

RM3100 Arduino Quick Guide

Table of Contents

1	RM3100 ARDUINO OVERVIEW	2
2	HARDWARE CONNECTION AND SET UP	2
3	INSTALL ARDUINO IDE SOFTWARE	7
4	RUN RM3100 ARDUINO CODE.....	9

<i>Version #</i>	<i>Changes</i>	<i>Date</i>
Draft	Initial Version Draft	11/24/2021

1 RM3100 ARDUINO OVERVIEW

PNI developed a demo program using the Arduino IDE on Win PC for customers to get started using the RM3100 with an Arduino. RM3100 supports I2C and SPI, it's up to customer to decide which interface to use. Connection between RM3100 and Arduino can be either I2C or SPI, Arduino as master and RM3100 as slave for both.

To run this demo program, following is the minimum requirement.

Hardware

RM3100 Evaluation Board or Break Board

Any Arduino (In this document, we used the [NUCLEO-L152RE](#) which supports Arduino)
Win7 or later PC

Software

[Arduino IDE](#)

2 HARDWARE CONNECTION AND SET UP

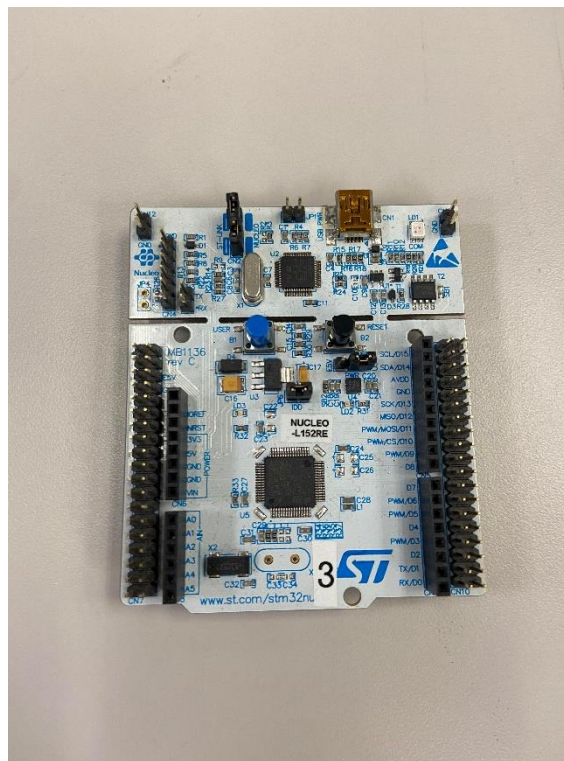
- I. RM3100 Test Boards Pin Assignments (Pin numbers run counterclockwise from top left)
Note: Pin 1 is next to the top left white triangle.

Pin#	Pin Name	Description
1	<u>SCK /</u> SCL	<u>SPI interface (SCK) – Serial clock input</u> I ² C interface (SCL) – Serial clock line
2	<u>SO /</u> SA1	<u>SPI interface (SO) – Master Input, Slave Output</u> I ² C interface – Bit 1 of slave address
3	<u>SI /</u> SDA	<u>SPI interface (SI) – Master Output, Slave Input Serial Data</u> I ² C interface (SDA) – Serial Data Line
4	<u>SSN /</u> SA0	<u>SPI interface – Active low to select port</u> I ² C interface – Bit 0 of slave address
5	DRDY	Status line
7	AVSS	Ground pin for analog section of ASIC
10	I2CEN	I ² C enable pin (HIGH = I ² C, LOW = SPI)
12	DVDD	Supply voltage for digital section of ASIC.
13	AVDD	Supply voltage for analog section of ASIC
14	DVSS	Ground pin for digital section of ASIC
6, 8, 9, 11	NC	Do not connect

Picture of RM 3100 Breakout Board below:

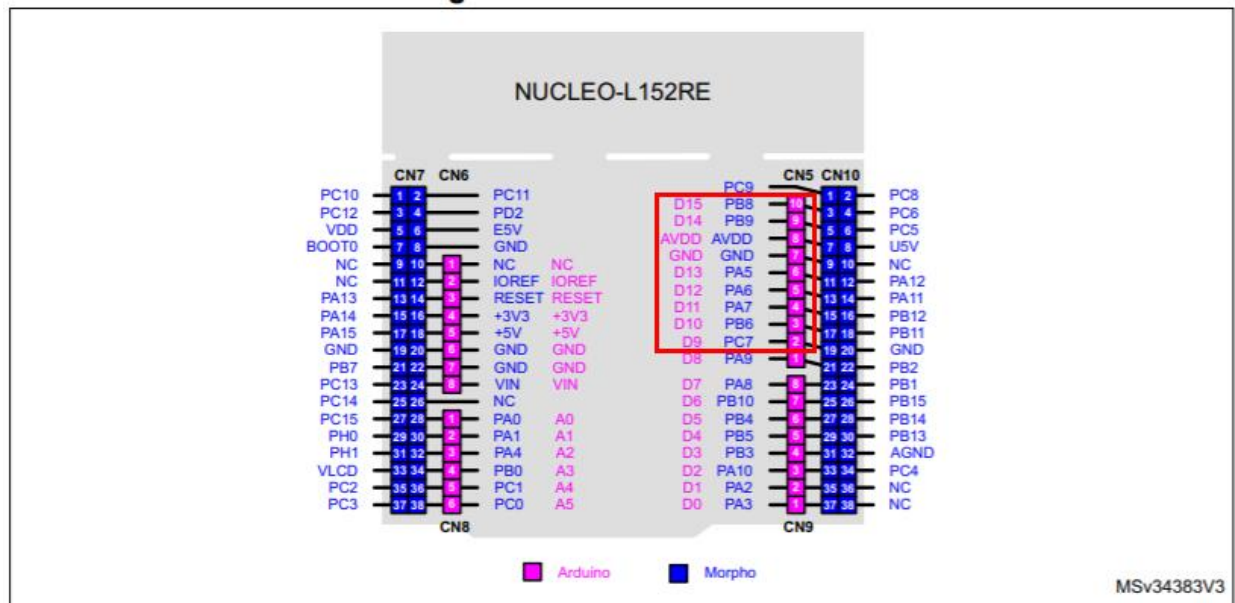


II. NUCLEO-L152RE (Arduino) SPI and I2C Pin Assignments



- a. Note: If you are using a regular Arduino board (Nano, Uno, Mega, etc.) you will have to find the correct corresponding pins from your specific board's data sheet as they will be different from this table below.

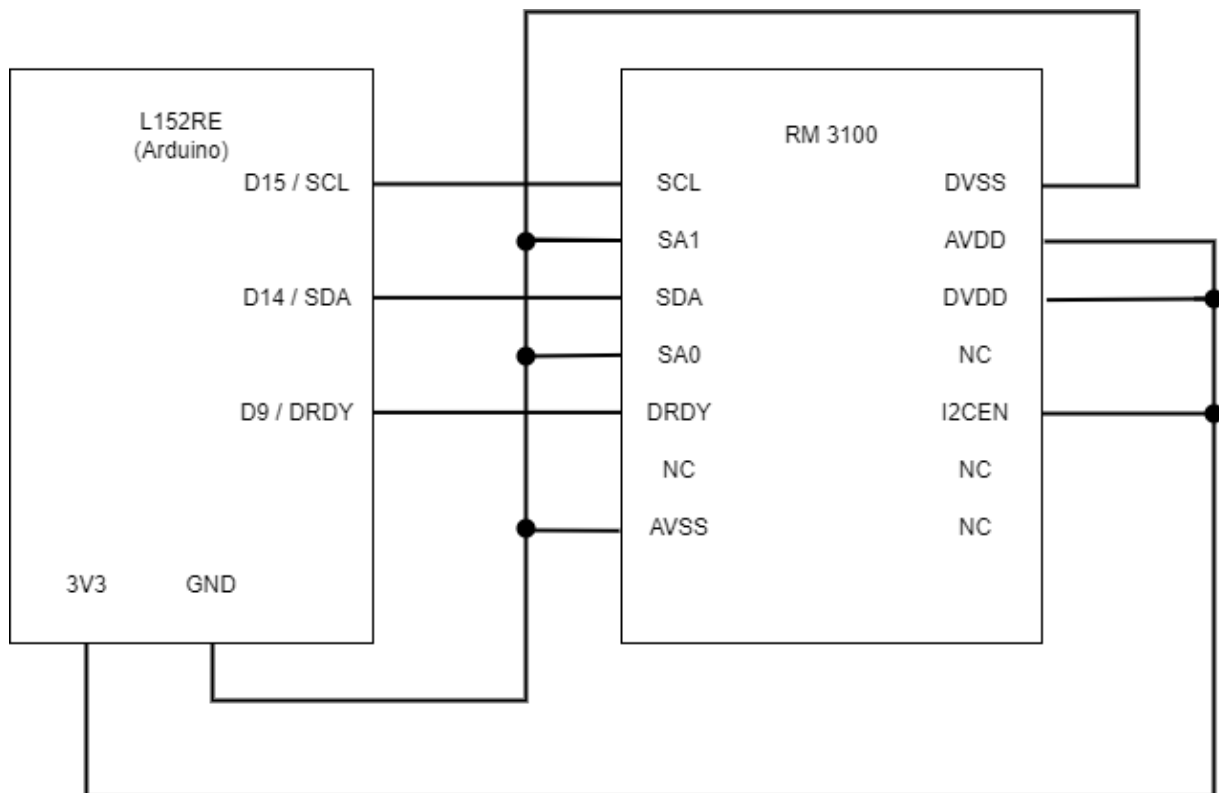
Pin#	Pin Name	Description
D15	SCL	I2C interface (SCL) – Serial clock line
D14	SDA	I2C interface (SDA) – Serial Data Line
AVDD	AVDD/+3V3	Power (+3.3 V)
GND	GND	Ground
D13	SCK	SPI Serial Clock
D12	MISO	SPI Master In Slave Out
D11	MOSI	SPI Master Out Slave In
D10	CS	Slave Select control line that allows slaves to be turned on and off via hardware control
D9	GPIO	General Purpose Input pin used for status line



Red box above highlights the pins used for I2C and SPI communications on NUCLEO-L152RE

III. I2C Interface. Connect RM3100 I2C pins to Arduino I2C pins, like in the following table.

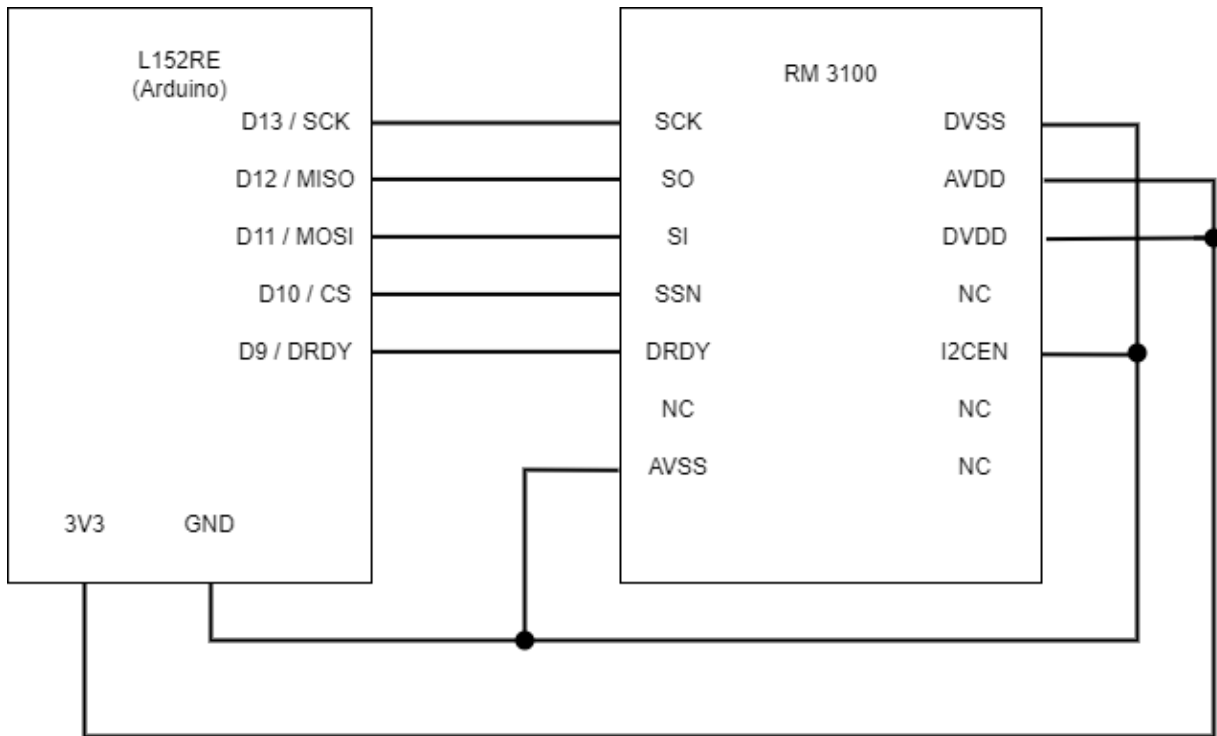
RM3100 I2C Pins			Arduino I2C
Pin#	Pin Name	Description	Pin#
1	SCL	I2C interface (SCL) – Serial clock line	D15 SCL
2	SA1	I2C interface – Bit 1 of slave address (LOW = 0)	GND
3	SDA	I2C interface (SDA) – Serial Data Line	D14 SDA
4	SA0	I2C interface – Bit 0 of slave address (LOW = 0)	GND
5	DRDY	Status line (Optional)	D9 (GPIO input)
7	AVSS	Ground pin for analog section of ASIC	GND
10	I2CEN	I2C enable pin (HIGH = I2C)	AVDD (+3V3)
12	DVDD	Supply voltage for digital section of ASIC	AVDD (+3V3)
13	AVDD	Supply voltage for analog section of ASIC	AVDD (+3V3)
14	DVSS	Ground pin for digital section of ASIC	GND
6, 8, 9, 11	NC	Do not connect	



Note: If you are using I2C, you may have to enable the internal pullup resistors in the code for your specific board. The sample code successfully enables pullup resistors for the NUCLEO-L152RE board specifically and hasn't been tested with other boards such as the Arduino Uno or Arduino Mega. You can also use external 4.7k Ohm pull up resistors on the SCL and SDA lines instead of the internal resistors.

- IV. If SPI interface is desired, please connect RM3100 SPI pins to Arduino SPI pins, like in the following table.

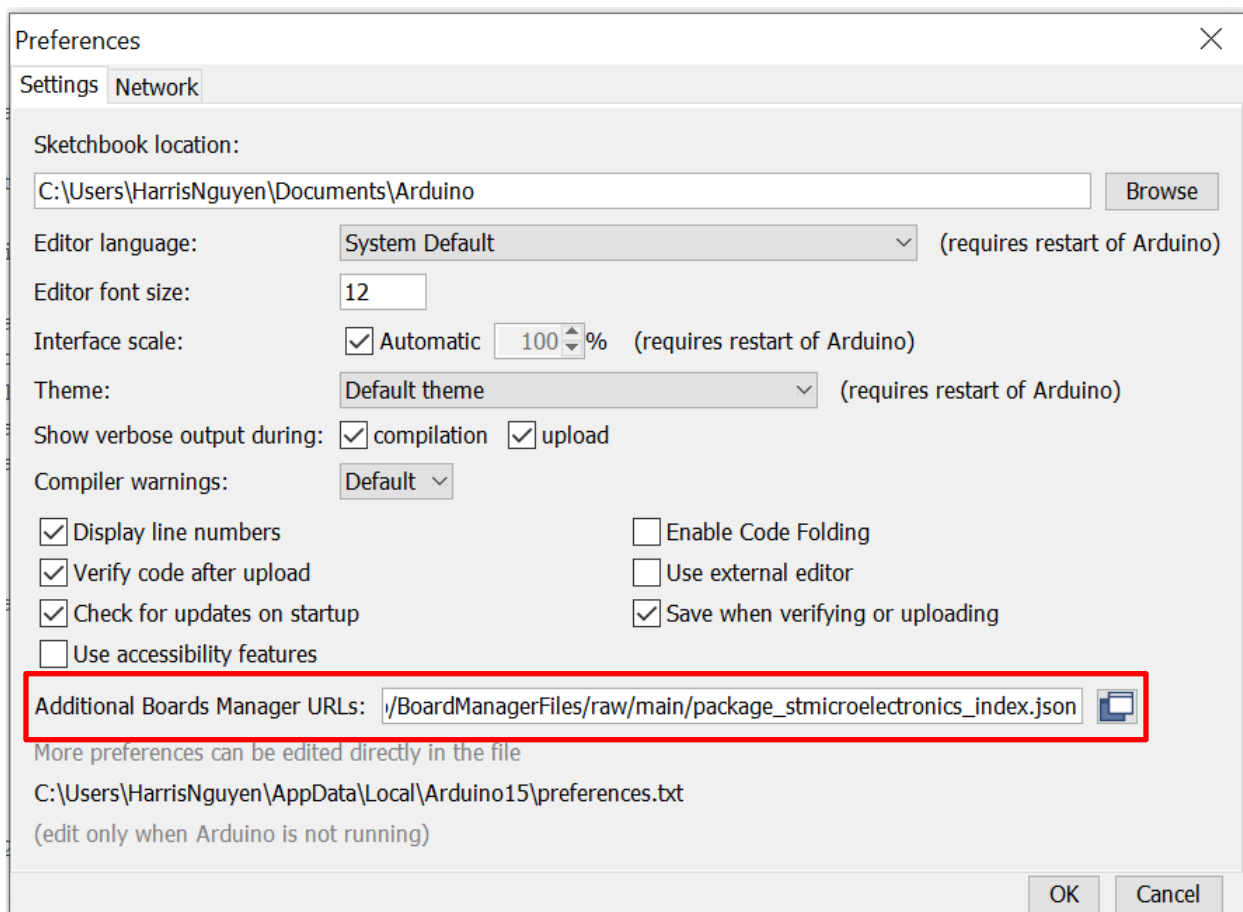
RM3100 SPI Pins			Arduino SPI
Pin#	Pin Name	Description	Pin#
1	SCK	SPI interface (SCK) – Serial clock input	D13 SCK
2	SO	SPI interface (SO) – Master Input, Slave Output	D12 MISO
3	SI	SPI interface (SI) – Master Output, Slave Input	D11 MOSI
4	SSN	SPI interface – Active low to select port	D10 SS
5	DRDY	Status line (Optional)	D9 (GPIO input)
7	AVSS	Ground pin for analog section of ASIC	GND
10	I2CEN	I2C enable pin (HIGH = I2C, LOW=SPI)	GND
12	DVDD	Supply voltage for digital section of ASIC	AVDD (+3V3)
13	AVDD	Supply voltage for analog section of ASIC	AVDD (+3V3)
14	DVSS	Ground pin for digital section of ASIC	GND
6, 8, 9, 11	NC	Do not connect	



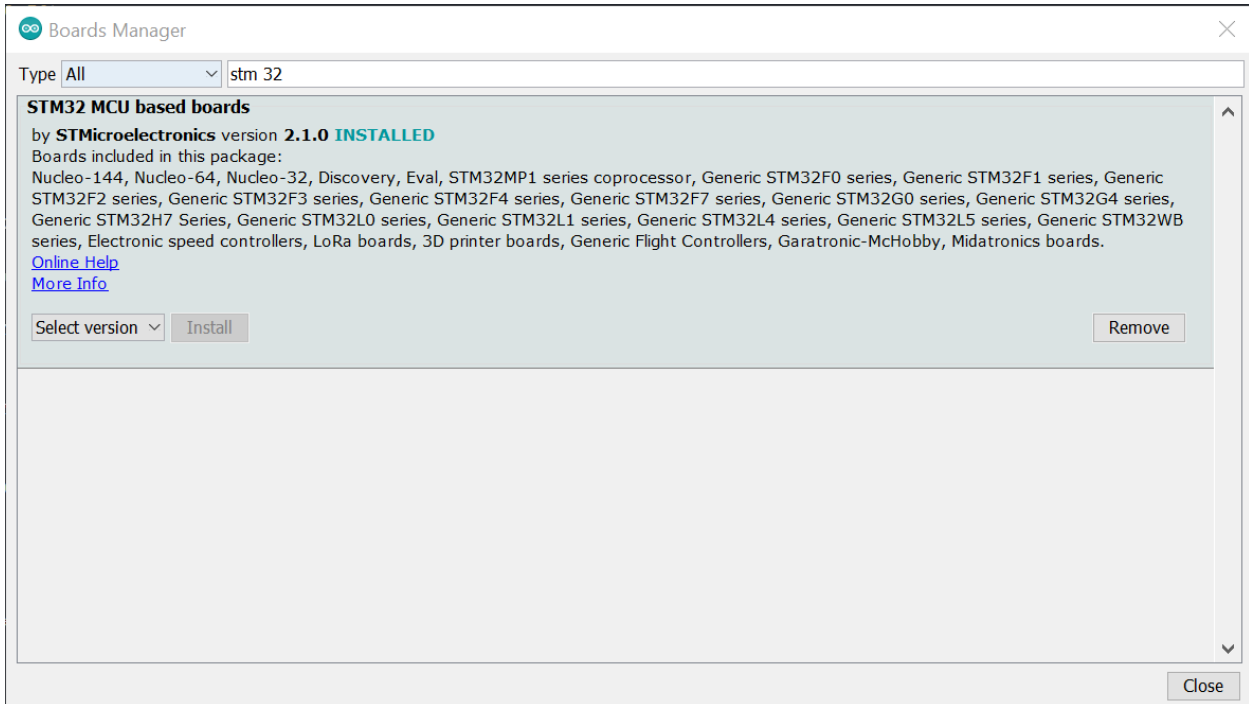
- V. Plug in Arduino USB cable to the USB port on your PC

3 INSTALL ARDUINO IDE SOFTWARE

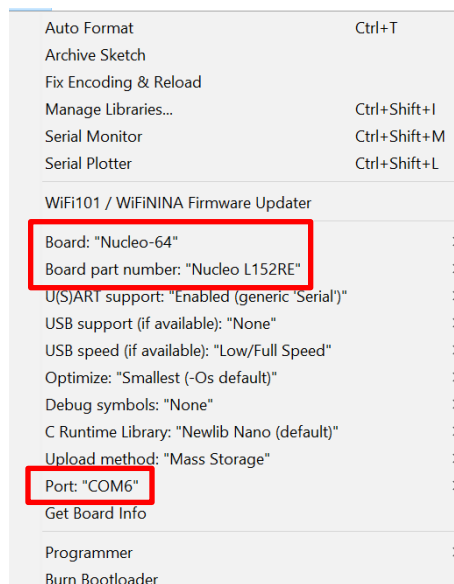
- I. To use the demo code, you will first need to have the Arduino IDE installed. If you don't have it installed, here is a link to [version 1.8.16 of Arduino IDE](#). You can also check [the IDE downloads page](#) to see if there are any newer updates but the demo code was written on version 1.8.16, so it may be incompatible with newer versions. Open the *.exe and install the program using any settings you want.
- II. If you are using a STM 32 board (like the NUCLEO-L152RE in this guide) then you must install the STM 32 board package first. If you are using a regular Arduino board (UNO, Mega, etc.) you can skip to step III.
 - a. Open the Arduino IDE program and in the toolbars press File > Preferences
 - b. In "Additional Boards Manager URLs:" copy and paste:
https://github.com/stm32duino/BoardManagerFiles/raw/main/package_stmicroelectronics_index.json and press OK to close the Preferences window



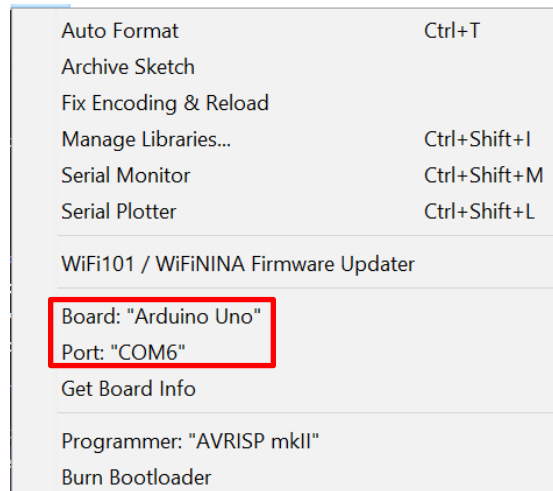
- c. In the toolbars press Tools > Board: > Boards Manager
- d. Search “stm 32” and install the latest version of “STM32 MCU based boards” then close the Boards Manager page



- e. Now select the corresponding NUCLEO board group in Tools > Board: and then the specific board in Tools > Board part number:
- f. Finally while the Arduino is connected to the PC, press Tools > Port and pick the USB port the Arduino is connected to (this should be something similar to “COM 6” or “COM 3”). Your “Tools” dropdown should look something like this:



- III. If you are using a standard Arduino (Nano, Uno, Mega, etc.), open the Arduino IDE program and in the tool bar press Tools > Boards and choose your specific Arduino board. Also, while the Arduino is connected to the PC, press Tools > Port and pick the USB port the Arduino is connected to (this should be something like “COM 6” or “COM 3”). Your “Tools” dropdown should look something like this:



4 RUN RM3100 ARDUINO CODE

- I. In the Arduino IDE, File > Open “**RM3100_Arduino_I2C.ino**” or “**RM3100_Arduino_SPI.ino**” file depending on which communication setup you have decided on. The code should open and look like this:

A screenshot of the Arduino IDE interface. The title bar shows 'RM3100_Arduino_I2C | Arduino 1.8.16'. The menu bar includes 'File', 'Edit', 'Sketch', 'Tools', and 'Help'. The toolbar has icons for file operations and execution. The code editor shows the following code:

```
1 #include <Arduino.h>
2 #include <Wire.h>
3
4 #define RM3100Address 0x20 // Hexadecimal slave address for RM3100 with Pin 2 and Pin 4 set to LOW
5
6 #define PIN_DRDY 9 //Set pin D9 to be the Data Ready Pin
7
8 //internal register values without the R/W bit
9 #define RM3100_REVID_REG 0x36 // Hexadecimal address for the Revid internal register
10 #define RM3100_POLL_REG 0x00 // Hexadecimal address for the Poll internal register
11 #define RM3100_CMM_REG 0x01 // Hexadecimal address for the CMM internal register
12 #define RM3100_STATUS_REG 0x34 // Hexadecimal address for the Status internal register
13 #define RM3100_CCK1_REG 0x04 // Hexadecimal address for Cycle Count X1 internal register
14 #define RM3100_CCK0_REG 0x05 // Hexadecimal address for the Cycle Count X0 internal register
15
16 //options
17 #define initialCC 200 // Set the cycle count
18 #define singleMode 1 //0 = use continuous measurement mode; 1 = use single measurement mode
19 #define useDRDYPin 1 //0 = not using DRDYPin ; 1 = using DRDYPin to wait for data
20
21 uint8_t revid;
22 uint16_t cycleCount;
23 float gain;
24
25 //Enable Pullup Resistors on Nucleo-L152RE I2C Pins (Pins PB_8/D14 and PB_9/D15)
26 const PinMap PinMap_I2C_SDA[] = {
27   {PB_9, I2C1, STM_PIN_DATA(STM_MODE_AF_OD, GPIO_PULLUP, GPIO_AF4_I2C1)},
28   {PB_8, I2C1, STM_PIN_DATA(STM_MODE_AF_OD, GPIO_PULLUP, GPIO_AF4_I2C1)},
29   {0, 0, 0}
28 }
```

The 'options' section (lines 16-19) is highlighted with a red rectangle. The status bar at the bottom shows 'Nucleo-64, Nucleo L152RE, Mass Storage, Enabled (generic 'Serial'), None, LowFull Speed, Smallest (-Os default), None, Newlib Nano (default) on COM6'.

- II. The red box above highlights the options you can change.
- a. The option `initialCC` is the cycle count you want the RM 3100 to measure in.
 - i. The internal clock count of RM3100 MagI2C ASIC establishes the number of sensor oscillation cycles. The number of oscillation cycle is “Cycle Count”. Cycle Count as default value is 200 or 0xC8 in hex.
 - ii. Increasing the cycle count value increases measurement gain and resolution. Lowering the cycle count value reduces acquisition time, which increases maximum achievable sample rate or, with a fixed sample rate, decreases power consumption.
 - iii. The minimum value is ‘0’ and the maximum is 65,536. However, quantization issues generally dictate working above a cycle count value of ~30, while noise limits the useful upper range to ~400 cycle counts.
 - b. The option “`singleMode`” is used to decide which measurement mode to use: single measurement mode or continuous measurement mode
 - i. Setting `singleMode` to “0” will enable continuous measurement mode
 - ii. Setting `singleMode` to “1” will enable single measurement mode
 - c. The option “`useDRDYPin`” is used to decide which method to use to wait for data to be ready
 - i. Setting “`useDRDYPin`” to “0” will make the Arduino read from the internal status register to see if data is ready
 - ii. Setting “`useDRDYPin`” to “1” will make the Arduino read from the `Pin_DRDY` (pin D9) to see if data is ready
 - 1. If you use the DRDY pin then you must have pin D9 connected to RM 3100 pin 5 (DRDY)

- III. Once you have the desired options selected, press upload button (Outlined in red in the top left of the image below) to verify and upload the code on to the Arduino
- You can also use “Ctrl + U” or press Sketch > Upload to upload



RM3100_Arduino_I2C | Arduino 1.8.16

File Edit Sketch Tools Help

RM3100_Arduino_I2C

```

4 #define RM3100Address 0x20 // Hexadecimal slave address for RM3100 with Pin 2 and Pin 4 set to LOW
5
6 #define PIN_DRDY 9 //Set pin D9 to be the Data Ready Pin
7
8 //internal register values without the R/W bit
9 #define RM3100_REVID_REG 0x36 // Hexadecimal address for the Revid internal register
10 #define RM3100_POLL_REG 0x00 // Hexadecimal address for the Poll internal register
11 #define RM3100_CMM_REG 0x01 // Hexadecimal address for the CMM internal register
12 #define RM3100_STATUS_REG 0x34 // Hexadecimal address for the Status internal register
13 #define RM3100_CX1_REG 0x04 // Hexadecimal address for Cycle Count X1 internal register
14 #define RM3100_CX0_REG 0x05 // Hexadecimal address for the Cycle Count X0 internal register
15
16 //options
17 #define initialCC 200 // Set the cycle count
18 #define singleMode 0 //0 = use continuous measurement mode; 1 = use single measurement mode
19 #define useDRDYPin 1 //0 = not using DRDYPin ; 1 = using DRDYPin to wait for data
20
21 uint8_t revid;
22 uint16_t cycleCount;
23 float gain;
24
25 //Enable Pullup Resistors on Nucleo-L152RE I2C Pins (Pins PB_8/D14 and PB_9/D15)
26 const PinMap PinMap_I2C_SDA[] = {
27   {PB_9, I2C1, STM_PIN_DATA(STM_MODE_AF_OD, GPIO_PULLUP, GPIO_AF4_I2C1)},
28   {NC, NP, 0}
29 };
30

```

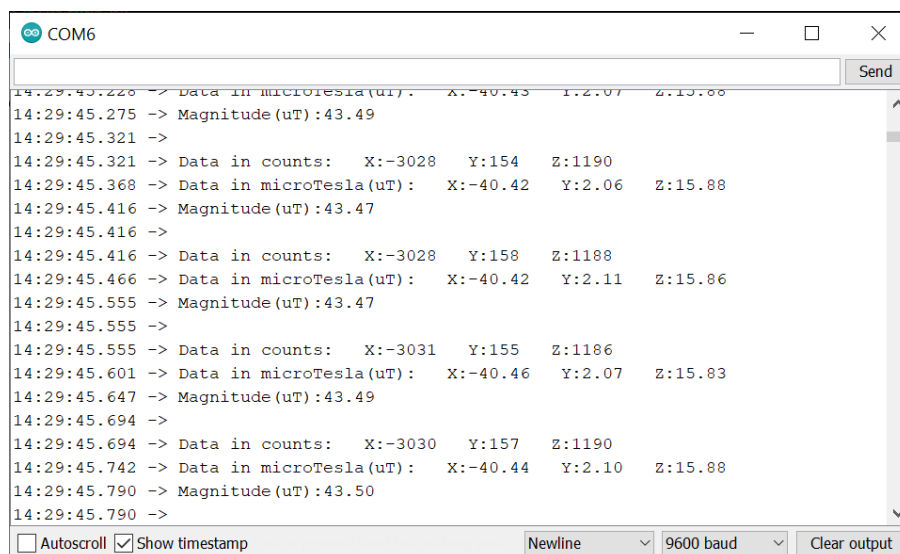
Save Canceled.

1 File(s) copied

Upload complete on NODE_L152RE (D:)

19 Nucleo-64, Nucleo L152RE, Mass Storage, Enabled (generic 'Serial'), None, Low/Full Speed, Smallest (-Os default), None, Newlib (Nano (default) on COM6

- IV. After the code is uploaded to the Arduino, press the orange highlighted button in the top right of the image above or press Tools > Serial Monitor or “Ctrl + Shift + M” to open the serial monitor. This is where the output of the RM 3100 will be printed



COM6

Send

```

14:29:45.228 -> Data in microTesla(uT): X:-40.43 Y:2.07 Z:15.88
14:29:45.275 -> Magnitude(uT):43.49
14:29:45.321 ->
14:29:45.321 -> Data in counts: X:-3028 Y:154 Z:1190
14:29:45.368 -> Data in microTesla(uT): X:-40.42 Y:2.06 Z:15.88
14:29:45.416 -> Magnitude(uT):43.47
14:29:45.416 ->
14:29:45.416 -> Data in counts: X:-3028 Y:158 Z:1188
14:29:45.466 -> Data in microTesla(uT): X:-40.42 Y:2.11 Z:15.86
14:29:45.555 -> Magnitude(uT):43.47
14:29:45.555 ->
14:29:45.555 -> Data in counts: X:-3031 Y:155 Z:1186
14:29:45.601 -> Data in microTesla(uT): X:-40.46 Y:2.07 Z:15.83
14:29:45.647 -> Magnitude(uT):43.49
14:29:45.694 ->
14:29:45.694 -> Data in counts: X:-3030 Y:157 Z:1190
14:29:45.742 -> Data in microTesla(uT): X:-40.44 Y:2.10 Z:15.88
14:29:45.790 -> Magnitude(uT):43.50
14:29:45.790 ->

```

☐ Autoscroll ☒ Show timestamp Newline 9600 baud Clear output