目录

1. 串口协议

基于 QT、ROS 和 STM32，Linux 的串口协议示例 ：

https://data.forsense-imu.com/ProductFile/link/Upper%20computer%20software/AHRS%20ROS2%20drive/download.php

串口通信具有两种模式：数据流模式(Stream Mode)和命令模式(Command Mode)，IMU 在上电初始化完成后，根据参数配置的模式值进入对应模式。

数据流模式：以固定频率周期性输出数据；

命令模式：在此模式下，停止周期性输出，用户通过发送命令与 IMU 进行通信，可通过 GET 指令获取传感器数据、状态、参数等，也可配置 IMU 的参数。

1.1 串口设置参数的指令

1.1.1 AT 指令

功能	指令	应答	描述
停止当前数据流输出	AT+SETNO\r\n	OK	可以停掉当前数据流(不改数据流参数)，应答 OK 后表示可以进行下一步操作。如果未响应，可以继续发送 AT+SETNO\r\n 命令直到应答 OK。
开启当前数据流输出	AT+SETYES\r\n	\	开启数据流，如果配置的数据流模式为 AHRS，则持续输出 AHRS 数据
查询版本号	AT+VERSION\r\n	SW_VERSION HW_VERSION BOARD_VERSION	固件版本 硬件版本 底板版本
查询用户参数	AT+CONFIG\r\n	BAUD_RATE ORIENT IMU_ODR STREAM_MODE1 STREAM_MODE2 STREAM_MODE3 LP_CONFIG_REG	当前串口波特率 当前坐标系 当前 IMU 的输出频率 当前串口 1 的数据流模式 当前串口 2 的数据流模式 当前串口 3 的数据流模式 当前 IMU 的滤波

设置 IMU 的 ODR	AT+SET_ODR=50\r\n	IMU_ODR:50	设置输出频率 ODR 为 50Hz
查询 IMU 的 ODR	AT+GET_ODR\r\n	IMU_ODR:50	查询 IMU 的 ODR
设置波特率	AT+SET_BAUD=115200\r\n	OK	设置 IMU 的波特率为 115200
查询波特率	AT+GET_BAUD\r\n	BAUD_RATE:115200	查询 IMU 当前波特率
设置滤波器	AT+SET_LPF=187\r\n	LP_CONFIG_REG:187	设置 IMU 的滤波为 no filter
查询滤波器	AT+GET_LPF\r\n	LP_CONFIG_REG:187	查询 IMU 当前滤波
设置 IMU 坐标系	AT+SET_ORIENT=101\r\n	+SET ORIENT orientation:101 OK	设置 IMU 坐标系为 101
读取 IMU 坐标系	AT+GET_ORIENT\r\n	ORIENT:101	查询 IMU 当前坐标系
设置姿态坐标系	AT+SET_ATT_AXIS=1\r\n	\	0：前右下（默认） 1：前左上 2：右前上
设置数据流类型	AT+SET_STREAM1=1\r\n	STREAM_MODE1:1	串口 1 设置数据流指令， 设置 1 为 AHRS 数据流， 设置 51 为 AHRS 扩展模式， 设置 100 为命令模式 设置其他数字 为自定义数据流 STREAM_MODE2 为串口 2 的数据流模式 STREAM_MODE3 为串口 3 的数据流模式 STREAM_MODE4 为串口 4 的数据流模式
获取串口数据流类型	AT+GET_STREAM1\r\n	STREAM_MODE1:1	获取串口 1 的数据流
设置横滚俯仰取反	AT+SET_ATT_ORIENTATION=00\r\n	OK	00:横滚俯仰不取反 01：横滚取反，俯仰不取反 10：横滚不取反，俯仰取反 11：横滚俯仰取反
获取横滚俯仰	AT+GET_ATT_ORIENTATION		新功能，大于 240987 的固件版本支持

状态			
重置航向角	AT+SET_HEADING=0\r\n	HEADING:0 OK	设置后，航向角被重置，分辨率 1，只支持整数
重置航向角高精度	AT+SET_HEADING_HIGH_LSB=0\r\n	HEADING:0.0 OK	设置后，航向角被重置，分辨率 0.01
开启 PPS 功能	AT+SETIIC=0\r\n	OK	618 系列开启 PPS 功能； 其他型号 IMU 的开启指令见 5.3.5 节
姿态校准	AT+TRIM_CALIB\r\n	INSTALL_ERR_ALPHA,INSTALL_ERR_BEAT OK	姿态校准，适用于非水平安装
保存参数	AT+SAVE\r\n	OK	保存参数到 flash

注 1：配置参数完成后需发送保存参数的指令，否则断电重启后会恢复为配置前的参数。

注 2：配置参数并保存后，需断电重启 IMU 生效。

1.1.2 二进制指令

二进制指令使用难度较大，但是支持功能更多，建议一般情况下用 AT 指令方式。

1.1.2.1 获取系统状态的 HEX 指令及应答格式

串口系统状态数据格式

	帧头	帧头	ID	length	payload	帧尾
数据类型	uint8	uint8	uint16	uint16	S1	uint32
编码	0xAA	0x55	0x00FF	N		crc32

注 ：不同 IMU 型号，此帧的长度会有差别，都代表 S1 的长度，需要根据 IMU 型号确认。

串口 S1 负载数据格式

offset	名称	数据类型	描述
0	Software_ver	uint32	软件版本号
4	Hardware_ver	uint32	硬件版本号
8	rev	uint16	保留字节
10	sn0	uint32	第一 SN 号
14	sn1	uint32	第二 SN 号
18	sn2	uint32	第三 SN 号
22	Board_version	uint32	底板版本号

26	Rev[16]	Uint8	后续都是保留字节

注 ：不同 IMU 型号，后续保留字节也不同，需要根据 IMU 型号进行确认，IMU614E 为 16 字节。

例：获取系统状态

输入数据：55 AA 01 00 18 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 BD DB 31 34

响应数据：AA 55 FF 00 2A 00 1F 39 03 00 65 6F 01 00 50 83 30 33 35 55 34 50 15 FF 8F 5F FF FF 50 83 FF 1F 29 00 00 00 00 E0 00 07 10 17 08 50 D0 37 10 3B 7A C3 00 02

根据响应数据，解析得到软件版本号 211231(1F 39 03 00),硬件版本号 94053(65 6F 01 00)。

1.1.2.2 读取参数的 HEX 指令及应答格式

读取参数 HEX 输入数据格式

	帧头	帧头	ID	length	payload	帧尾
数据类型	uint8	uint8	uint16	uint16	P1	uint32
编码	0x55	0xAA	0x0006	0x0018		crc32

读取参数 HEX 输出数据格式

	帧头	帧头	ID	length	payload	帧尾
数据类型	uint8	uint8	uint16	uint16	P1	uint32
编码	0xAA	0x55	0x7530	0x0018		crc32

P1 负载数据格式

offset	名称	数据类型	描述
0	Param1	float	获取的参数（输入数据可无视）
4	Param2	float	保留，默认为 0
8	Param3	uint32	设置的参数索引
12	Param4	uint32	保留，默认为 0
16	Param5	Int32	保留，默认为 0
20	Param6	Int32	保留，默认为 0

串口 P1 负载参数索引表

Param3	Param1	单位
3	串口输出波特率，支持以下波特率 115200、230400、460800、921600、1500000	bps

4	坐标系朝向（见表 25 坐标系朝向对应表）	/
8	X 轴陀螺零偏标定结果，GYRO_X_OFF	°/s
9	Y 轴陀螺零偏标定结果，GYRO_Y_OFF	°/s
10	Z 轴陀螺零偏标定结果，GYRO_Z_OFF	°/s
21	AHRS 输出频率，默认 100Hz	Hz
31	内部滤波器配置，定义同 SPI 的 FILTER_CTRL 对照表	

例：获取 AHRS 输出频率

输入数据：55 AA 06 00 18 00 00 00 00 00 00 00 00 00 15 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 66 CB 46 AC

响应数据：AA 55 30 75 18 00 00 00 48 42 00 00 00 00 15 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 31 2F A2 0A

根据响应数据，解析得到输出频率为 50Hz (00 00 48 42)。

1.1.2.3 设置参数的 HEX 指令及应答格式

串口输入 HEX 命令格式

	帧头	帧头	ID	length	payload	帧尾
数据类型	uint8	uint8	uint16	uint16	R1	uint32
编码	0x55	0xAA	CMD	0x0018		crc32

paload 格式

offset	名称	数据类型	描述
0	Param1	float	设置的参数
4	Param2	float	保留，默认为 0
8	Param3	uint32	设置的参数索引
12	Param4	uint32	保留，默认为 0
16	Param5	Int32	保留，默认为 0
20	Param6	Int32	保留，默认为 0

参数索引表

CMD	Param1	Param3	描述
1	0	0	触发获取一次系统状态数据
2	0	0	触发获取一次 AHRS 数据

3	<mode>	0	设置输出模式： Mode=1， 数据流输出 AHRS Mode=100，禁止数据流模式，进入 COMMAD 模式
5	0	0	保存当前参数到 FLASH
6	0	<value>	读取参数，value 为要读取的参数索引，即 P1.index，详见串口应答性输出-参数读取 例如需读取 AHRS 输出频率（ODR），则设置 value=21 例如需读取串口波特率，则设置 value=3 例如需读取内部滤波器，则设置 value=31 例如需读取坐标系方向，则设置 value=4
9	0	0	执行软件重启
14	<value>	3	设置串口输出波特率，单位 bps value 的有效值为： 115200，230400，460800，921600，1500000 value 为其他值时，默认采用 115200bps 设置波特率参数后，需要重启才能生效。 不断电的设置流程：设置波特率，保存参数到 flash，执行软件复位
14	<value>	21	设置周期性 AHRS 数据输出频率，单位 Hz value 的常用值为： 1，10，50，100，200，500，1000 输出频率与串口波特率的推荐对应关系 1000Hz：921600bps 500Hz：460800bps 250Hz：460800bps 200Hz：460800bps 100Hz：115200bps
14	<value>	31	内部滤波器配置，定义同 SPI 加速度计和陀螺仪滤波器配置，默认 0xBB，即 no filter
14	<value>	4	设置 IMU 坐标系朝向，value 的取值范围为 101~124，具体坐标系朝向对应关系见本章《1.7 坐标系设置》

注 1：请注意 参数索引表 中数值均为十进制

注 2：配置参数完成后需发送保存参数的指令，否则断电重启后会恢复为配置前的参数。

注 3：配置参数并保存后，需断电重启 IMU，否则可能不生效。

注 4：可使用上位机命令生成器功能生成对应命令发送。

注 5： HEX 指令生成的代码示例如下

```
C++
/**
 * 命令通用结构体
 */
struct   MULTI_LONG_CMD_STRUCT
{
uint8_t header1;
```

```
uint8_t header2;
uint16_t id;
uint16_t length;
float param1;
float param2;
uint32_t param3;
uint32_t param4;
int32_t param5;
int32_t param6;
uint32_t check_crc;
}__attribute__((packed));

struct    MULTI_LONG_CMD_STRUCT data_cmd_long;

void    Send_CMD_LONG(uint16_t cmd_id,float cm1,float cm2,uint32_t cm3,uint32_t
cm4,int32_t cm5,int32_t cm6)
{
      uint32_t check_crc=0;

      data_cmd_long.header1=0x55;
      data_cmd_long.header2=0xaa;
      data_cmd_long.id=cmd_id;
      data_cmd_long.length=sizeof(data_cmd_long)-10;

      data_cmd_long.param1=cm1;
      data_cmd_long.param2=cm2;
      data_cmd_long.param3=cm3;
      data_cmd_long.param4=cm4;
      data_cmd_long.param5=cm5;
      data_cmd_long.param6=cm6;

      int len=sizeof(data_cmd_long)-4;
      check_crc=1;
      data_cmd_long.check_crc=crc_crc32(check_crc,(uint8_t *)(&data_cmd_long), len);
}
```

示例：如执行开启 AHRS 输出：

CMD ID 填入 3，参数 1 填入 1，生成的十六进制数组可以填入串口助手或程序数组中发送给 IMU。

函数代码调用：

```
Plain Text
Send_CMD_LONG(3,1,0,0,0,0,0); //获取 AHRS 数据流
Send_CMD_LONG(5,0,0,0,0,0,0); //保存数据
```

上位机生成 HEX 指令示意图

HEX 指令设置参数的响应数据格式

	帧头	帧头	ID	length	ACK	Param3	帧尾
数据类型	uint8	uint8	uint16	uint16	uint16	uint16	uint32
编码	0xAA	0x55	0x753D	0x0004	0x7534	参数索引	crc32

例：设置串口输出波特率 115200

输入数据：
55,AA,0E,00,18,00,00,00,E1,47,00,00,00,00,03,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,56,2B,4D,93

响应数据：AA 55 3D 75 04 00 34 75 03 00 A7 98 2A 54

设置周期性 AHRS 数据输出频率 100Hz

输入数据：55 AA 0E 00 18 00 00 00 C8 42 00 00 00 00 15 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0A 2B 2C 8D

响应数据：AA 55 3D 75 04 00 34 75 15 00 70 2D B2 48

HEX 指令保存参数的响应数据格式

	帧头	帧头	ID	length	ACK	result	帧尾
数据类型	uint8	uint8	uint16	uint16	uint16	uint16	uint32
编码	0xAA	0x55	0x753D	0x0004	0x0005	0x01	crc32

保存当前参数到 FLASH

输入数据：55 AA 05 00 18 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 C9 2F E6 32

响应数据：AA 55 3D 75 04 00 05 00 01 00 5A CF B1 7C

串口用户命令响应数据格式

	帧头	帧头	ID	length	command	result	帧尾
数据类型	uint8	uint8	uint16	uint16	uint16	uint16	uint32
编码	0xAA	0x55	0x0064	0x0004	命令 ID	0x01	crc32

设置输出模式为 AHRS 数据流

输入数据：55 AA 03 00 18 00 00 00 80 3F 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 52 D8 8E E8

响应数据：AA 55 64 00 04 00 03 00 01 00 E7 87 E3 AD

1.2 AHRS 模式

以固定频率周期性的主动输出 AHRS 格式数据，数据流类型为 1。

1.2.1 协议格式

	帧头	帧头	ID	length	payload	帧尾
数据类型	uint8	uint8	uint16	uint16	A1	uint32
编码	0xAA	0x55	0x0002	0x002C		crc32

1.2.2 AHRS 模式 payload 表

payload				
offset	名称	数据类型	单位	描述
0	timer	uint32	μs	时间标
4	pitch	float	°	俯仰角
8	roll	float	°	横滚角
12	yaw	float	°	航向角
16	ax	float	g	X 轴加速度
20	ay	float	g	Y 轴加速度
24	az	float	g	Z 轴加速度 30
28	gx	float	°/s	X 轴角速度

32	gy	float	°/s	Y 轴角速度
36	gz	float	°/s	Z 轴角速度
40	temp	float	℃	IMU 芯片温度

1.2.3 协议解析

描述	数据类型	原始值	解析值
帧头 1	uint8	0xAA	0xAA
帧头 2	uint8	0x55	0x55
ID	uint16	0x0200	0x0002
长度	uint16	0x2C00	44
时间标	uint32	6D891605	85363053
俯仰角	float	8FC26540	3.59°
横滚角	float	14AE07BF	-0.53°
航向角	float	5C0FB243	356.12°
X 轴加速度	float	2506813D	0.063g
Y 轴加速度	float	BC74133C	0.009g
Z 轴加速度	float	60E580BF	-1.007g
X 轴角速度	float	EC5138BD	-0.045°/s
Y 轴角速度	float	0AD7A3BB	-0.005°/s
Z 轴角速度	float	CDCCCCBC	-0.025°/s
imu 芯片温度	float	D7A3EE41	29.83℃
crc32 校验	uint32	0CBF8480	2156183308

注 1 ：数据以小端格式传输，低字节在前，高字节在后

注 2：crc32 的初值为 1，CRC 计算不包括本身的本帧所有数据，查表计算法见文档末尾

原始值转化为解析值示例代码(以上表中的航向角为例)

```
c++
#include <stdio.h>
#include <stdint.h>
int main(){
float     yaw;//定义一个解析值
uint32_t     a = 0x43b20f5c;//定义一个原始值 a,注意为小端模式
yaw = *(float *)&a;//将原始值强转为浮点型
printf("%f\n",yaw);//打印输出解析值
return 0;
}
```

1.3 AHRS 扩展模式

以固定频率周期性的主动输出 AHRS 扩展格式数据，数据流类型为 51。

1.3.1 协议格式

	帧头	帧头	ID	length	payload	帧尾
数据类型	uint8	uint8	uint16	uint16	A1	uint32
编码	0xAA	0x55	0x0022	0x0031		crc32

1.3.2 AHRS 扩展模式 payload 表

payload				
offset	名称	数据类型	单位	描述
0	timer	uint32	μs	时间标
4	pitch	float	°	俯仰角
8	roll	float	°	横滚角
12	yaw	float	°	航向角
16	ax	float	g	X 轴加速度
20	ay	float	g	Y 轴加速度
24	az	float	g	Z 轴加速度
28	gx	float	°/s	X 轴角速度
32	gy	float	°/s	Y 轴角速度
36	gz	float	°/s	Z 轴角速度
40	temp	float	℃	IMU 芯片温度
44	flag	uint8	\	标志位 未扣零偏 0 已扣零偏 1
45	status	uint32	\	状态位 第一个字节置 1 是 Gyro 超量程 第二个字节置 1 表示是静态 其他字节预留

1.4 四元数模式

以固定频率周期性的主动输出四元数格式数据，数据流类型为 52。

1.4.1 协议格式

	帧头	帧头	ID	length	payload	帧尾
数据类型	uint8	uint8	uint16	uint16	A1	uint32
编码	0xAA	0x55	0x0023	0x0035		crc32

1.4.2 四元数模式 payload 表

payload				
offset	名称	数据类型	单位	描述
0	timer	uint32	μs	时间标
4	qw	float	°	qw
8	qx	float	°	qx
12	qy	float	°	qy
16	qz	float	°	qz
20	ax	float	g	X 轴加速度
24	ay	float	g	Y 轴加速度
28	az	float	g	Z 轴加速度
32	gx	float	°/s	X 轴角速度
36	gy	float	°/s	Y 轴角速度
40	gz	float	°/s	Z 轴角速度
44	temp	float	℃	IMU 芯片温度
48	flag	uint8	\	标志位 未扣零偏 0 已扣零偏 1
49	status	uint32	\	状态位 第一个字节置 1 是 Gyro 超量程 第二个字节置 1 表示是静态 其他字节预留

1.5 命令模式

命令模式为一问一答的形式，使用 HEX 指令设置为命令模式后，发送指令获取 AHRS 数据。详情参考 HEX 指令章节。

命令模式的数据流类型为 100。

```
C++
Send_CMD_LONG(3,100,0,0,0,0,0);//设置为命令模式
Send_CMD_LONG(5,0,0,0,0,0,0);//保存参数到 flash
Send_CMD_LONG(2,0,0,0,0,0,0);//触发获取一次 AHRS 数据
Send_CMD_LONG(9,0,0,0,0,0,0);//软件复位
Send_CMD_LONG(3,1,0,0,0,0,0);//设置为 AHRS 模式
Send_CMD_LONG(3,51,0,0,0,0,0);//设置为 AHRS 扩展模式
Send_CMD_LONG(3,52,0,0,0,0,0);//设置为四元数模式
```

1.6 滤波器设置

滤波器配置参数值为陀螺、加表滤波器值拼成一起，各占 4bit。

	编码	描述
加速度计/ 陀螺仪滤波器配置	4’b 0000	IIR filter fc=1 Hz
	4’b 0001	IIR filter fc=1 Hz
	4’b 0010	IIR filter fc=2 Hz
	4’b 0011	IIR filter fc=5 Hz
	4’b 0100	IIR filter fc=10 Hz
	4’b 0101	IIR filter fc=15 Hz
	4’b 0110	IIR filter fc=20 Hz
	4’b 0111	IIR filter fc=25 Hz
	4’b 1000	IIR filter fc=30 Hz
	4’b 1001	IIR filter fc=35 Hz
	4’b 1010	IIR filter fc=40 Hz
	4’b 1011	no filter

注：比如配置陀螺、加速度计滤波器为 10Hz，则写入滤波器设置值为 0x44，十进制为 68。

低通滤波值和 AT 指令对应数值

序号	IMU 低通滤波值	AT 指令对应的值
1	1	17
2	2	34
3	5	51

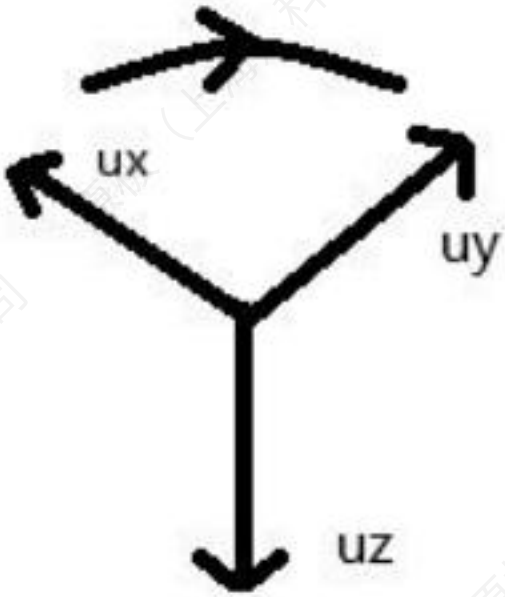
4	10	68
5	15	85
6	20	102
7	25	119
8	30	136
9	35	153
10	40	170
11	No filter	187

1.7 坐标系设置

本产品支持右手坐标系下的各种轴向变换，发送指令 AT 指令或 HEX 指令可修改坐标系轴向。

这种修改只能改 IMU 坐标系，姿态坐标系仍默认前右下，需要再设置其他参数才能改坐标系为前左上或右前上。

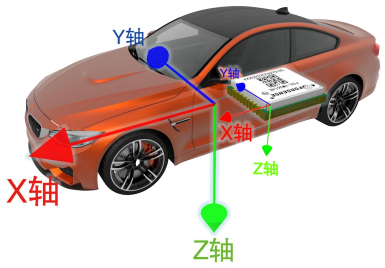
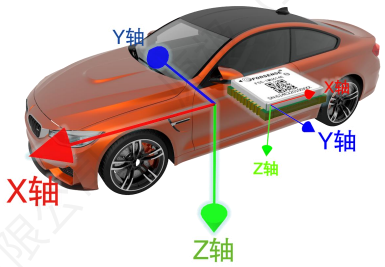
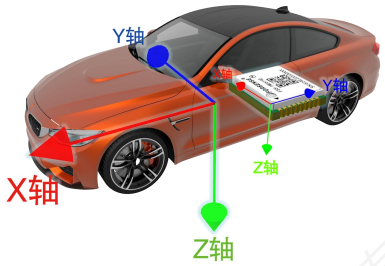
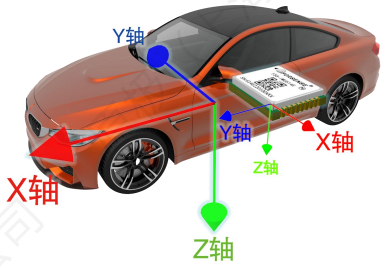
固件原始坐标系

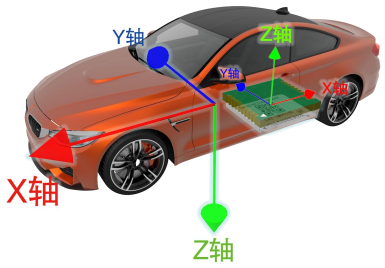
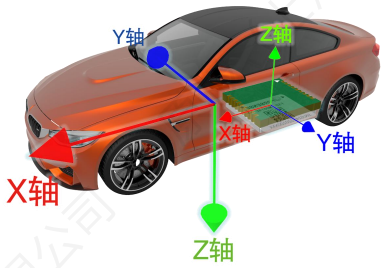
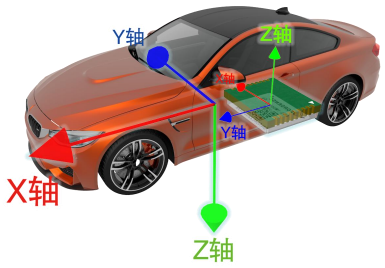
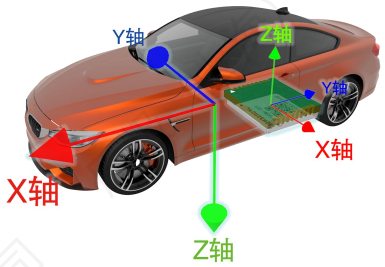


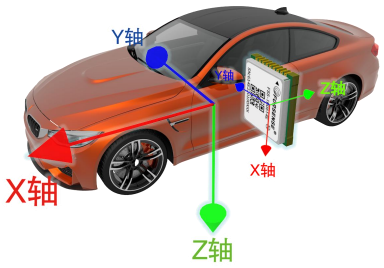
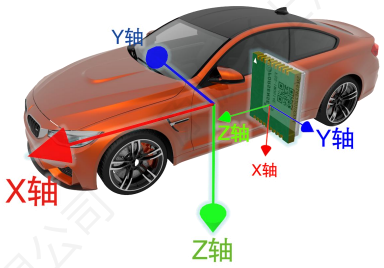
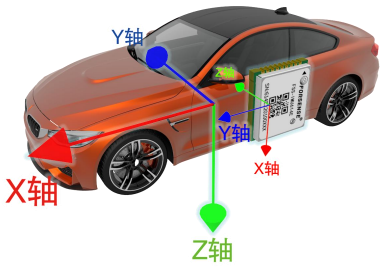
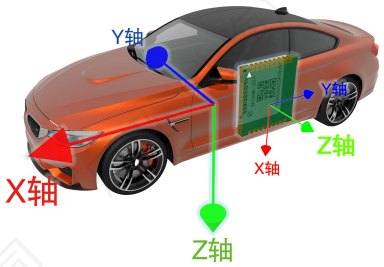
X/Y/Z 三轴的朝向总共有二十四种，如下表所示：

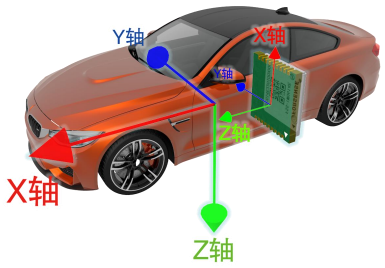
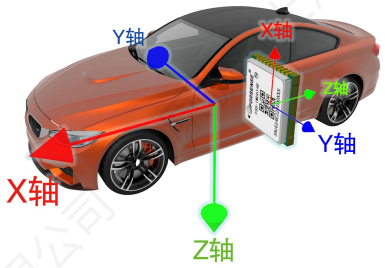
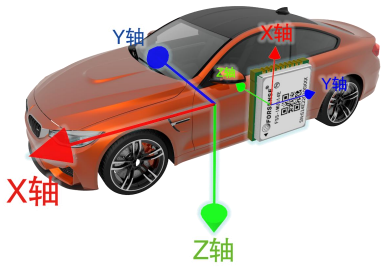
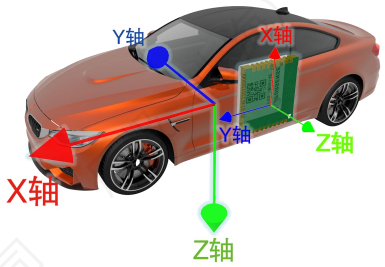
以 IMU614E 模块为例，小坐标轴是默认坐标系，大坐标轴是新坐标系

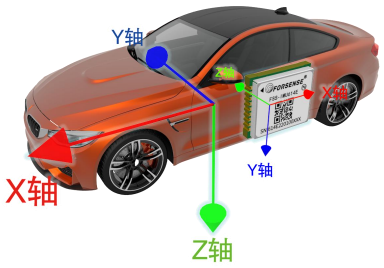
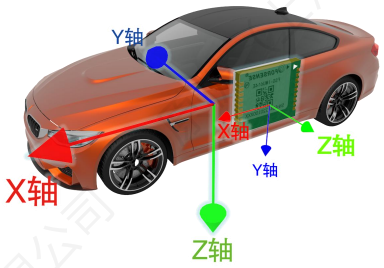
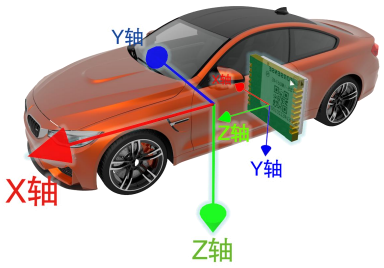
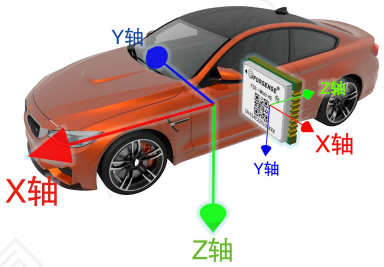
IMU 轴向	新轴向相比于原坐标系轴向			示意图
	XAxis	YAXis	ZAxis	

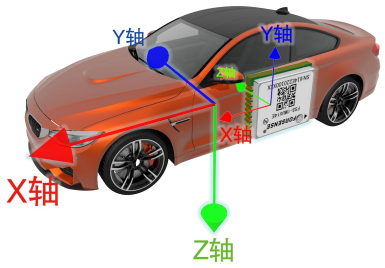
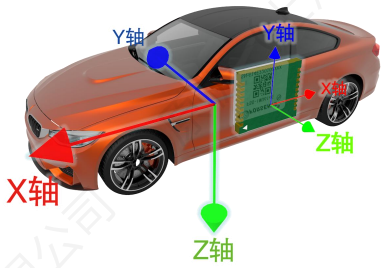
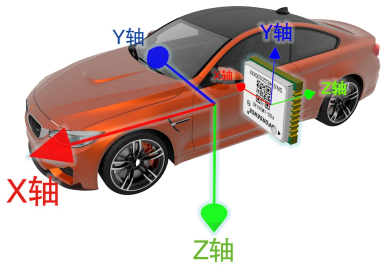
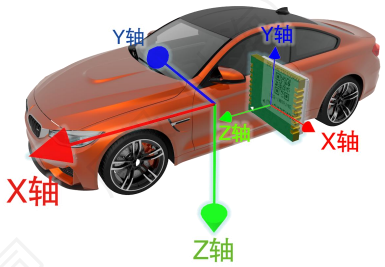
101	+Ux	+Uy	+Uz	 (默认坐标系)
102	-Ux	-Uy	+Uz	
103	-Uy	+Ux	+Uz	
104	+Uy	-Ux	+Uz	

105	-Ux	+Uy	-Uz	
106 （前左上）	+Ux	-Uy	-Uz	
107	+Uy	+Ux	-Uz	
108	-Uy	-Ux	-Uz	

109	-Uz	+Uy	+Ux	
110	+Uz	-Uy	+Ux	
111	+Uy	+Uz	+Ux	
112	-Uy	-Uz	+Ux	

113	+Uz	+Uy	-Ux	
114	-Uz	-Uy	-Ux	
115	-Uy	+Uz	-Ux	
116	+Uy	-Uz	-Ux	

117	-Ux	+Uz	+Uy	
118	+Ux	-Uz	+Uy	
119	+Uz	+Ux	+Uy	
120	-Uz	-Ux	+Uy	

121	+Ux	+Uz	-Uy	
122	-Ux	-Uz	-Uy	
123	-Uz	+Ux	-Uy	
124	+Uz	-Ux	-Uy	

如何更改坐标系为 102 朝向：

CMD ID 填入 14，参数 1 填入 102，参数 3 填入 4，生成的十六进制数组可以填入串口助手或程序数组中发送给 IMU。

更改坐标系

如何读取坐标系朝向：

CMD ID 填入 06，参数 3 填入 4，生成的十六进制数组可以填入串口助手或程序数组中发送给 IMU。

读取坐标系

例： 设置坐标系为 115 朝向

输入数据：

55,aa,0e,00,18,00,00,00,e6,42,00,00,00,00,04,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,46,6a,4e,86

响应数据：AA 55 3D 75 04 00 34 75 04 00 60 0E 6B 1B

解析得到参数索引为 04，设置成功

读取坐标系：

输入数据：55 AA 06 00 18 00 00 00 00 00 00 00 00 04 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 69 64 09 E4

响应数据：AA 55 30 75 18 00 00 00 E6 42 00 00 00 00 04 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 B2 2F 2D 4E

解析得到参数 1 为 115（float），参数 3 为 04。即坐标系为 115 朝向

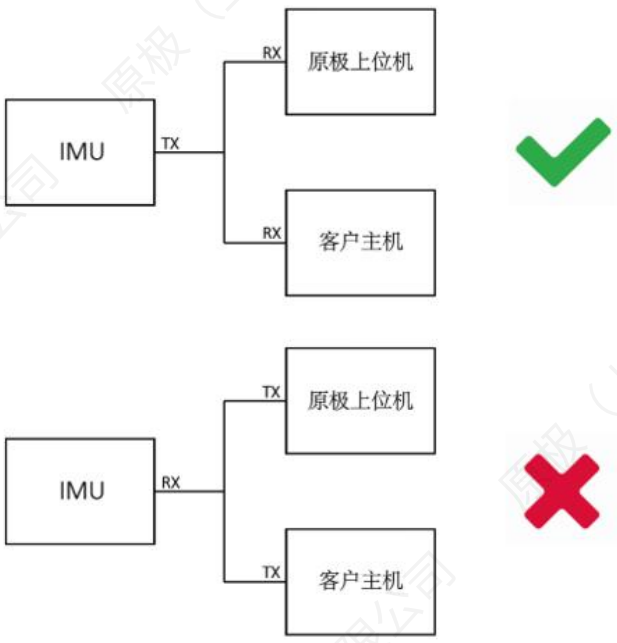
1.8 串口连接常见问题

1.8.1 IMU 的 RX 不能接 2 个主机 TX

串口的 RX 不能同时接 2 个 TX，所以如果需要连接原极上位机时，需要断开其与用户主机的串口通信，否则上位机只能接收到数据，不能发送命令给 IMU。

如下图所示：

串口连接方式示意图



注：IMU TX 可接多路 RX，RX 不可接多路 TX；
IMU串口不可同时连接客户主机和原极上位机；
IMU可以预留另外一路串口专门连接原极上位机。

1.8.2 获取不到版本号

检查串口线是否丢包，推荐使用 FT232 芯片的串口线，CH340、PL2303 数据线在高波特率时（>115200bps）会丢包

建议串口线直连，不建议串联，如 RS422 的接口接电脑，直接使用 RS422 转 USB 线，不要用 RS422 转 RS232+RS232Z 转 USB 线串联。

1.8.3 设置串口延时

如果上位机曲线显示卡顿，则需要设置串口延时。

如果是 FT232 数据线，用系统管理员打开上位机，自动配置串口延时。

手动在设备管理器中配置串口延时。

```
Plain Text
打开设备管理器。 ...
在端口（COM 和 LPT）项中找到要使用的串口，右击后点击属性。 ...
在弹出的属性界面中点击上方的端口设置，再点击高级
在高级设置界面中将延迟计时器（毫秒）选项的数值修改为 1，点击右上角的 确定即可完成
```

ROS 环境下，为了保证数据传输的实时性，也需要设置串口为低延迟，保证更新率符合预期。

安装 setserial:

```
Shell
sudo apt install setserial
```

插入 usb 线后执行:

```
Shell
 setserial /dev/ttyUSB0 low_latency
```

ttyUSB0 换成实际使用的串口

1.9 CRC 查表法计算

建议直接参考示例代码。

注意 CRC 初始值设定为 1，参考 1.1.2.3 节编程方式

```
c++
// CRC32 计算表
static const uint32_t crc32_tab[] = {
    0x00000000, 0x77073096, 0xee0e612c, 0x990951ba, 0x076dc419, 0x706af48f,
0xe963a535, 0x9e6495a3,
    0x0edb8832, 0x79dcb8a4, 0xe0d5e91e, 0x97d2d988, 0x09b64c2b, 0x7eb17cbd,
0xe7b82d07, 0x90bf1d91,
    0x1db71064, 0x6ab020f2, 0xf3b97148, 0x84be41de, 0x1adad47d, 0x6ddde4eb,
0xf4d4b551, 0x83d385c7,
    0x136c9856, 0x646ba8c0, 0xfd62f97a, 0x8a65c9ec, 0x14015c4f, 0x63066cd9,
0xfa0f3d63, 0x8d080df5,
    0x3b6e20c8, 0x4c69105e, 0xd56041e4, 0xa2677172, 0x3c03e4d1, 0x4b04d447,
0xd20d85fd, 0xa50ab56b,
    0x35b5a8fa, 0x42b2986c, 0xdbbbc9d6, 0xacbcf940, 0x32d86ce3, 0x45df5c75,
0xdcd60dcf, 0xabd13d59,
    0x26d930ac, 0x51de003a, 0xc8d75180, 0xbfd06116, 0x21b4f4b5, 0x56b3c423,
0xcfba9599, 0xb8bda50f,
    0x2802b89e, 0x5f058808, 0xc60cd9b2, 0xb10be924, 0x2f6f7c87, 0x58684c11,
0xc1611dab, 0xb6662d3d,
    0x76dc4190, 0x01db7106, 0x98d220bc, 0xefd5102a, 0x71b18589, 0x06b6b51f,
0x9fbfe4a5, 0xe8b8d433,
    0x7807c9a2, 0x0f00f934, 0x9609a88e, 0xe10e9818, 0x7f6a0dbb, 0x086d3d2d,
0x91646c97, 0xe6635c01,
    0x6b6b51f4, 0x1c6c6162, 0x856530d8, 0xf262004e, 0x6c0695ed, 0x1b01a57b,
0x8208f4c1, 0xf50fc457,
    0x65b0d9c6, 0x12b7e950, 0x8bbeb8ea, 0xfcb9887c, 0x62dd1ddf, 0x15da2d49,
0x8cd37cf3, 0xfbd44c65,
    0x4db26158, 0x3ab551ce, 0xa3bc0074, 0xd4bb30e2, 0x4adfa541, 0x3dd895d7,
0xa4d1c46d, 0xd3d6f4fb,
    0x4369e96a, 0x346ed9fc, 0xad678846, 0xda60b8d0, 0x44042d73, 0x33031de5,
0xaa0a4c5f, 0xdd0d7cc9,
    0x5005713c, 0x270241aa, 0xbe0b1010, 0xc90c2086, 0x5768b525, 0x206f85b3,
0xb966d409, 0xce61e49f,
    0x5edef90e, 0x29d9c998, 0xb0d09822, 0xc7d7a8b4, 0x59b33d17, 0x2eb40d81,
0xb7bd5c3b, 0xc0ba6cad,
    0xedb88320, 0x9abfb3b6, 0x03b6e20c, 0x74b1d29a, 0xead54739, 0x9dd277af,
0x04db2615, 0x73dc1683,
    0xe3630b12, 0x94643b84, 0x0d6d6a3e, 0x7a6a5aa8, 0xe40ecf0b, 0x9309ff9d,
0x0a00ae27, 0x7d079eb1,
    0xf00f9344, 0x8708a3d2, 0x1e01f268, 0x6906c2fe, 0xf762575d, 0x806567cb,
0x196c3671, 0x6e6b06e7,
```

```
    0xfed41b76, 0x89d32be0, 0x10da7a5a, 0x67dd4acc, 0xf9b9df6f, 0x8ebeeff9,
0x17b7be43, 0x60b08ed5,
    0xd6d6a3e8, 0xa1d1937e, 0x38d8c2c4, 0x4fdff252, 0xd1bb67f1, 0xa6bc5767,
0x3fb506dd, 0x48b2364b,
    0xd80d2bda, 0xaf0a1b4c, 0x36034af6, 0x41047a60, 0xdf60efc3, 0xa867df55,
0x316e8eef, 0x4669be79,
    0xcb61b38c, 0xbc66831a, 0x256fd2a0, 0x5268e236, 0xcc0c7795, 0xbb0b4703,
0x220216b9, 0x5505262f,
    0xc5ba3bbe, 0xb2bd0b28, 0x2bb45a92, 0x5cb36a04, 0xc2d7ffa7, 0xb5d0cf31,
0x2cd99e8b, 0x5bdeae1d,
    0x9b64c2b0, 0xec63f226, 0x756aa39c, 0x026d930a, 0x9c0906a9, 0xeb0e363f,
0x72076785, 0x05005713,
    0x95bf4a82, 0xe2b87a14, 0x7bb12bae, 0x0cb61b38, 0x92d28e9b, 0xe5d5be0d,
0x7cdcefb7, 0x0bdbdf21,
    0x86d3d2d4, 0xf1d4e242, 0x68ddb3f8, 0x1fda836e, 0x81be16cd, 0xf6b9265b,
0x6fb077e1, 0x18b74777,
    0x88085ae6, 0xff0f6a70, 0x66063bca, 0x11010b5c, 0x8f659eff, 0xf862ae69,
0x616bffd3, 0x166ccf45,
    0xa00ae278, 0xd70dd2ee, 0x4e048354, 0x3903b3c2, 0xa7672661, 0xd06016f7,
0x4969474d, 0x3e6e77db,
    0xaed16a4a, 0xd9d65adc, 0x40df0b66, 0x37d83bf0, 0xa9bcae53, 0xdebb9ec5,
0x47b2cf7f, 0x30b5ffe9,
    0xbdbdf21c, 0xcabac28a, 0x53b39330, 0x24b4a3a6, 0xbad03605, 0xcdd70693,
0x54de5729, 0x23d967bf,
    0xb3667a2e, 0xc4614ab8, 0x5d681b02, 0x2a6f2b94, 0xb40bbe37, 0xc30c8ea1,
0x5a05df1b, 0x2d02ef8d
};

// CRC32 计算函数
uint32_t crc_crc32(uint32_t crc, const uint8_t *buf, uint32_t size) {
    for (uint32_t i = 0; i < size; i++) {
        crc = crc32_tab[(crc ^ buf[i]) & 0xff] ^ (crc >> 8);

    }

    return crc;
}
```

注 1 ：数据以小端格式传输，低字节在前，高字节在后

注 2：crc32 的初值为 1，CRC 计算不包括本身的本帧所有数据

2. SPI 通信协议

基于 STM32 的 SPI 主机读取驱动示例 ：

https://data.forsense-imu.com/ProductFile/link/Upper%20computer%20software/AHRS%20ROS2%20drive/download.php

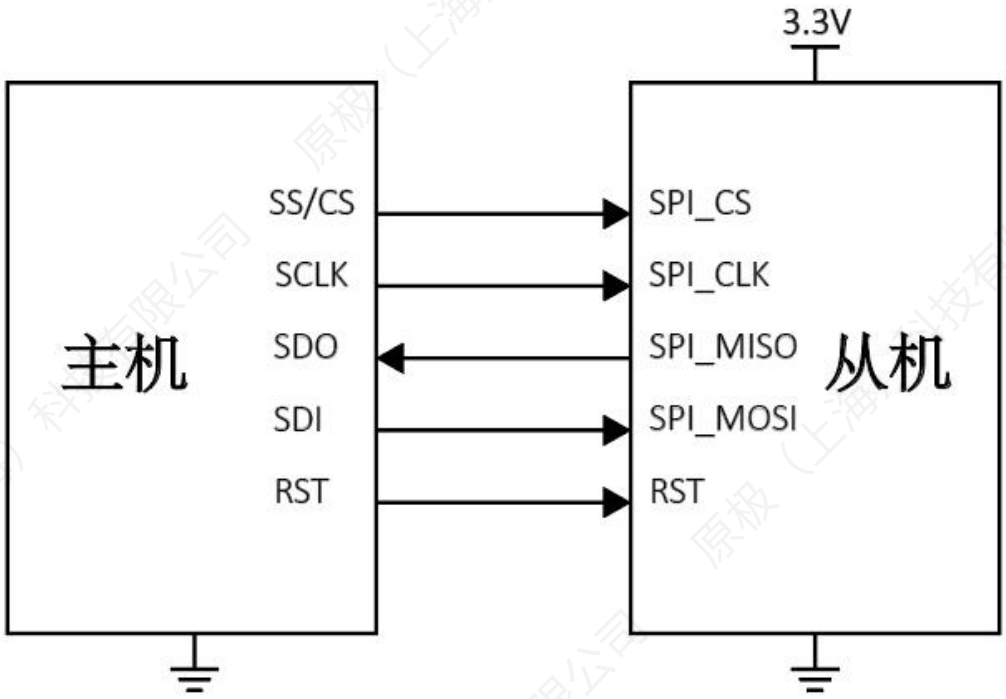
2.1 SPI 接口参数

SPI 接口参数

SPI 主机	本产品作为从机
SPI 速率	0.2~2MHz
SPI 字长	16bit
相位	上升沿触发（模式 3，CPHA=1）
极性	空闲为高电平（模式 3，CPOL=1）
位序	MSB 优先

2.2 SPI 连接示意图

SPI 连线示意图



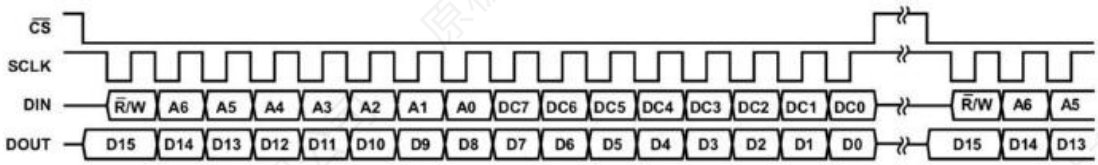
注 1：初始化读取前，需将 IMU 复位并等待 3s，使得 IMU 进入正常工作状态。

注 2：不同 IMU 型号的 SPI 引脚参考对应手册

2.3 SPI 通信位序

SPI 接口支持全双工串行通信（同时执行发送和接收），采用下图所示的位序。

SPI 通信位序示意图



其中，DIN 最高位表示读/写操作，[A6:A0]表示寄存器地址，[DC7:DC0]表示写入的数据（写操作）或 DUMMY 数据（读操作）。

当/W =1 时，此 SPI 周期的 DOUT 数据无意义。

当/W =0 时，此 SPI 周期的 DOUT 数据表示上两个周期的寄存器输出数据，具体见 BURST 读取示例。

2.4 SPI 寄存器

2.4.1 SPI 寄存器列表

SPI 寄存器列表

名称	地址	读/写	默认值	窗 ID	描述
BURST	0x00	RW		0	连续读取
FILTER_CTRL	0x07,0x06	RW	0x00BB	1	滤波器选择
PROD_ID1	0x6C	R	0x494d	1	ID 号 1
PROD_ID2	0x6E	R	0x5536	1	ID 号 2
WIN_CTRL	0x7F,0x7E	RW	0x0000	0，1	窗 ID 选择
TEMP_HIGH	0x0E	R	\	0	温度高字节
TEMP_LOW	0x10	R	\	0	温度低字节
XGYRO_HIGH	0x12	R	\	0	陀螺 X 轴高字节
XGYRO_LOW	0x14	R	\	0	陀螺 X 轴低字节
YGYRO_HIGH	0x16	R	\	0	陀螺 Y 轴高字节
YGYRO_LOW	0x18	R	\	0	陀螺 Y 轴低字节
ZGYRO_HIGH	0x1A	R	\	0	陀螺 Z 轴高字节
ZGYRO_LOW	0x1C	R	\	0	陀螺 Z 轴低字节
XACCEL_HIGH	0x1E	R	\	0	加表 X 轴高字节
XACCEL_LOW	0x20	R	\	0	加表 X 轴低字节

YACCEL_HIGH	0x22	R	\	0	加表 Y 轴高字节
YACCEL_LOW	0x24	R	\	0	加表 Y 轴低字节
ZACCEL_HIGH	0x26	R	\	0	加表 Z 轴高字节
ZACCEL_LOW	0x28	R	\	0	加表 Z 轴低字节
ROLL_HIGH	0x2A	R	\	0	横滚角高字节
ROLL_LOW	0x2C	R	\	0	横滚角低字节
PITCH_HIGH	0x2E	R	\	0	俯仰角高字节
PITCH_LOW	0x30	R	\	0	俯仰角低字节
YAW_HIGH	0x32	R	\	0	航向角高字节
YAW_LOW	0x34	R	\	0	航向角低字节
STATUS_H	0x36	R	\	0	状态高字节
STATUS_L	0x38	R	\	0	状态低字节
ODR_HIGH	0x41	R\W	\	1	ODR 寄存器高字节
ODR_LOW	0x40	R\W	\	1	ODR 寄存器低字节

STATUS_L 寄存器说明：

位	名称	描述
[0]	保留	
[1]	保留	
[2]	校准扇区故障码	0：故障　　1：无故障
[3]	配置扇区故障码	0：故障　　1：无故障
[4]	陀螺仪静止	0：非静止　　1：静止
[5]-[6]	保留	
[7]	陀螺仪超量程	0:超量程　　1：没超
[8]-[15]	保留	

STATUS_H 寄存器说明：

位	名称	描述
[0]-[15]	保留	

2.4.2 SPI BURST 寄存器（连续读取寄存器）

BURST 为连续读取寄存器，在一个数据流中读取所有数据，各 16 位段之间无停转。

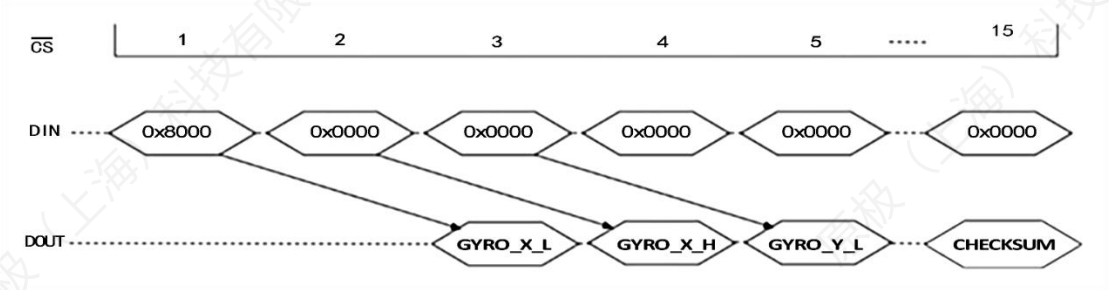
SPI BURST 寄存器格式

地址	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	读/写

0x01									RW
地址	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	读/写
0x00	BURST_CMD								RW

BURST 读取方法是：读取前发送 0x8000 表示设置 BURST 并开始读取，然后一直发送 0x0000 并接收数据，输出寄存器内容比读取指令发送偏移 2 个 SPI 周期，读取期间一直持续片选低电平。

SPI BURST 连续读取示意图



2.4.2.1 BURST

要输出模式 1 数据帧需要先往寄存器 0x8D 中写 0x00

SPI BURST 连续读取基本格式

发送顺序	1	2	3	4	5	6
发送内容	GYRO_X_L	GYRO_X_H	GYRO_Y_L	GYRO_Y_H	GYRO_Z_L	GYRO_Z_H
发送顺序	7	8	9	10	11	12
发送内容	ACCL_X_L	ACCL_X_H	ACCL_Y_L	ACCL_Y_H	ACCL_Z_L	ACCL_Z_H
发送顺序	13					
发送内容	CHKSM					

2.4.2.2 BURST2

SPI BURST2 连续读取基本格式

要输出模式 2 数据帧需要先往寄存器 0x8D 中写 0x02

发送顺序	1	2	3	4	5	6
发送内容	GYRO_X_L	GYRO_X_H	GYRO_Y_L	GYRO_Y_H	GYRO_Z_L	GYRO_Z_H
发送顺序	7	8	9	10	11	12
发送内容	ACCL_X_L	ACCL_X_H	ACCL_Y_L	ACCL_Y_H	ACCL_Z_L	ACCL_Z_H
发送顺序	13	14	15	16	17	18
发送内容	TEMP_L	TEMP_H	PITCH_L	PITCH_H	ROLL_L	ROLL_H

发送顺序	19	20	21	22	23	24
发送内容	YAW_L	YAW_H	STATUS_L	STATUS_H	CHECK_L	CHECK_H

2.4.2.3 BURST3

SPI BURST3 连续读取基本格式

发送顺序	1	2	3	4	5	6
发送内容	TIME	STATUS	GYRO_X_L	GYRO_X_H	GYRO_Y_L	GYRO_Y_H
发送顺序	7	8	9	10	11	12
发送内容	GYRO_Z_L	GYRO_Z_H	ACCL_X_L	ACCL_X_H	ACCL_Y_L	ACCL_Y_H
发送顺序	13	14	15	16	17	18
发送内容	ACCL_Z_L	ACCL_Z_H	PITCH_L	PITCH_H	ROLL_L	ROLL_H
发送顺序	19	20	21	22	23	24
发送内容	YAW_L	YAW_H	TEMP_L	TEMP_H	Q0_L	Q0_H
发送顺序	25	26	27	28	29	30
发送内容	Q1_L	Q1_H	Q2_L	Q2_H	Q3_L	Q3_H
发送顺序	31	32				
发送内容	CHECK_L	CHECK_H				

要输出模式 3 数据帧需要先往寄存器 0x8D 中写 0x03

BURST3 的说明：

注 1：BURST,BURST2,BURST3 中所有数据均为 16-bit 宽度

注 2：BURST,BURST2,BURST3 中的陀螺，加速度计，姿态，四元数数据拼接后格式表示为 int32

注 3：所 CHKSM 即 CHECKSUM，用于确认数据完整性。计算方法为将 CHECKSUM 之前的所有数据累加求和

2.4.3 SPI 读取数据后的还原关系说明

在 BURST 连续读取过程中，32 位的完整数据被拆分成高 16 位和低 16 位分别输出，输出时采用小端模式，即低字节先输出。用户需要将这两部分 16 位数据首尾拼接，还原出完整的 32 位数据。

SPI32 位数据还原示意图

得到完整的 32 位数据后，标准帧用户可根据以下公式将其转换为角速度、加速度、温度和姿态角信息。

名称	单位	公式	条件/备注
角速度	°/s	G=SF/65536*GYRO	GYRO 为上表中 X/Y/Z 轴的 GYRO 数据 陀螺刻度因子 SF= 0.016
加速度	mg	A=SF/65536*ACCL	ACCL 为上表中 X/Y/Z 轴的 ACCL 数据 Burst 模式时，SF = 0.2(单位 g) 单寄存器模式时，SF=0.2/1000（单位 mg）
温度	℃	T=SF/65536*(TEMP-172621824)+25	TEMP 为上表中的 TEMP 数据 温度刻度因子 SF=-1/263.4
姿态角	°	D=SF/65536*ATT	ATT 为上表中 ATT 数据 姿态刻度因子 SF = 0.00699411

标准帧 SPI 32 位数据转换公式

注 1 ：此表中的转换关系为标准帧（直接读取寄存器）使用，Burst，Burst2，Burst3 模式无需转换，直接为 float 类型。

2.4.4 SPI FILTER_CTRL 寄存器

FILTER_CTRL 寄存器为用户提供对数字低通滤波器的控制。 此寄存器为可读/写寄存器，写命令为发送 0x86XX，且当前 SPI 周期设置有效；读命令发送 0x0600，输出寄存器内容比读取指令发送偏移 2 个 SPI 周期。

SPI FILTER_CTRL 寄存器格式

地址	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	读/写
0x07									RW
地址	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	读/写
0x06	加速度计滤波器配置				陀螺仪滤波器配置				RW

滤波器配置同 1.4 节

	编码	描述
加速度计/ 陀螺仪滤波器配置	4'b 0000	IIR filter fc=1 Hz
	4'b 0001	IIR filter fc=1 Hz
	4'b 0010	IIR filter fc=2 Hz
	4'b 0011	IIR filter fc=5 Hz
	4'b 0100	IIR filter fc=10 Hz
	4'b 0101	IIR filter fc=15 Hz
	4'b 0110	IIR filter fc=20 Hz
	4'b 0111	IIR filter fc=25 Hz
	4'b 1000	IIR filter fc=30 Hz
	4'b 1001	IIR filter fc=35 Hz
	4'b 1010	IIR filter fc=40 Hz
	4'b 1011	no filter

注：比如配置陀螺、加速度计滤波器为 10Hz，则写入 0x8644 值。

低通滤波值和 AT 指令对应数值

序号	IMU 低通滤波值	AT 指令对应的值
1	1	17
2	2	34
3	5	51
4	10	68
4	15	85
5	20	102
6	25	119
7	30	136
8	35	153
9	40	170
10	No filter	187

2.4.5 SPI WIN_CTRL 寄存器

此寄存器用于控制切换窗口 ID，可读可写。窗口默认为 0，写入 0xFE01，则切换为 1。

地址	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	读/写
0x7F									RW
地址	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	读/写
0x7E	WINDOW_ID								RW

SPI WIN_CTRL 寄存器格式

SPI 寄存器 WIN_CTRL.WINDOW_ID 编码

名称	编码	描述
WINDOW_ID	0x00	window0，开始读取数据
	0x01	window1，进入配置

2.4.6 SPI ODR 寄存器

此寄存器用于设置 DRDY 引脚的输出频率，默认输出频率 100Hz,配置后若断电重启，则配置失效。

可读可写,ODR 默认为 100，写入 0x03E8，则切换为 1000Hz。

写入方式示例

```
C++
SPI1_ReadWriteByte16(0xC0E8);//写低地址寄存器 0x40
SPI1_ReadWriteByte16(0xC103);//写高地址寄存器 0x41
odr = SPI1_ReadReg(0x4000);
odr |= SPI1_ReadReg(0x4100)<<8;
/**
 * SPI 读取单个寄存器
 */
uint16_t SPI1_ReadReg(uint16_t addr)
{
	SPI1_ReadWriteByte16(addr);
	SPI1_ReadWriteByte16(DUMMY);
	return SPI1_ReadWriteByte16(DUMMY);
}
/**
 * SPI 双向传输数据
 */
uint16_t SPI1_ReadWriteByte16(uint16_t cmd)
{
	uint16_t rev;
	HAL_SPI_TransmitReceive(&hspi1,(uint8_t* )&cmd,(uint8_t* )&rev,1,0xFFFFFF);

	return rev;
}
```

地址	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	读/写
0x41									RW
地址	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	读/写
0x40	ODR								RW

3. I²C 通信协议

基于 STM32 的 I²C 主机读取驱动示例 ：

https://data.forsense-imu.com/page/download.html

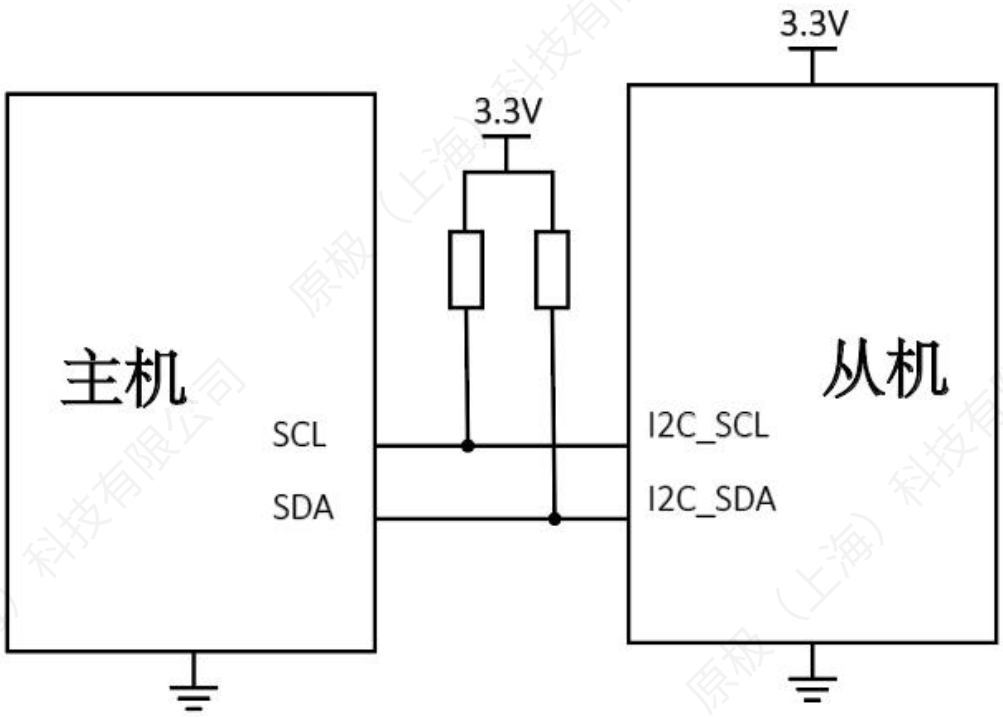
3.1 I²C 接口参数

I²C 接口参数

I2C 速率	400KHz
I2C 从机地址（7 位）	0x18

3.2 I²C 连接方式

I²C 连接方法



注：上拉电阻阻值为 4.7KΩ

3.3 I²C 寄存器

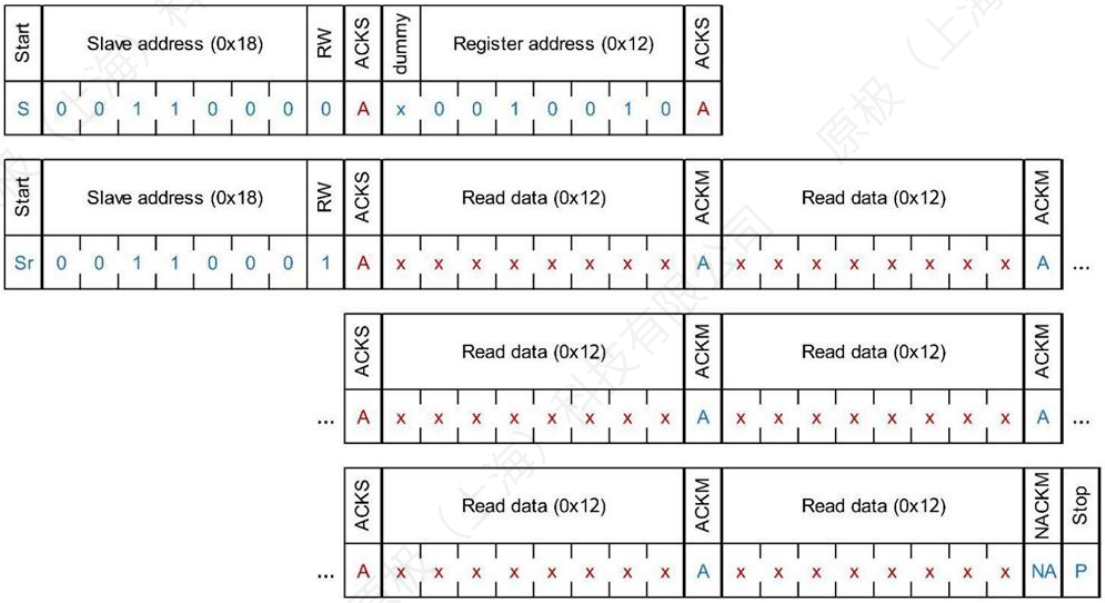
I²C 寄存器列表

名称	地址	读/写	默认值	描述
BURST	0x12	R		连续读取寄存器
FILTER_CTRL	0x06	RW	0xBB	滤波器选择
PROD_ID	0x6A	R		产品名称

3.3.1 I²C BURST 寄存器

本 I²C 协议支持连续读取，连续读寄存器地址 0x12，从机自动累加地址，以 8bit 模式连续输出 48 个字节，读取过程如下：

I²C 连续读取模式



帧定义如下：

I²C 连续读取数据格式

发送顺序	1	2	3
数据格式	uint32_t	float	float
发送内容	TIME	ACCL_X	ACCL_Y
发送顺序	4	5	6
数据格式	float	float	float
发送内容	ACCL_Z	GYRO_X	GYRO_Y
发送顺序	7	8	9
数据格式	float	float	float
发送内容	GYRO_Z	TEMP	ROLL
发送顺序	10	11	12

数据格式	float	float	uint32
发送内容	PITCH	YAW	CRC32

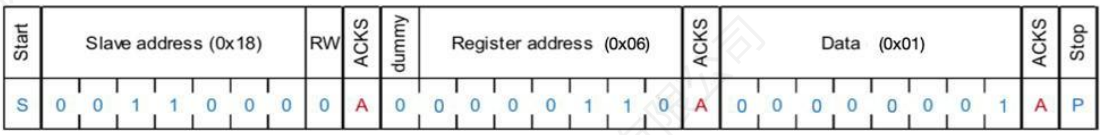
注 1：TEMP 单位为℃，陀螺仪输出单位为°/s，加速度计输出单位为 g，姿态输出单位为度

注 2：crc32 的初值为 1，CRC 计算不包括本身的本帧所有数据，查表计算法见附录 1

3.3.2 I²C FILTER_CTRL 寄存器

FILTER_CTRL 寄存器地址为 0x06，滤波器配置对照表同 1.4 节加速度计和陀螺仪滤波器配置。寄存器读取过程同 I²C BURST 读取方法，写寄存器过程如下图所示。

I²C FILTER_CTRL 寄存器写入方法



3.3.3 I²C ID 寄存器

ID 寄存器地址为 0x6A，数据内容为 ASCII 编码形式的字符“IMU61B”，读取过程同 I²C BURST，如下表所示。

I²C ID 寄存器读取模式

发送顺序	1	2	3	4
发送内容	0x00	0x00	0x49	0x4D
发送顺序	5	6	7	8
发送内容	0x55	0x36	0x31	0x*

注 1：所有数据均为 8-bit 宽度

注 2：0x* 表示的内容为产品 ID,0x32 代表 IMU612,0x34 代表 IMU614,0x38 代表 IMU618,0x41 代表 IMU6132A,0x42 代表 IMU6132B

4. CAN 通信协议

基于 STM32 的 CAN 主机读取驱动示例 ：

https://data.forsense-imu.com/ProductFile/link/Upper%20computer%20software/AHRS%20ROS2%20drive/download.php

各类产品的 DBC 文件详见官网下载页面
https://data.forsense-imu.com/page/download.html

4.1 CAN 默认配置

CAN 配置	值
波特率	250~1000Kbps
节点 ID	0
输出频率	1~200Hz
帧类型	标准帧
大小端	小端模式(intel)

4.2 标准帧格式

标准帧 ID（十进制）	1	2	3	4	5	6	7	8
101+节点 ID	ROLL				PITCH			
102+节点 ID	YAW				Gx			
103+节点 ID	Gy				Gz			
104+节点 ID	Ax				Ay			
105+节点 ID	Az				TEMP		status	index

注 1：姿态角、陀螺、加速度计数据表示为 float，温度表示为 int16,status 和 index 表示为 uint8

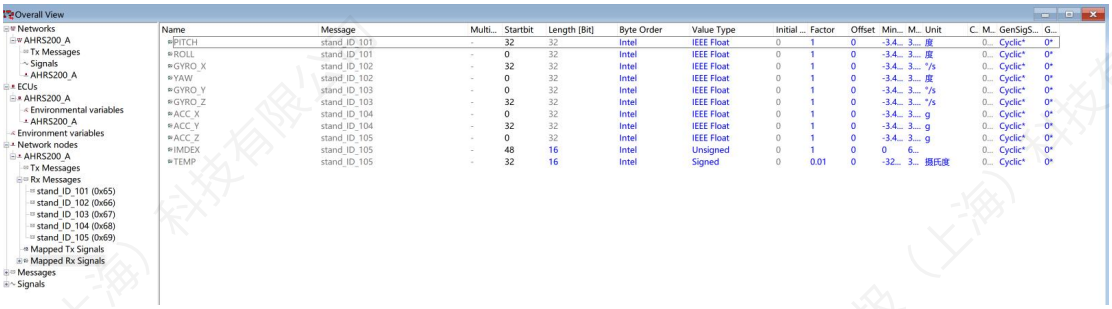
注 2：TEMP 单位为 100*℃，陀螺仪输出单位为°/s ，加速度计输出单位为 g，姿态输出单位为度

status 字节说明：

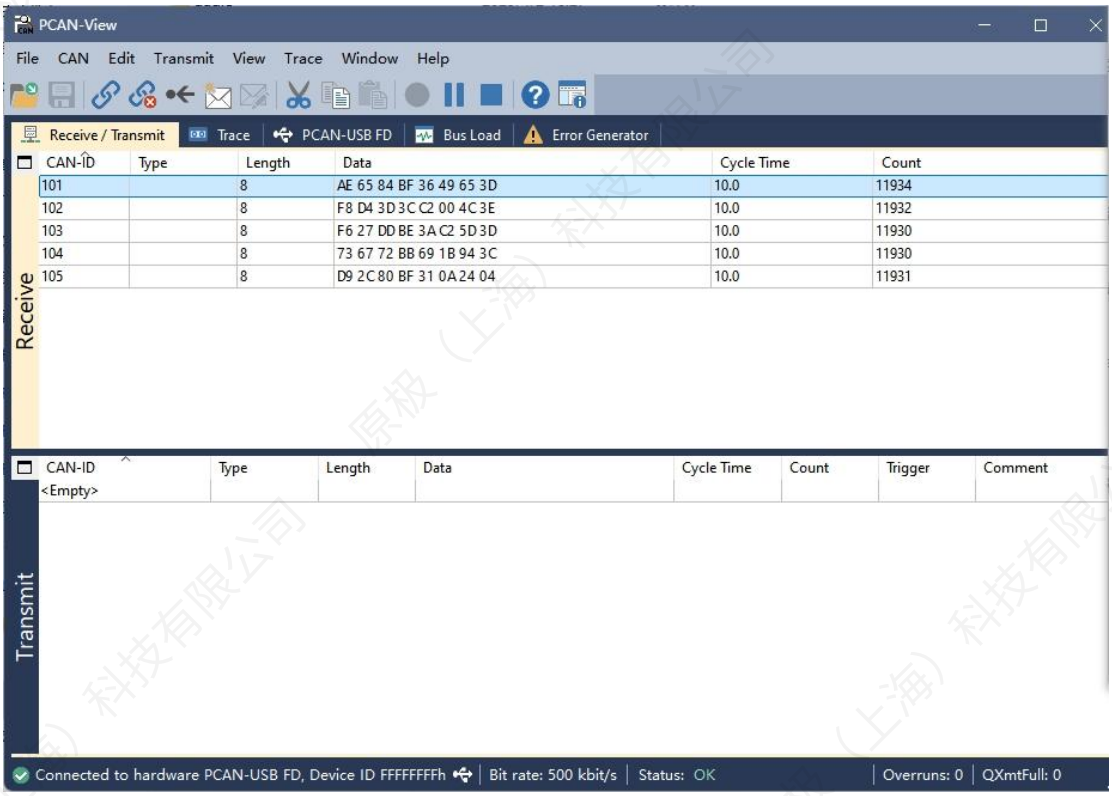
位	名称	描述
[0]	保留	
[1]	保留	
[2]	校准扇区故障码	0：故障　　1：无故障

[3]	配置扇区故障码	0：故障	1：无故障
[4]	陀螺仪静止	0：非静止	1：静止
[5]-[6]	保留		
[7]	陀螺仪超量程	0:超量程	1：没超

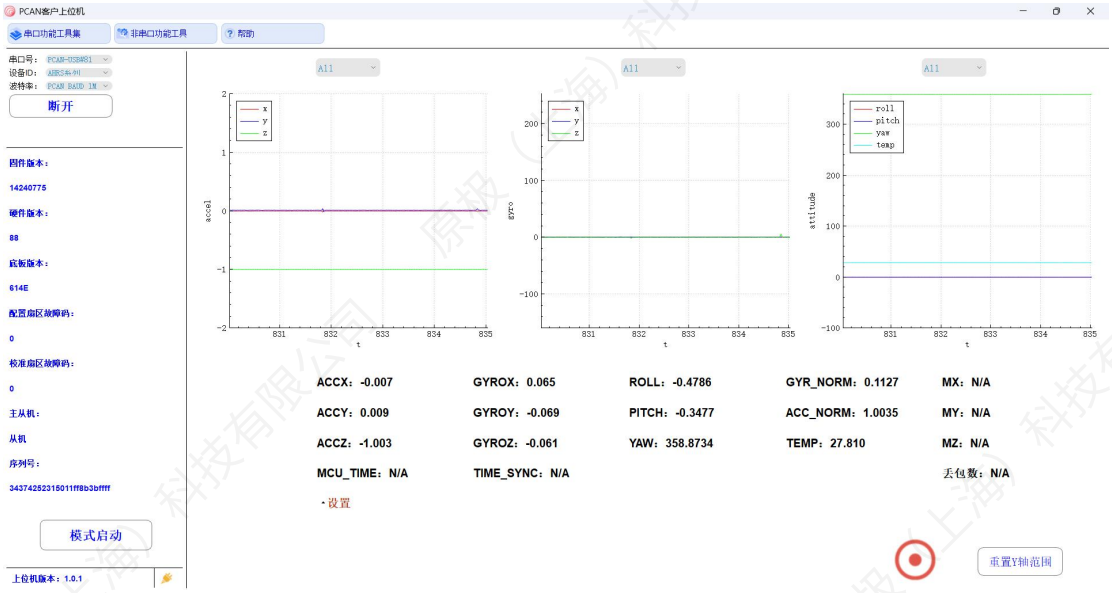
用 DBC 文件解析的：



用 PCAN-View 查看数据:



用 PCAN 上位机可以解析成 AHRS 数据：



4.3 CAN 参数配置

4.3.1 配置 CAN 波特率

配置 CAN 波特率，发送指令：

ID=0x619，DATA=0x20 0x21 0x22 0x23 0xXX 0x00 ID_H ID_L

IMU 应答如下：

ID=0x519，DATA=0xXX 0xFF 0xFF 0xFF 0xFF 0xFF ID_H ID_L

查询 CAN 波特率，发送指令：

ID=0x619，DATA=0x20 0x21 0x22 0x23 0x0A 0x00 ID_H ID_L

IMU 应答如下：

ID=0x519，DATA= 0xXX 0x0A 0xFF 0xFF 0xFF 0xFF ID_H ID_L

其中：

XX=01　　波特率为 250Kbps

XX=02　　波特率为 500Kbps

XX=03　　波特率为 1000Kbps

ID_H　　为节点 ID 的高字节

ID_L　　为节点 ID 的低字节

4.3.2 配置节点 ID

节点 ID 计算方式：期望 ID - 0x65 = 发送 ID

例配置节点 ID 为 0x403

需要发送的 ID 为 0x403-0x65=0x039E

即发送 ID=0x61A，DATA=0x30 0x31 0x32 0x33 0x03 0x9E 0x00 0x00

此配置选项立即生效，配置时会保存 flash，3s 内只响应一次指令，不能短时间内连续发送配置命令。

设置节点 ID 为 0X0167，发送指令：

ID=0x61A，DATA=0x30 0x31 0x32 0x33 0x01 0x02 0x00 0x00

IMU 应答如下：

ID=0x51A，DATA=0x01 0x02 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF

4.3.3 查询版本号

发送指令：

ID=0x618，DATA=0x10 0x11 0x12 0x13 0x00 0x00 0x00 0x00

IMU 应答如下：

ID=0x518，DATA=0x00 0xC9 0xBB 0xE9 0xFF 0xFF ID_H ID_L

版本号为 0x00C9BBE9，即固件版本为 13220841

ID_H 为节点 ID 的高字节

ID_L 为节点 ID 的低字节

4.3.4 查/设置终端电阻

去掉终端电阻，发送指令：

ID=0x61B，DATA=0x10 0x11 0x12 0x13 0x01 0xFF 0xFF 0xFF

IMU 应答如下：

ID=0x51B，DATA=0x01 0xFF 0xFF 0xFF 0xFF 0xFF ID_H ID_L

添加终端电阻，发送指令：

ID=0x61B，DATA=0x10 0x11 0x12 0x13 0x02 0xFF 0xFF 0xFF

IMU 应答如下：

ID=0x51B，DATA=0x02 0xFF 0xFF 0xFF 0xFF 0xFF ID_H ID_L

查询终端电阻

ID=0x61B，DATA=0x10 0x11 0x12 0x13 0x0A 0xFF 0xFF 0xFF

ID=0x51B，DATA=0xXX 0x0A 0xFF 0xFF 0xFF 0xFF ID_H ID_L

其中 XXXX=0x01 无终端电阻

XXXX=0x02 有终端电阻

ID_H 为节点 ID 的高字节

ID_L 为节点 ID 的低字节

4.3.5 设置输出频率

设置输出频率，发送指令：

ID=0x61C，DATA=0x10 0x11 0x12 0x13 0xXX 0xFF ID_H ID_L

IMU 应答如下：

ID=0x51C，DATA=0xXX 0xFF 0xFF 0xFF 0xFF 0xFF ID_H ID_L

查询输出频率，发送指令：

ID=0x61C，DATA=0x10 0x11 0x12 0x13 0x0A 0xFF ID_H ID_L

IMU 应答如下：

ID=0x51C，DATA=0xXX 0x0A 0xFF 0xFF 0xFF 0xFF ID_H ID_L

其中：

XX=01　输出频率为 1Hz

XX=02　输出频率为 10Hz

XX=03　输出频率为 50Hz

XX=04　输出频率为 100Hz

XX=05　输出频率为 200Hz

XX=06　输出频率为 20Hz

ID_H 为节点 ID 的高字节

ID_L 为节点 ID 的低字节

4.3.6 查/设置横滚俯仰取反

设置横滚俯仰取反，发送指令：

ID=0X61D,DATA=0X10 0X11 0X12 0X13 XXXX 0XFF 0XFF 0XFF

IMU 应答如下：

ID=0X51D,DATA=XXXX 0XFF 0XFF 0XFF 0XFF 0XFF ID_H ID_L

查询横滚俯仰取反状态，发送指令：

ID=0X61D,DATA=0X10 0X11 0X12 0X13 0X0A 0XFF 0XFF 0XFF

IMU 应答如下：

ID=0X51D,DATA=XXXX 0X0A 0XFF 0XFF 0XFF 0XFF ID_H ID_L

其中：

XXXX=0X00 横滚角与俯仰角均不变

XXXX=0X01 横滚角取反，俯仰角不变

XXXX=0X10 横滚角不变，俯仰角取反

XXXX=0X11 横滚角与俯仰角均取反

ID_H 为节点 ID 的高字节

ID_L 为节点 ID 的低字节

4.3.7 查/设置滤波器截止频率

设置滤波器截止频率，发送指令：

ID=0X61E,DATA=0X20 0X21 0X22 0X23 XXXX 0XFF 0XFF 0XFF

IMU 应答如下：

ID=0X51E,DATA=XXXX 0XFF 0XFF 0XFF 0XFF 0XFF ID_H ID_L

查询滤波器截止频率状态，发送指令：

ID=0X61E,DATA=0X20 0X21 0X22 0X0A 0XFF 0XFF 0XFF 0XFF

IMU 应答如下：

ID=0X51E,DATA=XXXX 0X0A 0XFF 0XFF 0XFF 0XFF ID_H ID_L

其中：

XXXX=0X44 截止频率 10Hz

XXXX=0X66 截止频率 20Hz

XXXX=0XAA 截止频率 40Hz

XXXX=0XBB 截止频率 no filter

ID_H 为节点 ID 的高字节

ID_L 为节点 ID 的低字节

4.3.8 查/设置坐标系

设置坐标系，发送指令：

ID=61F,DATA=0X30 0X31 0X32 0X33 XXXX 0XFF 0XFF 0XFF

IMU 应答如下：

ID=0x51F,DATA= XXXX 0xFF 0xFF 0xFF 0xFF 0xFF ID_H ID_L

查询朝向设置，发送指令：

ID=0x61F，DATA=0x30 0x31 0x32 0x0A 0xFF 0xFF 0xFF 0xFF

IMU 应答如下：

ID=0x51F，DATA=XXXX 0x0A 0xFF 0xFF 0xFF 0xFF ID_H ID_L

其中：

XXXX=0X65 默认朝向（101 坐标系）

ID_H 为节点 ID 的高字节

ID_L 为节点 ID 的低字节

4.3.9 关闭/扣除姿态角

设置关闭扣除姿态角或扣除姿态角，发送指令：

ID=0x620，DATA=0x10 0x11 0x12 0x13 XXXX 0xFF 0xFF 0xFF

IMU 应答如下：

ID=0x520，DATA=XXXX 0xFF 0xFF 0xFF 0xFF 0xFF ID_H ID_L

查询是否扣除姿态角设置，发送指令：

ID=0x620，DATA=0x10 0x11 0x12 0x0A 0xFF 0xFF 0xFF 0xFF

IMU 应答如下：

ID=0x520，DATA=XXXX 0x0A 0xFF 0xFF 0xFF 0xFF ID_H ID_L

其中：

XXXX=0X01 扣除姿态角

XXXX=0X00 没有扣除姿态角

ID_H 为节点 ID 的高字节

ID_L 为节点 ID 的低字节

4.3.10 保存指令

发送指令：

ID=0x6FF，DATA=0x10 0x11 0x12 0x13 0xFF 0xFF 0xFF 0xFF

IMU 响应如下：

ID=0x5FF，DATA=0xFF 0xFF 0xFF 0xFF 0xFF 0xFF ID_H ID_L

保存需要时间，保存成功后才返回报文，保存成功之后重启生效

ID_H 为节点 ID 的高字节

ID_L 为节点 ID 的低字节

4.3.11 SN 号查询

查询 SN 号高 6 字节：

例：

ID=0x621，DATA=0x10 0x11 0x12 0x01 0xFF 0xFF ID_H ID_L

ID=0x521，DATA=0x30 0x31 0x32 0x33 0x11 0x12 ID_H ID_L

查询 SN 号低 6 字节：

例：

ID=0x622，DATA=0x10 0x11 0x12 0x01 0xFF 0xFF ID_H ID_L

ID=0x522，DATA=0x15 0x16 0x17 0x18 0x19 0x20 ID_H ID_L

经过组合 SN 号：0x303132331112151617181920

ID_H 为节点 ID 的高字节

ID_L 为节点 ID 的低字节

4.3.12 设置航向角

设置航向角，发送指令：

ID=623,DATA=0X30 0X31 0X32 0X33 XXXX XXXX 0XFF 0XFF

IMU 应答如下：

ID=0x523,DATA= XXXX XXXX 0xFF 0xFF 0xFF 0xFF ID_H ID_L

其中：

XXXX XXXX=0X0000 表示设置航向为 0，如果=0x1234，表示设置航向为 46.6

ID_H 为节点 ID 的高字节

ID_L 为节点 ID 的低字节

4.3.13 安装偏差角标定

IMU 发送如下：

ID=624,DATA=0X30 0X31 0X32 0X33 XXXX 0xFF ID_H ID_L

IMU 应答如下：

ID=0x524,DATA= XXXX 0xFF 0xFF 0xFF 0xFF 0xFF ID_H ID_L

其中：

XXXX =0x01 清空历史标定

XXXX =0x02 开始标定流程

XXXX =0x03 开始数据采集

发送：XXXX =0x0A 查询标定状态　　应答：XXXX 标定状态，见《表：标定状态定义》

ID_H 为节点 ID 的高字节

ID_L 为节点 ID 的低字节

注：当进行标定过程中，每当一个轴标定成功后，IMU 会主动发送一帧 can 报文，用于告知当前标定状态，

发送内容为：ID=0x524,DATA= 0x02 XXXX 0xFF 0xFF 0xFF 0xFF ID_H ID_L

XXXX 表示标定状态,标定完成 XXXX=0x3F，见《表：标定状态定义》

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0	0	1	1	1	1	1	1
		-Z 标定完成	+Z 标定完成	+Y 标定完成	-Y 标定完成	-X 标定完成	+X 标定完成

表：标定状态定义

4.3.14 查/设置出厂编号

- 设置出厂编号

IMU 发送如下：

ID=0x626,DATA=0X20 0X21 0X10 0X1A 0xA8 0x8D ID_H ID_L

IMU 应答如下：

ID=0x526,DATA=0X10 0X1A 0xA8 0x8D 0xFF 0xFF ID_H ID_L

SN：0X10 0X1A 0xA8 0x8D

范围：0x101AA88D~0x16885901

- 查询出厂编号

ID=0x626,DATA=0X10 0X11 0X12 0X13 0x0A 0xFF ID_H ID_L

IMU 应答如下：

ID=0x526,DATA=0X10 0X1A 0xA8 0x8D 0x0A 0xFF ID_H ID_L

ID_H 为节点 ID 的高字节

ID_L 为节点 ID 的低字节

4.3.15 查/设置出厂标定状态

- 设置出厂标定状态

IMU 发送如下：

ID=0x627,DATA=0X10 0X11 0X12 0X13 0xXX 0xFF ID_H ID_L

IMU 应答如下：

ID=0x527,DATA=0XXX 0XFF 0xFF 0xFF 0xFF 0xFF ID_H ID_L

1. 查询出厂标定状态

IMU 发送如下：

ID=0x627,DATA=0X10 0X11 0X12 0X13 0x0A 0xFF ID_H ID_L

IMU 应答如下：

ID=0x527,DATA=0XXX 0X0A 0xFF 0xFF 0xFF 0xFF ID_H ID_L

ID_H 为节点 ID 的高字节

ID_L 为节点 ID 的低字节

5. 时间同步

通过时间同步，可以确保设备的内部时钟与外部时间基准保持一致，这样可以消除由于时钟漂移导致的时间偏差;

在多设备协同工作的系统中，所有设备的时间戳都会基于同一个时间基准，这有助于保证数据的一致性和准确性。

提供四种时间同步方法：

1. PPS 校准 IMU 内部时间戳
2. PPS 外部触发 IMU 数据输出
3. PPS 信号+RMC 报文

- 通过 DRDY 信号信号在主机端做时间同步

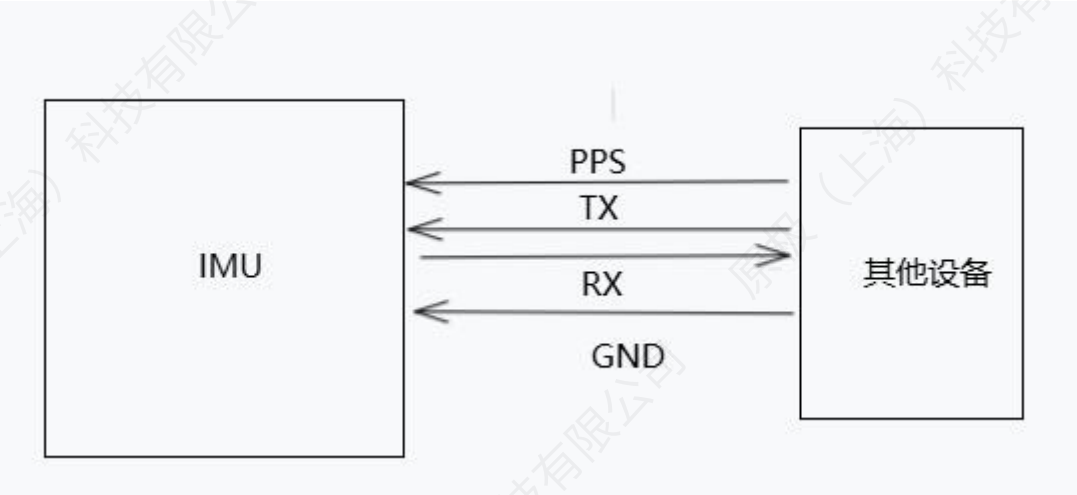
5.1 PPS 校准 IMU 内部时间戳

RTK 输出 1Hz 的 PPS,IMU 收到 PPS 后会对自身时间进行整秒对齐。输出的数据中时间戳为整秒对齐后的时间。

所需条件

- IMU 主动输出的频率为**非 0HZ**
- **同步模式 1　发送指令　AT+SET_TIME_SYNC=1\r\n**

1. PPS 秒脉冲：1s 一次，上升沿触发，脉宽 5ms。高电平不得高于 5V（特殊型号除外）。



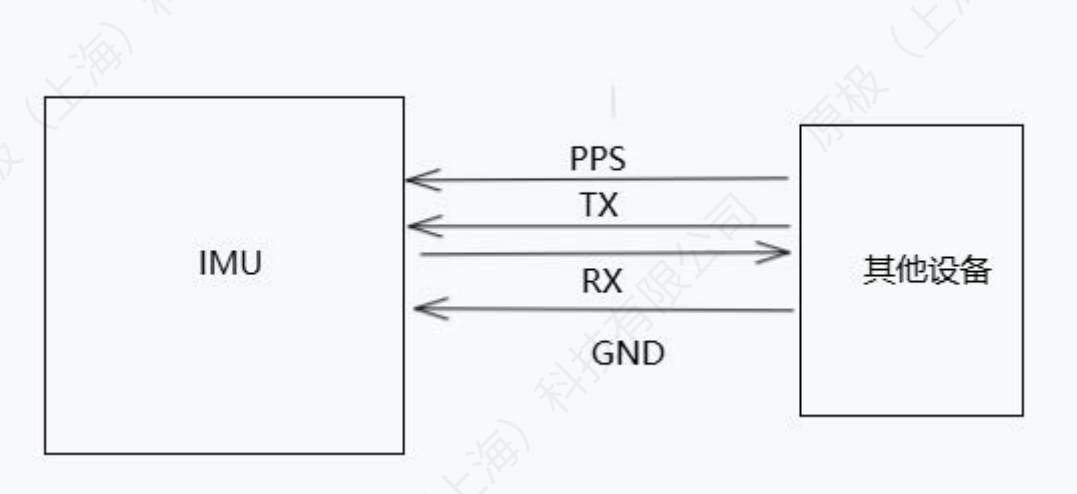
5.2 PPS 外部触发 IMU 数据输出

PPS 触发 IMU 输出数据，IMU 检测到 PPS 的**上升沿**后输出数据。主机发送 PPS 的时间，即可当成 IMU 输出数据的时间。

所需条件

2. IMU 主动输出的频率为 **0HZ**

- **同步模式 1　发送指令　AT+SET_TIME_SYNC=1\r\n**
- PPS 多少 HZ，IMU 输出就是多少 HZ



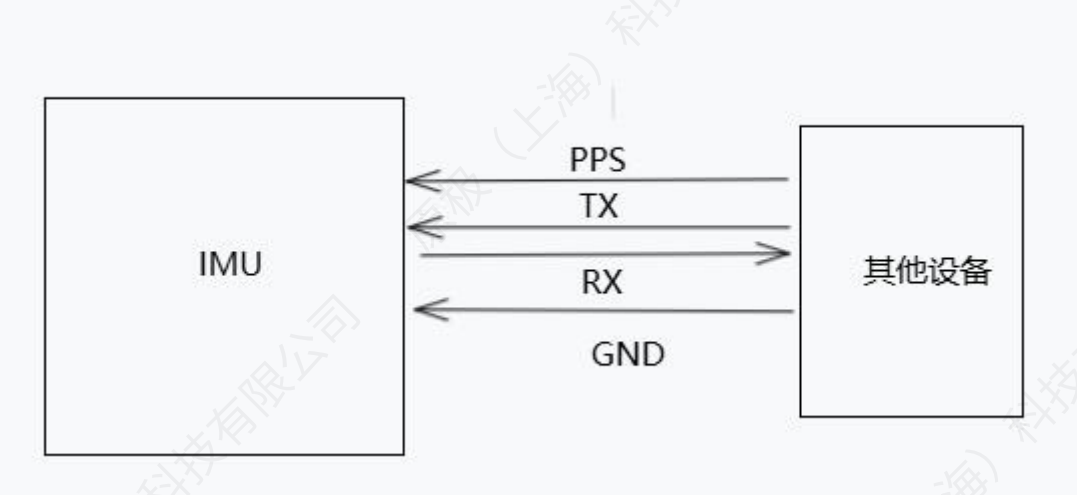
5.3 PPS 信号+RMC 报文

RTK 输出 1Hz 的 PPS,IMU 收到 PPS 后会对自身时间进行整秒对齐。输出的数据中时间戳为整秒对齐后的时间。

所需条件

1. IMU 主动输出的频率为**非 0HZ**

- **同步模式 1　发送指令　AT+SET_TIME_SYNC=1\r\n**
- PPS 秒脉冲：1s 一次，上升沿触发，脉宽 5ms。高电平不得高于 5V（特殊型号除外）。
- 需要给 IMU 灌入 GPRMC 报文

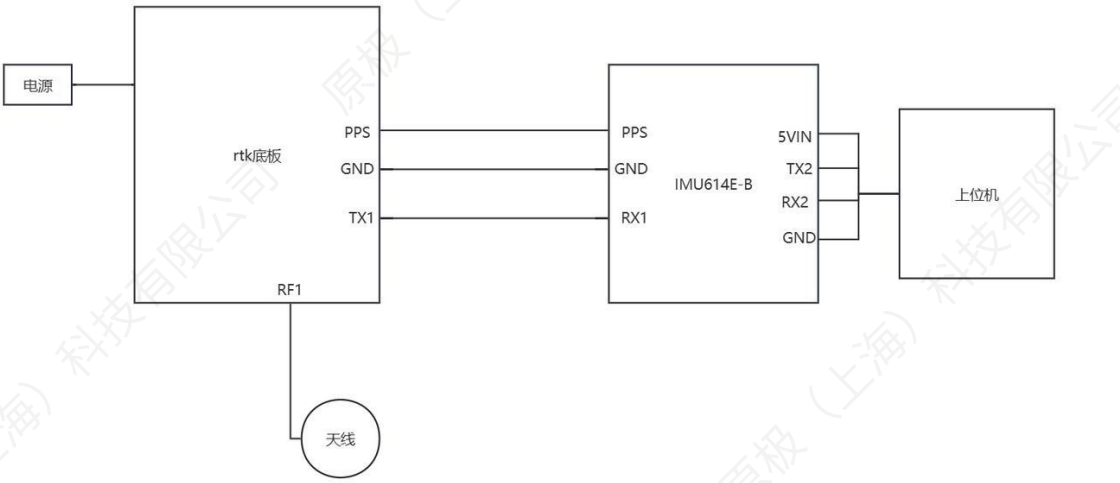


5.3.1 原理

IMU 通过串口传入的 RMC 协议，获取到 GPS 时间，本地的 ms 定时器去维护 GPS 时间，实现 ms 级自增，RTK 的 pps 在整秒时刻去校正 IMU 自身所维护的 GPS 时间，实现整秒对齐。时间同步以后，IMU 另一路串口发送的 AHRS 数据中时间戳就会变成同步后的 GPS 时间。

5.3.2 硬件连接

按照连接示意图将 IMU 与 RTK 底板连接。



5.3.3 RTK 配置要求

- GPRMC 10Hz，关闭 RTK 板卡的惯导辅助相关的功能关闭其他语句，RTK 的 TX 与 IMU 的 RX 相连（具体串口号根据型号有差异）。
- PPS 秒脉冲：1s 一次，上升沿触发，脉宽 5ms，对齐至 UTC 时间。高电平不得高于 5V（特殊型号除外）。

注 1 ：注意保证 RTK 串口波特率与 IMU 波特率一致。

3. RTK 设置：

- 连接 RTK 底板与主机，打开串口助手，依次输入指令：
- CONFIG（查看波特率配置）
- config com1　115200（对应 IMU 波特率）
- saveconfig （保存参数）

5.3.4 如何确认时间同步是否成功

确认步骤如下

- 同步前为 IMU 本身计数值，格式为：us

1. 同步后会变成单位为 ms 的 UTC 转化后的真值

以真值 43767630ms 为例,将给定的 ms 值转换为 s：

- 将 s 转换为 h、min 和 s：
- 首先，将秒数除以 3600（1h=3600s），得到小时数及剩余秒数。

2. 43,767s ÷ 3600 = 12h...567s（取整）

1. 接下来，将剩余的秒数除以 60（1min=60s），得到分钟数及剩余秒数。

2. 567s ÷ 60 = 9min...27s（取整）

1. 整理结果：

2. 将上述步骤中的 h、min、s 以及最初的小数部分组合起来，形成最终的 UTC 时间表示形式（hhmmss.sss）。

3. 最终 UTC 时间表示为：120927.63

4. 根据设置的不同 IMU 更新率，时间戳间隔也会相应变化。

1. 例如：

2. 当 IMU 更新率为 10Hz 时，对应的时间戳间隔为 100 ms；

1. 在这种情况下，时间戳将以每 10 ms 为一帧的周期进行发送，以确保与 IMU 数据的同步。以下为示例

```
%time_us,accx,accy,accz,gyrox,gyroy,gyroz,temperature,roll,pitch,yaw,mx,my,mz
43767630,0.000911493,-0.00593111,-0.999837,-0.112592,0.0775201,-0.239427,37.375,0.37371,0.0374005,359.763
43767640,0.00104886,-0.00639931,-1.00023,-0.11155,0.0945329,-0.198418,37.375,0.372709,0.0378392,359.763
43767650,0.0010143,-0.00654209,-1.00144,-0.143203,0.0859424,-0.187509,37.375,0.372681,0.038661,359.763
43767660,0.000973708,-0.00638983,-1.00176,-0.166009,0.092228,-0.200648,37.375,0.372481,0.0390947,359.763
43767670,0.00147395,-0.00683136,-1.001,-0.192246,-0.00178328,-0.157266,37.375,0.372781,0.0399402,359.763
43767680,0.00223095,-0.00695176,-1.00109,-0.0757273,-0.000226222,-0.138093,37.375,0.372781,0.0399402,359.763
43767690,0.00127585,-0.00571409,-1.00053,-0.173622,0.0492768,-0.163678,37.375,0.373524,0.0425388,359.763
43767700,0.00074174,-0.00655202,-1.00021,-0.115697,0.000131873,-0.160155,37.375,0.372256,0.0430011,359.763
43767710,0.00167231,-0.00625615,-1.00077,-0.196135,-0.000426489,-0.177769,37.375,0.372131,0.0433163,359.763
43767720,0.00185977,-0.0061884,-1.00042,-0.224989,-0.0107625,-0.0937578,37.375,0.371704,0.0457214,359.763
43767730,0.00024303,-0.00667565,-1.00119,-0.252968,0.0338021,-0.143835,37.375,0.371704,0.0457214,359.763
43767740,0.000261399,-0.00675453,-1.00142,-0.221033,0.0709242,-0.198763,37.375,0.37201,0.0447791,359.763
43767750,0.000898074,-0.00627877,-1.00137,-0.146918,0.0208479,-0.177816,37.375,0.372149,0.0439286,359.763
43767760,0.00114561,-0.00632768,-1.00142,-0.116412,0.0106449,-0.195621,37.375,0.371879,0.0448294,359.763
43767770,0.00031602,-0.0065025,-1.00118,-0.151384,0.107034,-0.151737,37.375,0.371925,0.0443909,359.763
43767780,0.000111739,-0.00596614,-1.00127,-0.191872,0.0392804,-0.190575,37.375,0.371925,0.0443909,359.763
43767790,-0.000446753,-0.00575444,-1.00054,-0.144282,0.0438216,-0.206097,37.375,0.371021,0.0432654,359.763
43767800,0.000744278,-0.00620892,-1.00097,-0.0806283,0.0402478,-0.2018,37.375,0.370079,0.0418417,359.763
43767810,0.000249961,-0.00627208,-1.00092,-0.205578,0.0244218,-0.173429,37.375,0.369413,0.0418069,359.763
43767820,-3.2315e-05,-0.00632706,-1.00116,-0.190721,0.0747152,-0.227714,37.375,0.3691,0.0408873,359.763
43767830,-0.00024524,-0.00670832,-1.00086,-0.117692,0.111822,-0.193005,37.375,0.3691,0.0408873,359.763
43767840,0.000537576,-0.00604259,-1.00108,-0.146042,0.043116,-0.178889,37.375,0.369541,0.0392609,359.763
43767850,0.00102963,-0.00625389,-1.00084,-0.165271,0.000858686,-0.146898,37.375,0.369012,0.0390227,359.763
43767860,0.00121603,-0.0058367,-1.00105,-0.207972,0.0699086,-0.172457,37.375,0.368091,0.0399286,359.763
43767870,0.00114882,-0.00628435,-1.00111,-0.192793,0.0395967,-0.177017,37.375,0.36763,0.0405705,359.763
```

5.3.5 如何验证时间同步成功后时间戳是否正确

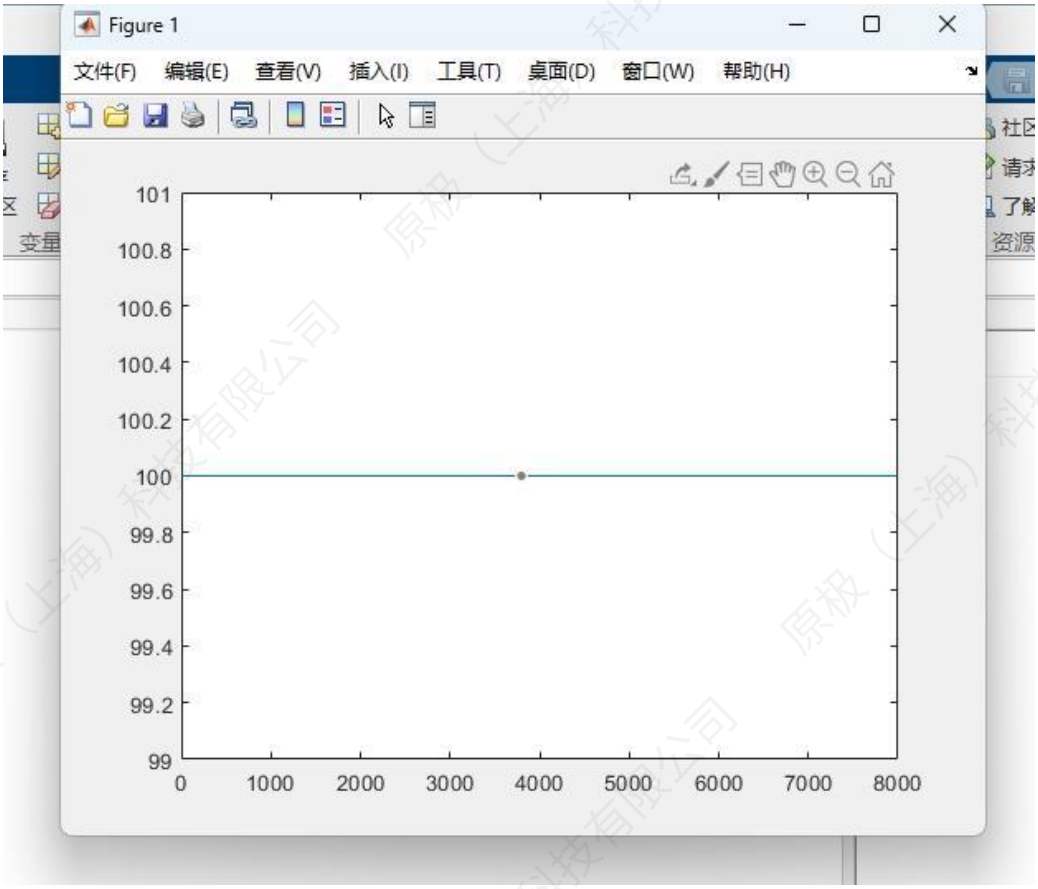
4. 同时采集 RMC 数据和时间同步后 AHRS 数据，并进行解码。

1. 将两者时间戳转换为相同格式，利用 matlab 等工具统计以下两组对应时间戳数据以下指标

2. 统计时间戳间隔是否稳定，确认是否有丢包，

1. 判断条件:10Hz 输出：间隔稳定为 100ms

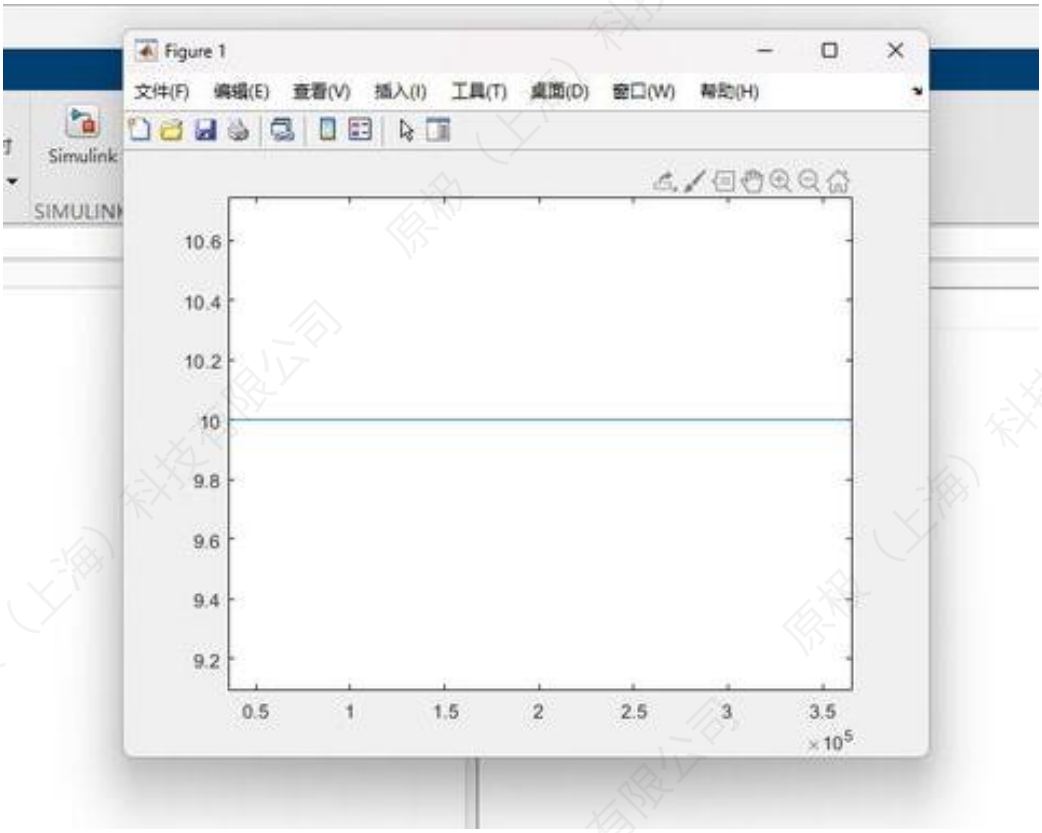
```
%time_us,accx,accy,accz,gyrox,gyroy,gyroz,temperature,roll,pitch,yaw,mx,my,mz
10138100,-0.00785668,0.000511979,-1.00068,0.00577856,0.0224228,-0.170839,43.6992,-0.185733,-0.373024,2.66486
10138200,-0.00548826,-8.32841e-06,-1.00069,0.120156,-0.0394001,0.0182559,43.6992,-0.177175,-0.373694,2.66124
10138300,-0.0177959,-0.0481571,-1.00444,-0.452245,-0.0625247,-6.75837,43.6992,-0.166894,-0.370253,2.60756
10138400,-0.00540888,-0.000368123,-0.999689,0.0587807,0.157406,-0.743163,43.7031,-0.179521,-0.368657,2.35367
10138500,-0.00681769,0.00110348,-1.00052,-0.0118609,-0.0201513,-0.0745728,43.6992,-0.174604,-0.366914,2.32512
10138600,-0.00541474,-2.34533e-05,-1.00088,0.0289074,-0.0428571,0.0164407,43.6992,-0.168918,-0.367043,2.32495
10138700,-0.00661371,0.000592246,-1.00062,0.00993176,-0.000438364,-0.0288885,43.6953,-0.163664,-0.364568,2.3255
10138800,-0.0048226,6.92813e-05,-1.00043,-0.00735649,0.0273419,-0.0158955,43.6992,-0.155855,-0.371403,2.33634
10138900,-0.00664255,0.000866315,-1.00145,0.0919172,-0.0107388,-0.0110982,43.7031,-0.151304,-0.366063,2.33678
10139000,-0.00515508,-0.000483297,-0.999668,-0.00652478,0.114841,-0.00554243,43.6992,-0.147101,-0.370124,2.3664
10139100,-0.00679933,8.07165e-05,-1.00015,0.0193778,-0.00205665,0.000614035,43.6953,-0.141752,-0.363385,2.36672
10139200,-0.00633591,0.000995697,-1.00043,0.0350645,-0.0332949,0.0570056,43.6875,-0.136313,-0.36819,2.37055
10139300,-0.00676747,0.000319971,-1.00062,0.0243442,0.0444976,0.00981862,43.6875,-0.133467,-0.362919,2.37656
10139400,-0.00663674,0.000352815,-1.00029,0.00787024,0.0201864,-0.00876646,43.6953,-0.129017,-0.368695,2.37739
10139500,-0.00679523,0.00118807,-1.00101,-0.00626396,-0.0206355,-0.00567084,43.6953,-0.12762,-0.359295,2.37722
10139600,-0.00583052,8.28852e-05,-0.99977,0.0120336,0.0297521,-0.00074564,43.6992,-0.122915,-0.366905,2.37741
10139700,-0.00690794,0.000215317,-1.00052,0.000730211,-0.0126271,-0.0145498,43.6992,-0.116187,-0.361241,2.37884
10139800,-0.00553715,0.000524005,-1.00127,0.0570455,-0.014946,0.0273024,43.6992,-0.109288,-0.369261,2.37931
10139900,-0.0059149,5.67176e-05,-1.00066,0.00898911,0.0247195,0.0501371,43.7031,-0.100622,-0.362934,2.39978
10140000,-0.00614014,-5.688e-06,-1.0002,-0.121098,0.0651305,-0.00916049,43.707,-0.100607,-0.366901,2.41725
10140100,-0.00616532,-0.000320024,-1.00114,0.0116914,0.00976389,-0.00659498,43.707,-0.0948647,-0.361206,2.41734
10140200,-0.00566859,-0.0003737,-1.00026,-0.00623558,0.013495,0.00671942,43.707,-0.0888662,-0.36321,2.41761
10140300,-0.00680167,-8.83485e-05,-1.00077,-0.100855,0.0120432,-0.00189858,43.707,-0.0788807,-0.362496,2.41875
```



IMU10Hz

- 100Hz 输出：间隔稳定为 10ms

```
%time_us,accx,accy,accz,gyrox,gyroy,gyroz,temperature,roll,pitch,yaw,mx,my,mz
12168590,-0.000783923,0.00227535,-0.999381,0.0285426,0.143975,0.0399726,45.8711,-0.261506,-0.0423138,0.597821
12168600,-0.00109725,0.00241267,-1.0014,0.00518191,-0.0503187,-0.0185507,45.8711,-0.260736,-0.0420368,0.598055
12168610,-0.000684899,0.00193684,-1.00129,0.0177225,-0.0174652,-0.00366374,45.875,-0.260757,-0.0421491,0.597998
12168620,-0.00110074,0.00186692,-0.999633,-0.0096959,-0.0503915,-0.0200717,45.8789,-0.259998,-0.0419311,0.597955
12168630,-0.00106939,0.00180071,-0.999801,-0.0493901,0.0672579,0.0065974,45.8789,-0.259534,-0.0427631,0.59775
12168640,-0.0012183,0.00200934,-1.00052,-0.0110379,0.0375103,-0.0019704,45.8789,-0.259273,-0.0431737,0.59775
12168650,-0.0014417,0.00172895,-0.999741,0.00149567,-0.0142063,0.00627559,45.8789,-0.259389,-0.0429253,0.597709
12168660,-0.00050238,0.00208577,-1.00103,-0.0168498,-0.0534377,-0.0177292,45.8789,-0.258919,-0.0438387,0.597691
12168670,-0.000328396,0.0019613,-1.00188,-0.00600252,0.0650926,0.0114503,45.8789,-0.258381,-0.0436349,0.597619
12168680,-0.000636144,0.00222931,-0.99907,-0.0270124,0.0734599,-0.000431323,45.8789,-0.258442,-0.0435879,0.597725
12168690,0.000112191,0.00221846,-1.00066,0.0151018,-0.056569,0.0147384,45.8789,-0.258043,-0.04342,0.597736
12168700,-0.000231573,0.00266871,-1.00237,0.00901595,-0.0218375,0.00337262,45.8789,-0.257449,-0.0417969,0.597825
12168710,8.72014e-05,0.00254247,-0.999538,0.0329301,-0.00658109,0.0136745,45.8789,-0.257216,-0.0417704,0.598
12168720,0.00029099,0.00226171,-1.0001,0.0078541,0.0694334,0.000107848,45.8789,-0.256338,-0.0408185,0.597988
12168730,0.000775636,0.00183092,-1.00108,-0.00733744,-0.0481737,-0.00525246,45.8789,-0.255627,-0.038995,0.597951
12168740,0.000808696,0.00174323,-1.00027,0.0295915,0.00286059,-0.0054113,45.8789,-0.25467,-0.0370789,0.598049
12168750,6.85134e-05,0.00170626,-0.999359,-0.0195651,0.0255302,0.0089159,45.8789,-0.254544,-0.0368456,0.597978
12168760,0.000233264,0.00187571,-1.00239,-0.0111067,0.0395717,-0.0164223,45.8789,-0.2539,-0.0362479,0.597931
12168770,0.000169324,0.00123586,-0.998619,-0.0318644,-0.0207638,0.0203835,45.8789,-0.253344,-0.0350068,0.59792
12168780,0.00034488,0.000749115,-0.999338,-0.0227521,0.00308496,-0.0240282,45.8789,-0.253686,-0.0353146,0.597991
12168790,0.000321285,0.0015652,-1.00105,-0.00523137,0.00901491,-0.0121445,45.8789,-0.253063,-0.0340612,0.597738
12168800,-0.000715598,0.0026208,-1.00151,0.0337185,0.0201632,0.00606421,45.8789,-0.252559,-0.0335472,0.597631
12168810,-0.000791669,0.00262174,-0.999473,0.0266765,0.0124253,-0.00269134,45.8789,-0.25235,-0.0333288,0.597486
```



IMU100Hz

2. 统计两组对应时间戳数据相同输出频率，相同起点下时间差是否为 0（需在卫星情况良好的情况下）

以下为统计示例:

4	10147500	10147500	0
5	10147600	10147600	0
6	10147700	10147700	0
7	10147800	10147800	0
8	10147900	10147900	0
9	10148000	10148000	0
10	10148100	10148100	0
11	10148200	10148200	0
12	10148300	10148300	0
13	10148400	10148400	0
14	10148500	10148500	0
15	10148600	10148600	0
16	10148700	10148700	0
17	10148800	10148800	0
18	10148900	10148900	0
19	10149000	10149000	0
20	10149100	10149100	0
21	10149200	10149200	0
22	10149300	10149300	0
23	10149400	10149400	0
24	10149500	10149500	0
25	10149600	10149600	0
26	10149700	10149700	0

IMU10Hz

	12174500	12174500	0
	12174600	12174600	0
	12174700	12174700	0
	12174800	12174800	0
	12174900	12174900	0
	12175000	12175000	0
	12175100	12175100	0
	12175200	12175200	0
	12175300	12175300	0
0	12175400	12175400	0
1	12175500	12175500	0
2	12175600	12175600	0
3	12175700	12175700	0
4	12175800	12175800	0
5	12175900	12175900	0
6	12176000	12176000	0
7	12176100	12176100	0
8	12176200	12176200	0
9	12176300	12176300	0
0	12176400	12176400	0
1	12176500	12176500	0
2	12176600	12176600	0
3	12176700	12176700	0

IMU100Hz

5.4 通过 DRDY 信号信号在主机端做时间同步

5.4.1 DRDY 信号作用

在 IMU 中，DRDY (Data Ready) 信号是一个重要的状态标志或中断信号，用于指示 IMU 的数据已经准备好，并可以被读取。当 IMU 完成一轮数据采集和处理后，DRDY 信号会变为有效状态（通常为低电平），这表示新的加速度、角速度等数据已经准备好，可以被读取。

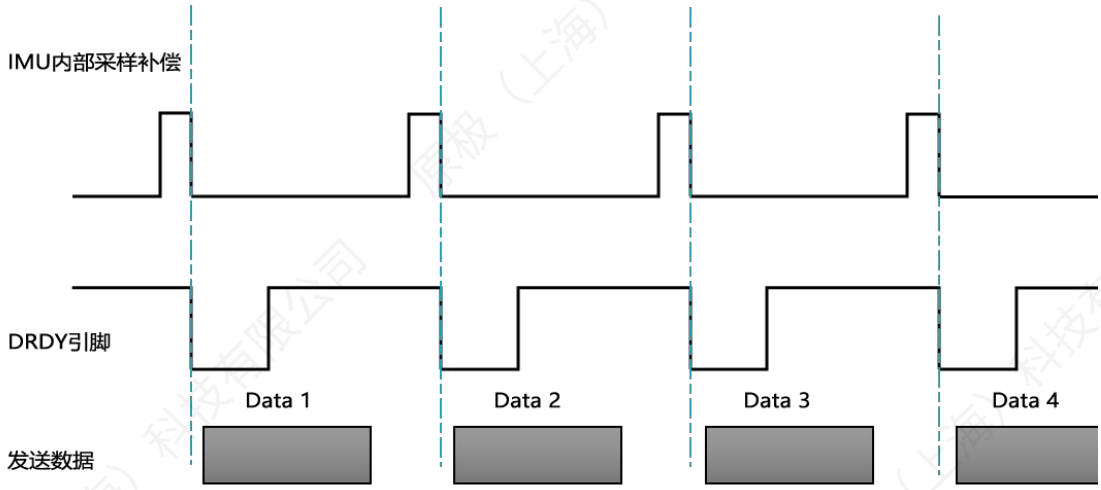
主机与 IMU 连接后，可以通过检测 DRDY 提供的来自 IMU 内部的时钟同步信号来确定数据准备就绪的确切时刻，并在此时刻添加时间戳并解析数据。这意味着每当 DRDY 信号变化时，主机就知道数据已经准备好，并可以记录此时的时间作为该数据的时间戳。

5.4.2 DRDY 信号

DRDY 引脚输出有两个目的：

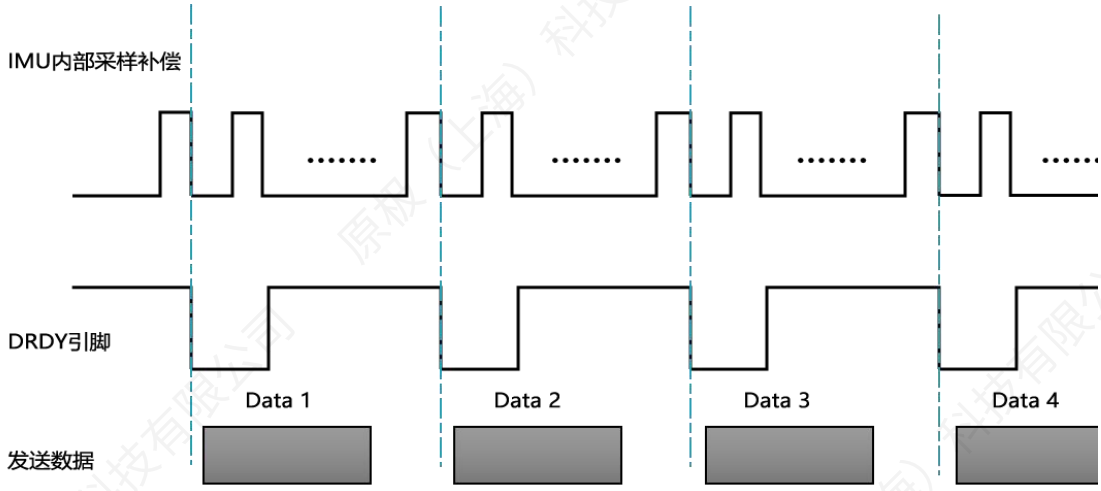
1. 提供来自 IMU 内部的时钟同步信号；
2. 提供信号表示开始传送数据帧。

内部采样频率与串口输出频率一致



当 IMU 内部采样频率（最大 ODR）与串口输出频率（当前 ODR）一致时，每当 imu 数据采样补偿完成后，DRDY 引脚将被立即拉低，此时数据帧将从串口发送，在下一周期 DRDY 引脚将被重新拉高。

串口输出频率小于 IMU 内部采样频率



当串口输出频率小于 IMU 内部采样频率时，每当 imu 数据采样补偿完成后，根据分频计数值（最大 ODR/当前 ODR）决定 DRDY 引脚是否被立即拉低。DRDY 拉低后数据帧将从串口发送，在下一 IMU 采样周期 DRDY 引脚将被重新拉高。

5.4.3 指令切换 PPS,DRDY

DRDY 默认开启，PPS 根据产品不同，需要发送 AT 指令进行开启。

1. 614E_B 时间同步指令　　AT+SET_TIME_SYNC=1
2. 6132 时间同步指令　　AT+DRY_ENABLE

3. 6132 时间同步关闭 AT+DRY_DISABLE
4. 618 时间同步指令 AT+SETIIC=0

6132/16488 系列&&618T&&P9

1. 时间同步指令 AT+DRY_ENABLE(618-T)
2. 时间同步关闭 AT+DRY_DISABLE

618 系列(不包含 618T)

1. 时间同步指令 AT+SETIIC=0

6. 升级功能 OTA

6.1 CAN 版本固件升级

使用 PCAN-OTA 上位机进行升级，需要使用 PCAN 设备。

PCAN-OTA 上位机升级界面

6.2 RS-485/RS-232/TTL 版本固件升级

使用原极 IMU 测试上位机——选择固件升级——打开固件——点击自动升级。

RS-485/RS-232/TTL 版本上位机升级界面

7. 用户参数功能

使用原极 IMU 测试上位机，用户可配置更新率和滤波器等值；

用户参数配置界面

8. 更新记录

手册最新版本：原极 IMU 通信协议手册

版本	日期	状态/注释
版本 1.0	2024.04.07	初稿
版本 1.1	2025.04.09	增加 spi 和参状态位说明
版本 1.2	2025.08.22	CAN 指令修改
版本 1.3	2026.04.15	修正四元数定义，补充时间同步描述