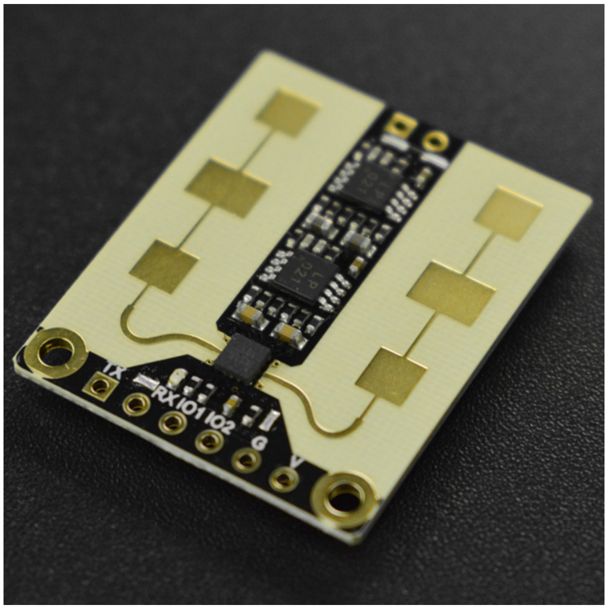
SKU:SEN0395 (https://www.dfrobot.com/product-2282.html)



(https://www.dfrobot.com/product-2282.html)

Introduction

This 24GHz millimeter-wave radar sensor employs FMCW, CW multi-mode modulation and separate transmitter and receiver antenna structure.
In working, the sensor first emits FMCW and CW radio waves to the sensing area. Next, the radio waves, reflected by all targets which are in moving, micro-moving, or extremely weak moving state in the area, are converted into electrical signals by the millimeter-wave MMIC circuit in the sensor system. After that, these signals will be sent to the processor and processed through the related signal and data algorithms. Then, the target information can be solved out.
The millimeter-wave radar can sense the human presence, stationary and moving people within the detection area. Moreover, it can even detect static or stationary human presence such as a sleeping person. There are two ways provided to output detection result: serial port and I/O port switch quantity. Besides that, the sensor module features strong sensing reliability, high sensitivity, small size, easy to be used or embedded in applications.

Features

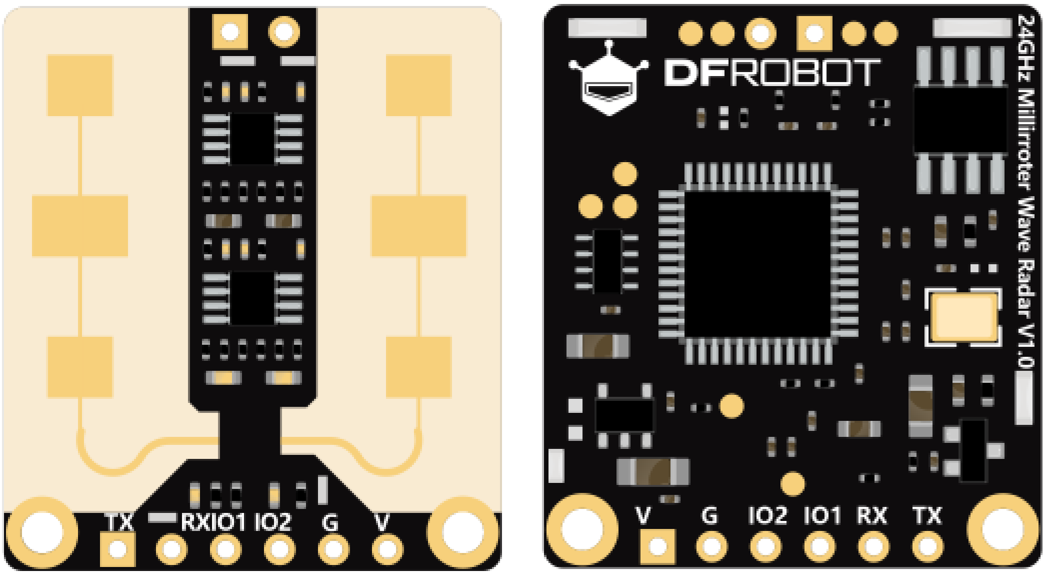
- Human presence sensing: sense if there is human body in areas
- I/O port switch quantity input and output control

- Serail port input and output control
- Strong anti-interference ability, not to be affected by snow, haze, temperature, humidity, dust, light, noise, etc.

Specification

- Power Supply: 3.6~5V
- Operating Current: 90mA
- Detection Distance: 9m
- Equivalent Transmit Power: 13-15dBM
- Beam Angle: 100×40°
- Modulation Mode: FMCW, CW
- Operating Frequency: 24GHz
- Operating Temperature: -40~85℃
- Buad Rate: 115200
- Dimension: 24×28mm/0.94×1.10”

Board Overview



The interface definition and function description of this millimeter wave are shown in Table 1.

Num	Label	Description
1	UART Tx	Sensor UART Transmiting

2	UART Rx	Sensor UART Receiving
3	GPIO1	Universal Input and Output

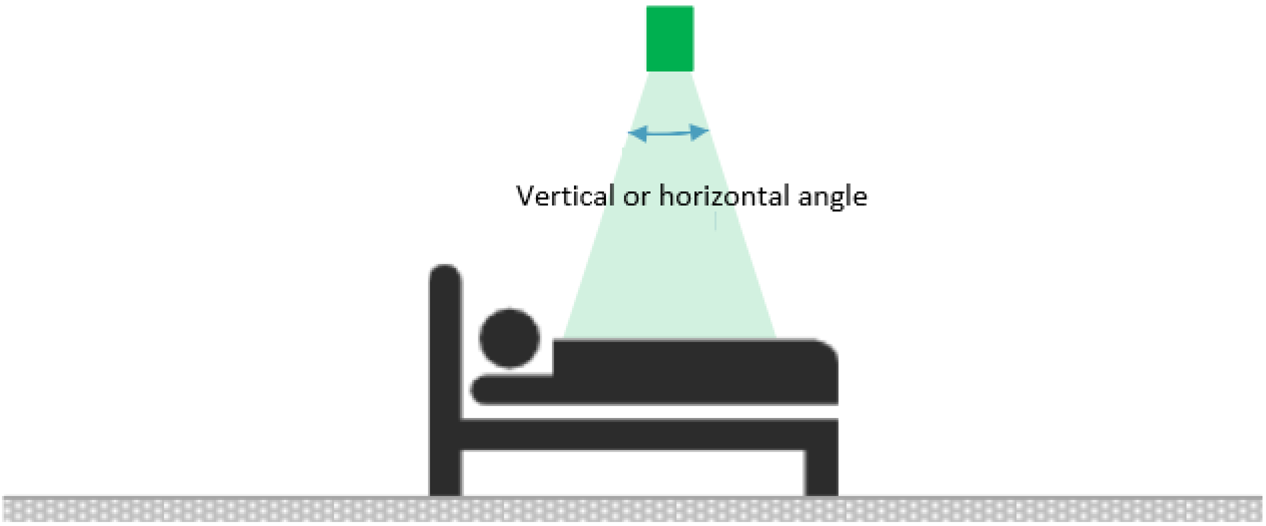
Num	Label	Description
4	GPIO2	Universal Input and Output(By default, it outputs high when people presence detected, otherwise, output low)
5	GND	Ground
6	VCC	Power +
7	NC	Reserved, left floating
8	NC	Reserved, left floating

The two I/O ports can be used for level trigger input and output applications; The 1 UART port can used to configure input and sense result.

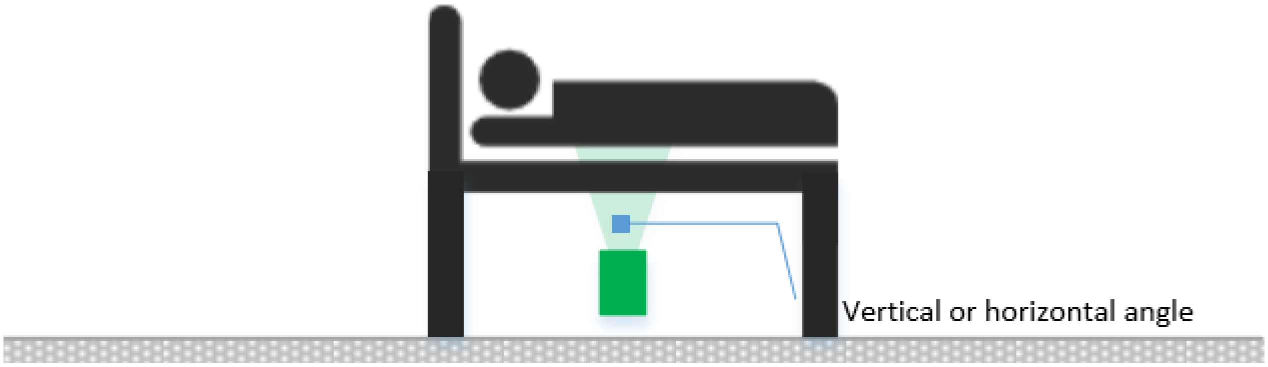
Installtion

The millimeter wave human sensor is strict on the installation method, improper installation will affect the performance and function of the sensor. Commonly-used installation methods for modules include top installation, bottom installation, horizontal installation and downward tilt installation.

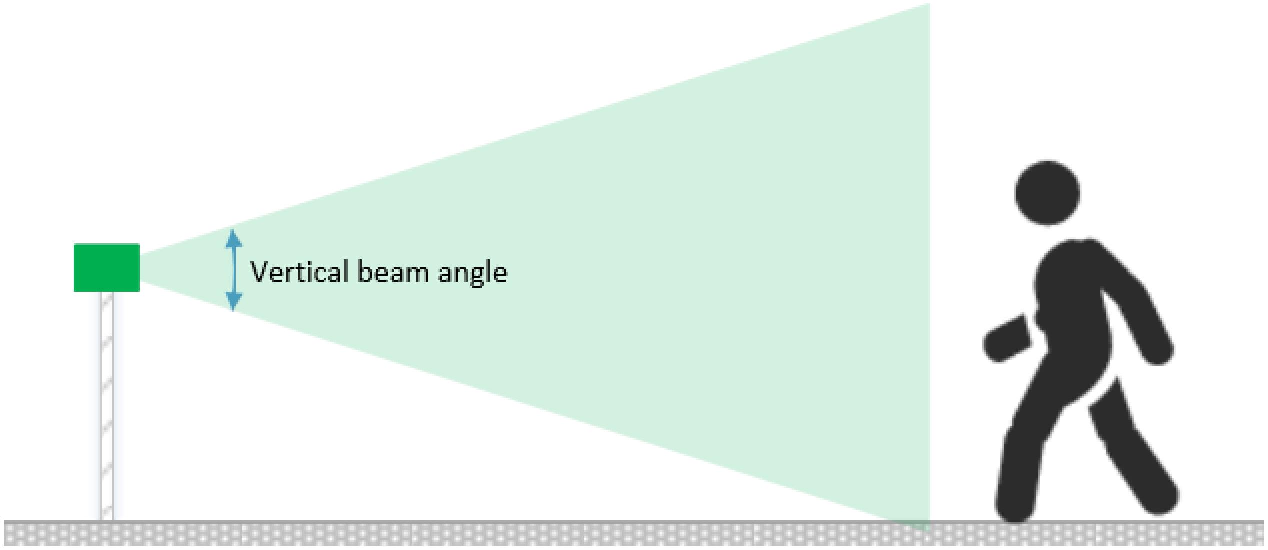
1. Top installation



2. Bottom installation



3. Horizontal installation



Instructions

The module adopts serial communication protocol and uses ASCII code string as command interation and data interaction.

- Serail communication configuration: 115200 baud rate, 1 stop bit, 8 data bits, no parity bit, no flow control
- Command interaction: ASCII code string format, ending with a new line; separated by space between commands to parameters, parameters to parameters.
- When configuring the working parameters of the sensor, the sensor must be in a halted state; after the parameter configuration is completed, a save parameter command must be sent to save the changed parameters in the non-power-loss memory.
- If using this module as a trigger switch, that is, only use the I/O port of the module, and at

- If using this module as a trigger switch, that is, only use the I/O port of the module, and at the same time, when the configuration parameters meet the performance and functional requirements, the serial port is not necessary, and also users do not need to care about the communication protocol.

Output Information Configuration

Description: The default configuration of the sensor is to output the sensing results only in ASCII code string format, and modification is not supported temporarily.

Human body induction result output: ASCII code string format

Description:

- The string information starts with "$" and ends with "*". Multiple parameters are separated by ",", and placeholder parameters(reserved) are replaced by spaces.
- The sensing result of the "Human Sensor" is output at a data refresh rate of 1Hz by default.

Response: `$JYBSS,par1,par2,par3,par4*`

Parameter Term	Description
par1	Induction result: 0: No one 1: People presence detected(stationary or moving)
par2	Placeholder parameter: reserved, replace with spaces
par3	Placeholder: reserved, replace with spaces
par4	Placeholder: reserved, replace with spaces

Example	Command
People presence detected (moving or stationary)	$JYBSS,1, , , *
No one detected	$JYBSS,0, , , *

Sensor Detection Area Configuration

Description:

- The module configures the sensing area as 128 equal parts by default, each equal part is about 15cm. The configuration parameters are selected from 0~127, as the index of the distance value in the sensing area.

- The sensing area allows configuration of 4 segments, the configuration values of the 4 segments must be configured in order from small to large, and cannot contain each other.

Command: `detRangeCfg par1 parA_s parA_e parB_s parB_e parC_s parC_e parD_s parD_e`

Parameter Term	**Description**
par1	Retained, constant -1
parA_s parA_e	The 1st segment of the sensing area configuration value: parA_s: the starting value index of the sensing area distance(≥0) parA_e: the ending value index of the sensing area distance
parB_s parB_e	The 2nd segment of the sensing area configuration value (the index must start after the 1st segment sensing area configuration index): parB_s: the starting value index of the sensing area distance parB_e: the ending value index of the sensing area distance
parC_s parC_e	The 3rd segment of the sensing area configuration value (the index must start after the 2nd segment sensing area configuration index): parC_s: the starting value index of the sensing area distance parC_e: the ending value index of the sensing area distance
parD_s parD_e	The 4th segment of the sensing area configuration value (the index must start after the 3rd segment sensing area configuration index): parD_s: the starting value index of the sensing area distance parD_e: the ending value index of the sensing area distance(≤127)

Note: Multiply the start or end value index of the sensing area distance by 15cm, it represents the start or end distance value.

Response	**Description**
Done	The command was executed successfully
Error	Command execution failed

Example	**Command**
(Default configuration) Sensing distance: "0m to 3m"	

(Default configuration) Sensing distance: 0m to 3m (0m=0 * 0.15cm, 3m=20 * 0.15cm)	detRangeCfg -1 0 20

Example	Command
Sensing distance: "1.5m to 3m" (1.5m=10 * 0.15cm, 3m=20 * 0.15cm)	detRangeCfg -1 10 20
Sensing distance: "1.5m to 3m" "7.5m to 12m" (1.5m=10 * 0.15cm, 3m=20*0.15cm) (7.5m=50 * 0.15cm， 12m=80 * 0.15cm)	detRangeCfg -1 10 20 50 80
Sensing distance: "1.5m to 3m" "7.5m to 12m" "13.5m to 15m" (1.5m=10 * 0.15cm， 3m=20 * 0.15cm) (7.5m=50 * 0.15cm, 12m=80 * 0.15cm) (13.5m=90 * 0.15cm, 15m=100 * 0.15cm)	detRangeCfg -1 10 20 50 80 90 100
Sensing distance: "1.5m to 3m" "7.5m to 12m" "13.5m to 15m" "15.75m to 16.5m" (1.5m=10 * 0.15cm, 3m=20 * 0.15cm) (7.5m=50 * 0.15cm, 12m=80 * 0.15cm) (13.5m=90 * 0.15cm, 15m=100 * 0.15cm) (15.75m=105 * 0.15cm, 16.5m=110 * 0.15cm)	detRangeCfg -1 10 20 50 80 90 100 105 110

Sensor Output Delay Configuration

Description: configure, when the sensor detects a target, the delay time for the output of the sensing result; after the target disappears, the delay time for the output of the sensing result. By default, when a target detected, the delayed output time is 2.5s; after the target disappears, the delayed output time is 10s.

Command: `outputLatency par1 par2 par3`

Parameter Term	Description
par1	Retained, constant -1
par2	When the target is detected, the delay time for output of sensing results: Value range: 0~65535, unit: 25ms

par3	After the target disappears, the delay time for output of sensing results: Value range: 0~65535, unit: 25ms

Response	Description
Done	The command was executed successfully
Error	Command execution failed

Example	Command
(Default configuration) The target is detected, the delay time is 2.5 seconds. The target disappears, the delay time is 10 seconds.	outputLatency -1 100 400
The target is detected, the delay time is 5 seconds. The target disappears, the delay time is 20 seconds.	outputLatency -1 200 800

Sensor Start-up Mode Configuration

Description: Configure the sensor to start running immediately after power-on or wait for the start command after power-on. The former is set by default.

Command: `sensorCfgStart par1`

Parameter Term	Description
par1	Enable the sensor to start running immediately after power-on: 0: Prohibit the sensor to start running immediately after power-on. The sensorStart command is required to start running (default value) 1: Enable the sensor to start running immediately after power-on. No sensorStart command is required to start running

Response	Description
Done	The command was executed successfully
Error	Command execution failed

Example	Example
Prohibit the sensor to start running immediately after power on	sensorCfgStart 0

	0
(Default configuration) Enable the sensor to start running immediately after power on	sensorCfgStart 1

Sensor Start Control

Description:

**- When the sensor is in an unstarted state and there are no set parameters to be saved, start the sensor to run.

- Through the sensorCfgStart command, when the sensor is configured to start running immediately after power-on, there is no need to execute the sensorStart command to start running.**

Command: `sensorStart`

Parameter Term	Description
Null	No parameters

Response	Description
Done	The command was executed successfully
Error	Command execution failed

Sensor Reset Control

Description: Reset the sensor by software

Command: `resetSystem`

Parameter Term	Description
Null	No parameters

Response	Description
Error	Command execution failed
No response string	After the command is executed successfully, the sensor will be reset directly, so there is no response string

Sensor Stop Control

Description: Stop the sensor when it is running.

Commands: `sensorStop`

Parameter Term	Description
Null	No parameter

Response	Description
Done	The command was executed successfully
Error	Command execution failed, the sensor is not in
operating status	

Configuration Save Parameter

Description: When the sensor parameters are reconfigured through the UART and have not been saved, this command saves the new configuration parameters to the sensor Flash

Command: `saveCfg par1 par2 par3 par4`

Parameter Term	Description
par1	Fixed value: 0x45670123
par2	Fixed value: 0xCDEF89AB
par3	Fixed value: 0x956128C6
par4	Fixed value: 0xDF54AC89

Response	Description
Done	The command was executed successfully
Error	Command execution failed (If there are no parameters to save, the command will fail to execute)

Save the configuration parameters to the non-power-loss memory unit, command: `saveCfg 0x45670123 0xCDEF89AB 0x956128C6 0xDF54AC89`

Factory reset

Description: Restore the current configuration parameter value of the sensor to the factory default value.

Command: `factoryReset par1 par2 par3 par4`

Parameter Term	Description
par1	Fixed value: 0x45670123
par2	Fixed value: 0xCDEF89AB
par3	Fixed value: 0x956128C6
par4	Fixed value: 0xDF54AC89

Response	Description
Done	The command was executed successfully
Error	Command execution failed

Restore configuration parameters to factory default values, command: `factoryReset 0x45670123 0xCDEF89AB 0x956128C6 0xDF54AC89`

Tutorial for Arduino

Requirements

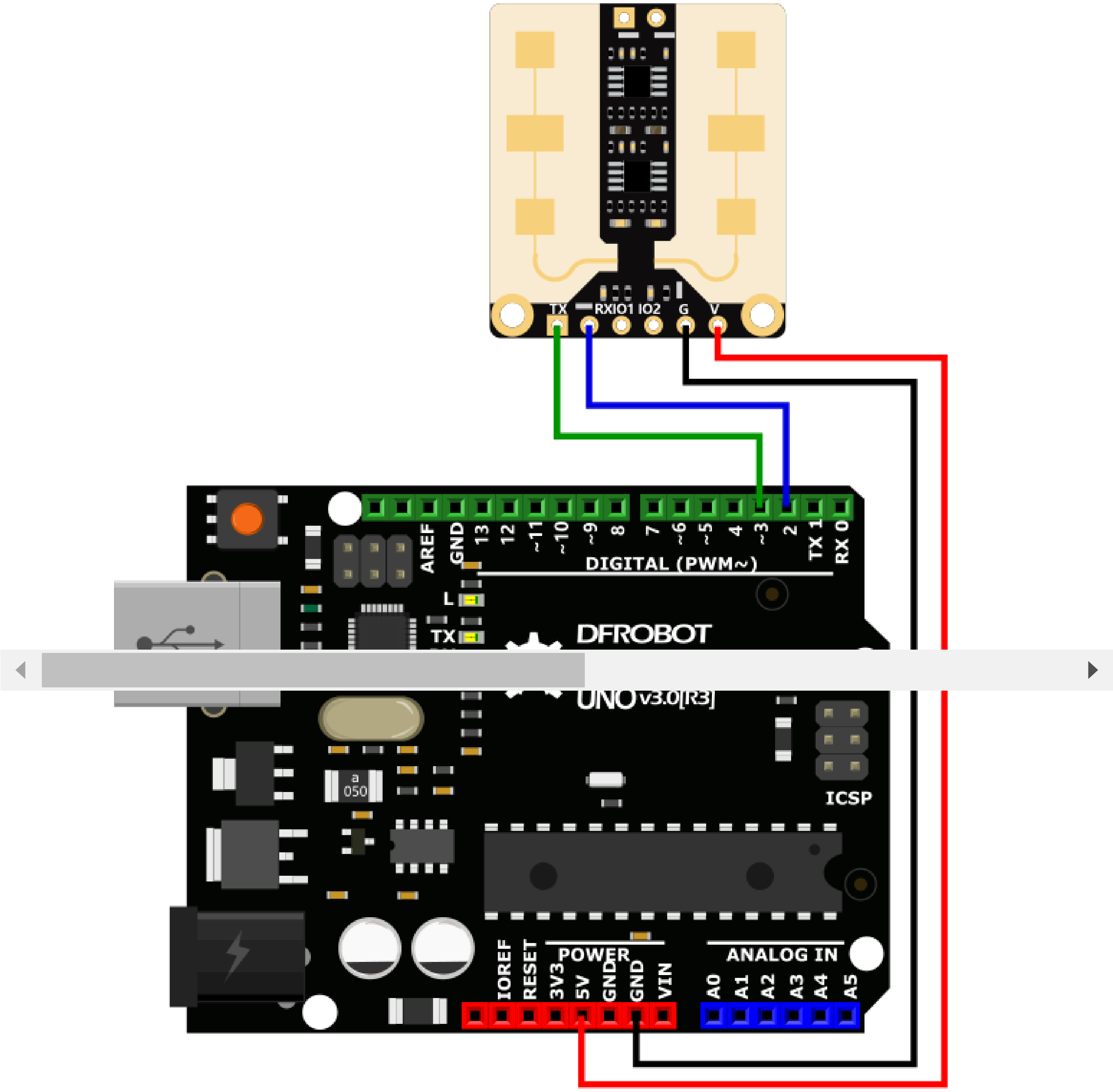
- **Hardware**
  - DFRduino UNO R3 (https://www.dfrobot.com/product-838.html) (or similar) x 1
  - mmWave Radar x1
  - M-M/F-M/F-F Jumper wires

- **Software**
  - Arduino IDE (https://www.arduino.cc/en/Main/Software)
  - Download and install the **DFRobot_mmWave Radar Library** (https://github.com/DFRobotdl/DFRobot_mmWave_Radar/archive/refs/heads/master.zip) (About how to install the library? (https://www.arduino.cc/en/Guide/Libraries#.UxU8mdzF9H0))

API Function List

```
/**
      @brief Constructor
      @param Stream  Software serial port interface
    */
    DFRobot_mmWave_Radar(Stream *s);

    /**
      @brief  Configure sensor detection area
      @param parA_s The sensing area distance starting value of the first segment, unit: 
      @param parA_e The sensing area distance ending value of the first segment, unit: m(
    */
    void DetRangeCfg(float parA_s, float parA_e);


    /**
      @brief  Configure sensor detection area
      @param parA_s The sensing area distance starting value of the first segment, unit: 
      @param parA_e The sensing area distance ending value of the first segment, unit: m(
      @param parB_s The sensing area distance starting value of the second segment, unit:
      @param parB_e The sensing area ending value of the second segment, unit: m(Must be 
    */
    void DetRangeCfg(float parA_s, float parA_e, float parB_s, float parB_e);


    /**
      @brief  Configure sensor detection area
      @param parA_s The sensing area distance starting value of the first segment, unit: 
      @param parA_e The sensing area distance ending value of the first segment, unit: m 
      @param parB_s The sensing area distance starting value of the second segment, unit:
      @param parB_e The sensing area distance ending value of the second segment, unit: m
      @param parC_s The sensing area distance starting value of the third segment, unit: 
      @param parC_e The sensing area distance ending value of the third segment, unit: m(
    */
    void DetRangeCfg(float parA_s, float parA_e, float parB_s, float parB_e, float parC_s


    /**
      @brief  Configure sensor detection area
      @param parA_s The sensing area distance starting value of the first segment, unit: 
      @param parA_e The sensing area distance ending value of the first segment, unit: m(
      @param parB_s The sensing area distance starting value of the second segment, unit:
      @param parB_e The sensing area distance ending value of the second segment, unit: m
      @param parB_e The sensing area distance ending value of the second segment, unit: m
      @param parC_s The sensing area distance starting value of the third segment, unit:
      @param parC_e The sensing area distance ending value of the third segment, unit: m(
      @param parD_s The sensing area distance starting value of the fourth segment, unit:
      @param parD_e The sensing area ditance ending value of the fourth segment, unit: m(

    */
    void DetRangeCfg(float parA_s, float parA_e, float parB_s, float parB_e, float parC_s


    /**
      @brief  Read whether there is people or object moving in the sensing area
      @return  Returning true means that there is people or animal moving in the detectio
    */
    bool readPresenceDetection(void);


    /**
      @brief  Configure sensor output delay time
      @param par1 When a target detected, delay the output time of sensing result, range:
      @param par2 When the target disappears, delay the output time of sensing result, ra
    */
    void OutputLatency(float par1, float par2);


    /**
      @brief  Restore the sensor current configuration to the factory settings.
    */
    void factoryReset(void);
```

Connection Diagram



mmWave Radar	Arduino Uno
VCC	5V
GND	GND

RX	D2
TX	D3

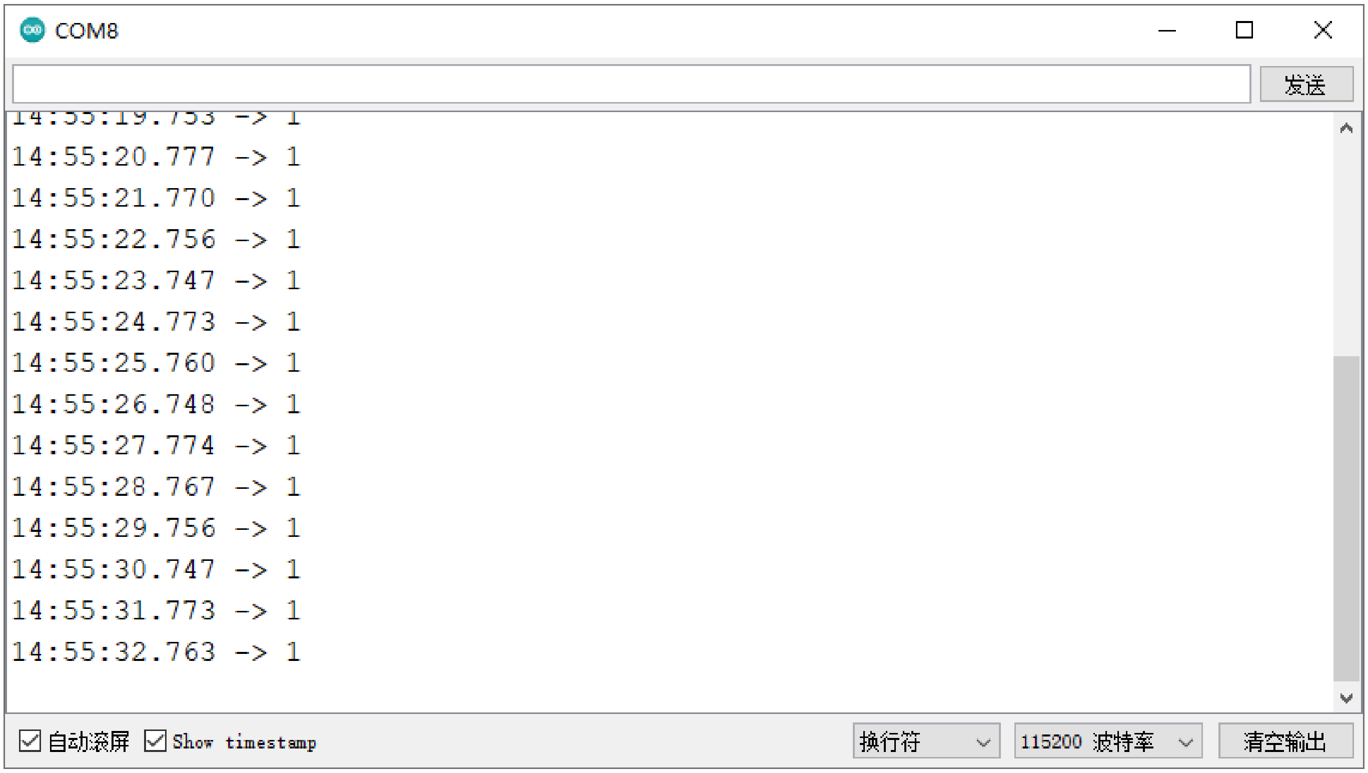
Copy the following code to your Arduino IDE and upload it.

Sample Code

```
/*!
   @file DFRobot_mmWave_Radar.ino
   @ Read whether there is people or object moving in the detection range of the sensor.
   @ The sensor detection range and output delay time can be configured. Also you can res
   @n Experimental phenomenon: When the sensor starts successfully, 0 or 1 will be printe
   @ 0 means that there is no human or object moving in sensing area, 1 means the oppposi
   @copyright   Copyright (c) 2010 DFRobot Co.Ltd (http://www.dfrobot.com)
   @licence     The MIT License (MIT)
   @author [huyujie](yujie.hu@dfrobot.com)
   @version  V1.0
   @date  2020-3-25
   @https://github.com/DFRobot
*/


#include <SoftwareSerial.h>

#include "DFRobot_mmWave_Radar.h"

SoftwareSerial mySerial(3, 2);
DFRobot_mmWave_Radar sensor(&mySerial);

int ledPin = 13;

void setup()
{
  Serial.begin(115200);
  mySerial.begin(115200);
  pinMode(ledPin, OUTPUT);

  sensor.factoryReset();    //Restore to the factory settings
  sensor.DetRangeCfg(0, 9);    //The detection range is as far as 9m
  sensor.OutputLatency(0, 0);
}

void loop()
{
  int val = sensor.readPresenceDetection();
  digitalWrite(ledPin, val);
```

```
    Serial.println(val);
  }
```

Expected Results



```
14:55:19.753 -> 1
14:55:20.777 -> 1
14:55:21.770 -> 1
14:55:22.756 -> 1
14:55:23.747 -> 1
14:55:24.773 -> 1
14:55:25.760 -> 1
14:55:26.748 -> 1
14:55:27.774 -> 1
14:55:28.767 -> 1
14:55:29.756 -> 1
14:55:30.747 -> 1
14:55:31.773 -> 1
14:55:32.763 -> 1
```

☑ 自动滚屏 ☑ Show timestamp 换行符 115200 波特率

FAQ

If you have any questions about using this product, please check the **FAQ list** (https://www.dfrobot.com/forum/topic/315484) for that product for a corresponding solution. And for any questions, advice or cool ideas to share, please visit the **DFRobot Forum** (https://www.dfrobot.com/forum/).

More Documents

Get **mmWave Radar-Human Presence Detection** (https://www.dfrobot.com/product-2282.html) from DFRobot Store or **DFRobot Distributor**. (https://www.dfrobot.com/index.php?route=information/distributorslogo)

Turn to the Top