UKMARSBOT I^{2}C Sensor Controller Library Documentation

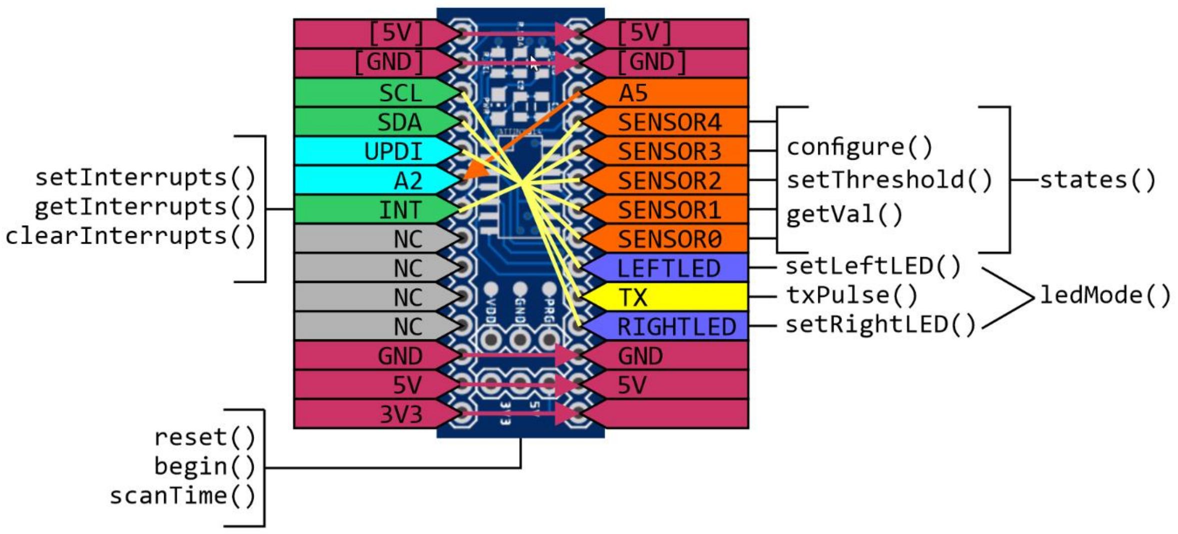
Table of Contents

The ISC has a supporting Arduino library to simplify the integration. This document provides an overview of the Arduino ISC library, describing the functions, their parameters and return values. Finally, example sketches are provided with the library to give practical examples of how the device may be used.

It is recommended, for better understanding of the capabilities of the device, that the ISC Hardware Datasheet be read in conjunction with this document.

1.0 LIBRARY OVERVIEW

The below diagram indicates schematically how the functions within the Arduino ISC Library may be used in conjunction with an ISC board.



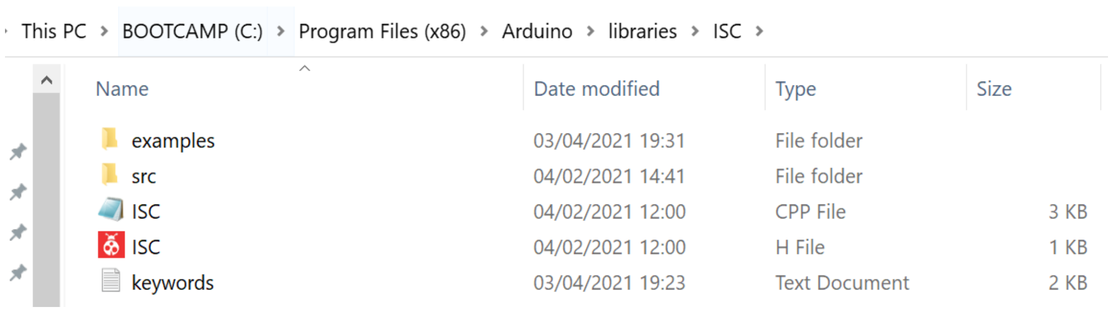
2.0 INSTALLATION

1. The ISC library and all other supporting files are provided at the following link:

   https://github.com/slinkyfish/ISC-Project

2. Download the project as a compressed file.

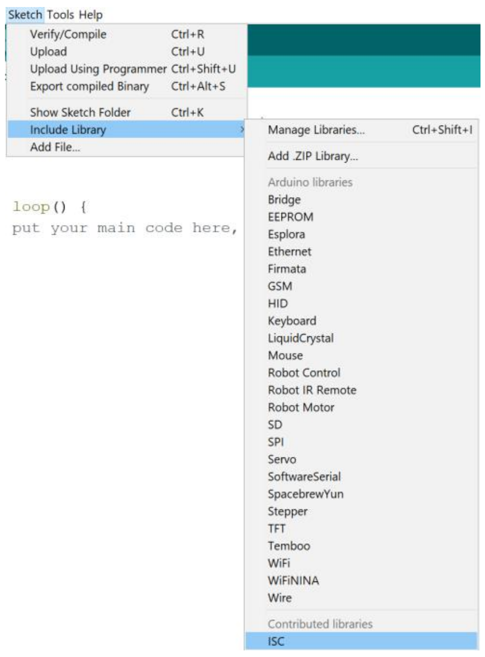
3. Extract the downloaded file and copy the ISC folder into the libraries folder within the Arduino installation. The directory structure should look like the below image.

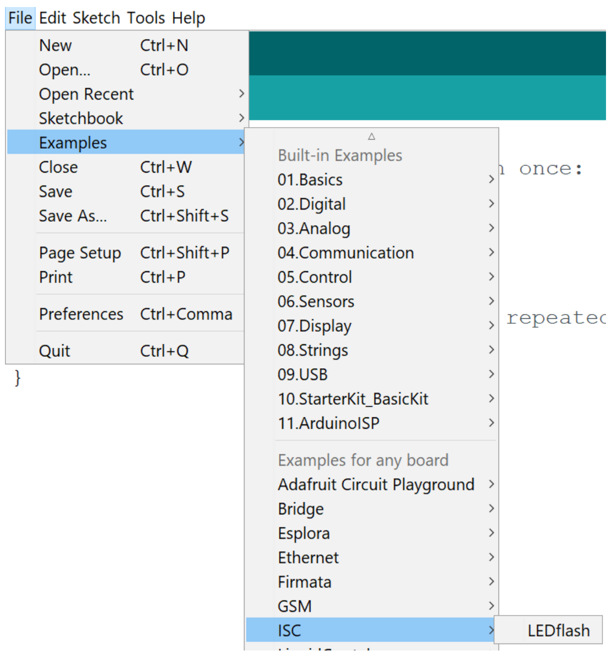


This PC › BOOTCAMP (C:) › Program Files (x86) › Arduino › libraries › ISC ›

Name	Date modified	Type	Size
examples	03/04/2021 19:31	File folder	
src	04/02/2021 14:41	File folder	
ISC	04/02/2021 12:00	CPP File	3 KB
ISC	04/02/2021 12:00	H File	1 KB
keywords	03/04/2021 19:23	Text Document	2 KB

4. Once the Arduino software is restarted, the library will be available to include, and example code detailed in Section **Error! Reference source not found.** will be available in the Examples menu.



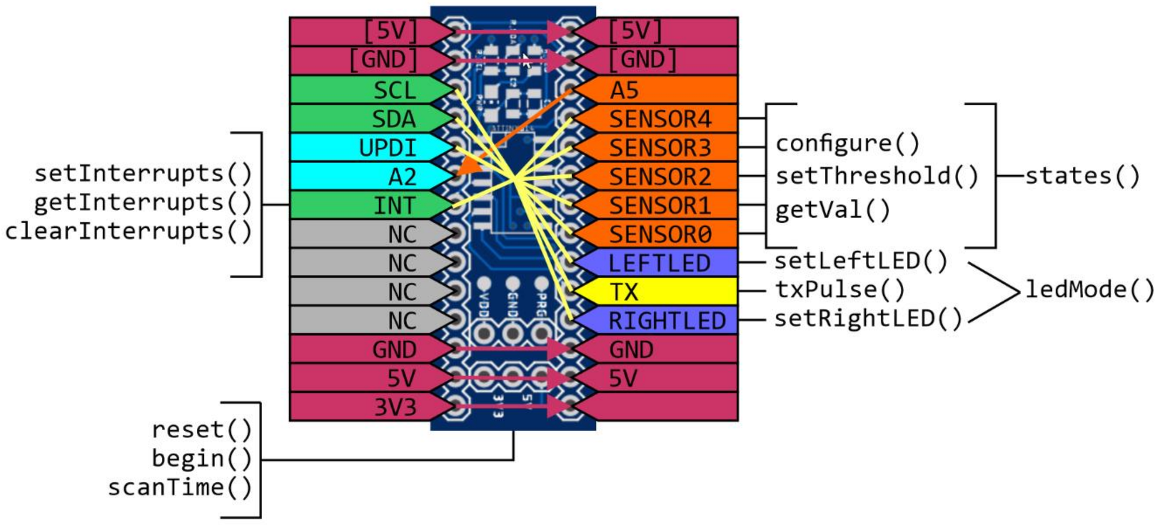


5. To use this library in any sketch, include the following at the start of the code:

```
#include <ISC.h>
```

3.0 FUNCTION DESCRIPTIONS

The functions included in the Arduino ISC Library are shown schematically in the graphic below, then listed with brief description and link to detailed description after.



Global

reset()	//Forces the ISC to a reset state, starting initial setup
begin()	//Starts the ISC in run mode, defines sensor board
scanTime()	//Returns the time taken for the ISC to complete a full cycle

LED Indicators

ledMode()	//Controls the Indicator LED function
setLeftLED()	//Direct control of Left LED state
setRightLED()	//Direct control of Right LED state

Individual sensors

configure()	//Individual sensor setup configuration
setThreshold()	//Specifies the value, above which, the state is ON
getVal()	//Returns the value of specified sensor

Overview of sensors

states()	//Returns a byte, with bits representing sensor states
txPulse()	//Sets time that Sensor Tx. is on before first sample is taken

Interrupts

setInterrupts()	//Set components that can generate an interrupt
getInterrupts()	//Determine which component(s) caused the interrupt
clearInterrupts()	//Acknowledges interrupts, clearing them down for next time

3.1 reset()

Description

Forces the ISC to a reset state, allowing the initial setup to be performed.

Syntax

```
void reset(uint8_t address);
```

Parameters

address	7-bit I^{2}C address for ISC device – printed on device.

Returns

N/A

Example

```
Isc.reset(0x50);  //Reset ISC with I2C (7-bit) Address of 0x50
```

3.2 begin()

Description

This function starts the ISC in run mode, confirming that setup is complete. It defines the sensor board that is attached.

Syntax

```
void begin(uint8_t boardType);
```

Parameters

boardType Type of sensor board connected to the ISC:

basicLineSensor	UKMARS Basic Line Sensor Board
basicWallSensor	UKMARS Basic Wall follower Board
spLineSensor	Line Sensor board by S. Pithouse
customBoard	Undefined board

Returns

N/A

Example

```
Isc.begin(basicLineSensor); //Start with Basic Line sensor attached
```

3.3 scanTime()

Description

This function returns the most recent time taken for the ISC to complete a full cycle. The units are µs.

Syntax

```
int scanTime();
```

Parameters

N/A

Returns

Most recent time taken for the ISC to complete a full cycle (µs).

Example

```
int myTime = Isc.scanTime();  //Store most recent scan time in myTime
```

3.4 setThreshold()

Description

This function allows the sensor threshold to be set for a specific sensor.

Syntax

```
void setThreshold(uint8_t sens, uint16_t threshVal);
```

Parameters

Sensor

SENSOR0	See diagram in Section ?? – A0 on Sensor Board
SENSOR1	See diagram in Section ?? – A0 on Sensor Board
SENSOR2	See diagram in Section ?? – A0 on Sensor Board
SENSOR3	See diagram in Section ?? – A0 on Sensor Board
SENSOR4	See diagram in Section ?? – A0 on Sensor Board

threshVal

Value between 0 – 1024

Returns

N/A

Example

```
Isc.setThreshold(SENSOR0, 500);
```

3.5 getVal()

Description

This function returns the most recent sensor value for the specified sensor.

Syntax

```
int getVal(uint8_t sensor);
```

Parameters

Sensor

SENSOR0	See diagram in Section ?? – ‘A0’ on Sensor Board
SENSOR1	See diagram in Section ?? – ‘A1’ on Sensor Board
SENSOR2	See diagram in Section ?? – ‘A2’ on Sensor Board
SENSOR3	See diagram in Section ?? – ‘A3’ on Sensor Board
SENSOR4	See diagram in Section ?? – ‘A4’ on Sensor Board

Returns

Value of specified sensor

Example

```
leftSensor = Isc.getVal(SENSOR0);
```

3.6 ledMode()

Description

This function controls the Indicator LED function

Syntax

```
void ledMode(uint8_t mode);
```

Parameters

mode

MASTERCTRL	MCU has control
FREQOUT	Scan frequency output on LEDs
LEDOFF	Turn LEDs off
LEDBRDCTRL	Allow board specific LED control
FASTBLINK	Start LEDS blinking fast
SLOWBLINK	Start LEDs blinking slowly

Returns

N/A

Example

```
Isc.ledMode(FASTBLINK);
```

3.7 setLeftLED()

Description

This function controls the Indicator LED function. Reads current led state, sets into Master control mode and turns Left LED on.

Syntax

```
void setLeftLED(uint8_t state);
```

Parameters

state

HIGH	Turn LED on
LOW	Turn LED off

Returns

N/A

Example

```
Isc.setLeftLED(HIGH);
```

3.8 setRightLED()

Description

This function controls the Indicator LED function. Reads current led state, sets into Master control mode and turns Left LED on.

Syntax

```
void setRightLED(uint8_t state);
```

Parameters

state

HIGH	Turn LED on
LOW	Turn LED off

Returns

N/A

Example

```
Isc.setRightLED(HIGH);
```

3.9 configure()

Description

This function allows individual sensor setup to be configured. There are keywords that may be combined with '+' as shown.

Syntax

```
Isc.configure(uint8_t sensor, uint8_t config);
```

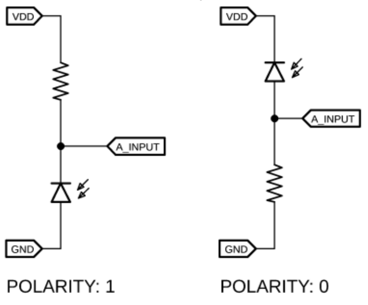
Parameters

sensor

SENSOR0	See diagram in Section ?? – 'A0' on Sensor Board
SENSOR1	See diagram in Section ?? – 'A1' on Sensor Board
SENSOR2	See diagram in Section ?? – 'A2' on Sensor Board
SENSOR3	See diagram in Section ?? – 'A3' on Sensor Board
SENSOR4	See diagram in Section ?? – 'A4' on Sensor Board

config

enb	Enable sensor – ISC will read the value
lowRes	Reduce value to 8-bit
txEnb	Enable Transmitter to help remove ambient light
fallingInterrupt	Sensor will cause interrupt on 1->0 state change
risingInterrupt	Sensor will cause interrupt on 0->1 state change
flip	[See Polarity]



Returns

N/A

Example

```
Isc.configure(SENSOR0, enb + txEnb + risingInterrupt);      //Marker
Sensor
```

3.10 txPulse()

Description

This function allows the time in µs that the Tx emitter is on before the first sample is taken to be adjusted.

Syntax

```
void txPulse(uint8_t length);
```

Parameters

Length

Amount of time in us…??

Returns

N/A

Example

```
Isc.txPulse(100);
```

3.11 states()

Description

This function gets the current states of the sensors and returns a byte with 1 representing HIGH sensor state.

Syntax

```
uint8_t states();
```

Parameters

N/A

Returns

Byte with bits indicating current sensor state

Example

```
leftState = Isc.states() & 0b1;
```

3.12 read()

Description

Reads a byte (or pair of bytes) from a specified register within the ISC.

Syntax

```
int read(uint8_t regAddr, uint16_t numVals);
```

Parameters

regAddr

Register address to read from

numVals

May be 1 or 2, for the number of consecutive registers to read (ie. 2 when a 2-byte value is to be read).

Returns

The value stored in that location (int)

Example

```
int timeTaken = Isc.read(SCANTIME, 1);//Read 1 byte at address SCANTIME
```

3.13 write()

Description

Writes a byte (or pair of bytes) to a specified register within the ISC.

Syntax

```
void write(uint8_t regAddr, uint16_t data, uint8_t numVals);
```

Parameters

regAddr

Register address to write to

data

Data to write

numVals

May be 1 or 2, for the number of consecutive registers to write (ie. 2 when a 2-byte value is to be written).

Returns

N/A

Example

```
Isc.write(SENS0THRSH, 500, 2);//Write 500 (2 bytes) for Sensor 0
Threshold
```

3.14 setInterrupts()

Description

Set components that can generate an interrupt.

Syntax

```
void setInterrupts(uint8_t interrupts);
```

Parameters

interrupts

SENSOR0INTERRUPT	Interrupt according to Sensor 0 Configuration
SENSOR1INTERRUPT	Interrupt according to Sensor 1 Configuration
SENSOR2INTERRUPT	Interrupt according to Sensor 2 Configuration
SENSOR3INTERRUPT	Interrupt according to Sensor 3 Configuration
SENSOR4INTERRUPT	Interrupt according to Sensor 4 Configuration

Returns

N/A

Example

```
setInterrupts(SENSOR0INTERRUPT + SENSOR1INTERRUPT);
//Interrupt on Sensor 0 and Sensor 1 as per their configuration
(rising/falling)
```

3.15 getInterrupts()

Description

Reads a byte (or pair of bytes) from a specified register within the ISC.

Syntax

```
uint8_t getInterrupts();
```

Parameters

N/A

Returns

Active Interrupts which may be interrogated

Example

```
    int activeInterrupts = getInterrupts(); //Store components which
                                            triggered interrupt
```

3.16 clearInterrupts()

Description

Acknowledges interrupts, clearing them down for next time.

Syntax

```
void clearInterrupts(uint8_t interrupts);
```

Parameters

interrupts

SENSOR0INTERRUPT	Interrupt according to Sensor 0 Configuration
SENSOR1INTERRUPT	Interrupt according to Sensor 1 Configuration
SENSOR2INTERRUPT	Interrupt according to Sensor 2 Configuration
SENSOR3INTERRUPT	Interrupt according to Sensor 3 Configuration
SENSOR4INTERRUPT	Interrupt according to Sensor 4 Configuration

Returns

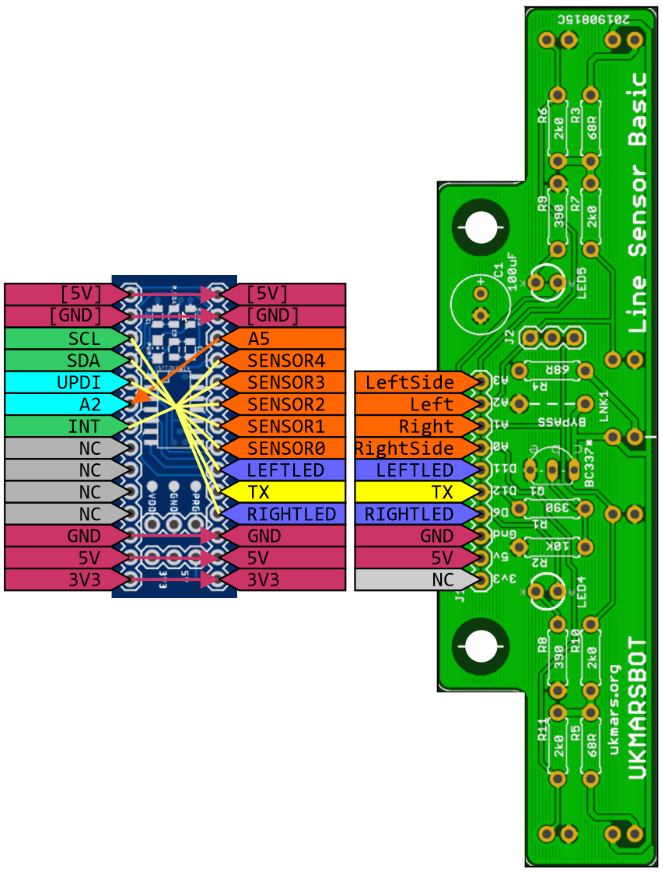
N/A

Example

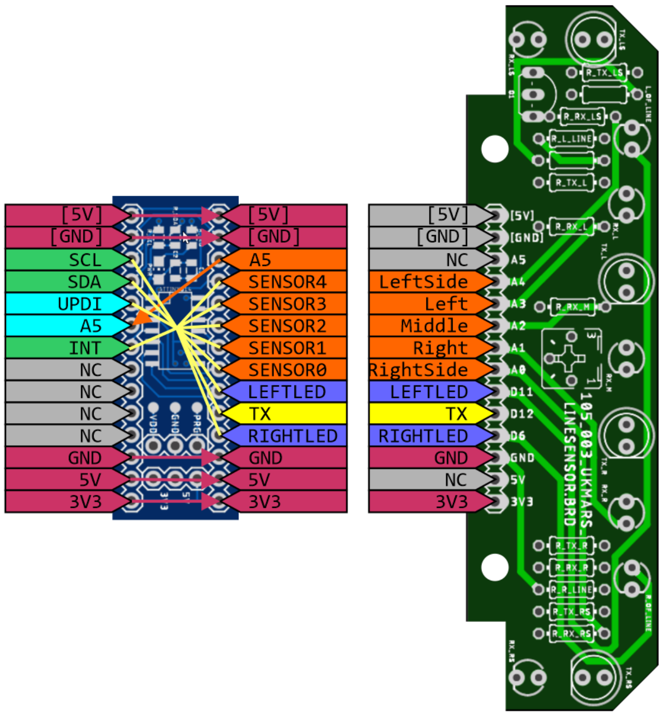
```
clearInterrupts(SENSOR0INTERRUPT + SENSOR1INTERRUPT);
//Clear down Sensor 0 and Sensor 1 Interrupt flags so they can trigger
again
```

4.0 QUICK REFERENCE –SENSOR BOARD CONNECTION

Basic Line Sensor Connections



SP Line Sensor Connections


[5V]
[GND]
SCL
SDA
UPDI
A5
INT
NC
NC
NC
NC
GND
5V
3V3
[5V]
[GND]
A5
SENSOR4
SENSOR3
SENSOR2
SENSOR1
SENSOR0
LEFTLED
TX
RIGHTLED
GND
5V
3V3
[5V]
[GND]
NC
LeftSide
Left
Middle
Right
RightSide
LEFTLED
TX
RIGHTLED
GND
NC
3V3
105_003_UKMARS
LINESENSOR.BRD