Command and Programming Manual

Rev 1.2

Applicable to CH10x/CH0x0/HI13/HI14/HI50 series

File: imu_cum_cn

Technical Support: support@hipnuc.com

Property: Public

Website: www.hipnuc.com

1. User Configuration

The default configuration of the product can meet the needs of most users. Therefore, it is recommended to carefully read this section before using the product and determine whether user configuration is necessary based on your specific requirements.

1.1 Static Detection Threshold - ZARU

In some scenarios, users may encounter very slow and nearly constant motion, such as a robot's return or the operation of a heavily loaded AGV. If the static detection threshold is set too high in these situations, it may result in zero offset update errors, causing sensor drift. We recommend a value of 0.2, but we advise users to adjust this value after actual testing.

> For serial interface configuration, please refer to 2.2.2.4 CONFIG - Static Detection Threshold.

1.2 Low Dynamic Motion

If your work conditions involve low dynamic motion, such as static angle measurement or slow carrier rotation, you can achieve higher measurement accuracy and reduced measurement noise output by setting a lower accelerometer bandwidth and range. The accelerometer bandwidth affects the static accuracy of Pitch/Roll, while the gyroscope's bandwidth does not affect static accuracy.

> For serial interface configuration, please refer to 2.2.2.4 CONFIG - Sensor Range and Bandwidth Configuration.

1.3 Magnetic Field Assisted Scenarios (AHRS/9-Axis Mode)

In the vast majority of cases, especially for robots and indoor environments, AHRS (9-axis) mode is susceptible to interference, resulting in heading angle errors. In a few open and magnetic field-free environments, you can try using the magnetic field-assisted mode, such as with drones. Before use, the module should be configured for magnetic field-assisted mode and magnetic field calibration should be performed.

> For serial interface configuration, please refer to 2.2.2.4 CONFIG - Mode Configuration.
> For magnetic field calibration methods, refer to the user manual - Magnetic Field Calibration section.

1.4 Synchronous Input and Output

- Data Synchronous Input (SIN): The pin operates in pull-up input mode, with a high level in idle state. When the module detects a falling edge, it outputs a data frame. It should be left unconnected when not in use.
- Data Synchronous Output (SOUT): The pin operates in output mode and is in a high level (idle) state when there is no data output. It transitions to a low level when a data frame starts transmitting and returns to a high level (idle) after data transmission is complete. By default, the module outputs data at 100Hz, and this pin outputs a 100Hz pulse with a 50% duty cycle. It should be left unconnected when

not in use.

> When using the synchronous input function, start by sending the command UNLOGALL via serial input to disable all timed outputs.

2. Serial Data Protocol and Commands

This protocol is a proprietary binary protocol designed to output all sensor information. Products with UART (RS-232/TTL), USB, and RS-485 interfaces support this protocol.

2.1 Serial Data Protocol

2.1.1 Data Frame Format

After module power-up, data frames are output at the default frame rate (100Hz) with the following frame format:

Field Name	Value	Length (Bytes)	Description
Frame Header	0x5A	1	0x5A
Frame Type	0xA5	1	0xA5
Data Length	1-512	2	Length of the data field within the frame, LSB (Little-Endian) Represents the length of the data field (excluding frame header, frame type, length, and CRC).
CRC Checksum	-	2	16-bit CRC checksum of all fields except the CRC byte (frame header, frame type, length, data field). LSB (Little-Endian)
Data Field	-	1-512	Data carried within one frame, composed of multiple sub-data packets, each consisting of a data packet tag and data. The tag determines the data type and length.

2.1.2 Data Field Content

The data field consists of 76 bytes, including module ID, temperature, raw IMU data, magnetic field, pressure, fused attitude data, and more.

Byte Offset	Data Type	Size (Bytes)	Unit	Description
0	uint8_t	1	-	Data packet tag: 0x91
1	uint8_t	1	-	Reserved
2	uint8_t	1	-	Reserved
3	int8_t	1	°C	Module average temperature
4	float	4	Pa	Pressure
8	uint32_t	4	ms	Node local timestamp information, accumulates from system boot, increasing by 1 every millisecond
12	float	12	mg	Acceleration after factory calibration, order: XYZ
24	float	12	deg/s	Angular velocity after factory calibration, order: XYZ
36	float	12	uT	Magnetic intensity, order: XYZ
48	float	12	deg	Euler angles, order: Roll, Pitch, Yaw
60	float	16	-	Node quaternion set, order: WXYZ

2.1.3 CRC

16-bit CRC Code

```
/*
    currectCrc: previous crc value, set 0 if it's first section
    src: source stream data
    lengthInBytes: length
*/
static void crc16_update(uint16_t *currectCrc, const uint8_t *src, uint32_t lengthInBytes)
{
    uint32_t crc = *currectCrc;
    uint32_t j;
    for (j=0; j < lengthInBytes; ++j)
    {
        uint32_t i;
        uint32_t byte = src[j];
        crc ^= byte << 8;
        for (i = 0; i < 8; ++i)
        {
            uint32_t temp = crc << 1;
            if (crc & 0x8000)
            {
                temp ^= 0x1021;
            }
            crc = temp;
        }
    }
    *currectCrc = crc;
}
```

2.1.4 Data Frame Structure Example

Sampling one frame of data using a serial assistant, which consists of 82 bytes in total. The first 6 bytes include the frame header, length, and CRC checksum value. The remaining 76 bytes constitute the data field. Assuming the data is received in a C language array buf, it looks like this:

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

1.Checking the frame header to obtain the data field length and CRC checksum value:

Frame Header: 5A A5

Frame Data Field Length: 4C 00 (0x00<<8) + 0x4C = 76

CRC Checksum Value: 6C 51 (0x51<<8) + 0x6C = 0x516C

2.Verifying CRC.

```
uint16_t payload_len;
uint16_t crc;
crc = 0;
payload_len = buf[2] + (buf[3] << 8);

/* calulate 5A A5 and LEN filed crc */
crc16_update(&crc, buf, 4);

/* calulate payload crc */
crc16_update(&crc, buf + 6, payload_len);
```

CRC value is 0x516C, which matches the CRC value carried within the frame, confirming a successful CRC verification.

3.Received Data

Starting from 0x91, this is the data field of the data packet. Define data structures and common conversion macros

```
#include "stdio.h"
#include "string.h"
/* common type conversion */
#define U1(p) (*((uint8_t *)(p)))
#define I1(p) (*((int8_t  *)(p)))
#define I2(p) (*((int16_t  *)(p)))
static uint16_t U2(uint8_t *p) {uint16_t u; memcpy(&u,p,2); return u;}
static uint32_t U4(uint8_t *p) {uint32_t u; memcpy(&u,p,4); return u;}
static int32_t  I4(uint8_t *p) {int32_t  u; memcpy(&u,p,4); return u;}
static float    R4(uint8_t *p) {float    r; memcpy(&r,p,4); return r;}
typedef struct
{
    uint8_t     tag;              /* item tag: 0x91       */
    uint32_t    id;               /* user define ID       */
    float       acc[3];           /* acceleration         */
    float       gyr[3];           /* angular velocity     */
    float       mag[3];           /* magnetic field       */
    float       eul[3];           /* attitude: eular angle */
```

```c
    float       quat[4];          /* attitude: quaternion  */
    float       pressure;         /* barometer          */
    uint32_t    timestamp;
}imu_data_t;
```

Upon receiving data, the payload portion begins from buf[6]=0x91

```c
    imu_data_t i0x91 = {0};
    int offset = 6; /* payload strat at buf[6] */
    i0x91.tag =             U1(buf+offset+0);
    i0x91.id =              U1(buf+offset+1);
    i0x91.pressure =        R4(buf+offset+4);
    i0x91.timestamp =       U4(buf+offset+8);
    i0x91.acc[0] =          R4(buf+offset+12);
    i0x91.acc[1] =          R4(buf+offset+16);
    i0x91.acc[2] =          R4(buf+offset+20);
    i0x91.gyr[0] =          R4(buf+offset+24);
    i0x91.gyr[1] =          R4(buf+offset+28);
    i0x91.gyr[2] =          R4(buf+offset+32);
    i0x91.mag[0] =          R4(buf+offset+36);
    i0x91.mag[1] =          R4(buf+offset+40);
    i0x91.mag[2] =          R4(buf+offset+44);
    i0x91.eul[0] =          R4(buf+offset+48);
    i0x91.eul[1] =          R4(buf+offset+52);
    i0x91.eul[2] =          R4(buf+offset+56);
    i0x91.quat[0] =         R4(buf+offset+60);
    i0x91.quat[1] =         R4(buf+offset+64);
    i0x91.quat[2] =         R4(buf+offset+68);
    i0x91.quat[3] =         R4(buf+offset+72);
```

Print the received data

```
printf("%-16s0x%X\r\n",                "tag:",          i0x91.tag);
printf("%-16s%d\r\n",                  "id:",           i0x91.id);
printf("%-16s%8.4f %8.4f %8.4f\r\n",    "acc(G):",       i0x91.acc[0], i0x91.acc[1],  i0x91.acc[2]);
printf("%-16s%8.3f %8.3f %8.3f\r\n",    "gyr(deg/s):",   i0x91.gyr[0], i0x91.gyr[1],  i0x91.gyr[2]);
printf("%-16s%8.3f %8.3f %8.3f\r\n",    "mag(uT):",      i0x91.mag[0], i0x91.mag[1],  i0x91.mag[2]);
printf("%-16s%8.3f %8.3f %8.3f\r\n",    "eul(deg):",     i0x91.eul[0], i0x91.eul[1],  i0x91.eul[2]);
printf("%-16s%8.3f %8.3f %8.3f %8.3f\r\n",  "quat:",           i0x91.quat[0], i0x91.quat[1], i0x91.quat[2], i0x91.quat[3]);
printf("%-16s%8.3f\r\n",                    "presure(pa):", i0x91.pressure);
printf("%-16s%d\r\n",                       "timestamp(ms):", i0x91.timestamp);
```

print results

```
tag:            0x91
id:             0
acc(G):           0.2242   0.7701   0.6910
gyr(deg/s):     -54.708  -20.077 -119.070
mag(uT):          19.183  -26.208  -34.542
eul(deg):         48.720  -21.014  -45.512
quat:              0.855    0.310   -0.310   -0.277
presure(pa):      -0.000
timestamp(ms):  310205
```

2.2 Configuration Commands

Module configuration is done using string commands, and each command must be terminated with a carriage return and line feed (`\r\n`) for the system to recognize it.

> When using our GUI-based software, you don't need to worry about this issue. However, when configuring the module using other methods, you should consider the presence of `\r\n` in your commands.

2.2.1 Command Set

Command	Function	Notes
REBOOT	Reset the module	
SAVECONFIG	Save all configuration parameters	
SERIALCONFIG	Set the serial port baud rate	
CONFIG	Set user parameters and modes	Requires SAVECONFIG and reboot to take effect
LOG	Print module information or configuration output data	
UNLOGALL	Disable all message output	
FRESET	Restore factory settings	

> All configuration commands require a SAVECONFIG (to save settings) and a reboot or power cycle to take effect.

2.2.2 Command Details

2.2.2.1 `REBOOT`

Resets the module immediately, having the same effect as a power cycle.

2.2.2.2 `SAVECONFIG`

Saves all user configurations to Flash memory.

2.2.2.3 `SERIALCONFIG`

Sets the serial port baud rate. Options: 9600/115200/256000/460800/921600.

Set the serial port baud rate to 115200:
`SERIALCONFIG COM0 115200`
`SAVECONFIG`

> Note: Be careful when using this command, as entering an incorrect baud rate can result in communication issues with the module.

2.2.2.4 `CONFIG`

Used to configure module operating parameters. All configurations require SAVECONFIG and a reboot to take effect.

- Mode Configuration:

`CONFIG ATT MODE 0` Configures the module for 6DOF mode.
`CONFIG ATT MODE 1` Configures the module for AHRS (9-axis) mode.

- Level Calibration:

`CONFIG ATT RST 2` Calibrates: Sets the current pitch and roll angles to 0 (Pitch = Roll = 0), while maintaining the yaw angle.
`CONFIG ATT RST 3` Cancels the level calibration, clearing the current pitch and roll angle calibration.
`CONFIG ATT RST 4` Resets the yaw angle to zero.

> When executing the CONFIG ATT RST command, the module should be kept still. Executing this command while the module is in motion may result in calibration errors.

- Static Detection Threshold:

Configuration command: `CONFIG IMU ZARU <VAL>`
`CONFIG IMU ZARU 0.2` Sets the static detection threshold to 0.2.

> VAL range: 0.01-2. Smaller VAL values may lead to cumulative drift in the yaw angle during long static periods, while larger VAL values may lead to erroneous zero bias updates during the mentioned conditions. VAL=0 is equivalent to turning off static zero bias updates.

- Sensor Range and Bandwidth Configuration

Setting Accelerometer Bandwidth and Range

`CONFIG IMU ABW <VAL>` Sets the accelerometer bandwidth. VAL range: 0 (5Hz), 1 (10Hz), 2 (20Hz), 3 (40Hz), 4 (80Hz), 5 (145Hz, default), 6 (230Hz).

`CONFIG IMU ARG <VAL>` Sets the accelerometer range. VAL range: 0 (3G), 1 (6G), 2 (12G, default), 3 (24G).

Example:

`CONFIG IMU ABW 0` Sets the accelerometer bandwidth to 5Hz.
`CONFIG IMU ARG 0` Sets the accelerometer range to 3G.

Setting Gyroscope Bandwidth and Range

`CONFIG IMU GBW  <VAL>` Sets the gyroscope bandwidth. VAL range: 0 (12Hz), 1 (23Hz), 2 (32Hz), 3 (47Hz), 4 (64Hz), 5 (116Hz, default), 6 (230Hz).

`CONFIG IMU GRG  <VAL>` Sets the gyroscope range. VAL range: 0 (250dps), 1 (500dps), 2 (1000dps), 3 (2000dps, default).

Example:

`CONFIG IMU GBW 0` Sets the gyroscope bandwidth to 12Hz.
`CONFIG IMU GRG 0` Sets the gyroscope range to 250dps.

> For low-dynamic scenarios (vehicle navigation, slow motion, deformation measurement, tilt detection, heavy-duty AGVs), lower bandwidth should be set. For high-dynamic scenarios (handheld devices/VR, humanoid robots, small aircraft), higher gyroscope bandwidth should be set. In general, accelerometer bandwidth is sensitive to acceleration (vibration) but not to rotation (attitude changes), and it does not need to be changed in typical applications. However, if the target application involves high-frequency vibration measurement, the accelerometer bandwidth may need to be increased.

- Coordinate System Rotation

`CONFIG IMU URFR C00,C01,C02,C10,C11,C12,C20,C21,C22`

Where C_{nn} supports floating-point values.

$$\begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix}_U = \begin{bmatrix} C00 & C01 & C02 \\ C10 & C11 & C12 \\ C20 & C21 & C22 \end{bmatrix} \cdot \begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix}_B \tag{1}$$

Where $\begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix}_U$ is the sensor data in the rotated sensor coordinate system, and $\begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix}_B$ is the sensor data in the original sensor coordinate system.

Examples:

The new sensor coordinate system is rotated -90° around the original X-axis (with the Y-axis positive direction pointing downwards, vertical installation), configuration command:

`CONFIG IMU URFR 1,0,0,0,0,1,0,-1,0`

The new sensor coordinate system is rotated 90° around the original X-axis (with the Y-axis positive direction pointing upwards, vertical installation), configuration command:

`CONFIG IMU URFR 1,0,0,0,0,-1,0,1,0`

The new sensor coordinate system is rotated 180°, configuration command:

`CONFIG IMU URFR 1,0,0,0,-1,0,0,0,-1`

The new sensor coordinate system is rotated 90° around the original Y-axis (with the X-axis positive direction pointing upwards, vertical installation), configuration command:

`CONFIG IMU URFR 0,0,-1,0,1,0,1,0,0`

The new sensor coordinate system is rotated -90° around the original Y-axis (with the X-axis positive direction pointing downwards, vertical installation), configuration command:

`CONFIG IMU URFR 0,0,1,0,1,0,-1,0,0`

The new sensor coordinate system is rotated 180°, configuration command:

`CONFIG IMU URFR -1,0,0,0,1,0,0,0,-1`

The new sensor coordinate system is rotated 90° around the original Z-axis, configuration command:

`CONFIG IMU URFR 0,-1,0,1,0,0,0,0,1`

The new sensor coordinate system is rotated -90° around the original Z-axis, configuration command:

`CONFIG IMU URFR 0,1,0,-1,0,0,0,0,1`

Restore factory defaults:

`CONFIG IMU URFR 1,0,0,0,1,0,0,0,1`

> Setting URFR requires a software reset or power cycle to take effect. It does not need to be sent every time the module is powered on.

- Set Inclinometer Output Range

`CONFIG IMU INC_FMT <VAL>`

`CONFIG IMU INC_FMT 0` Inclinometer output mode: 0° - 360°
`CONFIG IMU INC_FMT 1` Inclinometer output mode: -180° - 180°

2.2.2.5 `LOG`

- Enable/Disable Data Output

`LOG ENABLE` Enable data frame output (default).
`LOG DISABLE` Disable data frame output.

- Display Module Version Information

`LOG VERSION` Print firmware version information.

- Display User Configuration Information

`LOG USRCONFIG` Print user configuration information for verification.

```
ATT_MODE: 0 /* Operating mode: 0:6Dof,  1:AHRS(9 axis) */
AUTO_LV: 0 /* Reserved */
ACC_RG:  2 /* Accelerometer range */
GYR_RG:  3 /* Gyroscope range */
ACC_BW:  5 /* Accelerometer bandwidth */
GYR_BW:  5 /* Gyroscope bandwidth */
ZARU THR:0.500 /* Static detection threshold */
ACC_R:   1.00 /* Reserved */
MAG_R:   1.00 /* Reserved */
ATT_Q:   1.00 /* Reserved */
GYR_LMF: 1 /* Reserved */
MDBUSID: 0x50 /* Reserved */
```

- Display Serial Configuration Information

`LOG COMCONFIG` Print serial port and output protocol configuration information

```
LIST
    COM1: 115200 MSG:IMU91(100) /* COM1, baud rate 115200, protocol:91, 100Hz */
    COM2: 115200 MSG:IMU91(100) /* COM2, baud rate 115200, protocol:91, 100Hz */
OK
```

- Configure Output Messages

`LOG <MSG> <TYPE> <PERIOD>`

MSG:IMU91

TYPE: ONTIME： Periodic output， ONMARK: External trigger synchronous output

PERIOD: Output frame period in seconds, range: 1 (1Hz), 0.5 (2Hz), 0.1 (10Hz), 0.02 (50Hz), 0.01 (100Hz), 0.004 (250Hz), 0.002 (500Hz)

Examples (Periodic 100Hz Output):

`LOG IMU91 ONTIME 0.01` Set the output period of the current COM to 91 data packets to 0.01s(100Hz)

`SAVECONFIG` Save the settings.

`REBOOT` Reboot to apply the settings.

Examples (Disable Output):

`LOG IMU91 ONTIME 0` Disable 91 data packet output.

`SAVECONFIG` Save the settings

`REBOOT` Reboot to apply the settings.

Examples (Synchronous Trigger Output)

`UNLOGALL` Disable all data output

`LOG IMU91 ONMARK 1` Set the current COM port's 91 data packet to synchronous trigger mode

`SAVECONFIG` Save the settings

`REBOOT` Reboot to apply the settings.

> When the output frame rate is set to a higher value (e.g., 500Hz), the default baud rate of 115200 may not meet the output bandwidth requirements. In such cases, you will need to increase the module's baud rate (e.g., to 921600) to ensure correct data output.
> After setting the baud rate parameter, save it and reset the module to take effect. You will also need to adjust the baud rate on the host computer or other host devices accordingly.

2.2.2.6 `UNLOGALL`

Sets the output frequency of all timed output messages to 0 (disabled)

2.2.2.7 `FRESET`

Restores factory default settings.

3. RS-485 Data Protocol and commands (Modbus)

The RS-485 data protocol follows the Modbus RTU protocol specification. Data is sent and received in the form of registers, each register occupying 2 bytes, using big-endian byte order (high byte first). The default configuration and commands for the module are as follows:

- Modbus Commands:

  Write: 0x06 (Write Single Register): Write a single register (each Modbus register is 2 bytes).

  Read: 0x03 (Read Holding Registers): Read one or more registers' data.

  Custom Function Code: 0x50, used for Modbus ID automatic assignment, facilitating mass production deployment, firmware upgrades, etc.

- Modbus Device Address is user-configurable. The default is 80 (0x50).
- Baud Rates: 9600 or 115200, with the default being 115200. The format is 8 data bits, 1 stop bit, no parity (N8N1).

> RS-485 also supports serial protocols, with protocol content matching that of Chapter 2. Please contact us for more specific information

3.1 Data Frames

3.1.1 Read Register (0x03)

Sent by the Host (Master):

Field Name	Value	Description
ID	1-0xFF	Modbus device address
FUN_CODE	0x03	Command code
ADDR_H	-	High 8 bits of the register address to read
ADDR_L	-	Low 8 bits of the register address to read
LEN_H	-	High 8 bits of the register length (in the number of registers) to read
LEN_L	-	Low 8 bits of the register length (in the number of registers) to read
CRC_L	-	Low 8 bits of the CRC
CRC_H	-	High 8 bits of the CRC

Returned by the Slave (Module):

Field Name	Value	Description
ID	1-0xFF	Modbus device address
FUN_CODE	0x03	Command code
LEN	-	Length of the returned register data (excluding ID, FUN_CODE, LEN, and CRC fields) in bytes
DATAH	-	High 8 bits of the returned data
DATAL	-	Low 8 bits of the returned data
----	-	High 8 bits of the returned data
----	-	Low 8 bits of the returned data
CRC_L	-	Low 8 bits of the CRC
CRC_H	-	High 8 bits of the CRC

3.1.2 Write Register (0x06)

Field Name	Value	Description
ID	1-0xFF	Modbus device address
FUN_CODE	0x06	Command code
ADDR_H	-	High 8 bits of the register address to write
ADDR_L	-	Low 8 bits of the register address to write
DATA_H	-	High 8 bits of the data to write
DATA_L	-	Low 8 bits of the data to write
CRC_L	-	Low 8 bits of the CRC
CRC_H	-	High 8 bits of the CRC

Returned by the Slave (Module):

Field Name	Value	Description
ID	1-0xFF	Modbus device address
FUN_CODE	0x06	Command code
ADDR_H	-	High 8 bits of the register address to write
ADDR_L	-	Low 8 bits of the register address to write
DATA_H	-	High 8 bits of the data written
DATA_L	-	Low 8 bits of the data written
CRC_L	-	Low 8 bits of the CRC
CRC_H	-	High 8 bits of the CRC

3.1.3 CRC

- Online CRC：https://www.23bei.com/tool/59.html
- C Code：

```c
static const uint16_t modbus_crc_table[256] = {
   0x0000, 0xc0c1, 0xc181, 0x0140, 0xc301, 0x03c0, 0x0280, 0xc241,
   0xc601, 0x06c0, 0x0780, 0xc741, 0x0500, 0xc5c1, 0xc481, 0x0440,
   0xcc01, 0x0cc0, 0x0d80, 0xcd41, 0x0f00, 0xcfc1, 0xce81, 0x0e40,
   0x0a00, 0xcac1, 0xcb81, 0x0b40, 0xc901, 0x09c0, 0x0880, 0xc841,
   0xd801, 0x18c0, 0x1980, 0xd941, 0x1b00, 0xdbc1, 0xda81, 0x1a40,
   0x1e00, 0xdec1, 0xdf81, 0x1f40, 0xdd01, 0x1dc0, 0x1c80, 0xdc41,
   0x1400, 0xd4c1, 0xd581, 0x1540, 0xd701, 0x17c0, 0x1680, 0xd641,
   0xd201, 0x12c0, 0x1380, 0xd341, 0x1100, 0xd1c1, 0xd081, 0x1040,
   0xf001, 0x30c0, 0x3180, 0xf141, 0x3300, 0xf3c1, 0xf281, 0x3240,
   0x3600, 0xf6c1, 0xf781, 0x3740, 0xf501, 0x35c0, 0x3480, 0xf441,
   0x3c00, 0xfcc1, 0xfd81, 0x3d40, 0xff01, 0x3fc0, 0x3e80, 0xfe41,
   0xfa01, 0x3ac0, 0x3b80, 0xfb41, 0x3900, 0xf9c1, 0xf881, 0x3840,
   0x2800, 0xe8c1, 0xe981, 0x2940, 0xeb01, 0x2bc0, 0x2a80, 0xea41,
   0xee01, 0x2ec0, 0x2f80, 0xef41, 0x2d00, 0xedc1, 0xec81, 0x2c40,
   0xe401, 0x24c0, 0x2580, 0xe541, 0x2700, 0xe7c1, 0xe681, 0x2640,
   0x2200, 0xe2c1, 0xe381, 0x2340, 0xe101, 0x21c0, 0x2080, 0xe041,
   0xa001, 0x60c0, 0x6180, 0xa141, 0x6300, 0xa3c1, 0xa281, 0x6240,
   0x6600, 0xa6c1, 0xa781, 0x6740, 0xa501, 0x65c0, 0x6480, 0xa441,
   0x6c00, 0xacc1, 0xad81, 0x6d40, 0xaf01, 0x6fc0, 0x6e80, 0xae41,
   0xaa01, 0x6ac0, 0x6b80, 0xab41, 0x6900, 0xa9c1, 0xa881, 0x6840,
   0x7800, 0xb8c1, 0xb981, 0x7940, 0xbb01, 0x7bc0, 0x7a80, 0xba41,
   0xbe01, 0x7ec0, 0x7f80, 0xbf41, 0x7d00, 0xbdc1, 0xbc81, 0x7c40,
   0xb401, 0x74c0, 0x7580, 0xb541, 0x7700, 0xb7c1, 0xb681, 0x7640,
   0x7200, 0xb2c1, 0xb381, 0x7340, 0xb101, 0x71c0, 0x7080, 0xb041,
   0x5000, 0x90c1, 0x9181, 0x5140, 0x9301, 0x53c0, 0x5280, 0x9241,
   0x9601, 0x56c0, 0x5780, 0x9741, 0x5500, 0x95c1, 0x9481, 0x5440,
   0x9c01, 0x5cc0, 0x5d80, 0x9d41, 0x5f00, 0x9fc1, 0x9e81, 0x5e40,
   0x5a00, 0x9ac1, 0x9b81, 0x5b40, 0x9901, 0x59c0, 0x5880, 0x9841,
   0x8801, 0x48c0, 0x4980, 0x8941, 0x4b00, 0x8bc1, 0x8a81, 0x4a40,
   0x4e00, 0x8ec1, 0x8f81, 0x4f40, 0x8d01, 0x4dc0, 0x4c80, 0x8c41,
   0x4400, 0x84c1, 0x8581, 0x4540, 0x8701, 0x47c0, 0x4680, 0x8641,
   0x8201, 0x42c0, 0x4380, 0x8341, 0x4100, 0x81c1, 0x8081, 0x4040
};

 uint16_t modbus_crc_calc(uint8_t *buf, uint16_t len)
{
    uint16_t crc = 0xFFFFU;
```

```
    uint8_t nTemp;

    while (len--)
    {
        nTemp = *buf++ ^ crc;
        crc >>= 8;
        crc  ^= modbus_crc_table[(nTemp & 0xFFU)];
    }

    return(crc);
}
```

3.2 Register List

Address (Hex)	Address (Dec)	Name	Function	R/W	Description
0x00	0	CTL	Control	W	See Modbus Setting Module Chapter
0x04	4	BAUD	Baud Rate	R	Baud rate
0x05	5	ID	ID	R	Modbus ID
0x1F	31	BW	Bandwidth	R/W	Cutoff frequency: 0: 12Hz, 1: 23Hz, 2: 32Hz, 3: 47Hz (default), 4: 64Hz, 5: 116Hz
0x34	52	ACCX	Acceleration X	R	Unit: G (1G = 1 gravity), Scale Factor: 0.00048828
0x35	53	ACCY	Acceleration Y	R	Unit: G (1G = 1 gravity), Scale Factor: 0.00048828
0x36	54	ACCZ	Acceleration Z	R	Unit: G (1G = 1 gravity), Scale Factor: 0.00048828
0x37	55	GYRX	Angular Velocity X	R	Unit: deg/s, Scale Factor: 0.061035
0x38	56	GYRY	Angular Velocity Y	R	Unit: deg/s, Scale Factor: 0.061035
0x39	57	GYRZ	Angular Velocity Z	R	Unit: deg/s, Scale Factor: 0.061035
0x3A	58	MAGX	Magnetic Field X	R	Unit: uT, Scale Factor: 0.030517
0x3B	59	MAGY	Magnetic Field Y	R	Unit: uT, Scale Factor: 0.030517
0x3C	60	MAGZ	Magnetic Field Z	R	Unit: uT, Scale Factor: 0.030517
0x3D	61	ROLL_H	Roll Angle High 16-bit	R	Unit: deg, Scale Factor: 0.001
0x3E	62	ROLL_L	Roll Angle Low 16-bit	R	Unit: deg, Scale Factor: 0.001
0x3F	63	PITCH_H	Pitch Angle High 16-bit	R	Unit: deg, Scale Factor: 0.001
0x40	64	PITCH_L	Pitch Angle Low 16-bit	R	Unit: deg, Scale Factor: 0.001
0x41	65	YAW_H	Yaw Angle High 16-bit	R	Unit: deg, Scale Factor: 0.001
0x42	66	YAW_L	Yaw Angle Low 16-bit	R	Unit: deg, Scale Factor: 0.001
0x43	67	TEMP	Temperature	R	Unit: °C, Scale Factor: 0.01
0x44	68	PRS_H	Pressure High 16-bit	R	Unit: Pa, Scale Factor: 0.01
0x45	69	PRS_L	Pressure Low 16-bit	R	Unit: Pa, Scale Factor: 0.01
0x46	70	Q0	Quaternion QW	R	Quaternion, Scale Factor: 0.00003
0x47	71	Q1	Quaternion QX	R	Quaternion, Scale Factor: 0.00003
0x48	72	Q2	Quaternion QY	R	Quaternion, Scale Factor: 0.00003
0x49	73	Q3	Quaternion QZ	R	Quaternion, Scale Factor: 0.00003
0x4A	74	SINGLE_X	Inclinometer X-axis Angle	R	Inclinometer X-axis angle, 0-360, Unit: deg, Scale Factor: 0.005493
0x4B	74	SINGLE_Y	Inclinometer Y-axis Angle	R	Inclinometer Y-axis angle, 0-360, Unit: deg, Scale Factor: 0.005493

Address (Hex)	Address (Dec)	Name	Function	R/W	Description
0x66	102	KF_ACC_R	Accelerometer Attitude Feedback Coefficient	R/W	Accelerometer attitude correction coefficient adjustment, Default: 10, Range: 1-20, Smaller values result in greater accelerometer correction.
0x70-0x77	112-119	PNAME	Device Name	R	Device name string, ASCII, occupies 8 registers
0x78	120	SW_VERSION	Software Version	R	Software version
0x79	121	BL_VERSION	Bootloader Version	R	Bootloader version
0x7F-0x82	127-130	SN	Unique Product Serial Number	R	Unique product serial number, occupies 4 registers
0xA0-0xAF	160-175	ACC_CAL	Accelerometer Calibration Parameters	R	Accelerometer factory calibration parameters, Scale Factor: 0.001
0xB0-0xBF	176-191	GYR_CAL	Gyroscope Calibration Parameters	R	Gyroscope factory calibration parameters, Scale Factor: 0.001
0xC0-0xCF	192-207	MAG_CAL	Magnetometer Calibration Parameters	R	Magnetometer factory calibration parameters, Scale Factor: 0.001

3.3 Configuration Commands

> The following configuration examples assume a default Modbus address of 0x50. If the Modbus ID has been modified by the user, the ID field and CRC field should be adjusted.

Command	Value to Write to CTL Register	Command (Hex) ID=0X50
Save all configuration parameters to Flash	0x0000	50 06 00 00 00 00 84 4B
Restore factory settings	0x0001	50 06 00 00 00 01 45 8B
Set operating mode to 6-axis mode	0x0003	50 06 00 00 00 03 C4 4A
Set operating mode to 9-axis mode (AHRS)	0x0004	50 06 00 00 00 04 85 88
Set operating mode to pure gyroscope integration mode	0x0005	50 06 00 00 00 05 44 48
Set initial attitude offset (leveling) Pitch = Roll = Yaw = 0, effective immediately and saved even if powered off	0x0010	50 06 00 00 00 10 85 87
Set initial attitude (leveling) Pitch = Roll = 0, Yaw remains unchanged, effective immediately and saved even if powered off	0x0011	50 06 00 00 00 11 44 47
Set initial attitude (leveling) Pitch and Roll remain unchanged, Yaw = 0, effective immediately and saved even if powered off	0x0012	50 06 00 00 00 12 04 46
Clear all initial attitude settings, effective immediately and saved even if powered off	0x0013	50 06 00 00 00 13 C5 86
Set mounting orientation: set to horizontal installation (normal mode)	0x0020	50 06 00 00 00 20 85 93
Set mounting orientation: vertical installation with the positive Y-axis facing downward	0x0021	50 06 00 00 00 21 44 53
Set mounting orientation: vertical installation with the positive Y-axis facing upward	0x0022	50 06 00 00 00 22 04 52
Set mounting orientation: vertical installation with the positive X-axis facing upward	0x0023	50 06 00 00 00 23 C5 92
Set mounting orientation: vertical installation with the positive X-axis facing downward	0x0024	50 06 00 00 00 24 84 50
Reset	0x00FF	50 06 00 00 00 FF C4 0B
Configure baud rate to 4800 (takes effect after reset)	0x0100	50 06 00 00 01 00 85 DB
Configure baud rate to 9600 (takes effect after reset)	0x0101	50 06 00 00 01 01 44 1B
Configure baud rate to 19200 (takes effect after reset)	0x0102	50 06 00 00 01 02 04 1A
Configure baud rate to 38400 (takes effect after reset)	0x0103	50 06 00 00 01 03 C5 DA

Command	Value to Write to CTL Register	Command (Hex) ID=0X50
Configure baud rate to 57600 (takes effect after reset)	0x0104	50 06 00 00 01 04 84 18
Configure baud rate to 115200 (takes effect after reset)	0x0105	50 06 00 00 01 05 45 D8
Configure baud rate to 230400 (takes effect after reset)	0x0106	50 06 00 00 01 06 05 D9
Configure baud rate to 460800 (takes effect after reset)	0x0107	50 06 00 00 01 07 C4 19
Configure baud rate to 921600 (takes effect after reset)	0x0108	50 06 00 00 01 08 84 1D
Configure Modbus ID (takes effect after reset)	0x0200 + ID	Configure Modbus ID to 0x03 Send 50 06 00 00 02 03 C5 2A

3.4 Data Reading Examples

1. Reading the module's product name, software version, and SN number

   TX (Host sends): 50 03 00 70 00 13 08 5D

   ID=0x50, CMD=0x03, read starting address 0x70, read length: 0x13, CRC: 0x085D

   RX (Slave response): 50 03 26 00 43 00 48 00 31 00 30 00 58 00 28 00 4D 00 29 00 73 00 00 00 00 00 00 00 00 00 00 00 00 06 DD C2 9C 6D 06 97 0F 7F 5F

   50 03 26: Slave address 0x50, command code 0x03, data part is 0x26 = 38 bytes, 00 43 00 48 00 31 00 30 00 58 00 28 00 4D 00 29 00 73 00 00 00 00 00 00 00 00 00 00 00 00 06 DD C2 9C 6D 06 97 0F, data segment: Product name: CH10x(M), software version: 0x73 (115), SN: 06DDC29C6D06970F, 7F 5F: CRC check.

2. Reading IMU attitude data

   TX (Host sends): 50 03 00 34 00 18 09 8F

   ID=0x50, CMD=0x03, read starting address 0x34, read length: 0x18, CRC: 0x098F

   RX (Slave response): 50 03 30 FF 01 03 B0 06 50 FC C9 FF 7C 00 91 01 D5 FD DB FD 27 00 00 21 FF 00 00 7F F6 FF FD 73 E7 00 00 00 00 00 00 10 A6 0D 59 DD 4E 86 A8 06 30 17 82 1E CE

   - Acceleration: X = -255, Y = 944, Z = 1616 -> Results after multiplication factor: X = -0.1245, Y = 0.4609, Z = 0.7891G (1G = 1 gravity acceleration, approximately 9.8 m/s^2)
   - Angular velocity: X = -823, Y = -132, Z = 145 -> Results after multiplication factor: X = -50.2318, Y = -8.0566, Z = 8.8501 deg/s
   - Magnetic field: X = 469, Y = -549, Z = -729 -> Results after multiplication factor: X = 14.3125, Y = -16.7538, Z = -22.2469 uT
   - Euler angles: Roll = 8703, Pitch = 32758, Yaw = -166937 -> Results after multiplication factor: Roll = 8.703 deg, Pitch = 32.758 deg, Yaw = -166.937 deg

3.5 Modbus ID Auto Assignment

The ID address auto-assignment mechanism is used for mass deployment when multiple modules are connected to the same 485 bus. Modules can work with a host to complete automatic ID address assignment. This feature is only available to mass production customers. For specific information, please contact us.

The custom instruction format for 0x50 is: [ADDR] 0x50 [SUB_CMD] [DATA_LEN] [DATA]

Currently supported custom instructions include:

1. Setting the ID address via SN number (0x0031): Format: `00 50 00 31 00 0A [SN] [NEW_ADDR] CRC`

   SN: Device unique serial number, 8 bytes

   New ID address: 1-255, 2 bytes

2. Random ID address generation (0x0030): This command forces all modules on the bus to discard their original IDs and generate new IDs between MIN_ADDR and MAX_ADDR. Format: `00 50 00 30 00 06 [MIN] [MAX] FF FF CRC`

   MIN: Minimum value for generating a random ID, 2 bytes.

   MAX: Maximum value for generating a random ID, 2 bytes.

3.6 Firmware Upgrade

The Modbus firmware upgrade protocol is only available to bulk production customers. For specific information, please contact us.

4. CAN Data Protocol and Commands (CANopen)

The CAN interface conforms to the CANopen protocol, and all communication uses standard data frames, with data transmitted via TPDO1-7. It does not receive/send remote frames and extended data frames. All TPDOs operate in asynchronous timed trigger mode.

4.1 CANopen Default Settings

Default Configuration	Value
CAN Baud Rate	500KHz
CANopen Node ID	8
Initialization State	Operational
Heartbeat	None
TPDO Output Rate	1Hz - 200Hz (per TPDO)

4.2 CANopen TPDO(Asynchronous Transmission)

Channel	Frame ID	Data Length (DLC)	Output Frequency (Hz)	Data	Description
TPDO1	0x180+ID	6	100	Acceleration	Type: int16, Low byte first, 2 bytes for each axis, total of 6 bytes For X, Y, and Z-axis acceleration, unit: mG (0.001G)
TPDO2	0x280+ID	6	100	Angular Velocity	Type: int16, Low byte first, 2 bytes for each axis, total of 6 bytes For X, Y, and Z-axis angular velocity, unit: 0.1dps (°/s)
TPDO3	0x380+ID	6	100	Euler Angles	Type: int16, Low byte first, 2 bytes for each axis, total of 6 bytes In order: Roll, Pitch, Yaw. Unit: 0.01°
TPDO4	0x480+ID	8	100	Quaternions	Type: int16, Low byte first, 2 bytes for each element, total of 8 bytes For q_w q_x q_y q_z. The unit is 10,000 times the quaternion value. For example, when the quaternion is 1,0,0,0, the output is 10,000,0,0,0.
TPDO6	0x680+ID	4	20	barometer	Type: int32, total of 4 bytes. Unit: Pa
TPDO7	0x780+ID	8	100	Inclinometer Angles	Type: int32, Low byte first, 4 bytes for each axis, total of 8 bytes Order: X-axis, Y-axis. Unit: 0.01°

Here's an example of interpreting data for acceleration and angular velocity:

Acceleration CAN frame: ID=0x188, DATA = 4A 00 1F 00 C8 03

ID=0x188: Acceleration data frame sent by device with ID 8.

Acceleration X-axis = 0x004A = 74 mG

Acceleration Y-axis = 0x001F = 31 mG

Acceleration Z-axis = 0x03C8 = 968 mG

Angular Velocity CAN frame: ID=0x288, DATA = 15 00 14 01 34 00

ID=0x288: Angular velocity data frame sent by device with ID 8.

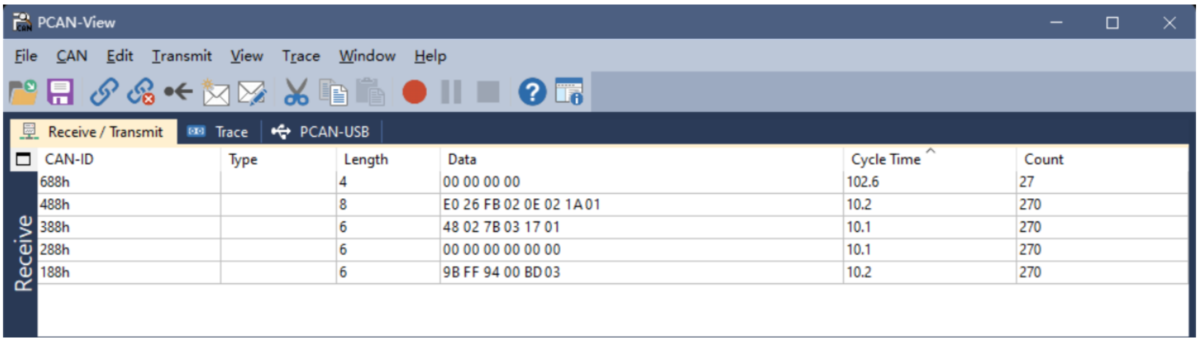
Angular velocity X-axis = 0x0015 = 2.1 dps

Angular velocity Y-axis = 0x0114 = 27.6 dps

Angular velocity Z-axis = 0x0034 = 5.2 dps

4.3 Connecting CAN Devices with Host Software

You can use the PCAN-View tool in combination with PCAN to display received CAN messages and frame rates in the Rx Message window, as shown in the image below.



4.4 Configuration Commands

All configuration changes need to be followed by a "Save Configuration" command to store them in Flash memory.

4.4.1 Global Enable/Disable Data Output (Enable Asynchronous Trigger)

Use the CANopen NMT protocol frames:

Enable global data output: `ID=0x000, DATA=01 08`

Disable global data output: `ID=0x000, DATA=02 08`

The configuration operations above all use quick SDO (Service Data Object) to write data dictionaries. The TPDO channels and their corresponding parameter indexes are as follows:

Channel	Frame ID	Parameter Index Address	Description
TPDO1	0x180+ID	0x1800	Acceleration
TPDO2	0x280+ID	0x1801	Angular Velocity
TPDO3	0x380+ID	0x1802	Euler Angles
TPDO4	0x480+ID	0x1803	Quaternions
TPDO6	0x680+ID	0x1804	barometer
TPDO7	0x780+ID	0x1805	Inclinometer Output

4.4.2 CAN Baud Rate

To modify the baud rate, use Layer Setting Services (LSS) layers with the basic format:

- `0x07E4 = LSS slave to LSS master`
- `0x07E5 = LSS master to LSS slave`

Change the CAN baud rate (takes effect after a power cycle):

- Set CAN baud rate to 1000 kbit/s: `ID=0x07E5, DATA=13,00,00,00,00,00,00,00`
- Set CAN baud rate to 800 kbit/s: `ID=0x07E5, DATA=13,00,01,00,00,00,00,00`
- Set CAN baud rate to 500 kbit/s: `ID=0x07E5, DATA=13,00,02,00,00,00,00,00`
- Set CAN baud rate to 250 kbit/s: `ID=0x07E5, DATA=13,00,03,00,00,00,00,00`
- Set CAN baud rate to 125 kbit/s: `ID=0x07E5, DATA=13,00,04,00,00,00,00,00`
- Set CAN baud rate to 100 kbit/s: `ID=0x07E5, DATA=13,00,05,00,00,00,00,00`
- Set CAN baud rate to 50 kbit/s: `ID=0x07E5, DATA=13,00,06,00,00,00,00,00`
- Set CAN baud rate to 20 kbit/s: `ID=0x07E5, DATA=13,00,07,00,00,00,00,00`
- Set CAN baud rate to 10 kbit/s: `ID=0x07E5, DATA=13,00,08,00,00,00,00,00`

4.4.3 Modify Node ID

`ID=0x07E5, DATA=11, ID, 00, 00, 00, 00, 00, 00`

> ID range for modification: 1-64. After applying, send a "Start Node" command (e.g., change the node start command data to 01 09) and SDO commands (when sending CAN frames with ID 0x609, be aware of the new address).

4.4.4 Save Configuration

`ID=0x07E5, DATA=17, 00, 00, 00, 00, 00, 00, 00`

> After making all configuration changes, send the "Save Configuration" command to store them in Flash memory.

4.4.5 Modify/Disable/Enable Data Output Rate

This configuration takes immediate effect:

- `ID=0x608, DATA=2B,00,18,05,00,00,00,00` to disable acceleration output.
- `ID=0x608, DATA=2B,00,18,05,05,00,00,00` to set acceleration output to 200Hz.
- `ID=0x608, DATA=2B,00,18,05,0A,00,00,00` to set acceleration output to 100Hz.
- `ID=0x608, DATA=2B,00,18,05,14,00,00,00` to set acceleration output to 50Hz.
- `ID=0x608, DATA=2B,00,18,05,32,00,00,00` to set acceleration output to 20Hz.
- `ID=0x608, DATA=2B,00,18,05,64,00,00,00` to set acceleration output to 10Hz (minimum 10Hz).
- `ID=0x608, DATA=2B,01,18,05,00,00,00,00` to disable angular velocity output.
- `ID=0x608, DATA=2B,01,18,05,05,00,00,00` to set angular velocity output to 200Hz.
- `ID=0x608, DATA=2B,01,18,05,0A,00,00,00` to set angular velocity output to 100Hz.
- `ID=0x608, DATA=2B,01,18,05,14,00,00,00` to set angular velocity output to 50Hz.
- `ID=0x608, DATA=2B,01,18,05,32,00,00,00` to set angular velocity output to 20Hz.
- `ID=0x608, DATA=2B,01,18,05,64,00,00,00` to set angular velocity output to 10Hz (minimum 10Hz).
- `ID=0x608, DATA=2B,02,18,05,00,00,00,00` to disable Euler angle output.
- `ID=0x608, DATA=2B,02,18,05,05,00,00,00` to set Euler angle output to 200Hz.
- `ID=0x608, DATA=2B,02,18,05,0A,00,00,00` to set Euler angle output to 100Hz.
- `ID=0x608, DATA=2B,02,18,05,14,00,00,00` to set Euler angle output to 50Hz.
- `ID=0x608, DATA=2B,02,18,05,32,00,00,00` to set Euler angle output to 20Hz.
- `ID=0x608, DATA=2B,02,18,05,64,00,00,00` to set Euler angle output to 10Hz (minimum 10Hz).
- `ID=0x608, DATA=2B,03,18,05,00,00,00,00` to disable quaternion output.
- `ID=0x608, DATA=2B,03,18,05,05,00,00,00` to set quaternion output to 200Hz.
- `ID=0x608, DATA=2B,03,18,05,0A,00,00,00` to set quaternion output to 100Hz.
- `ID=0x608, DATA=2B,03,18,05,14,00,00,00` to set quaternion output to 50Hz.
- `ID=0x608, DATA=2B,03,18,05,32,00,00,00` to set quaternion output to 20Hz.
- `ID=0x608, DATA=2B,03,18,05,64,00,00,00` to set quaternion output to 10Hz (minimum 10Hz).
- `ID=0x608, DATA=2B,04,18,05,00,00,00,00` to disable barometer output.
- `ID=0x608, DATA=2B,04,18,05,05,00,00,00` to set barometer output to 200Hz.
- `ID=0x608, DATA=2B,04,18,05,0A,00,00,00` to set barometer output to 100Hz.
- `ID=0x608, DATA=2B,04,18,05,14,00,00,00` to set barometer output to 50Hz.
- `ID=0x608, DATA=2B,04,18,05,32,00,00,00` to set barometer output to 20Hz.
- `ID=0x608, DATA=2B,04,18,05,64,00,00,00` to set barometer output to 10Hz (minimum 10Hz).

As an example, to set TPDO1 (acceleration) output rate to 100Hz (every 10ms): 0x2B represents an SDO write two bytes command. 0x00, 0x18 is the write index 0x1800. 0x05 is the sub-index. 0x00, 0x0A is calculated as (0x00 << 8) + 0x0A, which is 10 (unit: ms), with trailing zeros.

4.4.6 Configure TPDO for Synchronous Mode

- Disable all TPDOs (set TPDO output rate to 0).
- Send a CANopen sync frame. CANopen sync frame: ID: 80, DATA: Empty

5. CAN数据协议与指令(SAE-J1939)

The module's default output protocol is CANOpen. If you require the SAE J1939 protocol, please contact our company.

PGN	Description
Communication Mode	Broadcast Communication
Default Transmission Interval	100ms
Data Length	8 bytes per PGN
PF(PDU format)	0xFF
PS(PDU specific)	GE when PF > 0xF0, DA otherwise
Priority	3
Default J1939 Address	0x08
Data Format	All data in frames is in LSB (Least Significant Bit) order and is signed integer unless otherwise specified.

5.1 PGN Message List

5.1.1 PGN65332 (FF34) Acceleration

CANID=0x0CFF3408

Name	Position (byte)	Description
Acceleration X	0-1	Unit: G (1G = 1 gravitational acceleration), Scale Factor: 0.00048828
Acceleration Y	2-3	Unit: G (1G = 1 gravitational acceleration), Scale Factor: 0.00048828
Acceleration Z	4-5	Unit: G (1G = 1 gravitational acceleration), Scale Factor: 0.00048828
Reserved	6-7	-

5.1.2 PGN65335 (FF37) Angular Velocity

CANID=0x0CFF3708

Name	Position (byte)	Description
Angular Velocity X	0-1	Unit: deg/s, Scale Factor: 0.061035
Angular Velocity Y	2-3	Unit: deg/s, Scale Factor: 0.061035
Angular Velocity Z	4-5	Unit: deg/s, Scale Factor: 0.061035
Reserved	6-7	-

5.1.3 PGN65354 (FF4A) Inclinometer Output

CANID=0x0CFF4A08 (Only applicable to inclinometer products with J1939 protocol output)

Name	Position (byte)	Description
X Inclination Angle	0-3	Range: 0-360 or ±180, Unit: degrees, Scale Factor: 0.001
Y Inclination Angle	4-7	Range: 0-360 or ±180, Unit: degrees, Scale Factor: 0.001

5.2 Configuration Commands

5.2.1 Command Format

Host sends: ADDR+ CMD + STATUS + VAL , Slave responds: ADDR+ CMD + STATUS + VAL

Field	Size (Bytes)	Description
ADDR	2	Register Address
CMD	1	0x06: Write, 0x03: Read
STATUS	1	Reserved
VAL	4	Value to write or read

5.2.2 Command Collection

29-bit Extended Frame Address	Data	Description	Explanation
0x0CEF08xx	34 00 06 00 [VAL]	VAL: 4 bytes	PGN: FF34 transmission interval, in ms, range: 10-1000
0x0CEF08xx	37 00 06 00 [VAL]	VAL: 4 bytes	PGN: FF37 transmission interval, in ms, range: 10-1000
0x0CEF08xx	4A 00 06 00 [VAL]	VAL: 4 bytes	PGN: FF4A transmission interval, in ms, range: 10-1000
0x0CEF08xx	00 00 06 00 00 00 00 00	-	Save all configuration parameters to Flash
0x0CEF08xx	00 00 06 00 01 00 00 00	-	Restore factory settings
0x0CEF08xx	00 00 06 00 FF 00 00 00	-	Reset

> In the address field, xx represents the source address in the J1939 protocol, which can be any byte.
> In the data field, xx represents any byte.

6. Technical Support

New products and information can be obtained through our website and official public account.

Website:www.hipnuc.com

Sales:sales@hipnuc.com

Support:support@hipnuc.com