

CH110 用户手册

IMU/VRU姿态测量模块, Rev 1.0



CH110 用户手册

简介

特性

板载传感器

数据处理

通讯接口及供电

其他

硬件及尺寸

硬件参数

尺寸

接口定义

性能指标

姿态角输出精度

陀螺仪

加速度计

模块数据接口参数

融合及校准算法

陀螺仪校准

参考系定义

串口通讯协议

帧格式

数据包

数据包总览

产品支持数据包列表

0x90(用户ID)

0xA0(加速度)

0xB0(角速度)

0xC0(磁场强度)

0xD0(欧拉角)

0xD1(四元数)

0xF0(气压)

0x91(IMUSOL)

0x62(GWSOL)

出厂默认数据包

数据帧结构示例

数据帧配置为 0x90, 0xA0, 0xB0, 0xC0, 0xD0, 0xF0 数据包

数据帧配置为 0x91 数据包

CAN通讯协议

CAN默认设置

PTO传输细节

修改CAN接口配置

AT指令

产品支持数据包列表

AT+ID

AT+URFR

AT+INFO

AT+ODR

AT+BAUD

AT+EOUT

AT+RST

AT+TRG

AT+SETPEL

AT+MODE

AT+GYRCTL

AT+GWID

AT+GWCFG

附录C - 固件升级与恢复出厂设置

简介

CH110是超核电子推出的一款超低成本、高性能、小体积、低延时的惯性测量单元(IMU)，本产品集成了三轴加速度计、三轴陀螺仪和一款微控制器。可输出经过传感器融合算法计算得到的基于当地地理坐标的三维方位数据，包含无绝对参考的相对航向角，俯仰角和横滚角。同时也可以输出校准过的原始的传感器数据。

典型应用:

- 机器人航向跟踪/无人驾驶等

特性

板载传感器

- 三轴陀螺仪, 最大量程: $\pm 2000^{\circ}/s$
- 三轴加速度计, 最大量程: $\pm 8G$

数据处理

- 加速度和陀螺仪出厂前经过三轴非正交和标度因子校准
- 数据融合算法计算并输出地理坐标系下的旋转四元数及欧拉角等姿态信息

通讯接口及供电

- RS232串行接口/CAN2.0总线
- 供电电压：5-24V

其他

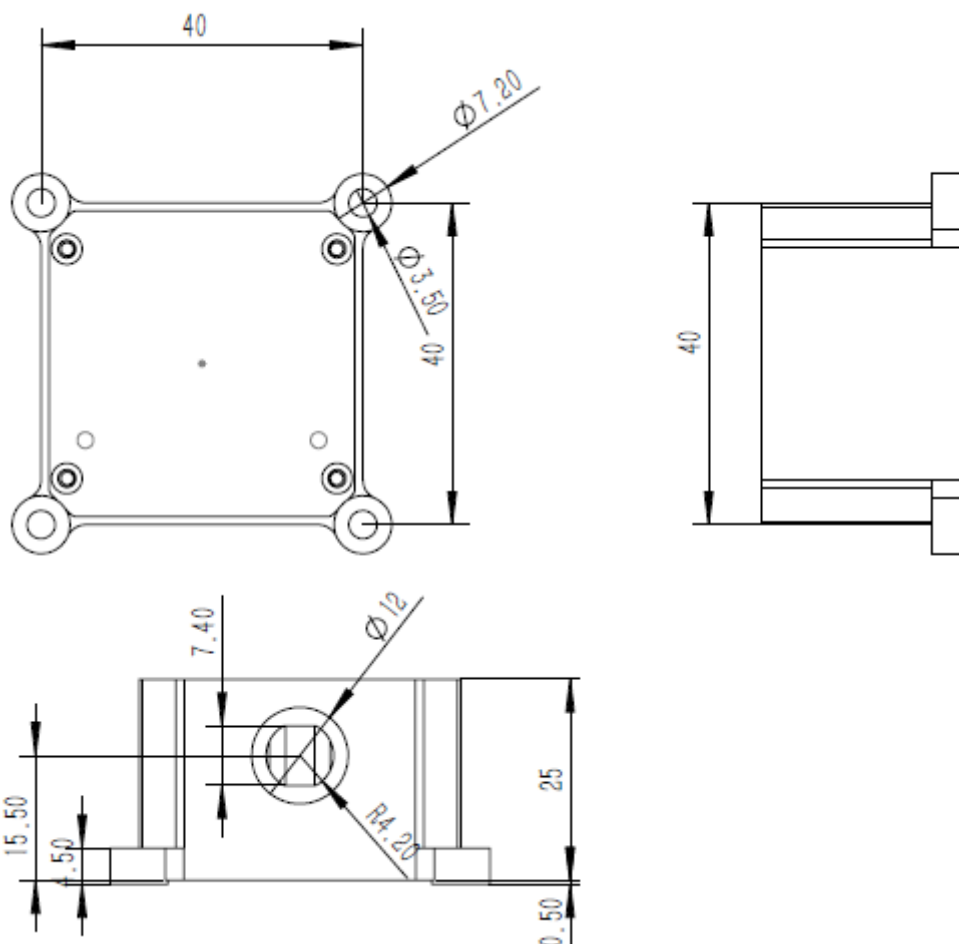
- PC端上位机程序，提供实时数据显示，波形，校准及excel 数据记录功能
- 多项模块参数用户可配置

硬件及尺寸

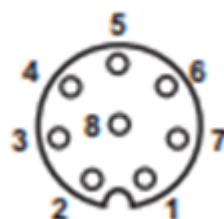
硬件参数

参数	描述
输出数据接口	RS232串行接口
工作电压	5-24V
温度范围	-20℃ - 85℃
最大线性加速度	0 - 115 m/s^2
尺寸	40 x 40 x 25mm (W x L x H)
板载传感器	三轴加速度计 三轴陀螺仪

尺寸



接口定义



引脚号	序号	功能(RS232+CA)	功能(RS485+CAN)
红	1	Vin	Vin
黑	2	GND	GND
蓝	3	RS232 TX	485 A
灰	4	RS232 RX	485 B
白	5	同步输出	同步输出
棕	6	同步输入	同步输入
绿	7	CAN_H	CAN_H
黄	8	CAN_L	CAN_L

性能指标

姿态角输出精度

姿态角	典型值
横滚角\俯仰角 - 静态误差	0.8°
横滚角\俯仰角 - 动态误差	2.5°
零偏稳定性	10%/h

陀螺仪

参数	值
测量范围	±2000°/s
非线性度	±0.1% (25°最佳)
噪声密度	0.08°/s/ \sqrt{Hz}
采样率	500Hz

加速度计

参数	值
测量范围	±8G (1G = 1x 重力加速度)
非线性度	±0.5% (25°最佳)
最大零点偏移	10mG
噪声密度	250 $\mu G\sqrt{Hz}$
采样率	500Hz

模块数据接口参数

参数	值
串口输出波特率	9600/115200可选
帧输出速率	10/50/100Hz 可选

融合及校准算法

陀螺仪校准

每一个姿态传感器都单独进行过全测量范围内的校准和测试。陀螺和加速度计的非正交和刻度因子误差参数都会保存在模块内部的Flash中。陀螺仪自动校准需要在上电后静止模块3s 左右以获得最好的校准效果。如果上电静置短于规定时间，则模块陀螺仪零偏校准效果会下降。

姿态传感器内建陀螺零速检测机制，当检测到长时间内三轴陀螺速度均小于1°/s时，模块认为当前为静止状态，陀螺输出为零偏，此次模块会将此时的陀螺读数记录下来作为零偏补偿。所以 **本产品不能用于旋转速度<1°/s的运动场景**。(既旋转速度低于秒针平均转速的1/6)

参考系定义

本产品采用右手(RH, Right-Hand)坐标系。输出的四元数及欧拉角为 传感器坐标系 到 惯性坐标系(世界坐标系) 的旋转。其中欧拉角旋转顺序为 ZYX(先转Z轴，再转Y轴，最后转X轴)旋转顺序，欧拉角具体定义如下：

- 绕 Z 轴方向旋转: 航向角\Yaw\phi(ψ) 范围: -180° - 180°
- 绕 Y 轴方向旋转: 俯仰角\Pitch\theta(θ) 范围: -90°-90°
- 绕 X 轴方向旋转:横滚角\Roll\psi(ϕ)范围: -180°-180°

本产品使用北西天(North-West-Up NWU) 坐标系统，即视为模块的地理坐标系(世界坐标系)定义如下：

- X 轴正方向指向北

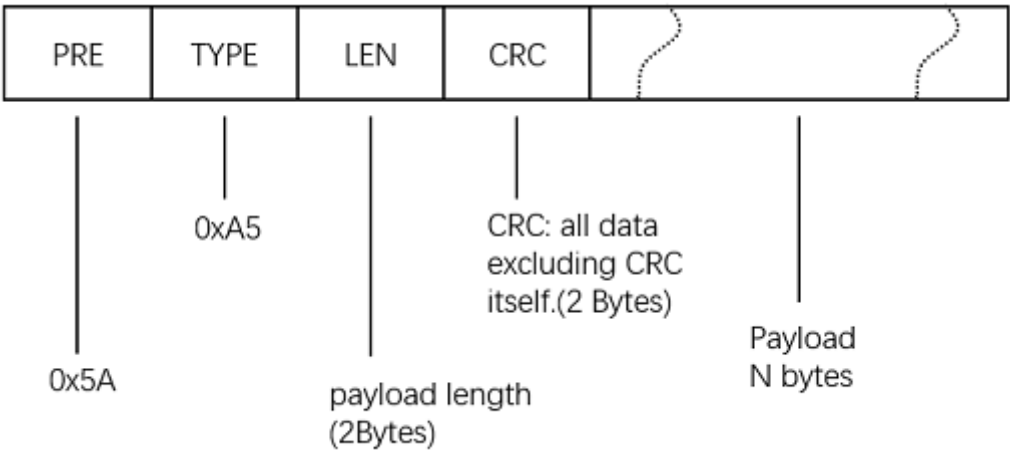
- Y 轴正方向指向西
- Z 轴正方向指向天

当采用 NWU 系时，如果将模块视为飞行器的话。X 轴应视为机头方向。当传感器系与惯性系重合时，欧拉角的理想输出为:Pitch = 0°, Roll = 0°, Yaw = 0°

串口通讯协议

帧格式

模块上电后，模块默认按100Hz(出厂默认输出速率) 输出帧数据，帧格式如下：



其中：

域	值	长度 (字节)	说明
PRE	0x5A	1	固定为0x5A
TYPE	0xA5	1	固定为0xA5
LEN	1-512	2	帧中数据域的长度。LSB(低字节在前)，长度表示数据域的长度，不包含 PRE,TYPE,LEN,CRC 字段。
CRC	-	2	除CRC 本身外其余所有帧数据的16 位CRC 校验和。LSB(低字节在前)
PAYLOAD	-	1-512	一帧携带的数据。PAYLOAD 由若干个 子数据包 组成。每个数据包包含：数据包标签(DATA_ID)和数据(DATA) 两部分。DATA_ID决定了数据的类型及长度，DATA 为数据包内容。

CRC实现函数：

```
1  /*
2      currentCrc: previous crc value, set 0 if it's first section
3      src: source stream data
4      lengthInBytes: length
5  */
6  static void crc16_update(uint16_t *currentCrc, const uint8_t *src, uint32_t
lengthInBytes)
7  {
8      uint32_t crc = *currentCrc;
9      uint32_t j;
10     for (j=0; j < lengthInBytes; ++j)
11     {
12         uint32_t i;
13         uint32_t byte = src[j];
```

```
14         crc ^= byte << 8;
15         for (i = 0; i < 8; ++i)
16         {
17             uint32_t temp = crc << 1;
18             if (crc & 0x8000)
19             {
20                 temp ^= 0x1021;
21             }
22             crc = temp;
23         }
24     }
25     *currentCrc = crc;
26 }
```

数据包

数据包总览

数据包标签(DATA_ID)	数据包长度(包含标签1字节)	名称	备注
0x90	2	用户ID	
0xA0	7	加速度	
0xB0	7	角速度	
0xC0	7	磁场强度	
0xD0	7	欧拉角	
0xD1	17	四元数	
0xF0	5	气压	输出0
0x91	76	IMUSOL(IMU数据集合)	推荐使用
0x62	变长	GWSOL(无线节点数据集合)	

产品支持数据包列表

下表列出所有产品支持的数据包,* 表示支持 -表示不支持

产品	90	A0	B0	C0	D0	D1	F0	91	62
HI226	*	*	*	*	*	*	-	*	-
HI229	*	*	*	*	*	*	-	*	-
CH110	-	-	-	-	-	-	-	*	-
HI221	*	*	*	*	*	*	-	*	-
HI221GW	-	-	-	-	-	-	-	-	*

0x90(用户ID)

共2字节，用户设置的ID。

字节偏移	类型	大小	单位	说明
0	uint8_t	1	-	数据包标签:0x90
1	uint8_t	1	-	用户ID

0xA0(加速度)

共7个字节，LSB。输出传感器的原始加速度

字节偏移	类型	大小	单位	说明
0	uint8_t	1	-	数据包标签:0xA0
1	int16_t	2	0.001G(1G = 1重力加速度)	X轴加速度
3	int16_t	2	0.001G	Y轴加速度
5	int16_t	2	0.001G	Z轴加速度

0xB0(角速度)

共7字节，LSB。输出传感器的原始角速度

字节偏移	类型	大小	单位	说明
0	uint8_t	1	-	数据包标签：0xB0
1	int16_t	2	0.1°/s	X轴角速度
3	int16_t	2	0.1°/s	Y轴角速度
5	int16_t	2	0.1°/s	Z轴角速度

0xC0(磁场强度)

共7字节，LSB。输出传感器的原始磁场强度

字节偏移	类型	大小	单位	说明
0	uint8_t	1	-	数据包标签:0xC0
1	int16_t	2	0.001Gauss	X轴磁场强度
3	int16_t	2	0.001Gauss	Y轴磁场强度
5	int16_t	2	0.001Gauss	Z轴磁场强度

0xD0(欧拉角)

共7字节，LSB。格式为int16，共三个轴，每个轴占2个字节，顺序为Pitch/Roll/Yaw。接收到Roll, Pitch为物理值乘以100后得到的数值，Yaw为乘以10得到的数值。

例：当接收到的Yaw = 100时，表示航向角为10°

字节偏移	类型	大小	单位	说明
0	uint8_t	1	-	数据包标签:0xD0
1	int16_t	2	0.01°	Pitch(俯仰角)
3	int16_t	2	0.01°	Roll(横滚角)
5	int16_t	2	0.1°	Yaw(航向角)

0xD1(四元数)

共17字节，格式为float，共4个值，顺序为:W X Y Z。每个值占4字节(float)，整个四元数为4个float, LSB。

字节偏移	类型	大小	单位	说明
0	uint8_t	1	-	数据包标签:0xD1
1	float	4	-	W
5	float	4	-	X
9	float	4	-	Y
13	float	4	-	Z

0XF0(气压)

共5字节，格式为float。(只针对有气压传感器的产品)

字节偏移	类型	大小	单位	说明
0	uint8_t	1	-	数据包标签:0xF0
1	float	4	Pa	大气压

0X91(IMUSOL)

共76字节，新加入的数据包，用于替代A0,B0,C0,D0,D1等数据包。集成了IMU的传感器原始输出和姿态解算数据。

字节偏移	类型	大小	单位	说明
0	uint8_t	1	-	数据包标签:0x91
1	uint8_t	1	-	ID
2	-	6	-	保留
8	uint32_t	4	ms	时间戳信息，从系统开机开始累加，每毫秒增加1
12	float	12	1G(1G = 1重力加速度)	X,Y,Z轴的加速度，注意单位和0xA0不同
24	float	12	deg/s	X,Y,Z轴的角速度，注意单位和0xB0不同
36	float	12	uT	X,Y,Z轴的磁场强度(HI229支持,注意单位和0xC0不同)
48	float	12	deg	节点欧拉角集合, 顺序为：横滚角(Roll)，俯仰角(Pitch)，航向角(Yaw)(注意顺序和单位与0xD0数据包不同)
60	float	16	-	节点四元数集合,顺序为WXYZ

0x62(GWSOL)

新版本无线接收机支持此数据包。数据包前8个字节为接收机信息。后面分为N个数据块。每个数据块描述一个节点的姿态数据(最大支持16个节点)。每个数据块大小为76字节，数据结构同0x91。

此协议包数据量较大，建议将波特率调整至921600以获得最高的帧率输出。格式如下：

字节偏移	大小	类型	单位	说明
0	1	uint8_t	-	数据包标签:0x62
1	1	uint8_t	-	GWID, 接收机网络ID
2	1	uint8_t	-	N, 此帧包含节点数据块个数
3	5	-	-	保留
---- 节点数据块开始----	-	-	-	数据结构同0x91
8+76*N(N=0-15)	1	uint8_t	-	数据包标签:0x91
9+76*N(N=0-15)	1	uint8_t	-	节点N的ID
10+76*N	10	-	-	保留
20+76*N	12	float	-	节点N三轴加速度
32+76*N	12	float	-	节点N三轴角速度
44+76*N	12	float	-	节点N轴磁场强度
56+76*N	12	float	-	节点N欧拉角
68+76*N	16	float	-	节点N四元数
---- 节点数据块结束----	-	-	-	-----

出厂默认数据包

出厂默认一帧中携带数据包数据定义如下：

产品	默认输出数据包
HI226	90,A0,B0,C0,D0,F0
HI229	90,A0,B0,C0,D0,F0
CH110	90,A0,B0,C0,D0,F0
HI221	91
HI221GW	62

数据帧结构示例

数据帧配置为 0x90, 0xA0, 0xB0, 0xC0, 0xD0, 0xF0 数据包

使用串口助手采样一帧数据,共41字节,前6字节为帧头,长度和CRC校验值。剩余35字节为数据域。假设数据接收到C语言数组buf中。如下所示:

5A A5 23 00 FD 61 **90** 00 **A0** 55 02 3D 01 E2 02 **B0** FE FF 17 00 44 00 **C0** 80 FF 60 FF 32 FF **D0** 64 F2 6C 0E BB 01 **F0** 00 00 00 00

- 第一步：判断帧头，得到数据域长度和帧CRC：

帧头:5A A5

帧数据域长度:23 00: $(0x00 < 8) + 0x23 = 35$

帧CRC校验值:FD 61: $(0x61 < 8) + 0xFD = 0x61FD$

- 第二步：校验CRC

```

1      uint16_t payload_len;
2      uint16_t crc;
3
4      crc = 0;
5      payload_len = buf[2] + (buf[3] << 8);
6
7      /* calculate 5A A5 and LEN filed crc */
8      crc16_update(&crc, buf, 4);
9
10     /* calculate payload crc */
11     crc16_update(&crc, buf + 6, payload_len);

```

得到CRC值为0x61FD, 与帧携带的CRC值相同, 帧CRC校验通过。

- 第三步：接收数据

90 00：ID 数据包, 0x90为数据包标签, ID = 0x00.

A0 55 02 3D 01 E2 02:加速度数据包, 0xA0为数据包标签，三轴加速度为：

X轴加速度 = $(\text{int16_t})((0x02 < 8) + 0x55) = 597$ (单位为mG)

Y轴加速度 = $(\text{int16_t})((0x01 < 8) + 0x3D) = 317$

Z轴加速度 = $(\text{int16_t})((0x02 < 8) + 0xE2) = 738$

B0 FE FF 17 00 44 00:角速度数据包, 0xB0为数据包标签，三轴角速度为：

X轴角速度 = $(\text{int16_t})((0xFF < 8) + 0xFE) = -2$ (单位为0.1°/s)

Y轴角速度 = $(\text{int16_t})((0x00 < 8) + 0x17) = 23$

Z轴角速度 = $(\text{int16_t})((0x00 < 8) + 0x44) = 68$

C0 80 FF 60 FF 32 FF:磁场数据包, 0xC0为数据包标签，三轴磁场为：

X轴角速度 = $(\text{int16_t})((0xFF < 8) + 0x80) = -128$ (单位为0.001Gauss)

Y轴角速度 = $(\text{int16_t})((0xFF < 8) + 0x60) = -160$

Z轴角速度 = $(\text{int16_t})((0xFF < 8) + 0x32) = -206$

D0 64 F2 6C 0E BB 01 欧拉角数据包, 0xD0为数据包标签

Pitch= (int16_t)((0xF2<<8)+ 0x64) / 100 = -3484 / 100 = -34.84°

Roll= (int16_t)((0x0E<<8)+ 0x6C) / 100 = 3692 / 100 = 36.92°

Yaw = (int16_t)((0x01<<8)+ 0xBB) / 10 = 443 / 10 = 44.3°

F0 00 00 00 00气压数据包，0xF0为数据包标签

```
1 float prs;
2 prs = memcpy(&prs, &buf[37], 4);
```

最后得到结果：

1	id	:	0		
2	acc (G)	:	0.597	0.317	0.738
3	gyr (deg/s)	:	-0.200	2.300	6.800
4	mag (uT)	:	-12.800	-16.000	-20.600
5	eul (R/P/Y)	:	36.920	-34.840	44.300

数据帧配置为 0x91 数据包

使用串口助手采样一帧数据,共82字节,前6字节为帧头,长度和CRC校验值。剩余76字节为数据域。假设数据接收到C语言数组buf中。如下所示:

5A A5 4C 00 6C 51 91 00 A0 3B 01 A8 02 97 BD BB 04 00 9C A0 65 3E A2 26 45 3F 5C E7 30 3F E2 D4
5A C2 E5 9D A0 C1 EB 23 EE C2 78 77 99 41 AB AA D1 C1 AB 2A 0A C2 8D E1 42 42 8F 1D A8 C1 1E
0C 36 C2 E6 E5 5A 3F C1 94 9E 3E B8 C0 9E BE BE DF 8D BE

- 第一步：判断帧头，得到数据域长度和帧CRC：

帧头:5A A5

帧数据域长度:4C 00: (0x00<<8) + 0x4C = 76

帧CRC校验值:6C 51: (0x51<<8) + 0x6C = 0x516C

- 第二步：校验CRC

```
1  uint16_t payload_len;
2  uint16_t crc;
3
4  crc = 0;
5  payload_len = buf[2] + (buf[3] << 8);
6
7  /* calculate 5A A5 and LEN filed crc */
8  crc16_update(&crc, buf, 4);
9
10 /* calculate payload crc */
11 crc16_update(&crc, buf + 6, payload_len);
```

得到CRC值为0x516C. 帧CRC校验通过。

- 第三步：接收数据

从0x91开始为数据包的数据域。在C语言中可以定义结构体来方便的读取数据：

定义0x91数据包结构体如下：

```
1  __packed typedef struct
2  {
3      uint8_t    tag;                /* data packet tag */
4      uint8_t    id;
5      uint8_t    rev[6];            /* reserved */
6      uint32_t    ts;                /* timestamp */
7      float      acc[3];
8      float      gyr[3];
9      float      mag[3];
10     float      eul[3];            /* euler angles: Roll,Pitch,Yaw */
11     float      quat[4];           /* quaternion */
12 }id0x91_t;
```

__packed为编译器关键字(Keil下)，表示结构体按字节紧对齐，结构体每一个元素一一对应0x91数据包的结构定义。接收数据时将接收到的数组直接memcpy到结构体即可：(注意定义结构体时必须4字节对齐),其中buf指向帧头,buf[6]指向帧中数据域。

```
1  /* 接收数据并使用0x91数据包结构定义来解释数据 */
2  __align(4) id0x91_t dat; /* struct must be 4 byte aligned */
3  memcpy(&dat, &buf[6], sizeof(id0x91_t));
```

最后得到dat数据结果：

```
1  id          : 0
2  timestamp   : 310205
3  acc         :    0.224    0.770    0.691
4  gyr         :  -54.708  -20.077 -119.070
5  mag         :   19.183  -26.208  -34.542
6  eul (R/P/Y) :   48.720  -21.014  -45.512
7  quat        :    0.855    0.310   -0.310   -0.277
```

CAN通讯协议

CAN通讯协议遵循CANopen协议。基本配置如下：

- 只使用PTO1, PTO2, PTO3, PTO4 传输姿态数据,节点不接收或发送远程帧，所有传输均采用标准数据帧。
- PTO采用异步定时触发模式, 默认输出速率为20Hz
- 节点上电处于预操作状态(pre-operational) 需要主机发送NMT协议将节点设置为operation状态才会开始发送数据

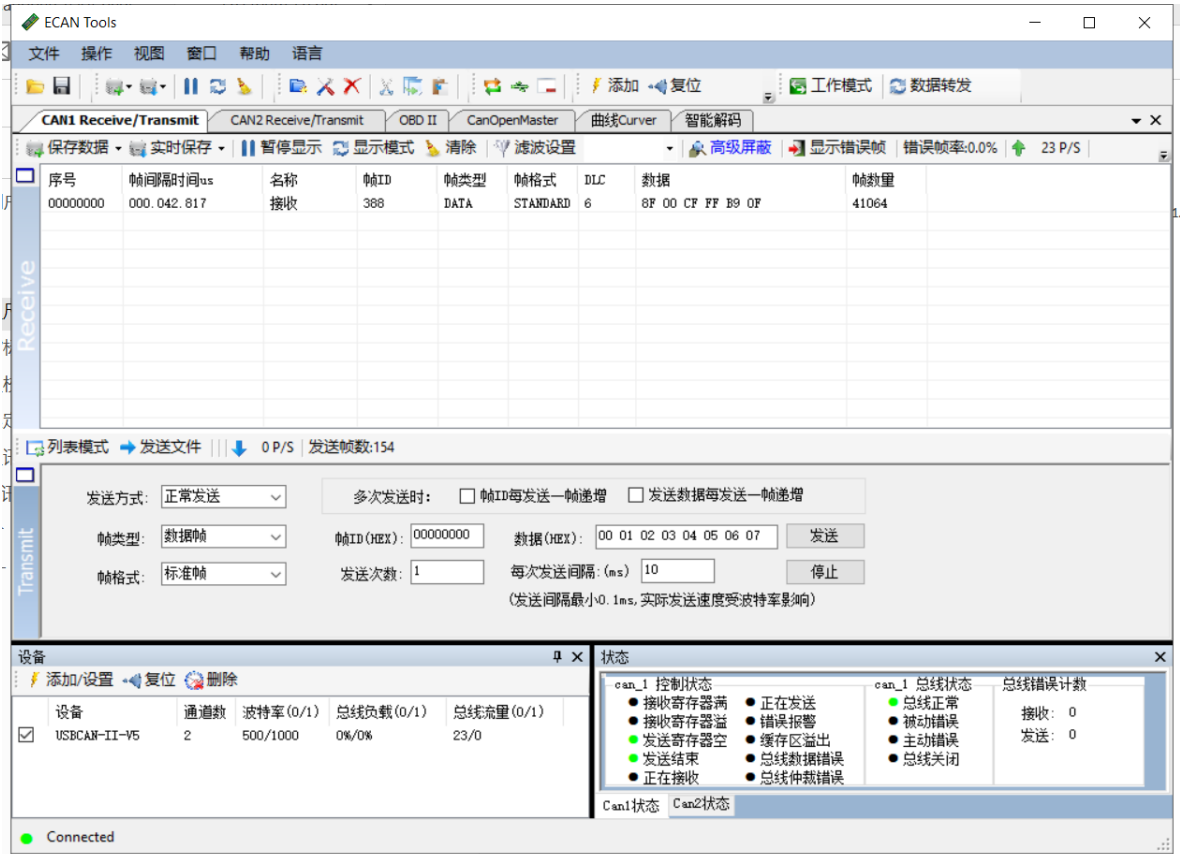
CAN默认设置

默认配置	值
CAN 波特率	500KHz
节点ID	8
初始化状态	预操作(Pre-operational),需要发送Start Remote Node命令模块才能开始输出
心跳包	无

PTO传输细节

PTO通道	PTO 帧ID	长度	PTO 传输方式	发送数据	说明
TPDO1	0x180+ID	6	循环同步 (0x01)	加速度	每轴数据类型为(INT16) , 分别为X,Y,Z轴加速度 , 单位为mG(0.001重力加速度)
TPDO2	0x280+ID	6	循环同步 (0x01)	角速度	每轴数据类型为(INT16) , 分别为X,Y,Z轴角速度 , 单位为0.1DPS(%s)
TPDO3	0x380+ID	6	异步定时 (0xFE)	欧拉角	每轴数据类型为(INT16) , 顺序分别为横滚角(Roll,绕X轴旋转),俯仰角(Pitch,绕Y轴旋转),航向角(Yaw绕Z轴旋转)。欧拉角单位为0.01°
TPDO4	0x380+ID	8	循环同步 (0x01))	四元数	每轴数据类型为(INT16) , 分别为 q_w q_x q_y q_z 。单位四元数扩大10000倍后结果。如四元数为1,0,0,0 时, 输出10000,0,0,0.

使用USB-CAN工具抓取默认CAN输出包截图如下：



其中 欧拉角(PTO3) CAN帧ID =0x380 + 8(默认ID) = 0x388

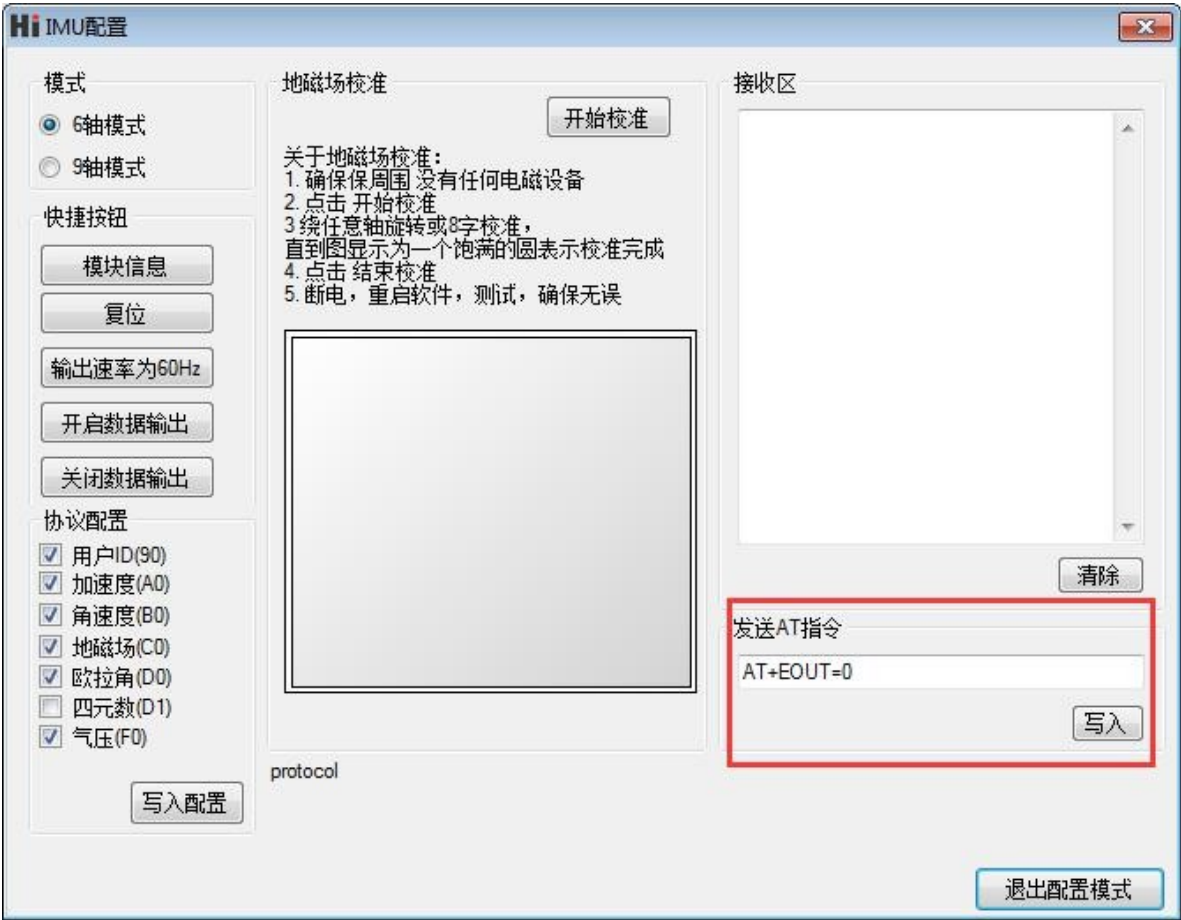
修改CAN接口配置

数据字典以下位置存放厂商参数配置数据, 可通过CANopen主站修改 , 掉电保存 , 重启生效

地址	名称	值类型	默认值	说明
0x2100	CAN_BAUD	INTEGER32	500000	CAN总线波特率
0x2101	NodeID	INTEGER32	8	节点ID

AT指令

当使用串口与模块通讯时,模块支持AT 指令集配置/查看模块参数。AT 指令总以ASCII 码AT 开头,后面跟控制字符,最后以回车换行\r\n结束。可使用串口调试助手进行测试：



通用模块 AT指令如下

指令	功能	掉电保存(Y)	立即生效(Y),复位生效(R)
AT+ID	设置模块用户ID	Y	R
AT+URFR	旋转模块传感器坐标系	Y	R
AT+INFO	打印模块信息	N	Y
AT+ODR	设置模块串口输出帧频率	Y	R
AT+BAUD	设置串口波特率	Y	R
AT+EOUT	数据输出开关	N	Y
AT+RST	复位模块	N	Y
AT+TRG	单次输出触发	N	Y
AT+SETPEL	设置输出数据包	Y	Y
AT+MODE	设置模块工作模式	Y	R
AT+GYRCTL	设置陀螺仪限幅滤波器参数	Y	R
AT+GWID	设置无线网关ID	Y	R
AT+GWCFG	设置接收机无线网络属性	Y	R

产品支持数据包列表

下表列出所有产品支持的数据包,* 表示支持 -表示不支持

产品	ID	URFR	INFO	ODR	BAUD	EOUT	RST	TRG	SETPTL	MODE	GYRCTL	GWID	GWCFG
HI226	*	*	*	*	*	*	*	*	*	-	*	-	-
HI229	*	*	*	*	*	*	*	*	*	*	*	-	-
CH110	*	*	*	*	*	*	*	-	*	-	*	-	-
HI221	*	*	*	*	*	*	*	-	-	-	*	*	-
HI221GW	*	*	*	*	*	*	*	-	-	-	*	*	*

AT+ID

设置模块用户ID

例 AT+ID=1

AT+URFR

某些情况下传感器需要倾斜垂直安装，这时候需要旋转传感器坐标系，这条指令提供了旋转传感器坐标系的接口：

AT+URFR=C00,C01,C02,C10,C11,C12,C20,C21,C22

其中 C_{nm} 支持浮点数

$$\begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix}_U = \begin{bmatrix} C00 & C01 & C02 \\ C10 & C11 & C12 \\ C20 & C21 & C22 \end{bmatrix} \cdot \begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix}_B$$

(1)

其中 $\begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix}_U$ 为旋转后的 传感器坐标系下 传感器数据， $\begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix}_B$ 为旋转前 传感器坐标系下 传感器数据

下面是几种常用旋转举例：

- 新传感器坐标系为 绕原坐标系X轴 旋转 90°， 输入命令：AT+URFR=1,0,0,0,0,1,0,-1,0
- 新传感器坐标系为 绕原坐标系X轴 旋转-90°， 输入命令：AT+URFR=1,0,0,0,0,-1,0,1,0
- 新传感器坐标系为 绕原坐标系X轴 旋转180°， 输入命令：AT+URFR=1,0,0,0,-1,0,0,0,-1
- 新传感器坐标系为 绕原坐标系Y轴 旋转 90°， 输入命令：AT+URFR= 0,0,-1,0,1,0,1,0,0
- 新传感器坐标系为 绕原坐标系Y轴 旋转-90°， 输入命令：AT+URFR= 0,0,1,0,1,0,-1,0,0
- 新传感器坐标系为 绕原坐标系Y轴 旋转180°， 输入命令：AT+URFR= -1,0,0,0,1,0,0,0,-1
- 恢复默认值：AT+URFR=1,0,0,0,1,0,0,0,1

AT+INFO

打印模块信息，包括产品型号，版本，固件发布日期等。AT+INFO可以拓展二级指令实现更多信息的查询

INFO二级拓展指令	功能	示例
CAL	显示模块内部校准参数	AT+INFO=CAL
RF	显示无线设备参数	AT+INFO=RF
VER	显示详细版本信息	AT+INFO=VER

AT+ODR

设置模块串口输出速率。掉电保存，复位模块生效

例 设置串口输出速率为100Hz: AT+ODR=100

AT+BAUD

设置串口波特率，可选值：4800/9600/115200/256000/460800`

例 AT+BAUD=115200

!!! note "注意"

- 使用此指令需要特别注意，输入错误波特率后可能会导致无法和模块通讯
- 波特率参数设置好后掉电保存，复位模块生效。上位机的波特率也要做相应修改。
- 升级固件时，需要切换回115200 波特率。

AT+EOUT

串口输出开关

例 打开串口输出 AT+EOUT=1 关闭串口输出 AT+EOUT=0

AT+RST

复位模块

例 AT+RST

AT+TRG

触发模块输出一帧数据，可以配合AT+ODR=0来实现单次触发输出。

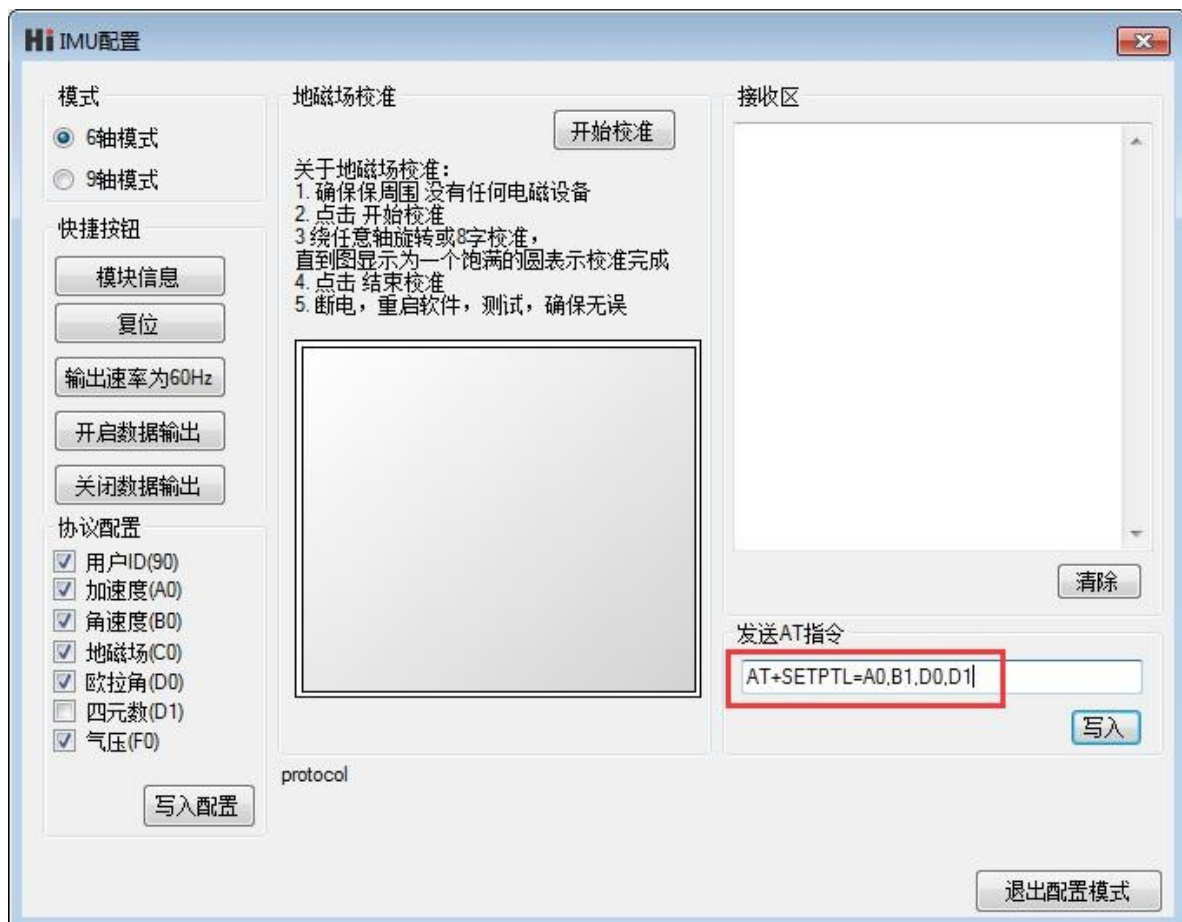
例 AT+TRG

AT+SETPEL

设置输出协议:

模块数据帧中的数据包组成可使用AT指令配置，格式为AT+SETPTL=<ITEM_ID>,<ITEM_ID>... 一帧输出可包含最多8个数据包。

例 配置模块输出加速度，角速度, 整形格式欧拉角和四元数的指令为：AT+SETPTL=A0,B1,D0,D1



AT+MODE

设置模块工作模式

例

- 设置模块工作在6轴模式(无磁校准) AT+MODE=0
- 设置模块工作在9轴模式(地磁场传感器参与航向角校正) AT+MODE=1

AT+GYRCTL

设置陀螺限幅滤波器阈值(出厂默认值1.0)

例

- 设置陀螺限幅滤波器限幅阈值为1.5°/s: AT+GYRCTL=LMF, 1.5

陀螺限幅滤波主要为了解决陀螺零偏问题，当模块静止经受机械振动时(比如安装在机器上，机器人上电待机但不移动时会有电机空载振动)，陀螺航向角会因为Z轴振动而缓慢飘移。当陀螺Z轴小于设定阈值时，模块则会强制把Z轴数值归0。

限幅滤波可以解决由于陀螺零偏或小范围振动时带来的航向角飘移问题，但缺点是小于设定阈值的转动会无法检测。一般情况下，阈值设定范围应在0.1 - 3.5之间。

限幅阈值过高会导致小于阈值的转动角速度无法检测

限幅阈值过低会导致陀螺零偏无法校准，航向角随时间缓慢飘移。

AT+GWID

可通过AT+GWID指令配置，GWID属性决定了接收器和节点的RF频率，只有节点的GWID 和接收器的GWID相同时，模块和接收器直接才能通讯。GWID相当于无线网段，当在同一地点使用多个接收机组成多个星形网络时，必须保证每个接收器的GWID(网段)不同。

例 将一个接收器设置为GWID=3， 并将3个模块的自身ID设置为 0,1,2 并连接到这个接收器上：

接收机配置：

AT+GWID=3

节点0配置：

AT+GWID=3

AT+ID=0

节点1配置：

AT+GWID=3

AT+ID=1

节点2配置：

AT+GWID=3

AT+ID=2

最后所有接收机节点复位/重新上电生效。

AT+GWCFG

配置接收机支持的节点数和无线通讯频率。接收机默认支持8个节点，每个节点100Hz通讯频率。(通讯频率乘以节点)的乘积受到RF带宽的限制，错误的配置将会导致无法输出数据。推荐的配置如下：

配置选项	通讯频率(Hz,每个节点)	支持的节点数
1(出厂默认)	100	8
2	200	4
3	30	16

例 配置接收机支持的最大节点数为16，每个节点接收频率为30Hz，依次输入：

AT+GWCFG=FRQ, 30

附录C - 固件升级与恢复出厂设置

本产品支持升级固件。固件升级步骤:

- 连接模块，打开上位机，将模块和上位机波特率都设置为115200. 打开固件升级窗口
- 点击连接按钮，如出现模块连接信息。则说明升级系统准备就绪，点击文件选择器(...)选择拓展名为.hex 的固件，然后点击开始编程。下载完成后会提示编程完成，此时关闭串口，重新给模块上电，模块升级完成。

