NAV680-SA

Dual-Antenna High-Precision Integrated Navigation System

Product Sheet

Forsense (Shanghai) Technology Co., Ltd

Please read this manual carefully before using this product

CONTENT

NAV680-SA(Water Proof) _Product Sheet

1 Product Description

1.1 Product List

When opening the package, please confirm the following products:

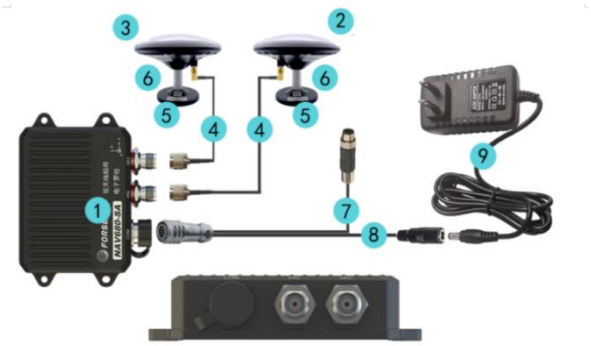
Name	Quantity	Diagram
① NAV680-SA Dual Antenna Compass	1	
②③ Satellite Antenna (No distinction between primary and secondary)	2	
(Type: 20 meters, TNC male to male)	2	
⑤ Antenna Suckers	2	
⑥ Antenna Column (20cm)	2	
⑦⑧ Harness (include 1×RS422, 1×power connector(9-24v))	1	
⑨ Power Adapter(12v, 1A)	1	

Fig.1 Product Material Drawing

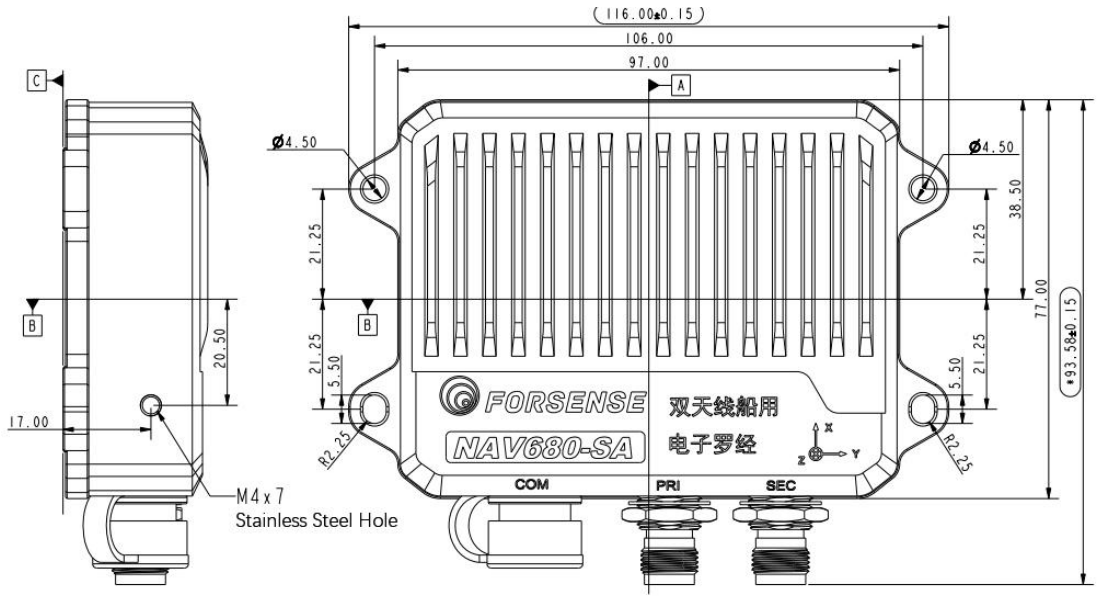
If there's anything missing, please contact salesperson in time.

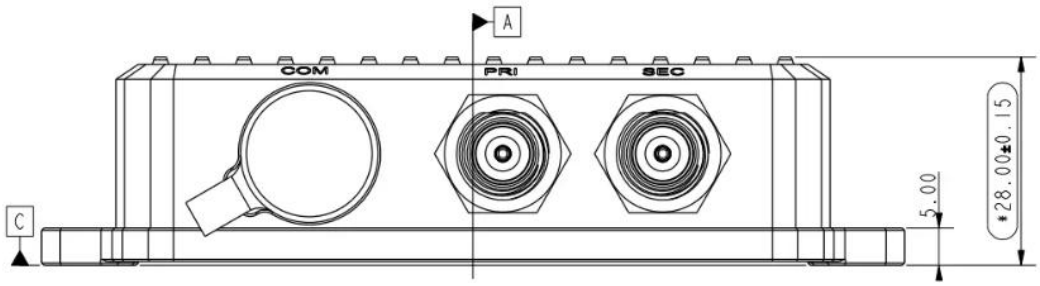
1.2 Technical Indicators

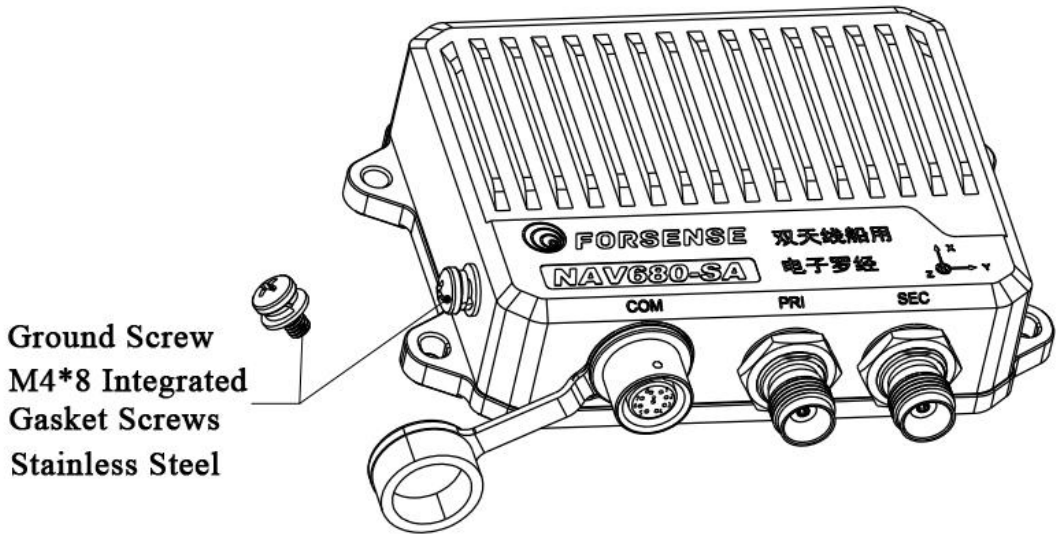
Table 1

Integrated Navigation Status bits	Attitude Accuracy	Roll/Pitch: <0.2° rms
	Bowhead Accuracy	<0.3°rms
	Heading Accuracy	<0.3°rms
	Update Rate	100Hz
	Velocity Accuracy (RMS)	0.02m/s
IMU Indicators:	Gyroscope Measurement Range	±500°/s
	Gyroscope Bias Instability	4deg/h @1σ
	Accelerometer Measurement Range	±6g
	Accelerometer Bias Instability	0.04mg @1σ
RTK Indicators:	Positioning Accuracy (RMS)	Single Point: Horizontal 1.5m RTK: Horizontal 1cm+1ppm
	Dual Antenna Positioning Accuracy (RMS)	0.1°/1m Baseline
	Velocity Accuracy (RMS)	0.03m/s
	PPS Accuracy (RMS)	20ns
	Update Rate	20Hz
	RTK Initialization Time	<5s
Interface		1×RS422 2×GNSS Antenna Interface 1×Power Connector

1.3 Mechanical Dimensions







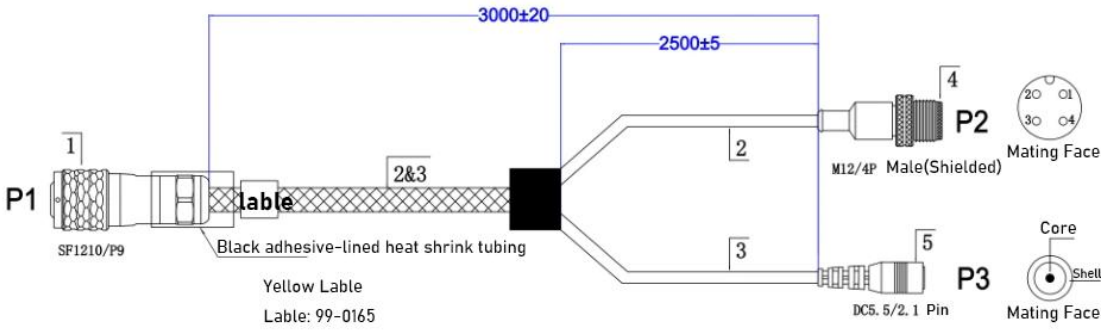
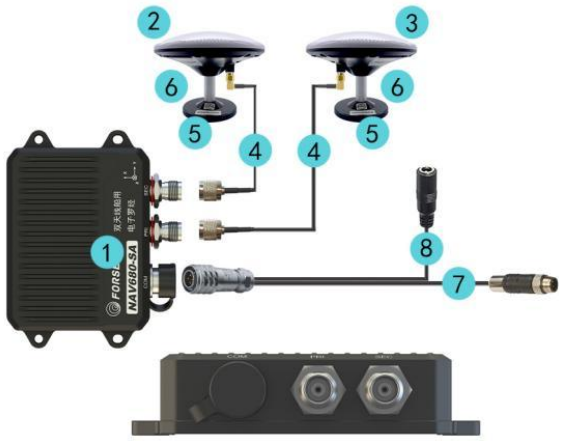


Fig. 2 Mechanical Dimensions (unit: mm)

1.4 Interface Instructions

NAV680-SA includes COM bus interface, which brings out the power cable and signal cable, primary antenna(PRI) and secondary antenna(SEC).



	Name	Quantity	Remarks
1	NAV680-SA Electronic Compass	1	
2	Primary Antenna (Positioning Antenna)	1	Anti-Salt spray
3	Secondary Antenna (Directional Antenna)	1	Anti-Salt spray
4	Antenna Feeder (20m)	1	Marine-Grade
5	Antenna suction Cup	2	
6	Antenna Pole	2	
7	Wire Harness - RS422-A Interface	1	
8	Wire Harness - Power Supply Interface	1	

1.5 Definitions for Digital Interface

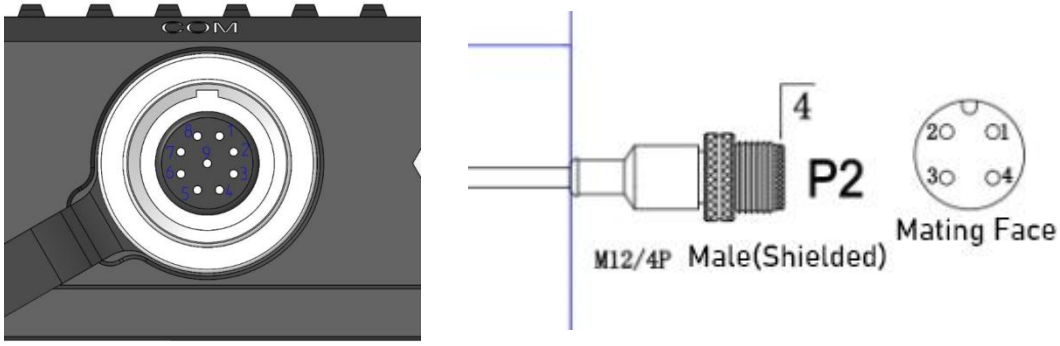


Fig. 3 Definitions for Digital Interface

PIN No.	Definitions	Ports	
1	RS422_TX+	M12 male head,Pin1	
2	VIN	DC female base	
3	RS422_TX-	M12 male head,Pin2	
4	GND	DC female base	
5	RS422_RX-	M12 male head,Pin3	
6	GND	/	
7	RS422_RX+	M12 male head,Pin4	
8	PGND	M12 male head, Cable shielding layer	The corresponding Pin are defined as follows
9	PPS	/	

1.6. Electrical Properties

Table 2

Parameter	Symbol	Minimum	Typical	Maximum	Unit
Power Input	VIN	9	12	32	V
Ground	GND				
Power Consumption	P		1.8		W
Operating Temperature	T	-40	\	85	℃
Storage Temperature	T	-40	\	95	℃
Waterproof Rating	IP67				
Humidity	95% non-condensing				
Size	116*85.9*28mm				
Weight	263g				

1.7 Antenna

NAV680-SA leads out primary and secondary antenna interfaces without connected to the antenna. When using a multimeter to test, i.e., no-load voltage is DC4.8-5.4V; when connecting the antenna to the RF port of the module, the module is able to provide an antenna feed of DC4.6V±0.2V at room temperature with an operating current of 30-100mA. When adapting active antenna to NAV680-SA, please noted the 50Ω impedance matching between antennas.

2 Examples of Use

2.1 Device Installation

1. Install requirements:

1) Dual Antenna Installation: It requires 2 on-boat antennas (primary- and secondary antenna) which connected to the compass. Please install as follows:

Satellite antennas are aligned to the sky and the carrier has no obstruction to the antennas, and the rays formed by the two antennas should be as parallel as possible to the hull of the ship.

Primary antenna is connected to PRI port, which is used for positioning and is installed on the stern of the ship.

Secondary antenna is connected to SEC port, which is used for orientation and is installed on the stern of the ship.

The distance between two antennas, i.e., the length of the baseline, needs to be greater than 0.5 m,

The hight of both antennas should be the same, the instructions are as Fig 20.. ;

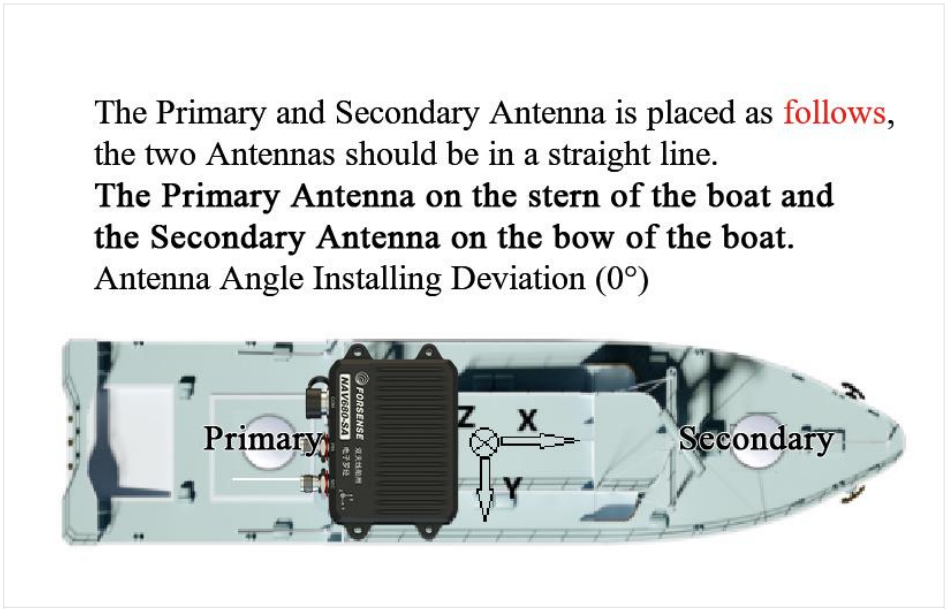
2) The installation should be away from sources of interference, such as electrical devices or systems that generate electromagnetic interference signals.

3) The GNSS antenna is screwed onto a strong magnetic plate and positioned at the center of test carrier. Place the antenna on the highest point of the vessel possible to ensure good GNSS signal reception.

4) To ensure optimum performance, the distance between GNSS primary antenna and the master should be shortened, especially the horizontal distance.

5) The module should be firmly fixed on a rigid plane rather than in a position of high vibration;

6) The module should be installed toward the bow of the boat as shown below;



Dual Antenna Installation Diagram

The correct installation method is as follows

X-axis toward the front of the vehicle

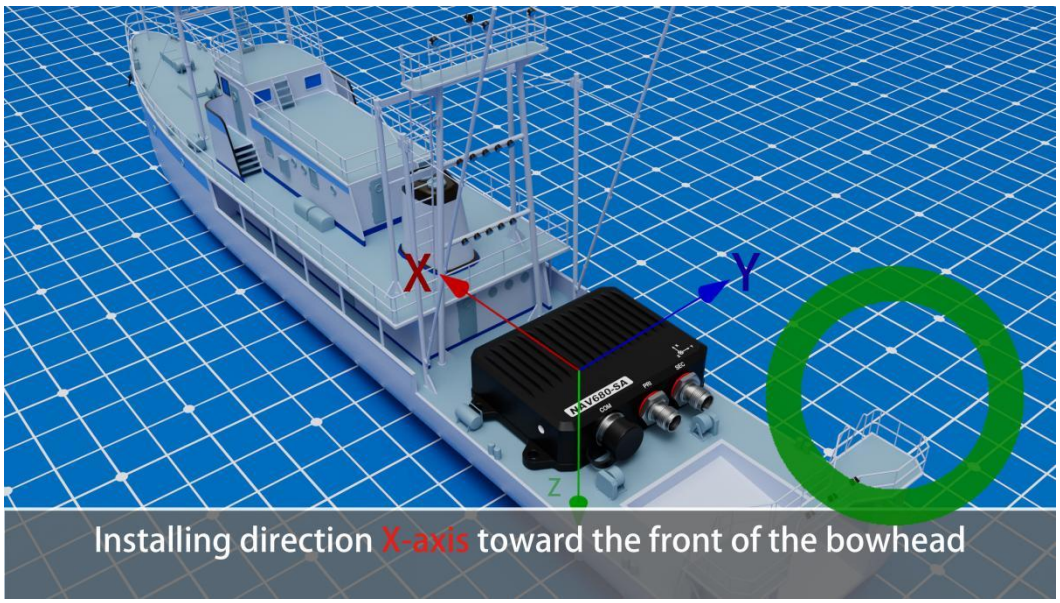
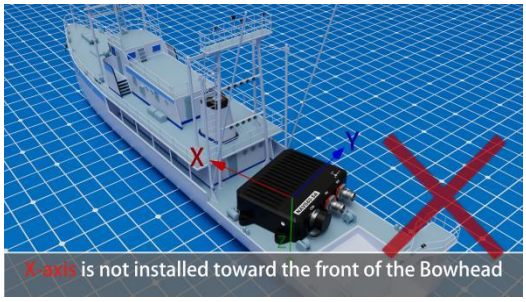


Fig. 4 Correct Installation Diagram

The following mounting methods are incorrect:



3 Definition of Coordinate System

Fig. 5 Coordinate System Diagram

This product is equipped with the Front-Right-Down (FRD) coordinate system, and the range of Euler angles is as follows:

Rotate around Z-axis: Yaw angle: 0° ~360°；

Rotate around X-axis: Roll angle: -180°~180°；

Rotate around Y-axis: Pitch angle: -90°~90°。

4 Output Protocols

4.1 Binary Protocols - INS Data Stream

Note:

- CRC checksum starts from the frame header covering all bytes of the frame except the CRC checksum bit itself, seeing the appendix for calculation methods and routines.
- The frame length is the total number of data bytes excluding frame header, frame ID, frame length and parity bits;
- In little-endian mode, and the low byte is sent first.

Parameter	Type	Relative position
Frame header 1: 0xAA	Uint8	0
Frame header 2: 0x55	Uint8	1
Frame ID: 0x0166	Uint16	2
Frame length: 0x005E	Uint16	4
Seconds in GPS week (ms)	Uint32	6
GPS week number count	Uint16	10
Latitude (deg x 10,000,000)	Int32	12
Longitude (deg x 10,000,000)	Int32	16
Height (mm)	Int32	20
Northward velocity(m/s)	Float	24
Eastward velocity(m/s)	Float	28
Ground velocity(m/s)	Float	32
Roll angle(°)	Float	36
Pitch angle(°)	Float	40
Heading angle(°)	Float	44
Dual antenna yaw angle(°)	Float	48
Track angle(°)	Float	52
Accelerometer X-axis (g)	Float	56
Accelerometer Y-axis (g)	Float	60
Accelerometer Z-axis (g)	Float	64
Gyroscope X-axis (deg/s)	Float	68
Gyroscope Y-axis (deg/s)	Float	72
Gyroscope Z-axis (deg/s)	Float	76
IMU temperature (°C)	Float	80
RTK positioning status (same as that in GGA) 0:No positioning 1:Single point positioning 2:Pseudorange differential positioning	Uint8	84

4:Fixed solution 5:Floating point solution		
Number of satellites	Uint8	85
Differential delay	Uint8	86
Dual Antenna Orientation Status 50 means oriented Other means not oriented	Uint8	87
Position accuracy factor (cm) Integrated navigation will be effective after the initialization	Uint16	88
Status bits: Bit0:1 indicates that RTK data is valid, 0 is invalid Bit1:1 indicates that the PPS signal is valid, 0 is invalid Bit2:1 means the INS is initialized, 0 is not.	Uint16	90
Reserved 1	Uint32	92
Reserved 2	Uint32	96
CRC verification	Uint32	100

4.2 NMEA protocol

- The product Supports output of combined data in NMEA format.
- Binary data stream can not output at the same time with the NMEA data stream. Before switching the data stream, please stop the current data stream output following the 7.2.2 instruction.
- Currently following statement are supported: See section 7.3 for the configuration method.

GPGGA

Chinese Description of the Message Format

Field	Structure	Descriptions	Pattern	Example
1	$GPGGA	Message Header		$GPGGA
2	utc	UTC Time (Hour/Minute/Second/Decimal Second)	hhmmss.ss	202134
3	lat	Latitude (DDmm.mm)	llll.ll	5106.9847
4	lat dir	Latitude Direction (N = North, S = South)	a	N
5	lon	Longitude (DDDmm.mm)	yyyyy.yy	11402.2986
6	lon dir	Longitude Direction (E = East, W = West)	a	W
7	quality	Solution Status	x	1
8	# sats	Number of satellites used in the solution, which may differ from the number of visible satellites.	xx	2
9	hdop	Horizontal Dilution of Precision (HDOP)	x.x	1
10	alt	Antenna Height, above or below sea level	x.x	1062.22
11	a-units	Antenna Height Unit (M = metres)	M	M
12	undulation	Height Anomaly	x.x	-16.271
13	u-units	Height Anomality Unit (M = metres)	M	M
14	age	Differential Data Age, s	xx	1
15	stn ID	Differential Base Station ID	xxxx	8
16	*xx	Verification	*hh	*48
17	[CR][LF]	Sentence Abort Character		[CR][LF]

Official Message Documentation

Field	Structure	Description	Symbol	Example
1	$GPGGA	Log header		$GPGGA
2	utc	UTC time status of position (hours/minutes/seconds/ decimal seconds)	hhmmss.ss	202134.00
3	lat	Latitude (DDmm.mm)	llll.ll	5106.9847
4	lat dir	Latitude direction (N = North, S = South)	a	N
5	lon	Longitude (DDDmm.mm)	yyyyy.yy	11402.2986
6	lon dir	Longitude direction (E = East, W = West)	a	W
7	quality	refer to Table: GPS Quality Indicators	x	1
8	# sats	Number of satellites in use. May be different to the number in view	xx	10
9	hdop	Horizontal dilution of precision	x.x	1.0
10	alt	Antenna altitude above/below mean sea level	x.x	1062.22
11	a-units	Units of antenna altitude (M = metres)	M	M
12	undulation	Undulation - the relationship between the geoid and the WGS84 ellipsoid	x.x	-16.271
13	u-units	Units of undulation (M = metres)	M	M
14	age	Age of correction data (in seconds) The maximum age reported here is limited to 99 seconds.	xx	(empty when no differential data is present)
15	stn ID	Differential base station ID	xxxx	(empty when no differential data is present)
16	*xx	Check sum	*hh	*48
17	[CR][LF]	Sentence terminator		[CR][LF]

GPS Quality Indicators

Indicator	Description
0	Fix not available or invalid
1	Single point
	Converging PPP (TerraStar-L)
2	Pseudorange differential
	Converged PPP (TerraStar-L)
	Converging PPP (TerraStar-C PRO, TerraStar-X)
4	RTK fixed ambiguity solution
5	RTK floating ambiguity solution
	Converged PPP (TerraStar-C PRO, TerraStar-X)
6	Dead reckoning mode
7	Manual input mode (fixed position)
8	Simulator mode
9	WAAS (SBAS)[1]

GPRMC

Description of the Message Format

Field	Structure	Descriptions	Pattern	Example
1	$GPRMC	Message Header		$GPRMC
2	utc	Scheduled UTC Time	hhmmss.ss	144326
3	pos status	Positioning Status (A = Valid Positioning, V = Invalid Positioning)	A	A
4	lat	Latitude (DDmm.mm)	llll.ll	5107.001774
5	lat dir	Latitude Direction (N = North, S = South)	a	N
6	lon	Longitude (DDDmm.mm)	yyyyy.yy	11402.32916
7	lon dir	Longitude Direction (E = East, W = West)	a	W
8	speed Kn	Ground Speed	x.x	0.08
9	track true	Ground Track, referenced to true north	x.x	323.3
10	date	UTC Date: (Year/Month/Day)	xxxxxx	210307
11	mag var	Magnetic Declination	x.x	0
12	var dir	Magnetic Declination Direction	a	E
13	mode ind	Positioning Mode Indicator	a	A
14	*xx	Verification	*hh	*20
15	[CR][LF]	Sentence Terminator		[CR][LF]

Official Message Documentation

Field	Structure	Description	Symbol	Example
1	$GPRMC	Log header		$GPRMC
2	utc	UTC of position	hhmmss.ss	144326.00
3	pos status	Position status (A = data valid, V = data invalid)	A	A
4	lat	Latitude (DDmm.mm)	llll.ll	5107.0017737
5	lat dir	Latitude direction: (N = North, S = South)	a	N
6	lon	Longitude (DDDmm.mm)	yyyyy.yy	11402.3291611
7	lon dir	Longitude direction: (E = East, W = West)	a	W
8	speed Kn	Speed over ground, knots	x.x	0.080
9	track true	Track made good, degrees True	x.x	323.3
10	date	Date: dd/mm/yy	xxxxxx	210307
11	mag var	Magnetic variation, degrees Note that this field is the actual magnetic variation and will always be positive. The direction of the magnetic variation is always positive.	x.x	0.0
12	var dir	Magnetic variation direction E/W Easterly variation (E) subtracts from True course. Westerly variation (W) adds to True course.	a	E
13	mode ind	Positioning system mode indicator, see Table: NMEA Positioning System Mode Indicator	a	A
14	*xx	Check sum	*hh	*20
15	[CR][LF]	Sentence terminator		[CR][LF]

NMEA Positioning System Mode Indicator

Mode	Indicator
A	Autonomous
D	Differential
E	Estimated (dead reckoning) mode
M	Manual input
N	Data not valid

GPHDT (Heading information)

Description of the Message Format

Field	Structure	Descriptions	Pattern	Example
1	$GPHDT	Message Header		$GPHDT
2	heading	Yaw angle	x.x	75.5554
3	TRUE	True North Angle	T	T
4	*xx	Verification	*hh	*36
5	[CR][LF]	Sentence Terminator		[CR][LF]

Official Message Documentation

Field	Structure	Description	Symbol	Example
1	$GPHDT	Log header		$GPHDT
2	heading	Heading in degrees	x.x	75.5554
3	True	Degrees True	T	T
4	*xx	Check sum	*hh	*36
5	[CR][LF]	Sentence terminator		[CR][LF]

GPVTG (Ground velocity information)

Description of the Message Format

Field	Structure	Descriptions	Pattern	Example
1	$GPVTG	Message Header		$GPVTG
2	track true	Bearing, referenced to true north	x.x	24.168
3	T	True North Indicator	T	T
4	track mag	Bearing, referenced to magnetic north	x.x	24.168
5	M	Magnetic North Indicator	M	M
6	speed Kn	Horizontal Velocity	x.x	0.4220347
7	N	Speed Indicator, knots	N	N
8	speed Km	Horizontal Velocity, km/s	x.x	0.781608
9	K	Speed Indicator (K = km/hr)	K	K
10	mode ind	Positioning Mode Indicator	a	A
11	*xx	Verification	*hh	*7A
12	[CR][LF]	Sentence Terminator		[CR][LF]

Official Message Documentation

Field	Structure	Description	Symbol	Example
1	$GPVTG	Log header		$GPVTG
2	track true	Track made good, degrees True	x.x	24.168
3	T	True track indicator	T	T
4	track mag	Track made good, degrees Magnetic; Track mag = Track true + (MAGVAR correction) See the **MAGVAR** command	x.x	24.168
5	M	Magnetic track indicator	M	M
6	speed Kn	Speed over ground, knots	x.x	0.4220347
7	N	Nautical speed indicator (N = Knots)	N	N
8	speed Km	Speed, kilometres/hour	x.x	0.781608
9	K	Speed indicator (K = km/hr)	K	K
10	mode ind	Positioning system mode indicator, see Table: NMEA Positioning System Mode Indicator	a	A
11	*xx	Check sum	*hh	*7A
12	[CR][LF]	Sentence terminator		[CR][LF]

NMEA Positioning System Mode Indicator

Mode	Indicator
A	Autonomous
D	Differential
E	Estimated (dead reckoning) mode
M	Manual input
N	Data not valid

GPZDA (UTC time and date)

Chinese Description of the Message Format

Field	Structure	Descriptions	Pattern	Example
1	$GPZDA	Message Header		$GPZDA
2	utc	UTC Time	hhmmss.ss	220238
3	day	±01/-31	xx	15
4	month	±01/-12	xx	7
5	year	year	xxxx	1992
6	null	Area Code, Not Available	xx	(empty when no data is present)
7	null	Local Area Minute Description, Not Available	xx	(empty when no data is present)
8	*xx	Verification	*hh	*6F
9	[CR][LF]	Sentence Terminator		[CR][LF]

Official Message Documentation

Field	Structure	Description	Symbol	Example
1	$GPZDA	Log header		$GPZDA
2	utc	UTC time status	hhmmss.ss	220238.00
3	day	Day, 01 to 31	xx	15
4	month	Month, 01 to 12	xx	07
5	year	Year	xxxx	1992
6	null	Local zone description—not available Local time zones are not supported by OEM7 family receivers. Fields 6 and 7 are always null.	xx	(empty when no data is present)
7	null	Local zone minutes description—not available	xx	(empty when no data is present)
8	*xx	Check sum	*hh	*6F
9	[CR][LF]	Sentence terminator		[CR][LF]

The format of GPATT are as follows

Name	Unit	Pattern	Example	Descriptions
Message Header		String	$GNATT	
UTC Time		hhmmss.sss	170834.000	17：08：34 UTC
Status bits		Character	1	0: Unavailable 1: Available
Roll angle	degree	3 decimal place	-4.891	Range: ±180°, defined as downward to the right. positive
Roll Indicator		Character	R	Roll Indicator
Pitch angle	degree	3 decimal place	3.122	Range: ±90°, defined as upward. positive
Pitch Indicator		Character	P	Pitch Indicator
Yaw angle	degree	3 decimal place	124.005	Range: 0° to 360°, defined as True North with clockwise being positive.
Roll Angle Uncertainty	degree	3 decimal place	0.432	0~360
Pitch Angle Uncertainty	degree	3 decimal place	0.811	0~360
Yaw Uncertainty	degree	3 decimal place	1.202	0~360
Checksum		HEX	*68	Used to check for transmission errors.

Name	*Unit*	*Format*	*Example*	*Description*
Sentence Identifier		String	$GNATT	
Time		hhmmss.sss	170834.000	17:08:34 UTC
Status		Character	1	0: invalid 1: valid
Roll Angle	degree	3 decimal places	-4.891	range ±90, right side down defined as positive
Indicator for roll		character	R	Roll indicator
Pitch Angle	degree	3 decimal places	3.122	range ±90, head up defined as positive
Indicator for Pitch		character	P	Pitch indicator
Heading Angle	degree	3 decimal places	124.005	range 0~360, to true North, counter clockwise defined as positive
Roll Angle uncertainty	degree	3 decimal places	0.432	range 0~360
Pitch Angle uncertainty	degree	3 decimal places	0.811	range 0~360
Heading Angle uncertainty	degree	3 decimal places	1.202	range 0~360
Checksum		Hex	*68	Used by program to check for transmission errors

5 Parameters Setting

5.1 Use CAN Interface to Configure Parameters

5.1.1 Configure GNSS Dual Antenna Mounting Angle

If the gnss dual antenna mounting angle is configured to be 0°, the configuration command is:

Command: AT+RTK_ANGLE=0\r\n

Response: ANGLE=0\r\n

Explanation: Installation angle is the angle between the ray of the primary antenna pointing to the secondary antenna and the direction of the front end of the vehicle. Clockwise motion is positive, while counterclockwise motion is considered negative. The angle input range is -180° to 180°.

Note: After saving the configuration commands, power off and restart are necessary steps. The distance between the two antennas should be more than 50cm.

If the INS is configured to output projection points, the configuration command is:

Command: AT+PROJ_VECTOR=1.0,2.0,3.0\r\n

Response: PROJ_VECTOR_X=1.0, PROJ_VECTOR_Y=2.0, PROJ_VECTOR_Z=3.0/r/n

Explanation: The default output of the integrated navigation is the antenna phase center coordinates, if other position coordinates is needed, configure the IMU phase center relative to the projected point position of the rod arm vector. The configuration method is the same as 8.1.1 Rod Arm Configuration.

5.1.2 Configure the Output of Data Stream in NMEA Format

If the NEMA statement is configured for output, the 7.2 INS data stream will not be output.

If wanted to switch to 7.2 INS data stream output, please stop the current data stream output by pressing the 7.2.11 command first.

The configuration commands are as follows

GPGGA

Example: Output GPGGA at 5Hz Statement: AT+GPGGA=5\r\n

Response: OK\r\n

GPRMC

Example: Output GPRMC at 1Hz Statement: AT+GPRMC=1\r\n

Response: OK\r\n

GPHDT (Heading information)

Example: Output GPHDT at 1Hz Statement: AT+GPHDT= 1\r\n

Response: OK\r\n

GPVTG (Ground velocity information)

Example: Output GPVTG at 1Hz Statement: AT+GPVTG= 1\r\n

Response: OK\r\n

GPZDA (UTC time and date)

Example: Output GPZDA at 1Hz Statement: AT+GPZDA= 1\r\n

Response: OK\r\n

GPATT (Customized message)

Example: Output GPATT at 1Hz Statement: AT+GPATT= 1\r\n

Response: OK\r\n

If the data output stops

Command: AT+SETNO\r\n

Response: OK\r\n

5.1.3 Stop the Output of the Current Data Stream

Command: AT+SETNO\r\n

Response: OK\r\n

5.1.4 Configure the Baud Rate

The default baud rate is 115200

Example: If configuring the IMU serial baud rate to be 230400, the command is:

Command: AT+BAUD=230400\r\n

Response: BAUD=230400\r\n

Note: It is necessary to power off and restart after configuring and saving the commands.

5.1.5 Output all Configuration Information

If you want to query all the configured information, command:

AT+CONFIG\r\n

5.1.6 Querying Version Numbers

Command: AT+VERSION\r\n

5.1.7 Configure the Mounting Rotation Angle

When the compass is not mounted as the default direction, it is possible to configure rotation

angle.

Currently only the following rotation angles are supported

x-axis rotation 180°

z-axis rotation 90° 180° 270°

The configuration commands are as follows

If the mounting rotation angle is 180° on the X-axis, the configuration command is:

Command: AT+INSTALL_ANGLE=180,0,0\r\n

Response: INST_ANGLE_X=180.000,INST_ANGLE_Y=0.000,INST_ANGLE_Z=0.000

If the mounting rotation angle is 180° on the Z-axis, the configuration command is:

Command: AT+INSTALL_ANGLE=0,0,180\r\n

Response: INST_ANGLE_X=0.000,INST_ANGLE_Y=0.000,INST_ANGLE_Z=180.000

5.1.8 Save Parameters

The above parameters include lever arms, data stream, output frequency, etc. After configuration, you need to enter the AT+SAVE command and restart the software.

Command: AT+SAVE\r\n

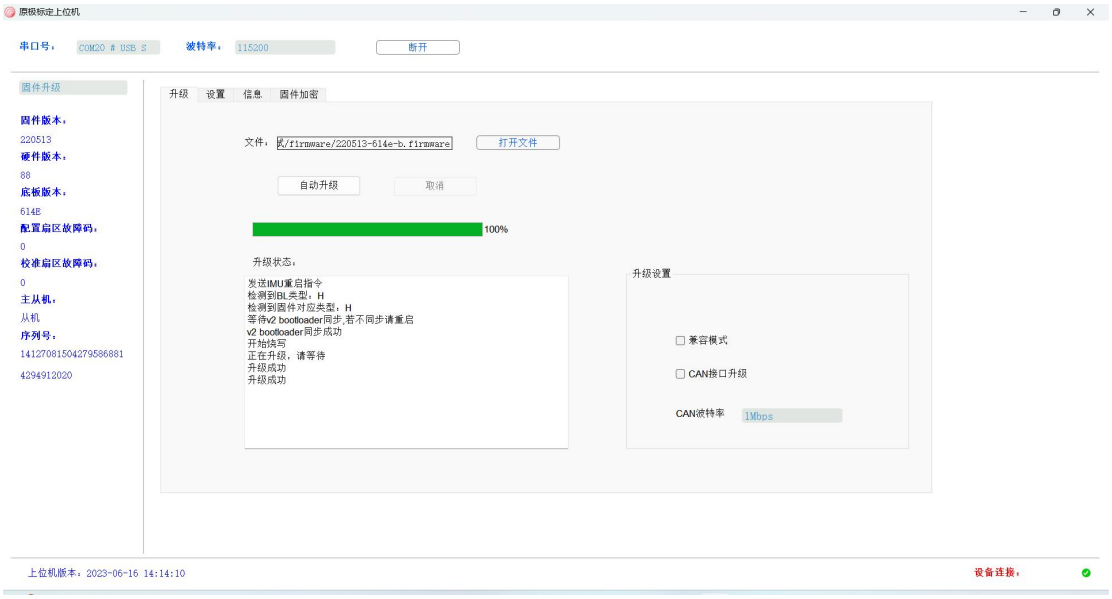
Response: OK\r\n

6 Firmware Upgrading

6.1 Firmware Upgrading via the PC Software

6.1.1 RS232 Serial Port Direct Connection

Use the Forsense testing PC software - Select firmware upgrade - Open the firmware - Click Auto Upgrade.



6.2 Serial OTA

The upgrade process is divided into the following steps:

Step 1: Send the upgrade command

Send upgrade command to the IMU module. After that, the device will record the upgrade flag bit in a certain area of flash and then perform a soft reboot. At this time, the module will enter the BOOTLOADER.

In BOOTLOADER.

```
上位机发送升级指令如下：
  cmd_bl[34] ={0x55,0xaa,0xbb,0x88,0x18,0x00,0x00,0x00,0xc8,0x42,0x00,0x00,
  0x48,0x43,0x2c,0x01,0x00,0x00,0x90,0x01,0x00,0x00,0xf4,0x01,0x00,0x00,
  0x58,0x02,0x00,0x00,0x40,0x97,0x46,0x6a};
  此时 imu 模块将会重启后进入 bootloader 中
```

Step 2: Send the **HC32MCU_FORSENSE** string

After the IMU module enters the bootloader, it will actively send the HC32_UPLOADER

string. During this period, the PC software needs to send the HC32MCU_FORSENSE string. If the module receives the string, it will not jump to APP, but stay in a pending upgrade state in the bootloader, and will also stop actively sending any messages.

```
//发送完升级指令后，紧接着发送如下字符串，确保 imu 模块不会跳转到 APP 区域，同时处于待升级状态
```

```
QString str = "HC32MCU_FORSENSE" ;
for(int i=0;i<10;i++)
{
  _port_device->write(str.toLatin1()); sleep_ms(50);
}
```

Step 3: Send the synchronization command

When the IMU module stays in a pending upgrade state in the bootloader, it will not send messages actively, but respond to the commands of the PC software passively. At this time, the PC software enters Send_CMD_LONG(0X21,0,0,0,0,0,0,0,0) and waits for the response from the IMU module. After receiving the command, IMU immediately replies with Send CMD ACK (0x64,0x10,0).

Data Bit Once the PC software determines that it has received the response data, it indicates that the synchronization is completed.

```
上位机与 imu 模块同步过程：
1.上位机发送同步命令，命令码为 0x21，发送后等待 imu 模块响应
 Send_CMD_LONG(0x21,0,0,0,0,0,0,0);
2.imu 模块收到 0x21 的命令码后，发送响应数据，数据命令如下：
Send_CMD_ACK(0x64,0x10,0);
3.上位机判断是否收到了 imu 的响应数据，如收到则表示完成了同步
注：Send_CMD_LONG 函数与 Send_CMD_ACK 函数见文章末尾
```

Step 4: Send erase command

The PC software sends the erase command Send_CMD_LONG(0x23,0,0,0,0,0,0,0), and then the IMU module will erase everything in the APP area and send the final execution result back to the PC software. The PC software will decide whether to send erase command again according to the result. Once the erase operation is successful, the APP of IMU module cannot be restored. In addition, the PC software must wait for the response of successful erase before proceeding to the next step, otherwise it may cause the subsequent upgrade failure.

```
1.上位机发送擦除命令，命令码为 0x23，发送完此命令后耐心等待 imu 回复执行结果
  Send_CMD_LONG(0x23,0,0,0,0,0,0,0);
2.imu 收到 0x23 的命令码后，将会把 APP 区域数据擦除干净，并将最终的执行结果回复给上位机
   send_CMD_ACK(0x64,0x10,0); //表示擦除成功
```

```
   send_CMD_ACK(0x64,0x11,0); //表示擦除失败，需要重新擦除
   注意：上位机必须收到擦除成功的响应，才能进行下一步
```

Step 5: Send upgrade packets

After a successful erase operation, it will enter the most important part of sending firmware data by using the function Send_Upload_Data. The PC software will divide firmware data into packets, and each packet has a fixed size of 64 bytes. If the last packet contains fewer than 64 bytes, it will be sent with the actual number of bytes. Each data frame contains the valid data length and the offset address of this packet within the entire firmware. Every time the PC software sends a single data frame, it must wait for the response from the IMU module and ensure that IMU has successfully obtained this frame data before sending the next. After the IMU module successfully receives the data packet from the PC software, it will send the response data and write it to the specified flash address according to the offset address.

If the writing process fails, it will send failure command; otherwise it will send nothing.

```
例：将大小 1000 字节的升级文件 uint8_t Upgrade_Data[1000]发送给 imu 模块
 //发送第一包：
  1.1 通过函数将 0~63 字节发送到 imu 模块
  Send_Upload_Data（0x27,0,0,0x40,Upgrade_Data);
  //上面函数第一参数 0x27 为固定值，第二参数 0 为固定值，第三参数 0 为偏移地址，
    第四参数 0x40 为有效字节长度，第五参数为发送数据的首地址

 1.2 imu 成功收到数据后，会发送响应数据，并根据偏移地址写入指定的 flash 地址
   send_CMD_ACK(0x753D,0x00,0);
   //上面函数第一参数 0x753D 为固定值，第二参数 0 为固定值，第三参数为偏移地址。
   imu 模块将数据写入 flash 过程中如果写入失败将发送失败命令 send_CMD_ACK(0x64,0x11,0);

   //发送第二包：
   2.1 上位机收到 imu 响应数据后，将发送第二包数据 Send_Upload_Data（0x27,0,0x40,0x40,Upgrade_Data+0x40);
   2.2 imu 成功收取到第二包数据后，发送响应： send_CMD_ACK(0x753D,0x00,0x40);
   2.3 imu 模块将数据写入 flash 过程中如果写入失败将发送失败命令 send_CMD_ACK(0x64,0x11,0);


   //发送第三包
   3.1 上位机收到 imu 响应数据后，将发送第三包数据 Send_Upload_Data（0x27,0,0x80,0x40,Upgrade_Data+0x80);
   3.2 imu 成功收取到第三包数据后，发送响应： send_CMD_ACK(0x753D,0x00,0x80);
   3.3 imu 模块将数据写入 flash 过程中如果写入失败将发送失败命令 send_CMD_ACK(0x64,0x11,0);
  ------------
  //发送第十五包：
```

```
15.1 上位机收到 imu 响应数据后，将发送第十五包数据 Send_Upload_Data（0x27,0,0x380,0x40,Upgrade_Data+0x380);
15.2 imu 成功收取到第十五包数据后，发送响应：  send_CMD_ACK(0x753D,0x00,0x380);
15.3 imu 模块将数据写入 flash 过程中如果写入失败将发送失败命令 send_CMD_ACK(0x64,0x11,0);
//发送第十六包：
15.1 上位机收到 imu 响应数据后，将发送第十六包数据 Send_Upload_Data（0x27,0,0x3C0,0x28,Upgrade_Data+0x3C0);
15.2 imu 成功收取到第十六包数据后，发送响应：  send_CMD_ACK(0x753D,0x00,0x3C0);
15.3 imu 模块将数据写入 flash 过程中如果写入失败将发送失败命令 send_CMD_ACK(0x64,0x11,0);
发送结束
```

Step 6: Acquire **CRC** check digits

Usually the command rule for upgrading firmware is imu614e-b#CRC1373387121.firmware, and the CRC string is immediately followed by the calculated CRC checksum value. After the upgrade file is sent, the PC software needs to send a check command to determine if the upgrade file received by the IMU module is correct. After the PC software enters Send_CMD_LONG(0x29,0,0,0,0,0,0,0,0) command, it will get the CRC check digits calculated by the IMU module itself.

If the PC software judges that there is an error in the CRC check digits, it will restart the upgrade process by resuming the 4th step of sending erase command.

```
上位机发送获取 crc 校验码指令，等待 imu 响应
 Send_CMD_LONG(0x29,0,0,0,0,0,0,0,0);
imu 模块响应发送 crc 校验值数据：
send_CMD_ACK(0x753C,0x10,crc32_data);
 其中 crc32_data 值为 imu 模块本身计算的 crc32 数据
```

Step 7: Send reboot command

After the PC software judges that the CRC check digits are correct, it will send reboot command, indicating successful upgrading.

```
判断 crc 校验值正确后，发送重启命令：
 Send_CMD_LONG(0x30,0,0,0,0,0,0,0,0);
```

When firmware upgrade is complete, powering off and rebooting to determine if the upgrade is successful by reading the version number.

Function definitions:

1. The definition of Send_CMD_LONG is as follows:

```
struct  MULTI_LONG_CMD_STRUCT
{
uint8_t header1;
uint8_t header2;
uint16_t id;
uint16_t length;
float  param1;
float  param2;
uint32_t param3;
uint32_t param4;
int32_t param5;
int32_t param6;
uint32_t check_crc;
}__attribute__((packed));
```

```
void  :Send_CMD_LONG(uint16_t cmd_id,float cm1,float cm2,uint32_t cm3,uint32_t cm4,int32_t cm5,int32_t cm6)
{
    uint8_t check_sum=0;
    struct MULTI_LONG_CMD_STRUCT data_cmd_long __attribute__((packed));
    data_cmd_long.header1=0x55;
    data_cmd_long.header2=0xAA;
    data_cmd_long.id=cmd_id;
    data_cmd_long.length=sizeof(data_cmd_long)-10;
    data_cmd_long.param1=cm1;
    data_cmd_long.param2=cm2;
    data_cmd_long.param3=cm3;
    data_cmd_long.param4=cm4;
    data_cmd_long.param5=cm5;
    data_cmd_long.param6=cm6;
    int len=sizeof(data_cmd_long)-4;
    uint32_t check_crc=1;
    data_cmd_long.check_crc=crc_crc32(check_crc,(uint8_t *)(&data_cmd_long), len);
    send((uint8_t *)(&data_cmd_long),sizeof(data_cmd_long));
}
```

2. The definition of Send_CMD_ACK is as follows:

```
struct  CMD_ACK_STRUCT
{
uint8_t header1;
uint8_t header2;
uint16_t id;
```

```
uint16_t length;
uint32_t command;/*<  Command ID (of acknowledged command).*/
uint32_t result;/*<  Result of command.*/
uint32_t check_crc;
}__attribute__((packed));

void Send_CMD_ACK(uint16_t cmd_id,uint16_t ack_id,uint32_t result)
{
    uint32_t check_crc=0;
    struct CMD_ACK_STRUCT data_cmd_ack __attribute__((packed));
    data_cmd_ack.header1=0xAA;
    data_cmd_ack.header2=0x55;
    data_cmd_ack.id=cmd_id;
    data_cmd_ack.length=sizeof(data_cmd_ack)-10;
    data_cmd_ack.command=ack_id;
    data_cmd_ack.result=result;
    int len=sizeof(data_cmd_ack)-4;
    check_crc=1;
    data_cmd_ack.check_crc=crc_crc32(check_crc,(uint8_t *)(&data_cmd_ack), len);
    Cout((uint8_t *)(&data_cmd_ack),sizeof(data_cmd_ack));
}
```

3. The definition of Send_Upload_Data is as follows:

```
struct  UPLOAD_DATA
{
uint8_t header1;
uint8_t header2;
uint16_t id;
uint16_t length;
uint8_t param[64];
uint32_t offset;
uint16_t size;
uint8_t cmd;
uint32_t check_crc;
}__attribute__((packed));

struct UPLOAD_DATA upload_data;
void Send_Upload_Data(uint8_t cmd_id,uint8_t cmd,uint32_t offset,uint16_t size,uint8_t*  param)
{
    upload_data.header1=0x55;
```

```
    upload_data.header2=0xAA;
    upload_data.id=cmd_id;
    upload_data.length=sizeof(UPLOAD_DATA)-10;
    upload_data.cmd=cmd;
    for(int i=0;i<size;i++)
    upload_data.param[i] = *(param+i);
    upload_data.offset=offset;
    upload_data.size=size;
    int len=sizeof(UPLOAD_DATA)-4;
    uint32_t check_crc=1;
    upload_data.check_crc=crc_crc32(check_crc,(uint8_t *)(&upload_data), len);
    send((uint8_t *)(&upload_data),sizeof(UPLOAD_DATA));
}
```

4. The CRC32 checksum function is as follows:

```
static const uint32_t crc32_tab[] = {
    0x00000000, 0x77073096, 0xee0e612c, 0x990951ba, 0x076dc419, 0x706af48f,
    0xe963a535, 0x9e6495a3, 0x0edb8832, 0x79dcb8a4, 0xe0d5e91e, 0x97d2d988,
    0x09b64c2b, 0x7eb17cbd, 0xe7b82d07, 0x90bf1d91, 0x1db71064, 0x6ab020f2,
    0xf3b97148, 0x84be41de, 0x1adad47d, 0x6ddde4eb, 0xf4d4b551, 0x83d385c7,
    0x136c9856, 0x646ba8c0, 0xfd62f97a, 0x8a65c9ec, 0x14015c4f, 0x63066cd9,
    0xfa0f3d63, 0x8d080df5, 0x3b6e20c8, 0x4c69105e, 0xd56041e4, 0xa2677172,
    0x3c03e4d1, 0x4b04d447, 0xd20d85fd, 0xa50ab56b, 0x35b5a8fa, 0x42b2986c,
    0xdbbbc9d6, 0xacbcf940, 0x32d86ce3, 0x45df5c75, 0xdcd60dcf, 0xabd13d59,
    0x26d930ac, 0x51de003a, 0xc8d75180, 0xbfd06116, 0x21b4f4b5, 0x56b3c423,
    0xcfba9599, 0xb8bda50f, 0x2802b89e, 0x5f058808, 0xc60cd9b2, 0xb10be924,
    0x2f6f7c87, 0x58684c11, 0xc1611dab, 0xb6662d3d, 0x76dc4190, 0x01db7106,
    0x98d220bc, 0xefd5102a, 0x71b18589, 0x06b6b51f, 0x9fbfe4a5, 0xe8b8d433,
    0x7807c9a2, 0x0f00f934, 0x9609a88e, 0xe10e9818, 0x7f6a0dbb, 0x086d3d2d,
    0x91646c97, 0xe6635c01, 0x6b6b51f4, 0x1c6c6162, 0x856530d8, 0xf262004e,
    0x6c0695ed, 0x1b01a57b, 0x8208f4c1, 0xf50fc457, 0x65b0d9c6, 0x12b7e950,
    0x8bbeb8ea, 0xfcb9887c, 0x62dd1ddf, 0x15da2d49, 0x8cd37cf3, 0xfbd44c65,
    0x4db26158, 0x3ab551ce, 0xa3bc0074, 0xd4bb30e2, 0x4adfa541, 0x3dd895d7,
    0xa4d1c46d, 0xd3d6f4fb, 0x4369e96a, 0x346ed9fc, 0xad678846, 0xda60b8d0,
    0x44042d73, 0x33031de5, 0xaa0a4c5f, 0xdd0d7cc9, 0x5005713c, 0x270241aa,
    0xbe0b1010, 0xc90c2086, 0x5768b525, 0x206f85b3, 0xb966d409, 0xce61e49f,
    0x5edef90e, 0x29d9c998, 0xb0d09822, 0xc7d7a8b4, 0x59b33d17, 0x2eb40d81,
    0xb7bd5c3b, 0xc0ba6cad, 0xedb88320, 0x9abfb3b6, 0x03b6e20c, 0x74b1d29a,
    0xead54739, 0x9dd277af, 0x04db2615, 0x73dc1683, 0xe3630b12, 0x94643b84,
    0x0d6d6a3e, 0x7a6a5aa8, 0xe40ecf0b, 0x9309ff9d, 0x0a00ae27, 0x7d079eb1,
    0xf00f9344, 0x8708a3d2, 0x1e01f268, 0x6906c2fe, 0xf762575d, 0x806567cb,
    0x196c3671, 0x6e6b06e7, 0xfed41b76, 0x89d32be0, 0x10da7a5a, 0x67dd4acc,
```

7 Accessories

Standard accessories:

NAV680-SA host

Harness

Primary antenna (for positioning)

Secondary antenna (for orientation)

Primary antenna connector

Secondary antenna connector

8 Revision History

Version	Date	Status/Notes
Version 1.0	01/09/2024	First release