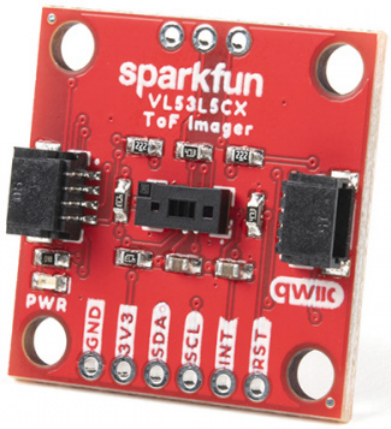
Qwiic ToF Imager - VL53L5CX Hookup Guide

Introduction

The Qwiic ToF Imager - VL53L5CX is here! This little breakout board is built around ST Electronics' VL53L5CX; a state of the art, Time-of-Flight (ToF), multizone ranging sensor enhancing the ST FlightSense product family. This chip integrates a SPAD array, physical infrared filters, and diffractive optical elements (DOE) to achieve the best ranging performance in various ambient lighting conditions with a range of cover glass materials.

Multizone distance measurements are possible up to 8x8 zones with a wide 63° diagonal FoV which can be reduced by software. Thanks to ST Histogram patented algorithms, the VL53L5CX is able to detect different objects within the FoV. The Histogram also provides immunity to cover glass crosstalk beyond 60 cm.

Ideal for 3D room mapping, obstacle detection for robotics, gesture recognition, IoT, laser-assisted autofocus, and AR/VR enhancement, the Qwiic connector on this sensor makes integration easy.



SparkFun Qwiic ToF Imager - VL53L5CX
SEN-18642

Product Showcase: SparkFun Time of Flight Sensor

Required Materials

To follow along with this tutorial, you will need the following materials. You may not need everything though depending on what you have. Add it to your cart, read through the guide, and adjust the cart as necessary.

Qwiic ToF Imager - VL53L5CX Hookup Guide Wish List SparkFun Wish List

SparkFun Qwiic ToF Imager - VL53L5CX
SEN-18642

Flexible Qwiic Cable - 100mm
PRT-17259

SparkFun Thing Plus - Artemis
WRL-15574

Suggested Reading

If you aren't familiar with the Qwiic system, we recommend reading here for an overview.

Qwiic Connect System

We would also recommend taking a look at the following tutorials if you aren't familiar with them.

Logic Levels
Learn the difference between 3.3V and 5V devices and logic levels.

I2C
An introduction to I2C, one of the main embedded communications protocols in use today.

How to Work with Jumper Pads and PCB Traces
Handling PCB jumper pads and traces is an essential skill. Learn how to cut a PCB trace, add a solder jumper between pads to reroute connections, and repair a trace with the green wire method if a trace is damaged.

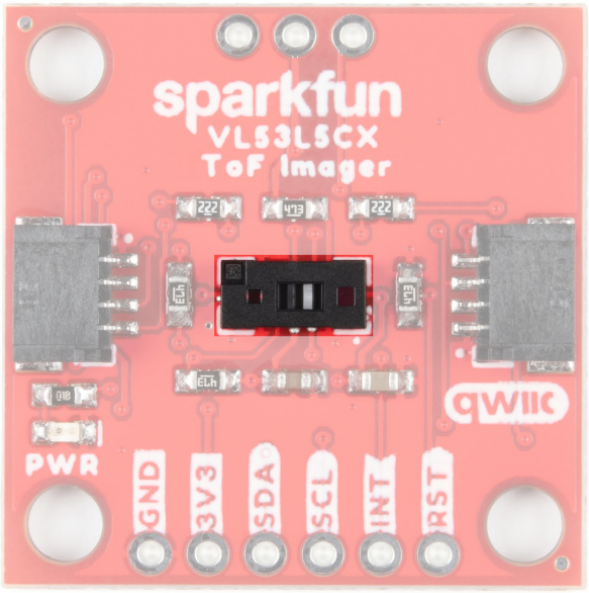
Hookup Guide for the SparkFun Artemis Thing Plus
Get started with our SparkFun Artemis Thing Plus - our popular Thing Plus footprint with the powerful Artemis module for ultimate functionality.

Hardware Overview

VL53L5CX

The SparkFun Qwiic ToF Imager is a state of the art, 64 pixel Time-of-Flight (ToF) 4 meter ranging sensor built around the VL53L5CX from ST. To see more details, refer to the datasheet.

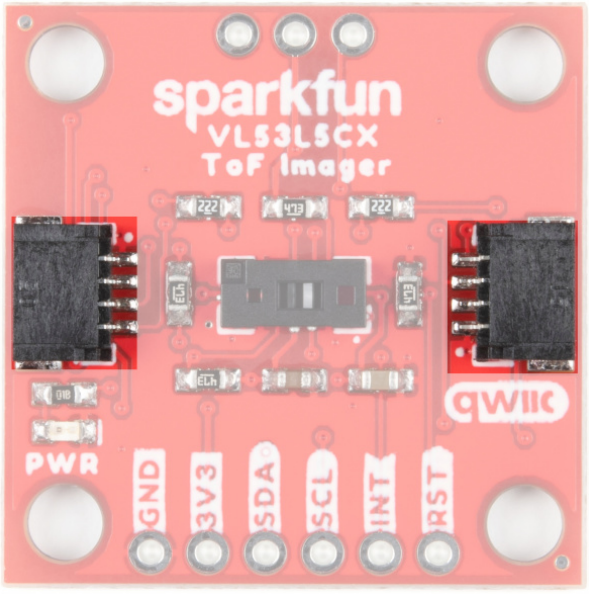
> ⚠ **Note:** This board ships with vacuum tape over the sensor. Please be sure to remove this tape before first use!



Qwiic Connectors

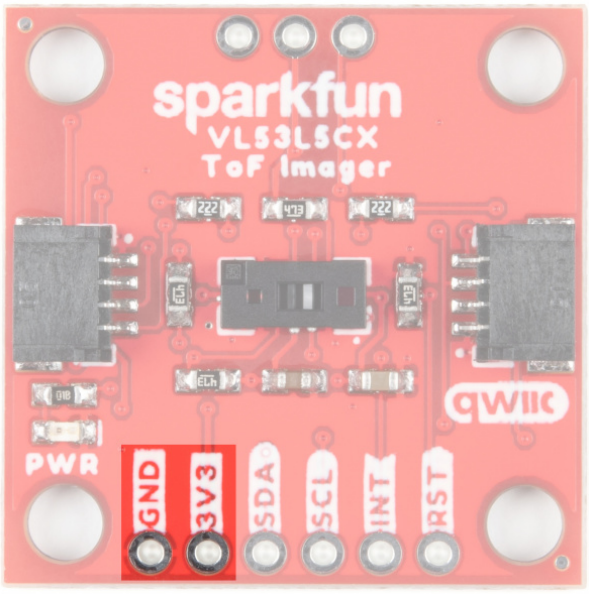
Our Qwiic Ecosystem makes sensors pretty much plug and play. There are two Qwiic connectors on either side of the Qwiic Air Velocity Sensor board to provide power and I^{2}C connectivity simultaneously.

The 7-bit I^{2}C address (most commonly used with Arduino) is **0x29**. The unshifted 8-bit I^{2}C address of the board is **0x52**.



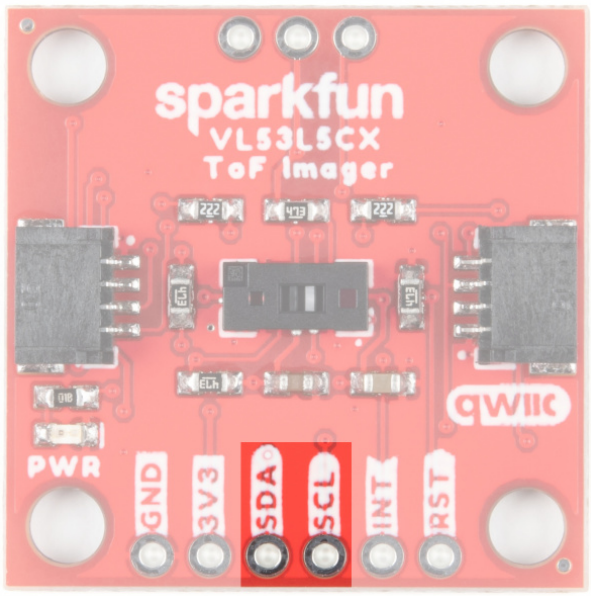
Power

Ideally power will be supplied by the Qwiic connector, but if you wish to supply your own power, pins have been broken out along the bottom side of the board labeled `3V3` and `GND`. The input voltage range should be between **2.7-3.3V**.



I^{2}C

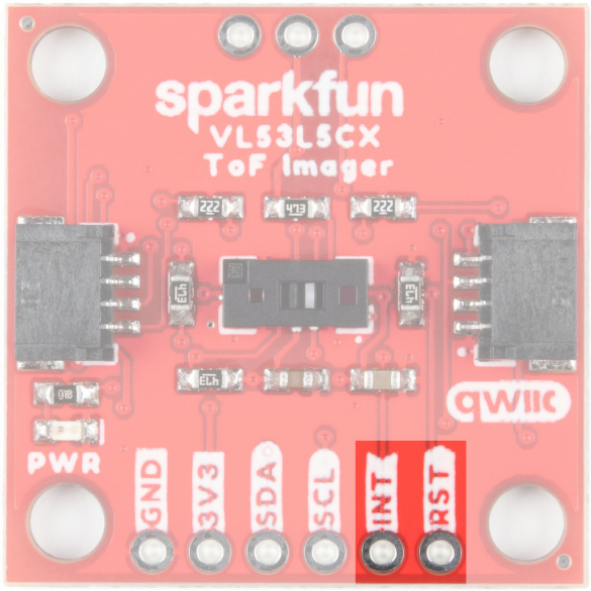
The I^{2}C pins break out the functionality of the Qwiic connectors. Depending on your application, you can connect to these pins via the plated through holes for SDA and SCL.



INT and RST

The interrupt pin is the interrupt output and defaults to an open-drain output. A 47 kΩ pull-up resistor to IOVDD is included.

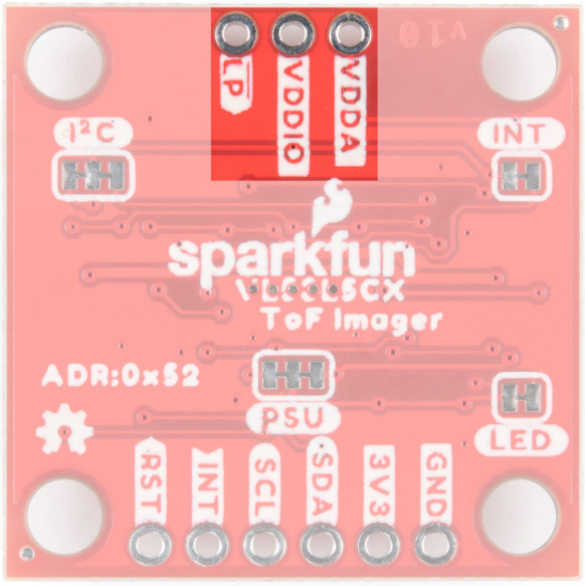
The reset pin is the I^{2}C interface reset pin and is active high. It is pulled to ground with a 47 kΩ resistor.



LP, VDDIO, & VDDA

LP is a *low power* enable pin. Drive this pin to logic 0 to disable the I^{2}C comms to reduce power consumption. Drive this pin to logic 1 to enable I^{2}C comms. This pin is typically only needed when you need to change the I2C address in multidevice systems. A 47 kΩ pull-up resistor to IOVDD is included so it can be left unconnected.

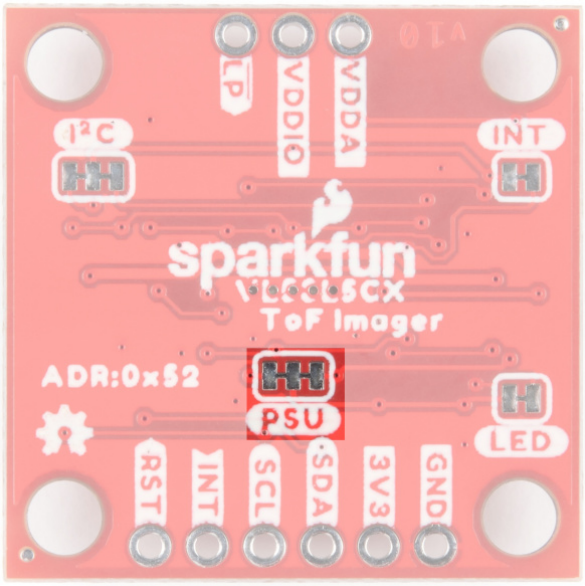
VDDIO/VDDA: These pins are used as an alternate power supply. By default, VDDIO and VDDA are tied together but by opening the PSU jumper they can be isolated. A user must then provide separate VDDIO and VDDA supplies. This is most applicable for users who want to use IO voltages (1.8, 2.8, or 3.3V) separate from AVDD voltages (2.8 or 3.3V) for maximum power reduction.



Jumpers

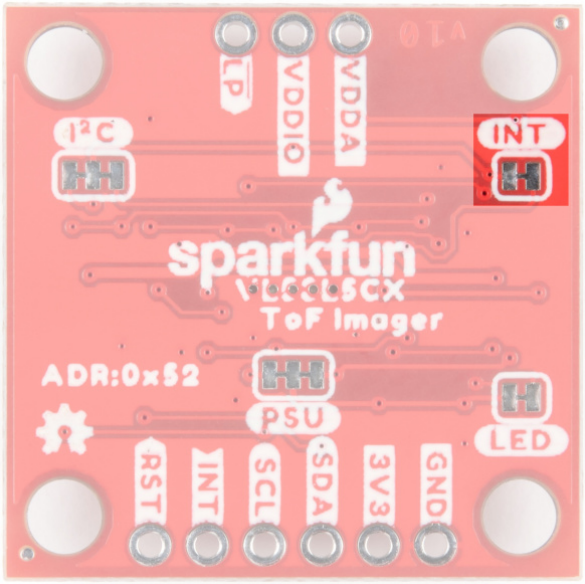
PSU

By default, VDDIO and VDDA are tied together. Cutting the **PSU** jumper will isolate the power rails. A user must then provide separate VDDIO and VDDA supplies. This is most applicable for users who want to use IO voltages (1.8, 2.8, or 3.3V) separate from AVDD voltages (2.8 or 3.3V) for maximum power reduction.



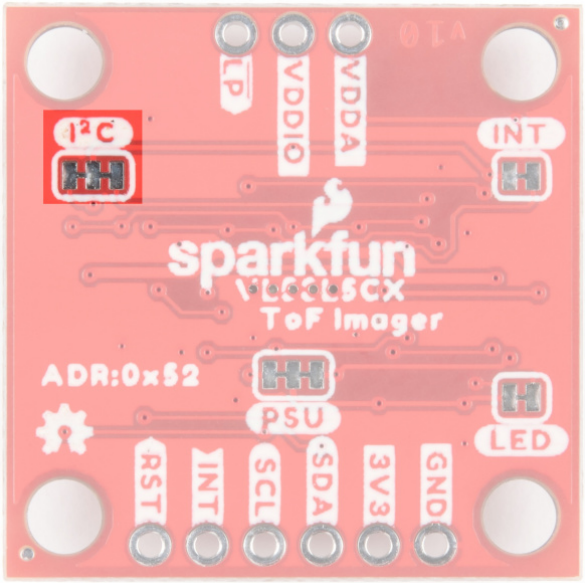
INT

Cut the **INT** jumper to remove the 47 kΩ pull-up resistor from the INT pin.



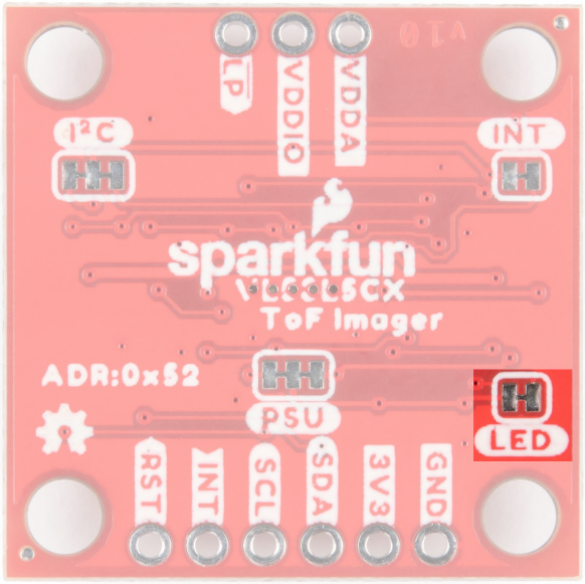
I²C

The SparkFun Qwiic ToF Imager Sensor has two 2.2 kΩ pull-up resistors attached to the I²C bus by default. If multiple sensors are connected to the bus with the pull-up resistors enabled the parallel equivalent resistance may create too strong of a pull-up for the bus to operate correctly. As a general rule of thumb, disable all but one pair of pull-up resistors if multiple devices are connected to the bus. If you need to disconnect the pull-up resistors they can be removed by cutting the traces on the corresponding jumper highlighted below.



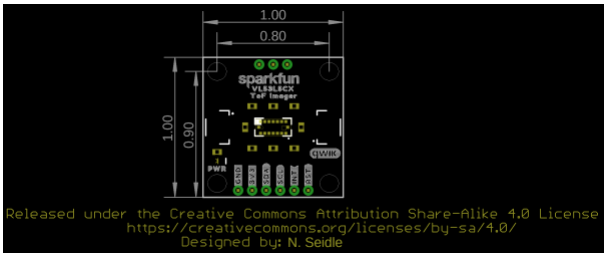
LED

If minimal power consumption is a concern, or you just don't want that Power LED on the front of the board to light up, go ahead and cut this jumper.



Board Outline

This sensor measures 1" x 1".

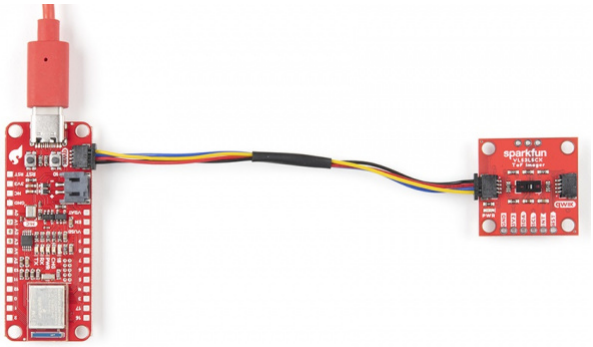


Click the image for a closer view

Hardware Hookup

⚠ **A note on choosing a board:** The VL53L5CX is unique in that it requires its firmware to be loaded at power-on over the I2C bus. Because this firmware is ~90k bytes, we recommend a microcontroller with enough flash to store VL53L5CX's firmware as well as your program code. Sorry, Uno's are out. But didn't you want an excuse to try out something new? We recommend choosing either an Artemis Thing Plus or an ESP32 Thing Plus board as your development board.

Using the Qwiic system, assembling the hardware is simple. All you need to do is connect your VL53L5CX Imager Breakout to your chosen development board with a Qwiic cable or adapter cable. If Qwiic is not your thing, or if your dev board doesn't have one built in you can always solder directly to the I^2C pins. If you are not using a Qwiic-enabled board, make sure your input voltage and logic are either running at **3.3V** or you are shifting the logic level from whatever logic your controller runs at to **3.3V**.



Click the image for a closer view

⚠ **Before Use:** Make sure to remove the vacuum tape from the VL53L5CX sensor before first use!

Software Setup and Programming

Note: Make sure you are using the latest stable version of the Arduino IDE on your desktop. If this is your first time using Arduino IDE, library, or board add-on, please review the following tutorials.

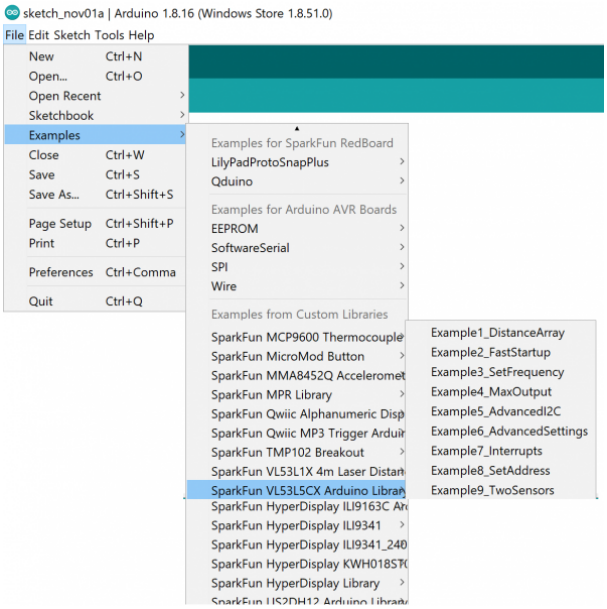
- Installing the Arduino IDE
- Installing an Arduino Library
- Installing Board Definitions in the Arduino IDE

We've written a simple Arduino library to quickly get started reading data from the Qwiic ToF Imager. Install the library through the Arduino Library Manager tool by searching for **"SparkFun VL53L5CX"**. Users who prefer to manually install it can get the library from the GitHub Repository or download the ZIP by clicking the button below:

SPARKFUN QWIIC TIME-OF-FLIGHT SENSOR VL53L5CX ARDUINO LIBRARY (ZIP)

Example1_DistanceArray

Hook up your ToF imager to your Artemis Thing Plus via the Qwiic cables, and click "**File** > **Examples** > **SparkFun VL53L5CX Arduino Library** > **Example1_DistanceArray**".

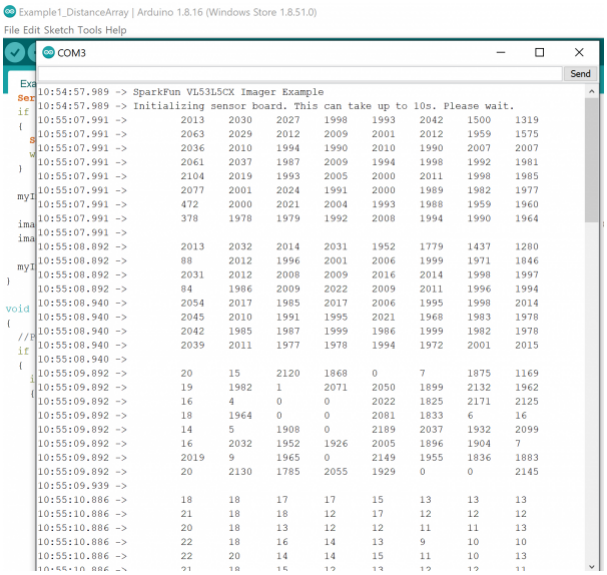


We'll assume that you have selected the board (in this case the **SparkFun Artemis Thing Plus**) and the correct COM port at this point. If you have the code open, hit the upload button. Otherwise, copy and paste the following into the Arduino IDE, make sure to select the correct board and COM port, and then upload:

```
/*
  Read an 8x8 array of distances from the VL53L5CX
  By: Nathan Seidle
  SparkFun Electronics
  Date: October 26, 2021
  License: MIT. See license file for more information but you can
  basically do whatever you want with this code.

  This example shows how to read all 64 distance readings at once.

  Feel like supporting our work? Buy a board from SparkFun!
  https://www.sparkfun.com/products/18642

*/

#include <Wire.h>

#include <SparkFun_VL53L5CX_Library.h> //http://librarymanager/All#SparkFun_VL53L5CX

SparkFun_VL53L5CX myImager;
VL53L5CX_ResultsData measurementData; // Result data class structure, 1356 byes of RAM

int imageResolution = 0; //Used to pretty print output
int imageWidth = 0; //Used to pretty print output

void setup()
{
  Serial.begin(115200);
  delay(1000);
  Serial.println("SparkFun VL53L5CX Imager Example");

  Wire.begin(); //This resets to 100kHz I2C

  Serial.println("Initializing sensor board. This can take up to 10s. Please wait.");
  if (myImager.begin() == false)
  {
    Serial.println(F("Sensor not found - check your wiring. Freezing"));
    while (1) ;
  }

  myImager.setResolution(8*8); //Enable all 64 pads

  imageResolution = myImager.getResolution(); //Query sensor for current resolution - either 4x4
or 8x8
  imageWidth = sqrt(imageResolution); //Calculate printing width

  myImager.startRanging();
}

void loop()
{
  //Poll sensor for new data
  if (myImager.isDataReady() == true)
```

```
  {
    if (myImager.getRangingData(&measurementData)) //Read distance data into array
    {
      //The ST library returns the data transposed from zone mapping shown in datasheet
      //Pretty-print data with increasing y, decreasing x to reflect reality
      for (int y = 0 ; y <= imageWidth * (imageWidth - 1) ; y += imageWidth)
      {
        for (int x = imageWidth - 1 ; x >= 0 ; x--)
        {
          Serial.print("\t");
          Serial.print(measurementData.distance_mm[x + y]);
        }
        Serial.println();
      }
      Serial.println();
    }
  }

  delay(5); //Small delay between polling
}
```

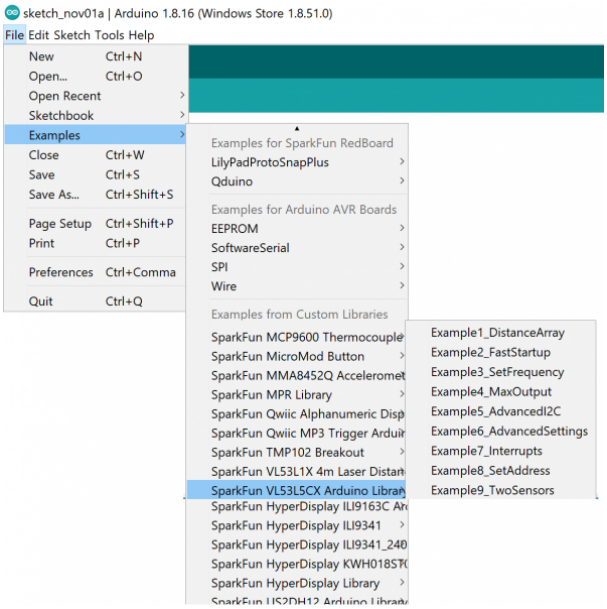
Open up your Serial Monitor, make sure the baud rate is set appropriately, and you should see something like the following:



Click the image for a closer view

Example2_FastStartup

Hook up your ToF imager to your Artemis Thing Plus via the Qwiic cables, and click "**File** > **Examples** > **SparkFun VL53L5CX Arduino Library** > **Example2_FastStartup**".



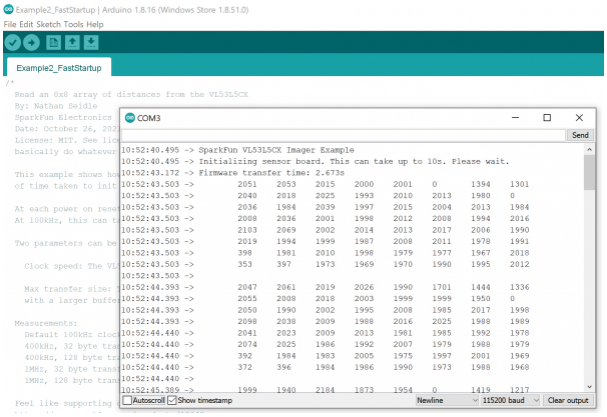
We'll assume that you have selected the board (in this case the **SparkFun Artemis Thing Plus**) and the correct COM port at this point. If you have the code open, hit the upload button. Otherwise, copy and paste the following into the Arduino IDE, make sure to select the correct board and COM port, and then upload:

```
/*
  Read an 8x8 array of distances from the VL53L5CX
  By: Nathan Seidle
  SparkFun Electronics
  Date: October 26, 2021
  License: MIT. See license file for more information but you can
  basically do whatever you want with this code.

  This example shows how to setup the I2C bus to minimize the amount
  of time taken to init the sensor.

  At each power on reset, a staggering 86,000 bytes of firmware have to be sent to the sensor.
  At 100kHz, this can take ~9.4s. By increasing the clock speed, we can cut this time down to ~
1.4s.

  Two parameters can be tweaked:

    Clock speed: The VL53L5CX has a max bus speed of 400kHz but we have had success up to 1MHz.

    Max transfer size: The majority of Arduino platforms default to 32 bytes. If you are using o
ne
    with a larger buffer (ESP32 is 128 bytes for example), this can help decrease transfer times
a bit.

  Measurements:
    Default 100kHz clock and 32 byte transfer: 9.4s
    400kHz, 32 byte transfer: 2.8s
    400kHz, 128 byte transfer: 2.5s
    1MHz, 32 byte transfer: 1.65s
    1MHz, 128 byte transfer: 1.4s

  Feel like supporting our work? Buy a board from SparkFun!
  https://www.sparkfun.com/products/18642

*/

#include <Wire.h>

#include <SparkFun_VL53L5CX_Library.h> //http://librarymanager/All#SparkFun_VL53L5CX

SparkFun_VL53L5CX myImager;
VL53L5CX_ResultsData measurementData; // Result data class structure, 1356 byes of RAM

int imageResolution = 0; //Used to pretty print output
int imageWidth = 0; //Used to pretty print output

void setup()
{
  Serial.begin(115200);
  delay(1000);
  Serial.println("SparkFun VL53L5CX Imager Example");

  Wire.begin(); //This resets I2C bus to 100kHz
```

```
  Wire.setClock(400000); //Sensor has max I2C freq of 400kHz
  //Wire.setClock(1000000); //Run sensor out of spec

  //myImager.setWireMaxPacketSize(128); //Increase default from 32 bytes to 128 - not supported
 on all platforms

  Serial.println("Initializing sensor board. This can take up to 10s. Please wait.");

  //Time how long it takes to transfer firmware to sensor
  long startTime = millis();
  bool startup = myImager.begin();
  long stopTime = millis();

  if (startup == false)
  {
    Serial.println(F("Sensor not found - check your wiring. Freezing"));
    while (1) ;
  }

  Serial.print("Firmware transfer time: ");
  float timeTaken = (stopTime - startTime) / 1000.0;
  Serial.print(timeTaken, 3);
  Serial.println("s");

  myImager.setResolution(8*8); //Enable all 64 pads

  imageResolution = myImager.getResolution(); //Query sensor for current resolution - either 4x4
or 8x8
  imageWidth = sqrt(imageResolution); //Calculate printing width

  myImager.startRanging();
}

void loop()
{
  //Poll sensor for new data
  if (myImager.isDataReady() == true)
  {
    if (myImager.getRangingData(&measurementData)) //Read distance data into array
    {
      //The ST library returns the data transposed from zone mapping shown in datasheet
      //Pretty-print data with increasing y, decreasing x to reflect reality
      for (int y = 0 ; y <= imageWidth * (imageWidth - 1) ; y += imageWidth)
      {
        for (int x = imageWidth - 1 ; x >= 0 ; x--)
        {
          Serial.print("\t");
          Serial.print(measurementData.distance_mm[x + y]);
        }
        Serial.println();
      }
      Serial.println();
    }
  }
```

```
  delay(5); //Small delay between polling
}
```

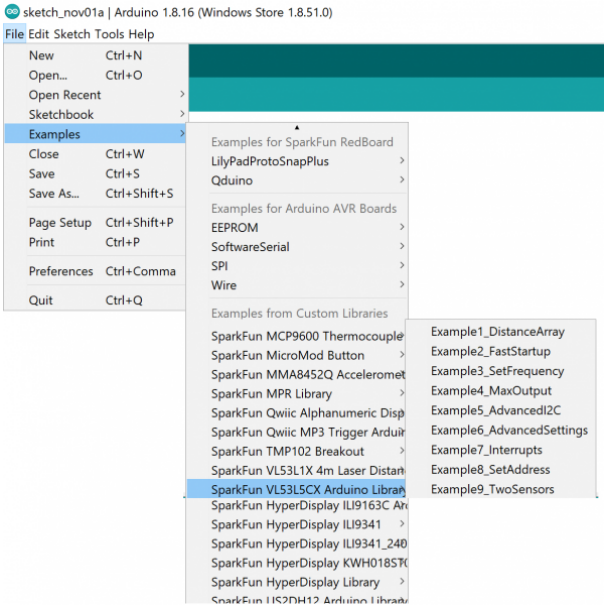
Open up your Serial Monitor, make sure the baud rate is set appropriately, and you should see something like the following:



Click the image for a closer view

Example3_SetFrequency

Hook up your ToF imager to your Artemis Thing Plus via the Qwiic cables, and click "**File** > **Examples** > **SparkFun VL53L5CX Arduino Library** > **Example3_SetFrequency**".

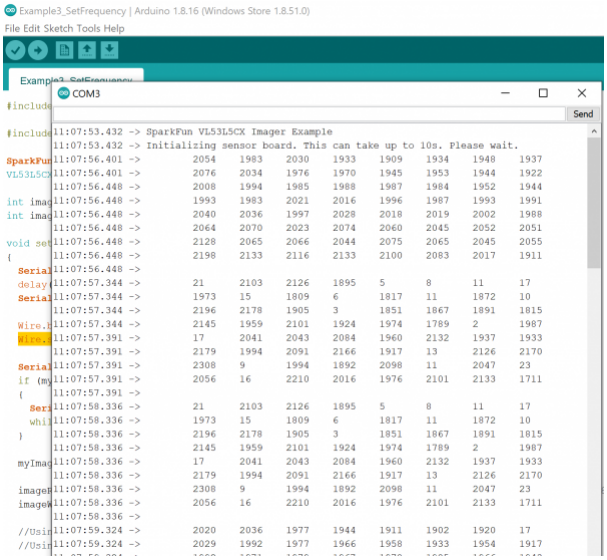


We'll assume that you have selected the board (in this case the **SparkFun Artemis Thing Plus**) and the correct COM port at this point. If you have the code open, hit the upload button. Otherwise, copy and paste the following into the Arduino IDE, make sure to select the correct board and COM port, and then upload:

```
/*
  Read an 8x8 array of distances from the VL53L5CX
  By: Nathan Seidle
  SparkFun Electronics
  Date: October 26, 2021
  License: MIT. See license file for more information but you can
  basically do whatever you want with this code.

  This example shows how to increase output frequency.

  Default is 1Hz.
  Using 4x4, min frequency is 1Hz and max is 60Hz
  Using 8x8, min frequency is 1Hz and max is 15Hz

  Feel like supporting our work? Buy a board from SparkFun!
  https://www.sparkfun.com/products/18642

*/

#include <Wire.h>

#include <SparkFun_VL53L5CX_Library.h> //http://librarymanager/All#SparkFun_VL53L5CX

SparkFun_VL53L5CX myImager;
VL53L5CX_ResultsData measurementData; // Result data class structure, 1356 byes of RAM

int imageResolution = 0; //Used to pretty print output
int imageWidth = 0; //Used to pretty print output

void setup()
{
  Serial.begin(115200);
  delay(1000);
  Serial.println("SparkFun VL53L5CX Imager Example");

  Wire.begin(); //This resets I2C bus to 100kHz
  Wire.setClock(400000); //Sensor has max I2C freq of 400kHz

  Serial.println("Initializing sensor board. This can take up to 10s. Please wait.");
  if (myImager.begin() == false)
  {
    Serial.println(F("Sensor not found - check your wiring. Freezing"));
    while (1) ;
  }

  myImager.setResolution(8 * 8); //Enable all 64 pads

  imageResolution = myImager.getResolution(); //Query sensor for current resolution - either 4x4
or 8x8
  imageWidth = sqrt(imageResolution); //Calculate printing width

  //Using 4x4, min frequency is 1Hz and max is 60Hz
  //Using 8x8, min frequency is 1Hz and max is 15Hz
```

```
  bool response = myImager.setRangingFrequency(15);
  if (response == true)
  {
    int frequency = myImager.getRangingFrequency();
    if (frequency > 0)
    {
      Serial.print("Ranging frequency set to ");
      Serial.print(frequency);
      Serial.println(" Hz.");
    }
    else
      Serial.println(F("Error recovering ranging frequency."));
  }
  else
  {
    Serial.println(F("Cannot set ranging frequency requested. Freezing..."));
    while (1) ;
  }

  myImager.startRanging();
}

void loop()
{
  //Poll sensor for new data
  if (myImager.isDataReady() == true)
  {
    if (myImager.getRangingData(&measurementData)) //Read distance data into array
    {
      //The ST library returns the data transposed from zone mapping shown in datasheet
      //Pretty-print data with increasing y, decreasing x to reflect reality
      for (int y = 0 ; y <= imageWidth * (imageWidth - 1) ; y += imageWidth)
      {
        for (int x = imageWidth - 1 ; x >= 0 ; x--)
        {
          Serial.print("\t");
          Serial.print(measurementData.distance_mm[x + y]);
        }
        Serial.println();
      }
      Serial.println();
    }
  }

  delay(5); //Small delay between polling
}
```

Open up your Serial Monitor, make sure the baud rate is set appropriately, and you should see something like the following:

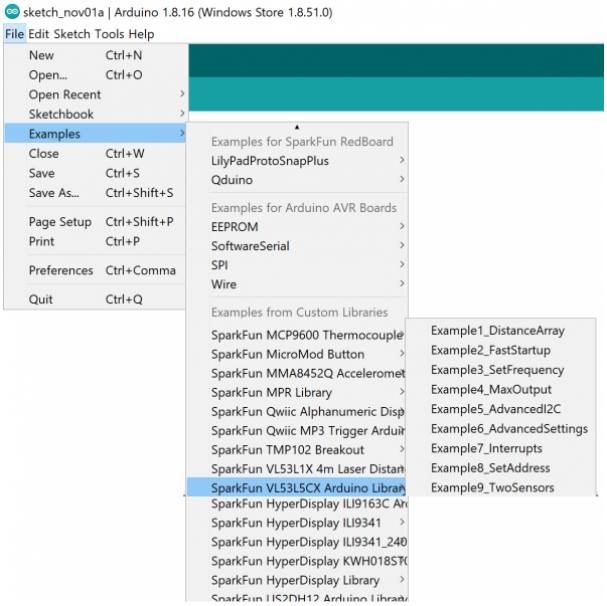


Click the image for a closer view

Visualizing the Output

To ‘see’ the output, *Example4_MaxOutput* can be used with the SparkFun Processing visualization app.

Hook up your ToF imager to your Artemis Thing Plus via the Qwiic cables, and click "**File** > **Examples** > **SparkFun VL53L5CX Arduino Library** > **Example4_MaxOutput**".



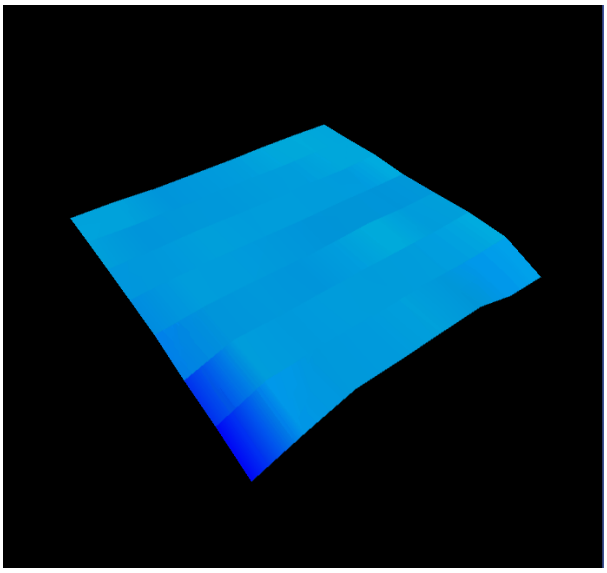
Load this sketch onto your platform and open the serial monitor. You should see the distance array output in CSV format.

Next, download the Processing sketch here and unzip it into a directory you can locate. If you don’t have it installed, download and unzip Processing into a directory of your choice. Open the *SparkFun VL53L5CX 3D Depth Map* sketch and modify the following line to match your COM port:

```
void setup(){

  size(700,700, P3D); // Make a 3D capable window

  port = new Serial(this, "COM13", 115200); // CHANGE COM13 TO YOUR SERIAL PORT
  port.bufferUntil(10); // ASCII LineFeed Character

  // Fill our list with 0s to avoid a null pointer exception
  // in case the sensor data is not immediately available
  for(int idx = 0; idx < 64; idx++){
    depths[idx] = 0;
  }
```

Modify this line to connect Processing to the sensor

Once connected you should see output like this:



Visualizing the distances

Our apologies if you suddenly realize an hour has gone by and you've done nothing but wave your hand in front of the sensor and looked at things like coffee cups from a meter away. It's really a lot of fun. Enjoy!

Troubleshooting

Need help?

If your product is not working as you expected or you need technical assistance or information, head on over to the SparkFun Technical Assistance page for some initial troubleshooting.

If you don't find what you need there, the SparkFun Forums are a great place to find and ask for help. If this is your first visit, you'll need to create a Forum Account to search product forums and post questions.

Resources and Going Further

Now that you've successfully got your Qwiic ToF sensor up and running, it's time to incorporate it into your own project!

For more information, check out the resources below:

- Schematic
- Eagle Files
- Datasheet
- Board Dimensions
- GitHub
- Library GitHub

Need some inspiration for your next project? Check out some of these related tutorials:

Displaying Your Coordinates with a GPS Module

This Arduino tutorial will teach you how to pinpoint and display your GPS coordinates with a press of a button using hardware from our Qwiic Connect System (I2C).

SparkFun Humidity Sensor Breakout - SHTC3 (Qwiic) Hookup Guide

A Hookup Guide to get started using the SHTC3 breakout.

MicroMod All The Pins (ATP) Carrier Board

Access All The Pins (ATP) of the MicroMod Processor Board with the Carrier Board!

MicroMod RP2040 Processor Board Hookup Guide

This tutorial covers the basic functionality of the MicroMod RP2040 Processor Board and highlights the features of the dual-core ARM Cortex-M0+ processors development board. Get started with the first microcontroller from the Raspberry Pi Foundation!