FORSENSE
原极科技

A800 Dynamic Inclination Measurement Module

A800 Product Manual

Characteristic

Tactical MEMS IMU

- 4.0°/h bias stability of gyroscope
- 30ug bias stability of accelerometer

Anti jamming dynamic obliquity algorithm

- Pitch / roll angle:<0.2° rms @Static
- <0.5° rms @Dynamic

Turntable Calibration

- Independent calibration for each module: sensitivity, bias, non-orthogonal error
- Provide user installation error calibration interface

High reliability

- Super shock resistance: 2000g (0.5ms, half sine, 3-axis)
- Super shock vibration:10g(10~2KHz,3-axis)
- Stable operation at full temperature: -40 ℃~85℃
- 100% magnetic shielding

Flexible digital interface, small size

- Configurable output sampling rate up to 100Hz
- Support serial port, CAN interface

Product description

A800 is a kind of dynamic inclination measurement sensor based on industrial IMU platform, which is developed by Forsence technology for construction machinery, intelligent robots and other fields. Fine factory calibration and temperature compensation ensure the consistency and stability of the module in the full temperature range. The built-in attitude fusion algorithm based on extended Kalman filter can effectively suppress the influence of dynamic interference on attitude accuracy and ensure the stability of attitude accuracy.

On the basis of standard performance and output parameters, Forsense also provides **customized services for your special needs**, and helps you in the product!

1. Performance index

The performance indexes of A800 are shown in Table 1:

Table 1 Performance Index

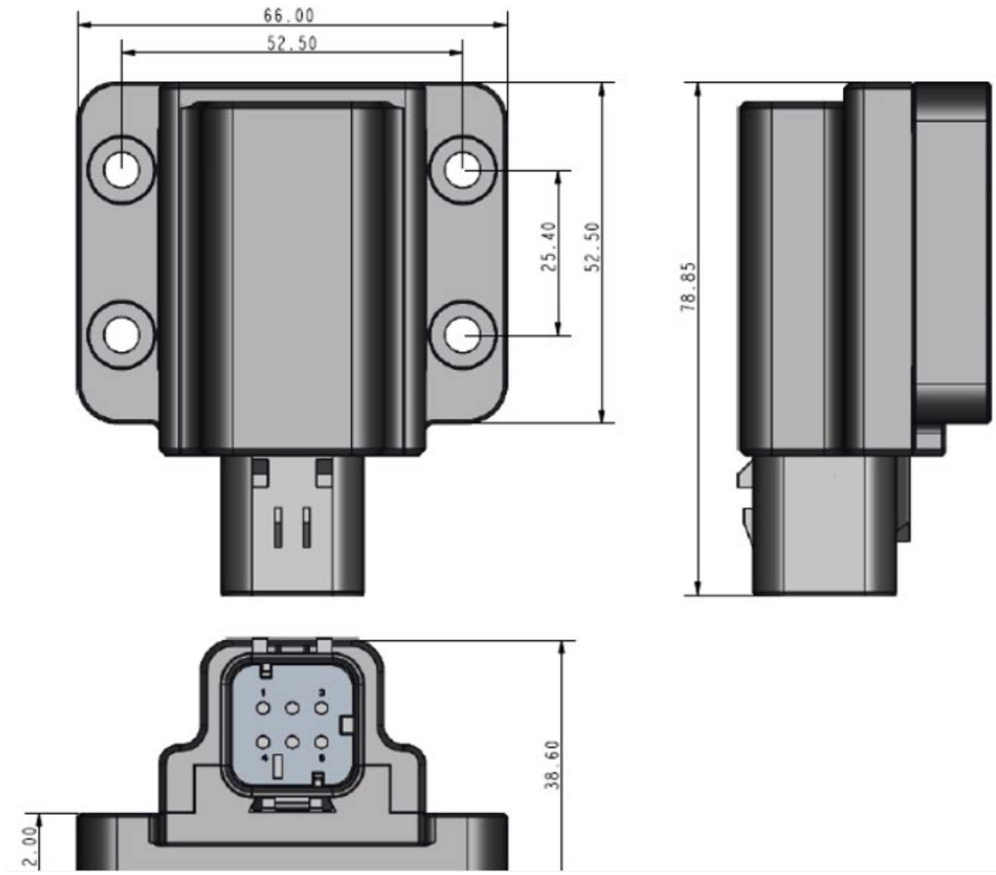
Sensor Performance	
Angle measurement range	Pitch angle: ±80° Roll angle: ±180°
Angle repeatability	<0.05°
Measurement range of angular velocity	±450°/s
Static accuracy[1]	0.2°
Dynamic accuracy[2]	0.5°
Acceleration measurement range	±8g
Renewal rate	100hz
Electrical characteristics	
Voltage input	8-32 V
power waste	
port	
Physical characteristics	
Connector model	DJ3062-1.6-11
size	66*52.2cm
Waterproof grade	IP67
Ambient temperature	
Operating temperature	-40~85℃
Storage temperature	-40~85℃

[1]RMS value of attitude error in static 25 ℃ environment

[2]RMS value of attitude error in vehicle test over 1 hour

The dimension drawing of A800 is shown in Figure 1:

Figure 1 A800 Product Dimension Diagram



The interface definition of A800 is shown in Table 2:

Table 2 A800 Interface Definition

PIN	Interface definition
1	CAN H
2	CAN L
3	GND
4	RS232 RX
5	TS232 TX
6	POWER

2. Temperature stability test

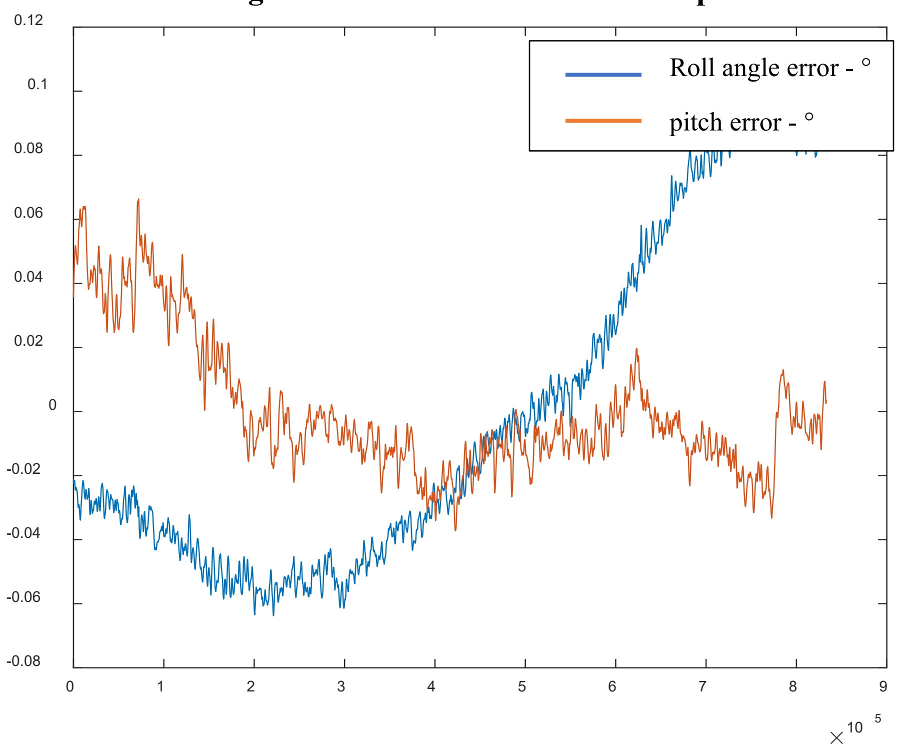
Test conditions:

A800 products were randomly selected and placed in vibration isolation temperature box.

Control the incubator to cycle from - 40 ℃ to 85 ℃ with a slope of 3 ℃ / min;

Take the attitude of 25 ℃ as the initial attitude.

Figure2 Attitude error curve--Temperature



3. Scene accuracy test of Construction Machinery

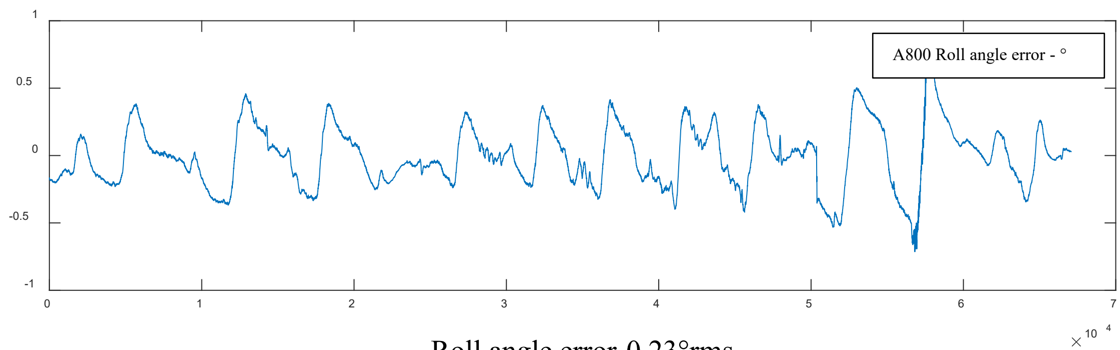
Test conditions:

Random sampling of A800 products;

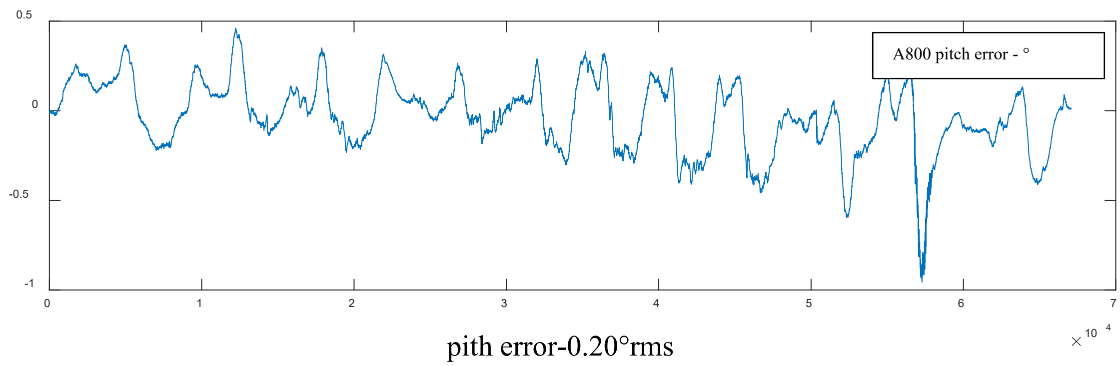
Fixed the A800 directly under the cab seat using the tractor as the test vehicle;

Driving speed is not greater than 18km/h;

The true value datum is postprocessed using Novotay SPAN-IGM-S1(it’s attitude accuracy is better than 0.015 degrees);



Roll angle error-0.23°rms



pith error-0.20°rms

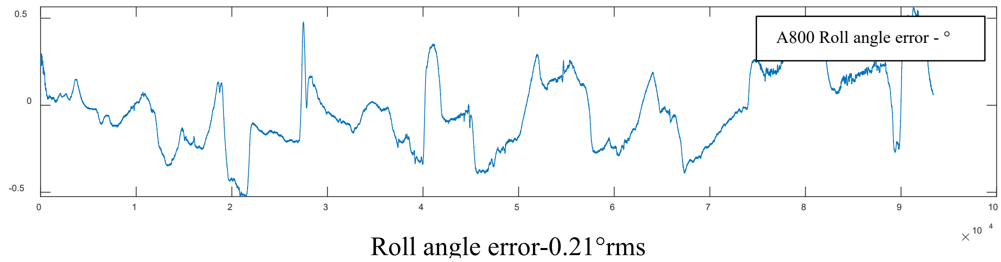
4. Accuracy Test of Passenger Car Scene

Test conditions:

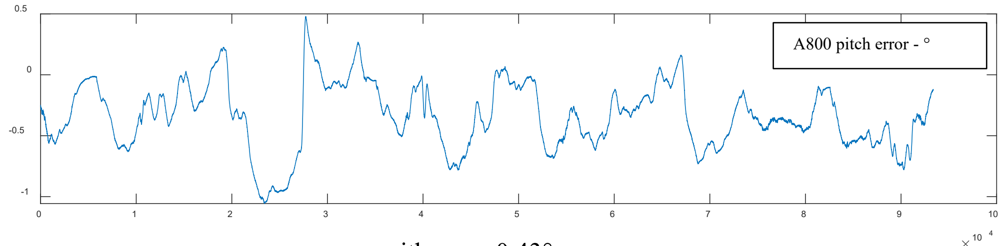
Random sampling of A800 products;

The test vehicle is a Toyota SUV with RAV4 and the test scene is an urban road.

True value benchmarks with travel speeds not greater than 60km/h are post-processed using Novotay SPAN-IGM-S1 (posture accuracy is better than 0.015 degrees);



Roll angle error-0.21°rms



pith error-0.43°rms

5. CAN Communication Protocol

Introduction to SAEJ1939 Protocol:

The A800 supports CAN's high-level protocol SAEJ1939, which manages communication within the network. J1939 is a set of standards defined by SAE. The physical layer (J1939/11) describes the electrical bus interface. The data link layer (J1939/21) describes the rules for constructing messages, accessing the bus, and detecting transmission errors. The application layer (J1939/71 and J1939/73) defines the specific data contained in each message sent over the network.

The A800 uses 29-bit identifiers defined in the CAN 2.0b protocol, as shown below:

Priority	Reserved Bits	Data Page	PDU Format Bit	PDU Specific Bit	Address Bit
3bits	1bits	1bits	8bits	8bits	8bits

Priority: The first three digits of an identifier are used to control the priority of a message during mediation. The zero value has the highest priority. High - speed control communication is usually given a higher priority value.

Reserved bit: Reserved bit for future use and should be set to zero when sending messages.

Data page: Used for data page selection

PDU format bit: Determines whether a message can be sent with the destination address or whether a message is always sent as a broadcast message.

PDU specific bit: The bit value of a PDU specific bit is related to the PF value. If PF is between 0 and 239, the message is addressable and the PDU specific bits contain the destination address. If PF is between 240 and 255, only messages can be broadcast, and PDU specific bits contain an extended group.

Address bit: address of the device that sends the message.

The term (PGN) refers to a combination of reserved bits, data pages, PDU format bits, and PDU specific bit fields into an 18-bit value.

5.1 A800 CAN protocol identifier parameters:

Table3 A800 protocol identifier

Message Name	PNG	PDU Format Bit	PDU Specific Bit	Data Length	Application
Slope sensor Information (SSI2)	61481	240	41	8bits	High resolution pitch and roll
Angular velocity information (ARI)	61482	240	42	8bits	Pitch, roll, yaw, angular velocity
Accelerator sensor (ACS)	61485	240	45	8bits	X, Y, Z, acceleration

Table4 A800 CAN communication 32-bit ID calculation method:

Reserved Bits	Priority Bits	Extension Page Bit	Data Page Bit	PDU Format Bit	PDU Specific Bit	Address Bit
[31:29]	[28:26]	[25:25]	[24:24]	[23:16]	[15:8]	[7:0]

5.2 Calculation of CAN communication ID value:

5.2.1 Bucket ID value calculation (coordinate value 201):

Table5 Calculate bucket ID value parameter table

Message Name	Reserved Bits	Priority	Extension Page	Data Page	PDU Format Bit	PDU Specific Bit	Address Bit
SSI2	0	3	0	0	240	41	129
ARI	0	2	0	0	240	42	129
ACS	0	2	0	0	240	45	129

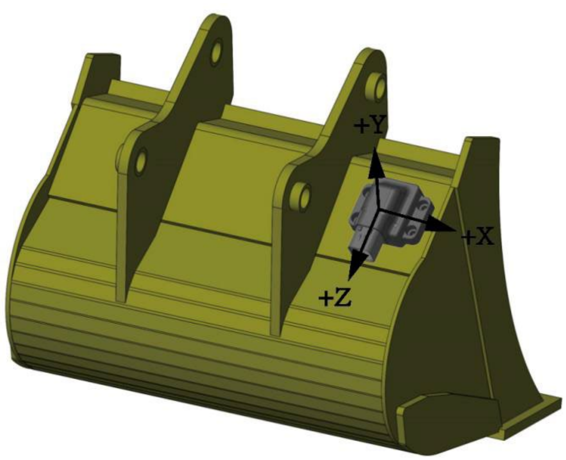
Result: Slope sensor information (SSI2) ID is 0x0CF02981

Angular velocity information (SSI2) ID: 0x0CF02A81

Accelerator sensor (ACS)ID: 0x08F02D81

序号	时间	消息名	帧ID(Hex)	PGN(Hex)	源&目标地址(Hex)	帧类型	计数	方向	长度	数据(Hex)
0	17:51:29.986	--	08F02D81	00F02D	81 --	扩展数据帧	8502	Rx	8	08 7D 30 79 30 79 00 00
1	17:51:29.986	--	0CF02A81	00F02A	81 --	扩展数据帧	8502	Rx	8	E9 7C 01 7D 01 7D 00 00
2	17:51:29.986	--	0CF02981	00F029	81 --	扩展数据帧	8502	Rx	8	9F 2B 7D C5 7F 93 00 00

Figure3 Bucket diagram (coordinate value 201)



5.2.2 Calculation of forearm ID value (coordinate value 18):

Table6 Calculate the parameter table of arm ID value

Message Name	Reserved Bits	Priority	Extension Page	Data Page	PDU Format Bit	PDU Specific Bit	Address Bit
SSI2	0	3	0	0	240	41	130
ARI	0	2	0	0	240	42	130
ACS	0	2	0	0	240	45	130

Result: Slope sensor information (SSI2) ID: 0x0CF02982

Angular velocity information (SSI2) ID: 0x0CF02A82

Accelerator sensor (ACS)ID: 0x08F02D82

序号	时间	消息名	帧ID(Hex)	PGN(Hex)	源&目标地址(Hex)	帧类型	计数	方向	长度	数据(Hex)
0	17:54:25.889	--	08F02D82	00F02D	82 --	扩展数据帧	474	Rx	8	09 7D 2F 79 2F 79 00 00
1	17:54:25.890	--	0CF02A82	00F02A	82 --	扩展数据帧	474	Rx	8	F7 7C 05 7D 05 7D 00 00
2	17:54:25.890	--	0CF02982	00F029	82 --	扩展数据帧	474	Rx	8	77 2E 7D C5 7F 93 00 00

5.2.3 Calculation of big arm ID value (coordinate value 18):

Table7 Calculate the parameter table of the ID value of the big arm

Message Name	Reserved Bits	Priority	Extension Page	Data Page	PDU Format Bit	PDU Specific Bit	Address Bit
SSI2	0	3	0	0	240	41	131
ARI	0	2	0	0	240	42	131
ACS	0	2	0	0	240	45	131

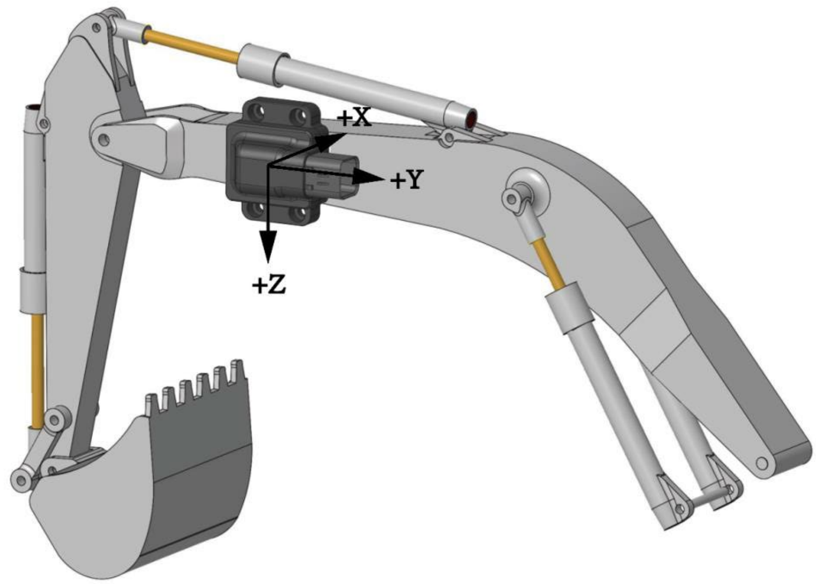
Result: Slope sensor (SSI2) ID: 0x0CF02983

Angular velocity information (SSI2) ID: 0x0CF02A83

Accelerator sensor (ACS)ID: 0x08F02D83

序号	时间	消息名	帧ID(Hex)	PGN(Hex)	源&目标地址(Hex)	帧类型	计数	方向	长度	数据(Hex)
0	17:55:22.265	--	08F02D83	00F02D	83 --	扩展数据帧	405	Rx	8	08 7D 2E 79 2E 79 00 00
1	17:55:22.265	--	0CF02A83	00F02A	83 --	扩展数据帧	405	Rx	8	E9 7C 00 7D 00 7D 00 00
2	17:55:22.265	--	0CF02983	00F029	83 --	扩展数据帧	405	Rx	8	D5 2D 7D C8 7F 93 00 00

Figure4 Schematic diagram of big arm (coordinate value 18)



5.2.4 Calculation of body ID value (coordinate value 228):

Table8 Calculate the body ID value parameter table

Message Name	Reserved Bits	Priority	Extension Page	Data Page	PDU Format Bit	PDU Specific Bit	Address Bit
SSI2	0	3	0	0	240	41	128
ARI	0	2	0	0	240	42	128
ACS	0	2	0	0	240	45	128

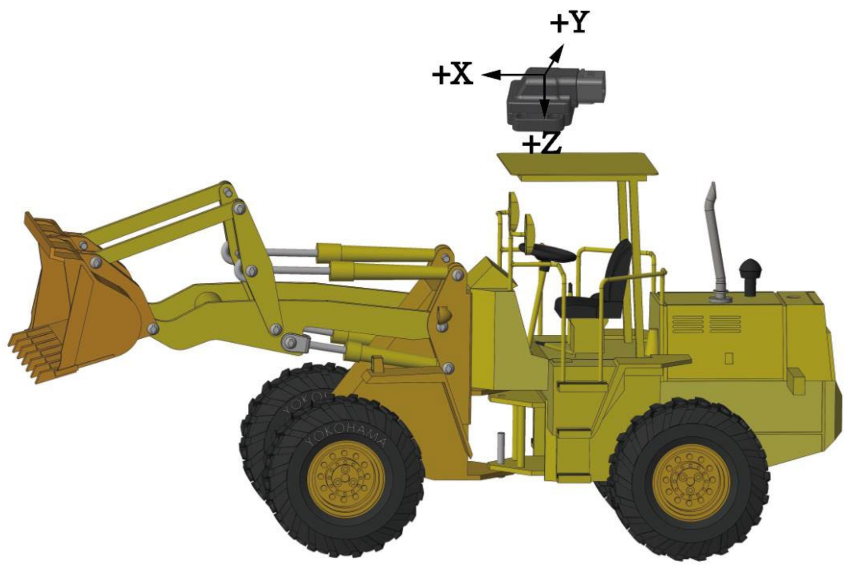
Result: Slope sensor information (SSI2) ID is 0x0CF02980

Angular velocity information (SSI2) ID: 0x0CF02A80

Accelerator sensor (ACS)ID: 0x08F02D80

序号	时间	消息名	帧ID(Hex)	PGN(Hex)	源&目标地址(Hex)	帧类型	计数	方向	长度	数据(Hex)
0	17:55:53.939	--	08F02D80	00F02D	80 --	扩展数据帧	246	Rx	8	08 7D 30 79 30 79 00 00
1	17:55:53.939	--	0CF02A80	00F02A	80 --	扩展数据帧	246	Rx	8	FB 7C 05 7D 05 7D 00 00
2	17:55:53.939	--	0CF02980	00F029	80 --	扩展数据帧	246	Rx	8	FC 2D 7D C7 7F 93 00 00

Figure5 Body diagram (coordinate value 228)



5.3 Can communication Data Definition (8 bytes)

Description:

The A800 CAN communication payload contains eight bytes and is in small-endian mode. The default baud rate is 500Kbs， following the SAEJ1939-DA definition. Specific definitions are as follows:

Table9 Slope sensor Information (SSI2) data definition

Bits	Parameter Names	Unit	Dimension scale	Range of value	Offset
1-3	Angle of Pitch	°	1/32768 °/Bit	-250 to 252	-250
4-6	Roll Position	°	1/32768 °/Bit	-250 to 252	-250
7-8	Reserved	-	-	-	-

Table10 Angular velocity information (ARI) data definition

Bits	Parameter Names	Unit	Dimension scale	Range of value	Offset
1-2	Rolling angular velocity	°/s	1/128 °/s/Bit	-250 to 250.99	-250
3-4	Pitch velocity	°/s	1/128 °/s/Bit	-250 to 250.99	-250
5-6	Yaw velocity	°/s	1/128 °/s/Bit	--250 to 250.99	-250
7-8	Reserved	-	-	-	-

Table11 Accelerator sensor (ACS)

Bits	Parameter Names	Unit	Dimension scale	Range of value	Offset
1-2	X-axis acceleration	m/s2	0.01 m/s2/Bit	-320 to 322.55	-320
3-4	Y-axis acceleration	m/s2	0.01 m/s2/Bit	-320 to 322.55	-320
5-6	Z-axis acceleration	m/s2	0.01 m/s2/Bit	-320 to 322.55	-320
7-8	Reserved	-	-	-	-

6. Serial communication protocol

Qt-based serial port protocol example:

https://github.com/forsense/qt_serial_protocol_demo

The serial port communication has two modes: Stream Mode and Command Mode. After the IMU is powered on and initialized, the IMU enters the corresponding Mode based on the configured Mode values.

Data flow mode: Periodically outputs AHRS data at a fixed frequency.

Command mode: In this mode, the user stops the periodic output and communicates with the IMU by sending commands. The user can use the GET command to obtain sensor data, status, and parameters, and configure IMU parameters.

6.1 Serial port parameters:

Table12 Serial Port Parameters

Transmission rate range	115200bps ~ 1.5Mbps
Default transmission rate	15200bps
Start bit	1 bit
Data bit	8 bits
Stop bit	1 bit
Odd-even check	-

6.2 Packet format

The data packet structure of IMU output and user input is as follows:

Table13 IMU output and user input data structures

Offset	Data type	Name	Description
0	Unit8	The frame head 1	IMU output frame header: 0xAA， 0x55 User input frame header: 0x55， 0xAA
1	Unit8	The frame head 1	
2	Unit16	ID low	The low byte of the serial port communication frame ID
3		ID high	The high byte of the serial port communication frame ID
4	Unit16	Data length low	The low byte of the serial port communication frame. Length is the number of bytes taken by the Payload, that is n
5		Data length high	The high byte of the serial port communication frame. Length is the number of bytes taken by the Payload, that is n
6	Unit8	Payload (n bit)	Data load
6+n		CRC_CEHCK (low bit)	CRC check
7+n	Unit32	CRC_CEHCK (mid-low bit)	
8+n		CRC_CEHCK (mid-high bit)	
9+n		CRC_CEHCK (high bit)	

[1]Data is transferred in a small-endian format, with low bytes first and high bytes last.

[2]The initial value of CRC32 is 1. CRC calculation does not include all the data of this frame. Refer to the table lookup calculation method at the end of the document.

6.3 Data stream frames - AHRS data

Table14 Serial port AHRS data format

Type	Frame head	Frame head	ID	length	payload	Frame tail
Data type	uint8	uint8	Uint16	Uint16	A1	Uint32
Code	0xAA	0x55	0x0002	0x002C		Crc32

The maximum output update rate is not greater than 200Hz@115200bps

Table15 Format of serial port A1 load data

Offset	Name	Data type	Unit	Description
0	timer	uint32	μs	Time scale
4	pitch	float	°	Angle of Pitch
8	roll	float	°	Angle of Roll
12	yaw	float	°	Angle of Course
16	ax	float	g	X-axis acceleration
20	ay	float	g	Y-axis acceleration
24	az	float	g	Z-axis acceleration
28	gx	float	°/s	X angular velocity
32	gy	float	°/s	Y angular velocity
36	gz	float	°/s	Z angular velocity
40	temp	float	℃	Temperature of IMU chip

6.4 Command mode GET Output - System status

Table16 Serial port system status data format

Type	Frame head	Frame head	ID	length	payload	Frame tail
Data type	uint8	uint8	Uint16	Uint16	S1	Uint32
Code	0xAA	0x55	0x00FF	0x0042		Crc32

Table17 Serial port S1 Load data format

Offset	Name	Data type	Description
0	Software-ver	uint32	Software Version number
4	Hardware-ver	uint32	Hardware Version number
8	Board-ver	uint32	IMU model
10	sn0	uint32	The first SN no.
14	sn1	uint32	The second SN no.
18	sn2	uint32	The third SN no.
22	system_status	uint32	System state
26	cpu_load	Uint16	CPU usage
28	drop_rate_com	Uint16	Communication error bit count
30	rev[9]	uint32	Reserved byte

6.5 Command mode GET Output - Reads parameters

Table18 Serial port parameter Output data format

Type	Frame head	Frame head	ID	length	payload	Frame tail
Data type	uint8	uint8	Uint16	Uint16	P1	Uint32
Code	0xAA	0x55	0x0006	0x0018		Crc32

Table19 Serial port P1 Load data format

Offset	Name	Data type	Description
0	Param1	float	Set parameters
4	Param2	float	Reserved, the default value is 0
8	Param3	uint32	Parameter index to set
12	Param4	uint32	Reserved, the default value is 0
16	Param5	int32	Reserved, the default value is 0
20	Param6	int32	Reserved, the default value is 0

Table20 Serial port P1 Load parameter index

Param3	Param1	Unit
3	The serial port supports the following baud rates: 115200,230400,460800,921600,1500000	bps
8	Zero deviation calibration results of X-axis gyro, GYRO_X_OFF	°/s
9	Zero deviation calibration results of Y-axis gyro, GYRO_Y_OFF	°/s
10	Zero deviation calibration results of Z-axis gyro, GYRO_Z_OFF	°/s
16	X axis installation error Angle, NSTALL_X_OFF	°
17	Y axis installation error Angle, NSTALL_Y_OFF	°
18	Z axis installation error Angle, NSTALL_Z_OFF	°
21	AHRS output frequency, default 100Hz	Hz

6.6 Command mode SET command

Table21 Serial port input command format

Type	Frame head	Frame head	ID	length	payload	Frame tail
Data type	uint8	uint8	Uint16	Uint16	R1	Uint32
Code	0x55	0xAA	CMD	0x0018		Crc32

Note 1: For the relationship between CMD and R1, see the INDEX table of R1 load parameters

Table22 Serial port R1 Load data format

Offset	Name	Data type	Description
0	Param1	float	Set parameters
4	Param2	float	Reserved, the default value is 0
8	Param3	uint32	Parameter index to set
12	Param4	uint32	Reserved, the default value is 0
16	Param5	int32	Reserved, the default value is 0
20	Param6	int32	Reserved, the default value is 0

Table23 Serial port R1 Load parameter index

CMD	Param1	Param3	Description
1	0	0	Trigger to obtain system status data once
2	0	0	Trigger to get AHRS data once
3	<mode>	0	Set the output mode: Mode=1, data stream output AHRS Mode=100, disable data flow Mode and enter COMMAD Mode
4	<heading>	0	Reset AHRS heading Angle: The heading input ranges from 0 to 360, in degrees
5	0	0	Save the current parameter to FLASH
6	0	<value>	Value is the index of the parameter to be read, and that is P1.index For example, to read THE AHRS output frequency (ODR), set value to 21
7	0	0	Start installation error calibration
8	0	0	Trigger to obtain calibration status of one installation error
9	0	0	Performing a software restart
10	0	0	Perform static gyro zero offset calibration
14	<value>	3	Set the baud rate (BPS) of the serial port output Value Can be 115200, 230400, 460800, 921600, 1500000 If value is another value, 115200bps is used by default
14	<value>	21	Set the output frequency of periodic AHRS data, in Hz A valid value of value is: 1, 10, 50, 100, 200, 400 If value is another value, 100Hz is used by default

Note 1: Please note that the values in this table are decimal.

Note 2: You can use the command generator function of upper computer to generate corresponding commands and send them. For the usage method, see the upper computer Usage section of this manual.

6.7 Command mode output - User command response

Table24 Format of serial port user command response data

Type	Frame head	Frame head	ID	length	payload	Frame tail
Data type	uint8	uint8	Uint16	Uint16	K1	Uint32
Code	0xAA	0x55	0x0064	0x0007		Crc32

Table25 Serial port R1 Load data format

Offset	Name	Data type	Description
0	command	Uint16	ID of the command to be responded
2	result	Uint16	Result

7. CRC32 table lookup method

```
static const uint32_t crc32_tab [ ] = {
0x00000000, 0x77073096, 0xee0e612c, 0x990951ba, 0x076dc419, 0x706af48f, 0xe963a535, 0x9e6495a3,
0x0edb8832, 0x79dcb8a4, 0xe0d5e91e, 0x97d2d988, 0x09b64c2b, 0x7eb17cbd, 0xe7b82d07, 0x90bf1d91,
0x1db71064, 0x6ab020f2, 0xf3b97148, 0x84be41de, 0x1adad47d, 0x6ddde4eb, 0xf4d4b551, 0x83d385c7,
0x136c9856, 0x646ba8c0, 0xfd62f97a, 0x8a65c9ec, 0x14015c4f, 0x63066cd9, 0xfa0f3d63, 0x8d080df5,
0x3b6e20c8, 0x4c69105e, 0xd56041e4, 0xa2677172, 0x3c03e4d1, 0x4b04d447, 0xd20d85fd, 0xa50ab56b,
0x35b5a8fa, 0x42b2986c, 0xdbbbc9d6, 0xacbcf940, 0x32d86ce3, 0x45df5c75, 0xdcd60dcf, 0xabd13d59,
0x26d930ac, 0x51de003a, 0xc8d75180, 0xbfd06116, 0x21b4f4b5, 0x56b3c423, 0xcfba9599, 0xb8bda50f,
0x2802b89e, 0x5f058808, 0xc60cd9b2, 0xb10be924, 0x2f6f7c87, 0x58684c11, 0xc1611dab, 0xb6662d3d,
0x76dc4190, 0x01db7106, 0x98d220bc, 0xefd5102a, 0x71b18589, 0x06b6b51f, 0x9fbfe4a5, 0xe8b8d433,
0x7807c9a2, 0x0f00f934, 0x9609a88e, 0xe10e9818, 0x7f6a0dbb, 0x086d3d2d, 0x91646c97, 0xe6635c01,
0x6b6b51f4, 0x1c6c6162, 0x856530d8, 0xf262004e, 0x6c0695ed, 0x1b01a57b, 0x8208f4c1, 0xf50fc457,
0x65b0d9c6, 0x12b7e950, 0x8bbeb8ea, 0xfcb9887c, 0x62dd1ddf, 0x15da2d49, 0x8cd37cf3, 0xfbd44c65,
0x4db26158, 0x3ab551ce, 0xa3bc0074, 0xd4bb30e2, 0x4adfa541, 0x3dd895d7, 0xa4d1c46d, 0xd3d6f4fb,
0x4369e96a, 0x346ed9fc, 0xad678846, 0xda60b8d0, 0x44 042d73, 0x33031de5, 0xaa0a4c5f, 0xdd0d7cc9,
0x5005713c, 0x270241aa, 0xbe0b1010, 0xc90c2086, 0x5768b525, 0x206f85b3, 0xb966d409, 0xce61e49f,
0x5edef90e, 0x29d9c998, 0xb0d09822, 0xc7d7a8b4, 0x59b33d17, 0x2eb40d81, 0xb7bd5c3b, 0xc0ba6cad,
0xedb88320, 0x9abfb3b6, 0x03b6e20c, 0x74b1d29a, 0xead54739, 0x9dd277af, 0x04db2615, 0x73dc1683,
0xe3630b12, 0x94643b84, 0x0d6d6a3e, 0x7a6a5aa8, 0xe40ecf0b, 0x9309ff9d, 0x0a00ae27, 0x7d079eb1,
0xf00f9344, 0x8708a3d2, 0x1e01f268, 0x6906c2fe, 0xf762575d, 0x806567cb, 0x196c3671, 0x6e6b06e7,
0xfed41b76, 0x89d32be0, 0x10da7a5a, 0x67dd4acc, 0xf9b9df6f, 0x8ebeeff9, 0x17b7be43, 0x60b08ed5,
0xd6d6a3e8, 0xa1d1937e, 0x38d8c2c4, 0x4fdff252, 0xd1bb67f1, 0xa6bc5767, 0x3fb506dd, 0x48b2364b,
0xd80d2bda, 0xaf0a1b4c, 0x36034af6, 0x41047a60, 0xd f60efc3, 0xa867df55, 0x316e8eef, 0x4669be79,
0xcb61b38c, 0xbc66831a, 0x256fd2a0, 0x5268e236, 0xcc0c7795, 0xbb0b4703, 0x220216b9, 0x5505262f,
0xc5ba3bbe, 0xb2bd0b28, 0x2bb45a92, 0x5cb36a04, 0xc2d7ffa7, 0xb5d0cf31, 0x2cd99e8b, 0x5bdeae1d,
0x9b64c2b0, 0xec63f226, 0x756aa39c, 0x026d930a, 0x9c0906a9, 0xeb0e363f, 0x72076785, 0x05005713,
0x95bf4a82, 0xe2b87a14, 0x7bb12bae, 0x0cb61b38, 0x92d28e9b, 0xe5d5be0d, 0x7cdcefb7, 0x0bdbdf21,
0x86d3d2d4, 0xf1d4e242, 0x68ddb3f8, 0x1fda836e, 0x81be16cd, 0xf6b9265b, 0x6fb077e1, 0x18b74777,
0x88085ae6, 0xff0f6a70, 0x66063bca, 0x11010b5c, 0x8f659eff, 0xf862ae69, 0x616bffd3, 0x166ccf45,
0xa00ae278, 0xd70dd2ee, 0x4e048354, 0x3903b3c2, 0xa7672661, 0xd06016f7, 0x4969474d, 0x3e6e77db,
0xaed16a4a, 0xd9d65adc, 0x40d f0b66, 0x37d83bf0, 0xa9bcae53 , 0xdebb9ec5, 0x47b2cf7f, 0x30b5ffe9,
0xbdbdf21c, 0xcabac28a, 0x53b39330, 0x24b4a3a6, 0xbad03605, 0xcdd70693, 0x54de5729, 0x23d967bf,
0xb3667a2e, 0xc4614ab8, 0x5d681b02, 0x2a6f2b94, 0xb40bbe37, 0xc30c8ea1, 0x5a05df1b, 0x2d02ef8d
}
uint32_t crc_crc32 (uint32_t crc, const uint8_t *buf, uint32_t size ) {
    for (uint32_t i=0; i<size ; i++) {
    crc = crc32_tab [ (crc ^ buf [i ] ) & 0xff] ^ (crc >> 8 ) ;
}
    return crc;
}
```