AHRS IMU Sensor | WT61C

The Robust Acceleration, Angular velocity & Angle Detector

The WT61C is a IMU sensor device, detecting acceleration, angular velocity as well as angle. The robust housing and the small outline makes it perfectly suitable for industrial applications such as condition monitoring and predictive maintenance. Configuring the device enables the customer to address a broad variety of application by interpreting the sensor data by smart algorithms and Kalman filtering.

BUILT-IN SENSORS

Accelerometer

Gyroscope

Tutorial Link

Google Drive

Link to instructions DEMO:
WITMOTION Youtube Channel
WT61C Playlist

If you have technical problems or cannot find the information that you need in the provided documents, please contact our support team. Our engineering team is committed to providing the required support necessary to ensure that you are successful with the operation of our AHRS sensors.

Contact

Technical Support Contact Info

Application

- AGV Truck
- Platform Stability
- Auto Safety System
- 3D Virtual Reality
- Industrial Control
- Robot
- Car Navigation
- UAV
- Truck-mounted Satellite Antenna Equipment

Contents

1 Overview

WT61C's scientific name is AHRS IMU sensor. A sensor measures 3-axis angle, angular velocity, acceleration. Its strength lies in the algorithm which can calculate three-axis angle accurately.

WT61C is employed where the highest measurement accuracy is required. WT61C offers several advantages over competing sensor:

• Heated for best data availability: new WITMOTION patented zero-bias automatic detection calibration algorithm outperforms traditional accelerometer sensor

• High precision Roll Pitch Yaw (X Y Z axis) Acceleration + Angular Velocity + Angle output

• Low cost of ownership: remote diagnostics and lifetime technical support by WITMOTION service team

• Developed tutorial: providing manual, datasheet, Demo video, PC software, mobile phone APP, and 51 serial, STM32, Arduino, and Matlab sample code, communication protocol

• WITMOTION sensors have been praised by thousands of engineers as a recommended attitude measurement solution

2 Features

- The default baud rate of this device is 115200 and could be changed as 9600
- The interface of this product only leads to a serial port
- The module consists of a high precision gyroscope, accelerometer, geomagnetic field and barometer sensor. The product can solve the current real-time motion posture of the module quickly by using the high-performance microprocessor, advanced dynamic solutions and Kalman filter algorithm.
- The advanced digital filtering technology of this product can effectively reduce the measurement noise and improve the measurement accuracy.
- Maximum 100Hz data output rate. Output content can be arbitrarily selected, the output speed 20HZ or 100HZ adjustable.

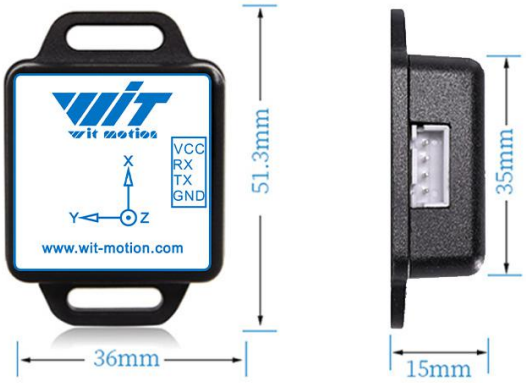
3 Specification

3.1 Parameter

Parameter	Specification
➢ Working Voltage	RS232:5V-36V
➢ Current	<30mA
➢ Size	51.3mm x 36mm X 15mm
➢ Data	Angle: X Y Z, 3-axis Acceleration: X Y Z, 3-axis Angular Velocity: X Y Z, 3-axis Time
➢ Output frequency	20Hz,100Hz
➢ Interface	Serial RS232 level
➢ Baud rate	115200(default, could be changed as 9600)

Measurement Range & Accuracy		
Sensor	Measurement Range	Accuracy/ Remark
➢ Accelerometer	X, Y, Z, 3-axis ±16g	Accuracy: 0.01g Resolution: 16bit Stability: 0.005g
➢ Gyroscope	X, Y, Z, 3-axis -±2000°/s	Resolution: 16bit Stability: 0.05°/s
➢ Angle/ Inclinometer	X, Y, Z, 3-axis X, Z-axis: ±180° Y ±90° (Y-axis 90° is singular point)	Accuracy:X, Y-axis: 0.05° Z-axis: 1° (Angle of Z-axis will have accumulated error)

3.2 Size



Parameter	Specification	Tolerance	Comment
Length	51.3	±0.1	Unit: millimeter.
Width	36	±0.1	
Height	15	±0.1	
Weight	13	±1	Unit: gram

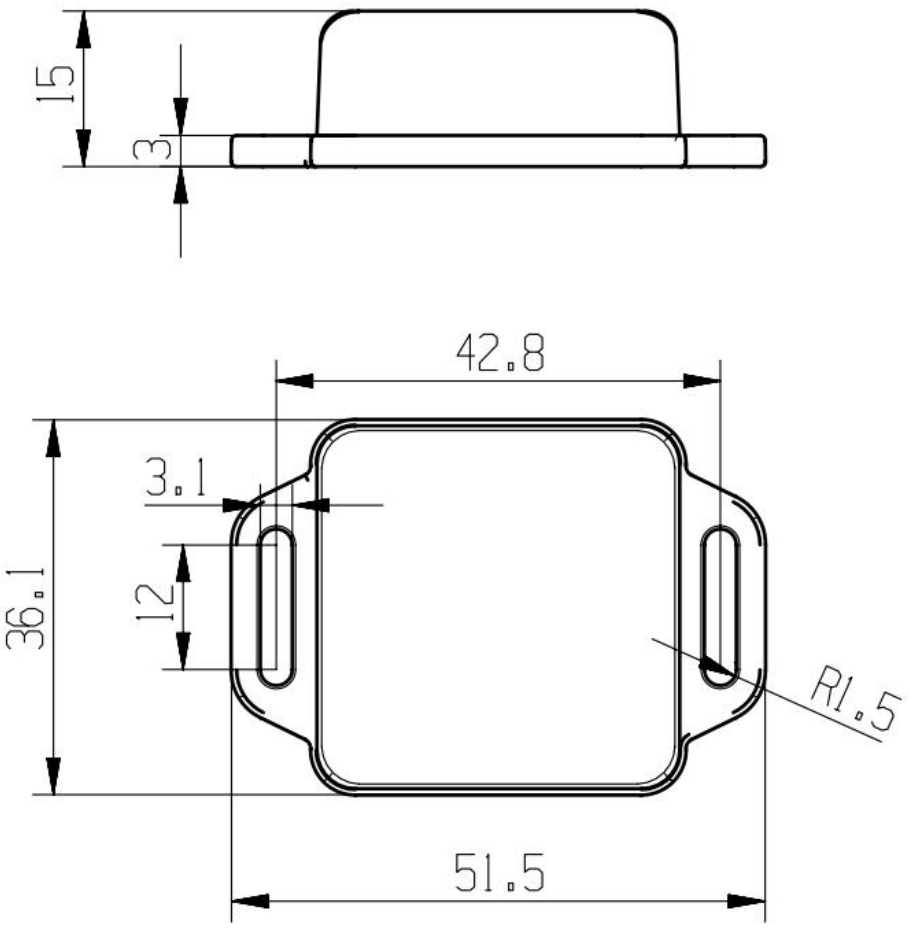
3.3 Axial Direction

The coordinate system used for attitude angle settlement is the northeast sky coordinate system. Place the module in the positive direction, as shown in the figure below, direction left is the Y-axis, the direction forward is the X-axis, and direction upward is the Z-axis. Euler angle represents the rotation order of the coordinate system when the attitude is defined as Z-Y-X, that is, first turn around the Z-axis, then turn around the Y-axis, and then turn around the X-axis.

4 Pin Definition

PIN	Function
➢ VCC	3.3-5V input supply
➢ RX	Serial data input, RS232 interface
➢ TX	Serial data output, RS232 interface
➢ GND	Ground

5 Casing Specification



6 Communication Protocol

Level: RS232 level

Band rate: 9600 , 115200(default), Stop bit 1, check digit 0.

Every frame data that the module sent to the PC software is divided into 3 packets, acceleration pocket, angular velocity packet and angle packet.

6.1 Output Data Format

6.1.1 Acceleration Output

Data Number	Data Content	Implication
0	0x55	Header of packet
1	0x51	Acceleration pack
2	AxL	X axis acceleration low type
3	AxH	X axis acceleration high type
4	AyL	Y axis acceleration low type
5	AyH	Y axis acceleration high type
6	AzL	Z axis acceleration low type
7	AzH	Z axis acceleration high type
8	TL	Temperature low type
9	TH	Temperature high type
10	Sum	Checksum

Formula for calculating acceleration：

ax=((AxH<<8)|AxL)/32768*16g(g is gravity acceleration, 9.8m/s2)
ay=((AyH<<8)|AyL)/32768*16g(g is gravity acceleration, 9.8m/s2)
az=((AzH<<8)|AzL)/32768*16g(g is gravity acceleration, 9.8m/s2)

Formula for calculating temperature：
T=((TH<<8)|TL) /340+36.53℃

Checksum：
Sum=0x55+0x51+AxH+AxL+AyH+AyL+AzH+AzL+TH+TL

Description:
1. The data is sent in hexadecimal not ASCII code.
2. Each data is transmitted in order of low byte and high byte, and the two are combined into a signed short type data. For example, the X-axis acceleration data Ax, where AxL is the low byte and AxH is the high byte. The conversion method is as follows:

Assuming that Data is actual data, DataH is the high byte part, and DataL is the low byte part, then: Data = ((short) DataH << 8) | DataL. It must be noted here that DataH needs to be converted to a signed short data first and then shifted, and the data type of Data is also a signed short type, so that it can represent negative numbers.

6.1.2 Angular Velocity Output

Data Number	Data Content	Implication
0	0x55	Header of packet
1	0x52	Angular velocity
2	wxL	X axis angular low type
3	wxH	X axis angular high type
4	wyL	Y axis angular low type
5	wyH	Y axis angular high type
6	wzL	Z axis angular low type
7	wzH	Z axis angular high type
8	TL	Temperature low type
9	TH	Temperature high type
10	Sum	Checksum

Formula for calculating angular velocity:

wx=((wxH<<8)|wxL)/32768*2000(°/s)
wy=((wyH<<8)|wyL)/32768*2000(°/s)
wz=((wzH<<8)|wzL)/32768*2000(°/s)

Formula for calculating temperature:
T=((TH<<8)|TL) /340+36.53℃

Checksum:
Sum=0x55+0x52+wxH+wxL+wyH+wyL+wzH+wzL+TH+TL

6.1.3 Angle Output

Data Number	Data Content	Implication
0	0x55	Header of packet
1	0x53	Angle packet
2	RollL	X axis angle low type
3	RollH	X axis angle high type
4	PitchL	Y axis angle low type
5	PitchH	Y axis angle high type
6	YawL	Z axis angle low type
7	YawH	Z axis angle high type
8	TL	Temperature low type
9	TH	Temperature high type
10	Sum	Checksum

Formula for calculating angle:

Roll angle(x axis) Roll=((RollH<<8)|RollL)/32768*180(°)
The pitching angle(y axis) Pitch=((PitchH<<8)|PitchL)/32768*180(°)
Yaw angle(zaxis) Yaw=((YawH<<8)|YawL)/32768*180(°)

Formula for calculating temperature：
T=((TH<<8)|TL) /340+36.53℃

Checksum：
Sum=0x55+0x53+RollH+RollL+PitchH+PitchL+YawH+YawL+TH+TL

Attention：

The coordinate system used for attitude angle settlement is the northeast sky coordinate system. Place the module in the positive direction, as shown in the figure on Chapter 3.3, direction left is the Y-axis, the direction forward is the X-axis, and direction upward is the Z-axis. Euler angle represents the rotation order of the coordinate system when the attitude is defined as Z-Y-X, that is, first turn around the Z-axis, then turn around the Y-axis, and then turn around the X-axis.

6.2 Config Commands

Instruction content	Function	Remark
0xFF 0xAA 0x52	Angle initialization	Z-axis to 0
0xFF 0xAA 0x67	Accelerometer Calibration	X,Y axis to 0
0xFF 0xAA 0x60	Dormancy/break dormancy	Standby state/working state
0xFF 0xAA 0x61	Use serial port, disable IIC	Set to serial output
0xFF 0xAA 0x62	Disable serial port, use IIC	Set to IIC interface output
0xFF 0xAA 0x63	Baud rate 115200, return rate 100HZ	Baud rate 115200
0xFF 0xAA 0x64	Baud rate 9600, return rate 20HZ	Baud rate 9600
0xFF 0xAA 0x65	Horizontal installation	Horizontal placed
0xFF 0xAA 0x66	Vertical installation	Vertical placed

Introductions:

1. When the module is on, it needs to be static first. Because the MCU inside the module will be automatically calibrated when it is static. After calibration, the angle of the Z-axis will be initialized to 0, Which can be considered as a signal that has been calibrated.

2. The factory default setting uses a serial port with a baud rate of 115200 and a frame rate of 100Hz. Configuration can be configured through the host computer software, because all configurations are saved after power-off, so only need to configure once.

6.3 Date Analysis and Sample Code

```
double a[3],w[3],Angle[3],T;

void DecodeIMUData(unsigned char chrTemp[])

{

    switch(chrTemp[1])

    {

    case 0x51:

        a[0] = (short(chrTemp[3]<<8|chrTemp[2]))/32768.0*16;

        a[1] = (short(chrTemp[5]<<8|chrTemp[4]))/32768.0*16;

        a[2] = (short(chrTemp[7]<<8|chrTemp[6]))/32768.0*16;

        T = (short(chrTemp[9]<<8|chrTemp[8]))/340.0+36.25;

        break;

    case 0x52:

        w[0] = (short(chrTemp[3]<<8|chrTemp[2]))/32768.0*2000;

        w[1] = (short(chrTemp[5]<<8|chrTemp[4]))/32768.0*2000;

        w[2] = (short(chrTemp[7]<<8|chrTemp[6]))/32768.0*2000;

        T = (short(chrTemp[9]<<8|chrTemp[8]))/340.0+36.25;

        break;

    case 0x53:
```

```
            Angle[0] = (short(chrTemp[3]<<8|chrTemp[2]))/32768.0*180;
            Angle[1] = (short(chrTemp[5]<<8|chrTemp[4]))/32768.0*180;
            Angle[2] = (short(chrTemp[7]<<8|chrTemp[6]))/32768.0*180;
            T = (short(chrTemp[9]<<8|chrTemp[8]))/340.0+36.25;
            printf("a = %4.3f\t%4.3f\t%4.3f\t\r\n",a[0],a[1],a[2]);
            printf("w = %4.3f\t%4.3f\t%4.3f\t\r\n",w[0],w[1],w[2]);
printf("Angle
= %4.2f\t%4.2f\t%4.2f\tT=%4.2f\r\n",Angle[0],Angle[1],Angle[2],T);
            break;
        }
}
```

6.4 Data Analysis Sample Under Embedded Environment

The code is divided into two parts, one of them is interrupt reception. Find the header of the data and then put the data-packet into an array. The other one is data analysis, which is put into the main program section.

Interrupt unit(The following is AVR microcontroller code, different MCU reads the register a little different):

```
unsigned char Re_buf[11],counter=0;

unsigned char sign;

interrupt [USART_RXC] void usart_rx_isr(void) //USART serial receiving

interrupt

{

Re_buf[counter]=UDR; //Different micro-controller is slightly

Different

if(counter==0&&Re_buf[0]!=0x55) return; //NO.0 data is not a frame

header ,skip.

counter++;

if(counter==11) //11 data has been received

{ counter=0; // Reassigned,ready for the next frame data reception

sign=1;

}
```

```
}
Main program section：
float a[3],w[3],angle[3],T;
extern unsigned char Re_buf[11],counter;
extern unsigned char sign;
while(1)
{
if(sign)
{
sign=0;
if(Re_buf[0]==0x55) //Check the frame header
{
switch(Re_buf [1])
{c
ase 0x51:
a[0] = (short(Re_buf [3]<<8| Re_buf [2]))/32768.0*16;
a[1] = (short(Re_buf [5]<<8| Re_buf [4]))/32768.0*16;
a[2] = (short(Re_buf [7]<<8| Re_buf [6]))/32768.0*16;
T = (short(Re_buf [9]<<8| Re_buf [8]))/340.0+36.25;
break;
```

```
case 0x52:
w[0] = (short(Re_buf [3]<<8| Re_buf [2]))/32768.0*2000;
w[1] = (short(Re_buf [5]<<8| Re_buf [4]))/32768.0*2000;
w[2] = (short(Re_buf [7]<<8| Re_buf [6]))/32768.0*2000;
T = (short(Re_buf [9]<<8| Re_buf [8]))/340.0+36.25;
break;
case 0x53:
angle[0] = (short(Re_buf [3]<<8| Re_buf [2]))/32768.0*180;
angle[1] = (short(Re_buf [5]<<8| Re_buf [4]))/32768.0*180;
angle[2] = (short(Re_buf [7]<<8| Re_buf [6]))/32768.0*180;
T = (short(Re_buf [9]<<8| Re_buf [8]))/340.0+36.25;
break;
}
}
      }
```