

Command and Programming Manual

Rev 1.3.2

Applicable to CH0x0/HI02/04/05/06/14/50/90/70 Series



File: imu_cum_en

Technical Support: support@hipnuc.com

Property: Public

Website: www.hipnuc.com



(C) 2016-2025 Beijing Hipnuc Electronic Technology Co., Ltd. All rights reserved. The information contained in this document may change without prior notice.

Command and Programming Manual

- 1 Module Configuration Overview
 - 1.1 Work Mode Configuration - AHRS/9-Axis Mode
 - 1.2 Synchronous Input - SYNC_IN/PPS
 - 1.2.1 SYNC_IN Data Trigger
 - 1.2.2 PPS Pulse Input
 - 1.3 Synchronous Output - SOUT
 - 1.4 Vessel Heave/Surge/Sway Output
 - 1.4.1 Usage Limitations
- 2 Product Feature Support Matrix
- 3 Module Configuration Commands (ASCII)
 - 3.1 Command Overview
 - 3.2 Command Details
 - 3.2.1 REBOOT
 - 3.2.2 SAVECONFIG
 - 3.2.3 SERIALCONFIG
 - 3.2.4 CONFIG
 - 3.2.4.1 Work Mode Configuration
 - 3.2.4.2 User Attitude Calibration
 - 3.2.4.3 Coordinate Rotation (Vertical Installation)
 - 3.2.4.4 Multi-Function IO (PMUX) Mapping
 - 3.2.4.5 SOUT (PMUX2) Divider
 - 3.2.4.6 Magnetometer Calibration
 - 3.2.4.7 User Gyroscope Calibration
 - 3.2.5 LOG
 - 3.2.5.1 Enable/Disable Output
 - 3.2.5.2 Module Version
 - 3.2.5.3 UART Configuration Snapshot
 - 3.2.5.4 Configure Frame Type and Rate
 - 3.2.6 FRESET
- 4 RS-232/TTL/USB Binary Protocol
 - 4.1 Data Frame Format
 - 4.2 Factory Default Output
 - 4.3 Payload Content
 - 4.3.1 Floating-Point IMU Packet (HI91)
 - 4.3.2 MAIN_STATUS Bit Definitions
 - 4.4 CRC Verification
 - 4.5 Data Frame Example (HI91)
 - 4.6 C Parsing Example (HI91)
 - 4.6.1 CRC Check
 - 4.6.2 Data Structures
 - 4.6.3 Receive Payload
 - 4.6.4 Print Results
 - 4.7 Maximum Throughput

- 5 RS-485 Output Protocol (Modbus)
 - 5.1 Modbus Command Overview
 - 5.2 Frame Structure
 - 5.2.1 Read Registers (0x03)
 - 5.2.2 Write Register (0x06)
 - 5.3 Register List
 - 5.4 Common Configurations
 - 5.4.1 Save All Parameters to Flash
 - 5.4.2 Restore Factory Defaults
 - 5.4.3 Reset
 - 5.4.4 Configure Baud Rate (Register 0x04)
 - 5.4.5 Configure Node ID (Register 0x05)
 - 5.4.6 Installation Mode (Register 0xA6)
 - 5.4.7 Horizontal Calibration (Register 0xA5)
 - 5.4.8 6-Axis vs. 9-Axis Mode (Register 0x06)
 - 5.5 Read Module Version (0x70-0x82)
 - 5.6 Read Sensor Data (0x34-0x4B)
- 6 CAN Data Protocol (J1939)
 - 6.0.1 CAN Extended Frame Format
 - 6.0.2 Endianness
 - 6.1 Protocol Parameters
 - 6.2 PGN List
 - 6.2.1 PGN 65327 (0xFF2F) - Time Information
 - 6.2.2 PGN 65332 (0xFF34) - Acceleration
 - 6.2.3 PGN 65335 (0xFF37) - Angular Velocity
 - 6.2.4 PGN 65341 (0xFF3D) - Roll & Pitch
 - 6.2.5 PGN 65345 (0xFF41) - Heading
 - 6.2.6 PGN 65338 (0xFF3A) - Magnetic Field
 - 6.2.7 PGN 65350 (0xFF46) - Quaternion
 - 6.2.8 PGN 65354 (0xFF4A) - Inclinometer Output
 - 6.3 Configuration Protocol
 - 6.3.1 Frame Format
 - 6.3.2 Register Mapping
 - 6.4 Configuration Examples (default node ID = 8)
 - 6.5 Time Synchronization
 - 6.5.1 Principle
 - 6.5.2 Procedure
 - 6.5.3 Example
- 7 CAN Data Protocol (CANopen)
 - 7.1 Default Settings
 - 7.2 CANopen TPDO Mapping
 - 7.2.1 Data Example
 - 7.3 Connecting via PC Software
 - 7.4 Configuration Commands (SDO)

- 7.4.1 Expedited SDO Format
- 7.4.2 Common Commands
 - 7.4.2.1 Change Node ID (0x20A0)
 - 7.4.2.2 Save to Flash (0x2000)
 - 7.4.2.3 Reset (0x2000)
 - 7.4.2.4 Restore Factory Defaults (0x2000)
 - 7.4.2.5 Change CAN Baud (0x209A)
- 7.4.3 TPDO Configuration
 - 7.4.3.1 Modify / Disable / Enable Output Rate (sub-index 5)
 - 7.4.3.2 Inclinator Polarity (0x209E, 0x209F)
 - 7.4.3.3 Inclinator Zero (0x20A5)
- 7.4.4 Sync Mode
- 7.4.5 Heartbeat
- 8 Magnetic Calibration
 - 8.1 When to Calibrate
 - 8.2 Calibration Steps
 - 8.3 Troubleshooting
 - 8.4 FAQ
 - 8.5 Calibration Frequency & Mode Selection
- 9 Appendix 1 Quaternion / Euler / Rotation Matrix
 - 9.1 Quaternion to Rotation Matrix
 - 9.2 Quaternion <-> Euler (ENU, 312 sequence: Z -> X -> Y)
 - 9.3 Quaternion <-> Euler (NED, 321 sequence: Z -> Y -> X)
- 10 Appendix 2 Firmware Upgrade
- 11 Appendix 3 URFR Derivation
- 12 Appendix 4 Understanding Magnetic Interference
- 13 Appendix 5 Technical Support

1. Module Configuration Overview

Read this chapter carefully before operating the product and check whether additional user configuration is required for your scenario.

1.1 Work Mode Configuration - AHRS/9-Axis Mode

! Warning

AHRS (9-axis) mode is vulnerable to environmental magnetic field interference, especially around robots or in indoor environments. Distorted magnetic fields will translate directly into heading errors.

You may enable magnetometer aiding only in open areas with minimal magnetic distortion (for example, outdoor UAV flights). Before enabling this mode, make sure to:

1. Switch the module to magnetometer-aided mode.
2. Perform magnetic calibration (see the "Magnetic Calibration" chapter).

1.2 Synchronous Input - SYNC_IN/PPS

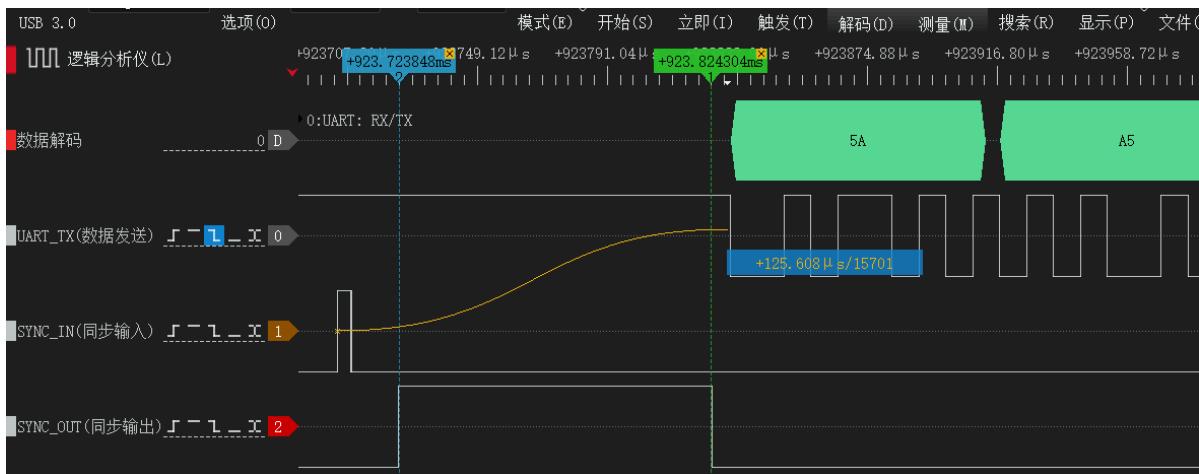
Data/time synchronization relies on the SYNC_IN/PPS pin. Two trigger modes are available:

- SYNC_IN edge trigger for data output
- PPS (pulse-per-second) for UTC time alignment

1.2.1 SYNC_IN Data Trigger

Some products expose a SYNC_IN/PPS pin for IMU synchronization. Leave the pin floating or tie it to ground when not in use. When the output protocol is configured to ONMARK (see the LOG command), the module outputs one frame each time a rising edge is detected on SYNC_IN.

Typical use case: connect the pin to a controller that generates a high-precision square wave so that each pulse triggers a high-frequency, phase-aligned IMU sample.



1.2.2 PPS Pulse Input

The same SYNC_IN/PPS pin can accept a GPS PPS signal. Together with an RMC (or GGA) sentence on the UART, the PPS pulse provides accurate UTC alignment. Once the module receives valid PPS edges and RMC data, its timestamp becomes UTC-based.

1. SYNC_IN/PPS electrical requirements

- Period: 1 s
- Trigger edge: rising edge
- Pulse width: 1~100 ms
- Alignment: rising edge aligned to UTC integer seconds
- Logic level: ≤ 5 V (TTL/CMOS compatible)

2. UART input requirements

- NMEA update rate: 1~10 Hz
- Baud rate: identical to the IMU UART setting
- Supported messages: RMC or GGA only

Example GPRMC sentence:

```
,020550.00,A,2813.9891299,N,11252.6278784,E,0.033,315.7,161117,0.0,E,A*30
```

Timestamp format after synchronization: once UTC is aligned, the timestamp shown in each data frame (uint32_t) is the millisecond counter since 00:00:00 UTC of the current day.

Parameter	Value
Range	0~86,399,999 ms
Interval	00:00:00.000 to 23:59:59.999
Accuracy	1 ms

Timestamp conversion examples and C helper:

Timestamp (ms)	UTC Time
0	00:00:00.000
3,661,000	01:01:01.000
43,200,000	12:00:00.000
86,399,999	23:59:59.999

```
`c
```

```
// Time conversion formula:
```

```
// Hours = timestamp / 3600000
```

```
// Minutes = (timestamp % 3600000) / 60000
```

```
// Seconds = (timestamp % 60000) / 1000
```

```
// Milliseconds = timestamp % 1000
```

```
void ms_to_utc_time(uint32_t total_ms, char *utc_time, int buf_size)
```

```
{
```

```
    uint32_t total_seconds = total_ms / 1000;
```

```
    uint32_t ms_part = total_ms % 1000;
```

```
1  uint8_t hours = (total_seconds / 3600) % 24;
2  uint8_t minutes = (total_seconds % 3600) / 60;
3  uint8_t seconds = total_seconds % 60;
4
5  sprintf(utc_time, buf_size, "%02d:%02d:%02d.%03d",
6          hours, minutes, seconds, ms_part);
```

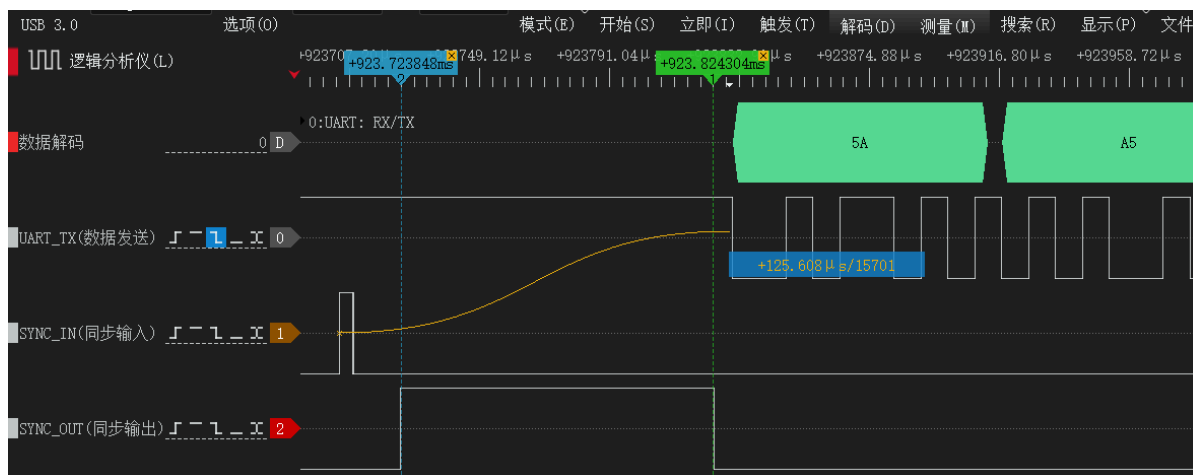
```
}
```

Note Notes

- The timestamp contains no date information. It only counts milliseconds since UTC 00:00:00 of the current day.
- The value is UTC+0 with no time-zone offset. Apply the local time offset yourself when needed.

1.3 Synchronous Output - SOUT

SOUT is an output pin. It stays low while idle and emits a high pulse immediately before each frame is transmitted. Data bytes follow the falling edge of that pulse.



The pin also supports clock division. You can configure it to output one pulse every 2/4/8/16 frames so that low-rate devices (for example cameras) receive precise triggers. See the "Module Configuration" chapter for details.

1.4 Vessel Heave/Surge/Sway Output

Supported products can estimate the vessel motion induced by waves (heave, surge, sway) and provide real-time displacement, velocity, and period for all three axes.

Axis	Description	Output
Heave	Vertical heave	Vertical displacement, velocity, and period
Surge	Longitudinal	Longitudinal displacement, velocity, period
Sway	Lateral sway	Lateral displacement, velocity, and period

Typical applications

- Vessel motion monitoring and attitude stabilization
- Offshore lifting, docking, and motion compensation
- Wave statistics and research
- Offshore platform stability analysis
- Assisting dynamic positioning (DP) systems

1.4.1 Usage Limitations

! Critical limitations

- Only valid for periodic reciprocating motion. The algorithm cannot handle:
 - Motions with periods longer than 30 s
 - One-direction linear motion
 - Step-type displacement
- **Zero-mean assumption:** long-term displacement must average to zero.
- **Initialization:** allow a steady **5-20 wave cycles** to initialize before using the output.
- **Frequency response:** optimized for 3-20 s wave periods; accuracy degrades outside this range.

Note **Product compatibility**

Heave/surge/sway outputs are only available on HI7x/HI9x series devices (see the "Product Feature Support" table).

2. Product Feature Support Matrix

The following table lists product-specific command availability. Commands that are not listed are supported by all models.

Model / Command	HI02	HI05/06	CH0x0, HI04/12/13/14	HI50	HI7x/9x
Magnetometer aiding / calibration	No	Yes	Yes	No	Yes
User attitude alignment	No	Yes	Yes	Yes	Yes
IO pin mux (PMUX)	No	Yes	Yes	No	No
CAN interface	No	Yes	Yes	Yes	Yes
Vessel heave output	No	No	No	No	Yes
Inclinometer output	No	No	No	Yes	No

3. Module Configuration Commands (ASCII)

Module parameters are configured through ASCII commands sent over the UART. Each command must end with `\r\n` (similar to AT commands). All configuration changes take effect only after a reset or power cycle unless explicitly stated otherwise.

3.1 Command Overview

Command	Function	Remarks
REBOOT	Reset the module	Same as a power cycle
SAVECONFIG	Save all configuration parameters	Takes effect immediately
SERIALCONFIG	Configure the current UART baud rate	Takes effect immediately
CONFIG	Configure operating modes and features	Save + reboot to take effect
LOG	Query module info or configure outputs	Takes effect immediately
FRESET	Restore factory defaults	Takes effect immediately

3.2 Command Details

3.2.1 REBOOT

Reset the module. Example: REBOOT

3.2.2 SAVECONFIG

Store all user configurations to flash memory. Example: SAVECONFIG

3.2.3 SERIALCONFIG

Set the current UART baud rate. Format: SERIALCONFIG . Supported baud rates: 9600, 115200, 230400, 256000, 460800, 921600.

Example:

- SERIALCONFIG 115200 - switch the active UART to 115200 bps

! Important

The new baud rate takes effect immediately. Switch the host interface to the same rate before sending further commands.

3.2.4 CONFIG

CONFIG changes module behavior. Run SAVECONFIG and then reset (or power cycle) to apply the new settings.

3.2.4.1 Work Mode Configuration

Switch between 6-axis and 9-axis (magnetometer-aided) attitude solutions. Format: CONFIG ATT MODE .

- CONFIG ATT MODE 0 - VRU (6-axis) mode
- CONFIG ATT MODE 1 - AHRS (9-axis) mode
- CONFIG ATT MODE 4 - Humanoid robot mode

3.2.4.2 User Attitude Calibration

Set relative attitude angles. Format: CONFIG ATT RST . Keep the module stationary when issuing the command to avoid errors.

- CONFIG ATT RST 1 - Heading reset (set yaw to 0deg)
- CONFIG ATT RST 2 - Relative zero (set current pitch/roll to 0deg)
- CONFIG ATT RST 3 - Auto-level:
 - If pitch/roll are near 0deg (flat, right-side-up), level to Pitch=0deg, Roll=0deg
 - If pitch~0deg and roll~180deg (flat, upside-down), level to Pitch=0deg, Roll=180deg
 - "Near" means |Pitch| and |Roll| < 15deg
- CONFIG ATT RST 5 - Cancel leveling: clear relative pitch/roll offsets and revert to absolute angles

3.2.4.3 Coordinate Rotation (Vertical Installation)

Use this command to describe non-standard mounting orientations (for example vertical mounting). Save and reboot after configuration. There is no need to resend the command every time the module powers up.

Format: CONFIG IMU URFR <C00,C01,C02,C10,C11,C12,C20,C21,C22>

Each (C_{nn}) accepts floating-point values.

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

- $(\{X,Y,Z\}_U)$: sensor data in the **rotated** (user) frame
- $(\{X,Y,Z\}_B)$: sensor data in the original body frame

Common presets:

No.	Mounting pose	Command	Notes
0	Horizontal, Z-up (default)	CONFIG IMU URFR 1,0,0,0,1,0,0,0,1	Factory default
1	Rotate -90deg about X (Y+ points downward)	CONFIG IMU URFR 1,0,0,0,0,1,0,-1,0	Typical vertical mounting
2	Rotate +90deg about X (Y+ points upward)	CONFIG IMU URFR 1,0,0,0,0,-1,0,1,0	Typical vertical mounting
3	Rotate +90deg about Y (X+ points upward)	CONFIG IMU URFR 0,0,-1,0,1,0,1,0,0	Typical vertical mounting
4	Rotate -90deg about Y (X+ points downward)	CONFIG IMU URFR 0,0,1,0,1,0,-1,0,0	Typical vertical mounting
5	Rotate 180deg about X	CONFIG IMU URFR 1,0,0,0,-1,0,0,0,-1	Upside-down installation
6	Rotate 180deg about Y	CONFIG IMU URFR -1,0,0,0,1,0,0,0,-1	Upside-down installation
7	Rotate +90deg about Z (combo nav, -X forward)	CONFIG IMU URFR 0,-1,0,1,0,0,0,0,1	Common for navigation units
8	Rotate -90deg about Z (combo nav, +X forward)	CONFIG IMU URFR 0,1,0,-1,0,0,0,0,1	Common for navigation units
9	Rotate 180deg about Z (combo nav, -Y forward)	CONFIG IMU URFR -1,0,0,0,-1,0,0,0,1	Common for navigation units

3.2.4.4 Multi-Function IO (PMUX) Mapping

Each IOx pin has a default function but can be reassigned in software. Format: CONFIG .

Parameters:

- **PMUX** - function tag (PMUX1-PMUX3)
- **IO** - pin index (IO1-IO5)

Function	Name	Dir	Description	Default IO
PMUX1	SIN/PPS	I	Sync pulse input (see "Synchronous Input and Output")	IO1
PMUX2	SOUT	O	Sync pulse output (high pulse before each frame)	IO2
PMUX3	LED	O	Status LED	IO3

Examples:

- CONFIG PMUX3 IO2 - route the LED function to IO2
- CONFIG PMUX2 IO1 - route the sync output (SOUT) to IO1

Note **Note**

Not every IO pin is populated on every product. Refer to the hardware section of the user manual for pin availability.

3.2.4.5 SOUT (PMUX2) Divider

Divide the SOUT pulse rate to drive slower peripherals such as cameras. Format: CONFIG PMUX2 DIV .

- **N** - divider, range 1-100, default 1 (no division)

Example: CONFIG PMUX2 DIV 5 -> with a 100 Hz data rate, SOUT outputs a 20 Hz pulse train.

3.2.4.6 Magnetometer Calibration

Start a manual mag calibration: CONFIG MCAL START. After the command is issued, the module collects samples automatically-no explicit stop command is required. Rotate the module slowly in place and keep all rigidly attached parts (PCB, housing, robot, etc.) fixed relative to the IMU. See the "Magnetic Calibration" chapter for full guidance.

Check calibration status with LOG MCAL STAT. Example response:

```
STAT=1
```

```
PROGRESS=8
```

```
OK
```

- **STAT** - 0: idle, 1: calibrating, 3: calibration complete, 4: calibration failed
- **PROGRESS** - 0-100, calibration progress

Note **Requirement**

Magnetometer calibration commands require firmware $\geq 1.7.0$.

3.2.4.7 User Gyroscope Calibration

This procedure calibrates the Z-axis gyroscope scale factor and improves long-term heading accuracy (post-calibration scale factor error $\leq 0.1\%$). Recommended for AGVs or large robots that demand low heading drift. Format: CONFIG USRCAL START .

- **ANGLE** - desired rotation angle, 720deg-1800deg (2-5 turns). More turns reduce statistical error but must maintain uniform speed.

Procedure:

1. Place the module (together with the rigidly attached platform) on a flat, level surface. Ensure stable temperature and avoid vibration.
2. Send the start command, e.g., CONFIG USRCAL START 720.
3. Rotate the system on the horizontal plane according to at 20-100deg/s (roughly 5-20 s per turn). Keep the speed uniform, avoid pauses or sudden acceleration.

4. After completing the rotation (actual angle within +/-5deg of the target), send CONFIG USRCAL STOP.
The device replies OK on success or ERR on failure.

Common reasons for failure:

Cause	Description
Improper rotation	Module not kept level or stable
Poor handling	Rotation too fast/slow, pauses mid-way
Environmental vibration	Strong vibration from nearby machines or actuators
Larger error after calibration	Likely due to inaccurate rotation. For example, targeting 720deg but rotating 725deg introduces a 0.6% scale error (5/720).

3.2.5 LOG

3.2.5.1 Enable/Disable Output

- LOG ENABLE - globally enable data frame output
- LOG DISABLE - globally disable data frame output

3.2.5.2 Module Version

LOG VERSION - print firmware information

3.2.5.3 UART Configuration Snapshot

LOG COMCONFIG - print UART and output protocol settings

3.2.5.4 Configure Frame Type and Rate

Format: LOG

- **MSG** - frame type (HI91, etc.), see "RS-232/TTL/USB Binary Protocol"
- **TYPE** - ONTIME for periodic output, ONMARK for SYNC_IN/PPS trigger
- **VALUE** - period in seconds, 0.001 (1 kHz) to 1 (1 Hz)

Examples:

- LOG HI91 ONTIME 0.01 - output HI91 at 100 Hz
- LOG HI91 ONTIME 0.05 - output HI91 at 20 Hz
- LOG HI91 ONTIME 0 - stop HI91 output
- LOG HI91 ONMARK 1 - switch HI91 to SYNC_IN/PPS trigger mode
- LOG HI91 ONMARK ONCE - trigger one HI91 frame immediately (same effect as a PPS edge)

! Bandwidth Note

High frame rates (e.g., 500 Hz) cannot be delivered at 115200 bps. Increase the baud rate (for example to 921600) before enabling high-rate output.

3.2.6 FRESET

FRESET - restore all configurable parameters, including baud rate, to factory defaults. The command takes effect immediately. Use with caution.

4. RS-232/TTL/USB Binary Protocol

RS-232, UART-TTL, and USB (virtual COM) ports all provide the same stream-style interface and support Hipnuc's proprietary binary protocol.

4.1 Data Frame Format

Field	Value	Length (bytes)	Description
SOF	5A A5	2	Start-of-frame sync word
LEN	1-4096	2	Payload length (LSB first). Does not include header/type/CRC.
CRC	-	2	CRC-16 applied to header, length, and payload (excludes the CRC field itself)
Payload	-	1-4096	One or more sub-packets. Each sub-packet has a tag and data block that define its type and length.

4.2 Factory Default Output

Default frame: floating-point IMU packet HI91.

4.3 Payload Content

4.3.1 Floating-Point IMU Packet (HI91)

The payload is 76 bytes and contains module ID, temperature, raw IMU data, magnetometer, barometer, and fused attitude.

Offset	Name	Type	Size (bytes)	Unit	Scale	Description
0	tag	uint8_t	1	-	-	Packet tag = 0x91
1	main_status	uint16_t	2	-	-	Status word (see below)
3	temperature	int8_t	1	degC	1	Module average temperature
4	air_pressure	float	4	Pa	1	Barometric pressure
8	system_time	uint32_t	4	ms	1	Local timestamp before GPS sync; UTC timestamp after sync
12	acc_b	float	4x3	G	1	Calibrated accelerometer XYZ (1G ~ 9.8 m/s ²)
24	gyr_b	float	4x3	deg/s	1	Calibrated gyroscope XYZ
36	mag_b	float	4x3	uT	1	Magnetic field XYZ
48	roll	float	4	deg	1	Roll angle
52	pitch	float	4	deg	1	Pitch angle
56	yaw	float	4	deg	1	Heading
60	quat	float	4x4	-	-	Quaternion (WXYZ order)

4.3.2 MAIN_STATUS Bit Definitions

Bits	Name	Description
0-2	Reserved	-
3	WB_CONV	Bias convergence alarm. 1: bias estimate accuracy is poor; keep the module still for 3-5 s to regain heading accuracy. 0: bias estimate is healthy.
4	MAG_DIST	Magnetic anomaly alarm. 1: detected interference or uncalibrated magnetometer. Heading may drift in 9-axis mode. 0: Magnetic environment OK, or the system is in 6-axis mode.
5-9	Reserved	-
10	MAG_AIDING	Magnetometer aiding flag. 1: magnetometer contributes to heading (9-axis). 0: magnetometer is not used (6-axis).
11	UTC_TIME	Time sync flag. 1: UTC not synchronized (local time). 0: UTC synchronized.
12	SOUT_PULSE	Indicates whether this frame coincides with an SOUT pulse output. 1: frame is aligned with an SOUT pulse. 0: frame is not aligned with SOUT.
13-15	Reserved	-

Note **Compatibility**

The status word is available on firmware $\geq 1.7.0$.

4.4 CRC Verification

CRC uses the CRC-16/CCITT polynomial. Example C implementation:

```
/*
   currentCrc: previous CRC value (set to 0 for the first block)
   src: byte stream
   lengthInBytes: number of bytes
*/
static void crc16_update(uint16_t *currentCrc, const uint8_t *src, uint32_t
lengthInBytes)
{
    uint32_t crc = *currentCrc;
    for (uint32_t j = 0; j < lengthInBytes; ++j)
    {
        crc ^= src[j] << 8;
        for (uint32_t i = 0; i < 8; ++i)
        {
```

```
uint32_t temp = crc << 1;
if (crc & 0x8000)
{
temp ^= 0x1021;
}
crc = temp;
}
}
*currentCrc = crc;
}
```

4.5 Data Frame Example (HI91)

A single HI91 frame captured by a serial console contains 82 bytes. The first six bytes hold the header, length, and CRC; the remaining 76 bytes form the payload. Suppose the data is stored in a C array `buf`:

```
5A A5 4C 00 14 BB 91 08 15 23 09 A2 C4 47 08 15 1C 00 CC E8 61 BE 9A 35 56 3E 65 EA
72 3F 31 D0 7C BD 75 DD C5 BB 6B D7 24 BC 89 88 FC 40 01 00 6A 41 AB 2A 70 C2 96 D4
50 41 ED 03 43 41 41 F4 F4 C2 CC CA F8 BE 73 6A 19 BE F0 00 1C 3D 8D 37 5C 3F
```

Decoded fields

Field	Type	Raw	Value	Description
Header	-	5A A5	-	Frame header
Payload size	-	4C 00	0x004C = 76	Payload length
CRC	-	14 BB	0xBB14	CRC
tag	-	91	0x91	Packet tag (payload starts here)
main_status	uint16_t	08 15	0x1508	Status word
temperature	int8_t	23	35	degC
air_pressure	float	09 A2 C4 47	100676	Pa
system_time	uint32_t	08 15 1C 00	1,840,392	ms
acc_b_x	float	CC E8 61 BE	-0.220615	G
acc_b_y	float	9A 35 56 3E	0.209189	G
acc_b_z	float	65 EA 72 3F	0.948889	G
gyr_b_x	float	31 D0 7C BD	-0.061722	deg/s
gyr_b_y	float	75 DD C5 BB	-0.00603836	deg/s
gyr_b_z	float	6B D7 24 BC	-0.0100611	deg/s
mag_b_x	float	89 88 FC 40	7.89167	uT
mag_b_y	float	01 00 6A 41	14.625	uT
mag_b_z	float	AB 2A 70 C2	-60.0417	uT
roll	float	96 D4 50 41	13.0519	deg
pitch	float	ED 03 43 41	12.1885	deg
yaw	float	41 F4 F4 C2	-122.477	deg
q_w	float	CC CA F8 BE	-0.485922	-
q_x	float	73 6A 19 BE	-0.14982	-
q_y	float	F0 00 1C 3D	0.0380868	-
q_z	float	8D 37 5C 3F	0.860223	-

4.6 C Parsing Example (HI91)

Reference implementation:

4.6.1 CRC Check

```
`c
int16_t payload_len;
uint16_t crc = 0;
payload_len = buf[2] + (buf[3] << 8);

/* Header + LEN */
crc16_update(&crc, buf, 4);

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

The resulting CRC (0xBB14) matches the in-frame value, so the frame is valid.

4.6.2 Data Structures

```
`c
#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 */
    float acc[3]; /* Acceleration */
    float gyr[3]; /* Angular velocity */
    float mag[3]; /* Magnetic field */
    float eul[3]; /* Attitude: Euler angle */
    float quat[4]; /* Attitude: quaternion */
    float pressure; /* Air pressure */
    uint32_t timestamp; /* Timestamp */
} imu_data_t;
`
```

4.6.3 Receive Payload

```
`c
imu_data_t i0x91 = {0};
int offset = 6; // Payload starts at buf[6]

i0x91.tag =      U1(buf+offset+0);
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);
`
```

4.6.4 Print Results

```
c
printf("%-16s0x%X\r\n", "tag:", i0x91.tag);
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", "pressure(pa):", i0x91.pressure);
printf("%-16s%d\r\n", "timestamp(ms):", i0x91.timestamp);
```

4.7 Maximum Throughput

Frame	Bytes	9600 bps	115200 bps	230400 bps	256000 bps	460800 bps	921600 bps
HI91	76	10 Hz	100 Hz	250 Hz	250 Hz	500 Hz	1000 Hz

5. RS-485 Output Protocol (Modbus)

Modbus is the most widely used industrial protocol. All Hipnuc products with an RS-485 or Ethernet interface support Modbus RTU.

5.1 Modbus Command Overview

The RS-485 interface follows Modbus RTU. Data is exchanged per register (2 bytes, big-endian). CRC uses the standard [Modbus polynomial](#).

Supported function codes

- **0x06** - Write Single Register (2 bytes)
- **0x03** - Read Holding Registers (one or more)
- **0x50** - Vendor specific (auto ID assignment, mass production tools, firmware upgrade)

Default node ID: 0x50 (decimal 80)

5.2 Frame Structure

5.2.1 Read Registers (0x03)

Host request

Field	Value	Description
ID	1-0xFF	Modbus node ID
FUN_CODE	0x03	Function code
ADDR_H	-	Register address high byte
ADDR_L	-	Register address low byte
LEN_H	-	Number of registers (high byte)
LEN_L	-	Number of registers (low byte)
CRC_L	-	CRC low byte
CRC_H	-	CRC high byte

Module response

Field	Value	Description
ID	1-0xFF	Modbus node ID
FUN_CODE	0x03	Function code
LEN	-	Byte length of data (excluding header and CRC)
DATAH	-	Data high byte
DATAL	-	Data low byte
...	-	Additional data
CRC_L	-	CRC low byte
CRC_H	-	CRC high byte

5.2.2 Write Register (0x06)

Host request

Field	Value	Description
ID	1-0xFF	Modbus node ID
FUN_CODE	0x06	Function code
ADDR_H	-	Register address high byte
ADDR_L	-	Register address low byte
DATA_H	-	Data high byte
DATA_L	-	Data low byte
CRC_L	-	CRC low byte
CRC_H	-	CRC high byte

Module response - echoes the request.

5.3 Register List

Address (Hex)	Address (Dec)	Name	Type	Function	R/W	Description
0x00	0	CTRL	u16	Control	W	See control register description
0x04	4	UART1_BAUD	u16	Baud rate	R/W	UART baud rate
0x05	5	MD_ID	u16	Modbus ID	R/W	Valid range 1-128
0x06	6	HEADING_MODE	u16	Heading mode	R/W	0: 6-axis (relative heading), 1: 9-axis (absolute heading)
0x34	52	ACCX	i16	Acc X	R	Unit: G, scale 0.00048828
0x35	53	ACCY	i16	Acc Y	R	Unit: G, scale 0.00048828
0x36	54	ACCZ	i16	Acc Z	R	Unit: G, scale 0.00048828
0x37	55	GYRX	i16	Gyro X	R	Unit: deg/s, scale 0.061035
0x38	56	GYRY	i16	Gyro Y	R	Unit: deg/s, scale 0.061035
0x39	57	GYRZ	i16	Gyro Z	R	Unit: deg/s, scale 0.061035
0x3A	58	MAGX	i16	Mag X	R	Unit: uT, scale 0.030517
0x3B	59	MAGY	i16	Mag Y	R	Unit: uT, scale 0.030517
0x3C	60	MAGZ	i16	Mag Z	R	Unit: uT, scale 0.030517
0x3D	61	R_H	i32	Roll high	R	Unit: deg, scale 0.001
0x3E	62	R_L	-	Roll low	R	Unit: deg, scale 0.001
0x3F	63	P_H	i32	Pitch high	R	Unit: deg, scale 0.001
0x40	64	P_L	-	Pitch low	R	Unit: deg, scale 0.001
0x41	65	Y_H	i32	Yaw high	R	Unit: deg, scale 0.001
0x42	66	Y_L	-	Yaw low	R	Unit: deg, scale 0.001
0x43	67	TEMP	i16	Temperature	R	Unit: degC, scale 0.01
0x44	68	PRS_H	i32	Pressure high	R	Unit: Pa, scale 0.01
0x45	69	PRS_L	-	Pressure low	R	Unit: Pa, scale 0.01
0x46	70	Q0	u16	Quaternion W	R	Scale 0.0001

Address (Hex)	Address (Dec)	Name	Type	Function	R/W	Description
0x47	71	Q1	u16	Quaternion X	R	Scale 0.0001
0x48	72	Q2	u16	Quaternion Y	R	Scale 0.0001
0x49	73	Q3	u16	Quaternion Z	R	Scale 0.0001
0x4A	74	INCLL_X	i16	Inclinometer X	R	Dual-axis: +/-180deg, 0.011deg/LSB; single-axis: 0-360deg
0x4B	75	INCLL_Y	i16	Inclinometer Y	R	Dual-axis: +/-90deg, 0.011deg/LSB; single-axis products reserve this register
0x4E	78	HEAVE	i16	Vessel heave	R	Displacement, unit m, scale 0.01
0x51	81	HEAVE_PERIOD	i16	Heave period	R	Unit s, scale 0.001
0x70-0x77	112-119	PNAME	u16	Product name	R	ASCII string, 8 registers
0x78	120	SW_VERSION	u16	Firmware	R	Firmware version
0x79	121	BL_VERSION	u16	Bootloader	R	Bootloader version
0x7F-0x82	127-130	SN	u16	Serial number	R	Unique serial number (4 registers)
0xA5	165	ATT_RST	u16	Auto leveling	W	3: auto-level (Pitch/Roll near 0deg->0deg, inverted->0deg/180deg); 5: cancel leveling; others: reserved
0xA6	166	URFR	u16	Mounting mode	W	0: horizontal; 1: vertical Y+ down; 2: Y+ up; 3: X+ up; 4: X+ down

CTRL register (0x00)

Action	Value
Save parameters to flash	0x0000
Restore factory defaults	0x0001
Reset	0x00FF

5.4 Common Configurations

Note Note

Examples assume the Modbus node ID is 0x50 (factory default). If you changed the node ID, update both the ID and CRC fields accordingly.

5.4.1 Save All Parameters to Flash

```
50 06 00 00 00 00 84 4B
```

5.4.2 Restore Factory Defaults

```
50 06 00 00 00 01 45 8B
```

5.4.3 Reset

50 06 00 00 00 FF C4 0B

5.4.4 Configure Baud Rate (Register 0x04)

Baud rate	Command (Hex, ID=0x50)
4800	50 06 00 04 00 00 C5 8A
9600	50 06 00 04 00 01 04 4A
19200	50 06 00 04 00 02 44 4B
38400	50 06 00 04 00 03 85 8B
57600	50 06 00 04 00 04 C4 49
115200	50 06 00 04 00 05 05 89
230400	50 06 00 04 00 06 45 88
460800	50 06 00 04 00 07 84 48
921600	50 06 00 04 00 08 C4 4C

5.4.5 Configure Node ID (Register 0x05)

Format: [CURRENT_ID] 06 00 05 00 [NEW_ID] CRC

- **CURRENT_ID** - current Modbus node ID
- **NEW_ID** - desired ID

Examples (current ID = 0x50):

- New ID = 0x50: 50 06 00 05 00 50 94 76
- New ID = 0x51: 50 06 00 05 00 51 55 B6
- New ID = 0x52: 50 06 00 05 00 52 15 B7
- New ID = 0x53: 50 06 00 05 00 53 D4 77

! Important

The new ID becomes active immediately. Update the ID field in subsequent Modbus frames. If you are unfamiliar with Modbus, use the PC tool instead.

5.4.6 Installation Mode (Register 0xA6)

Target pose	Command (Hex, ID=0x50)
0: Horizontal (default)	50 06 00 A6 00 00 64 68
1: Vertical, Y+ down	50 06 00 A6 00 01 A5 A8
2: Vertical, Y+ up	50 06 00 A6 00 02 E5 A9
3: Vertical, X+ up	50 06 00 A6 00 03 24 69
4: Vertical, X+ down	50 06 00 A6 00 04 65 AB

5.4.7 Horizontal Calibration (Register 0xA5)

- Start auto leveling: 50 06 00 A5 00 02 15 A9
- Cancel auto leveling: 50 06 00 A5 00 05 54 6B

5.4.8 6-Axis vs. 9-Axis Mode (Register 0x06)

- 6-axis mode: 50 06 00 06 00 00 64 4A
- 9-axis mode: 50 06 00 06 00 01 A5 8A

5.5 Read Module Version (0x70-0x82)

Request: 50 03 00 70 00 14 49 9F

Field	Value	Description
ID	0x50	Node ID
Function	0x03	Read holding registers
Start address	0x0070	Product info base
Length	0x0014	20 registers
CRC	0x9F49	-

Response: 50 03 28 48 49 31 34 52 32 4E 2D 34 38 35 2D 30 30 30 00 00 98 00 6B 00 00 00 00 00 00 00 00 00 00 04 7D 95 5F 8D 2A 17 08 00 00 4D 0C

Field	Data	Description
ID	0x50	Node ID
Function	0x03	Function code
Length	0x28 (40 bytes)	20 registers x 2 bytes
Product	48 49 ... 30 30	"CH10x(M)"
Firmware	0x98	V1.52
Bootloader	0x6B	V1.07
Serial No	04 7D 95 5F 8D 2A 17 08	SN

5.6 Read Sensor Data (0x34-0x4B)

Request: 50 03 00 34 00 18 09 8F

Field	Value	Description
ID	0x50	Node ID
Function	0x03	Read holding registers
Start address	0x0034	Sensor data base
Length	0x0018	24 registers
CRC	0x8F09	-

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 (G, 1G ~ 9.8 m/s²)

Axis	Hex	Dec	Scale	Value
X	FF 01	-255	0.00048828	-0.1245
Y	03 B0	944	0.00048828	0.4609
Z	06 50	1616	0.00048828	0.7891

Gyro (deg/s)

Axis	Hex	Dec	Scale	Value
X	FC C9	-823	0.061035	-50.2318
Y	FF 7C	-132	0.061035	-8.0566
Z	00 91	145	0.061035	8.8501

Magnetic field (uT)

Axis	Hex	Dec	Scale	Value
X	01 D5	469	0.030517	14.3125
Y	FD DB	-549	0.030517	-16.7538
Z	FD 27	-729	0.030517	-22.2469

Euler angles (deg)

Axis	Hex	Dec	Scale	Value
Roll	00 00 21 FF	8,703	0.001	8.703
Pitch	00 00 7F F6	32,758	0.001	32.758
Yaw	FF FD 73 E7	-166,937	0.001	-166.937

6. CAN Data Protocol (J1939)

The default CAN output uses SAE J1939, fully compliant with the [SAE J1939 standard](#). J1939 builds on CAN 2.0B extended frames and is widely adopted in commercial vehicles and industrial equipment. For a short primer, see [this tutorial](#).

6.0.1 CAN Extended Frame Format

J1939 uses 29-bit identifiers with the following layout:

```
Bits (MSB -> LSB):
[28:26] Priority (P)
[25]   Reserved (R)
[24]   Data Page (DP)
[23:16] PDU Format (PF)
[15:8] PDU Specific (PS)
[7:0]  Source Address (SA)
```

Identifier calculation:

```
CAN_ID = (Priority << 26) | (Reserved << 25) | (DataPage << 24) |
         (PF << 16) | (PS << 8) | SourceAddress
```

6.0.2 Endianness

J1939 payloads use **little-endian** ordering:

- Multi-byte values store the least significant byte first.
- Example: 0x12345678 is transmitted as 78 56 34 12.

6.1 Protocol Parameters

Parameter	Value	Description
CAN baud rate	500 kbit/s (default)	Supports 125k/250k/500k/800k/1M
Frame format	CAN 2.0B extended (29-bit)	J1939 compliant
Data length	8 bytes (fixed)	Payload per PGN
Frame priority	3 (default)	0 = highest priority
Default node address	8	Range 1-127, broadcast 255
PF	0xFF	Proprietary PGNs
PS	Type selector	Distinguishes sensor data
Data format	Little-endian	LSB first
Data type	Signed/unsigned integer	Depends on PGN

6.2 PGN List

All PGNs use proprietary format (PF=0xFF) with 8-byte payloads.

6.2.1 PGN 65327 (0xFF2F) - Time Information

Provides UTC time for synchronization.

CAN ID: x0CFF2F[SA] (priority 3, PF=0xFF, PS=0x2F, SA default 0x08)

Byte	Field	Type	Range	Unit	Scale	Description
0	UTC Year	uint8	0-99	year	1	20 -> 2020
1	UTC Month	uint8	1-12	month	1	-
2	UTC Day	uint8	1-31	day	1	-
3	UTC Hour	uint8	0-23	hour	1	24-hour format
4	UTC Min	uint8	0-59	min	1	-
5	UTC Sec	uint8	0-59	s	1	-
6-7	UTC ms	uint16	0-999	ms	1	Little-endian

Example:

CAN ID: 0x0CFF2F08

Data: 18 06 12 0E 1E 2D 58 02

-> 2024-06-18 14:30:45.600 UTC

6.2.2 PGN 65332 (0xFF34) - Acceleration

CAN ID: x0CFF34[SA]

Bytes	Field	Type	Range	Unit	Scale	Description
0-1	Acc X	int16	+/-32767	G	0.00048828	Little-endian
2-3	Acc Y	int16	+/-32767	G	0.00048828	Little-endian
4-5	Acc Z	int16	+/-32767	G	0.00048828	Little-endian
6-7	Reserved	uint16	0	-	-	Fixed 0

6.2.3 PGN 65335 (0xFF37) - Angular Velocity

CAN ID: x0CFF37[SA]

Bytes	Field	Type	Range	Unit	Scale	Description
0-1	Gyro X	int16	+/-32767	deg/s	0.061035	Little-endian
2-3	Gyro Y	int16	+/-32767	deg/s	0.061035	Little-endian
4-5	Gyro Z	int16	+/-32767	deg/s	0.061035	Little-endian
6-7	Reserved	uint16	0	-	-	Fixed 0

6.2.4 PGN 65341 (0xFF3D) - Roll & Pitch

CAN ID: x0CFF3D[SA]

Bytes	Field	Type	Range	Unit	Scale	Description
0-3	Roll	int32	+/-180,000,000	deg	0.001	Little-endian
4-7	Pitch	int32	+/-90,000,000	deg	0.001	Little-endian

6.2.5 PGN 65345 (0xFF41) - Heading

CAN ID: x0CFF41[SA]

Bytes	Field	Type	Range	Unit	Scale	Description
0-3	Heading (0-360deg)	uint32	0-360,000,000	deg	0.001	Clockwise positive
4-7	Heading (+/-180deg)	int32	+/-180,000,000	deg	0.001	Counter-clockwise positive

6.2.6 PGN 65338 (0xFF3A) - Magnetic Field

CAN ID: x0CFF3A[SA]

Bytes	Field	Type	Range	Unit	Scale	Description
0-1	Mag X	int16	+/-32767	uT	0.030517	Little-endian
2-3	Mag Y	int16	+/-32767	uT	0.030517	Little-endian
4-5	Mag Z	int16	+/-32767	uT	0.030517	Little-endian
6-7	Reserved	uint16	0	-	-	Fixed 0

6.2.7 PGN 65350 (0xFF46) - Quaternion

CAN ID: x0CFF46[SA]

Bytes	Field	Type	Range	Unit	Scale	Description
0-1	qw	int16	+/-32767	-	0.0001	Real part
2-3	qx	int16	+/-32767	-	0.0001	i component
4-5	qy	int16	+/-32767	-	0.0001	j component
6-7	qz	int16	+/-32767	-	0.0001	k component

6.2.8 PGN 65354 (0xFF4A) - Inclinator Output

Applies to inclinometer products only.

CAN ID: x0CFF4A[SA]

Bytes	Field	Type	Range	Unit	Scale	Description
0-3	X angle	int32	0-360,000,000	deg	0.001	Configurable 0-360deg or +/-180deg
4-7	Y angle	int32	0-360,000,000	deg	0.001	Configurable 0-360deg or +/-90deg

6.3 Configuration Protocol

Configuration messages use proprietary PGNs that mirror the Modbus register map.

6.3.1 Frame Format

- CAN ID: x0CEF[DA][SA]
 - **SA**: source address (host controller)
 - **DA**: destination address (device node ID or broadcast 255)

Payload layout:

Byte(s)	Field	Type	Description
0-1	ADDR	uint16	Register address (little-endian)
2	CMD	uint8	Command type (0x06 write, 0x03 read)
3	STATUS	uint8	Status (0 when writing)
4-7	VAL	uint32	Data (little-endian)

6.3.2 Register Mapping

Identical to the Modbus register list.

6.4 Configuration Examples (default node ID = 8)

CAN ID	Data	Description
0x0CEF08xx	34 01 06 00 [VAL]	Set PGN FF34 (acceleration) period in ms (5-1000)
0x0CEF08xx	37 01 06 00 [VAL]	Set PGN FF37 (gyro) period
0x0CEF08xx	3D 01 06 00 [VAL]	Set PGN FF3D (roll/pitch) period
0x0CEF08xx	41 01 06 00 [VAL]	Set PGN FF41 (heading) period
0x0CEF08xx	3A 01 06 00 [VAL]	Set PGN FF3A (magnetic field) period
0x0CEF08xx	46 01 06 00 [VAL]	Set PGN FF46 (quaternion) period
0x0CEF08xx	4A 01 06 00 [VAL]	Set PGN FF4A (inclinometer) period
0x0CEF08xx	96 00 06 00 [PGN]	Trigger one-time output for the specified PGN
0x0CEF08xx	A5 00 06 00 [VAL]	Attitude reset (1: heading reset, 2: zero pitch/roll, 3: auto-level, 5: cancel)
0x0CEF08xx	9D 00 06 00 01 00 00 00	Enable all outputs
0x0CEF08xx	9D 00 06 00 00 00 00 00	Disable all outputs
0x0CEF08xx	00 00 06 00 00 00 00 00	Save configuration
0x0CEF08xx	00 00 06 00 01 00 00 00	Restore defaults
0x0CEF08xx	00 00 06 00 FF 00 00 00	Reset
0x0CEF08xx	9A 00 06 00 [VAL]	Set CAN baud (0:1000k, 1:800k, 2:500k, 3:250k, 4:125k)
0x0CEF08xx	9C 00 06 00 [VAL]	Set J1939 node ID (1-128)
0x0CEF08xx	9E 00 06 00 [VAL]	Inclinometer X polarity (0: default, 1: invert)
0x0CEF08xx	9F 00 06 00 [VAL]	Inclinometer Y polarity (0: default, 1: invert)

Note **Address notes**

- xx in the CAN ID is the source address of the host.

- [VAL] indicates any 4-byte little-endian value.

Example: ID x0CEF0855, data 37 01 06 00 64 00 00 00 -> set PGN FF37 (gyro) period to 100 ms (10 Hz).

6.5 Time Synchronization

J1939 synchronization combines the hardware PPS input with PGN 65327.

6.5.1 Principle

1. **Hardware:** connect the external PPS (1 Hz, TTL, rising-edge) signal to SYNC_IN/PPS.
2. **Software:** send PGN 65327 frames containing UTC time.

6.5.2 Procedure

1. **Hardware connection** - wire PPS to SYNC_IN/PPS, ensure TTL level and stable pulses.
2. **PPS validation** - module checks at least three consecutive pulses with ± 10 ms spacing error.
3. **Send time info** - host transmits PGN 65327 (see definition above).
4. **Apply sync** - upon receiving valid time data, the module waits for the next PPS rising edge and sets the internal clock to the specified UTC value.

6.5.3 Example

Goal: sync to 2024-06-18 14:30:45.600 UTC.

1. Confirm PPS is present and stable.
2. Host sends PGN 65327:

CAN ID: 0x0CFF2F55

Data: 18 06 12 0E 1E 2D 58 02

-> Year=24, Month=6, Day=18, Hour=14, Minute=30, Second=45, Millisecond=600

3. Device waits for the next PPS edge (within 1 s), then sets its clock to 14:30:46.000 UTC. Subsequent PGN 65327 outputs reflect the synchronized UTC timestamp.

7. CAN Data Protocol (CANopen)

Modules ship with J1939 enabled. Contact support if you need CANopen firmware. CANopen output uses standard data frames and TPDO1-TPDO7. Remote frames and extended identifiers are not used. All TPDOs default to asynchronous periodic mode.

7.1 Default Settings

Item	Value
CAN baud	500 kbit/s
Node ID	8
NMT state	Operational
TPDO rate	1-200 Hz per TPDO

7.2 CANopen TPDO Mapping

TPDO	COB-ID	DLC	Mode (Transmission Type)	Rate (Hz)	Data	Description
TPDO1	0x180+ID	6	Asynchronous (0xFE)	100	Acceleration	int16 XYZ, LSB first, unit mG
TPDO2	0x280+ID	6	Asynchronous (0xFE)	100	Angular rate	int16 XYZ, unit 0.1 deg/s
TPDO3	0x380+ID	6	Asynchronous (0xFE)	100	Euler angles	int16 roll/pitch/yaw, unit 0.01deg
TPDO4	0x480+ID	8	Asynchronous (0xFE)	100	Quaternion	int16 w,x,y,z scaled x10000
TPDO6	0x680+ID	4	Asynchronous (0xFE)	20	Pressure	int32 Pa
TPDO7	0x780+ID	8	Asynchronous (0xFE)	100	Inclinometer	int32 X/Y, unit 0.01deg

7.2.1 Data Example

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

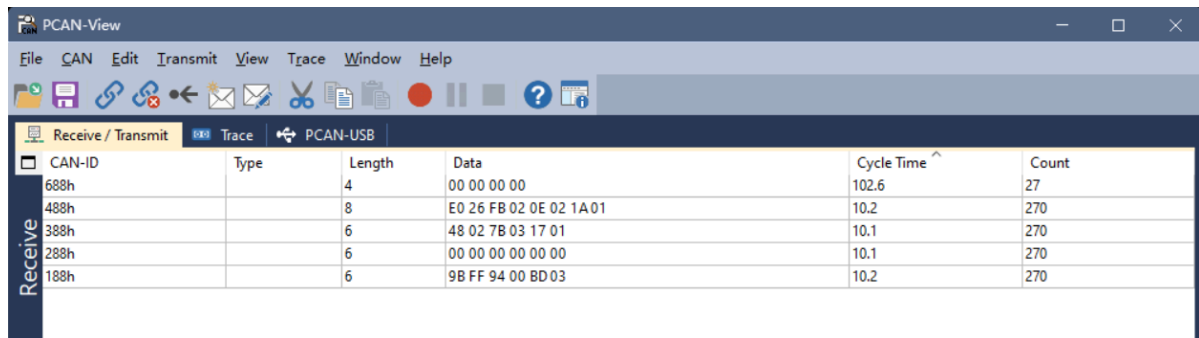
- Device ID 8 sends acceleration
- X = 0x004A = 74 mG
- Y = 0x001F = 31 mG
- Z = 0x03C8 = 968 mG

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

- X = 0x0015 = 21 -> 2.1 deg/s
- Y = 0x0114 = 276 -> 27.6 deg/s
- Z = 0x0034 = 52 -> 5.2 deg/s

7.3 Connecting via PC Software

Use PCAN-View (with a PCAN adapter) to monitor frames and verify rates.



The screenshot shows the PCAN-View software interface. The main window displays a table of received CAN frames. The table has columns for CAN-ID, Type, Length, Data, Cycle Time, and Count. The data is as follows:

CAN-ID	Type	Length	Data	Cycle Time ^	Count
688h		4	00 00 00 00	102.6	27
488h		8	E0 26 FB 02 0E 02 1A 01	10.2	270
388h		6	48 02 7B 03 17 01	10.1	270
288h		6	00 00 00 00 00 00	10.1	270
188h		6	9B FF 94 00 BD 03	10.2	270

7.4 Configuration Commands (SDO)

All configuration uses expedited SDO writes. Save settings to flash before rebooting if you need persistence.

7.4.1 Expedited SDO Format

Host -> Device

COB-ID	Byte0 (CS)	Index (2B, LSB first)	Sub-index	Data (4B, LSB first)
0x600+ID	0x23 (write 4 bytes)	index	sub	data

Device -> Host

COB-ID	Byte0	Index	Sub-index	Data
0x580+ID	0x60 (write ack)	index	sub	reserved

7.4.2 Common Commands

7.4.2.1 Change Node ID (0x20A0)

ID=0x600+ID, DATA=23 A0 20 00 [ID] 00 00 00

- Range: 1-127
- Save to flash and reset to apply.

7.4.2.2 Save to Flash (0x2000)

ID=0x600+ID, DATA=23 00 20 00 00 00 00 00

7.4.2.3 Reset (0x2000)

ID=0x600+ID, DATA=23 00 20 00 FF 00 00 00

7.4.2.4 Restore Factory Defaults (0x2000)

ID=0x600+ID, DATA=23 00 20 00 01 00 00 00

! Restoring defaults resets baud rate, node ID, and every parameter. Takes effect after power cycle.

7.4.2.5 Change CAN Baud (0x209A)

ID=0x600+ID, DATA=23 9A 20 00 [BAUD_CODE]

- 0: 1000 kbit/s
- 2: 500 kbit/s
- 3: 250 kbit/s
- 4: 125 kbit/s

Save and reset to apply.

7.4.3 TPDO Configuration

TPDO communication parameters map to indexes 0x1800-0x1805:

TPDO	COB-ID	Index
1	0x180+ID	0x1800
2	0x280+ID	0x1801
3	0x380+ID	0x1802
4	0x480+ID	0x1803
6	0x680+ID	0x1804
7	0x780+ID	0x1805

7.4.3.1 Modify / Disable / Enable Output Rate (sub-index 5)

ID=0x600+ID, DATA=2B, [index LSB], [index MSB], 05, [period LSB], [period MSB], 00, 00

- Period is in milliseconds. 0 disables the TPDO.
- Example for TPDO1 (index 0x1800):
 - ... 00 18 05 00 00 00 00 - disable acceleration
 - ... 00 18 05 05 00 00 00 - 5 ms (200 Hz)
 - ... 00 18 05 0A 00 00 00 - 10 ms (100 Hz)
 - ... 00 18 05 14 00 00 00 - 20 ms (50 Hz)
 - ... 00 18 05 32 00 00 00 - 50 ms (20 Hz)
 - ... 00 18 05 64 00 00 00 - 100 ms (10 Hz)

Repeat with index 0x1801 for TPDO2 (gyro), 0x1802 for TPDO3 (Euler), etc.

Note Example: to set TPDO1 (acceleration) to 100 Hz:

ID=0x600+ID, DATA=2B 00 18 05 0A 00 00 00 (0x0A = 10 ms)

7.4.3.2 Inclinometer Polarity (0x209E, 0x209F)

- 23 9E 20 00 00 00 00 00 - X axis default
- 23 9E 20 00 01 00 00 00 - X axis inverted
- 23 9F 20 00 00 00 00 00 - Y axis default
- 23 9F 20 00 01 00 00 00 - Y axis inverted

7.4.3.3 Inclinometer Zero (0x20A5)

- 23 A5 20 00 02 00 00 00 - set current pose as zero (X=0, Y=0)
- 23 A5 20 00 05 00 00 00 - clear zero offset

7.4.4 Sync Mode

Set sub-index 2 of the TPDO communication parameter (0x180x.2) to 0x01 for synchronous mode, or 0xFF for asynchronous (default).

Example (TPDO1):

- 2F 00 18 02 01 00 00 00 - synchronous mode
- 2F 00 18 02 FF 00 00 00 - asynchronous mode

Send SYNC frames (ID=0x80, no data) to trigger synchronous TPDOs.

7.4.5 Heartbeat

Heartbeat interval is set via 0x1017.0 (uint16, ms). 0 disables heartbeats.

Example: 2B 17 10 00 64 00 00 00 -> 100 ms heartbeat period.

8. Magnetic Calibration

8.1 When to Calibrate

The 9-axis (magnetometer-aided) mode requires:

1. At least one magnetic calibration before first use.
2. A low-distortion environment (outdoors and away from steel/electrical equipment is recommended).
3. A rigid connection between the module and its carrier (PCB, housing, robot, etc.) during calibration and operation.

8.2 Calibration Steps

1. **Switch to 9-axis mode** - see "Work Mode Configuration".
2. **Check the environment**
 - Common interference sources: steel furniture, monitors, motors, transformers, chargers, reinforced concrete.
 - Preferred locations:
 - Best: open outdoor area ≥ 5 m away from buildings/vehicles.
 - Acceptable: indoor area ≥ 30 cm from ferromagnetic objects.

! Important

If the module is already mounted on a PCB, metal housing, robot, or any magnetic structure, you **must** calibrate the entire assembly as one rigid body. Calibrating the bare module first and then installing it will not work.

3. **Run calibration**

- Send CONFIG MCAL START (firmware $\geq 1.7.0$).
- Rotate the module slowly within a small spatial area. Keep the mounting base fixed relative to the module.
- Rotate at least 360deg per axis (2-3 turns recommended) at 20-100deg/s (5-20 s per turn). Avoid pauses.
- Typical duration: 30-60 s.

4. **Check status** - send LOG MCAL STAT.

STAT=1

PROGRESS=8

OK

STAT	Meaning	Recommendation
0	Idle	Ready to start
1	Calibrating	Keep rotating

STAT	Meaning	Recommendation
2	Verifying	Keep still
3	Completed	Calibration succeeded
4	Failed	See troubleshooting

PROGRESS ranges 0-100. Calibration succeeds when STAT=3 and PROGRESS=100.

5. **Validate** - place the module flat and rotate 360deg. Heading should sweep smoothly 0-360deg with $\leq \pm 5$ deg error. Large jumps (> 10 deg) indicate a failed calibration.

8.3 Troubleshooting

1. **Check magnetic field strength** - measure $|B| = \sqrt{B_x^2 + B_y^2 + B_z^2}$.

Magnitude	Assessment	Action
20-60 uT	Normal	Proceed
<20 uT or >60 uT	Abnormal	Move away from interference sources

2. **Verify rotation quality** - ensure ≥ 360 deg per axis, 20-100deg/s, no stops.
3. **Check spatial variation** - if field changes > 10 uT when facing different directions in the same location, the area has spatial distortion; relocate or use 6-axis mode.

8.4 FAQ

Cause	Symptom	Fix
Strong environmental distortion	STAT=4 or heading jumps	Calibrate outdoors
Improper rotation	PROGRESS stagnates	Rotate each axis sufficiently
Translational motion	Calibration completes but heading still off	Keep the module at one fixed spot while rotating
Module removed/reinstalled	Heading drifts after reassembly	Re-calibrate after any mechanical change

! Warning 1: Indoor limitations - Indoor magnetic distortion cannot be removed; heading accuracy in 9-axis mode depends heavily on the environment.

! Warning 2: Indoor usage - Labs, factories, garages, etc. often yield poor results even after calibration. Prefer 6-axis mode indoors.

! Warning 3: Fixed relative position - If interference sources move relative to the module, calibration is invalid. Calibrate the complete assembly and keep it rigid.

! Warning 4: Motor vibration - Motors introduce vibration and varying magnetic fields. Calibrate with motors off, or calibrate while motors run if they are part of the fixed platform.

8.5 Calibration Frequency & Mode Selection

- Fixed installations: calibrate once.
- Mobile platforms: recalibrate whenever the environment changes.
- After any mechanical modification: recalibrate.
- Long-term use (>1 year): recalibrate annually.
- If heading deviates unexpectedly: recalibrate immediately.

Scenario	Recommended Mode	Reason
Outdoor open area (UAV, etc.)	9-axis	Stable magnetic field -> absolute heading
Indoor robots / AGVs	6-axis	Indoor distortion causes magnetometer errors
Near motors / EMI sources	6-axis	Disturbances cannot be calibrated out
Need absolute heading	9-axis	Requires stable magnetic environment
Only relative heading needed	6-axis	Heading zeroes at power-up

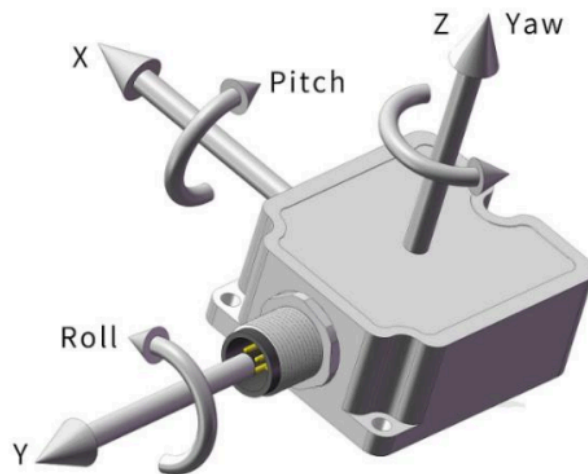
9. Appendix 1 Quaternion / Euler / Rotation Matrix

9.1 Quaternion to Rotation Matrix

Given $(Q_{b2n}) = [q_0, q_1, q_2, q_3]^T$, the direction cosine matrix is

$$C_{b2n} = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1 q_2 - q_0 q_3) & 2(q_1 q_3 + q_0 q_2) \\ 2(q_1 q_2 + q_0 q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2 q_3 - q_0 q_1) \\ 2(q_1 q_3 - q_0 q_2) & 2(q_2 q_3 + q_0 q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}$$

9.2 Quaternion <-> Euler (ENU, 312 sequence: Z -> X -> Y)



Let $(Q_{b2n}) = [q_0, q_1, q_2, q_3]^T$ with scalar part (q_0) .

- pitch (θ): rotation about X, range $([-\pi/2, \pi/2])$
- roll (ϕ): rotation about Y, range $([-\pi, \pi])$
- yaw (ψ): rotation about Z, range $([-\pi, \pi])$

Quaternion -> Euler:

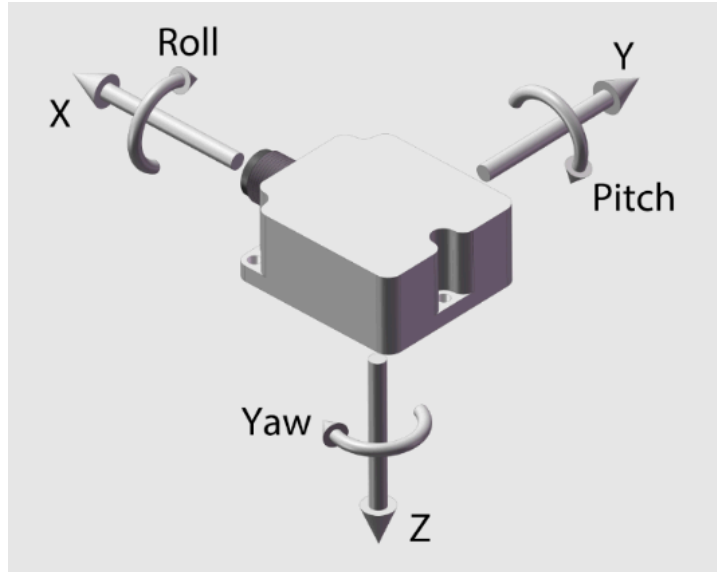
$$\begin{aligned} \text{pitch} &= \arcsin(2(q_0 q_1 + q_2 q_3)) \\ \text{roll} &= -\arctan2(2(q_1 q_3 - q_0 q_2), q_0^2 - q_1^2 - q_2^2 + q_3^2) \\ \text{yaw} &= -\arctan2(2(q_1 q_2 - q_0 q_3), q_0^2 - q_1^2 + q_2^2 - q_3^2) \end{aligned}$$

Euler -> Quaternion:

$$\begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} \end{bmatrix}$$

$$\begin{bmatrix} \cos\frac{\theta}{2}\cos\frac{\phi}{2}\cos\frac{\psi}{2} - \sin\frac{\theta}{2}\sin\frac{\phi}{2}\sin\frac{\psi}{2} \\ \cos\frac{\phi}{2}\cos\frac{\psi}{2}\sin\frac{\theta}{2} - \cos\frac{\theta}{2}\sin\frac{\phi}{2}\sin\frac{\psi}{2} \\ \cos\frac{\theta}{2}\cos\frac{\psi}{2}\sin\frac{\phi}{2} + \cos\frac{\phi}{2}\sin\frac{\theta}{2}\sin\frac{\psi}{2} \\ \cos\frac{\theta}{2}\cos\frac{\phi}{2}\sin\frac{\psi}{2} + \sin\frac{\theta}{2}\sin\frac{\phi}{2}\cos\frac{\psi}{2} \end{bmatrix}$$

9.3 Quaternion <-> Euler (NED, 321 sequence: Z -> Y -> X)



- roll: rotation about X, range $([-\pi, \pi])$
- pitch: rotation about Y, range $([-\pi/2, \pi/2])$
- yaw: rotation about Z, range $([-\pi, \pi])$

Quaternion -> Euler:

$$\begin{aligned} \text{roll} &= \arctan2(2(q_0 q_1 + q_2 q_3), 1 - 2(q_1^2 + q_2^2)) \\ \text{pitch} &= \arcsin(2(q_0 q_2 - q_1 q_3)) \\ \text{yaw} &= \arctan2(2(q_0 q_3 + q_1 q_2), 1 - 2(q_2^2 + q_3^2)) \end{aligned}$$

Euler -> Quaternion:

$$\begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} \cos\frac{\phi}{2}\cos\frac{\theta}{2}\cos\frac{\psi}{2} + \sin\frac{\phi}{2}\sin\frac{\theta}{2}\sin\frac{\psi}{2} \\ \sin\frac{\phi}{2}\cos\frac{\theta}{2}\cos\frac{\psi}{2} - \cos\frac{\phi}{2}\sin\frac{\theta}{2}\sin\frac{\psi}{2} \\ \cos\frac{\phi}{2}\sin\frac{\theta}{2}\cos\frac{\psi}{2} + \sin\frac{\phi}{2}\cos\frac{\theta}{2}\sin\frac{\psi}{2} \\ \sin\frac{\phi}{2}\sin\frac{\theta}{2}\cos\frac{\psi}{2} - \cos\frac{\phi}{2}\sin\frac{\theta}{2}\sin\frac{\psi}{2} \end{bmatrix}$$

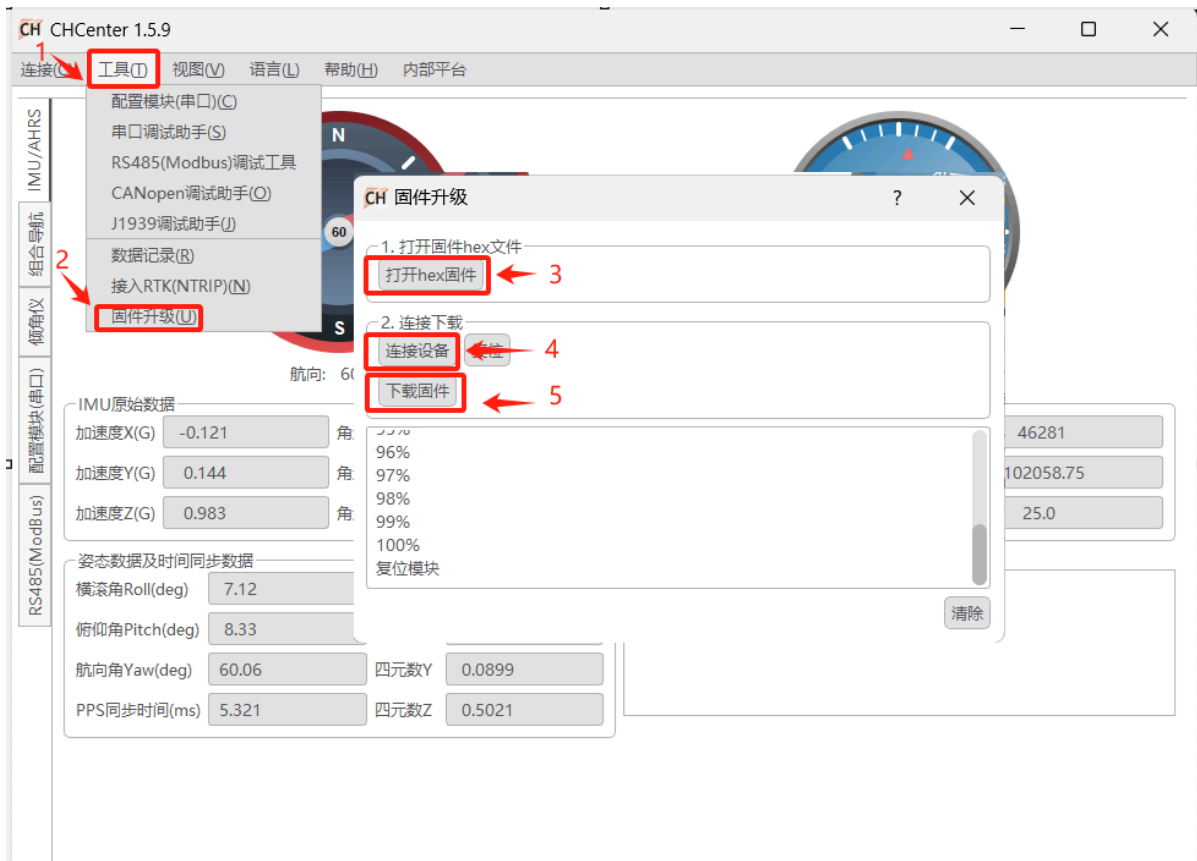
```

\sin\frac{\phi}{2}\cos\frac{\theta}{2}\cos\frac{\psi}{2} - \cos\frac{\phi}{2}\sin\frac{\theta}{2}
{2}\sin\frac{\psi}{2} \
\cos\frac{\phi}{2}\sin\frac{\theta}{2}\cos\frac{\psi}{2} + \sin\frac{\phi}{2}\cos\frac{\theta}{2}
{2}\sin\frac{\psi}{2} \
\cos\frac{\phi}{2}\cos\frac{\theta}{2}\sin\frac{\psi}{2} - \sin\frac{\phi}{2}\sin\frac{\theta}{2}
{2}\cos\frac{\psi}{2}
\end{bmatrix}

```

10. Appendix 2 Firmware Upgrade

Use the CHCenter PC tool to upgrade firmware (*.hex). Contact support to obtain the firmware package.



11. Appendix 3 URFR Derivation

How to determine URFR parameters

Example: rotate -90deg about the X-axis (Y+ points downward).

Constraints:

- $(X_U = X_B)$
- $(Y_U = -Z_B)$
- $(Z_U = Y_B)$

Rotation matrix:



1	0	0
0	0	-1
0	1	0

URFR expects the transpose of this matrix:

1	0	0
0	0	1
0	-1	0

12. Appendix 4 Understanding Magnetic Interference

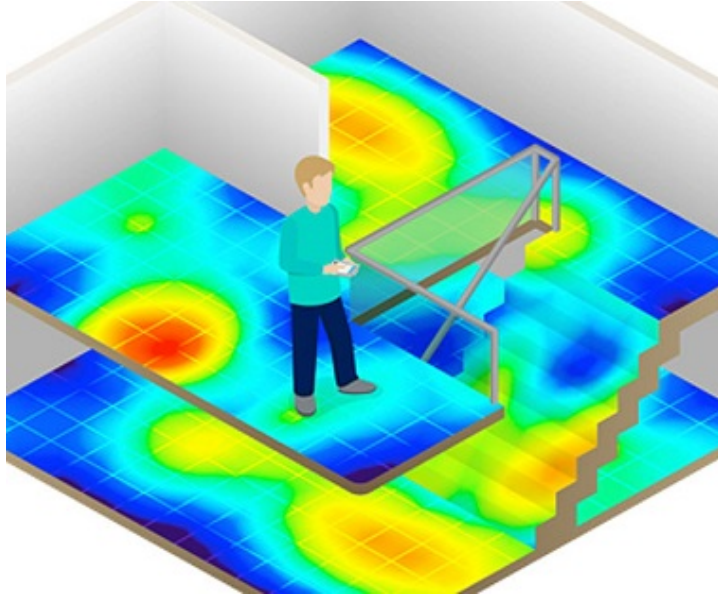
Magnetic interference can be tied to the sensor body (hard/soft iron) or exist in free space.

Distortions that move with the sensor	Distortions that do not move with the sensor
	
<ul style="list-style-type: none"> • Calibration errors • Hard iron effects • Soft iron effects • Etc. 	<ul style="list-style-type: none"> • Spatial distortions • Temporal distortions • Etc.

Type	Sensor-frame interference (hard/soft iron)	Spatial interference
Characteristics	Moves with the sensor	Fixed in space
Typical sources	PCB, metal housing, UAV frame	Furniture, appliances, rebar
Calibration	Possible via magnetic calibration	Impossible - causes large heading errors
Mitigation	Perform user calibration	Use 6-axis mode or relocate

Spatial interference is the primary reason magnetometer-aided heading is unreliable indoors. Distortion

cannot be calibrated away and severely degrades heading accuracy.



Note **Illustration** - Example of typical indoor field distortion (blue = weak, red = strong).

13. Appendix 5 Technical Support

Latest product news and documentation are available on our website and social media channels.

WeChat:



Telegram:

HS



@HIPNUC