FORSENSE
原极科技

IMU61X Interface Protocol and Instructions

Catalogue

Legend

Table Example

1. Serial Communication Protocol

QT-based serial port protocol example:
http://www.forsense.cn/cn/h-col-128.html
Serial communication has two modes: Stream Mode and Command Mode. After the initialization of power-on, IMU enters the corresponding mode according to the mode value configured by the parameters.
Data flow mode: periodically output AHRS data at a fixed frequency;
Command mode: In this mode, stop periodic output, users can communicate with IMU by sending commands, get sensor data, status, parameters, etc. through GET commands, and configure parameters of IMU.

1.1 Serial Interface Parameters

Table 1 Serial Interface Parameters

Transmission Rate Range	**115200bps ~ 1.5Mbps**
Default Transfer Rate	115200bps
Beginning Bit	1 bit
Data Bits	8 bits
Stop Bit	1 bit
Parity Check	-

1.2 Packet Format

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

Table 2 IMU Output and User Input Data Structures

Offset	Data Type	Name	Describe
0	uint8	Header 1	IMU Output Header:0xAA，0x55 User Input Header:0x55，0xAA
1	uint8	Header 2	
2	uint16	ID Low	Low-bit bytes of serial communication frame ID
3		ID High	High-bit bytes of serial communication frame ID
4	uint16	Low Data Length Bit	Low-bit bytes of serial communication frame length, length is the number of bytes payload occupies, that is, n
5		High data length bits	High-bit bytes of serial communication frame length, Length is the number of bytes payload occupies, that is, n
6	uint8	Payload （N bytes）	Data Load
6+n	uint32	CRC_CEHCK（32-bit data low bytes）	CRC Check
7+n		CRC_CEHCK（Low bytes in 32-bit data）	
8+n		CRC_CEHCK（High bytes in 32-bit data）	
9+n		CRC_CEHCK（32-bit data high bytes）	

Note 1 Data is transmitted in a small-end format, with low bytes first and high bytes last
Note 2 The initial value of CRC32 is 1. The CRC calculation does not include all the data of this frame. See Chapter 7 for table lookup calculation

1.3 Stream Frame - AHRS Data

Table 3 Serial AHRS Data Format

Type	Frame Header	Frame Header	ID	Length	Payload	End of Frame
Data Type	uint8	uint8	uint16	uint16	**A1**	uint32
Code	0xAA	0x55	0x0002	0x002C		crc32

Note 1 Maximum Output Update Rate is not greater than200Hz@115200bps

Table 4 Serial A1 Load Data Format

Offset	Name	Data Type	Units	Describe
0	timer	uint32	μs	Time Scale
4	pitch	float	°	Pitch angle
8	roll	float	°	Roll angle
12	yaw	float	°	Heading angle
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-axis Angular Speed
32	gy	float	°/s	Y-axis Angular Speed
36	gz	float	°/s	Z-axis Angular Speed
40	temp	float	℃	IMU chip temperature

1.4 Command Mode GET Output - System Status

Table 5 Serial System Status Data Format

Type	Frame Header	Frame Header	ID	Length	Payload	End of Frame
Data Type	uint8	uint8	uint16	uint16	**S1**	uint32
Code	0xAA	0x55	0x00FF	0x0042		crc32

Note: The length of this frame varies depending on the IMU model. All frames represent the length of S1

Table 6 Serial S1 Load Data Format

Offset	Name	Data Type	Describe
0	Software_version	uint32	Software Version Number
4	Hardware_version	uint32	Hardware Version Number
8	rev	uint16	Reserved bytes
10	sn0	uint32	First SN Number
14	sn1	uint32	Second SN Number
18	sn2	uint32	Third SN Number
22	Board_version	uint32	Backplane Version number
30	rev[n]	uint32	Reserved bytes

1.5 Command Mode GET Output - Read Parameters

Table 7 Serial Port Parameter Output Data Format

Type	Frame Header	Frame Header	ID	Length	Payload	End of Frame
Data type	uint8	uint8	uint16	uint16	**P1**	uint32
Code	0xAA	0x55	0x0006	0x0018		crc32

Table8 Serial P1 Load Data Format

Offset	Name	Data Type	Describe
0	Param1	float	Set parameters
4	Param2	float	Reserved, default 0
8	Param3	uint32	Set Parameter Index
12	Param4	uint32	Reserved, default 0
16	Param5	Int32	Reserved, default 0
20	Param6	Int32	Reserved, default 0

Table 9 Serial P1 Load Parameter Index Table

Param3	Param1	Units
3	Serial port output baud rate, supporting the following baud rates: 115200 230400 460800 921600 1500000	bps
8	X-Axis Gyro Zero Bias Calibration Results，GYRO_X_OFF	°/s
9	Y-Axis Gyro Zero Bias Calibration Results，GYRO_Y_OFF	°/s
10	Z-Axis Gyro Zero Bias Calibration Results，GYRO_Z_OFF	°/s
16	X-axis mounting error angle，INSTALL_X_OFF	°
17	Y-axis mounting error angle，INSTALL_Y_OFF	°
18	Z-axis mounting error angle，INSTALL_Z_OFF	°
21	AHRS output frequency, default 100Hz	Hz
27	SPI or I2C mode 0：SPI or I2C is not supported, only serial port is supported 1：Support Serial Port and SPI 2：Support Serial Port and I2C 3：Select SPI mode through external IO port setting mode, suspend or lower pin, and I2C mode for higher pin	
30	I2C address, default 0x18	
31	Internal filter configuration，define FILTER with SPICTRL control table	

1.6 Command Mode SET Instructions

Table 10 Serial Input Command Format

	Frame Header	Frame Header	ID	Length	Payload	End of Frame
Data Type	uint8	uint8	uint16	uint16	**R1**	uint32
Code	0x55	0xAA	**CMD**	0x0018		crc32

Note 1 The relationship between CMD and R1, as detailed in the R1 load parameter index table

Table 11 Serial R1 Load Data Format

Offset	Name	Data Type	Describe
0	Param1	float	Set parameters
4	Param2	float	Reserved, default 0
8	Param3	uint32	Set Parameter Index
12	Param4	uint32	Reserved, default 0
16	Param5	Int32	Reserved, default 0
20	Param6	Int32	Reserved, default 0

Table 12 Serial R1 Load Parameter Index Table

CMD	Param1	Param3	Describe
1	0	0	Trigger to get system state data once
2	0	0	Trigger to get AHRS data once
3	<mode>	0	Set output mode: Mode=1, data stream output AHRS Mode=100, data flow mode disabled, enter COMMAD mode
4	<heading>	0	Reset the heading angle of AHRS: Heading input ranges from 0 to 360 in degrees
5	0	0	Save current parameters to FLASH
6	0	<value>	Read parameters, value is the parameter index to read, and that is, P1.index, as detailed in Serial Responsive Output-Parameter Read For example, to read the AHRS output frequency (ODR), set value=21
7	0	0	Start installation error calibration
8	0	0	Trigger to get a calibration status for installation errors
9	0	0	Execute Software Restart
10	0	0	Perform Zero Bias Calibration for Static Gyroscope
14	<value>	3	Set Serial Output Baud Rate, Unit: BPS Value is valid for: 115200，230400，460800，921600，1500000 When value is another value, 115200bps is used by default
14	<value>	21	Set periodic AHRS data output frequency in Hz Value is valid for: 1，10，50，100，200，400 When value is another value, the default is 100Hz
14	<value>	27	Set SPI or I2C mode value=0，SPI or I2C is not supported, only serial port is supported value=1，Support Serial Port and SPI value=2，Support Serial Port and I2C Value=3，Represents configuring SPI or I2C mode based on external IO port Default value=1, which supports serial port and SPI
14	<value>	30	Set I2C address, value defaults to 0x18, other valid values are: 0x18，0x19，0x1A，0x1B
14	<value>	31	Internal filter configuration, defines the same SPI accelerometer and gyro filter configuration, default 0xBB, or 47Hz
15	0	0	Restore factory setup parameters

Note1 Note that the values in this table are all decimal

Note2 You can use the PC Command Generator function to generate corresponding commands for sending, as described in the PC usage section of this manual.

If the AHRS output is enabled:
Enter CMD ID 3 and parameter 1 1, and the resulting hexadecimal array can be filled into the serial port helper or program array and sent to the IMU.

1.7 Command Mode Output - User Command Response

Table 13 Serial User Command Response Data Format

Type	Frame Header	Frame Header	ID	Length	Payload	End of Frame
Data Type	uint8	uint8	uint16	uint16	**K1**	uint32
Code	0xAA	0x55	0x0064	0x0007		crc32

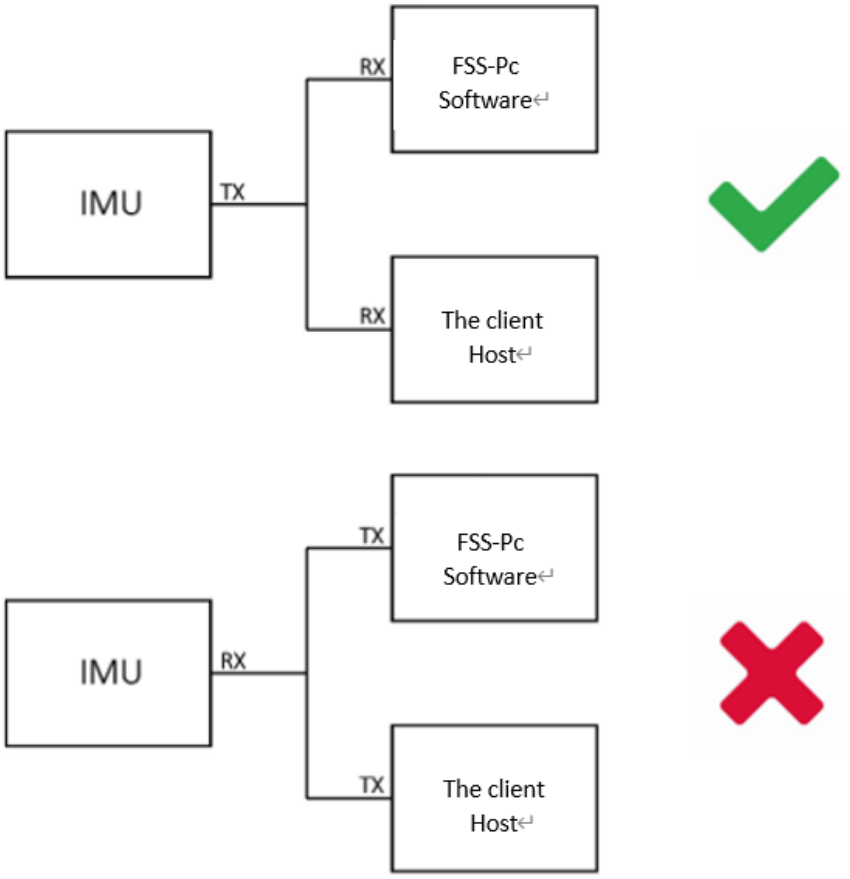
Table 14 Serial K1 Load Data Format

Offset	Name	Data Type	Describe
0	command	uint16	Command ID to be responded to
2	result	uint16	Result

1.8 Pay attention to the serial port connection mode

Note:
The RX of the serial port cannot be connected to two TX at the same time. Therefore, if you need to connect the host of the original port, you need to disconnect the serial port communication between the host and the user host. Otherwise, the host can only receive data and cannot send commands to the IMU.
As shown below:

Figure 1 Serial port connection mode



Note1: IMU TX can be connected to multi-channel RX, while RX cannot be connected to multi-channel TX.

Note2: THE IMU serial port cannot be connected to the client host and Forsense Pc Software at the same time.

Note3: The IMU can reserve another serial port for connecting to t Forsense Pc Software

2. SPI Communication Protocol

Sample SPI host read driver based on STM32:

http://www.forsense.cn/cn/h-col-128.html

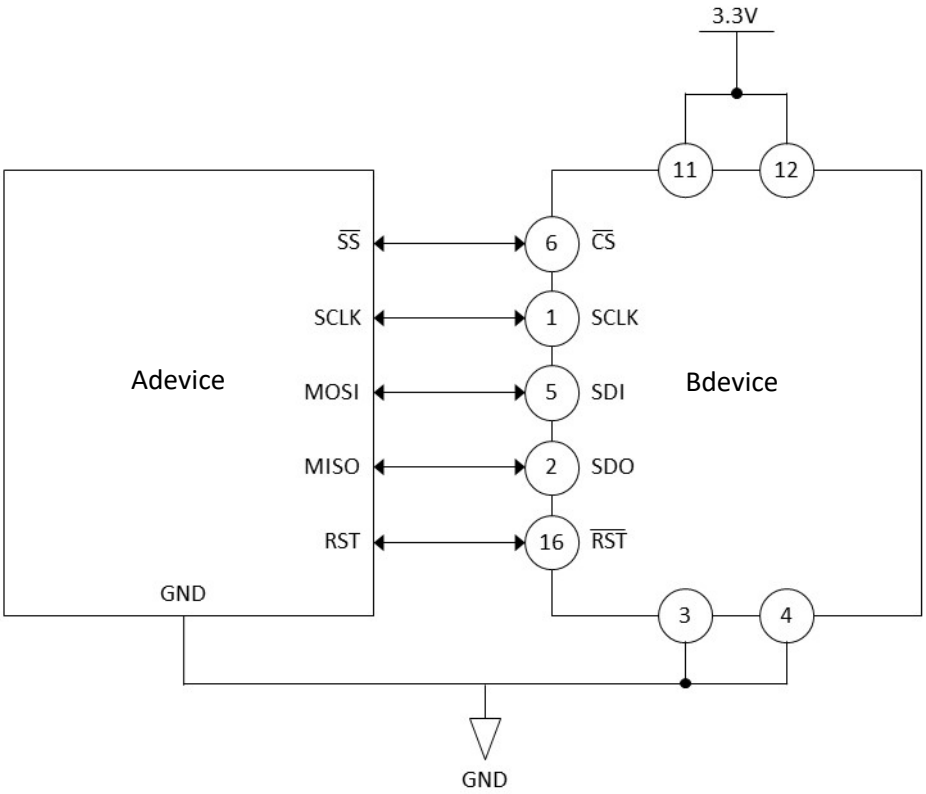
2.1 SPI interface Parameters

Table 15 SPI Interface Parameters

SPI HOST	**This product acts as a slave**
SPI Rate	0.2 ~ 2MHz
SPI WORD LENGTH	16 bit
PHASE	Rising edge triggering
POLARITY	Idle at low level
ORDER	MSB Priority

2.2 SPI Connection Diagram

Figure 2 SPI Connection Diagram

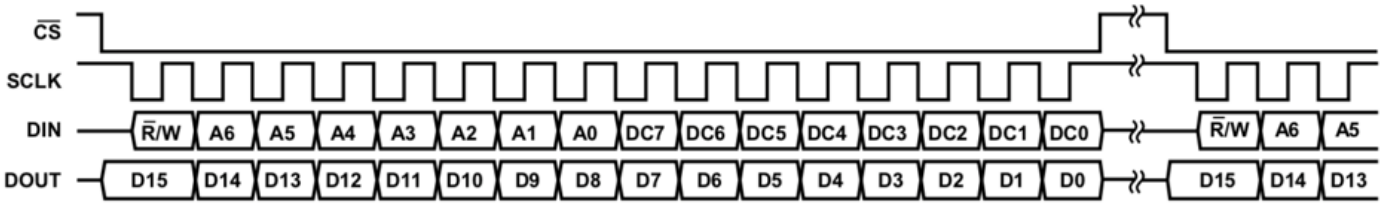


Note1: Before initializing the read, reset the IMU and wait for 3S for it to work properly.
Note2: Refer to corresponding manuals for SPI pins of different IMU models.

2.3 SPI Communication Bit Order

The SPI interface supports full duplex serial communication (both sending and receiving) in the bit order shown in the following figure.

Figure 3 SPI communication bit sequence diagram



Among them, the highest bit of DIN represents read/write operations, [A6:A0] represents register addresses, [DC7:DC0] represents written data (write operations) or DUMMY data (read operations). At that time, the DOUT data for this SPI cycle was meaningless. At that time, the DOUT data for this SPI cycle represented register output data for the last two cycles, as shown in the BURST reading example.。

2.4 SPI Register

Table 16 SPI Register List

Name	address	Read/Write	Default value	Window ID	Describe
BURST	0x00	RW		0	Continuous Read
BURST_CTRL	0x0D,0x0C	RW	0x0000	1	Continuous Read Configuration
FILTER_CTRL	0x07,0x06	RW	0x00BB	1	Filter Selection
PROD_ID1	0x6A	R	0x0000	1	ID No. 1
PROD_ID2	0x6C	R	0x494d	1	ID No. 2
PROD_ID3	0x6E	R	0x5536	1	ID No. 3
PROD_ID4	0x70	R	0x3132	1	ID No. 4(IMU612)
			0x3134	1	ID No. 4(IMU614)
			0x3138	1	ID No. 4(IMU618)
			0x3141	1	ID No. 4(IMU6132A)
			0x3142	1	ID No. 4(IMU6132B)
WIN_CTRL	0x7F,0x7E	RW	0x0000	0，1	Window ID Selection
TEMP_HIGH	0x0E	R	\	0	Temperature High Byte
TEMP_LOW	0x10	R	\	0	Temperature Low Byte
XGYRO_HIGH	0x12	R	\	0	X Axis Height of Gyro in Bytes
XGYRO_LOW	0x14	R	\	0	X Axis Low of Gyro in Bytes
YGYRO_HIGH	0x16	R	\	0	Y Axis Height of Gyro in Bytes
YGYRO_LOW	0x18	R	\	0	Y Axis Low of Gyro in Bytes
ZGYRO_HIGH	0x1A	R	\	0	Z Axis Height of Gyro in Bytes
ZGYRO_LOW	0x1C	R	\	0	Z Axis Low of Gyro in Bytes
XACCEL_HIGH	0x1E	R	\	0	Add Table X Axis Height Bytes
XACCEL_LOW	0x20	R	\	0	Add Table X Axis Low Bytes
YACCEL_HIGH	0x22	R	\	0	Add Table Y Axis Height Bytes
YACCEL_LOW	0x24	R	\	0	Add Table Y Axis Low Bytes
ZACCEL_HIGH	0x26	R	\	0	Add Table Z Axis Height Bytes
ZACCEL_LOW	0x28	R	\	0	Add Table Z Axis Low Bytes

2.4.1 SPI BURST Register

BURST is a continuous read register that reads all data in one stream without stopping between 16-bit segments.

Table 17 SPI BURST Register Format

Address	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	Read/Write
0x01									RW
Address	**bit7**	**bit6**	**bit5**	**bit4**	**bit3**	**bit2**	**bit1**	**bit0**	**Read/Write**
0x00	BURST_CMD								RW

BURST reading method is to send 0x8000 before reading to set BURST and start reading, then send 0x0000 all the time and receive data, output register content is offset by 2 SPI cycles from read command sending, and continue to select low level during reading.

Figure 4 SPI BURST Continuous Reading Diagram

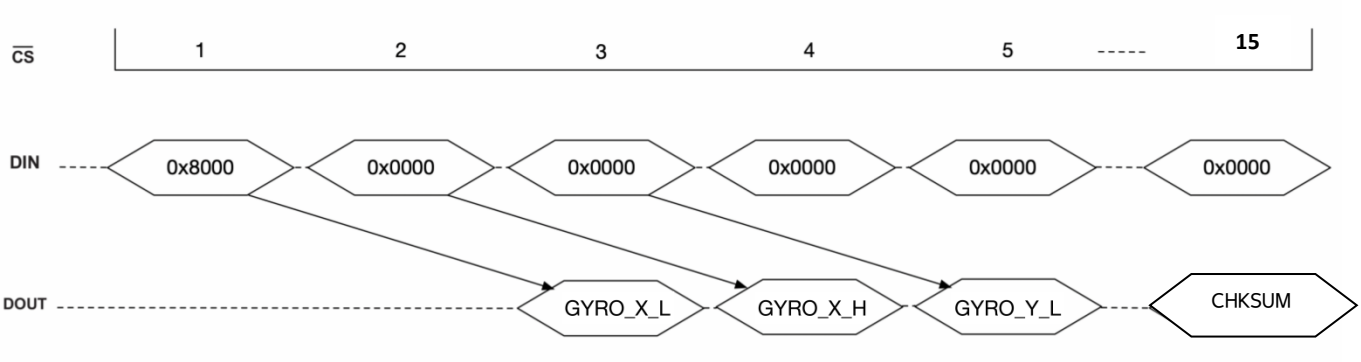


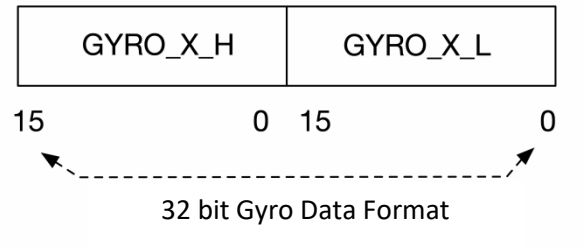
Table 18 SPI BURST Continuous Read Basic Format

Send Order	1	2	3	4	5	6
Send Content	GYRO_X_L	GYRO_X_H	GYRO_Y_L	GYRO_Y_H	GYRO_Z_L	GYRO_Z_H
	7	8	9	10	11	12
	ACCL_X_L	ACCL _X_H	ACCL _Y_L	ACCL _Y_H	ACCL _Z_L	ACCL _Z_H
	13					
	CHKSM					

Note 1 All data is 16-bit wide
Note 2 The data of gyroscope and accelerometer are joined and formatted as int32
Note 3 The CHKSM, CHECKSUM, is used to confirm data integrity Calculated by summing all data before CHECKSUM

During BURST continuous reading, 32-bit complete data is split into high 16-bit and low 16-bit output, respectively, using the small-end mode, that is, low-byte output first Users need to stitch the end and end of the 16-bit data to restore the complete 32-bit data.

Figure 5 SPI 32-bit Data Restore Diagram



Once complete 32-bit data is obtained, standard frame users can convert it to angular velocity, acceleration, temperature, and attitude angle information according to the following formulas.

Table 19 Standard Frame SPI 32 Bit Data Conversion Formula

Name	Units	Formula	Conditions/Notes
Angular velocity	°/s	$G = \frac{SF}{65536} \times \text{GYRO}$	▪ GYRO is the GYRO data for the X/Y/Z axis in the table above ▪ Gyro scale factor:$SF = 0.016$
Acceleration	mg	$A = \frac{SF}{65536} \times \text{ACCL}$	▪ ACCL is the ACCL data of X/Y/Z axis in the table above ▪ Burst mode: $SF = 0.2$ ▪ Single register mode: SF=0.2/1000
Temperature	℃	$T = \frac{SF}{65536} \times (\text{TEMP} - 172621824) + 25$	▪ TEMP is TEMP data from the table above ▪ Temperature scale factor$SF = -1/263.4$
Attitude angle	°	$D = \frac{SF}{65536} \times \text{ATT}$	▪ ATT is the ATT data in the table above ▪ Attitude scale factor:$SF = 0.00699411$

2.4.2 SPI FILTER_CTRL Register

FILTER_CTRL registers provide users with control over digital low-pass filters. This register is a read/write register, the write command is to send 0x86XX, and the current SPI cycle setting is valid; The read command sends 0x0600, and the output register contents are offset by 2 SPI cycles from the read command.

Table 20 SPI FILTER_CTRL register format

Address	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	Read/Write
0x07									RW
Address	**bit7**	**bit6**	**bit5**	**bit4**	**bit3**	**bit2**	**bit1**	**bit0**	**Read/Write**
0x06	Accelerometer Filter Configuration				Gyro Filter Configuration				RW

Table21 Filter configuration:

	Code	Describe
Accelerometer/Gyro Filter Configuration	4'b 0000	IIR filter fc=0.2 Hz
	4'b 0001	IIR filter fc=1 Hz
	4'b 0010	IIR filter fc=2 Hz
	4'b 0011	IIR filter fc=5 Hz
	4'b 0100	IIR filter fc=10 Hz
	4'b 0101	IIR filter fc=15 Hz
	4'b 0110	IIR filter fc=20 Hz
	4'b 0111	IIR filter fc=25 Hz
	4'b 1000	IIR filter fc=30 Hz
	4'b 1001	IIR filter fc=35 Hz
	4'b 1010	IIR filter fc=40 Hz
	4'b 1011	IIR filter fc=47 Hz
	4'b 1100	MOVING AVERAGE FILTER TAP=4
	4'b 1101	MOVING AVERAGE FILTER TAP=16
	4'b 1110	MOVING AVERAGE FILTER TAP=32
	4'b 1111	MOVING AVERAGE FILTER TAP=64

For example, if the gyroscope and accelerometer filter are set to 10Hz, the value 0x8644 is written.

2.4.3 SPI ID Register

The ID register is a read-only register, and the data content is the ASCII encoded character "IMU61x", which is similar to BURST data reading: send 0x6A00~0x7000 and receive data while reading. The output register content is offset by two cycles from the read instruction send.
The complete ID of the product can be obtained by splicing 4 16-bit ID data into ASICII code. Splicing method with BURST continuous read data, PROD_ID1 in high position, PROD_ID4 is low.

Table 22 SPI ID register format

Address	bit15 ~ bit0	Code	Read/Write
0x6A	PROD_ID1	0x0000	R
0x6C	PROD_ID2	0x494D	R
0X6E	PROD_ID3	0x5536	R
0x70	PROD_ID4 Coded content represents product ID	0x3132(IMU612)	R
		0x3134(IMU614)	
		0x3138(IMU618)	
		0x3141(IMU6132A)	
		0x3142(IMU6132B)	

2.4.4 SPI WIN_CTRL Register

This register is used to control the toggle window ID and is readable and writable. Window defaults to 0, write 0xFE01, then switch to 1.

Table 23 SPI WIN_CTRL Register Format

Address	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	Read/Write
0x7F									RW
Address	**bit7**	**bit6**	**bit5**	**bit4**	**bit3**	**bit2**	**bit1**	**bit0**	**Read/Write**
0x7E	WINDOW_ID								RW

Table 24 SPI Register WIN_CTRL.WINDOW_ID Encoding

Name	Code	Describe
WINDOW_ID	0x00	window0，Start reading data
	0x01	window1，Enter Configuration

3. I2C Communication Protocol

Sample I2C host read driver based on STM32:
http://www.forsense.cn/cn/h-col-128.html

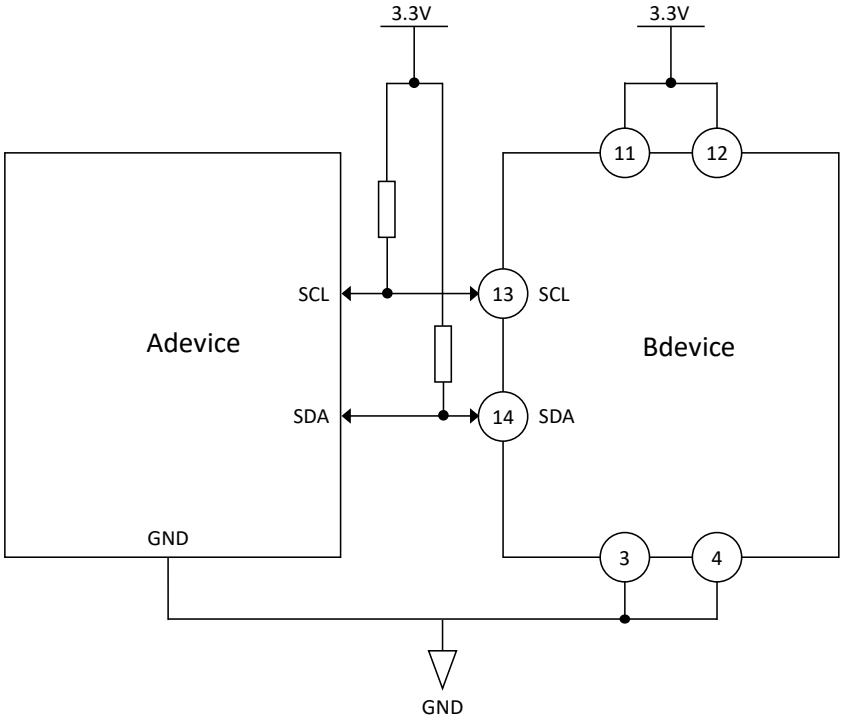
3.1 I2C Interface Parameters

Table 25 I2C Interface Parameters

I2C RATE	400KHz
I2C SLAVE ADDRESS (7 BITS)	0x18

3.2 I2C Connection Method

Figure 6 I2C Connection Method



Note: The pull-up resistance value is 4.7K

3.3 I2C Register

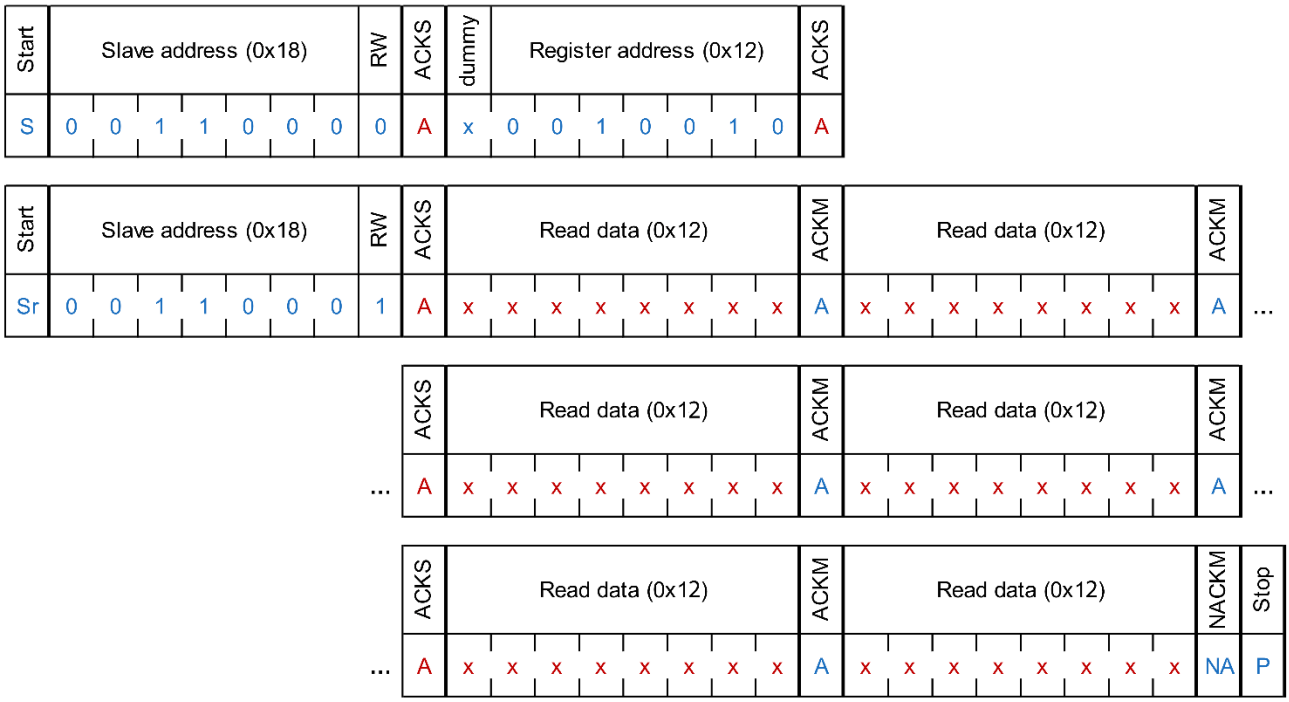
Table 26 I2C Register List

Name	Address	Read/Write	Default value	Describe
BURST	0x00	R		Continuous Read Register
FILTER_CTRL	0x06	RW	0xBB	Filter Selection
PROD_ID	0x6A	R		Product Name

3.3.1 I2C BURST Register

This I2C protocol supports continuous reading, continuous reading register address 0x12, automatic additive address from machine, and continuous output of 32 bytes in 8bit mode. The reading process is as follows:

Figure 7 I2C Continuous Read Mode



The frames are defined as follows:

Table 27 I2C Continuous Read Data Format

Send Order	1	2	3	4	5	6
Data Format	float	float	float	float	float	float
Send Content	ACCL_X	ACCL_Y	ACCL_Z	GYRO_X	GYRO_Y	GYRO_Z
	7	8				
	float	uint32				
	TEMP	CRC32				

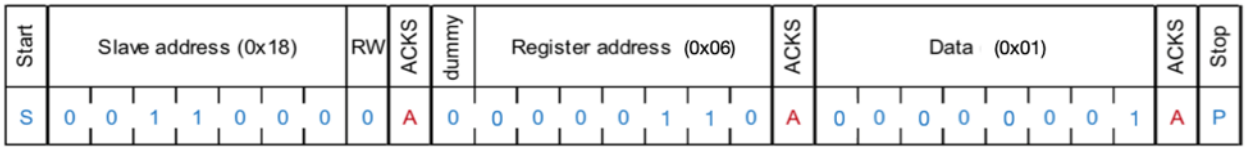
Note 1 TEMP is in degrees, gyroscope output is in degrees, accelerometer output is in g, and attitude output is in degrees.

Note 2 The initial value of CRC32 is 1. The CRC calculation does not include all the data of this frame. See Appendix 1 for the table-looking calculation.

3.3.2 I2C FILTER_CTRL Register

FILTER_ CTRL register address is 0x06, and the filter configuration reference table is the same as the SPI accelerometer and gyro filter configuration. The register reading process is the same as the I2C BURST reading method, and the writing register process is illustrated below.

Figure 8 I2C FILTER_CTRL Register Writing Method



3.3.3 I2C ID Register

The ID register address is 0x6A and the data content is the ASCII encoded character "IMU61B". The reading process is the same as I2C BURST, as shown in the table below.

Table 28 I2C ID Register Read Mode

Send Order	1	2	3	4
Send Content	0x00	0x00	0x49	0x4D
	5	6	7	8
	0x55	0x36	0x31	0x*

Note 1 All data is 8-bit wide

Note 2 0x* for product ID, 0x32 for IMU612, 0x34 for IMU614, 0x38 for IMU618, 0x41 for IMU6132A, 0x42 for IMU6132B

4. CAN Communication Protocol

Sample CAN host read driver based on STM32:
http://www.forsense.cn/cn/h-col-128.html

4.1 Communication parameters

- Interface form: CAN, standard frame
- CAN rate: 250Kbps~1Mbps (configurable)

4.2 Standard Frame Format

Table 29 CAN Standard Frame Format 101

Standard Frame ID	1	2	3	4	5	6	7	8
101	ROLL				PITCH			

Table 30 CAN Standard Frame Format 102

Standard Frame ID	1	2	3	4	5	6	7	8
102	YAW				Gx			

Table 31 CAN Standard Frame Format 103

Standard Frame ID	1	2	3	4	5	6	7	8
103	Gy				Gz			

Table 32 CAN Standard Frame Format 104

Standard Frame ID	1	2	3	4	5	6	7	8
104	Ax				Ay			

Table 33 CAN Standard Frame Format 105

Standard Frame ID	1	2	3	4	5	6	7	8
105	Az				TEMP		INDEX	

Note 1 Posture angle, gyro, accelerometer data are represented as float, temperature, count data are represented as int16
Note 2 TEMP unit is 100*C, the gyroscope output unit is °/s, the accelerometer output unit is g, and the attitude output unit is degree.

5. CRC32 Table Lookup Calculation

```
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,
    0xf9b9df6f, 0x8ebeeff9, 0x17b7be43, 0x60b08ed5, 0xd6d6a3e8, 0xa1d1937e,
    0x38d8c2c4, 0x4fdff252, 0xd1bb67f1, 0xa6bc5767, 0x3fb506dd, 0x48b2364b,
    0xd80d2bda, 0xaf0a1b4c, 0x36034af6, 0x41047a60, 0xdf60efc3, 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,
    0x40df0b66, 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;
}
```