4D 30 31 2D
35 32 30 54 35 31 2D 30 2E 31 2E 33 30 31 2E 30 38 30 39 00 32 13
B5

3.2.3. 功能码 0x06

功能码 6 用于写入单个保持寄存器的值，及向四区单个寄存器写入数据，指令格式如下：

设备号	功能码	地址	数据	CRC 校验
1 byte	1 byte	2 bytes	2 bytes	2 bytes

返回值格式如下所示：

设备号	功能码	地址	数据	CRC 校验
1 byte	1 byte	2 byte	2 bytes	2 bytes

以设备号是 1 为例，电机使能的指令如下：

发送： 01 06 00 00 01 01 49 9A
接收： 01 06 00 00 01 01 49 9A

3.2.4. 功能码 0x10

功能码 16 用于写入多个保持寄存器的值，及向四区多个寄存器写入数据，指令格式如下：

设备号	功能码	起始地址	数据长度	字节数	数据 N	CRC 校验
1 byte	1 bytes	2 bytes	2 bytes	1 byte	N*(2 bytes)	2 bytes

返回值格式如下所示：

设备号	功能码	起始地址	数据长度	CRC 校验
1 byte	1 byte	2 byte	2 bytes	2 bytes

以设备号是 1 为例，设置运行速度为 10000 指令如下：

发送：01 10 00 03 00 02 04 40 00 46 1C 95 D3

接收：01 10 00 03 00 02 B1 C8

3.3. 指令异常

由于通讯或者地址错误等原因，当读写寄存器的时候有时会发生异常。当发生异常的时候，驱动器返回异常指令，格式如下：

设备号	功能码（高位置 1）	错误码	CRC 校验
1 byte	1 byte	1 bytes	2 bytes

【错误码】

错误码	名称	说明
01	不合法功能代码	功能码 3、4、6、16 以外的功能码不支持
02	不合法寄存器地址	寄存器地址不正确或者超出范围
03	不合法数据	设置的值不正确或者超出范围

以设备号 1 为例，指令如下：

发送：01 04 00 00 00 7D 30 2B

接收：01 84 02 C2 C1

3.4. 驱动器状态寄存器（3 区）

驱动器状态寄存器位于 3 区，地址范围是 0~19，分别存储了当前运行速度、当前位置、状态信息、版本型号等信息，如下图所示：

地址	PLC 地址	长度	功能码	类型	说明
00	30001	2	0x04	float	当前速度
02	30003	2	0x04	int32	当前位置
04	30005	2	0x04	uint32	当前状态信息
06	30007	14	0x04	char	型号版本

3.4.1. 读取当前速度

当前速度存储在三区的 00 和 01 寄存器中，数据类型是 float 型，单位是脉冲频率（每秒发送脉冲个数）。通过功能码 0x04 读取。

3.4.2. 读取当前位置

当前位置存储在三区的 02 和 03 寄存器中，数据类型是 int32 型，单位是脉冲个数。通过功能码 0x04 读取。

3.4.3. 读取当前状态信息

当前状态信息存储在三区的 04 和 05 寄存器中，数据类型是 uint32 型。通过功能码 0x04 读取。每一个 bit 位代表一个状态，如下图所示：

状态位	说明	值
0	传感器 S1 状态	0：低电平；1：高电平
1	传感器 S2 状态	0：低电平；1：高电平
2	传感器 S3 状态	0：低电平；1：高电平
3	保留	—
4	当前位置与目标位置是否相等	0：不相等；1：相等
5	当前速度与目标速度是否相等	0：不相等；1：相等
6	硬件错误（需重新上电）	0：正常；1：发生硬件错误
7	原点标志位	0：不在原点；1：在原点
8	停止标志位	0：运转；1：停止
9	指令正确与否标志位	0：指令正确；1：指令错误
10	参数读写错误标志位	0：正常；1：读写异常
11	保留	—
12	保留	—
13	电机使能标志	0：失能；1：使能
14	归零结束标志	0：归零中；1：归零结束

15	保留	
16	保留	—
17	保留	—
18	保留	
19	保留	
20	过热保护标志位	0：正常；1：过热保护
21	过流保护标志位	0：正常；1：过流保护
22	低压保护标志位	0：正常；1：低压保护

3.5. 驱动器控制和设置寄存器（4 区）

3.5.1. 控制寄存器总览

地址	PLC 地址	长度	功能码	类型	说明
00	40001	1	3, 6, 16		驱动器控制指令
01	40002	2	3, 6, 16	int32	目标位置 (-2147483647~2147483647)
03	40004	2	3, 6, 16	float	目标速度 (-192000~192000)
05	40006	2	3, 6, 16	float	加速度 (0~192000000)
07	40008	2	3, 6, 16	float	减速度 (0~192000000)
09	40010	2	3, 6, 16	float	加速电流 (0 ~ 2.5)
11	40012	2	3, 6, 16	float	运行电流 (0 ~ 2.5)
13	40014	2	3, 6, 16	float	保持电流 (0 ~ 2.5)
15	40016	1	保留		-
16	40017	1	3, 6, 16	uint16	S1 传感器功能
17	40018	1	3, 6, 16	uint16	S2 传感器功能
18	40019	1	3, 6, 16	uint16	S3 传感器功能
19	40020	1	保留		-
20	40021	1	保留		-
21	40022	1	保留		-
22	40023	1	保留		-
23	40024	1	保留		-
24	40025	1	3, 6, 16	uint16	归零功能: 0: 关闭; 1: 开启
25	40026	1	3, 6, 16	uint16	归零传感器类型: 0: 常闭; 1: 常开
26	40027	1	3, 6, 16	uint16	归零用传感器: 0: S1; 1: S2; 2: S3
27	40028	2	3, 6, 16	float	归零速度
29	40030	2	3, 6, 16	int32	安全位置
31	40032	1	3, 6, 16	uint16	站点 ID 号 (1 ~ 32)
32	40033	2	3, 6, 16	uint32	通讯波特率 (2400 ~ 921600)
34	40035	1	3, 6, 16	uint16	微步细分 (0 ~ 5) 0: 整步; 1: 1/2 微步; 2: 1/4 微步; 3: 1/8 微步; 4: 1/16 微步; 5: 1/32 微步
35	40036	1	保留		-
36	40037	1	保留		-
37	40038	1	保留		-
38	40039	1	保留		-
39	40040	1	3, 6, 16	uint16	上电是否归零: 0: 不归零; 1: 归零
40	40041	1	3, 6, 16	uint16	上电是否使能: 0: 不使能; 1: 使能
41	40042	1	3, 6, 16	uint16	数据格式设置 0: 低字节在前, 高字节在后 1: 高字节在前, 低字节在后

3.5.2. 驱动器控制寄存器地址 00

地址 00 寄存器用于控制驱动器的控制功能。驱动器的控制指令包括失能、使能、运行、停止等功能，通过地址 00 寄存器来控制，指令表如下所示：（高 8 位代表指令，第 8 位代表参数）

指令	参数	值	说明
1	0	0x0100	电机失能
	1	0x0101	电机使能
2	0	0x0200	设置原点
3	0	0x0300	速度模式电机连续运行
	1	0x0301	绝对位置模式运行（默认）
	2	0x0302	相对位置模式运行
4	0	0x0400	减速停止
	1	0x0401	立刻停止
5	0	0x0500	参数保存到驱动器
6	0	0x0600	停止归零动作
	1	0x0601	启动归零动作

- ※ 在绝对位置模式下，POS 输入的脉冲数量表示相对于零位的脉冲数
- ※ 在相对位置模式下，POS 输入的脉冲数量表示相对于现在位置的脉冲数

3.5.3. 目标位置设置寄存器地址 0102

地址 01 和 02 寄存器，用于设置位置模式下电机运行的脉冲数。在绝对位置模式下，地址 01，长度为 2 的寄存器表示相对于零位的脉冲数，在相对位置模式下，地址 01，长度为 2 的寄存器表示相对于当前运行的脉冲数，功能码是 0x10，如下所示：

功能	脉冲数	发送字符串	返回字符串
写入运行位置	10000	01 10 00 01 00 02 04 27 10 00 00 39 12	01 10 00 01 00 02 10 08

3.5.4. 速度设置寄存器地址 0304

地址 03 和 04 寄存器用于设置运行速度，可以实时改变速度，单位是脉冲频率，功能码是 0x10，如下所示：

功能	脉冲频率	发送字符串	返回字符串
设置速度	10000	01 10 00 03 00 02 04 40 00 46 1C 95 D3	01 10 00 03 00 02 B1 C8

3.5.5. 加速度设置寄存器地址 0506

地址 05 和 06 寄存器用于设置加速度，范围是 0~192000000，单位是脉冲频率，功能码是 0x10，如下所示：

功能	脉冲频率	发送字符串	返回字符串
设置加速度	100000	01 10 00 05 00 02 04 50 00 47 C3 51 31	01 10 00 05 00 02 51 C9

3.5.6. 减速度设置寄存器地址 0708

地址 07 和 08 寄存器用于设置减速度，范围是 0~192000000，单位是脉冲频率，类型是 float 型，功能码是 0x10，如下所示：

功能	脉冲频率	发送字符串	返回字符串
设置减速度	100000	01 10 00 07 00 02 04 50 00 47 C3 D0 E8	01 10 00 07 00 02 F0 09

3.5.7. 加速电流设置寄存器地址 0910

地址 09 和 10 寄存器用于设置加速电流，范围是 0~2.5，单位是安培，类型是 float 型，功能码是 0x10，如下所示：

功能	数值	发送字符串	返回字符串
设置加速电流	1.5	01 10 00 09 00 02 04 00 00 3F C0 22 65	01 10 00 09 00 02 91 CA

3.5.8. 运行电流设置寄存器地址 1112

地址 11 和 12 寄存器用于设置运行电流，范围是 0~2.5，单位是安培，类型是 float 型，功能码是 0x10，如下所示：

功能	数值	发送字符串	返回字符串
设置运行电流	1.5	01 10 00 0B 00 02 04 00 00 3F C0 A3 BC	01 10 00 0B 00 02 30 0A

3.5.9. 保持电流设置寄存器地址 1314

地址 13 和 14 寄存器用于设置保持电流，保持电流是指电机在使能状态下，不转动时的电流，范围是 0~2.5，单位是安培，类型是 float 型，功能码是 0x10，如下所示：

功能	数值	发送字符串	返回字符串
设置保持电流	1.5	01 10 00 0B 00 02 04 00 00 3F C0 A3 BC	01 10 00 0B 00 02 30 0A

3.5.10. S1 传感器功能设置寄存器地址 16

地址 16 寄存器用于设置 S1 传感器功能，传感器触发包括下降沿和上升沿触发，高 8 位代表下降沿触发功能，低 8 位代表上升沿触发功能，及 s1f : s1r 组成下降沿和上升沿功能，功能码是 0x06。传感器功能如下所示：

功能号	说明
0	无动作（只有状态位变化通知）
1	重新设置原点位置
2	减速停止
3	减速停止，并在停止后重新设置原点位置
4	立刻停止
5	立刻停止，并在停止后重新设置原点位置
6	正向连续运转（正速度）
7	反向连续运转（负速度）

例如，下降沿减速停止，并在停止后重新设置原点位置，上升沿不起作用，则需要设置成：03：00，收发如下：

发送字符串：01 06 00 10 03 00 88 FF

接收字符串：01 06 00 10 03 00 88 FF

3.5.11. S2 传感器功能设置寄存器地址 17

地址 17 寄存器用于设置 S2 传感器功能，传感器触发包括下降沿和上升沿触发，高 8 位代表下降沿触发功能，低 8 位代表上升沿触发功能，及 s2f : s2r 组成下降沿和上升沿功能，功能码是 0x06。传感器功能如下所示：

功能号	说明
0	无动作（只有状态位变化通知）
1	重新设置原点位置
2	减速停止
3	减速停止，并在停止后重新设置原点位置
4	立刻停止
5	立刻停止，并在停止后重新设置原点位置
6	正向连续运转（正速度）
7	反向连续运转（负速度）

例如，下降沿减速停止，并在停止后重新设置原点位置，上升沿不起作用，则需要设置成：03：00，收发如下：

发送字符串：01 06 00 11 03 00 D9 3F
接收字符串：01 06 00 11 03 00 D9 3F

3. 5. 12. S3 传感器功能设置寄存器地址 18

地址 18 寄存器用于设置 S3 传感器功能，传感器触发包括下降沿和上升沿触发，高 8 位代表下降沿触发功能，低 8 位代表上升沿触发功能，及 s3f：s3r 组成下降沿和上升沿功能，功能码是 0x06。传感器功能如下所示：

功能号	说明
0	无动作（只有状态位变化通知）
1	重新设置原点位置
2	减速停止
3	减速停止，并在停止后重新设置原点位置
4	立刻停止
5	立刻停止，并在停止后重新设置原点位置
6	正向连续运转（正速度）
7	反向连续运转（负速度）

例如，下降沿减速停止，并在停止后重新设置原点位置，上升沿不起作用，则需要设置成：03：00，收发如下：

发送字符串：01 06 00 12 03 00 29 3F
接收字符串：01 06 00 12 03 00 29 3F

3. 5. 13. 归零功能设置寄存器地址 24

地址 24 寄存器用于设置归零功能的开启和关闭。0 为关闭，1 为开启。功能码是 0x06，如下所示：

功能	数值	发送字符串	返回字符串
开启	1	01 06 00 18 00 01 C8 0D	01 06 00 18 00 01 C8 0D
关闭	0	01 06 00 18 00 00 09 CD	01 06 00 18 00 00 09 CD

3. 5. 14. 归零用传感器类型设置寄存器地址 25

传感器分常开和常闭型，是指传感器在没有触发时候的状态，分为高电平和低电平。地址 25 寄存器用于设置归零用传感器在没有触发时候的状态，0 为低电平，1 为高电平，功能码是 0x06，如下所示：

功能	数值	发送字符串	返回字符串
低电平	0	01 06 00 19 00 00 58 0D	01 06 00 19 00 00 58 0D
高电平	1	01 06 00 19 00 01 99 CD	01 06 00 19 00 01 99 CD

3.5.15. 归零用传感器设置寄存器地址 26

地址 26 寄存器用于设置归零用传感器，范围 0~2，功能码 0x06，如下所示：

功能	数值	发送字符串	返回字符串
S1 用于归零	0	01 06 00 1A 00 00 A8 0D	01 06 00 1A 00 00 A8 0D
S2 用于归零	1	01 06 00 1A 00 01 69 CD	01 06 00 1A 00 01 69 CD
S3 用于归零	2	01 06 00 1A 00 02 29 CC	01 06 00 1A 00 02 29 CC

3.5.16. 归零速度设置寄存器地址 2728

地址 27 和 28 寄存器用于设置归零速度，单位是脉冲频率（正负号代表方向），类型是 float 型，功能码是 0x10，如下所示：

功能	数值	发送字符串	返回字符串
归零速度	-6400	01 10 00 1B 00 02 04 00 00 C5 C8 E1 D6	01 10 00 1B 00 02 31 CF

3.5.17. 归零后停止位置设置寄存器地址 2930

地址 29 和 30 寄存器用于设置归零后停止的位置，单位是脉冲数，类型是 int32，功能码是 0x16，如下所示：

功能	数值	发送字符串	返回字符串
设置归零后 停止位置	1000	01 10 00 1D 00 02 04 03 E8 00 00 B3 4A	01 10 00 1D 00 02 D1 CE

3.5.18. 站点 ID 号设置寄存器地址 31

地址 31 寄存器用于设置 485 总线站点 ID 号，范围 1~32。功能码是 0x10，如下所示：

功能	数值	发送字符串	返回字符串
ID 为 1	1	01 06 00 1F 00 01 79 CC	01 06 00 1F 00 01 79 CC
ID 为 2	2	01 06 00 1F 00 02 39 CD	01 06 00 1F 00 02 39 CD

※ ID 号更改保存后，需要断电重启生效。

3.5.19. 通讯波特率设置寄存器地址 3233

地址 32 和 33 寄存器用于设置通讯波特率，范围：2400～921600，类型是 uint32，功能码 0x10，如下所示：

功能	数值	发送字符串	返回字符串
波特率	9600	01 10 00 20 00 02 04 25 80 00 00 FB 53	01 10 00 20 00 02 40 02
波特率	115200	01 10 00 20 00 02 04 C2 00 00 01 0D CF	01 10 00 20 00 02 40 02

※ 波特率更改保存后，需要断电重启生效。

3.5.20. 细分设置寄存器地址 34

地址 34 寄存器用于设置微步细分，范围：0～5，功能码是 0x06，如下所示：

功能	数值	发送字符串	返回字符串
整步	0	01 06 00 22 00 00 29 C0	01 06 00 22 00 00 29 C0
1/2 细分	1	01 06 00 22 00 01 E8 00	01 06 00 22 00 01 E8 00
1/4 细分	2	01 06 00 22 00 02 A8 01	01 06 00 22 00 02 A8 01
1/8 细分	3	01 06 00 22 00 03 69 C1	01 06 00 22 00 03 69 C1
1/16 细分	4	01 06 00 22 00 04 28 03	01 06 00 22 00 04 28 03
1/32 细分	5	01 06 00 22 00 05 E9 C3	01 06 00 22 00 05 E9 C3

3.5.21. 上电自动归零设置寄存器地址 39

地址 39 寄存器用于设置上电是否自动归零，范围：0 和 1，0 表示上电后不归零，1 表示上电后执行归零功能，功能码 0x06，如下所示：

功能	数值	发送字符串	返回字符串
上电不归零	0	01 06 00 27 00 00 39 C1	01 06 00 27 00 00 39 C1
上电归零	1	01 06 00 27 00 01 F8 01	01 06 00 27 00 01 F8 01

3.5.22. 上电自动使能设置寄存器地址 40

地址 40 寄存器用于设置上电后是否使能，范围：0 和 1，0 表示不使能，1 表示使能。功能码 0x06，如下所示：

功能	数值	发送字符串	返回字符串
上电不使能	0	01 06 00 28 00 00 09 C2	01 06 00 28 00 00 09 C2

上电使能	1	01 06 00 28 00 01 C8 02	01 06 00 28 00 01 C8 02
------	---	-------------------------	-------------------------

3.5.23. 数据格式设置寄存器地址 41

地址 41 寄存器用于设置 modbus 数据格式。范围是 0 和 1。0 代表低字节在前，高字节在后；1 代表高字节在前，低字节在后。功能码 0x06，如下所示：

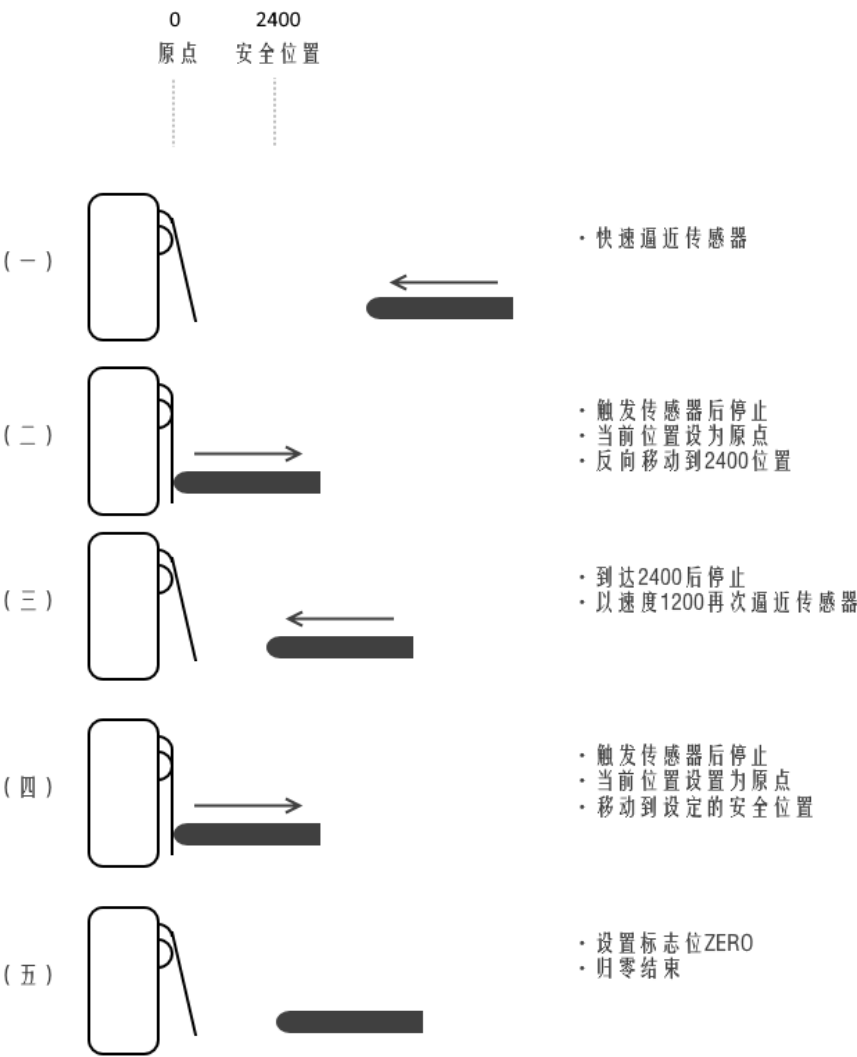
功能	数值	发送字符串	返回字符串
低字节在前 高字节在后	0	01 06 00 29 00 00 58 02	01 06 00 29 00 00 58 02
高字节在前 低字节在后	1	01 06 00 29 00 01 99 C2	01 06 00 29 00 01 99 C2

4. 归零功能设置

4.1.1. 归零功能简介

VSMD 驱动器内置功能，设定好归零参数后，通过指令，驱动器自动完成整个归零过程。
VSMD104/105 系列驱动器采用二次逼近归零方式。归零结束后，会将状态位 ZERO 置位，通过查询状态位 Bit14 可以判断归零动作是否完成。

二次逼近归零过程如下图所示（zmd=1 zsd=-1200 zsp=2400 snr=0 OSV=0）：



- ※ 如果归零开始时，传感器已经处于触发状态，则从（二）开始运行。
- ※ 归零的速度，以及合适的安全位置，要根据实际情况来设置。
- ※ 请注意归零速度以及安全位置的方向（正负）

4.1.2. 归零参数设置

VSMD 驱动器归零功能，需要设置五个参数，如下图所示：

寄存器地址	说明
24	归零模式设置。1：开启；0：关闭
25	归零用传感器选择
26	传感器类型。1：常开；0：常闭
2728	归零速度
2930	归零后停止位置

4.1.3. 归零动作执行

00 地址寄存器中写入 0x0601 启动归零动作；写入 0x0600 停止归零动作。

5. 指示灯

蓝色 LED 指示当前驱动器的工作状态。

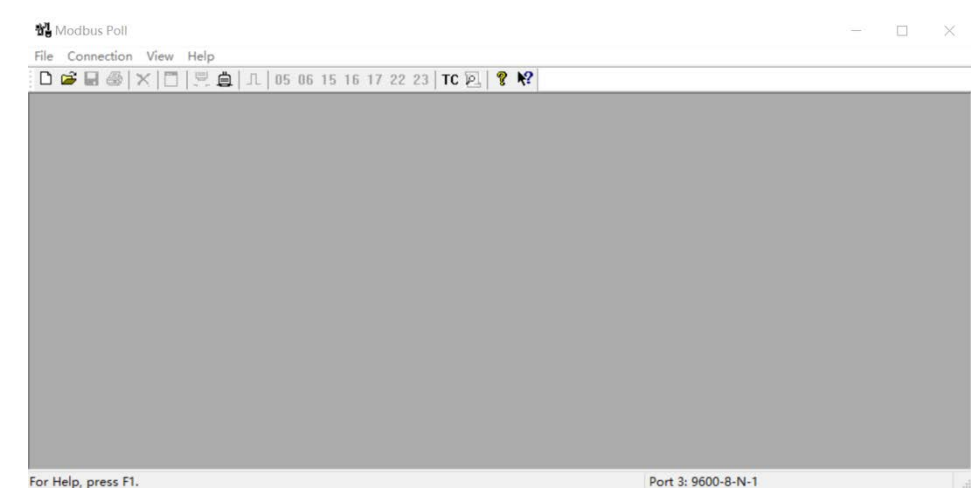
序号	LED 指示	状态
1	长亮/长灭	位置故障
2	慢闪	停止
3	快闪	运行
4	双闪	驱动器硬件故障

6. PC 端控制配置工具

PC 端控制配置，软件采用 modbus poll，映射文件为 vsmd1x4&5_x3.mbp 和 vsmd1x4&5_x4.mbp，分别代表寄存器 3 区和 4 区。

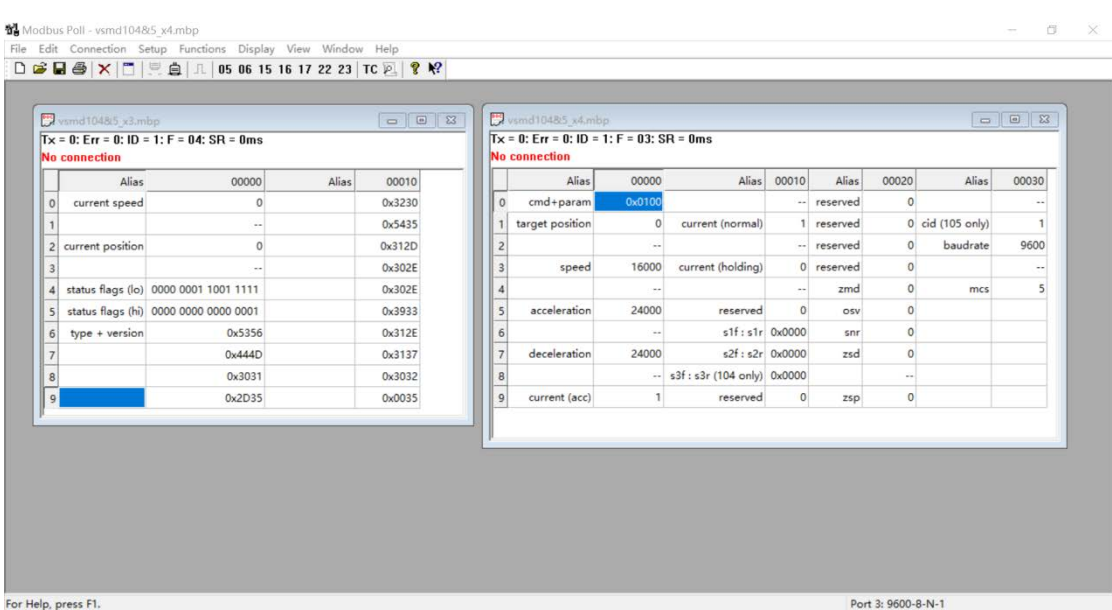
6.1. 主界面

启动 Modbus Poll 软件，会显示主界面，如下图所示：



6.2. 打开映射文件

选择 File -> Open, 选择 vsmd1x4&5_x3.mbp 和 vsmd1x4&5_x4.mbp 文件，分别打开，如下图所示：



6.3. 串口连接

选择 Connection -> Connect 菜单，出现串口设置对话框，如下图所示：



串口设置好后，点击 OK 按钮，这样和驱动器就建立了连接。

6.4. 驱动器状态寄存器

驱动器状态寄存器映射文件是 vsmd1x4&5_x3.mbp，如下图所示：

Tx = 72: Err = 0: ID = 1: F = 04: SR = 0ms				
	Alias	00000	Alias	00010
0	current speed	0		0x3230
1		--		0x5435
2	current position	0		0x312D
3		--		0x302E
4	status flags (lo)	0000 0001 1011 1111		0x312E
5	status flags (hi)	0000 0000 0000 0001		0x3630
6	type + version	0x5356		0x312E
7		0x444D		0x3039
8		0x3031		0x3233
9		0x2D35		0x0035

起始地址	名称	长度	类型	说明
00	current speed	2	float	当前运行速度
02	current position	2	int32	当前位置
04	status flags(lo)	1	uint16	当前状态低 16 位
05	status flags(hi)	1	uint16	当前状态高 16 位
06	type + version	14	char	版本号

6.5. 驱动器控制寄存器

驱动器控制寄存器映射文件是 vsmd1x4&5_x4.mbp，如下图所示：

vsmd1x4&5_x4.mbp

Tx = 264; Err = 0; ID = 1; F = 03; SR = 0ms

	Alias	00000	Alias	00010	Alias	00020	Alias	00030	Alias	00040
0	cmd+param	0x0000		-- s5f : s5r	0x0000		--	pae		0
1	target position	0	current (normal)	0.4 s6f : s6r	0x0000	cid (115 only)	1	mbs		0
2		--		-- reserved	0	baudrate	9600	reserved		0
3	speed	12000	current (holding)	0 reserved	0		--	enc mode		0
4		--		-- zmd	0	mcs	5	enc lines		0
5	acceleration	12000	sensor config bits	0000 0000 0000 0000	osv	0	msr	0	enc steps	0
6		--	s1f : s1r	0x0000	snr	0	msv	0	enc retry	0
7	deceleration	12000	s2f : s2r	0x0000	zsd	0	psr	0	enc dir	0
8		--	s3f : s3r	0x0000		--	psv	0	enc ewr	0
9	current (acc)	0.6	s4f : s4r	0x0000	zsp	0	zar	0	enc ez	0

起始地址	名称	长度	类型	说明
00	cmd + param	1	-	驱动器控制命令寄存器
01	target position	2	int32	设置目标位置
03	speed	2	float	设置运行速度
05	acceleration	2	float	设置加速度
07	deceleration	2	float	设置减速度
09	current (acc)	2	float	设置加速电流
11	current (normal)	2	float	设置运行电流
13	current (holding)	2	float	设置保持电流
16	s1f:s1r	1	uint16	设置 S1 传感器功能
17	s2f:s2r	1	uint16	设置 S2 传感器功能
18	s3f:s3r	1	uint16	设置 S3 传感器功能
24	zmd	1	-	设置归零功能
25	osv	1	-	设置归零传感器常开或者常闭
26	snr	1	-	设置归零用传感器编号
27	zsd	2	float	设置归零速度
29	zsp	2	int32	设置归零后停止位置
31	cid	1	uint16	设置站点 ID 号（485 总线）
32	baudrate	2	int32	设置通讯波特率
34	mcs	1	uint16	设置微步细分

39	zar	1	uint16	设置上电自动归零
40	pae	1	uint16	设置上电自动使能
41	mbs	1	uint16	设置数据格式

7. 性能指标

7.1. 电气性能

工作电压 (DC)	12V~40V
峰值电流	2.5A, 实际电流可调
驱动方式	RS232 -Modbus RTU/RS485-ModbusRTU 命令控制
励磁方式	1/2/4/8/16/32
最大输出脉冲频率	192KHz
绝缘电阻	常温常压下>100M Ω
绝缘强度	常温常压下 0.5KV, 1 分钟

7.2. 使用环境

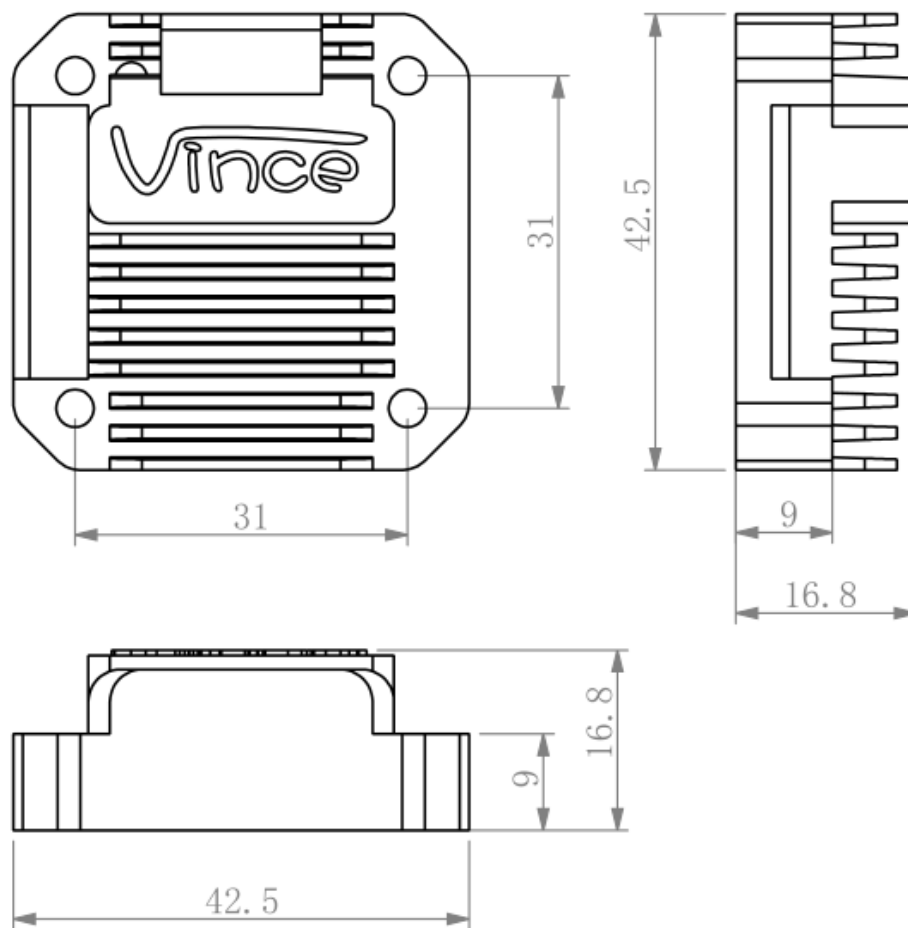
冷却方式	自然冷却
工作温度	-30℃~80℃
工作湿度	≤80%

7.3. 尺寸及重量

外形尺寸	42.5mm × 42.5mm × 16.8mm
重 量	0.07kg

8. 附件

8.1. 外形尺寸图



8.2. CRC 校验程序

```
const uint8_t auch_crc_hi[] = {
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
    0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
    0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
    0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
    0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40
};
```

```
const uint8_t auch_crc_lo[] = {
    0x00, 0xC0, 0xC1, 0x01, 0xC3, 0x03, 0x02, 0xC2, 0xC6, 0x06, 0x07, 0xC7, 0x05, 0xC5, 0xC4, 0x04,
    0xCC, 0x0C, 0x0D, 0xCD, 0x0F, 0xCF, 0xCE, 0x0E, 0x0A, 0xCA, 0xCB, 0x0B, 0xC9, 0x09, 0x08, 0xC8,
    0xD8, 0x18, 0x19, 0xD9, 0x1B, 0xDB, 0xDA, 0x1A, 0x1E, 0xDE, 0xDF, 0x1F, 0xDD, 0x1D, 0x1C, 0xDC,
    0x14, 0xD4, 0xD5, 0x15, 0xD7, 0x17, 0x16, 0xD6, 0xD2, 0x12, 0x13, 0xD3, 0x11, 0xD1, 0xD0, 0x10,
    0xF0, 0x30, 0x31, 0xF1, 0x33, 0xF3, 0xF2, 0x32, 0x36, 0xF6, 0xF7, 0x37, 0xF5, 0x35, 0x34, 0xF4,
    0x3C, 0xFC, 0xFD, 0x3D, 0xFF, 0x3F, 0x3E, 0xFE, 0xFA, 0x3A, 0x3B, 0xFB, 0x39, 0xF9, 0xF8, 0x38,
    0x28, 0xE8, 0xE9, 0x29, 0xEB, 0x2B, 0x2A, 0xEA, 0xEE, 0x2E, 0x2F, 0xEF, 0x2D, 0xED, 0xEC, 0x2C,
    0xE4, 0x24, 0x25, 0xE5, 0x27, 0xE7, 0xE6, 0x26, 0x22, 0xE2, 0xE3, 0x23, 0xE1, 0x21, 0x20, 0xE0,
    0xA0, 0x60, 0x61, 0xA1, 0x63, 0xA3, 0xA2, 0x62, 0x66, 0xA6, 0xA7, 0x67, 0xA5, 0x65, 0x64, 0xA4,
    0x6C, 0xAC, 0xAD, 0x6D, 0xAF, 0x6F, 0x6E, 0xAE, 0xAA, 0x6A, 0x6B, 0xAB, 0x69, 0xA9, 0xA8, 0x68,
    0x78, 0xB8, 0xB9, 0x79, 0xBB, 0x7B, 0x7A, 0xBA, 0xBE, 0x7E, 0x7F, 0xBF, 0x7D, 0xBD, 0xBC, 0x7C,
    0xB4, 0x74, 0x75, 0xB5, 0x77, 0xB7, 0xB6, 0x76, 0x72, 0xB2, 0xB3, 0x73, 0xB1, 0x71, 0x70, 0xB0,
    0x50, 0x90, 0x91, 0x51, 0x93, 0x53, 0x52, 0x92, 0x96, 0x56, 0x57, 0x97, 0x55, 0x95, 0x94, 0x54,
    0x9C, 0x5C, 0x5D, 0x9D, 0x5F, 0x9F, 0x9E, 0x5E, 0x5A, 0x9A, 0x9B, 0x5B, 0x99, 0x59, 0x58, 0x98,
    0x88, 0x48, 0x49, 0x89, 0x4B, 0x8B, 0x8A, 0x4A, 0x4E, 0x8E, 0x8F, 0x4F, 0x8D, 0x4D, 0x4C, 0x8C,
    0x44, 0x84, 0x85, 0x45, 0x87, 0x47, 0x46, 0x86, 0x82, 0x42, 0x43, 0x83, 0x41, 0x81, 0x80, 0x40
};
```

```
uint16_t dev_comm::crc_checksum(uint8_t* data, int size)
{
    uint8_t hi = 0xff;
    uint8_t lo = 0xff;
    uint8_t index;

    for(;size;size--)
    {
        index = hi ^ *data;
        hi = lo ^ auch_crc_hi[index];
        lo = auch_crc_lo[index];
        data++;
    }

    return (uint16_t)(hi << 8 | lo);
}
```