Proximity & Light Module

BMS33M332
Arduino Library Description

Revision: V1.00 Date: May 22, 2023

www.bestmodulescorp.com

Contents

Introduction

The Best Modules BMS33M332 is a proximity sensing and ambient light detection module, which uses the I²C communication method. This document provides the description of the BMS33M332 Arduino Lib functions and how to install the Arduino Lib. The example demonstrates the function of reading A/D values for proximity sensing and ambient light.

Arduino Library Functions

Arduino Lib Name: BMS33M332		Lib version: V1.0.1
Constructors & Initialisation		
1	BMS33M332(uint8_t intPin, TwoWire *theWire=&Wire)	
	Description	Constructor
	Parameter	intPin: INT pin *theWire: I²C communication interface selection
	Return Value	—
	Note	—
2	void begin(uint8_t addr=BMS33M332_IICADDR)	
	Description	Module initialisation
	Parameter	Addr: Slave address, 0x47
	Return Value	void
	Note	—
Performance Functions		
3	uint16_t readRawProximity()	
	Description	Read the proximity sensing AD value
	Parameter	—
	Return Value	The proximity sensing AD value
	Note	—
4	uint16_t readRawAmbient()	
	Description	Read the ambient light AD value
	Parameter	—
	Return Value	Ambient light AD value
	Note	Ambient light value=AD value×(0.8204/alsIt/alsGain), unit: LUX. Among them,alsIt= 2^time, which can be set through the setALSIntegrationTime function; alsGain can be set using the setALSGain function.
5	float readAmbient()	
	Description	Read the ambient light value
	Parameter	—
	Return Value	Ambient light value, unit: LUX
	Note	—
6	uint8_t getPDTID()	
	Description	Get the device ID
	Parameter	—
	Return Value	Device ID
	Note	—

7	void setINT(uint16_t thdh, uint16_t thdl, bool isEnable=true)	
	Description	Set the interrupt trigger, which is used to control the thresholds and the enable operation
	Parameter	thdh: the PS high threshold thdl: the PS low threshold isEnable: true: enable (default) false: disable
	Return Value	void
	Note	—
8	uint8_t getINT()	
	Description	Get the INT pin level
	Parameter	—
	Return Value	INT pin level 0: LOW 1: HIGH
	Note	The INT pin level becomes LOW when the proximity sensing AD value exceeds the high threshold; and returns to HIGH when it returns below the low threshold
9	uint8_t getPositionStatus()	
	Description	Get the location status and detects whether a trigger that meets the condition exists
	Parameter	—
	Return Value	Location status 0: Interrupt is triggered 1: The interrupt was not triggered
	Note	The INT pin level becomes LOW when the proximity sensing AD value exceeds the high threshold; and returns to HIGH when it returns below the low threshold
10	void reset()	
	Description	reset
	Parameter	—
	Return Value	void
	Note	—
11	void writeReg(uint8_t addr, uint8_t data)	
	Description	Register write operation
	Parameter	addr: Register address data: Data to be written
	Return Value	void
	Note	—
12	uint8_t readReg(uint8_t addr)	
	Description	Register read operation
	Parameter	addr: Register address
	Return Value	Register data
	Note	—
13	void readReg(uint8_t addr, uint8_t rBuf[], uint8_t rLen)	
	Description	Register read continuously operation, storing rLen bytes in array rBuf[]
	Parameter	Addr: Register address rBuf[]: Used to store read data rLen: Bytes number to be read consecutively
	Return Value	void
	Note	—

Parameter Configuration		
14	uint8_t getLEDcurrent()	
	Description	Get the parameters of LED constant current
	Parameter	—
	Return Value	Parameters of LED constant current 0: 3.125mA 1: 6.25mA 2: 12.5mA 3: 25mA 4: 50mA 5: 100mA 6: 150mA
	Note	—
15	uint8_t getMeasureIntervalTime()	
	Description	Get the parameters of measuring interval time
	Parameter	—
	Return Value	Parameters of measuring interval time
	Note	Measuring interval time=(time+1)×1.54 ms
16	uint16_t getPSHighThreshold()	
	Description	Get the PS high threshold
	Parameter	—
	Return Value	PS high threshold
	Note	—
17	uint16_t getPSLowThreshold()	
	Description	Get the PS low threshold
	Parameter	—
	Return Value	PS low threshold
	Note	—
18	uint16_t getALSHighThreshold()	
	Description	Get the ALS high threshold
	Parameter	—
	Return Value	ALS high threshold
	Note	—
19	uint16_t getALSLowThreshold()	
	Description	Get the ALS low threshold
	Parameter	—
	Return Value	ALS low threshold
	Note	—
20	void setLEDcurrent(uint8_t current)	
	Description	Set the LED constant current
	Parameter	Current: LED constant current selection 0(CURRENT_3_125MA): 3.125mA 1(CURRENT_6_25MA): 6.25mA 2(CURRENT_12_5MA): 12.5mA 3(CURRENT_25MA): 25mA 4(CURRENT_50MA): 50mA 5(CURRENT_100MA): 100mA 6(CURRENT_150MA): 150mA
	Return Value	void
	Note	—

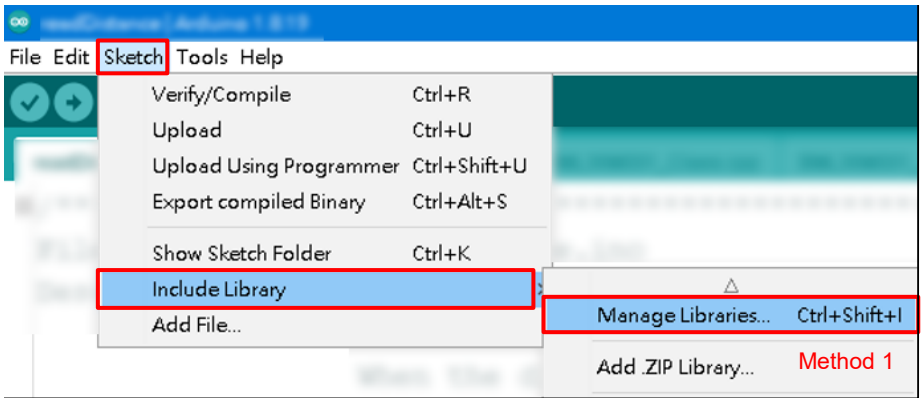
21	void setMeasureIntervalTime(uint8_t time, bool isEnable=true)	
	Description	Set the measuring interval time
	Parameter	time: Parameters of measuring interval time, measuring interval time=(time+1)×1.54 ms isEnable: true: enable (default) false: disable
	Return Value	void
	Note	—
22	void setPSHighThreshold(uint16_t thdh)	
	Description	Set the PS high threshold
	Parameter	thdh: PS high threshold
	Return Value	void
	Note	—
23	void setPSLowThreshold(uint16_t thdl)	
	Description	Set the PS low threshold
	Parameter	thdl: PS low threshold
	Return Value	void
	Note	—
24	void setALSHighThreshold(uint16_t thdh)	
	Description	Set the ALS high threshold
	Parameter	thdh: ALS high threshold
	Return Value	void
	Note	—
25	void setALSLowThreshold(uint16_t thdl)	
	Description	Set the ALS low threshold
	Parameter	Thdl: ALS low threshold
	Return Value	void
	Note	—

Arduino Lib Download and Installation

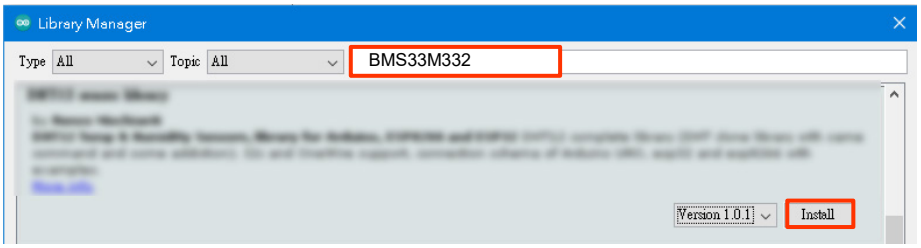
BMS33M332 Library: Refer to the following two methods to install the BMS33M332 Arduino Library.

Method 1: Search for installation

Search for installation: Arduino IDE→Sketch→Include Library→Manage Libraries...→Search BMS33M332→Install



Search for Installation Step 1

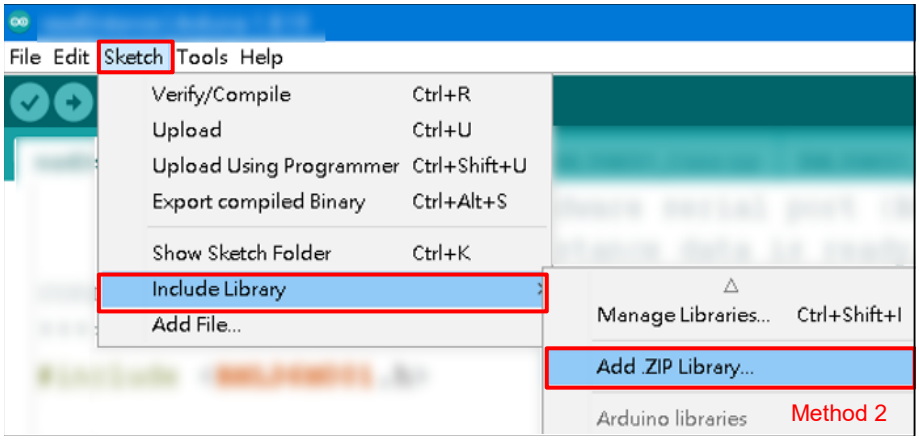


Search for Installation Step 2

Method 2: Download the .ZIP library before adding it

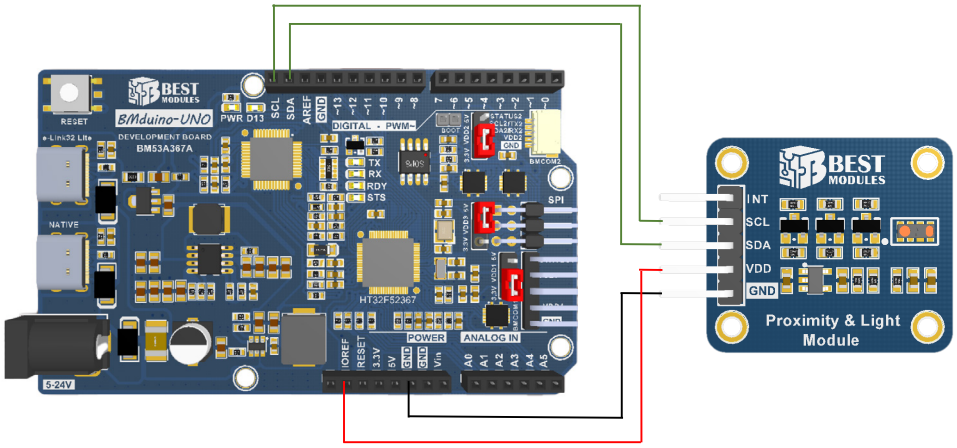
Download the Arduino example (BMS33M332 Library) under the DOCUMENTS menu from the Best Modules website (https://www.bestmodulescorp.com/bms33m332.html#tab-product2).

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



Arduino Example

Example1: readAmbientAndProximity



Physical Connection Diagram

Function: When the development board communicates with the module in I²C mode, it will obtain the proximity sensing A/D value and ambient light A/D value, and display it in the serial monitor.

1. Open the example:

   File→Examples→Select Lib (BMS33M332)→Select example (readAmbientAndProximity)

2. Example Description:

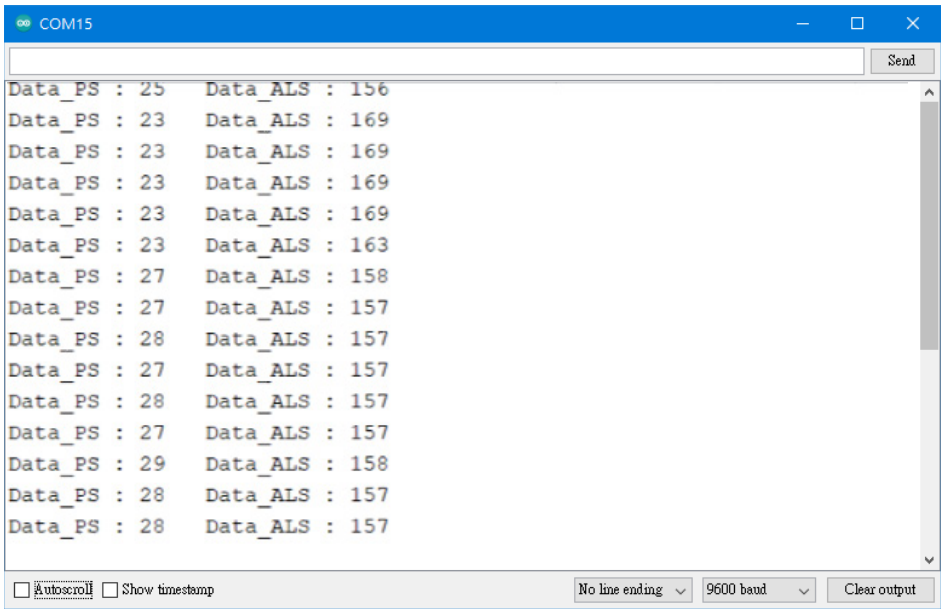
   a. Create object & initialise object

```
#include "BMS33M332.h"
BMS33M332  Alsps(8); // Select the Pin8 as the INT pin
uint16_t alsValue;
uint16_t psValue;
void setup()
{
  Serial.begin(9600);// Serial monitor initialisation, baud rate to be
                     // 9600
  Alsps.begin();     // Module initialisation
}
```

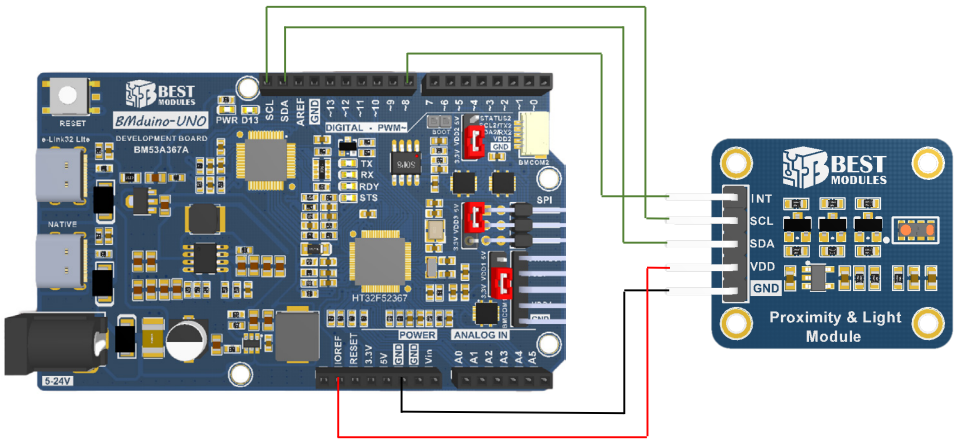
   b. The development board obtains the proximity sensing A/D value and ambient light A/D value, and display it in the serial monitor.

```
void loop()
{
  psValue = Alsps.readProximity(); // Read the proximity sensing A/D
                                   // value
  Serial.print("Data_PS : ");
  Serial.print(psValue);
  alsValue = Alsps.readAmbient();  // Read the ambient light A/D value
  Serial.print("Data_ALS : ");
  Serial.print(alsValue);
  Serial.println();
  delay(1000);
}
```

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



Example2: getPositionStatus



Physical Connection Diagram

Function: Communicate with the module in I²C mode to determine whether there is an object approaching. If there is an action that meets the setting, the result will be displayed in the serial monitor.

1. Open the example:

   File→Examples→Select Lib (BMS33M332)→Select example (getPositionStatus)

2. Example Description:

a. Create & Initialise objects

```
#include "BMS33M332.h"
BMS33M332  Alsps(8);          // Select the Pin8 as the INT pin
void setup()
{
  Serial.begin(9600);         // Serial monitor initialisation,baud
                              // rate to be 9600
  Alsps.begin();              // Module initialisation
  Alsps.setINT(400,200);      // Set the proximity sensing threshold:
                              // thdh = 400, thdl = 200
}
```

b. Get the status flag to determine whether there is an object approaching, and displays it in the "Serial Monitor"

```
void loop()
{
  if(Alsps.getPositionStatus() == 1) // Read the status flag bits
  {
    Serial.println("No objects found ):");
  }
  else Serial.println("Find object!");
  delay(1000);
}
```

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