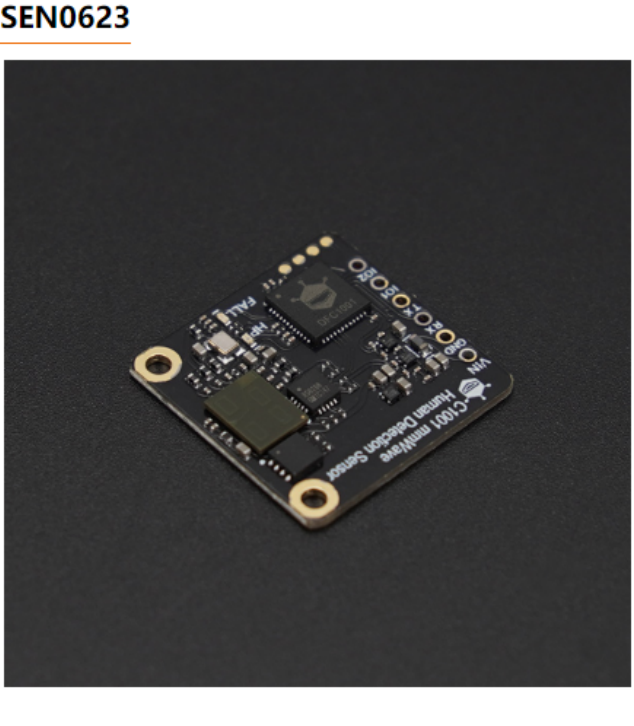
Product Introduction
Technical Specifications
Dimension Diagram
Interface Definition
Installation Method
Demonstration Routine
FAQ

SKU:SEN0623

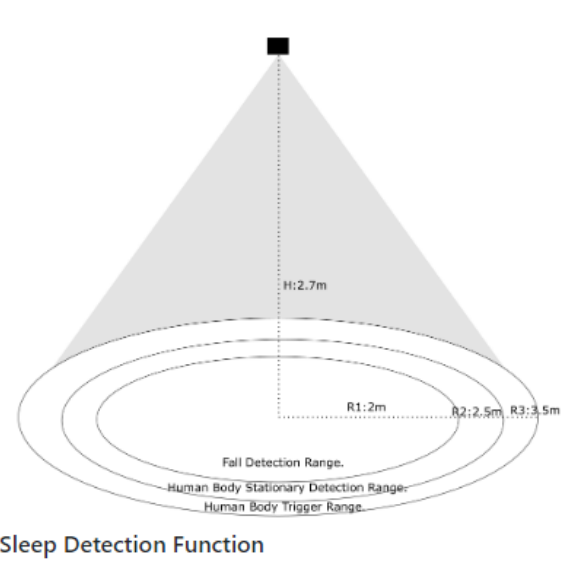
Product Introduction



The C1001 mmWave Human Detection Sensor is a millimeter wave radar that operates at a frequency of 60GHz. Unlike the 24GHz millimeter wave, the C1001 mmWave Human Detection Sensor can achieve more detailed detection functions.

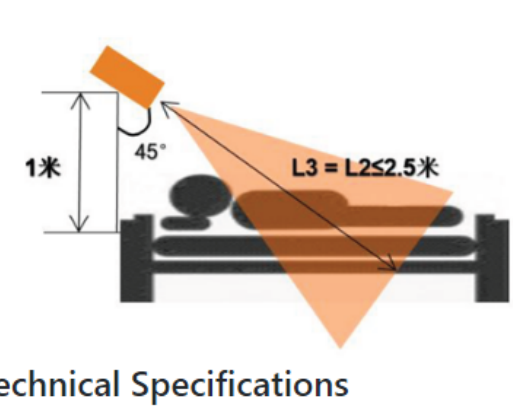
Fall Detection Function

The C1001 mmWave Human Detection Sensor can recognize human postures through point cloud imaging algorithms, accurately detect lying flat, and precisely detect life. It accurately reports the fallen state, stay time, and static body stay state.



Sleep Detection Function

The C1001 mmWave Human Detection Sensor can realize the perception of human biological existence and human motion perception, continuously record the existence of the human body, and make real-time judgments on the target's sleep state and breathing heart rate according to the changes in body movement amplitude and breathing heart rate during sleep. After a sleep process, it outputs sleep scores and combines the output of related sleep parameters into health detection applications.

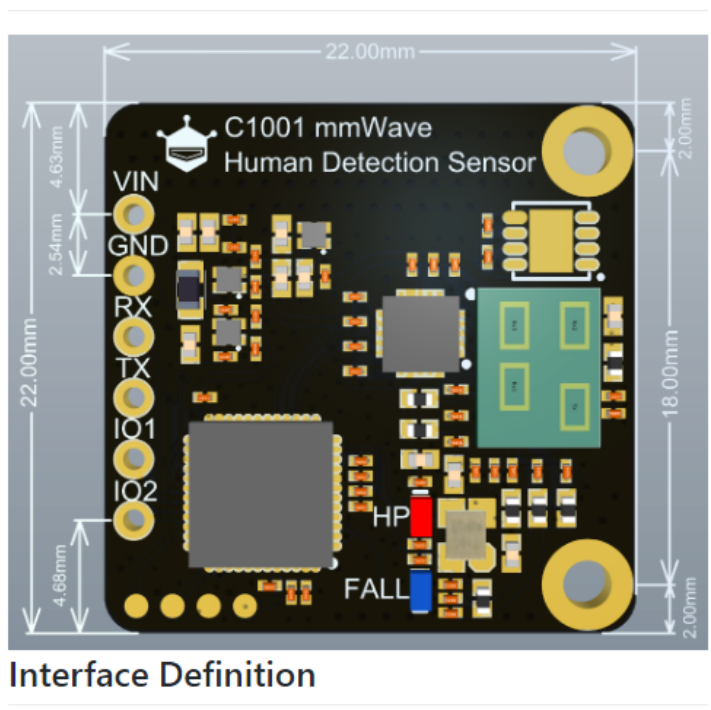


Technical Specifications

- Working voltage: 5V
- Working current: ≤100mA
- Working frequency: 61~61.5GHz
- Transmission power: 6dBm
- Farthest detection distance: 11m
- Radar detection angle: 100×100 degrees
- Sleep detection distance (chest): 0.4-2.5m
- Breathing heartbeat detection distance (chest): 0.4-1.5m
- Breathing measurement range: 10-25 times/minute
- Heartbeat measurement range: 60-100 times/minute
- Working temperature: -20~60℃

`Note: This product is not a professional medical instrument and cannot be used as an auxiliary accessory for diagnosis and treatment.`

Dimension Diagram



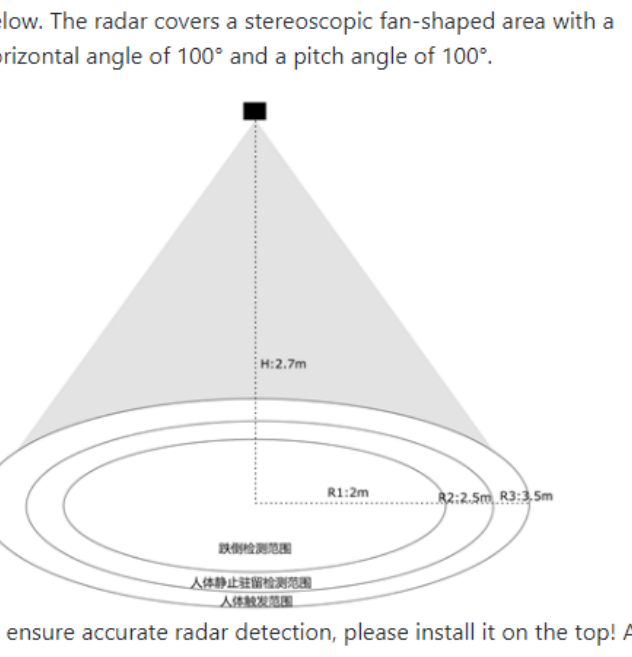
Interface Definition

Definition	Explanation
VIN	Power Supply
GND	Ground
RX	Sensor Serial Reception
TX	Sensor Serial Transmission
IO1	Human Presence Level Output (3.3V)
IO2	Fall Status Level Output (3.3V)

Installation Method

Top Installation for Fall Mode

The radar module beam coverage range is as shown in the figure below. The radar covers a stereoscopic fan-shaped area with a horizontal angle of 100° and a pitch angle of 100°.

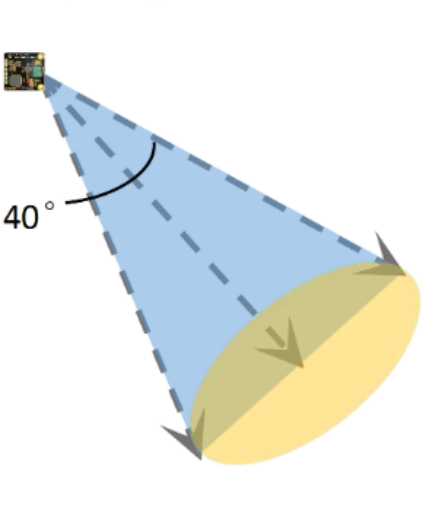


To ensure accurate radar detection, please install it on the top! As shown in the figure below.

`Note: Avoid easily swaying objects such as exhaust fans/metal blinds/blackout curtain coatings in the room`

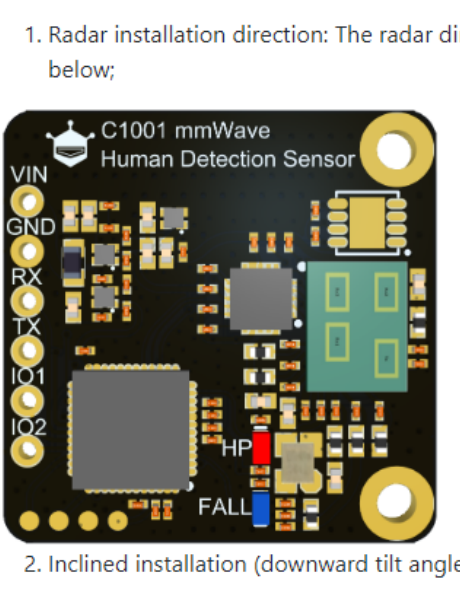
Sleep Mode

The radar beam coverage range is as shown in the figure below. The radar covers a stereoscopic fan-shaped area with a horizontal angle of 40° and a pitch angle of 40°.

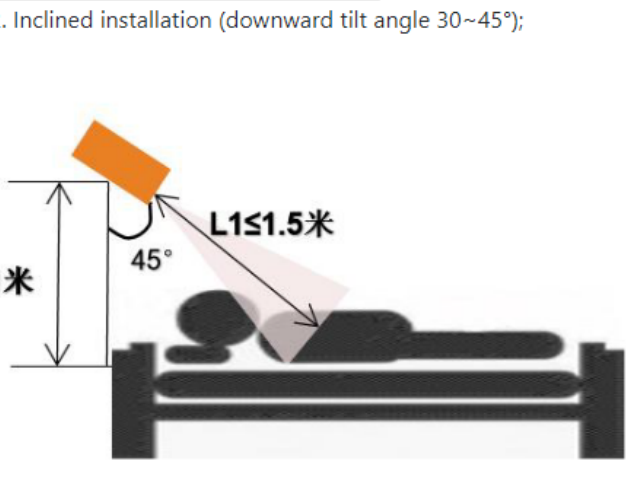


Within the radar detection range, in order to meet the stable detection of human presence and sleep breathing and heartbeat, there are certain installation requirements for radar installation.

1. Radar installation direction: The radar direction is required as below;



2. Inclined installation (downward tilt angle 30~45°);



Demonstration Routine

Presence Information & Respiration Rate & Heart Rate Acquisition

Preparation

- Hardware

  C1001 Millimeter Wave Human Detection Sensor

  ESP32-S3-DevKitC-1 Development Board, or other ESP32-based development boards.

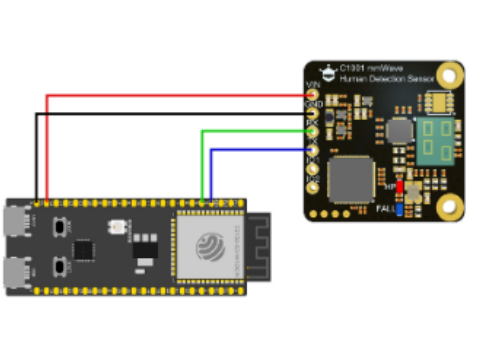
- Software

  Arduino IDE, Click to download Arduino IDE

  DFRobot_HumanDetection library, Click to download DFRobot_HumanDetection library

  How to install library files, Click the link

Wiring Diagram



Millimeter Wave	ESP32-S3
VIN	5V
GND	GND
RX	IO5
TX	IO4

Installation Method

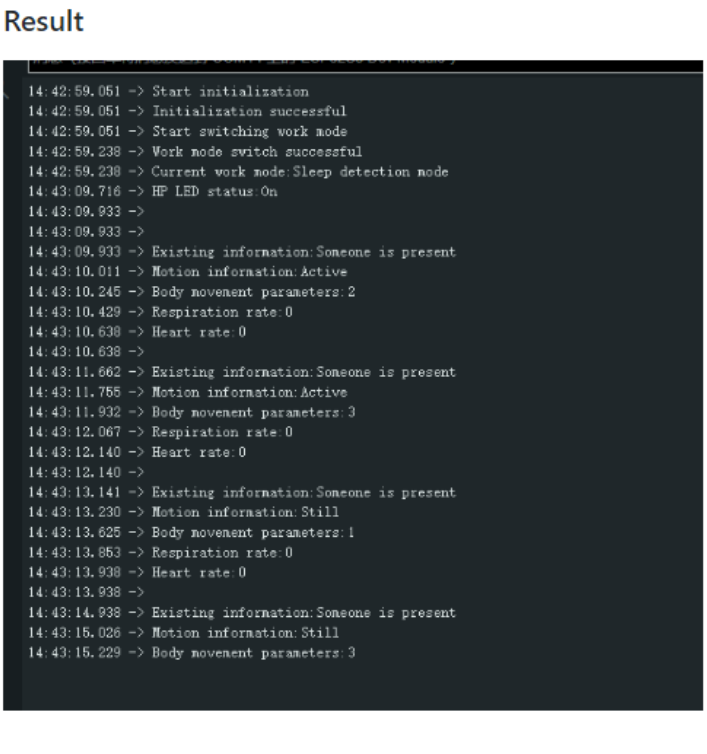
There are no special installation requirements for human presence detection, both top and side installations are possible. For respiration rate and heart rate, the sensor should be placed 1.5m in front of the human body and the sensor should be directly facing the chest of the detected person.

Sample Code

Copy the following code into your Arduino IDE and upload.

```
/**!
 * @file sleep.ino
 * @brief This is an example of the C1001 mmWave Human Detecti
 * @copyright  Copyright (c) 2010 DFRobot Co.Ltd (http://www.d
 * @license     The MIT License (MIT)
 * @author [tangjie](jie.tang@dfrobot.com)
 * @version  V1.0
 * @date  2024-06-03
 * @url https://github.com/DFRobot/DFRobot_HumanDetection
 */
#include "DFRobot_HumanDetection.h"

DFRobot_HumanDetection hu(&Serial1);

void setup() {
  Serial.begin(115200);
  Serial1.begin(115200, SERIAL_8N1, 4, 5);

  Serial.println("Start initialization");
  while (hu.begin() != 0) {
    Serial.println("init error!!!");
    delay(1000);
  }
  Serial.println("Initialization successful");

  Serial.println("Start switching work mode");
  while (hu.configWorkMode(hu.eSleepMode) != 0) {
    Serial.println("error!!!");
    delay(1000);
  }
  Serial.println("Work mode switch successful");

  Serial.print("Current work mode:");
  switch (hu.getWorkMode()) {
    case 1:
      Serial.println("Fall detection mode");
      break;
    case 2:
      Serial.println("Sleep detection mode");
      break;
    default:
      Serial.println("Read error");
  }

  hu.configLEDLight(hu.eHPLed, 1);  // Set HP LED switch, it w
  hu.sensorRet();                   // Module reset, must perf

  Serial.print("HP LED status:");
  switch (hu.getLEDLightState(hu.eHPLed)) {
    case 0:
      Serial.println("Off");
      break;
    case 1:
      Serial.println("On");
      break;
    default:
      Serial.println("Read error");
  }

  Serial.println();
  Serial.println();
}

void loop() {
  Serial.print("Existing information:");
  switch (hu.smHumanData(hu.eHumanPresence)) {
    case 0:
      Serial.println("No one is present");
      break;
    case 1:
      Serial.println("Someone is present");
      break;
    default:
      Serial.println("Read error");
  }

  Serial.print("Motion information:");
  switch (hu.smHumanData(hu.eHumanMovement)) {
    case 0:
      Serial.println("None");
      break;
    case 1:
      Serial.println("Still");
      break;
    case 2:
      Serial.println("Active");
      break;
    default:
      Serial.println("Read error");
  }

  Serial.printf("Body movement parameters:%d\n", hu.smHumanDat
  Serial.printf("Respiration rate:%d\n", hu.getBreatheValue())
  Serial.printf("Heart rate:%d\n", hu.gitHeartRate());
  Serial.println();
  delay(1000);
}
```

Result



```
14:42:59.051 -> Start initialization
14:42:59.051 -> Initialization successful
14:42:59.051 -> Start switching work mode
14:42:59.238 -> Work mode switch successful
14:42:59.238 -> Current work mode:Sleep detection mode
14:43:09.716 -> HP LED status:On
14:43:09.933 ->
14:43:09.933 ->
14:43:09.933 -> Existing information:Someone is present
14:43:10.011 -> Motion information:Active
14:43:10.245 -> Body movement parameters:2
14:43:10.429 -> Respiration rate:0
14:43:10.638 -> Heart rate:0
14:43:10.638 ->
14:43:11.662 -> Existing information:Someone is present
14:43:11.788 -> Motion information:Active
14:43:11.932 -> Body movement parameters:3
14:43:12.067 -> Respiration rate:0
14:43:12.140 -> Heart rate:0
14:43:12.140 ->
14:43:13.141 -> Existing information:Someone is present
14:43:13.230 -> Motion information:Still
14:43:13.625 -> Body movement parameters:1
14:43:13.863 -> Respiration rate:0
14:43:13.938 -> Heart rate:0
14:43:13.938 ->
14:43:14.938 -> Existing information:Someone is present
14:43:15.026 -> Motion information:Still
14:43:15.229 -> Body movement parameters:3
```

Sleep Information Acquisition

Preparation

- Hardware

  C1001 Millimeter Wave Human Detection Sensor

  ESP32-S3-DevKitC-1 Development Board, or other ESP32-based development boards.

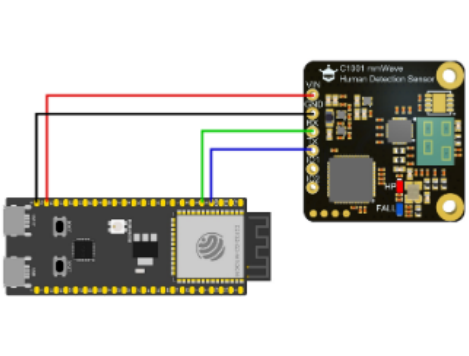
- Software

  Arduino IDE, Click to download Arduino IDE

  DFRobot_HumanDetection library, Click to download DFRobot_HumanDetection library

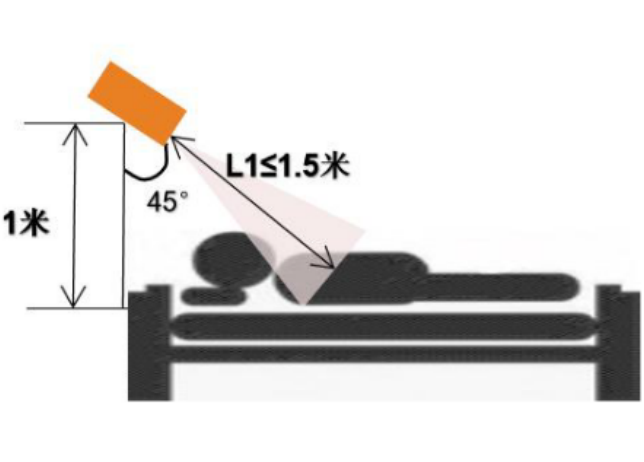
  How to install library files, Click the link

Wiring Diagram



Millimeter Wave	ESP32-S3
VIN	5V
GND	GND
RX	IO5
TX	IO4

Installation Method

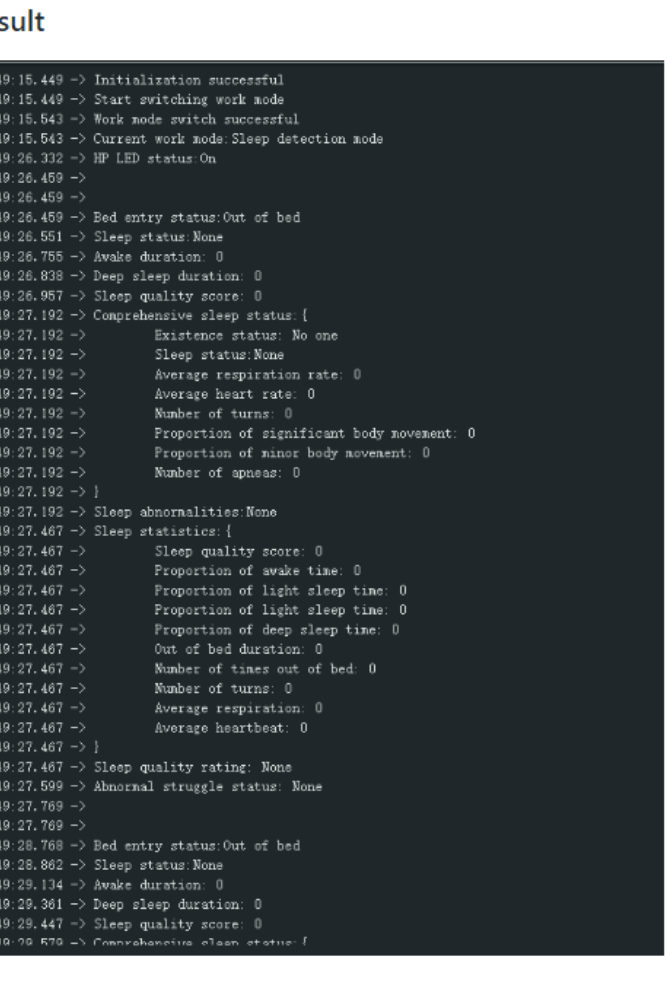


Sample Code

Copy the following code into your Arduino IDE and upload.

```
/**!
 * @file sleep.ino
 * @brief This is an example of sleep detection using human mi
 * @copyright  Copyright (c) 2010 DFRobot Co.Ltd (http://www.d
 * @license     The MIT License (MIT)
 * @author [tangjie](jie.tang@dfrobot.com)
 * @version  V1.0
 * @date  2024-06-03
 * @url https://github.com/DFRobot/DFRobot_HumanDetection
 */
#include "DFRobot_HumanDetection.h"

DFRobot_HumanDetection hu(&Serial1);

void setup() {
  Serial.begin(115200);
  Serial1.begin(115200, SERIAL_8N1, 4, 5);

  Serial.println("Start initialization");
  while (hu.begin() != 0) {
    Serial.println("init error!!!");
    delay(1000);
  }
  Serial.println("Initialization successful");

  Serial.println("Start switching work mode");
  while (hu.configWorkMode(hu.eSleepMode) != 0) {
    Serial.println("error!!!");
    delay(1000);
  }
  Serial.println("Work mode switch successful");

  Serial.print("Current work mode:");
  switch (hu.getWorkMode()) {
    case 1:
      Serial.println("Fall detection mode");
      break;
    case 2:
      Serial.println("Sleep detection mode");
      break;
    default:
      Serial.println("Read error");
  }

  hu.configLEDLight(hu.eHPLed, 1);  // Set HP LED switch
  hu.sensorRet();                   // Module reset, must perf

  Serial.print("HP LED status:");
  switch (hu.getLEDLightState(hu.eHPLed)) {
    case 0:
      Serial.println("Off");
      break;
    case 1:
      Serial.println("On");
      break;
    default:
      Serial.println("Read error");
  }

  Serial.println();
  Serial.println();
}

void loop() {
  Serial.print("Bed entry status:");
  switch (hu.smSleepData(hu.eInOrNotInBed)) {
    case 0:
      Serial.println("Out of bed");
      break;
    case 1:
      Serial.println("In bed");
      break;
    default:
      Serial.println("Read error");
  }

  Serial.print("Sleep status:");
  switch (hu.smSleepData(hu.eSleepState)) {
    case 0:
      Serial.println("Deep sleep");
      break;
    case 1:
      Serial.println("Light sleep");
      break;
    case 2:
      Serial.println("Awake");
      break;
    case 3:
      Serial.println("None");
      break;
    default:
      Serial.println("Read error");
  }
  Serial.printf("Awake duration: %d\n", hu.smSleepData(hu.eWak
  Serial.printf("Deep sleep duration: %d\n", hu.smSleepData(hu
  Serial.printf("Sleep quality score: %d\n", hu.smSleepData(hu

  sSleepComposite comprehensiveState = hu.getSleepComposite();
  Serial.println("Comprehensive sleep status:{");

  Serial.print("\tExistence status: ");
  switch (comprehensiveState.presence) {
    case 0:
      Serial.println("No one");
      break;
    case 1:
      Serial.println("Someone is present");
      break;
    default:
      Serial.println("Read error");
  }

  Serial.print("\tSleep status:");
  switch (comprehensiveState.sleepState) {
    case 0:
      Serial.println("Deep sleep");
      break;
    case 1:
      Serial.println("Light sleep");
      break;
    case 2:
      Serial.println("Awake");
      break;
    case 3:
      Serial.println("None");
      break;
    default:
      Serial.println("Read error");
  }

  Serial.printf("\tAverage respiration rate: %d\n", comprehens
  Serial.printf("\tAverage heart rate: %d\n", comprehensiveSta
  Serial.printf("\tNumber of turns: %d\n", comprehensiveState.
  Serial.printf("\tProportion of significant body movement: %d
  Serial.printf("\tProportion of minor body movement: %d\n", c
  Serial.printf("\tNumber of apneas: %d\n", comprehensiveState
  Serial.println("}");

  Serial.print("Sleep abnormalities:");
  switch (hu.smSleepData(hu.eSleepDisturbances)) {
    case 0:
      Serial.println("Sleep duration less than 4 hours");
      break;
    case 1:
      Serial.println("Sleep duration more than 12 hours");
      break;
    case 2:
      Serial.println("Long time abnormal absence of person");
      break;
    case 3:
      Serial.println("None");
      break;
    default:
      Serial.println("Read error");
  }

  sSleepStatistics statistics = hu.getSleepStatistics();  // G
  Serial.println("Sleep statistics:{");
  Serial.printf("\tSleep quality score: %d\n", statistics.slee
  Serial.printf("\tProportion of awake time: %d\n", statistics
  Serial.printf("\tProportion of light sleep time: %d\n", stat
  Serial.printf("\tProportion of light sleep time: %d\n", stat
  Serial.printf("\tProportion of deep sleep time: %d\n", stati
  Serial.printf("\tOut of bed duration: %d\n", statistics.time
  Serial.printf("\tNumber of times out of bed: %d\n", statisti
  Serial.printf("\tNumber of turns: %d\n", statistics.turnOver
  Serial.printf("\tAverage respiration: %d\n", statistics.aver
  Serial.printf("\tAverage heartbeat: %d\n", statistics.averag
  Serial.println("}");

  Serial.print("Sleep quality rating: ");
  switch (hu.smSleepData(hu.eSleepQualityRating)) {
    case 0:
      Serial.println("None");
      break;
    case 1:
      Serial.println("Good sleep quality");
      break;
    case 2:
      Serial.println("Average sleep quality");
      break;
    case 3:
      Serial.println("Poor sleep quality");
      break;
    default:
      Serial.println("Read error");
  }

  Serial.print("Abnormal struggle status: ");
  switch (hu.smSleepData(hu.eAbnormalStruggle)) {
    case 0:
      Serial.println("None");
      break;
    case 1:
      Serial.println("Normal status");
      break;
    case 2:
      Serial.println("Abnormal struggle status");
      break;
    default:
      Serial.println("Read error");
  }

  Serial.println();
  Serial.println();
  delay(1000);
}
```

Result



```
14:49:15.449 -> Initialization successful
14:49:15.449 -> Start switching work mode
14:49:15.543 -> Work mode switch successful
14:49:15.543 -> Current work mode:Sleep detection mode
14:49:26.332 -> HP LED status:On
14:49:26.459 ->
14:49:26.459 ->
14:49:26.459 -> Bed entry status:Out of bed
14:49:26.551 -> Sleep status:None
14:49:26.755 -> Awake duration: 0
14:49:26.838 -> Deep sleep duration: 0
14:49:26.957 -> Sleep quality score: 0
14:49:27.192 -> Comprehensive sleep status:{
14:49:27.192 ->         Existence status: No one
14:49:27.192 ->         Sleep status:None
14:49:27.192 ->         Average respiration rate: 0
14:49:27.192 ->         Average heart rate: 0
14:49:27.192 ->         Number of turns: 0
14:49:27.192 ->         Proportion of significant body movement: 0
14:49:27.192 ->         Proportion of minor body movement: 0
14:49:27.192 ->         Number of apneas: 0
14:49:27.192 -> }
14:49:27.192 -> Sleep abnormalities:None
14:49:27.467 -> Sleep statistics:{
14:49:27.467 ->         Sleep quality score: 0
14:49:27.467 ->         Proportion of awake time: 0
14:49:27.467 ->         Proportion of light sleep time: 0
14:49:27.467 ->         Proportion of light sleep time: 0
14:49:27.467 ->         Proportion of deep sleep time: 0
14:49:27.467 ->         Out of bed duration: 0
14:49:27.467 ->         Number of times out of bed: 0
14:49:27.467 ->         Number of turns: 0
14:49:27.467 ->         Average respiration: 0
14:49:27.467 ->         Average heartbeat: 0
14:49:27.467 -> }
14:49:27.599 -> Sleep quality rating: None
14:49:27.599 -> Abnormal struggle status: None
14:49:27.769 ->
14:49:27.769 ->
14:49:28.768 -> Bed entry status:Out of bed
14:49:28.862 -> Sleep status:None
14:49:29.134 -> Awake duration: 0
14:49:29.361 -> Deep sleep duration: 0
14:49:29.447 -> Sleep quality score: 0
```

Fall Information Acquisition

Preparation

- Hardware

  C1001 Millimeter Wave Human Detection Sensor

  ESP32-S3-DevKitC-1 Development Board, or other ESP32-based development boards.

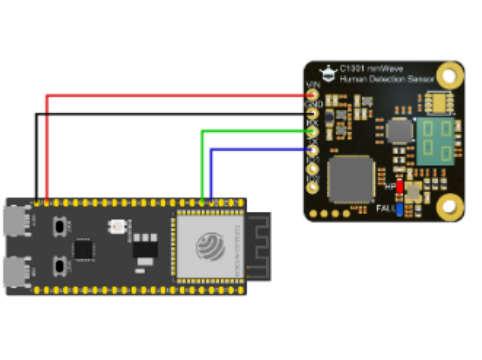
- Software

  Arduino IDE, Click to download Arduino IDE

  DFRobot_HumanDetection library, Click to download DFRobot_HumanDetection library

  How to install library files, Click the link

Wiring Diagram



Millimeter Wave	ESP32-S3
VIN	5V
GND	GND
RX	IO5
TX	IO4

Installation Method

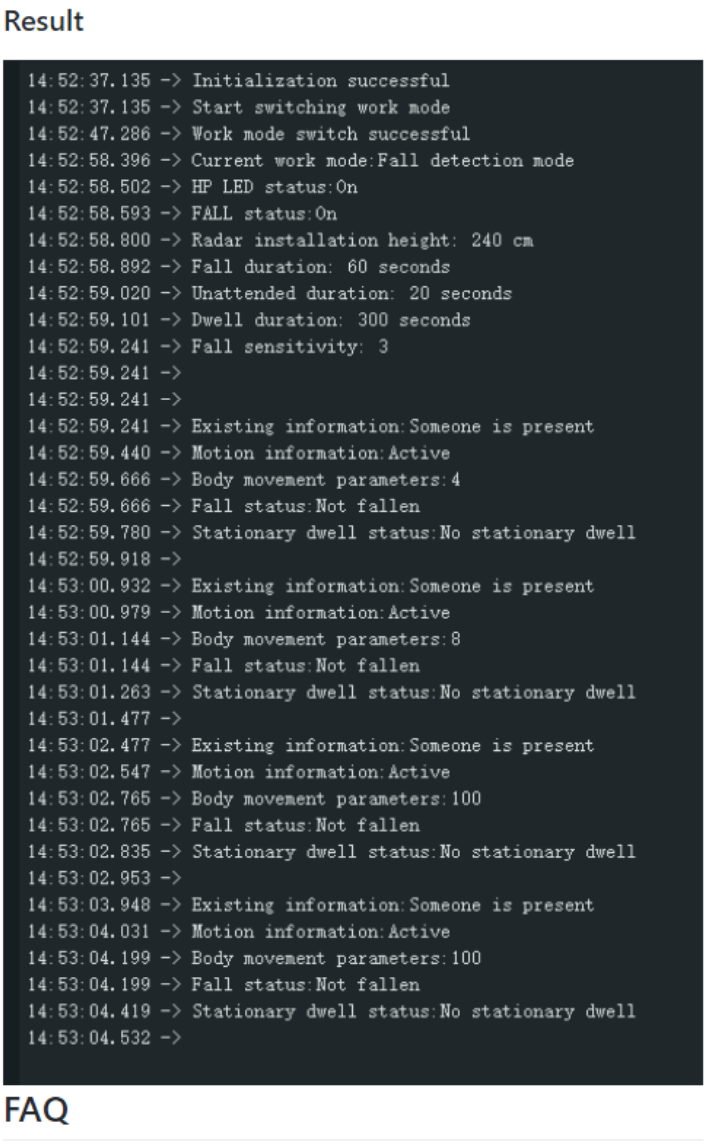
There are no special installation requirements for human presence detection, both top and side installations are possible. For fall detection, the sensor should be installed directly above the test area, with the front of the sensor facing down. As shown below.

Sample Code

Copy the following code into your Arduino IDE and upload.

```
/**!
 * @file basics.ino
 * @brief This is the fall detection usage routine of the C100
 * @copyright  Copyright (c) 2010 DFRobot Co.Ltd (http://www.d
 * @license     The MIT License (MIT)
 * @author [tangjie](jie.tang@dfrobot.com)
 * @version  V1.0
 * @date  2024-06-03
 * @url https://github.com/DFRobot/DFRobot_HumanDetection
 */

#include "DFRobot_HumanDetection.h"

DFRobot_HumanDetection hu(&Serial1);

void setup() {
  Serial.begin(115200);
  Serial1.begin(115200, SERIAL_8N1, 4, 5);

  Serial.println("Start initialization");
  while (hu.begin() != 0) {
    Serial.println("init error!!!");
    delay(1000);
  }
  Serial.println("Initialization successful");

  Serial.println("Start switching work mode");
  while (hu.configWorkMode(hu.eFallingMode) != 0) {
    Serial.println("error!!!");
    delay(1000);
  }
  Serial.println("Work mode switch successful");

  hu.configLEDLight(hu.eFALLLed, 1);         // Set HP LED swi
  hu.configLEDLight(hu.eHPLed, 1);           // Set FALL LED s
  hu.dmInstallHeight(270);                   // Set installati
  hu.dmFallTime(5);                          // Set fall time,
  hu.dmUnmannedTime(1);                      // Set unattended
  hu.dmFallConfig(hu.eResidenceTime, 200);   // Set dwell time
  hu.dmFallConfig(hu.eFallSensitivityC, 3);  // Set fall sensi
  hu.sensorRet();                            // Module reset,

  Serial.print("Current work mode:");
  switch (hu.getWorkMode()) {
    case 1:
      Serial.println("Fall detection mode");
      break;
    case 2:
      Serial.println("Sleep detection mode");
      break;
    default:
      Serial.println("Read error");
  }

  Serial.print("HP LED status:");
  switch (hu.getLEDLightState(hu.eHPLed)) {
    case 0:
      Serial.println("Off");
      break;
    case 1:
      Serial.println("On");
      break;
    default:
      Serial.println("Read error");
  }
  Serial.print("FALL status:");
  switch (hu.getLEDLightState(hu.eFALLLed)) {
    case 0:
      Serial.println("Off");
      break;
    case 1:
      Serial.println("On");
      break;
    default:
      Serial.println("Read error");
  }

  Serial.printf("Radar installation height: %d cm\n", hu.dmGet
  Serial.printf("Fall duration: %d seconds\n", hu.getFallTime(
  Serial.printf("Unattended duration: %d seconds\n", hu.getUnm
  Serial.printf("Dwell duration: %d seconds\n", hu.getStaticRe
  Serial.printf("Fall sensitivity: %d \n", hu.getFallData(hu.e
  Serial.println();
  Serial.println();
}

void loop() {
  Serial.print("Existing information:");
  switch (hu.smHumanData(hu.eHumanPresence)) {
    case 0:
      Serial.println("No one is present");
      break;
    case 1:
      Serial.println("Someone is present");
      break;
    default:
      Serial.println("Read error");
  }

  Serial.print("Motion information:");
  switch (hu.smHumanData(hu.eHumanMovement)) {
    case 0:
      Serial.println("None");
      break;
    case 1:
      Serial.println("Still");
      break;
    case 2:
      Serial.println("Active");
      break;
    default:
      Serial.println("Read error");
  }

  Serial.printf("Body movement parameters:%d\n", hu.smHumanDat
  Serial.print("Fall status:");
  switch (hu.getFallData(hu.eFallState)) {
    case 0:
      Serial.println("Not fallen");
      break;
    case 1:
      Serial.println("Fallen");
      break;
    default:
      Serial.println("Read error");
  }

  Serial.print("Stationary dwell status:");
  switch (hu.getFallData(hu.estaticResidencyState)) {
    case 0:
      Serial.println("No stationary dwell");
      break;
    case 1:
      Serial.println("Stationary dwell present");
      break;
    default:
      Serial.println("Read error");
  }
  Serial.println();
  delay(1000);
}
```

Result



```
14:52:37.135 -> Initialization successful
14:52:37.135 -> Start switching work mode
14:52:47.286 -> Work mode switch successful
14:52:58.396 -> Current work mode:Fall detection mode
14:52:58.502 -> HP LED status:On
14:52:58.593 -> FALL status:On
14:52:58.800 -> Radar installation height: 240 cm
14:52:58.892 -> Fall duration: 60 seconds
14:52:59.020 -> Unattended duration: 20 seconds
14:52:59.101 -> Dwell duration: 300 seconds
14:52:59.241 -> Fall sensitivity: 3
14:52:59.241 ->
14:52:59.241 ->
14:52:59.241 -> Existing information:Someone is present
14:52:59.440 -> Motion information:Active
14:52:59.666 -> Body movement parameters:4
14:52:59.666 -> Fall status:Not fallen
14:52:59.780 -> Stationary dwell status:No stationary dwell
14:52:59.918 ->
14:53:00.932 -> Existing information:Someone is present
14:53:00.979 -> Motion information:Active
14:53:01.144 -> Body movement parameters:8
14:53:01.144 -> Fall status:Not fallen
14:53:01.263 -> Stationary dwell status:No stationary dwell
14:53:01.477 ->
14:53:02.477 -> Existing information:Someone is present
14:53:02.547 -> Motion information:Active
14:53:02.765 -> Body movement parameters:100
14:53:02.765 -> Fall status:Not fallen
14:53:02.835 -> Stationary dwell status:No stationary dwell
14:53:02.953 ->
14:53:03.948 -> Existing information:Someone is present
14:53:04.031 -> Motion information:Active
14:53:04.199 -> Body movement parameters:100
14:53:04.199 -> Fall status:Not fallen
14:53:04.419 -> Stationary dwell status:No stationary dwell
14:53:04.532 ->
```

FAQ

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

Get **C1001 mmWave Human Detection Sensor** from DFRobot Store or **DFRobot Distributor**.