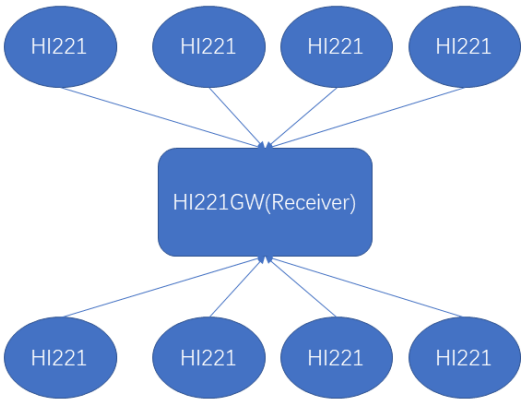
HI221/HI221GW User Guide

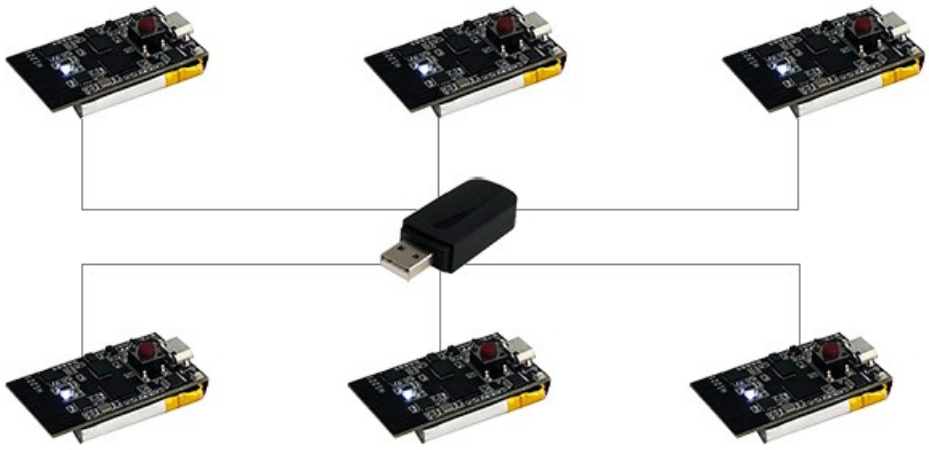
HI221 wireless IMU and receiver system, Rev 0.2

HI221/HI221GW User Guide

Introduction

H221/HI221GW is a miniature wireless inertial measurement unit (IMU) system launched by HIPNUC. This module features low cost, high performance, small size, and low latency. It can output accurate 3D attitude data which is calibrated by factory. The data are processed by our fusion algorithm, including roll angle, pitch angle, and relative heading angle. It can also output raw sensor data.
H221/HI221GW system consists of HI221GW (receiver) and HI221 (attitude module). A HI221GW can connect up to 8 HI221 modules to form a star network structure. Each HI221 can output attitude data real-time, and the output rate can reach 100Hz.





Features

On-board sensors

- Three-axis gyroscope with
    - maximum range: ± 2000 °/s
    - output rate up to 2000Hz
- Three-axis accelerometer with
    - maximum range: ± 8g
    - output rate up to 125Hz
- Triaxial geomagnetic field sensor with
    - maximum range: 800mG
    - internal sampling rate up to 100Hz

Data process

- Accelerometer and gyroscope are calibrated by factory to correct 3-axis non-orthogonal and scale factor error.
- Quaternions and Euler angles are calculated in geographic coordinate system by data fusion algorithm.

Communication interface and power supply of HI221

- Serial port (compatible with TTL, which can be directly connected with 5V or 3.3V serial port device)
- Supply voltage: 3.3 (+/- 100 mV)
- Power consumption at peak：120mA (While using RF and Tx emitting)

Others

- We provide GUI on the PC(Win)side, providing real-time data display, waveform, calibration and data logging functions.
- Configurable parameters.

Hardware Specifications (nodes)

Parameter	Description
Data Interface	UART(TTL 1.8V - 3.3V) or 2.4RF Radio
Supply Voltage	3.3V (± 100mV)
Power Consumption	396mW @3.3V
Temperature Tolerance	-20℃ - 85 ℃

Parameter	Description
Maximum Linear Accelerations	0 - 115 m/s^2
Size	20 x 38 x 8.5mm (W x L x H)
On-board Sensors	3-axis gyroscope, 3-axis accelerometer and 3-axis magnetometer

Hardware Installation

Due to the sensor manufacturing process, the performance of the X/Y and Z axes is slightly different. It is recommended that :

- Make the module's Z axis parallel the direction of gravity in your installation. In other words, install the module horizontally.
- Keep the module at least 10cm away from magnetic components such as iron housings and low-power motors.

Hardware Performance

Output accuracy of attitude

Attitude	Type	Maximum
Roll Angle \ Pitch Angle - Error at static situation	0.2°	0.4°
Roll Angle \ Pitch Angle - Error at dynamic situation	0.5°	2.0°
Heading Angle	-	-

Gyroscope

Parameter	Value
Measuring Range	±2000°/s
Non-linearity	±0.1% (Has best performance at 25°)

Parameter	Value
Noise density	$0.08°/s/\sqrt{Hz}$
Sampling Rate	2000Hz

Accelerometer

Parameter	Value
Measuring Range	±8G (1G = 1x Gravitational acceleration)
Non-linearity	±0.5% (Has best performance at 25°)
Maximum zero offset	10mG (Calibrated)
Noise density	$250\ uG\sqrt{Hz}$
Sampling Rate	125Hz

Magnetometer

Parameter	Value
Measuring Range	±8Gauss
Non-linearity	±0.1%
Sampling Rate	100Hz

Data interface specifications (UART)

Parameter	Value
Serial Output Baud Rate	4800/9600/115200/460800 (Optional)
Output Frame Rate	1 - 400Hz

Data interface specifications (2.4G RF)

Parameters	Value

Parameters	Value
In Air Baud Rate	1Mbps
Output Frame Rate	100Hz
Maximum Number of Connected Devices	8

Definition of Reference Frame

This product uses **right-hand** (cartesian) coordinate system. The output of quaternions and Euler angles are the rotation from the sensor coordinate system to the inertial coordinate system (which is also called world coordinate system).

The rotation order of Euler angles is ZYX (Z axis first, then Y axis, and finally X axis), which is specifically defined as follows :

- Rotate around Z axis : Yaw, phi (ψ) . The range is -180° to 180°
- Rotate around Y axis : Pitch, theta (θ) . The range is -90° to 90°
- Rotate around X axis : Roll, psi (ϕ) . The range is -180° to 180°

This product uses (North-West-Up, NWU) coordinate system, which is defined as follows：

- Positive X axis points to north
- Positive Y axis points to west
- Positive Z axis points to the sky

When using the NWU system and the module is simulated as an aircraft, the X axis should be considered as heading direction. When the coordinate system of sensor and world are coincide, the ideal output of the Euler angles should be :

- Pitch = 0 °, Roll = 0 °, Yaw = 0 °

Protocol of Serial Communication

Format of a Packet

For more applications, we provide data analysis functions by C and C# in supporting resources. After the module is powered on, the packets output rate is set by default at 100Hz (factory default output rate). The format of data packet is described as follows :

Field	Syncing frame header	Frame type	Frame length	CRC16	Data in a frame
name	PRE	TYPE	LEN	CRC	REG_ADDR(N) + DATA(N)
size (byte)	0	1	2	2	variable (1-64)
shift (byte)	0	1	2	4	6
value (hex)	0x5A	0xA5	length value	CRC check code	check more details in the next section
type	uint8_t	uint8_t	uint16_t	uint16_t	-

- PRE
  It's fixed at 0x5A.
- TYPE
  It's fixed at 0xA5 representing a data frame.
- LEN
  The length of data field in a data frame. The maximum of a data frame is 256 bytes LSB (low byte first), and the length only includes of the real data, not including `PRE`,`TYPE`,`LEN`,`CRC` numeric field.
- CRC
  16-bit CRC checksum of all the other data and LSB[^LSB] in a frame, except the CRC itself. CRC implementing functions is presented as follows：

```
/*
    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;
}
```

- REG_ADDR and DATA
  A frame of data can be composed of multiple data packets. Each data packet contains two parts: register address (REG_ADDR) and register data (DATA). The register address determines the type and length of the data, and DATA is the content of register data. Supported list of registers in the module is described as follows :

Register address	Bytes in register	Name	Unit
0x90	1	user ID of the module	N/A
0xA0	6	acceleration	0.001G[1]
0xA5	6	linear acceleration	0.001G
0xB0	6	angular velocity	0.1°/s
0xC0	6	strength of magnetic field	0.001Gauss
0xD0	6	Euler angles (as integer)	1°
0xD9	12	Euler angles (as float/double)	1°
0xD1	16	quaternion	N/A
0xF0	4	air pressure	Pa

Register address	Bytes in register	Name	Unit
0x71	128-256 bytes (variable)	Quaternion collection from wireless nodes	N/A
0x72	48-96 bytes (variable)	Euler angles collection from wireless nodes	1°
0x75	48-96 bytes (variable)	acceleration collection from wireless nodes	0.001G [1]
0x78	48-96 bytes (variable)	angular velocity collection of wireless nodes	0.1°/s
0x61	3	extensive identification of the wireless data frame	N/A

- 0x90
  user ID of the module
- 0xA0
  Raw acceleration of the sensor, outputted as int16, and three axes in total. Each axis occupies 2 bytes, so the total of X, Y, Z axes is 6 bytes, and LSB.
- 0xA5
  Linear acceleration value without gravity in geographic coordinate system, outputted as int16. There are 3 axes, X, Y, and Z, each axis occupies 2 bytes, so the total is 6 bytes, and LSB.
- 0xB0
  Angular velocity of the sensor, outputted as int16. There are 3 numbers for 3 axes, X, Y, and Z, and each number occupies 2 bytes, so the total of them is 6 bytes, LSB.
- 0xC0
  The strength of magnetic field measured by the sensor, outputted as int16. There are numbers in 3 axes, X, Y, and Z, and each number occupies 2 bytes, so the total is 6 bytes, LSB.
- 0xD0
  Euler angles of the sensor, outputted as int16. There are 3 numbers, , and the order is Pitch-Roll-Yaw for 3 axes, X, Y, and Z, . Each number occupies 2 bytes, LSB. The values of Roll and Pitch you received need to be divided by 100 , and Yaw needs to be divided by 10 to get the true angles：
    - ex. When you receive Yaw = 100, the heading angle is 10 °.

- 0xD9
  Euler angles of the sensor, outputted as float. There are 3 numbers, Pitch, Roll and Yaw for 3 axes, X, Y, and Z. Each number occupies 4 bytes (float), LSB.
- 0XD1
  Quaternion of the sensor, outputted as float. The data contains four number, which is put in order of W-X-Y-Z. Each of the number occupies 4 bytes (float), so the total size of quaternion is 16 bytes, and LSB.
- 0XF0
  Air pressure. Only works for products with pressure sensor.
- 0x71
  Only support HI221GW(receiver). The collection of quaternions from wireless nodes. A frame consists of a series of quaternions from the nodes, in order of the **user ID** you set. For example, you set nodes ID from 0 to 5, there will be 6 nodes totally. Each node occupies 16 bytes , and consists of a quaternion that is W, X, Y, and Z. Every value is stored in the float type, and each float occupies 4 bytes, and LSB.
- 0x72
  Only support HI221GW(receiver). The collection of Euler angles of wireless nodes. A frame consists of a series of Euler angles from the nodes, in order of the **user ID** you set. For example, you set nodes ID from 0 to 5, there will be 6 nodes totally. Each node occupies 6 bytes, and consists of 3 integers(int16) in order of Pitch-Roll-Yaw, and each integer occupies 2 bytes, and LSB. The values of Roll and Pitch you received need to be divided by 100 , and Yaw needs to be divided by 10 to get the true angles：
    - ex. When you receive Yaw = 100, the heading angle is 10 °.
- 0x75
  Only support HI221GW(receiver). The collection of accelerations from wireless nodes. This section consists of a series of accelerations from the nodes, in order of the **user ID** you set. Each node contains 3 **int16_t**, in order of X, Y, and Z. Note that an **int16_t** occupies 2bytes, and LSB.
- 0x78
  Only support HI221GW(receiver). The collection of angular velocities from wireless nodes . This section consists of a series of angular velocities from the nodes, in order of the **user ID** you set. Each node contains 3 **int16_t**, in order of X, Y, and Z. Note that an **int16_t** occupies 2bytes, and LSB.
- 0x61

Only support HI221GW(receiver). Get extensive identification of the wireless data frame, 3 bytes in total.

Bytes offset in data frame extension identification	Value	Description
0	-	N/A
1	GWID	GWID of a receiver
2	CNT	the count of nodes contained in this frame : 1-16

Factory Default Register

The register data carried in one frame by factory default is defined as follows :

HI226/HI229:

Order	Data packet	Description
1	0x90	user ID of module
2	0xA0	accelerations
3	0xB0	angular velocities
4	0xC0	strength of magnetic field
5	0xD0	Euler angles as integer
6	0xF0	air pressure

HI221GW(wireless receiver of nodes):

Order	in Register	Description
1	0x71	quaternions
2	0x75	angular velocities

Example of Data Structure in a Frame

Let's assume that `A0, B0, D0` are in a frame of some output data . Use the serial assistant to sample a frame of data, and find the following value :

`5A A5 15 00 A9 8B A0 EA FF D0 03 45 FF B0 00 00 00 00 00 00 D0 87 00 6F 27 F5 FF`

where：

`5A A5` is frame header.

`15 00` is the length of data field : (0x00<<8) + 0x15 = 21

`A9 8B` is the checksum of CRC : (0x8B<<8) + 0xA9 = 0x8BA9

- `A0 EA FF D0 03 45 FF` are the accelerations, `A0` is the register address of accelerations. Therefore, the linear accelerations of 3 axes are：

  AccX = (int16_t)((0xFF<<8)+ 0xEA) = -22
  AccY = (int16_t)((0x03<<8)+ 0xD0) = 976
  AccZ = (int16_t)((0xFF<<8)+ 0x45) = -187
- `B0 00 00 00 00 00 00` are angular velocities, `B0` is the register address of angular velocities. From these values, we find that angular velocities around 3 axes are all zero.
- `D0 87 00 6F 27 F5 FF` are Euler angles, `D0` is the register address of Euler angles. From these values, we find that :

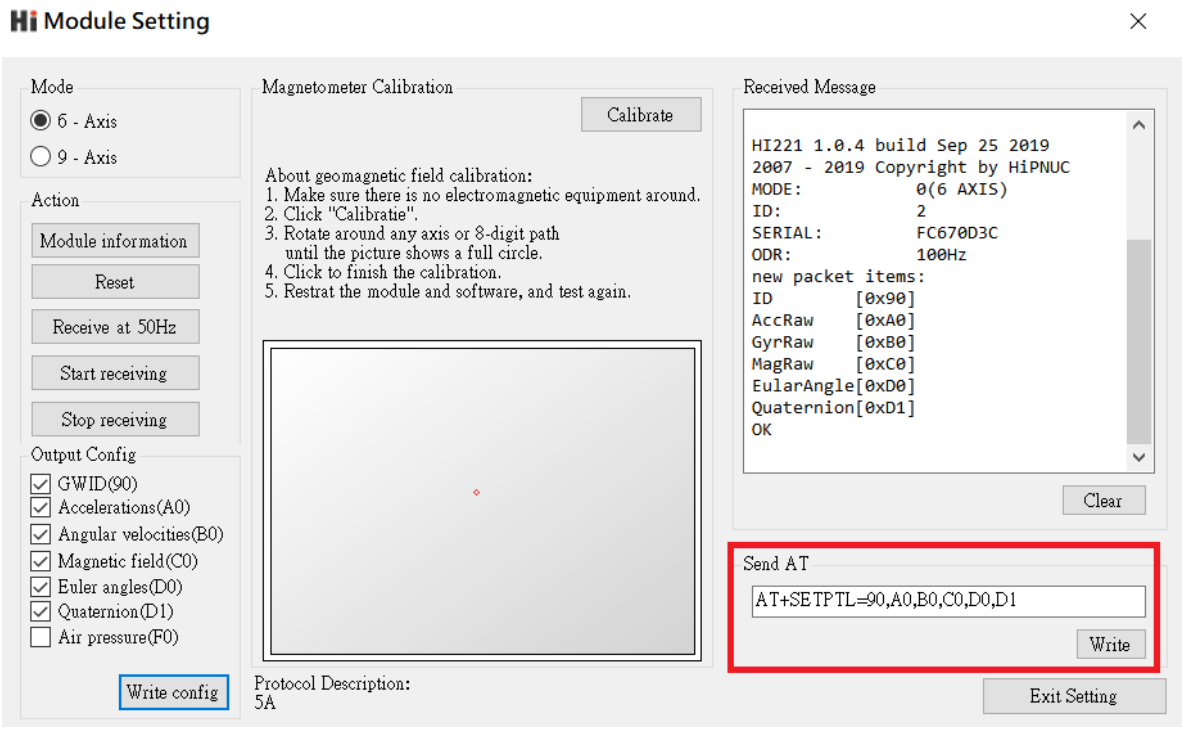
  Pitch= (int16_t)((0x00<<8)+ 0x87) / 100 = 1.35°
  Roll= (int16_t)((0x27<<8)+ 0x6F) / 100 = 100.95°
  Yaw = (int16_t)((0xFF<<8)+ 0xF5) / 10 = -1.1°
- Calculate the CRC ：
  Remember that the frame of data is received and stored in the buffer of C language **uint8_t** array :

```
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);
```

After calculating, the CRC checksum is 0x8BA9, same as the CRC value carried in the frame. The check result is correct

General AT Command

The Module parameters can be configured and checked by AT commands. AT commands always start with the ASCII code `AT`, followed by the control characters, and end with a carriage return and linefeed `\r\n`. You can use any serial debugging assistant for testing.



General AT Commands :

Command	Function	Configure once (N) / Configure permanent after restart (Y)
AT+ID	Set a user ID for the module	Y
AT+GWID	Assign an ID to the wireless network domain (for wireless product)	Y
AT+URFR	Rotate the coordinate system of the module	Y
AT+INFO	Print out the information of module	N

Command	Function	Configure once (N) / Configure permanent after restart (Y)
AT+ODR	Set the output frequency for a frame of module data	Y
AT+BAUD	Set Baud for serial port	Y
AT+EOUT	A switch for the output data	N
AT+RST	Reset the module	N
AT+TRG	Trigger the module to output a frame	N
AT+SETPEL	Configure the content in a frame	Y
AT+MODE	Set an operation mode of the module	Y

AT+ID

Set a user ID for the module

ex. `AT+ID=1`

AT+GWID

Only support HI221. HI221GW (receiver) and HI221 (node) have GWID attribute, you can assign a number of GWID for specific radio frequency by AT+GWID command, and only when both node and the receiver are in the same GWID, they can communicate with each other. GWID is just like a wireless network domain. If you're using more than one receiver to establish multiple star networks, you have to assign different GWID to each receiver。

ex. set GWID=3 for a receiver, meanwhile there are three nodes are individually set to 0,1, and 2. Let them be able to communicate with the receiver.

Command for

receiver：`AT+GWID=3`

node 0： `AT+GWID=3 AT+ID=0`

node 1：`AT+GWID=3 AT+ID=1`

node 2：`AT+GWID=3 AT+ID=2`

AT+URFR

In some cases the IMU sensor needs to be installed tilted or vertically. This command helps you to rotate the coordinate system of the sensor：

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

where C_{nn} support float and double type.

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

where $\begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix}_U$ are the measurement data after coordinate system correction, and $\begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix}_B$ are the measurement data before coordinate system calibration.

Some examples of commands：

- Rotate N° around original X or Y or Z axis as a new coordinate system
  - 90° around original X axis : `AT+URFR=1,0,0,0,0,1,0,-1,0`
  - -90° around original X axis : `AT+URFR=1,0,0,0,0,-1,0,1,0`
  - 180° around original X axis : `AT+URFR=1,0,0,0,-1,0,0,0,-1`
  - 90° around original Y axis : `AT+URFR= 0,0,-1,0,1,0,1,0,0`
  - -90° around original Y axis : `AT+URFR= 0,0,1,0,1,0,-1,0,0`
  - 180° around original Y axis : `AT+URFR= -1,0,0,0,1,0,0,0,-1`
- Factory reset：`AT+URFR=1,0,0,0,1,0,0,0,1`

AT+INFO

Print the module information, including model, version, firmware and release date, etc. There are secondary instructions for AT + INFO to achieve more information.

INFO secondary instruction	Function	Example

INFO secondary instruction	Function	Example
CAL	Print internal calibration parameters of the module.	AT+INFO=CAL
RF	Print parameters of the wireless product.	AT+INFO=RF
VER	Print details of the firmware version	AT+INFO=VER

AT+ODR

Set the serial output rate of the module. It can be stored when the power off, and takes effect after restarting the module.

ex. set the rate to 100Hz: `AT+ODR=100`

AT+BAUD

Set Baud only in these options：4800/9600/115200/256000/460800`

ex. `AT+BAUD=115200`

!!! Notice

- Beware that wrong Baud will result in failure of communication with the module.
- The receiver and module must be in the same Baud.
- Baud must be set to 115200 before you updating the firmware.

AT+EOUT

A switch of the output from module.

ex.

- Open the serial port of module : `AT+EOUT=1`
- Close the serial port of module : `AT+EOUT=0`

AT+RST

Reset the module.

ex. `AT+RST`

AT+TRG

Trigger the module to output a frame. It can cooperate with AT + ODR = 0 to trigger a single output。

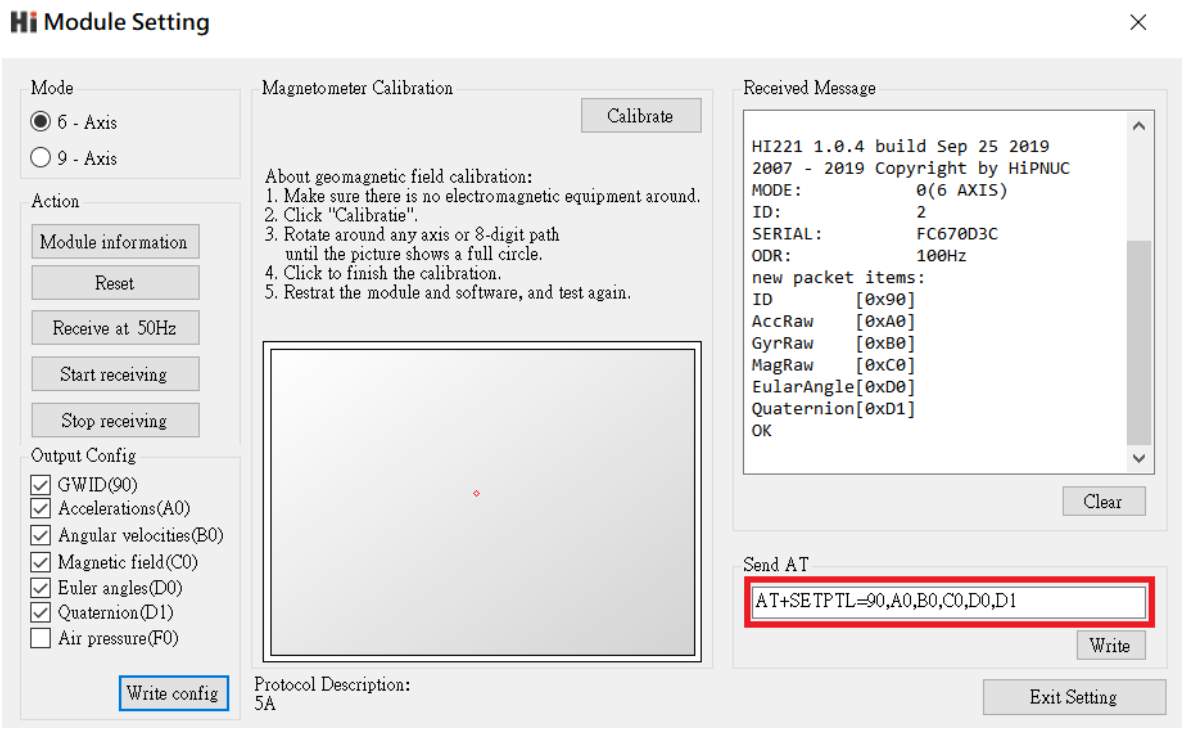
ex. `AT+TRG`

AT+SETPEL

Set the output protocol:

The content in a frame of data can be configured using AT commands, by following the format:
`AT+SETPTL=<ITEM_ID>,<ITEM_ID>...`
A frame of data can contain up to 8 packets.

ex. Configure the module to output acceleration, angular velocity, Euler angle and quaternion in the format ：`AT+SETPTL=A0,B1,D0,D1`



AT+MODE

Set the operation mode for the module.

ex.

- Set the module to work in 6-axis mode (without magnetic calibration) `AT+MODE=0`
- Set the module to work in 9-axis mode (will calibrate the heading angle by geomagnetic field sensor) `AT+MODE=1`

Appendix B - Conversion Between Quaternion and Euler Angles

Basic conceptions of quaternion

Quaternion is a number system that extends the complex numbers, representing a point in four-dimensional space：$q \in \mathbb{R}^4 = \mathbb{H}$

This table shows several representation of quaternions：

in complex numbers	in vector	representation 1
$q = q_0 + \mathrm{i}q_1 + \mathrm{j}q_2 + \mathrm{k}q_3$	$q = [q_0, \mathbf{q}] = \begin{bmatrix} q_0, \begin{pmatrix} q_1 \\ q_2 \\ q_3 \end{pmatrix} \end{bmatrix}$	$q = [q_0, q_1, q_2, q_3]$

How to multiplicate basis elements：

$$\mathrm{i}^2 = \mathrm{j}^2 = \mathrm{k}^2 = \mathrm{ijk} = -1$$

$$\mathrm{ij} = \mathrm{k} = -\mathrm{ji}, \quad \mathrm{jk} = \mathrm{i} = -\mathrm{kj}, \quad \mathrm{ki} = \mathrm{j} = -\mathrm{ik}$$

How to multiplicate two quaternions：

$$\mathbf{p} \otimes \mathbf{q} = \begin{bmatrix} p_w q_w - p_x q_x - p_y q_y - p_z q_z \\ p_w q_x + p_x q_w + p_y q_z - p_z q_y \\ p_w q_y - p_x q_z + p_y q_w + p_z q_x \\ p_w q_z + p_x q_y - p_y q_x + p_z q_w \end{bmatrix}$$

An unit quaternion can always can be ：$q_R(\alpha, \mathbf{u}) = \left[\cos\frac{\alpha}{2}, \sin\frac{\alpha}{2} \cdot \mathbf{u}\right]$

where α is rotation angle，$\mathbf{u} \in \mathbb{R}^3$ is rotation axis，and $\|\mathbf{u}\| = 1$.

Conversion between quaternions, rotation matrices, and Euler angles

Quaternion -> Rotation matrix

(Where quaternion is b->n, R is n->b)

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

Quaternion -> Euler angles

Rotation matrix, quaternion and Euler angles are three common ways to represent rotation. However, the rotation order must be specified first before you converse **quaternion to Euler angles** and **rotation matrix to Euler angles**. This product uses the "ZYX" rotation sequence which rotates heading angle first, and then the pitch angle, and the last is roll angle.

Formula :

$$\begin{bmatrix} \phi(Roll) \\ \theta(Pitch) \\ \psi(Heading) \end{bmatrix} = \begin{bmatrix} \operatorname{atan2}\left(2q_2q_3 + 2q_0q_1, q_3^2 - q_2^2 - q_1^2 + q_0^2\right) \\ -\operatorname{asin}(2q_1q_3 - 2q_0q_2) \\ \operatorname{atan2}\left(2q_1q_2 + 2q_0q_3), q_1^2 + q_0^2 - q_3^2 - q_2^2\right) \end{bmatrix}$$

Euler angles -> Quaternion

From $s_\phi = \sin\frac{\phi}{2}, c_\phi = \cos\frac{\phi}{2}$, we got :

$$\mathbf{q} = \begin{bmatrix} c_{\phi/2}c_{\theta/2}c_{\psi/2} + s_{\phi/2}s_{\theta/2}s_{\psi/2} \\ -c_{\phi/2}s_{\theta/2}s_{\psi/2} + c_{\theta/2}c_{\psi/2}s_{\phi/2} \\ c_{\phi/2}c_{\psi/2}s_{\theta/2} + s_{\phi/2}c_{\theta/2}s_{\psi/2} \\ c_{\phi/2}c_{\theta/2}s_{\psi/2} - s_{\phi/2}c_{\psi/2}s_{\theta/2} \end{bmatrix}$$

Euler angles -> Rotation matrix (n->b)

$$R_n^b = \begin{bmatrix} c_\theta c_\psi & c_\theta s_\psi & -s_\theta \\ s_\phi s_\theta c_\psi - c_\phi s_\psi & s_\phi s_\theta s_\psi + c_\phi c_\psi & c_\theta s_\phi \\ c_\phi s_\theta c_\psi + s_\phi s_\psi & c_\phi s_\theta s_\psi - s_\phi c_\psi & c_\theta c_\phi \end{bmatrix}$$

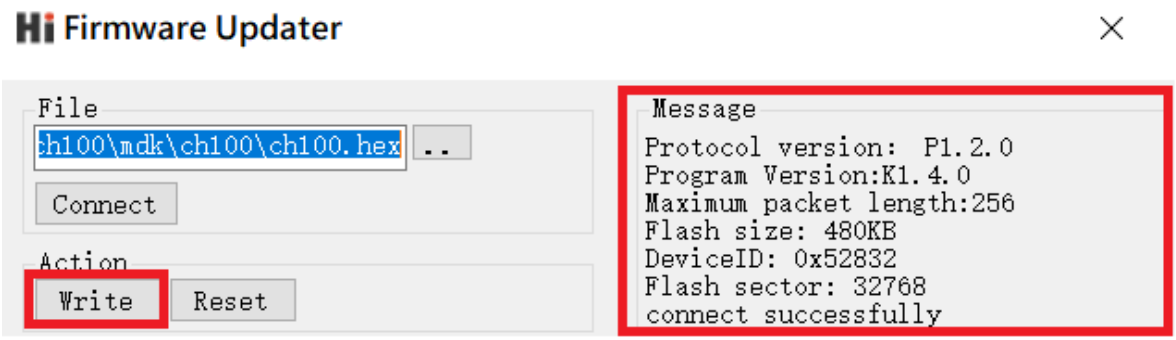
Rotation matrix (n->b) -> Euler angles

$$\begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} \operatorname{atan2}\left(r_{23}, r_{33}\right) \\ -\operatorname{asin}(r_{13}) \\ \operatorname{atan2}\left(r_{12}, r_{11}\right) \end{bmatrix}$$

Appendix C - Firmware Upgrade and Factory Reset

This product supports online firmware upgrade. Please pay attention to the official website of Supercore Electronics www.hipnuc.com for the latest firmware.
Firmware upgrade steps:

- Get the latest firmware file. The extension of the file is (.hex).
- Connect the module, and run "Uranus". Switch to the firmware upgrade window, and set Baud (Baudrate) to 115200.
- Click "connect" button. If the module information shows successfully, meaning that the system is ready to upgrade.
- Now you can click the file selector (...), and select the firmware with the extension xxx.hex and click to start programming. After the download is completed, there will be a successful notification.
- Close the serial port and restart the module. Now it's upgraded.



---

1. 1G = 1x (Local gravitational acceleration)↩↩