Laser Ranging Module

BM42S5321-1
Arduino Library V1.0.1 Description

Revision: V1.10 Date: June 27, 2024

www.bestmodulescorp.com

Contents

Introduction

The Best Modules BM42S5321-1 is a laser ranging module, which uses the I²C and UART communication methods. This document describes the Arduino Lib function of the BM42S5321-1 and how to install the Arduino Lib. The example demonstrates the function of reading the distance with the BML36K532.

Applicable model:

Part No.	Description
BM42S5321-1	Laser ranging module
BML36K532	Integrated an adapter board, an adapter cable and the BM42S5321-1

Arduino Lib Functions

Arduino Lib Name: BM42S5321-1 Lib Version: V1.0.1		
Constructors & Initialisation		
1	BM42S5321_1(uint8_t intPin, HardwareSerial *theSerial=&Serial)	
	Description	Constructor, select hardware UART interface
	Parameter	intPin: The INT pin which is connected to the BM42S531-1 INT pin or BML36K532 INT pin *theSerial: selects hardware UART interface (default serial interface)
	Return Value	—
	Note	—
2	BM42S5321_1(uint8_t intPin, uint8_t rxPin, uint8_t txPin)	
	Description	Constructor, select software UART interface
	Parameter	intPin: The INT pin which is connected to the BM42S531-1 INT pin or the BML36K532 INT pin rxPin: The RX pin which is connected to the BM42S531-1 TX pin or the BML36K532 TX pin txPin: The TX pin which is connected to the BM42S531-1 RX pin or the BML36K5322 RX pin
	Return Value	—
	Note	—
3	BM42S5321_1(uint8_t intPin, TwoWire *theWire=&Wire, uint8_t i2c_addr=0x07)	
	Description	Constructor, select I²C communication
	Parameter	intPin: The INT pin which is connected to the BM42S531-1 INT pin or the BML36K532 INT pin *theWire: I²C communication interface selection (default wire interface) i2c_addr: Module I²C communication address (default 0x07)
	Return Value	—
	Note	—
4	void begin(uint8_t buard)	
	Description	Module initialisation
	Parameter	buard: UART baud rate selection 0x00 (BUARD_9600): 9600bps 0x01 (BUARD_115200): 115200bps
	Return Value	void
	Note	This function can be called if the UART communication method is selected

5	void begin()	
	Description	Module initialisation
	Parameter	—
	Return Value	void
	Note	The function can be called if the I²C communication method is selected
Performance Functions		
6	uint8_t getINT()	
	Description	Obtain the INT pin level
	Parameter	—
	Return Value	INT pin level 0x00: Low 0x01: High
	Note	If the measured distance value is below the lower threshold or above the upper threshold, the INT pin outputs a low level until it returns to the threshold value of 30mm. Refer to the corresponding user guide for details
7	uint8_t startMeasure()	
	Description	Start measuring the distance
	Parameter	—
	Return Value	Execution result: 0x00: Succeeded 0x01: Failed
	Note	Note
8	uint8_t stopMeasure()	
	Description	Stop measuring the distance
	Parameter	—
	Return Value	Execution result: 0x00: Succeeded 0x01: Failed
	Note	Note
9	uint16_t readDistance(uint16_t &dataState)	
	Description	Read the distance
	Parameter	&dataState: Store the data state of the current distance (dataState&0x0f00)=0: Currently updated data (dataState&0x0f00)=1: No data update (dataState&0x000f)=0: The data is reliable (dataState&0x000f)=1: The data is unreliable, the sigma error (dataState&0x000f)=2: The data is unreliable, the signal received by the sensor is too low (dataState&0x000f)=4: The data is unreliable, the sensor exceeds the threshold (dataState&0x000f)=7: The data is unreliable, the surround error
	Return Value	Distance, unit: mm
	Note	—
10	uint8_t calibrateOffset()	
	Description	Offset calibration
	Parameter	—
	Return Value	Execution result: 0x00: Start calibration 0x01: Setting failed
	Note	Note: 1. The module provides a calibration function to help users improve ranging accuracy. It is required to implement calibration in a calibration environment where the object to be measured is largerthan 10cm×10cm and placed at 140mm. Start calibration by using the calibration command, thecalibration process lasts about 10s, do not touch the module during the calibration period 2. Use with the IsCalibration() function

11	bool IsCalibration()	
	Description	Obtain the calibrationis complete or not
	Parameter	—
	Return Value	Calibration status: TRUE: Calibration complete FALSE: Calibration is not complete
	Note	If the calibration is not completed, it is forbidden to set other functional parameters
12	uint8_t sleep()	
	Description	Enter the Sleep mode
	Parameter	—
	Return Value	Execution result: 0x00: Succeeded 0x01: Failed
	Note	Use the wakeUp function to wake up the device
13	uint8_t wakeUp()	
	Description	Exit the Sleep mode
	Parameter	—
	Return Value	Execution result: 0x00: Succeeded 0x01: Failed
	Note	—
14	uint8_t resetMCU()	
	Description	Reset the MCU
	Parameter	—
	Return Value	Execution result: 0x00: Succeeded 0x01: Failed
	Note	—
15	uint8_t restoreDefault()	
	Description	Factory reset
	Parameter	—
	Return Value	Execution result: 0x00: Succeeded 0x01: Failed
	Note	It should be powered on again
16	uint8_t saveSettings()	
	Description	Save the relevant settings
	Parameter	—
	Return Value	Execution result: 0x00: Succeeded 0x01: Failed
	Note	The measurement mode, the ranging mode, the single sample time, the continuous measurement period, ROI, upper and lower thresholds, the baud rate, the module address and the user calibration parameters should be set
17	uint16_t getFWver()	
	Description	Obtain the FW version number
	Parameter	—
	Return Value	FW version number
	Note	For example: If a 0x0102 value is returned, it indicates that the current version is 1.02

Parameter Configuration & Acquisition		
18	uint8_t getDistanceMode()	
	Description	Obtain the ranging mode
	Parameter	—
	Return Value	Ranging mode: 0x00: Continuous measurement 0x01: Single measurement 0x02: Low power measurement
	Note	—
19	uint8_t getMeasureMode()	
	Description	Obtain the measure mode
	Parameter	—
	Return Value	Measure mode: 0x00: Long-range mode 0x01: Short-range mode
	Note	—
20	uint16_t getTimingBudgetInMs()	
	Description	Obtain the single sample time
	Parameter	—
	Return Value	The single sample time, unit: ms
	Note	—
21	uint16_t getIntermeasurementPeriod()	
	Description	Obtain the measurement period
	Parameter	—
	Return Value	Measurement period, unit: ms
	Note	—
22	uint8_t getRoiRegion()	
	Description	Obtain the sensor ROI
	Parameter	—
	Return Value	Sensor ROI: 0x00: ROI array, 16×16 0x01: ROI array, 8×8 0x02: ROI array, 4×4
	Note	—
23	void getThshold(uint16_t &upperLimit, uint16_t &lowerLimit)	
	Description	Obtain the alarm lower and upper thresholds
	Parameter	&upperLimit: Store the alarm upper threshold, unit: mm &lowerLimit: Store the alarm lower threshold, unit: mm
	Return Value	void
	Note	—
24	uint16_t getLigthKcps()	
	Description	Obtain the light intensity
	Parameter	—
	Return Value	Light intensity
	Note	Note: 1.The function requires starting ranging to obtain the light intensity 2. The greater the return value, the stronger the light intensity
25	uint16_t getFactoryVal()	
	Description	Obtain the factory calibration values
	Parameter	—
	Return Value	Factory calibration values
	Note	—

26	uint8_t setDistanceModeLong()	
	Description	Set the measure mode to the Long-range mode (measure range is 40mm~4000mm)
	Parameter	—
	Return Value	Execution result: 0x00: Succeeded 0x01: Failed
	Note	The factory default measure mode is the Long-range mode
27	uint8_t setDistanceModeShort()	
	Description	Set the measure mode to the Short-range mode (measure range is 40mm~1300mm)
	Parameter	—
	Return Value	Execution result: 0x00: Succeeded 0x01: Failed
	Note	The factory default measure mode is the Long-range mode
28	uint8_t setMeasureMode(uint8_t measureMode)	
	Description	Set the ranging mode
	Parameter	measureMode: Ranging mode 0x00 (ContinuousRanging): Continuous measurement (default) 0x01 (SingleRanging): Single measurement 0x02 (PowerSaveRanging): Low power measurement
	Return Value	Execution result: 0x00: Succeeded 0x01: Failed
	Note	Note
29	uint8_t setTimingBudgetInMs(uint16_t timingBudget)	
	Description	Set the sample time
	Parameter	timingBudget: Sample time 15 (Smaptim_15): 15ms 20 (Smaptim_20): 20ms 33 (Smaptim_33): 33ms 50 (Smaptim_50): 50ms 100 (Smaptim_100): 100ms (default) 200 (Smaptim_200): 200ms 500 (Smaptim_500): 500ms
	Return Value	Execution result: 0x00: Succeeded 0x01: Failed
	Note	—
30	uint8_t setIntermeasurementPeriod(uint16_t intermeasurement)	
	Description	Set the measurement period
	Parameter	intermeasurement: Measurement period, which is the time interval between two ranging measurements during continuous measurement, range from 0 to 5000, unit: ms
	Return Value	Execution result: 0x00: Succeeded 0x01: Failed
	Note	It needs to be greater than the single sample time. The factory default is 100ms

31	uint8_t setRoiRegion(uint8_t region)	
	Description	Set the sensor ROI
	Parameter	region: Sensor ROI 0x00 (RoiRegion_16x16): ROI array, 16×16 (default) 0x01 (RoiRegion_8x8): ROI array, 8×8 0x02 (RoiRegion_4x4): ROI array, 4×4
	Return Value	Execution result: 0x00: Succeeded 0x01: Failed
	Note	ROI arrays affect signal reception and power consumption, the larger the ROI, the easier the signal is to be received and the greater the power consumption, it is usually unchanged
32	uint8_t setThshold(uint16_t upperLimit, uint16_t lowerLimit)	
	Description	Set the alarm lower and upper thresholds
	Parameter	upperLimit: the alarm upper threshold, range from 40 to 4000, unit: ms, default is 4000 lowerLimit: the alarm lower threshold, range from 40 to 4000, unit: ms, default is 0
	Return Value	Execution result: 0x00: Succeeded 0x01: Failed
	Note	Note: 1. The alarm upper threshold must be greater than the alarm lower threshold 2. If the value is below the lower threshold or above the upper threshold, the INT pin outputs a low level until it returns to the threshold value of 30mm.
33	uint8_t setUartBuard(uint8_t buard)	
	Description	Set the UART baud rate
	Parameter	buard: UART baud rate 0x00 (BUARD_9600): 9600bps (default) 0x01 (BUARD_115200): 115200bps
	Return Value	Execution result: 0x00: Succeeded 0x01: Failed
	Note	—
34	uint8_t setI2CAddress(uint8_t i2c_addr)	
	Description	Set the I²C address
	Parameter	i2c_addr: I²C address, range from 0 to 127, default is 0x07
	Return Value	Execution result: 0x00: Succeeded 0x01: Failed
	Note	—

Note: Set the ranging mode by the setMeasureMode function (Function 28)

Continuous measurement: After selecting the continuous measurement, the distance can be continuously measured by the startMeasure function (Function 7). The measured distance can be read by the readDistance function (Function 9) and stop measuring distance by the stopMeasure function (Function 8).

Single measurement: After selecting the single measurement, the distance can be measured once by the startMeasure fuction (Function 7), the distance measurement can be automatically stopped after the distance is measured, and the measured distance can be read by the readDistance function (Function 9).

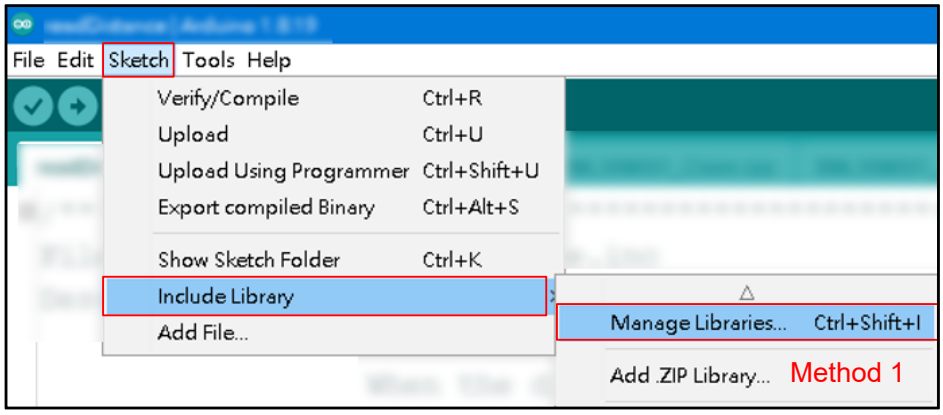
Low-power measurement: After selecting the low-power measurement, the default measurement period is 500ms and the default single sampling period is 20ms. Other operations are the same as those for continuous measurement.

Arduino Lib Download and Installation

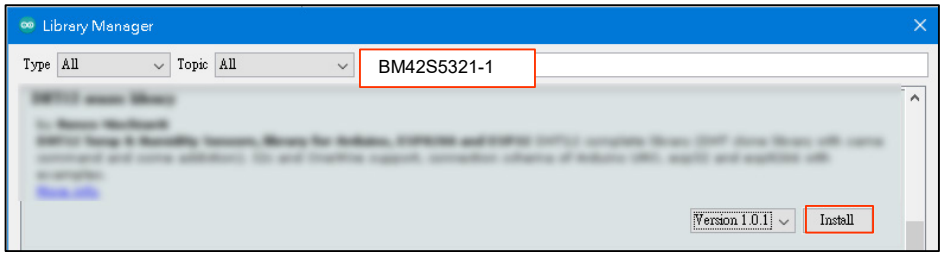
BM42S5321-1 Library: Refer to the following two methods to install the BM42S5321-1 Arduino Library.

Method 1: Search for installation

Arduino IDE→Sketch→Include Library→Manage Libraries...→Search BM42S5321-1→Install



Search for Installation Step 1

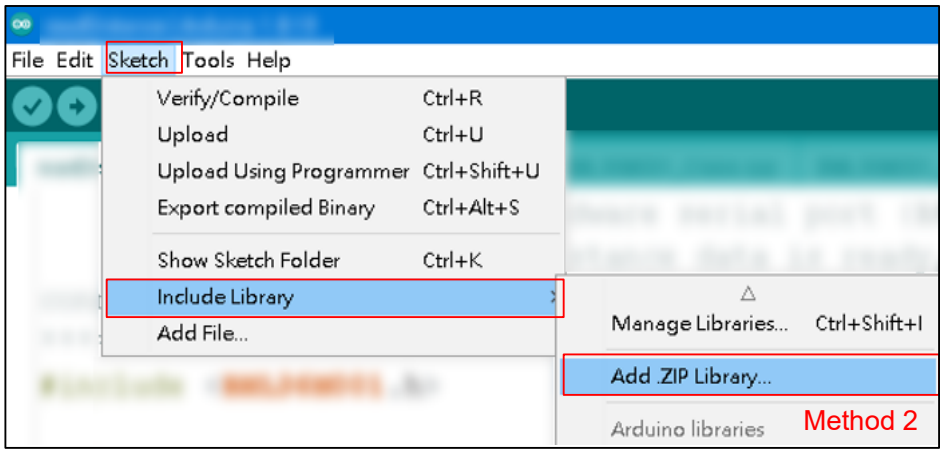


Search for Installation Step 2

Method 2: Download the .ZIP library before adding it

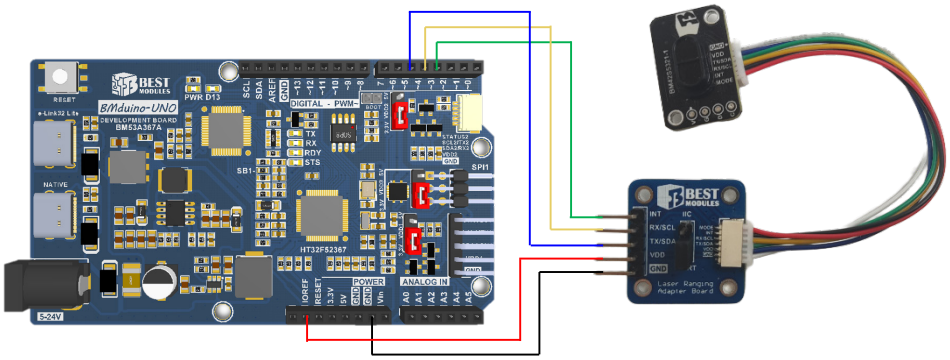
Download the Arduino example (BM42S5321-1Library) under the DOCUMENTS menu from the Best Modules website (https://www.bestmodulescorp.com/bm42s5321-1.html).

Add .ZIP library: Arduino IDE→Sketch→Include Library→Add .ZIP Library...



Arduino Example

Example 1: readDistanceWithUART



Physical Connection Diagram

Example function: Using the UART communication method, which need to short-connect the UART side of the jumper on the adapter board, set the module in the Long-range mode, read the distance every 200ms and display on the serial port monitor.

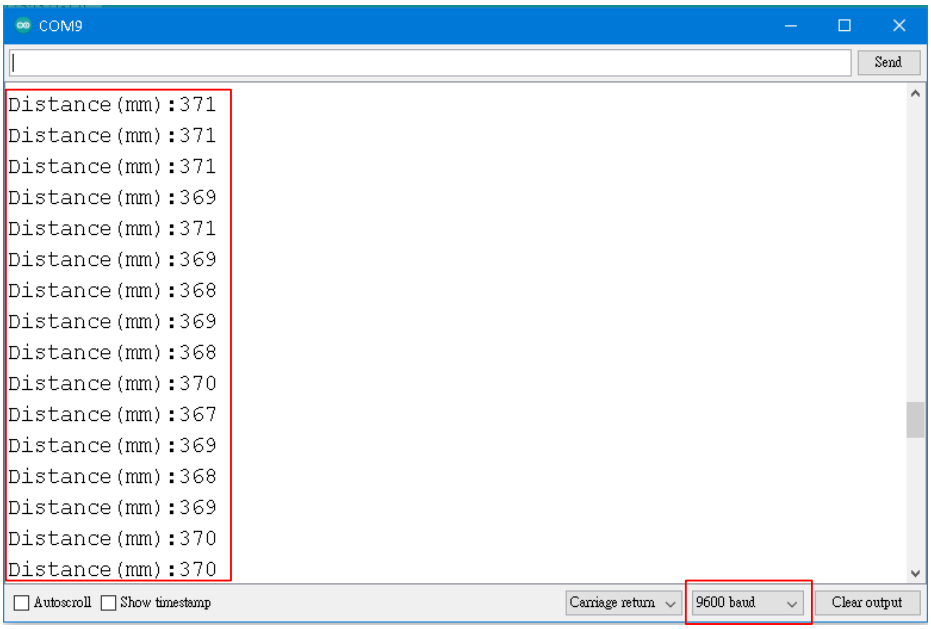
1. Open the example: File→Examples→Select Lib (BM42S5321-1)→Select example (readDistanceWithUART)
2. Example Description:
   a. Create object & module initialisation

```
#include "BM42S5321-1.h"
BM42S5321_1 BM42(3,5,4); // Create object, the INT pin connects to the
                         // development board D3, the TX pin connects
                         // to the development board D5, the RX pin
                         // connects to the development board D4
uint16_t dataState;
uint16_t Distance;
void setup()
{
  Serial.begin(9600);          // Serial port monitor initialisation
  BM42.begin(BUARD_9600);      // Module initialisation
  BM42.setDistanceModeLong(); // Select the Long-range mode BM42.
  setMeasureMode(ContinuousRanging); // Select the ranging mode is
                                     // continuous measurement
  BM42.startMeasure();               // Start measuring
}
```

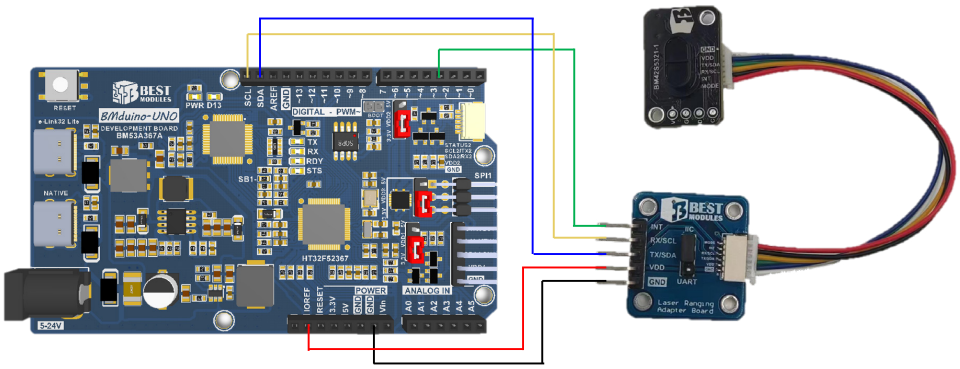
b. Read the distance every 200ms and display on the serial port monitor

```
void loop()
{
  Distance = BM42.readDistance(dataState); // Obtain the measurement
                                           // distance
  if(dataState == 0)// Determine whether the distance data is reliable
  {
    Serial.print("Distance(mm):");
    Serial.println(Distance); // Display the measured distance
  }
  else
  {
    Serial.println("Distance data is unreliable"); // Display the
                                                   // distance data is
                                                   // unreliable
  }
  delay(200);
}
```

3. Open the serial monitor and select the baud rate as to be 9600. The serial monitor will display as follows.



Example 2: readDistanceWithIIC



Physical Connection Diagram

Example function: Using the I²C communication method,which need to short-connect the I²C side of the jumper on the adapter board, set the module in the Long-range Mode, read the distance every 200ms and display on the serial port monitor.

1. Open the example: File→Examples→Select Lib (BM42S5321-1)→Select example (readDistanceWithIIC)
2. Example Description:
   a. Create object & module initialisation

```
#include "BM42S5321-1.h"
BM42S5321_1 BM42(3,&Wire); // Create object, the INT pin connects to
                           // the development board D3, the SCL pin
                           // connects to the development board D19,
                           // the SDA pin connects to the development
                           // board D18
uint16_t dataState;
uint16_t Distance;
void setup()
{
  Serial.begin(9600);          // Serial port monitor initialisation
  BM42.begin();                // Module initialisation
  BM42.setDistanceModeLong(); // Select the Long-range mode
  BM42.setMeasureMode(ContinuousRanging);// Select the ranging mode is
                                         // continuous measurement
  BM42.startMeasure();         // Start measuring
}
```

b. Read the distance every 200ms and display on the serial port monitor

```
void loop()
{
  Distance = BM42.readDistance(dataState); // Obtain the measurement
                                           // distance
  if(dataState == 0)// Determine whether the distance data is reliable
  {
    Serial.print("Distance(mm):");
    Serial.println(Distance); // Display the measured distance
  }
  else
  {
    Serial.println("Distance data is unreliable");// Display the
                                                 // distance data is
                                                 // unreliable
  }
  delay(200);
}
```

3. Open the serial monitor and select the baud rate as to be 9600. The serial monitor will display as follows.