Introduction

An ultrasonic sensor measures distance by calculating the time difference between sending a signal and receiving an echo from an object. It uses echolocation to measure distances, just like a bat.

The URM07 is an ultra low-power sensor. The effect range is up to 7.5m, and it has a wide range power supply of 3.0 - 5.5V. It is directly compatible with 3.3V or 5V devices such as Arduino, Raspberry Pi .etc. The average operating current is only 5mA and standby current is only 14uA.

According to the principle of ultrasound sensors, the accuracy of the distance value will be affected by the air temperature and wind direction, so using a DFRobot URM07 built-in temperature sensor, you can read the ambient temperature, and automatically calculate temperature compensation. The sensor uses an integrated ultrasonic probe with a detection angle of approximately 60° in a compact and lightweight unit. The unit uses a 2.54mm pitch 4Pin interface, using UART communication, compatible with most common microcontrollers.
URM07 UART ultrasonic distance sensors have undertaken rigorous road testing and optimization. We guarantee that this sensor has first-class response speed, ultra high stability, the highest sensitivity and ultra low power consumption. If you have harsh environmental requirements for your sensor performance in your design, the URM07 is definitely the best choice.

Features

- Streamlined design
- High Stability
- High Sensitivity
- Built-in temperature compensation
- Low Power Consumption
- Reverse polarity protection (short-term protection)
- Overvoltage Protection
- Convenient connection and Usage

Dimension

- URM07 Detecting Angle

Specification
- Operating Voltage: 3.0 ~ 5.5V DC
- I/O Operating Mode: Open Drain (OD), integrated pull-up resistor
- Effective Range: 20 ~ 750 cm
- Direction Angle: 60 °
- Standby Current: <14 uA (mainly by the power chip consumption, the core circuit consumption <1uA)
- Peak Current: <9 mA
- Average Current: <5 mA (measured under continuous measurement, for reference only)
- Distance Resolution: 1cm
- Distance Error: 1%
- Operating Temperature Range: -10 ~ 70 °C
- Temperature Error: ± 1 °C
- Operating Humidity Range: RH <75%
- Acoustic Frequency: 38 ~ 42KHz
- Boot Start Time: <20ms
- Measurement Period: <60ms
- Dimension: 27 * 27 mm/ 1.06 * 1.06 inches
- Weight: 4.2g.

Communication Protocol

- Factory parameters:
- Serial parameters: 19200 (bps) 8N1
- Factory address: 0x11

Communication Command Frame Format:

<table>
<thead>
<tr>
<th>Header</th>
<th>Address</th>
<th>Data Length</th>
<th>Command</th>
<th>Data</th>
<th>Checksum</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x55</td>
<td>0xAA</td>
<td>1 byte</td>
<td>1 byte</td>
<td>Data</td>
<td>1 byte</td>
</tr>
</tbody>
</table>

Commands List:

<table>
<thead>
<tr>
<th>Command</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read distance</td>
<td>0x02</td>
</tr>
<tr>
<td>Read temperature</td>
<td>0x03</td>
</tr>
<tr>
<td>Set address</td>
<td>0x55</td>
</tr>
<tr>
<td>Set baud rate</td>
<td>0x08</td>
</tr>
</tbody>
</table>

The host sends a frame command to the ultrasonic module through the UART interface to trigger the module to start detecting, and then receives the distance value command returned by the module.
URM07 Distance Measurement

For example, the address of the ultrasonic module is 0x11, the host sends: \textbf{0x55 0xAA 0x11 0x00 0x02 0x12 [ 55 AA 11 00 02 12 ]}

Description:
Frame header \textbullet\ 0x55
Frame header \textbullet\ 0xAA
Device address \textbullet\ 0x11
Data length \textbullet\ 0x00
Command \textbullet\ 0x02
Checksum \textbullet\ 0x12

The ultrasonic module returns data as: \textbf{0x55 0xAA 0x11 0x02 0x02 0x00 0xCA 0xDE [ 55 AA 11 00 02 02 00 CA DE ]}

Description:
Frame header \textbullet\ 0x55
Frame header \textbullet\ 0xAA
Local address \textbullet\ 0x11
Data length \textbullet\ 0x02
Command \textbullet\ 0x02
Distance High \textbullet\ 0x00
Distance low \textbullet\ 0xCA (Distance value 0x00CA unit is centimeter, that is, decimal 202 cm)
Checksum \textbullet\ 0xDE
Read the Measured Temperature

The host reads the current temperature measured by the ultrasonic module via the UART interface.

URM07 Temperature Measurement

For example, the ultrasonic module address is 0x11, the host sends: \texttt{0x55 0xAA 0x11 0x00 0x03 0x13} [ \texttt{55 AA 11 00 03 13} ]

Description:
Frame header --------- 0x55
Frame header --------- 0xAA
Device address ----- 0x11
Data length ----- 0x00
Command --------- 0x03
Checksum ------- 0x13

The ultrasonic module returns data as: \texttt{0x55 0xAA 0x11 0x02 0x03 0x00 0xFF 0x14}

Description:
Frame header ------- 0x55
Frame header ------- 0xAA
Local address ----- 0x11
Data length ----- 0x02
Command -------- 0x03
Temperature high ----- 0x01
Temperature low ----- 0x13 (temperature value 0x0113, 27.5 degrees. Note: T = Returned value/10)
Checksum ------- 0x29
Set the Address of the Ultrasonic Module

The host sets the ultrasonic module address via the UART interface.

URM07 Modify the Address

For example, set the ultrasonic module address 0x12, the host sends: 0x55 0xAA 0xAB 0x01 0x55 0x12 0x12 [ 55 AA AB 01 55 12 12 ]

Description:
Frame header -------- 0x55
Frame header -------- 0xAA
Device address ------ 0xAB (0xAB for the broadcast address, that is, all the common address of the module, you can use 0xAB instead of the device itself address if you are not sure the current Address)
Data length ------ 0x01
Command ---------- 0x55
Set the address ------ 0x12 (need to set the device address 0x12)
Checksum -------- 0x12
Set the success of the ultrasonic module return command 0x55 0xAA 0x12 0x01 0x55 0xCC 0x33

Description:
Frame header --------- 0x55
Frame header --------- 0xAA
Device address ----- 0x12 (modified device address)
Data length ----- 0x01
Command --------- 0x55
Operation status ----- 0xCC (0xCC - operation completed 0xEE - operation failed)
Checksum ------- 0x33

Set the UART Communication Baud Rate

The host sets the ultrasonic module communication baud rate via the UART interface.

URM07 Modify the Baudrate
For example, if the address of the ultrasonic module is set to 0x11, the host sends: 0x55 0xAA 0x11 0x01 0x08 0x0B 0x24 //Set baudrate to 256000bps [ 55 AA 01 08 0B 24 ]

Description:
Frame header -------- 0x55
Frame header -------- 0xAA
Device address ----- 0x11
Data length ----- 0x01
Command --------- 0x08
Baud rate selection --- 0x0B
Checksum ------- 0x24

If the setting is successful, the ultrasonic module returns the command as: 0x55 0xAA 0x11 0x01 0x08 0xCC 0xE5

Description:
Frame header -------- 0x55
Frame header -------- 0xAA
Device address ----- 0x11
Data length ----- 0x01
Command --------- 0x08
Operation status ----- 0xCC (0xCC - operation completed 0xEE - operation failed)
Checksum ------- 0xE5

Baud rate Selection List:

<table>
<thead>
<tr>
<th>Baud rate Selection</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1200bps</td>
<td>0x00</td>
</tr>
<tr>
<td>2400bps</td>
<td>0x01</td>
</tr>
<tr>
<td>4800bps</td>
<td>0x02</td>
</tr>
<tr>
<td>9600bps</td>
<td>0x03</td>
</tr>
<tr>
<td>14400bps</td>
<td>0x04</td>
</tr>
<tr>
<td>19200bps</td>
<td>0x05</td>
</tr>
<tr>
<td>28800bps</td>
<td>0x06</td>
</tr>
<tr>
<td>38400bps</td>
<td>0x07</td>
</tr>
<tr>
<td>57600bps</td>
<td>0x08</td>
</tr>
<tr>
<td>115200bps</td>
<td>0x09</td>
</tr>
<tr>
<td>128000bps</td>
<td>0xA</td>
</tr>
<tr>
<td>256000bps</td>
<td>0xB</td>
</tr>
</tbody>
</table>
For example, when the device address is default 0x11, the baud rate setting command is as follows:

```
55 AA 11 01 08 00 19 //Set the baud rate 1200bps
55 AA 11 01 08 01 1A //Set the baud rate 2400bps
55 AA 11 01 08 02 1B //Set the baud rate 4800bps
55 AA 11 01 08 03 1C //Set the baud rate 9600bps
55 AA 11 01 08 04 1D //Set the baud rate 14400bps
55 AA 11 01 08 05 1E //Set the baud rate 19200bps
55 AA 11 01 08 06 1F //Set the baud rate 28800bps
55 AA 11 01 08 07 20 //Set the baud rate 38400bps
55 AA 11 01 08 08 21 //Set the baud rate 57600bps
55 AA 11 01 08 09 22 //Set the baud rate 115200bps
55 AA 11 01 08 0A 23 //Set the baud rate 128000bps
55 AA 11 01 08 0B 24 //Set the baud rate 256000bps
```

Arduino URM07 Ultrasonic Tutorial

Requirements

- **Hardware**
  DFRduino UNO (or similar) x 1
  URM07-UART Ultrasonic Sensor xn
  M-M/F-M/F-F Jumper wires

- **Software**
  Arduino IDE
  [Click to Download Arduino IDE from Arduino®](https://www.arduino.cc/en/Main/Software)
Read Distance Value via Arduino (Single)

Arduino URM07 Ultrasonic Single Control

Description:

Default Address: 0x11; Default Baudrate: 19200 bps
Leonardo GPIO Power Supply. (40mA)

// # Author: Strictus.zhang@dfrobot.com
// # Date: 20.08.2016
// # Product Name: URM07-UART Ultrasonic Sensor
// # SKU: SEN0153
// # version number: 1.0
// # Code Description: 20-750cm distance measurement, the received data is not verified

// # Connection: Arduino LEonardo GPIO Power Supply
// # Pin VCC (URM07 V1.0) -> D3 (Arduino Leonardo)
// # Pin GND (URM07 V1.0) -> D2 (Arduino Leonardo)
// # Pin RX (URM07 V1.0) -> TX1/D1 (Arduino Leonardo)
// # Pin TX (URM07 V1.0) -> RX1/D0 (Arduino Leonardo)

#define header_H   0x55 //Header
#define header_L   0xAA //Header
#define device_Addr 0x11 //Address
#define data_Length 0x00 //Data length
#define get_Dis_CMD 0x02 //Command: Read Distance
#define checksum   (header_H+header_L+device_Addr+data_Length+get_Dis_CMD) / Checksum

unsigned char i=0;
unsigned int  Distance=0;
unsigned char Rx_DATA[8];
unsigned char CMD[6]={{
    header_H,header_L,device_Addr,data_Length,get_Dis_CMD,checksum}; //Distance command package

void setup() {
    pinMode(2, OUTPUT);
    pinMode(3, OUTPUT);
    digitalWrite(3, HIGH);  //Ultrasonic VCC
    digitalWrite(2, LOW);  //Ultrasonic GND
    Serial1.begin(19200);  //Serial1: Ultrasonic Sensor Communication Serial Port, Baudrate: 19200
    Serial.begin(19200);   //Serial: USB Serial Data output, baudrate: 19200
}

void loop() {
    for(i=0;i<6;i++){
        Serial1.write(CMD[i]);
    }
    delay(150);  //Wait for the result
    i=0;
while (Serial1.available()){  //Read the return data (Note: this demo is only for the reference, no data verification)
    Rx_DATA[i++]=(Serial1.read());
}

Distance=((Rx_DATA[5]<<8)|Rx_DATA[6]);  //Read the distance value
Serial.print(Distance);               //print distance value
Serial.println("cm");
}
# Read Temperature Value via Arduino (Single)

// # Author: Strictus.zhang@dfrobot.com
// # Date: 20.08.2016
// # Product Name: URM07-UART Ultrasonic Sensor
// # SKU: SEN0153
// # version number: 1.0
// # Code Description: Temperature measurement, the received data is not verified

// # Connection: Arduino LEonardo GPIO Power Supply
// # Pin VCC (URM07 V1.0) -> D3 (Arduino Leonardo)
// # Pin GND (URM07 V1.0) -> D2 (Arduino Leonardo)
// # Pin RX (URM07 V1.0) -> TX1/D1 (Arduino Leonardo)
// # Pin TX (URM07 V1.0) -> RX1/D0 (Arduino Leonardo)

#define header_H     0x55 //Header
#define header_L     0xAA //Header
#define device_Addr  0x11 //Address
#define data_Length  0x00 //Data length
#define get_Temp_CMD 0x03 //Command: Read Temperature
#define checksum     (header_H+header_L+device_Addr+data_Length+get_Temp_CMD)

unsigned char i=0;
int  temperature=0;
unsigned char Rx_DATA[8];

void setup() {
    pinMode(2, OUTPUT);
    pinMode(3, OUTPUT);
    digitalWrite(3, HIGH); // Ultrasonic Sensor VCC
```java
digitalWrite(2, LOW);  // Ultrasonic Sensor GND

Serial1.begin(19200);  // Serial1: Ultrasonic Sensor Communication Serial Port, Baud rate: 19200
Serial.begin(19200);   // Serial: USB Serial Monitor

void loop() {
    for(i=0;i<6;i++){
        Serial1.write(CMD[i]);
    }
    delay(50);  //Wait Data Return
    i=0;
    while (Serial1.available()){
        Rx_DATA[i++]=(Serial1.read());
    }
    temperature=((Rx_DATA[5]<<8)|Rx_DATA[6]);  //Read temperature Value (10 times)
    Serial.print(temperature/10);             //Print Temperature
    Serial.print('.');
    Serial.print(temperature%10);
    Serial.println("C");
}
```
URM07 Temperature Display

Read Distance Value via Arduino (Cascade)

URM07 Cascade Connection
unsigned char i=0,j=0;
unsigned int Distance=0;
unsigned char Rx_DATA[8];
unsigned char CMD[4][6] = {
    {0x55,0xAA,0x11,0x00,0x02,0x12},
    {0x55,0xAA,0x22,0x00,0x02,0x23},
    {0x55,0xAA,0x33,0x00,0x02,0x34},
    {0x55,0xAA,0x44,0x00,0x02,0x45},
}; //Distance Measurement Package

void setup() {
    pinMode(2, OUTPUT);
    pinMode(3, OUTPUT);
    digitalWrite(3, HIGH); //Ultrasonic Sensor VCC
digitalWrite(2, LOW);  //Ultrasonic Sensor GND
Serial1.begin(19200);  //Serial1: Ultrasonic Sensors Serial Communication Port, baudrate: 19200
Serial.begin(19200);   //Serial: USB Serial Monitor
}

void loop() {
    for(j=0;j<4;j++)
    {
        for(i=0;i<6;i++){
            Serial1.write(CMD[j][i]);
        }
        delay(150);  //Wait returned result
        i=0;
        while (Serial1.available()){
            Rx_DATA[i++]=(Serial1.read());
        }
        Distance=((Rx_DATA[5]<<8)|Rx_DATA[6]); //Read distance value
    }
    Serial.print("URM07-UART module[";
    Serial.print(j);
    Serial.print("");
    Serial.print(Distance);
    Serial.println("cm");
}
Serial.print("\r\n\r\n");
}
URM07 Cascade

For any questions, advice or cool ideas to share, please visit the DFRobot Forum.