Overview
Introduction
Feature
Application Scenarios
Specification
Outline Dimensions
Working Principle
Communication Interface
Wiring Description
Software and Use
Example
Windows Quick Test
Preparation
Hardware Connection
Test Step
Working with Raspberry Pi
Environment
Configuration
UART Port
Configuration
Enable I2C Interface
Install WiringPi
32-bit Raspberry Pi System
64-bit Raspberry Pi System
Hardware Connection
Software Example
Preparation
C
Python
Working with Raspberry Pi Pico
Hardware Connection
Software Debugging
Preparation
Running Example
Working with Arduino
Hardware Connection
Software Debugging
Working with ESP32
Hardware Connection
Software Debugging
Preparation
Running Example
Resource
Demo
Software
Related File
FAQ
To Top

TFmini-S LiDAR Range Sensor

Overview

TFmini-S LiDAR Range Sensor

I2C, UART

Introduction

TFmini-S LiDAR Sensor is an upgrade project based on TFmini, which is a miniaturized, single-point distance measurement product based on the ToF (Time of Flight) principle with unique optical, electrical, and algorithm design, mainly to achieve stable, accurate, high sensitivity and high-speed distance measurement functions. In addition to the low cost, small size, and long range of TFmini, the product has higher range accuracy, better adaptability to different environments such as outdoor bright light, different temperatures, and different reflectivity, lower power consumption, and more flexible detection frequency. The product is compatible with both UART and I2C communication interfaces and can be switched by command.

Feature

- Compact structure
- Lightweight
- Low-power consumption
- High frame rate (up to 1000Hz output frequency)

Application Scenarios

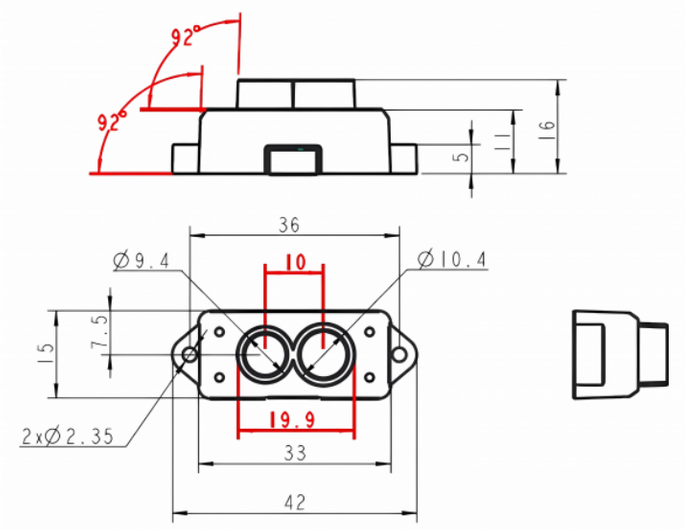
- Pedestrian detection
- Vehicle detection
- Anti-smash barrier
- Altimeter

Specification

Product	TFmini-S LiDAR Range Sensor	
PERFORMANCE	Ranging distance	0.1m~12m@90% reflectivity 0.1m~7m@10% reflectivity 0.1m~12m@90% reflectivity (Indoor 70Klux) 0.1m~7m@10% reflectivity (Indoor 70Klux)
	Accuracy	±6cm@(0.1m-6m) ±2%@(6m-12m)
	Distance resolution	1cm
	Frame rate	1-1000Hz (Default 100Hz)
	Ambient light resistance	70Klux
	Operating temperature	0℃~60℃
OPTICAL PARAMETERS	Light source	VCSEL
	Central wavelength	850nm
	Eye safety	Class1(IEC60825)
	Field of view	2°
ELECTRICAL PARAMETERS	Power supply	5V±0.1V
	Average current	≤140mA
	Power	≤0.7W
	Peak current	200mA
	Communication level	LVTTL(3.3V)
	Communication interface	UART/I2C

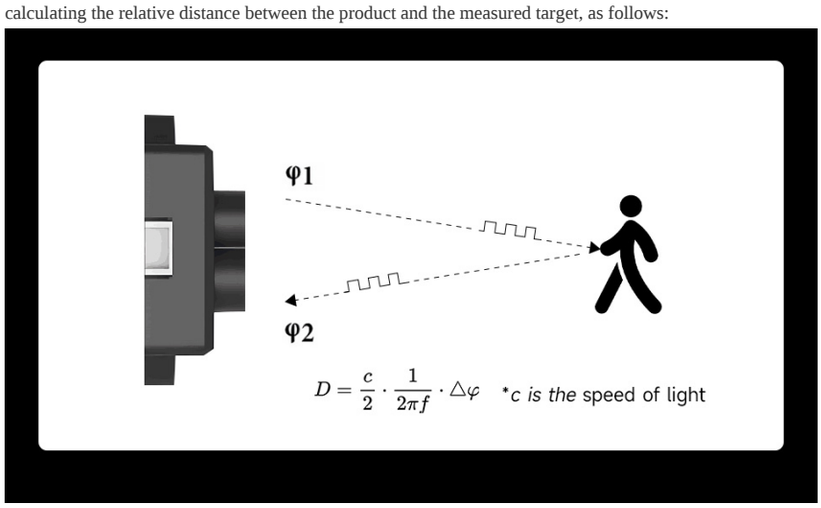
OTHERS	Dimension	42mm*15mm*16mm (L*W*H)
	Case material	ABS/PC
	Storage temperature	-20℃~75℃
	Weight	5g±0.3g
	Cable Length	10cm

Outline Dimensions



Working Principle

This product is based on the TOF (Time of Flight) principle. Specifically, the product periodically sends out a modulated wave of near-infrared light, which is reflected when it meets an object. The time of flight is obtained by measuring the phase difference between the modulated wave and the round trip, and then calculating the relative distance between the product and the measured target, as follows:



Communication Interface

- UART / I2C

UART Interface

Default Baudrate	**115200 (Adjustable)**
Data Bit	8
Stop Bit	1
Parity Bit	None

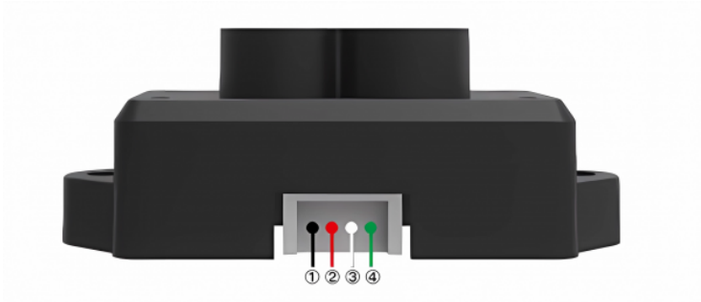
I2C Interface

Maximum Transmission Rate	**400kbps**
Host Mode	Slave Mode
Default Address	0x10
Address Range	0x08~0x77

- I/O mode:

  In I/O output mode, the detection information is indicated by the high and low levels of pin 6. When the output is near-zone level, the output switches to far-zone level after the measured value is greater than the far end point of the hysteresis interval; when the output is far-zone level, the output switches to near-zone level when the measured value is less than the near end point of the hysteresis interval. (High level: 3.3V, low level: 0V)

Wiring Description



Number	Color	Function	Note
1	Black	GND	Ground
2	Red	+5V	Power Input
3	White	RXD/SDA	Receive/Data
4	Green	TXD/SCL	Transmit/Clock

For more information on how to switch communication interfaces and set interface parameters, see: TFmini-S Note

Software and Use Example

The following section describes how the module can be used under Windows PC, Raspberry Pi, Raspberry Pi Pico, Arduino, ESP32-S3, and other operating system/open-source hardware development platforms.

OS/Hardware Platform	Programming Languages / Software
Windows	Host computer software
Raspberry Pi	Python, C（WiringPi）
Raspberry Pi Pico	MicroPython
Arduino	C（Arduino IDE）
ESP32	C（Arduino IDE）

Windows Quick Test

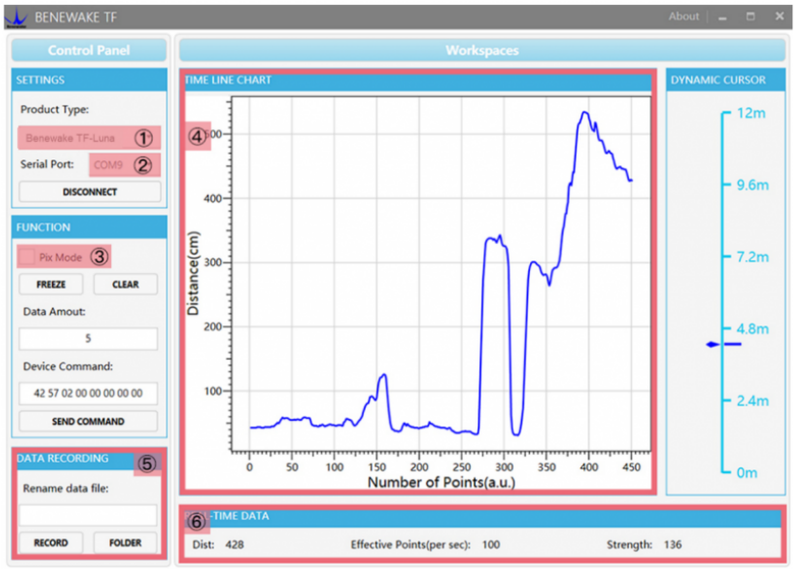
Preparation

- TFmini-S
- USB TO TTL
- Windows (Click to download Host computer software)

Hardware Connection

Test Step

1. Refer to the hardware connection diagram, connect the TFmini-S module to the computer via the USB TO TTL UART module.
2. Open the TF host computer, choose 『① TFmini-S』, and select the occupied serial port that is automatically identified (here is 『② COM9』). Then, click 『CONNECT』 to connect to the upper computer. After a successful connection, the continuous output data image will appear in the right 『④ TIME LINE CHART』 area, and the bottom 『⑥ REAL TIME DATA』 area will show the current test distance (Dist), effective data points per second (Effective Points) and signal strength (Strength) in real-time.



Note:

a) If there is no data in the "④ TIME LINE CHART" area, please check the connection and cable sequence, TFmini-S is powered on successfully, and there will be a faint red light inside the emitting lens when viewed from the front.

b) If TFmini-S is outputting in Pixhawk format, you need to check "③ Pix Mode" first, then the "④ TIME LINE CHART" area will output the data image normally. After Pix Mode is checked, the distance unit will be changed to m.

c) The distance output Dist value is different according to the output unit, the default unit is cm, if the distance unit of TFmini-S is changed to mm by the command, the upper computer can not distinguish, the unit of "④ TIME LINE CHART" is still cm. For example, if the actual distance measured by TFmini-S is 1m, the unit in mm is 1000, and the value read by the upper unit is 1000, but the unit on the upper unit does not change and still shows cm.

Working with Raspberry Pi

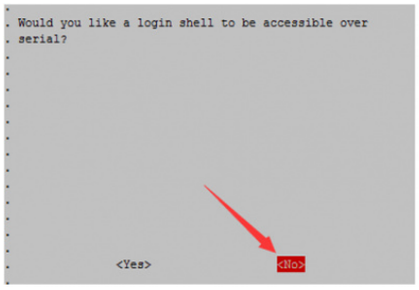
Environment Configuration

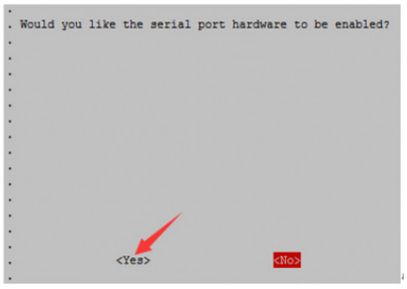
UART Port Configuration

Since the Raspberry Pi serial port is used for terminal debugging by default, if you need to use the serial port, you need to modify the Raspberry Pi settings. Execute the following command to enter the Raspberry Pi configuration:

```
sudo raspi-config
```

Select Interfacing Options -> Serial -> no -> yes, and disable the serial debug function.





Open the /boot/config.txt file and find the following configuration statement to enable the serial port, if not, add it at the end of the file:

```
enable_uart=1
```

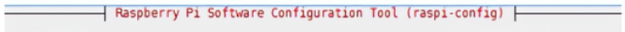
And then reboot the Raspberry Pi:

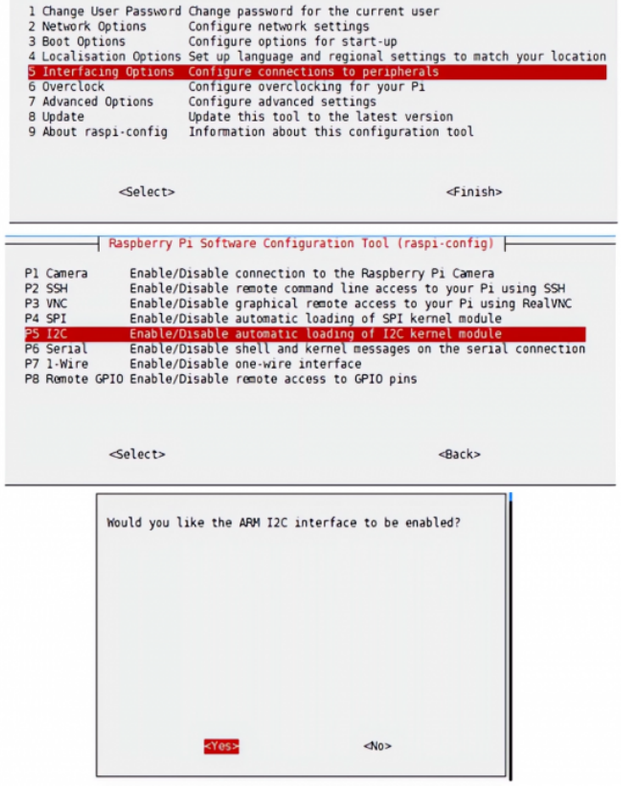
```
sudo reboot
```

Enable I2C Interface

Open the Raspberry Pi terminal and run the following commands to enter the configuration interface.

```
sudo raspi-config
Select Interfacing Options -> I2C -> Yes, enable the I2C interface
```





And then reboot the Raspberry Pi:

```
sudo reboot
```

Install WiringPi

Make sure your computer is 32-bit or 64-bit before installation.

32-bit Raspberry Pi System

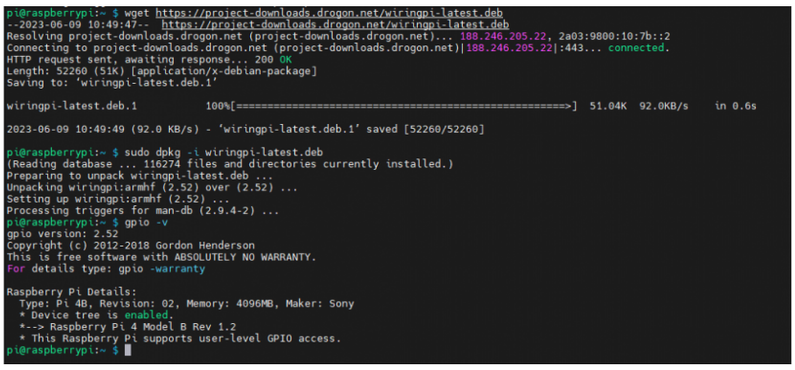
Execute the following commands separately for WiringPi installation:

```
wget https://project-downloads.drogon.net/wiringpi-latest.deb
sudo dpkg -i wiringpi-latest.deb
gpio -v
```

(Optional) If an error occurs during installation, install all missing dependencies and other necessary packages by running the following command, and then re-run the installation command:

```
sudo apt --fix-broken install
```

Run "gpio -v" and version 2.52 will appear. If it does not appear, there is an installation error:



64-bit Raspberry Pi System

Copy the resource package to the Raspberry Pi using the command:

```
wget https://www.waveshare.com/w/upload/8/8c/WiringPi-master.zip
```

(optional, you can skip this step if you have used the unzip command) Install the unzip environment:

```
sudo apt-get install unzip
```

Go to the file location and execute the unzip command:

```
unzip WiringPi-master.zip
```

Go to the file directory (go to the "WiringPi-master" folder):

```
cd WiringPi-master/
```

Run sudo . /build:

```
sudo . /build
```

(optional, see point 4 for errors) If . /build does not work, execute "chmod +x . /build" and then "sudo . /build":

```
chmod +x . /build
```

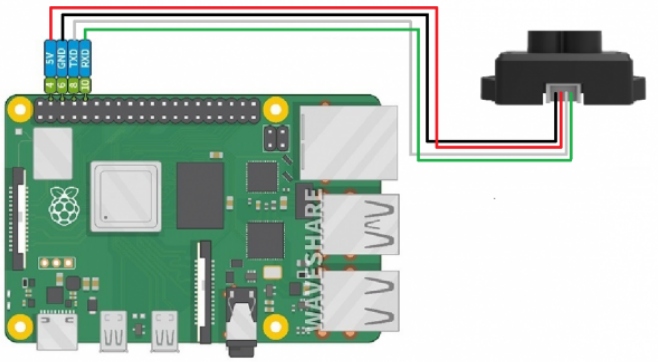
Example:

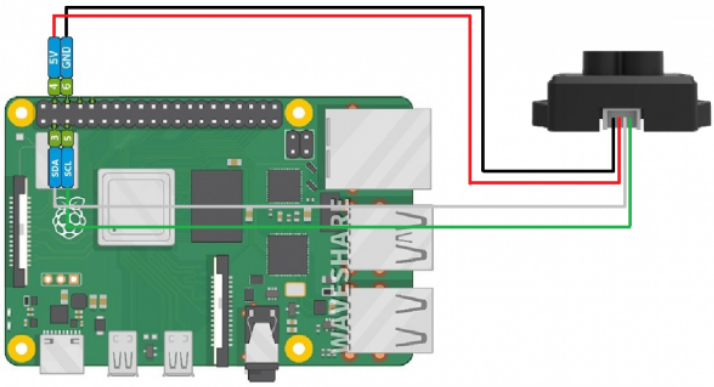
```
pi@raspberrypi:~/Desktop $ wget https://www.waveshare.net/w/upload/8/8c/WiringPi-master.zip
--2023-06-12 04:58:59--  https://www.waveshare.net/w/upload/8/8c/WiringPi-master.zip
Resolving www.waveshare.net (www.waveshare.net)... 120.77.147.21
Connecting to www.waveshare.net (www.waveshare.net)|120.77.147.21|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 286485 (280K) [application/zip]
Saving to: 'WiringPi-master.zip'

WiringPi-master.zip                 100%[===================================================================================>] 279.77K  --.-KB/s    in 0.04s

2023-06-12 04:58:59 (7.44 MB/s) - 'WiringPi-master.zip' saved [286485/286485]

pi@raspberrypi:~/Desktop $ sudo apt-get install unzip
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
unzip is already the newest version (6.0-26+deb11u1).
The following package was automatically installed and is no longer required:
  libfuse2
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
pi@raspberrypi:~/Desktop $ unzip WiringPi-master.zip
Archive:  WiringPi-master.zip
pi@raspberrypi:~/Desktop $ ls
WiringPi-master  WiringPi-master.zip
pi@raspberrypi:~/Desktop $ cd WiringPi-master/
pi@raspberrypi:~/Desktop/WiringPi-master $ sudo ./build
wiringPi Build script
====================
```

Hardware Connection

UART mode: (Connect to GPIO14 and GPIO15).

IIC mode: (Connect to GPIO2 and GPIO3).





Software Example

Preparation

Download the program packet to the Raspberry Pi.

```
wget https://www.waveshare.com/w/upload/6/65/TFmini-S-LiDAR-Range-Sensor-Demo.zip
```

Unzip the file and create a new file with the same name and extract it to the appropriate folder:

```
unzip TFmini-S-LiDAR-Range-Sensor-Demo.zip -d ./TFmini-S-LiDAR-Range-Sensor-Demo
```

Example:

```
pi@raspberrypi:~/Desktop $ wget https://www.waveshare.net/w/upload/6/65/TFmini-S-LiDAR-Range-Sensor-Demo.zip
--2023-06-12 07:40:28--  https://www.waveshare.net/w/upload/6/65/TFmini-S-LiDAR-Range-Sensor-Demo.zip
Resolving www.waveshare.net (www.waveshare.net)... 120.77.147.21
Connecting to www.waveshare.net (www.waveshare.net)|120.77.147.21|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 16067 (16K) [application/zip]
Saving to: 'TFmini-S-LiDAR-Range-Sensor-Demo.zip'

TFmini-S-LiDAR-Range-Sensor-Demo.zip       100%[===================================================================================>]  15.69K  --.-KB/s    in 0s

2023-06-12 07:40:28 (63.8 MB/s) - 'TFmini-S-LiDAR-Range-Sensor-Demo.zip' saved [16067/16067]

pi@raspberrypi:~/Desktop $ ls
TFmini-S-LiDAR-Range-Sensor-Demo.zip
pi@raspberrypi:~/Desktop $ unzip TFmini-S-LiDAR-Range-Sensor-Demo.zip -d ./TFmini-S-LiDAR-Range-Sensor-Demo
Archive:  TFmini-S-LiDAR-Range-Sensor-Demo.zip
```

C

The default mode is UART, if you switch modes, please remember the current mode.

The following operation is based on running the "UART" demo as an example. If you are using the I2C interface, please make sure you have connected and set up the corresponding interface, and do the same.

Go to the package path "TFmini-S-LiDAR-Range-Sensor-Demo" in the terminal.

```
cd TFmini-S-LiDAR-Range-Sensor-Demo/
```

Enter the sample file path, the 4 files under the folder corresponding to the 4 main controls, and the files in the subordinate folder are the data acquisition demos and mode switching procedures for the TFmini-S in the corresponding mode (example: using UART mode to acquire data).

```
cd Raspberry\ Pi/WiringPi/UART
```

Use "gcc -o main main.c -lwiringPi" to compile the program.

```
gcc -o main main.c -lwiringPi
```

Run the main program (Previously, you need to open the relevant functions in the front Raspberry Pi configuration).

```
sudo ./main
```

Execute:

```
pi@raspberrypi:~/Desktop $ ls
TFmini-S-LiDAR-Range-Sensor-Demo  TFmini-S-LiDAR-Range-Sensor-Demo.zip
pi@raspberrypi:~/Desktop $ cd TFmini-S-LiDAR-Range-Sensor-Demo/Raspberry\ Pi/WiringPi/UART
pi@raspberrypi:~/Desktop/TFmini-S-LiDAR-Range-Sensor-Demo/Raspberry Pi/WiringPi/UART $ ls
main.c
pi@raspberrypi:~/Desktop/TFmini-S-LiDAR-Range-Sensor-Demo/Raspberry Pi/WiringPi/UART $ gcc -o main main.c -lwiringPi
pi@raspberrypi:~/Desktop/TFmini-S-LiDAR-Range-Sensor-Demo/Raspberry Pi/WiringPi/UART $ ls
main  main.c
pi@raspberrypi:~/Desktop/TFmini-S-LiDAR-Range-Sensor-Demo/Raspberry Pi/WiringPi/UART $ sudo ./main
TF-Luna C portion
Distance: 331 cm
Strength: 768
Chip Temperature: 60℃
TF-Luna C portion
Distance: 331 cm
Strength: 776
Chip Temperature: 60℃
TF-Luna C portion
Distance: 332 cm
Strength: 773
Chip Temperature: 60℃
TF-Luna C portion
Distance: 331 cm
Strength: 778
Chip Temperature: 60℃
TF-Luna C portion
Distance: 330 cm
Strength: 768
Chip Temperature: 60℃
TF-Luna C portion
```

Python

The default mode is UART, if you switch modes, please remember the current mode.

The following operation is based on running the "UART" demo as an example. If you are using the I2C interface, please make sure you have connected and set up the corresponding interface, and do the same.

Go to the package path "TFmini-S-LiDAR-Range-Sensor-Demo" in the terminal.

```
cd TFmini-S-LiDAR-Range-Sensor-Demo/
```

Enter the sample file path, the 4 files under the folder corresponding to the 4 main controls, and the files in the subordinate folder are the data acquisition demos and mode switching procedures for the TFmini-S in the corresponding mode (for example: using UART mode to acquire data).

```
cd Raspberry\ Pi/Python/UART
```

Run the demo:

```
python3 main.py
```

Example:

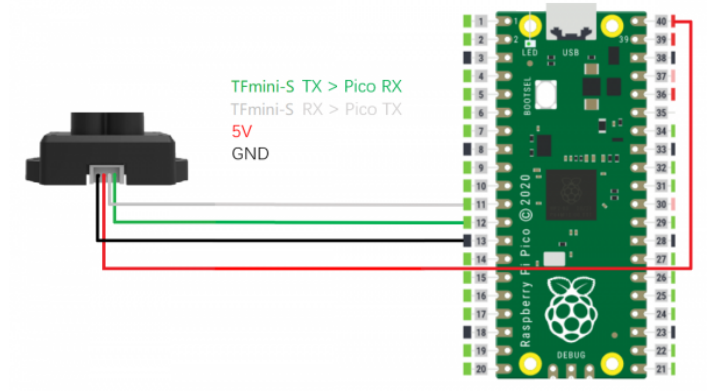
```
pi@raspberrypi:~/Desktop $ ls
 TFmini-S-LiDAR-Range-Sensor-Demo  TFmini-S-LiDAR-Range-Sensor-Demo.zip
pi@raspberrypi:~/Desktop $ cd TFmini-S-LiDAR-Range-Sensor-Demo/Raspberry\ Pi/Python/UART
pi@raspberrypi:~/Desktop/TFmini-S-LiDAR-Range-Sensor-Demo/Raspberry Pi/Python/UART $ ls
 main.py
pi@raspberrypi:~/Desktop/TFmini-S-LiDAR-Range-Sensor-Demo/Raspberry Pi/Python/UART $ python3 main.py
 TF-Luna python3 portion
 Distance:332cm, Strength:782Chip Temperature:60.0℃
 TF-Luna python3 portion
 Distance:331cm, Strength:784Chip Temperature:60.0℃
 TF-Luna python3 portion
 Distance:333cm, Strength:782Chip Temperature:60.0℃
 TF-Luna python3 portion
 Distance:333cm, Strength:781Chip Temperature:60.0℃
 TF-Luna python3 portion
 Distance:331cm, Strength:779Chip Temperature:60.0℃
 TF-Luna python3 portion
 Distance:331cm, Strength:781Chip Temperature:60.0℃
 TF-Luna python3 portion
 Distance:332cm, Strength:778Chip Temperature:60.0℃
 TF-Luna python3 portion
 Distance:332cm, Strength:777Chip Temperature:60.0℃
 TF-Luna python3 portion
 Distance:332cm, Strength:785Chip Temperature:60.0℃
 TF-Luna python3 portion
 Distance:333cm, Strength:776Chip Temperature:60.0℃
 TF-Luna python3 portion
 Distance:333cm, Strength:786Chip Temperature:60.0℃
 TF-Luna python3 portion
 Distance:331cm, Strength:774Chip Temperature:60.0℃
 TF-Luna python3 portion
 Distance:331cm, Strength:779Chip Temperature:60.0℃
 TF-Luna python3 portion
 Distance:332cm, Strength:784Chip Temperature:60.0℃
```

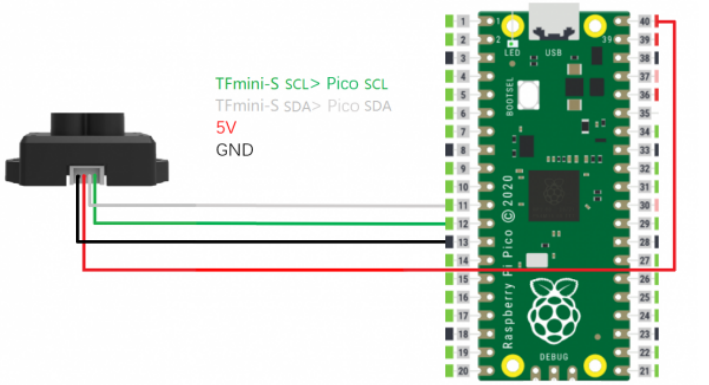
Working with Raspberry Pi Pico

Hardware Connection

UART mode: (Connect to GPIO8 and GPIO9).

IIC mode: (Connect to GPIO8 and GPIO9).



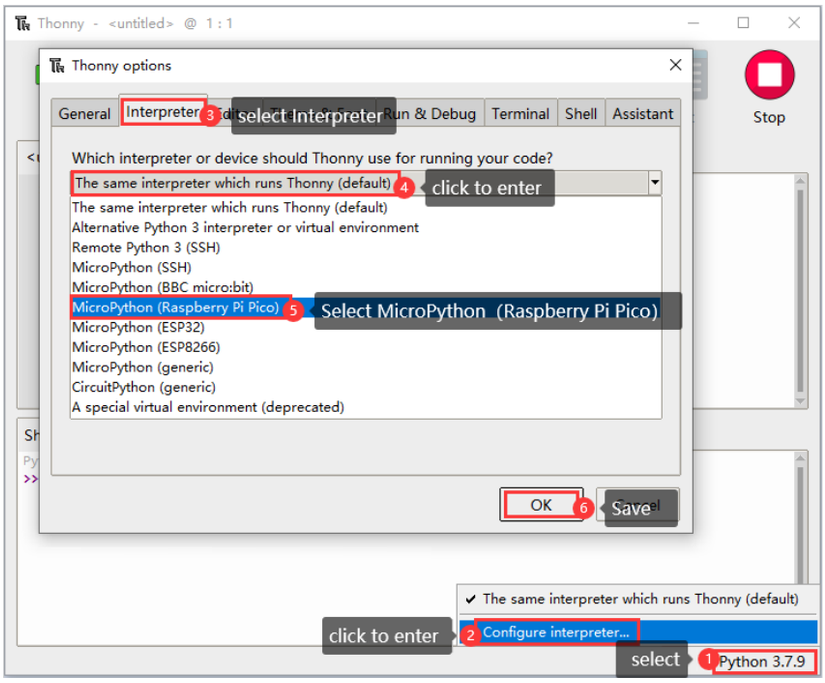


Software Debugging

Preparation

1. Install Thonny. (Thonny Python IDE (Windows V3.3.3))
2. Press the "BOOTSEL" key on the Raspberry Pi Pico, and release it after powering it on.
3. A new disk will appear on your computer, extract the firmware (Raspberry Pi Pico firmware) and copy the firmware (suffix uf2) to that disk (the disk will automatically disappear if the copy is successful).
4. Open Thonny, click on "Python x.x.x" at the bottom right, and select "Configure interpreter".
5. Select "Interpreter" in the pop-up window -> select "MicroPython (Raspberry Pi Pico)" as the interpreter -> select "Auto-detect port" as the port.
6. Click on "Stop", and the Shell window will show "MicroPython v1.20.0-50-g786013d46 on 2023-05-04; Raspberry Pi Pico with RP2040 Type "help()" for more information. " is a successful connection.

- The following is the procedure for Step 4 and 5:



The successful connection is shown below:

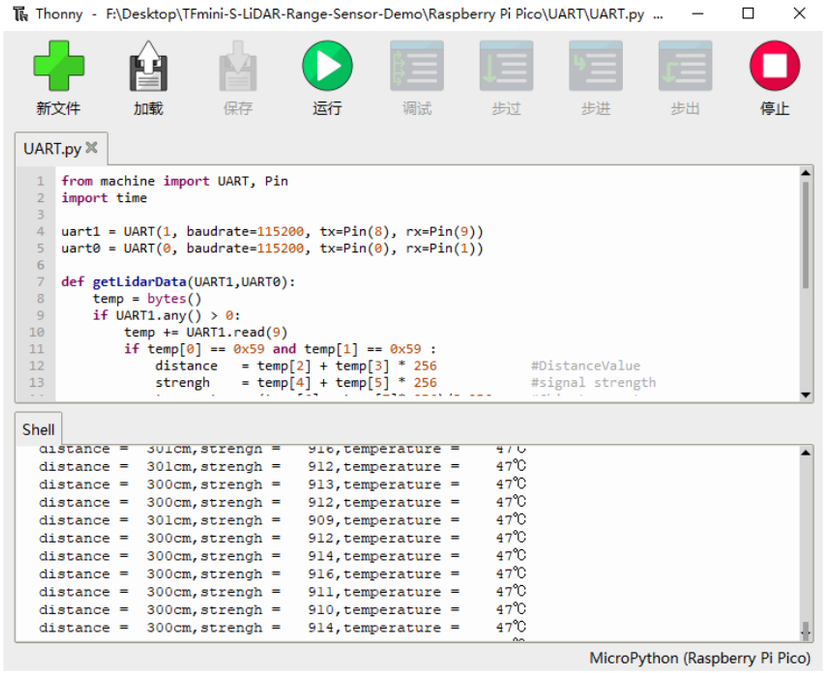
Running Example

The default mode is UART. If you switch the mode, please remember the current mode.

The following operation is to run the "UART" demo as an example. If you are using the I2C interface, please make sure you have connected and set up the corresponding interface, the operation is similar.

1. Open the downloaded demo example. (Demo example).
2. Enter "TFmini-S-LiDAR-Range-Sensor-Demo" -> "Raspberry Pi Pico".
3. Under the "Raspberry Pi Pico" folder, there are 4 files in order, each representing its own function.
4. Go to the "UART" folder and use Thonny to open UART.py (an example is to get data in UART mode).
5. Click Run in Thonny.

The debug effect is shown below:

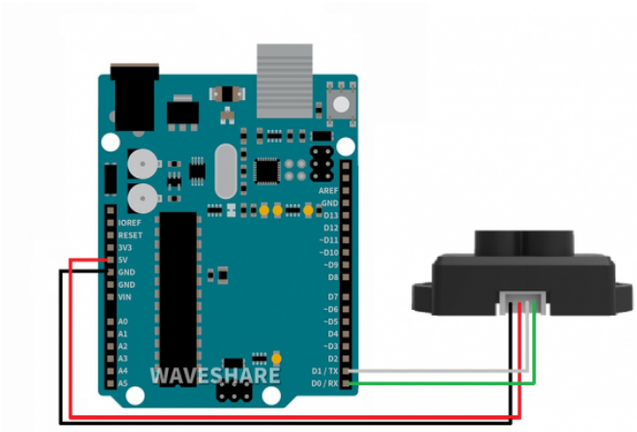


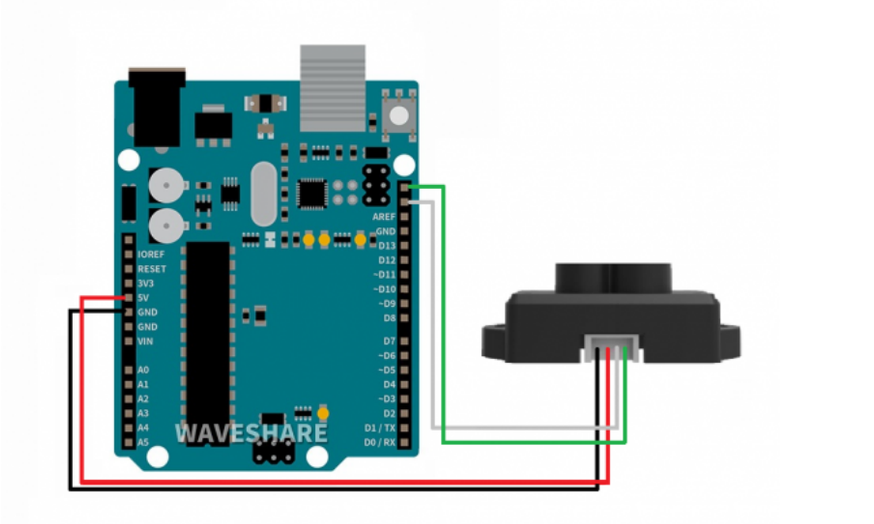
Working with Arduino

Hardware Connection

UART mode: (Connect to D0 and D1).

IIC mode: (Connect to SDA and SCL).





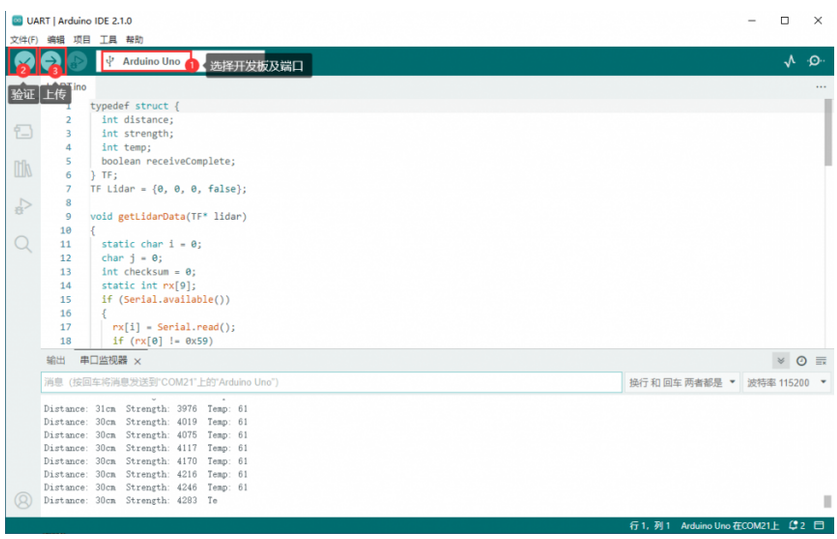
Software Debugging

The default mode is UART, if you switch modes, please remember the current mode.

The following operation is based on running the "UART" demo as an example. If you are using the I2C interface, please make sure that the corresponding interface is connected and set up, and the operation is similar.

1. Install Arduino. (Click here to download).
2. Open the downloaded demo example. (Demo example).
3. Go to "TFmini-S-LiDAR-Range-Sensor-Demo" -> "Arduino".
4. Under the "Arduino" folder, there are 4 files in order, each representing its own function.
5. Enter the "UART" file folder, and use Arduino IDE to open the UART.ino. (Example for data acquisition in UART mode).
6. Connect the Arduino board, and select the corresponding development board and port on the Arduino IDE.
7. Click to verify and click to upload.
8. As it is in UART mode, you can wait for uploading and then connect the device to the Arduino board. (Make sure that the D0 and D1 ports of the Arduino has not connected to the device when you upload.)

- Debugging effect as shown below:



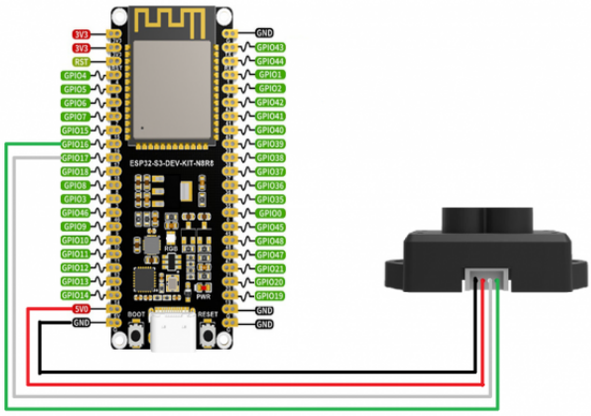
Working with ESP32

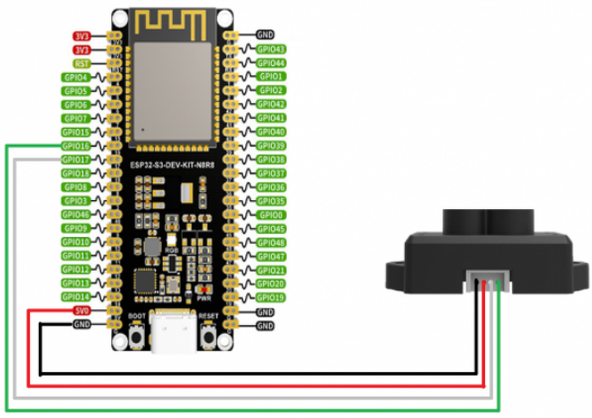
ESP32-S3-DEV-KIT-N8R8 development board is used in this demo.

Hardware Connection

UART mode: (connect to GPIO16 and GPIO17).

IIC mode: (connect to GPIO16 and GPIO17).





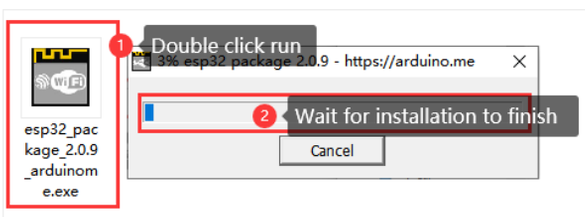
Software Debugging

Preparation

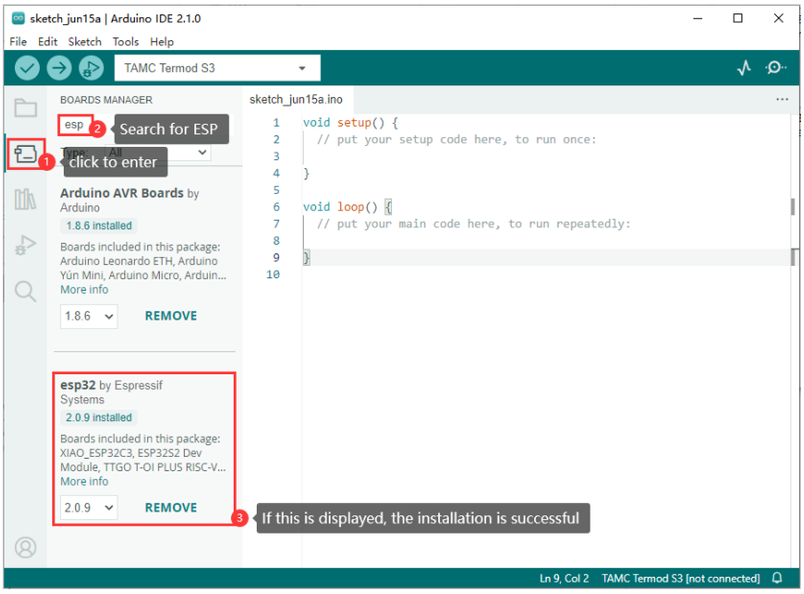
- Install Arduino. (Click here to download).
- Install the ESP32 library in the Arduino IDE. Click to download.
- Go to the file opened in point 2 and select the corresponding version of ESP32 to download.

名称	所有者	上次修...	文件大小
8266_package_3.1.1_arduino...		2023年6月25日	44.7 MB 下载
esp32_package_1.0.6_arduin...		2023年6月25日	49.6 MB
esp32_package_2.0.1_arduin...		2023年6月25日	155.2 MB
esp32_package_2.0.3_arduin...		2023年6月25日	244.4 MB
esp32_package_2.0.4_arduin...		2023年6月25日	248.8 MB
esp32_package_2.0.5_arduin...		2023年6月25日	243.1 MB
esp32_package_2.0.6_arduin...		2023年6月25日	257.8 MB
esp32_package_2.0.7_arduin...		2023年6月25日	280.3 MB
esp32_package_2.0.9_arduin...		2023年6月25日	277.2 MB
esp8266_package_3.0.1_ardu...		2023年6月25日	44.1 MB
esp8266_package_3.0.2_ard...		2023年6月25日	44.1 MB

- Double click the APP downloaded:



- Open the Arduino IDE to check whether the installation is successful.



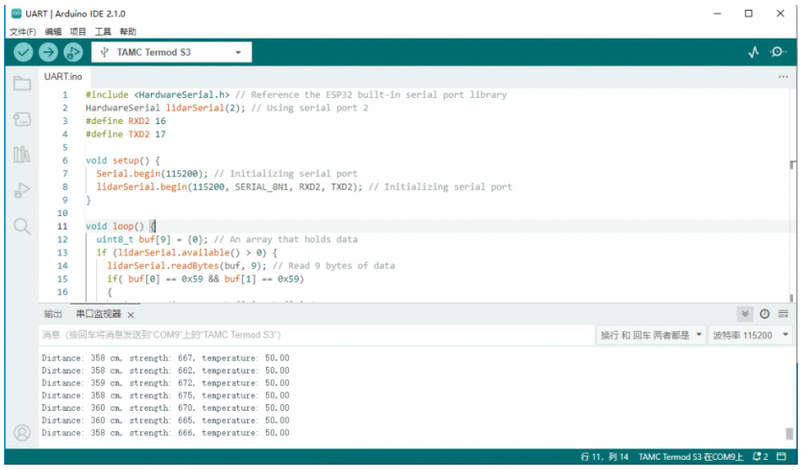
Running Example

The default mode is UART, and you need to remember it before switching to other modes.

The following operation is based on running the "UART" demo as an example. If you are using the I2C interface, please make sure that you have connected and set up the corresponding interface, and do the same.

1. Open the downloaded sample program (Demo example).
2. Go to "TFmini-S-LiDAR-Range-Sensor-Demo" -> "ESP32".
3. Under the folder "ESP32", there are 4 files in order, representing their respective functions.
4. Enter the "UART" folder, and use Arduino IDE to open UART.ino (an example is to get data in UART mode).
5. Connect the ESP32 board, and select the corresponding development board, and port on the Arduino IDE.
6. Click verify, then click upload.

- The debugging effect is shown below:



Resource

Demo

- Demo code

Wait for the output of "UART To IIC: End of configuration." or "IIC To UART: End of configuration." when switching modes, and wait for about 3 seconds. Power on again.

There are 4 files under "TFmini-S-LiDAR-Range-Sensor-Demo" folder, corresponding to 4 hardware platforms, and 4 files under each folder with the following functions:

File Folder	Description
UART	Device is in UART mode for data acquisition
UART_IIC	Device is in UART mode for mode switching
IIC	Device is in I2C mode for data acquisition
IIC_UART	Device is in I2C mode for mode switching

Software

- Thonny Python IDE (Windows V3.3.3)

- Host computer Assistant
- ESP32/8266 Library File

Related File

- TFmini-S Datasheet
- TFmini-S Product_Manual

FAQ

Question: Does TFmini-S support 3.3V or other voltage power supply?

Answer:
TFmini-S standard power supply 5V±0.1V is not supported at the moment, if you have other requirements, please contact our sales staff for customization.

Question: TFmini-S will get hot after working for a while, is it broken?

Answer:
This is the normal working condition of the product. Slight heating is normal after the chip and the board work continuously.

Question: Can TFmini-S be used with Arduino or Raspberry Pi?

Answer:
TFmini-S uses the serial communication protocol and can be used by any control board that supports serial communication.

Question: Will two TFmini-S units working together interfere with each other?

Answer:
When two TFmini-S are placed in the same direction and the light spots hit the same object and overlap, there will be no mutual interference; when more than two TFmini-S are placed in the same direction and the light spots overlap, there will be mutual interference; when two TFmini-S are working face to face, there will be serious interference.