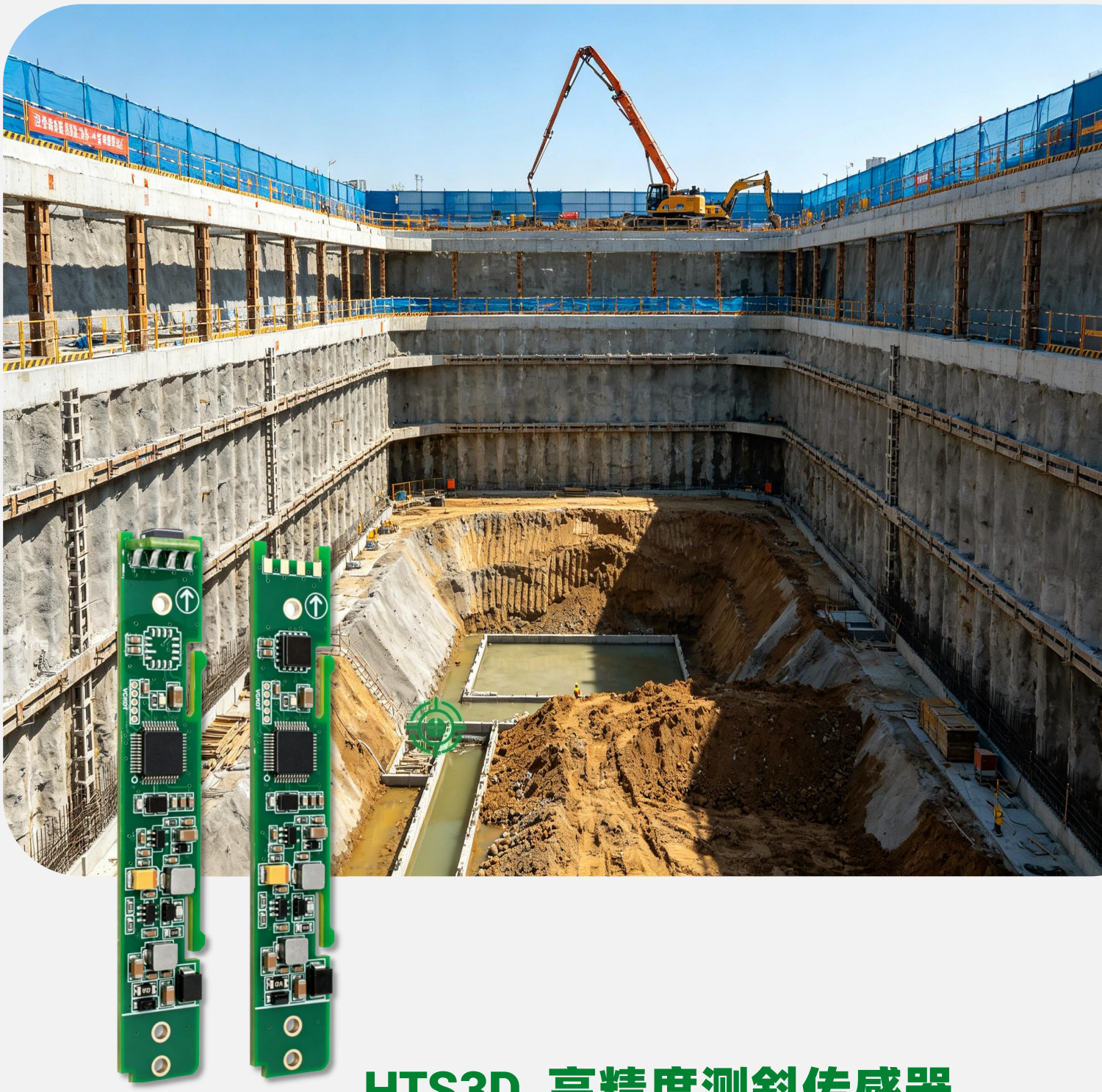
RUICOM

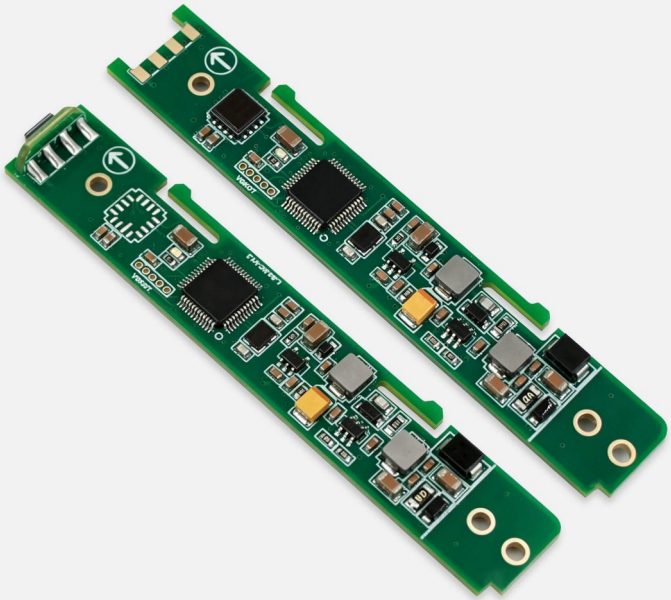


HTS3D 高精度测斜传感器

产品说明书

V1.5-20260519

技术亮点



- 双轴倾角测量；
- 测量范围：±85°；
- 测量精度：0.01°；
- 分辨率 0.001°；
- DC 5~36V 宽电压输入。

产品介绍

HTS3 是瑞惯科技推出的一款双轴向测斜传感器，主要适用于基坑测斜与监测、桥梁检测等工业工程领域。该产品继承了 HTS 系列的稳定性，在精度方面有所提升，温度漂移性能表现更佳，长期稳定性可控制在 0.01 度以内。通过内置的 5 阶滤波算法，结合双轴重力复合运算，能够解算传感器的横滚角和俯仰角度。出厂前，每只产品均经过多面转台对双轴交叉轴误差进行校正，以有效降低大量程下测量轴之间的相互影响。产品配备 RS485 接口，并集成保护电路，以避免对 485 总线造成干扰；电源部分采用多级稳压及过压过流保护设计。该产品符合工业级应用标准，具备稳定的性能和良好的扩展性，可适用于多种恶劣工业控制环境。

应用范围

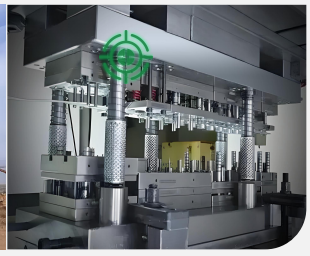
该系列产品典型应用场景包括：基坑测斜检测、铁路机车监测、工业自动化、工程机械平台调平系统，光伏跟踪支架控制系统，航空航天姿态参考系统，精密仪器水平校准装置，平台调平等高精度倾角测量领域。

工程机械设备测控

桥梁大坝安全监测

挖掘机角度测控

高精度大型治具



性能参数

HTS3D	条件	参数	单位
测量范围		±85	°
测量轴		双轴：X、Y	轴
分辨率		0.001	°
测量精度	@25℃	0.01	°
长期稳定性		0.01	°
零点温度系数	-40～85℃	±0.0002	°/℃
灵敏度温度系数	-40～85℃	≤50	ppm/℃
上电启动时间		1	S
响应带宽		5	Hz
输出信号	RS485		
电磁兼容性	依照 EN61000 和 GBT17626		
绝缘电阻	≥100 兆欧		
抗冲击	100g@11ms、三轴向(半正弦波)		
抗振动	10grms、10～1000Hz		
重量	≤10g(不含电缆线)		

1) 分辨率：在有效量程内可识别的最小角度变化量，反映其对微小倾角波动的监测能力。

2) 测量精度：指在标准环境温度下，传感器各项性能指标的综合评定，包括线性度误差、重复性误差、迟滞误差、零点偏移误差以及交叉轴灵敏度误差的总体表现。

3) 长期稳定性：指传感器在常温环境条件下，连续工作一年时间内，输出值所出现的最大值与最小值之间的偏差量。

4) 零点温度系数：传感器在零输入状态下，其输出值随温度变化的比率，定义为额定工作温度范围内零点偏移量与常温基准值的比值。

5) 灵敏度温度系数：传感器满量程输出值随温度变化的稳定性指标，表征额定温度范围内灵敏度相对于常温参考值的漂移率。

电气参数

参数	条件	最小值	典型值	最大值	单位
供电电压	标准	5	12	36	V
	可定制		其他电压		
工作电流	无负载		5		mA
工作温度		-40		+85	℃
存储温度		-40		+85	℃

订购选型

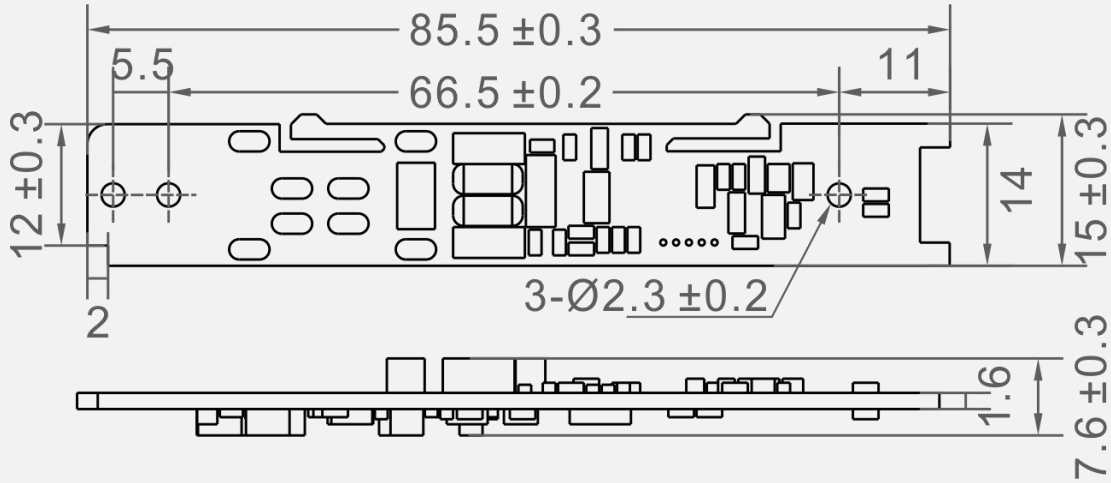
型号		通信协议		安装方向
HTS3D	-	77：77 协议	-	H:水平安装
		MB：MODBUS 协议		V:垂直安装
例：HTS3D-77-H：77 协议/水平安装。				

产品尺寸

垂直安装板子尺寸图

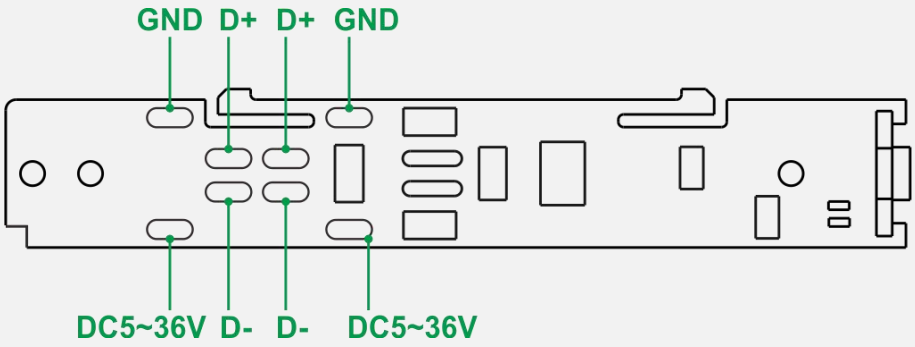
产品尺寸：L86×W15×H9mm

水平安装板子尺寸图



产品尺寸：L85.5×W15×H7.6mm

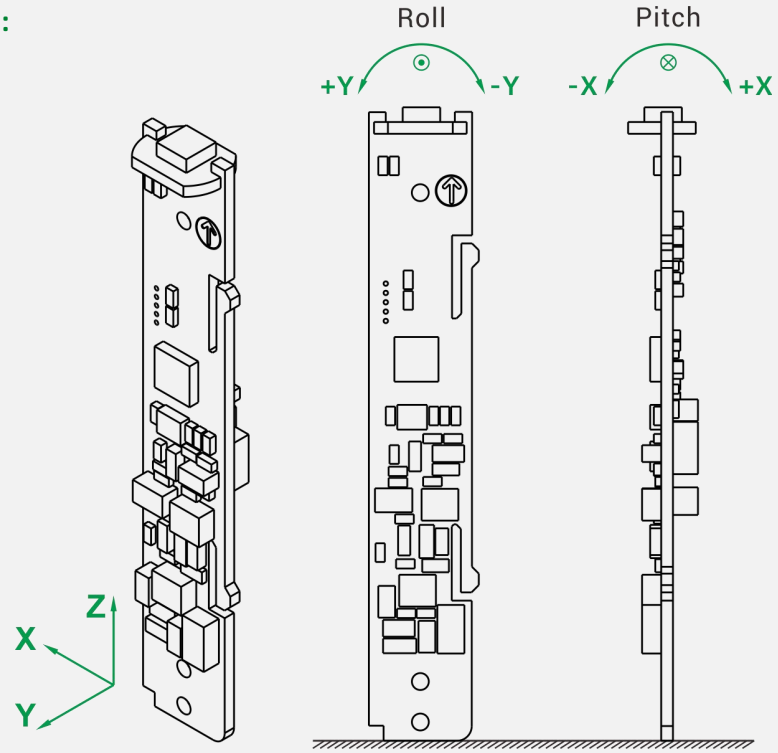
接口定义



测量方向

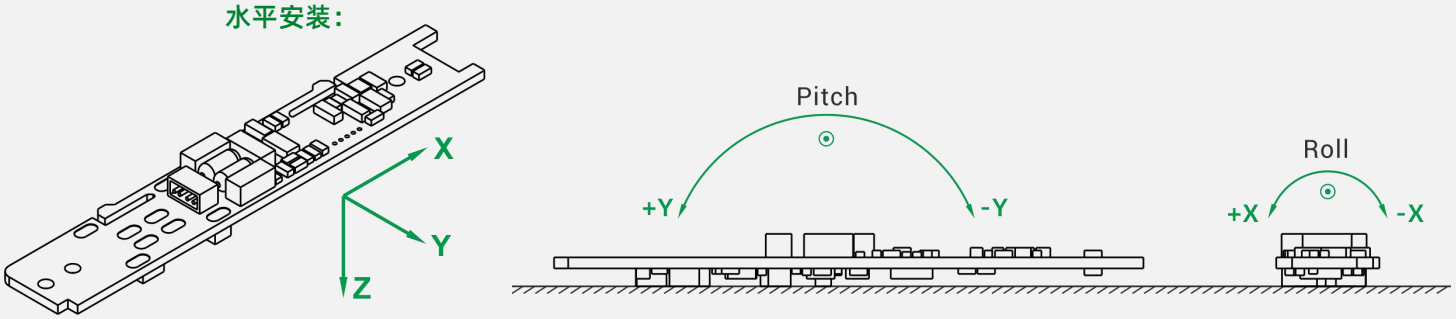
安装时应保持传感器安装面与被测目标面平行，并减少动态和加速度对传感器的影响。安装方式请参考下面示意图：

垂直安装：



*坐标系:右-前-上，对应 X-Y-Z。

水平安装：



*坐标系:前-右-下，对应 X-Y-Z。

串口通信协议

1. 数据帧格式：(默认参数：8 位数据位，1 位停止位，无校验，地址码 0，波特率 2400。)

标示符 (1byte)	数据长度 (1byte)	地址码 (4byte)	命令字 (1byte)	数据域 (n byte)	校验和 (1byte)
0x77	length	address	CMD	numbers	cksum

数据格式：16 进制；

标示符：0x77，帧起始标志；

数据长度：从数据长度域到校验和的字节长度，不包括标示符；

地址码：采集模块的地址，地址 4 个字节，高字节在前，低字节在后(0x00000000 为万能地址)；

数据域：根据命令不同，内容和长度相应变化；

校验和：数据长度，地址码，命令字和数据域的和，只考虑低字节。

2. 命令格式

命令字	功能	注意
0x01	读 X 轴角度	
0x02	读 Y 轴角度	
0x03	查询当前水平横滚角(R 角)	用在水平对安装时读取 R 角。
0x04	读 X, Y，Z 轴角度及温度	Z 轴角度无意义
0x05	设置相对/绝对零点	
0x0A	设置保存	
0x0B	设置波特率	
0x0D	查询相对/绝对零点	
0x0F	设置地址	
0x1F	查询当前地址	

2.1 读 X 轴角度(命令字 0x01)

发送命令：77 07 00 00 00 00 01 08

标示符 (1byte)	数据长度 (1byte)	地址码 (4byte)	命令字 (1byte)	数据域 (0 byte)	校验和 (1byte)
0x77	07	00 00 00 00	01	NULL	cksum

应答格式：

标示符 (1byte)	数据长度 (1byte)	地址码 (4byte)	命令字 (1byte)	数据域 (4 byte)	校验和 (1byte)
0x77	0x0B	00 00 00 00	81	SX XX YY Y0	cksum

注意：数据域为 4 字节， 返回角度为压缩 BCD 码， S 为符号位(0 为正，1 为负)，XXX 为 3 位整数值，YYY 为 3 位小数值。其他轴与此相同，如 10268110 表示-26.811 度。

2.2 读 Y 轴角度(命令字 0x02)

发送命令：77 07 00 00 00 00 02 09

标示符 (1byte)	数据长度 (1byte)	地址码 (4byte)	命令字 (1byte)	数据域 (0 byte)	校验和 (1byte)
0x77	07	00 00 00 00	02	NULL	cksum

应答格式：

标示符 (1byte)	数据长度 (1byte)	地址码 (4byte)	命令字 (1byte)	数据域 (4 byte)	校验和 (1byte)
0x77	0x0B	00 00 00 00	82	SX XX YY Y0	cksum

注意：数据域为 4 字节， 返回角度为压缩 BCD 码， S 为符号位(0 为正，1 为负)，XXX 为 3 位整数值，YYY 为 3 位小数值。其他轴与此相同，如 00268110 表示 26.811 度。

2.3 查询当前水平横滚角(R 角)(命令字 0x03)

发送命令：77 07 00 00 00 00 03 0A

标示符 (1byte)	数据长度 (1byte)	地址码 (4byte)	命令字 (1byte)	数据域 (0 byte)	校验和 (1byte)
0x77	07	00 00 00 00	03	NULL	cksum

应答格式：

标示符 (1byte)	数据长度 (1byte)	地址码 (4byte)	命令字 (1byte)	数据域 (4 byte)	校验和 (1byte)
0x77	0x0B	00 00 00 00	83	SX XX YY Y0	cksum

注：用于水平安装对准时，读取当前的安装横滚角(R 角)。数据域为 4 字节， 返回角度为压缩 BCD 码， S 为符号位(0 为正，1 为负)，XXX 为 3 位整数值，YYY 为 3 位小数值。其他轴与此相同，如 00268110 表示 26.811 度。

2.4 读 X, Y, Z 轴角度及温度(命令字 0x04)

发送命令：77 07 00 00 00 00 04 0B

标示符 (1byte)	数据长度 (1byte)	地址码 (4byte)	命令字 (1byte)	数据域 (0 byte)	校验和 (1byte)
0x77	07	00 00 00 00	04	NULL	cksum

应答格式：

标示符 (1byte)	数据长度 (1byte)	地址码 (4byte)	命令字 (1byte)	数据域 (16 byte)	校验和 (1byte)
0x77	0x17	00 00 00 00	0x84	SX XX xx x0 SY YY yy y0 SZ ZZ zz z0 ST TT tt t0	cksum

注意：数据域为 16 字节， 返回数据为压缩 BCD 码：

SX XX xx x0：为 X 角度值， S 为符号位(0 为正，1 为负)，XXX 为 3 位整数值，xxx0 为 3 位小数值。其他轴与此相同，如 00268110 表示 26.811 度。

SY YY yy y0：为 Y 角度值， S 为符号位(0 为正，1 为负)，YYY 为 3 位整数值，yyy0 为 3 位小数值。其他轴与此相同，如 10255550 表示-25.555 度。

SZ ZZ zz z0：为 Y 角度值， S 为符号位(0 为正，1 为负)，ZZZ 为 3 位整数值，zzz0 为 3 位小数值。其他轴与此相同，如 10255550 表示-25.555 度。

ST TT tt t0：为温度值， S 为符号位(0 为正，1 为负)，TTT 为 3 位整数值，ttt0 为 3 位小数值。其他轴与此相同，如 00233500 表示 23.35 度。

备注：由于倾角内部初始化，在上电之后约 2 秒内读角度指令将会无数据回复。

2.5 设置相对/绝对零点(命令字 0x05)

发送命令：77 08 00 00 00 00 05 01 0E

标示符 (1byte)	数据长度 (1byte)	地址码 (4byte)	命令字 (1byte)	数据域 (1 byte)	校验和 (1byte)
0x77	08	00 00 00 00	05	00: 绝对零点 01: 相对零点	cksum

应答格式：

标示符 (1byte)	数据长度 (1byte)	地址码 (4byte)	命令字 (1byte)	数据域 (1 byte)	校验和 (1byte)
0x77	0x08	00 00 00 00	0x85	00: 设置成功 FF: 设置失败	cksum

注：如果设成绝对零点，则测量角度以出厂设置的零点为基准，如果设成相对零点，则测量角度以当前位置为零点基准。

2.6 设置波特率(命令字 0x0B)

发送命令：77 08 00 00 00 00 0B 00 13

标示符 (1byte)	数据长度 (1byte)	地址码 (4byte)	命令字 (1byte)	数据域 (1 byte)	校验和 (1byte)
0x77	08	00 00 00 00	0B	00: 2400 01: 4800 02: 9600 03: 19200 04: 115200	cksum

注：波特率为 bps;

00: 2400 (默认值) 01: 4800 02: 9600 03: 19200 04: 115200。

应答格式：

标示符 (1byte)	数据长度 (1byte)	地址码 (4byte)	命令字 (1byte)	数据域 (1 byte)	校验和 (1byte)
0x77	0x08	00 00 00 00	0x8B	00: 设置成功 FF: 设置失败	cksum

2.7 设置模块地址(命令字 0x0F)

发送命令：77 0B 00 00 00 00 0F 12 34 56 78 XX

标示符 (1byte)	数据长度 (1byte)	地址码 (4byte)	命令字 (1byte)	数据域 (4 byte)	校验和 (1byte)
0x77	0x0B	00 00 00 00	0F	12 34 56 78	cksum

应答格式：

标示符 (1byte)	数据长度 (1byte)	地址码 (4byte)	命令字 (1byte)	数据域 (1 byte)	校验和 (1byte)
0x77	0x08	12 34 56 78	0x8F	00: 设置成功 FF: 设置失败	cksum

注：地址不能设为 0x00000000, 0x00000000 为万能地址。

2.8 查询相对/绝对零点(命令字 0x0D)

发送命令：77 07 00 00 00 00 0D 14

标示符 (1byte)	数据长度 (1byte)	地址码 (4byte)	命令字 (1byte)	数据域 (0 byte)	校验和 (1byte)
0x77	07	00 00 00 00	0D	NULL	cksum

应答格式：

标示符 (1byte)	数据长度 (1byte)	地址码 (4byte)	命令字 (1byte)	数据域 (1 byte)	校验和 (1byte)
0x77	0x08	00 00 00 00	0x8D	00: 绝对零点 01: 相对零点	cksum

2.9 设置保存(命令字 0x0A)

发送命令：77 07 00 00 00 00 0A 11

标示符 (1byte)	数据长度 (1byte)	地址码 (4byte)	命令字 (1byte)	数据域 (0 byte)	校验和 (1byte)
0x77	07	00 00 00 00	0A	NULL	cksum

应答格式：

标示符 (1byte)	数据长度 (1byte)	地址码 (4byte)	命令字 (1byte)	数据域 (1 byte)	校验和 (1byte)
0x77	0x08	00 00 00 00	0x8A	00: 设置成功 FF: 设置失败	cksum

注：此命令用于保存当前的设置参数，执行此命令后，才有掉电保存功能，否则更改的参数不具有掉电保存功能，波特率在重新掉电后方能生效。

3.0 查询当前地址(命令字 0x1F)(地址码出厂默认为 0)

发送命令：77 07 00 00 00 00 1F 26

标示符 (1byte)	数据长度 (1byte)	地址码 (4byte)	命令字 (1byte)	数据域 (0 byte)	校验和 (1byte)
0x77	07	00 00 00 00	1F	NULL	cksum

应答格式：

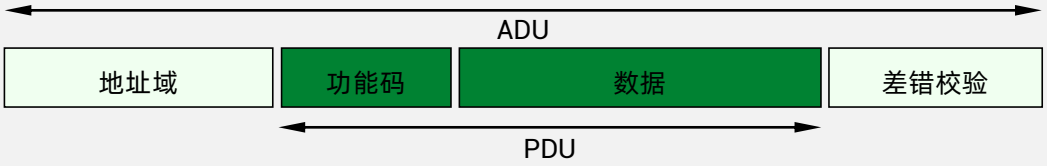
标示符 (1byte)	数据长度 (1byte)	地址码 (4byte)	命令字 (1byte)	数据域 (4 byte)	校验和 (1byte)
0x77	0x0B	12 34 56 78	0x9F	12 34 56 78	cksum

MODBUS 通信协议

通信格式：1 个起始位，8 个数据位，无校验位，1 个停止位；默认参数：地址码 1，波特率 9600。

MODBUS 协议规定两条数据帧之间应至少大于 3.5 个字节时间，本传感器将此时间延长到 15mS。

帧格式：



1. 传输格式

(1)命令报文格式

发送读数据命令：

地址	功能码	数据起始地址高位	数据起始地址低位	数据个数高位	数据个数低位	CRC16位校验
高位在前						

发送读数据命令返回：

地址	功能码	字节长度	数据 1 输入	数据 2 输入	...	CRC16 位校验
高位在前	高位在前					

发送写数据命令：

地址	功能码	数据起始地址高位	数据起始地址低位	数据高位	数据低位	位校验CRC16
高位在前						

发送写数据命令返回—命令原样返回：

地址	功能码	数据起始地址高位	数据起始地址低位	数据高位	数据低位	位校验CRC16
高位在前						

(2)异常应答返回

非法功能:

从站地址	功能码	异常码	CRC16 校验
	80H+原功能码	01	

非法数据地址:

从站地址	功能码	异常码	CRC16 校验
	80H+原功能码	02	

非法数据值:

从站地址	功能码	异常码	CRC16 校验
	80H+原功能码	03	

请求数据值为空(未有值):

从站地址	功能码	异常码	CRC16 校验
	80H+原功能码	0F	

2. 模块 RTU 指令及说明信息

功能代码	数据起始地址	数据字个数	数据字节数	指令说明
04H	0016H	8	16	读当前测量值 XYZ 和温度值--浮点数据
04H	001EH	2	4	读水平时的横滚角-浮点数据
06H	0004H	1	2	0x00A0: 2400 0x00A1: 4800 0x00A2: 9600 0x00A3: 19200 0x00A4: 38400 0x00A5: 115200 (更改波特率重新上电有效)。
06H	0006H	1	2	写入 0x00FF,清除 GAMMA 角(γ)。写入 0x0001，产生一个 GAMMA 角(γ)。产生的 GAMMA 角存在 FLASH 中。
03H	0008H	2	4	读取 GAMMA(γ)角(此值存在 FLASH 中)，若没有产生 GAMMA 角(γ)，则回复数据空异常。
03H	0030H	1	2	读地址码—整型数据
06H	0030H	1	2	修改地址码—整型数据
10H	0002H	2	4	ms MBRTU 模式密码认证

2.1 读当前测量值和温度值指令

微机发送的命令帧:

Addr	04h	0h	16h	0h	08h	CRC16H	CRC16L

模块返回的数据帧:

Addr	04h	Length=16	Data1 ~ Data16	CRC16H	CRC16L

指令返回数据为 16 字节的浮点数；高字节在前，低字节在后。

Length： 16 返回数据长度。

Data1~Data16：返回数据。

注：如果发送指令时， Addr 使用的是 0xFA 通用地址，返回时 Addr 为模块当前地址。 所用指令处理相同。

示例:

a.读 X Y 方位角和温度值 功能码 04 寄存器地址 16 字节数 16

Send: 01 04 00 16 00 08 10 08
Recv: 01 04 10 C0 8F 12 6F 3E FA E1 48 43 16 19 9A 42 04 00 00 F9 31
ID<1> X: -4.471°
ID<1> Y: +0.490°

ID<1> 方位角: 150.10°

ID<1> 温度: 33℃

注意:由于传感器的初始化,上电之后约 2 秒内读取当前测量值和温度指令回复的是:Addr 84 0F CR16H CR16L。

2.2 地址码指令

a. 读地址指令(**地址码出厂默认为** 1)

微机发送的命令帧:

Addr	03h	00h	30h	00h	01h	CRC16H	CRC16L

模块返回的数据帧:

Addr	03h	Length=2	Data1 ~ Data2	CRC16H	CRC16L

指令返回数据为 2 字节整型数据,模块地址编码 1~255;0xfa 是万能地址码。

可用作读地址码指令的前始地址,读取模块读取地址码。 发送时 Addr 使用 0xFA,

返回时, Addr 和 Data1~Data2 都是模块当前地址码。

b.广播读取 ID 功能码 03 寄存器地址 30 字节数 1 万能码 fa。

Send: FA 03 00 30 00 01 91 8E
Recv: 01 03 02 00 01 79 84

c.修改地址码指令

微机发送的命令帧:

Addr	06h	0h	30h	0h	New Addr	CRC16H	CRC16L

模块返回的数据帧:

New Addr	06h	0h	30h	0h	New Addr	CRC16H	CRC16L

New Addr:模块地址编码 1~255。 发送指令时,不管 Addr 使用的是通用地

址 0xfa,还是实际地址码;返回时,都是当前地址码。

操作前需密码认证,操作成功返回原指令, 否则出错码。

示例:

设置 ID 功能码 06 寄存器地址 30 字节数 2

Send: 01 06 00 30 00 02 08 04
Recv: 02 06 00 30 00 02 08 37

2.3 MBRTU 模式密码认证指令

微机发送的命令帧:

Addr	10h	0h	02h	0h	2h	04h	Data1~Data4	CRC16H	CRC16L

模块返回的数据帧:

Addr	10h	0h	02h	0h	02h	CRC16H	CRC16L

Data1~Data4: 50 53 57 44。

设置成功返回以上指令,否则出错码。

注: MBRTU 模式,所有写指令和调校指令都需密码认证。

示例:

功能:密码认证

发送:01 10 00 02 00 02 04 50 53 57 44 AD 64

接收:01 10 00 02 00 02 E0 08

密码认证- 成功!

功能：修改地址 地址 1 修改 2

发送：01 06 00 30 00 02 08 04

接收: 02 06 00 30 00 02 08 37

地址已修改为:【 2 】

2.4 生成 GAMMA(γ)角

条件：将设备水平放置，将所有管子摆直成一条直线，尽量拉直，保持管子不动，先执行清除 GAMMA(γ)角命令， 然后再执行产生 GAMMA(γ)角命令，成功后的 GAMMA(γ)角将保存在 FLASH 中，此后可以通过读 GAMMA(γ)角命令进行读取使用。

注意，GAMMA(γ)角的产生管子必须水平放置，使所有管子拉直。清除和产生 GAMMA(γ)角指令之前都需进行密码认证。

a. 清除 GAMMA(γ)角

微机发送的命令帧：

Addr	06h	00h	06h	00h	FFh	CRC16H	CRC16L

模块返回的数据帧：

Addr	06h	00h	06h	00h	FFh	CRC16H	CRC16L

b. 产生 GAMMA(γ)角

微机发送的命令帧：

Addr	06h	00h	06h	00h	01h	CRC16H	CRC16L

模块返回的数据帧：

Addr	06h	00h	06h	00h	01h	CRC16H	CRC16L

c. 读取 GAMMA(γ)角

微机发送的命令帧：

Addr	03h	0h	08h	0h	02h	CRC16H	CRC16L

模块返回的数据帧：

Addr	03h	Length=4	Data1 ~ Data4	CRC16H	CRC16L

指令返回数据为 4 字节的浮点数；高字节在前，低字节在后。

若 GAMMA(γ)角为空则返回异常响应：

Addr	83h	0Fh	CRC16H	CRC16L

注意：如果没有产生 GAMMA 角，则读取 GAMMA 角时回复： Addr 83 0F CRC16H CRC16L。

● CRC 码的计算方法是：

1) 预置 1 个 16 位的寄存器为十六进制 FFFF(即全为 1)；称此寄存器为 CRC 寄存器；

2) 把第一个 8 位二进制数据(既通讯信息帧的第一个字节)与 16 位的 CRC 寄存器的低 8 位相异或把结果放于 CRC 寄存器；

3) 把 CRC 寄存器的内容右移一位(朝低位)用 0 填补最高位，并检查右移后的移出位；

4) 如果移出位为 0：重复第 3 步(再次右移一位)；

如果移出位为 1：CRC 寄存器与多项式 A001(1010 0000 0000 0001)进行异或；

5) 重复步骤 3 和 4，直到右移 8 次，这样整个 8 位数据全部进行了处理；

6) 重复步骤 2 到步骤 5，进行通讯信息帧下一个字节的处理；

7) 将该通讯信息帧所有字节按上述步骤计算完成后，得到的 16 位 CRC 寄存器的高、低字节进行交换；

8) 最后得到的 CRC 寄存器内容即为：CRC 码。

```
unsigned char const auchCRCHi[] = {
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,
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
} ;

unsigned char const auchCRCLo[] = {
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
};

unsigned short MB_Make_CRC16( unsigned char *puchMsg, unsigned short usDataLen )
{
    unsigned char uchCRCHi = 0xFF;
    unsigned char uchCRCLo = 0xFF;
    unsigned uIndex;

    while (usDataLen--)
    {
        uIndex = uchCRCLo^*puchMsg++ ;
        uchCRCLo = uchCRCHi^auchCRCHi[uIndex];
        uchCRCHi = auchCRCLo[uIndex];
    }

    return (uchCRCLo<<8|uchCRCHi);
}
```