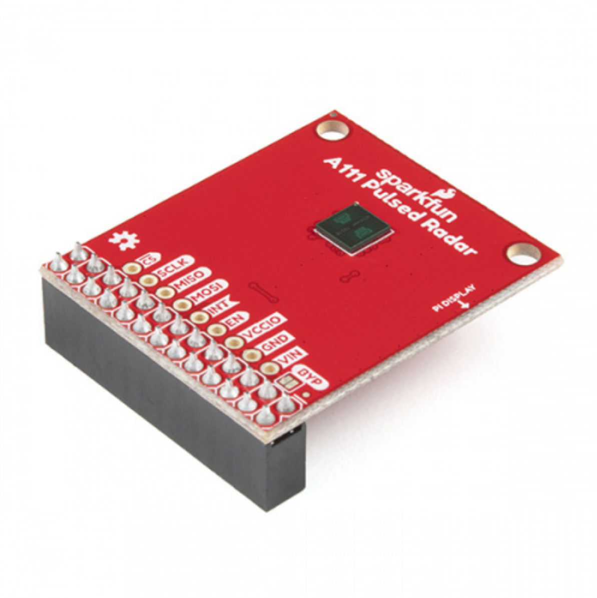
Getting Started with the A111 Pulsed Radar Sensor

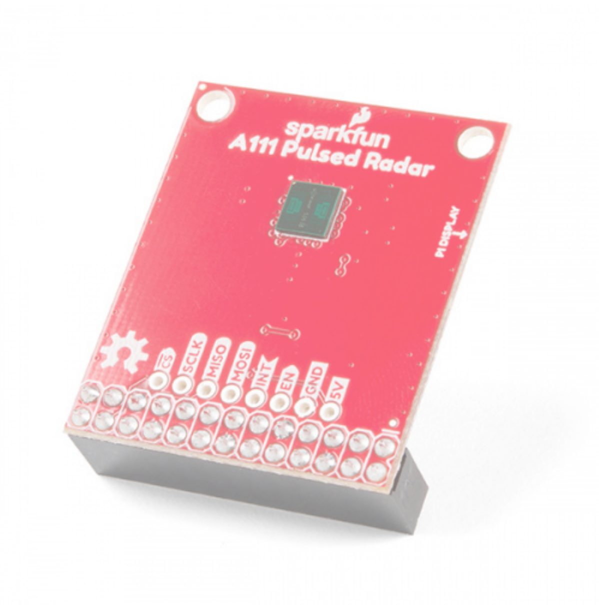
Introduction

Heads up! If you are looking for the hookup guide for the SparkX version of the A111, you can check out the retired tutorial: Using the A111 Pulsed Radar Sensor with a Raspberry Pi. However, the older tutorial is outdated due to the SDK version used in the previous guide.

Does your project require high-precision, cutting-edge distance, speed, motion, and/or gesture sensing? We're not talking ultrasonic, or even infrared here, but 60GHz radar! Say hello to our tiny, pulsed radar friend the Acconeer A111!



SparkFun Pulsed Radar Breakout - A111
⊖ SEN-16826

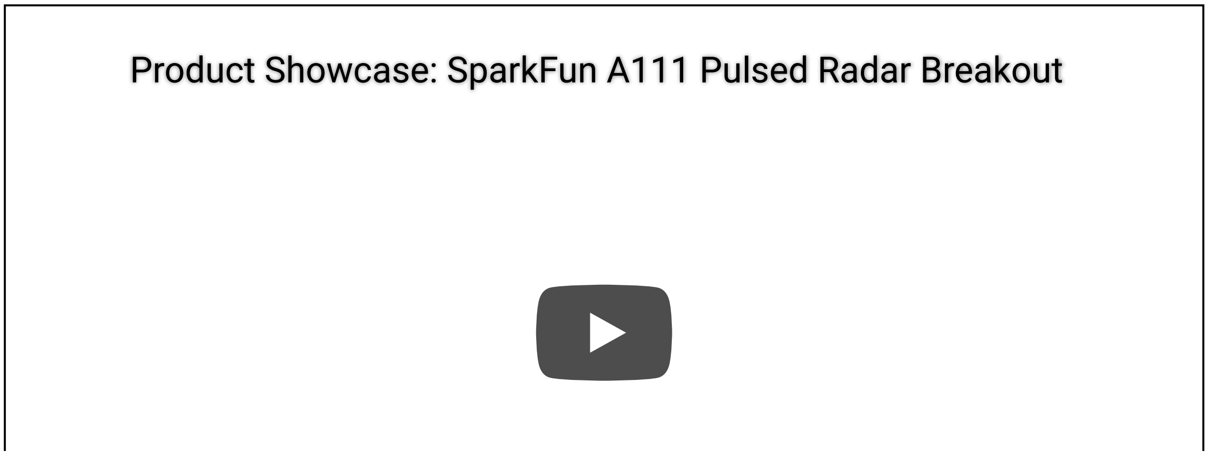


SparkFun Pulsed Radar Breakout - A111

⊘ SEN-15577

★★★⯪☆ 4 **Retired**

Product Showcase: SparkFun Pulsed Radar Breakout



Product Showcase: SparkFun A111 Pulsed Radar Breakout

The A111 is a single-chip solution for pulsed coherent radar (PCR) -- it comes complete with antennae and an SPI interface capable of speeds of up to 50MHz. Applications for PCR include distance-sensing, gesture, motion, and speed detection. The sensor can monitor one-or-more objects at distances of up to two meters.

Our breakout board for the A111 includes a 1.8V regulator, voltage-level translation, and it breaks out all pins of the pulsed radar sensor to both 0.1-inch and Raspberry Pi-friendly headers.

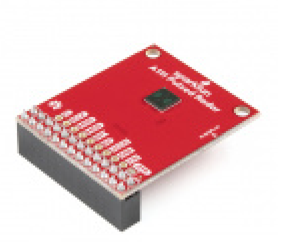
Required Materials

Heads up! The A111 was designed to work with an ARMv7 to work at a minimum. We were able to get the sensor working on the Raspberry Pi 3, 3B+, and 4.

To use the A111 you'll need either an ARMv7 or an ARM Cortex-M4 -- the closed-source SDK currently only supports these architectures. This tutorial will explain how to use the radar sensor with a **Raspberry Pi** -- a platform based on an architecture supported by the A111's SDK.

The A111 Breakout **includes a 20-pin, 2x10 female header**, which should mate to Raspberry Pi's of any generation. If you'd rather manually wire the A111 to your Raspberry Pi and about 9x male-to-female wires should do the trick.

Raspberry Pi 3 B+ Starter Kit
⊖ KIT-15361



SparkFun Pulsed Radar Breakout - A111
⊖ SEN-16826

Jumper Wires Premium 12" M/F Pack of 10
◉ PRT-09385

Optional Materials

You have several options when it comes to working with the Raspberry Pi. Most commonly, the Pi is used as a standalone computer, which requires a monitor, keyboard, and mouse (listed below). To save on costs, the Pi can also be used as a *headless* computer (without a monitor, keyboard, and mouse). This setup has a slightly more difficult learning curve, as you will need to use the *command-line interface* (CLI) from another computer. However, you can also connect to a headless Pi using a remote desktop connection with VNC as well from another computer after configuring the settings.

Raspberry Pi LCD - 7" Touchscreen
○ LCD-13733

Logitech K400 Plus Wireless Touch Keyboard
⊖ WIG-16300

Suggested Reading

If you have not set up a Raspberry Pi before, we recommend looking at the following tutorials to get started. Overall, the setup is the same regardless of the version that you have.

Raspberry Pi 4 Kit Hookup Guide

MARCH 14, 2020

Guide for hooking up your Raspberry Pi 4 Model B basic, desktop, or hardware starter kit together.

Raspberry Pi 3 Starter Kit Hookup Guide

APRIL 11, 2016

Guide for getting going with the Raspberry Pi 3 Model B and Raspberry Pi 3 Model B+ starter kit.

If you aren’t familiar with the following concepts, we recommend checking out these tutorials before continuing. Using the Pi as a headless setup or with VNC can be useful when developing applications with the Pi.

Raspberry Pi SPI and I2C Tutorial

Learn how to use serial I2C and SPI buses on your Raspberry Pi using the wiringPi I/O library for C/C++ and spidev/smbus for Python.

Headless Raspberry Pi Setup

Configure a Raspberry Pi without a keyboard, mouse, or monitor.

How to Use Remote Desktop on the Raspberry Pi with VNC

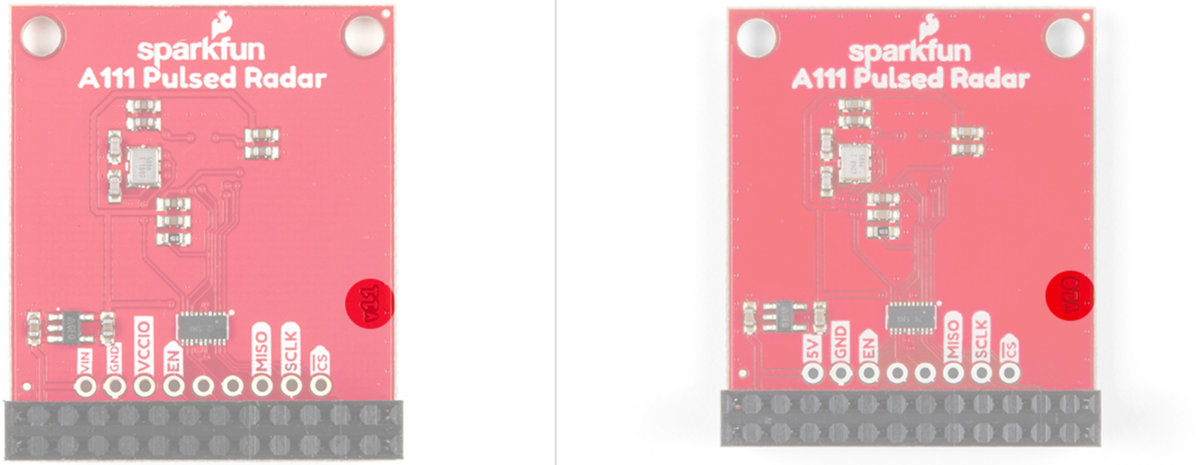
Use RealVNC to connect to your Raspberry Pi to control the graphical desktop remotely across the network.

Hardware Overview

Revision Changes: Overall, the functionality between v1.0 for v1.1 is the same. Minor changes in v1.1 include:

- PTH for VCCIO
- Bypass jumper to set VCCIO to VIN
- Silkscreen name change for input voltage (5V => VIN)
- Improved logic level translation on the Raspberry Pi side

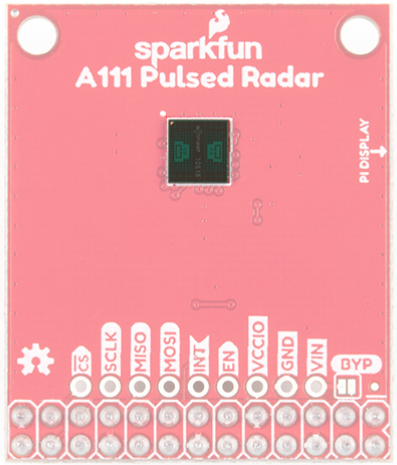
This section covers the hardware overview of both versions of the board. To see what version of the board that you have, you can check the back of the board near the CS pin. You'll also notice the additional bypass jumper on v1.1 that was not included on v1.0.

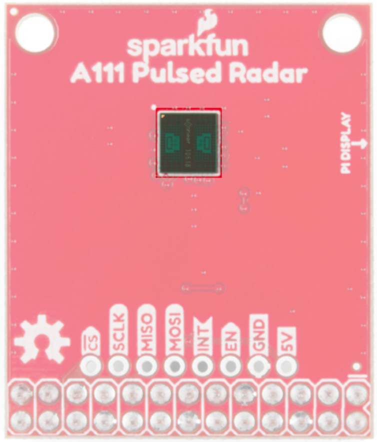


Version number on the back of the board for v1.1 and v1.0

Sensor and Crystal

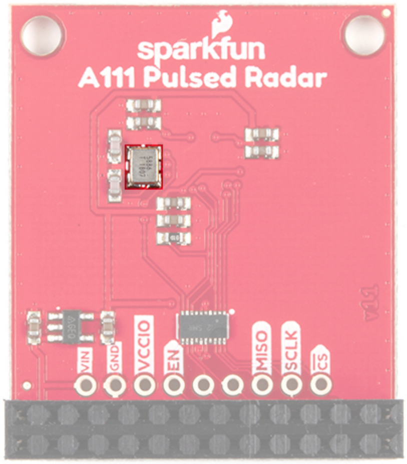
The breakout board is populated with the A111 IC and an external 26MHz crystal oscillator. The IC comes with two antennae and uses mmWave radio to sense objects via radar pulses.

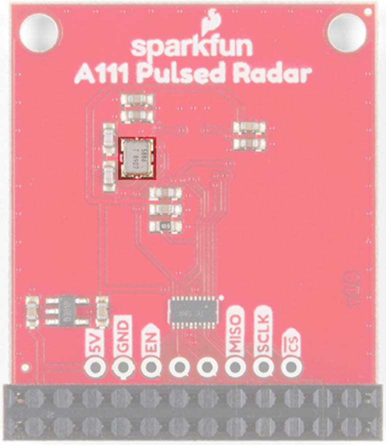




A111 Populated on Top v1.1 and v1.0

You'll find the external crystal oscillator populated on the back of the board. The location is the same on both versions of the board.

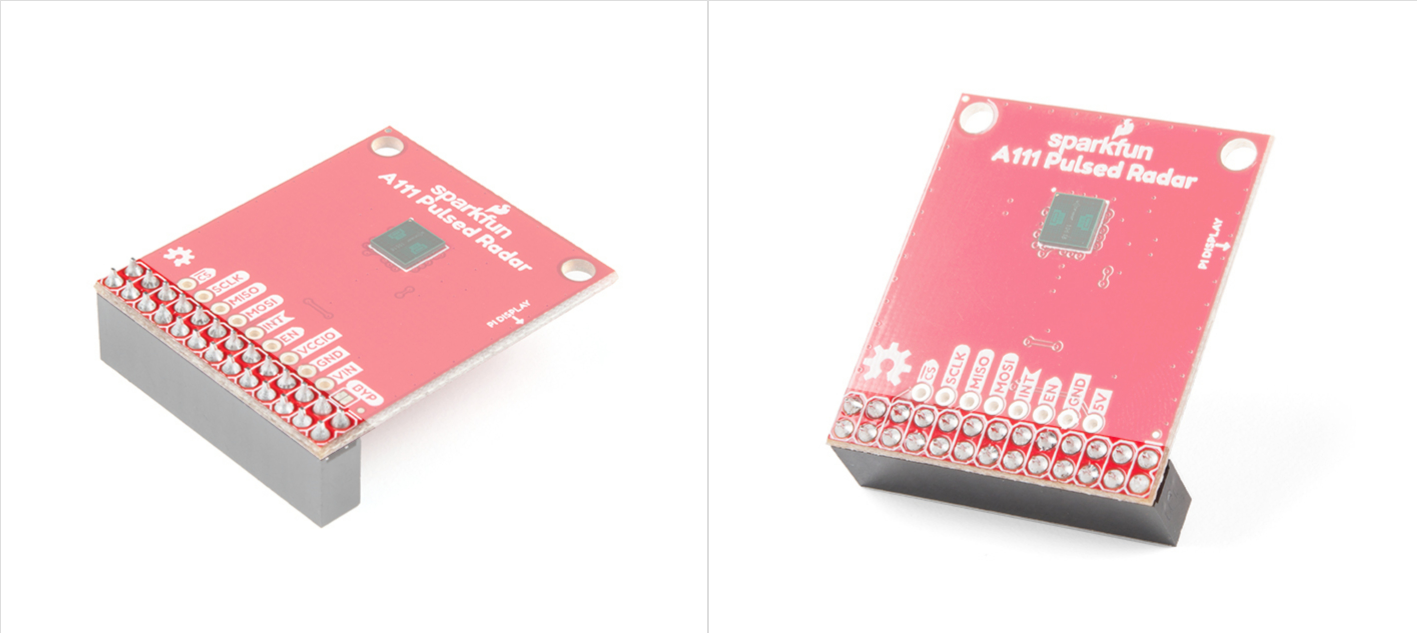




Crystal Oscillator Populated on Back v1.1 and v1.0

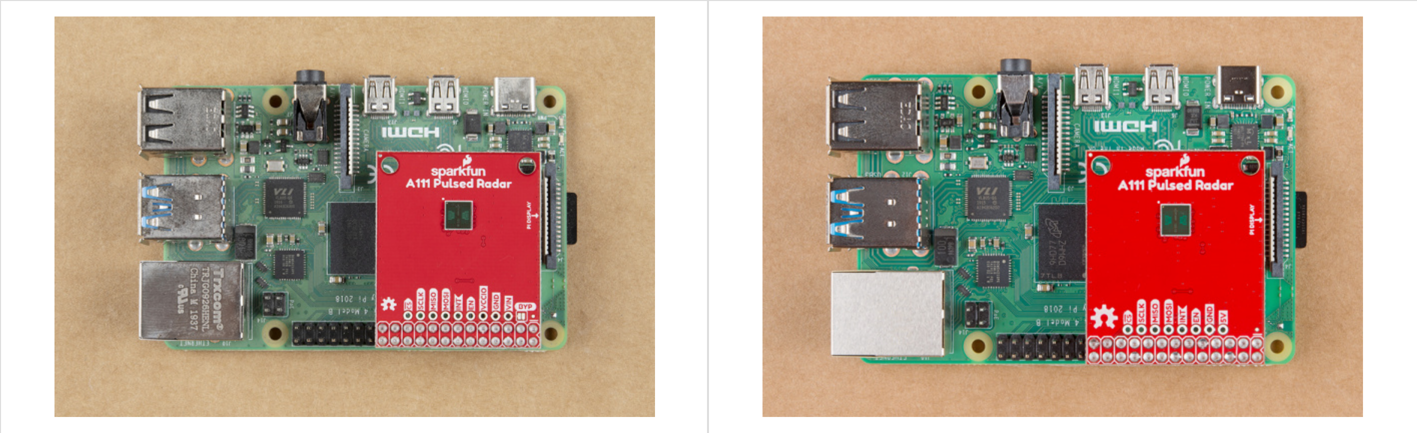
2x13 Female Header

To keep the size of the board small, there is a 2x13 female header soldered on the board to easily mate with a Raspberry Pi's standard header.



HAT Standard Header with Female Connector v1.1 and v1.0

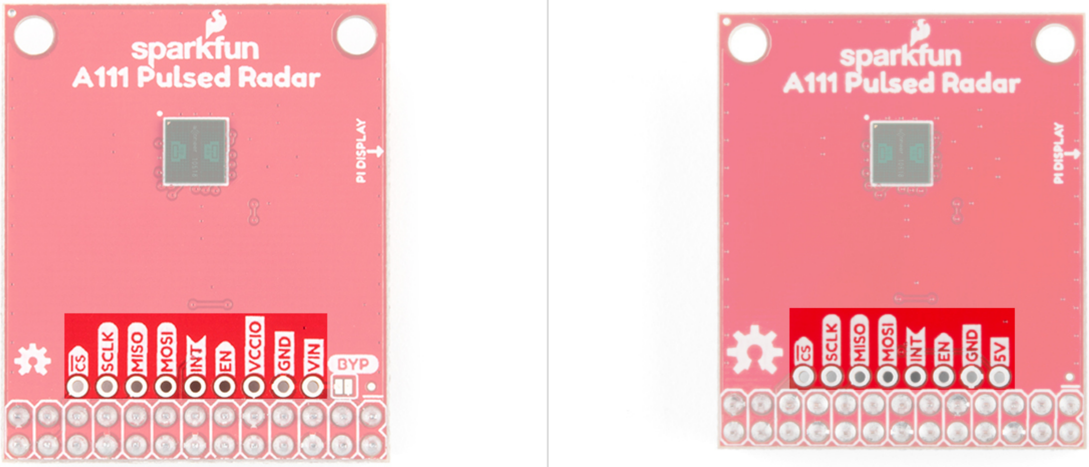
This should only cover the part of the 2x20 male headers on the Pi. Regardless of the version number, the HAT will be stacked near the display port's ribbon cable connector.



A111 HAT v1.1 and v1.0 Stacked on a Raspberry Pi

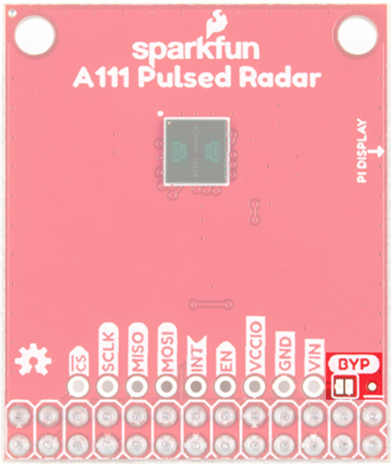
Additional Pins and Jumper Broken Out

Just above the 26-Pin header, there are additional pins broken out for SPI, interrupt, enable, and power if you need access to the pins. Note that in v1.1, the input voltage is labeled as `VIN` while 1.0 labels the pin as `5V`. Both pins are connected to 5V of the Raspberry Pi's HAT standard pinout. For v1.1, the additional PTH for VCCIO is included between the enable and GND pin. For v1.1, additional pins are shifted and align with the 2x13 header.



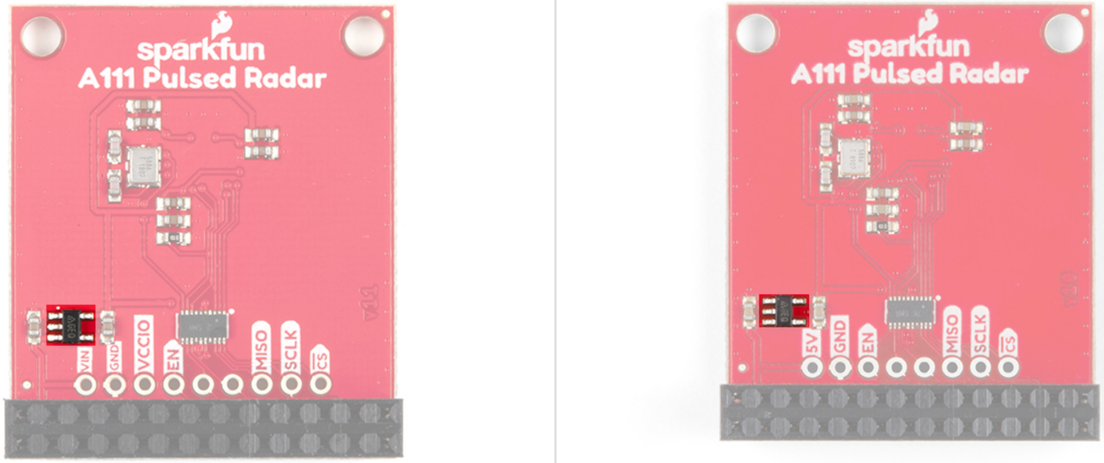
Pins broken out on v1.1 and v1.0

For v1.1, you'll also notice a bypass jumper (labeled as `BYP`). Adding a solder jumper to the board will set `VCCIO` to `VIN` .



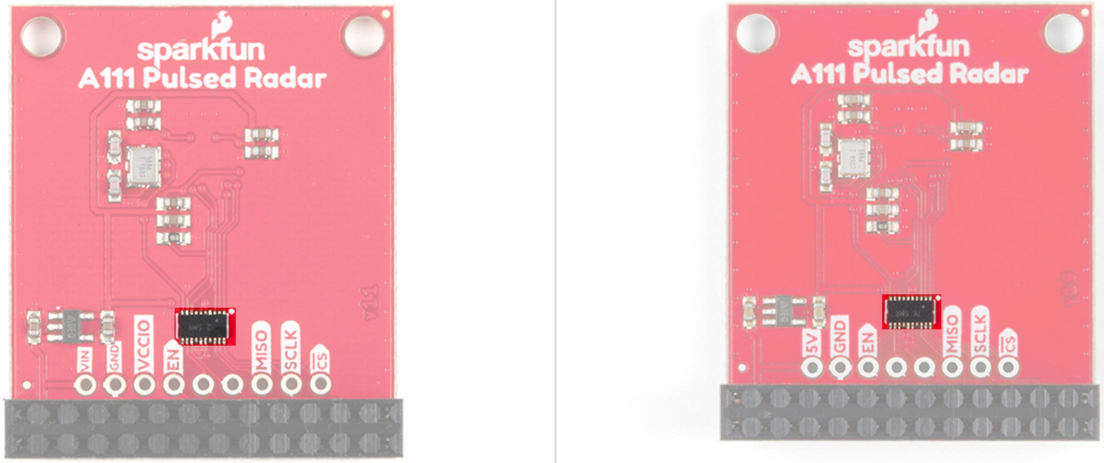
Note: Adding a solder jumper to the v1.1's `BYP` pin will set VCCIO to VIN. If you are stacking the HAT on a Raspberry Pi, this will set the logic level on the Raspberry Pi's side to 5V.We recommend leaving this jumper alone when stacking the board on a Pi.

The A111 IC requires an input voltage of **1.8V**, so a voltage regulator is populated on the back.



Voltage Regulator on v1.1 and v1.0

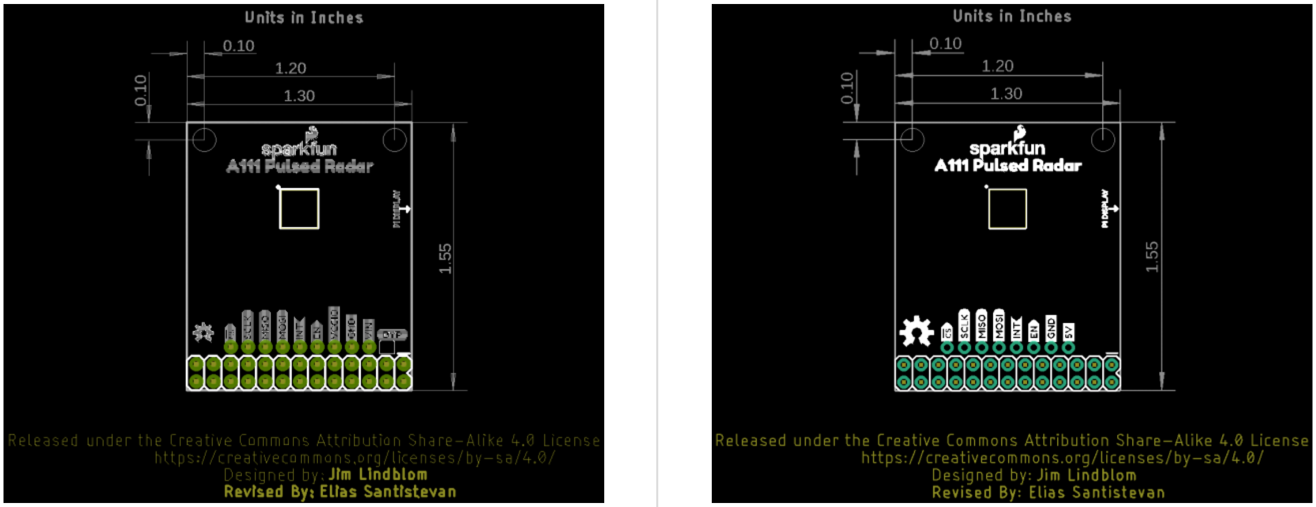
Additionally, the I/O logic level on the A111 is **1.8V**, so a logic converter is included to translate the signals between the Raspberry Pi.



Logic Level Converter on v1.1 and v1.0

Board Dimensions

The overall board size is 1.30in x 1.55in. There are two additional mounting holes spaced 0.10 inches away from the edge of the board.

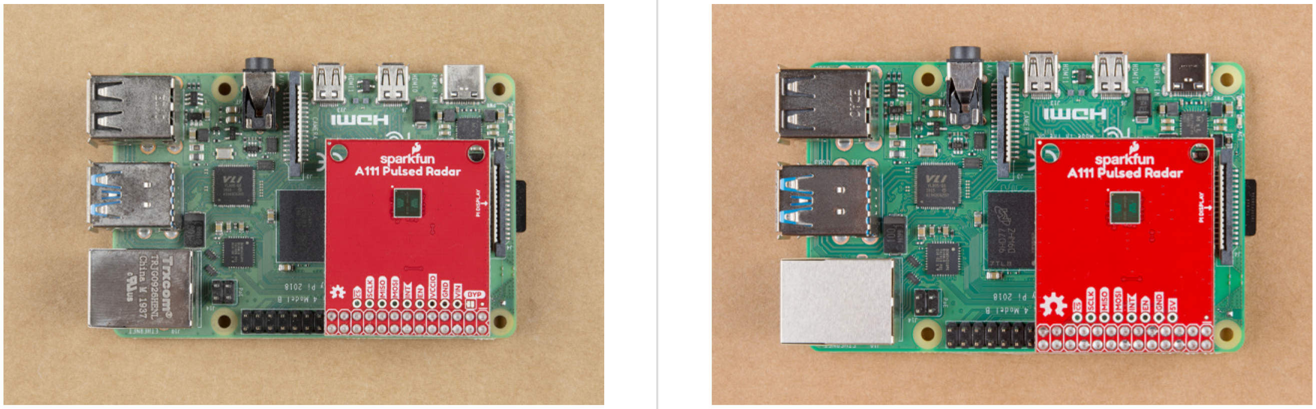

Units in Inches
0.10
1.20
1.30
0.10
1.55
sparkfun
A111 Pulsed Radar
PI DISPLAY
CS
SCLK
MISO
MOSI
INT
EN
VCCIO
GND
VIN
GND
5V
Released under the Creative Commons Attribution Share-Alike 4.0 License
https://creativecommons.org/licenses/by-sa/4.0/
Designed by: Jim Lindblom
Revised By: Elias Santistevan

Board Dimensions for v1.1 and v1.0

Hardware Assembly

The A111 Pulsed Radar Breakout is designed to sit directly on top of a Raspberry Pi. Again, it doesn't span all 40 (2x20) pins of a Raspberry Pi B+ (or later), but the 26-pin — 2x13 — header should be compatible with any Pi.

Connect the shield to a Raspberry Pi ensuring that the "**Pi Display**" text on the breakout matches up with the display header on your Pi. The sensor should be facing up after plugging it in.



A111 HAT v1.1 and v1.0 Stacked on a Raspberry Pi

Or, if you'd like to manually wire the breakout up to a Raspberry Pi's standard pinout, here is the pin-out we'll use through the rest of this tutorial. As stated in the hardware overview for v1.1, the input voltage is labeled as `VIN` while 1.0 labels the pin as `5V`. Both pins are connected to 5V of the Raspberry Pi's HAT standard pinout. For v1.1, the additional PTH for VCCIO is included between the enable and GND pin.

v1.1 Breakout Pin	v1.0 Breakout Pin	Raspberry Pi Pin Name	RasPi Hardware Pin Number
CS	CS	SPI0 CS0	24
SCLK	SCLK	SPI0 SCLK	23
MISO	MISO	SPI0 MISO	21

MOSI	MOSI	SPI0 MOSI	19
INT	INT	GPIO25	22
EN	EN	GPIO27	13
VCCIO		**3.3V**	1, 17
GND	GND	GND	6, 14, 20, etc.
VIN	5V	**5V**	2, 4

Input Voltage and Logic Levels: "5V" should power the sensor, which can consume ~80mA.

When connecting to the additional pins broken out next to the Raspberry Pi's I/O, make sure to not accidentally connect anything higher than 3.3V since it is only 3.3V tolerant.

Configure Your Pi

Raspbian and SPI

This tutorial assumes you've already set up a Raspberry Pi with Raspbian. For help installing the Debian-based OS on your Pi, check out the docs on Raspberrypi.org. Or -- better yet! -- check out our Headless Raspberry Pi Setup tutorial.

You'll also need to **enable SPI** on your Pi. For help with that, check out our SPI on Pi tutorial.

We are going to assume you already have a Raspberry Pi up and running with Raspbian. We'll also assume that it is connected to the Internet. If not, check out our starter kits and tutorials on setting up a Raspberry Pi. We recommend looking at the following tutorials to get started. Overall, the setup is the same regardless of the version that you have.

Raspberry Pi 4 Kit Hookup Guide

MARCH 14, 2020

Guide for hooking up your Raspberry Pi 4 Model B basic, desktop, or hardware starter kit together.

Raspberry Pi 3 Starter Kit Hookup Guide

APRIL 11, 2016

Guide for getting going with the Raspberry Pi 3 Model B and Raspberry Pi 3 Model B+ starter kit.

Make sure to update the image so that we have the latest distribution. Enter the following commands in the command line individually to update your image.

```
sudo apt-get update
sudo apt-get dist-upgrade
```

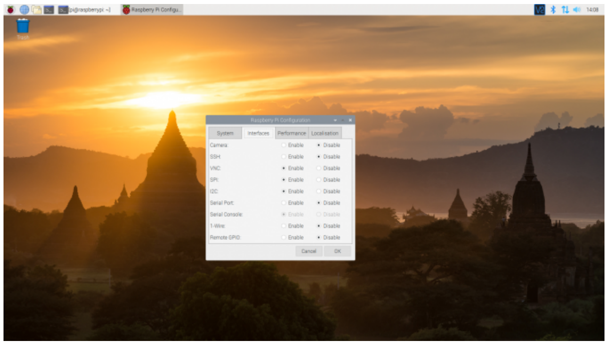
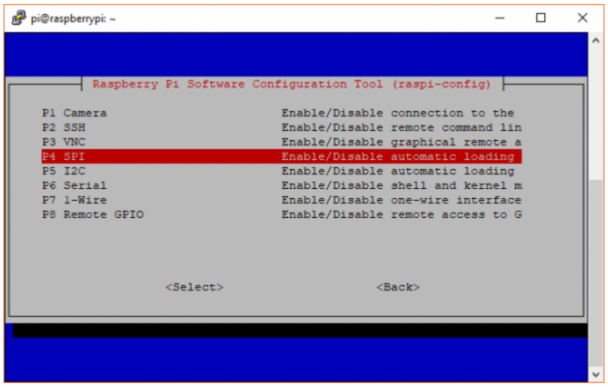
> **Note:** sudo stands for "Super User Do", it is a way to give you superuser powers from the command line. Be careful whenever using `sudo` .

User Configuration Settings

Once you are set up, I highly recommend changing your default login and password: (**username**: pi, **password**: raspberry). The Raspberry Pi Configuration tool is a quick way to change your password as well as setup the network, language, keyboard, etc. There are two methods to adjust the settings. You can use the GUI by heading to the **Pi Start Menu** > **Preferences** > **Raspberry Pi Configuration** > **Interfaces**. For the old school heads, you can also use the **raspi-config** tool by typing the following command using the command line and then go through the menus to update your information.

```
sudo raspi-config
```

You'll want to enable the SPI pins using the tool to read sensor with SPI. If you are using hte GUI, simple select **Enable** and click on the OK button. If you are using the **raspi-config**, follow the prompts to finish setting up the SPI. For more information, check out our tutorial to set up the SPI pins on a Pi.

	
Enabling SPI via Desktop GUI	*raspi-config tool for SPI in the Terminal*

Raspberry Pi Configuration

System | Interfaces | Performance | Localisation

Camera:	○ Enable	● Disable
SSH:	○ Enable	● Disable
VNC:	● Enable	○ Disable
SPI:	● Enable	○ Disable
I2C:	● Enable	○ Disable
Serial Port:	○ Enable	● Disable
Serial Console:	● Enable	○ Disable
1-Wire:	○ Enable	● Disable
Remote GPIO:	○ Enable	● Disable

Cancel OK

```
pi@raspberrypi: ~

Raspberry Pi Software Configuration Tool (raspi-config)

P1 Camera                        Enable/Disable connection to the
P2 SSH                           Enable/Disable remote command lin
P3 VNC                           Enable/Disable graphical remote a
P4 SPI                           Enable/Disable automatic loading
P5 I2C                           Enable/Disable automatic loading
P6 Serial                        Enable/Disable shell and kernel m
P7 1-Wire                        Enable/Disable one-wire interface
P8 Remote GPIO                   Enable/Disable remote access to G


                 <Select>                     <Back>
```

You'll need to restart your Pi for the changes to take effect.

Get the SDK

> **Heads up!** The following instructions are listed for SDK v2.1. Acconeer provides frequent software updates to the SDK packages to fix bugs and provide the latest features. The steps may not apply for higher SDK versions. You may need to adjust the path names depending on the changes. for higher versions.

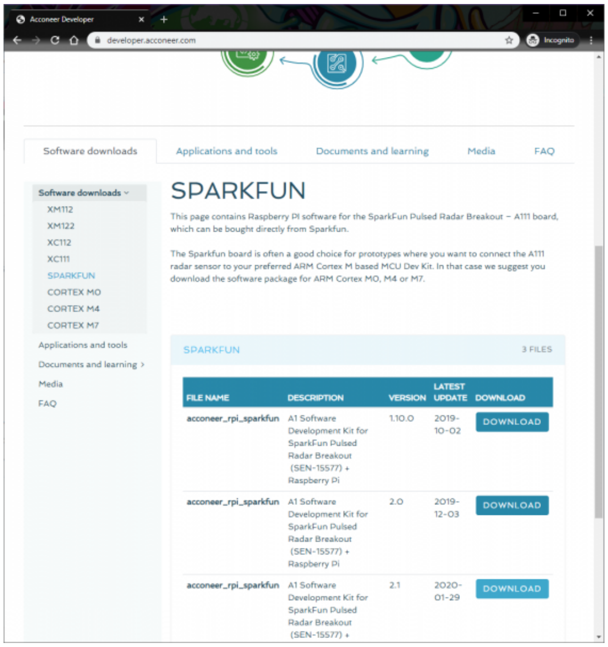
The software development kit (SDK) for the A111 is, unfortunately, locked behind a closed source blob that currently only supports Cortex-M4 and ARMv7 platforms. However, the folks over at Acconeer have agreed to host the latest version of their SDK with our example code from their website. To download the latest SDK from Acconeer, visit Acconeer's "Developer" page on your Raspberry Pi's browser.

VISIT ACCONEER'S "DEVELOPER" FOR THE SDK

You'll need an account in order to have access to the files. Click on the link at the bottom of the page to create one or log in. Or click on the shortcuts below.

SIGN UP WITH ACCONEER **LOG IN WITH ACCONEER**

Once you are logged in, the "**Software Download**" tab will include additional links to download the SDK. Scroll down the webpage to the section for the **SPARKFUN**. Acconeer is constantly updating and improving the software. We used **v2.1** of the software when writing this tutorial so the version may be different from the version that you download. Click on the **Download** button for the latest version. In this case, it was v2.1.



SPARKFUN

This page contains Raspberry PI software for the SparkFun Pulsed Radar Breakout – A111 board, which can be bought directly from Sparkfun.

The Sparkfun board is often a good choice for prototypes where you want to connect the A111 radar sensor to your preferred ARM Cortex M based MCU Dev Kit. In that case we suggest you download the software package for ARM Cortex M0, M4 or M7.

SPARKFUN 3 FILES

FILE NAME	DESCRIPTION	VERSION	LATEST UPDATE	DOWNLOAD
acconeer_rpi_sparkfun	A1 Software Development Kit for SparkFun Pulsed Radar Breakout (SEN-15577) + Raspberry Pi	1.10.0	2019-10-02	DOWNLOAD
acconeer_rpi_sparkfun	A1 Software Development Kit for SparkFun Pulsed Radar Breakout (SEN-15577) + Raspberry Pi	2.0	2019-12-03	DOWNLOAD
acconeer_rpi_sparkfun	A1 Software Development Kit for SparkFun Pulsed Radar Breakout (SEN-15577) +	2.1	2020-01-29	DOWNLOAD

Download the SDK from Acconeer.

Read through the license, agree, and click on the **Download acconeer_rpi_sparkfun** button for the software. Depending on your browser, you may need to refresh the page and repeat the directions before the button is active.

Once downloaded, head to directory where you downloaded the ZIP. You can use the GUI or you can use the terminal to unzip the SDK using the following commands (included are commands to install unzip). Just make sure to update the file name with the version that you downloaded. Assuming that you are in the **Downloads** folder, you will be using the following command in the Terminal.

```
unzip acconeer_rpi_sparkfun_v2_1_0.zip
```

Note: Make sure you replace the acconeer ZIP file name with that of your downloaded SDK version.

Then `cd` to the "**rpi_sparkfun**" directory to prepare to build the example software. Depending on the version that you downloaded, the path might be slightly different. Make sure to use the `ls` to verify the contents of the unzipped **a111** folder if you have issues navigating through the directory.

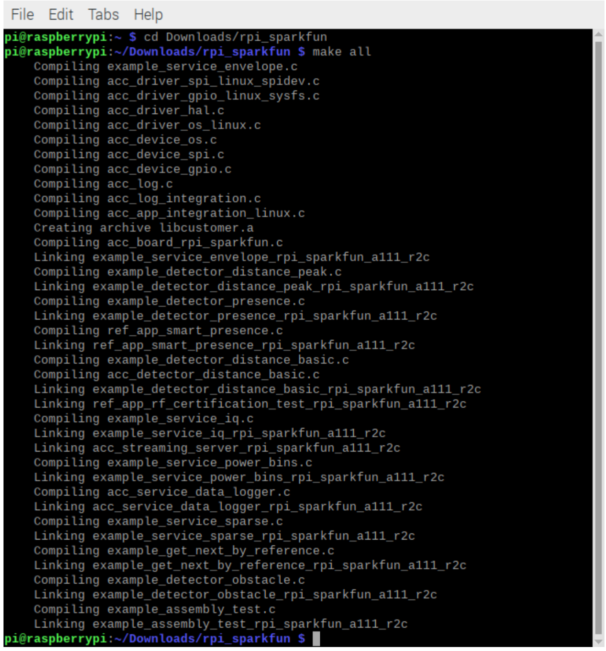
SDK Overview

The A111 SDK includes source code, archived libraries, include files, and documentation for using the A111 pulsed radar sensor. Here's a quick overview of what's included with the SDK:

- **doc** — Doxygen-generated documentation for the A111 API and source code.
- **include** — Header and API files which describe how to interact with the pre-compiled A111 libraries.
- **lib** — Pre-compiled A111 static archives. API for these files are provided in the "include" directory.
- **out** — Compiled board and example object and executable files.
- **rule** — Recursive Makefile rules for board and example files.
- **source** — C source files for custom boards and example applications.
- **makefile** — Top-level makefile. Recursively calls files in the "rule" directory to build example and board files.

Build and Run the Test Sketch

In the terminal, executing the make file — and it's recursive dependencies — should build all of the examples you may use with the **A111**. To build all board and example files, navigate to the SDK's top-level directory (in this case, it should be similar to **...Downloads/rpi_sparkfun**) where your **makefile** is located and type `make` or `make all` command in the terminal. You should see something similar to the output below.



```
pi@raspberrypi:~ $ cd Downloads/rpi_sparkfun
pi@raspberrypi:~/Downloads/rpi_sparkfun $ make all
    Compiling example_service_envelope.c
    Compiling acc_driver_spi_linux_spidev.c
    Compiling acc_driver_gpio_linux_sysfs.c
    Compiling acc_driver_hal.c
    Compiling acc_driver_os_linux.c
    Compiling acc_device_os.c
    Compiling acc_device_spi.c
    Compiling acc_device_gpio.c
    Compiling acc_log.c
    Compiling acc_log_integration.c
    Compiling acc_app_integration_linux.c
    Creating archive libcustomer.a
    Compiling acc_board_rpi_sparkfun.c
    Linking example_service_envelope_rpi_sparkfun_a111_r2c
    Compiling example_detector_distance_peak.c
    Linking example_detector_distance_peak_rpi_sparkfun_a111_r2c
    Compiling example_detector_presence.c
    Linking example_detector_presence_rpi_sparkfun_a111_r2c
    Compiling ref_app_smart_presence.c
    Linking ref_app_smart_presence_rpi_sparkfun_a111_r2c
    Compiling example_detector_distance_basic.c
    Compiling acc_detector_distance_basic.c
    Linking example_detector_distance_basic_rpi_sparkfun_a111_r2c
    Linking ref_app_rf_certification_test_rpi_sparkfun_a111_r2c
    Compiling example_service_iq.c
    Linking example_service_iq_rpi_sparkfun_a111_r2c
    Linking acc_streaming_server_rpi_sparkfun_a111_r2c
    Compiling example_service_power_bins.c
    Linking example_service_power_bins_rpi_sparkfun_a111_r2c
    Compiling acc_service_data_logger.c
    Linking acc_service_data_logger_rpi_sparkfun_a111_r2c
    Compiling example_service_sparse.c
    Linking example_service_sparse_rpi_sparkfun_a111_r2c
    Compiling example_get_next_by_reference.c
    Linking example_get_next_by_reference_rpi_sparkfun_a111_r2c
    Compiling example_detector_obstacle.c
    Linking example_detector_obstacle_rpi_sparkfun_a111_r2c
    Compiling example_assembly_test.c
    Linking example_assembly_test_rpi_sparkfun_a111_r2c
pi@raspberrypi:~/Downloads/rpi_sparkfun $ 
```

Troubleshooting

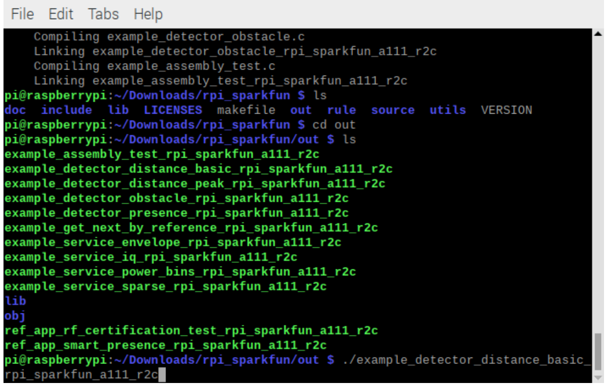
If you have any trouble building the board and example files, ensure that you have gcc and make packages installed. (E.g. `apt-get install make gcc`)

Running the Example Applications

Once compiled you can run the example applications. Navigate to the **.../out** folder with the `cd` in the terminal window. You can run the example by clicking on the executable or typing the following command. We recommend typing the command in the terminal to view the data. A few of the applications will automatically close the window after taking sensor data if you choose to click on the executable.

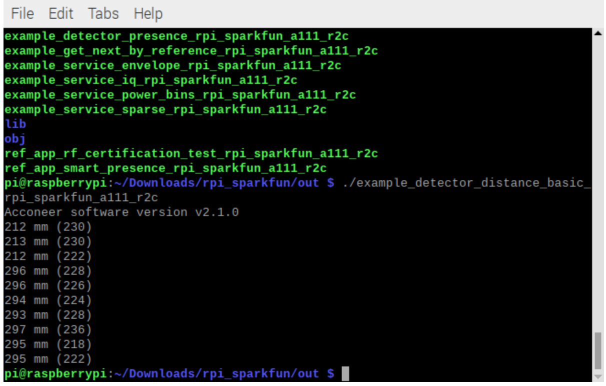
```
./example_detector_distance_basic_rpi_sparkfun_a111_r2c
```

Once you type (or paste the command in the Terminal), hit `Enter` on your keyboard to execute the command.



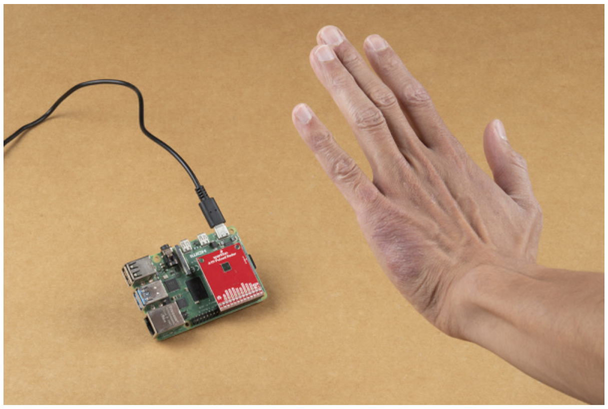
```
    Compiling example_detector_obstacle.c
    Linking example_detector_obstacle_rpi_sparkfun_a111_r2c
    Compiling example_assembly_test.c
    Linking example_assembly_test_rpi_sparkfun_a111_r2c
pi@raspberrypi:~/Downloads/rpi_sparkfun $ ls
doc  include  lib  LICENSES  makefile  out  rule  source  utils  VERSION
pi@raspberrypi:~/Downloads/rpi_sparkfun $ cd out
pi@raspberrypi:~/Downloads/rpi_sparkfun/out $ ls
example_assembly_test_rpi_sparkfun_a111_r2c
example_detector_distance_basic_rpi_sparkfun_a111_r2c
example_detector_distance_peak_rpi_sparkfun_a111_r2c
example_detector_obstacle_rpi_sparkfun_a111_r2c
example_detector_presence_rpi_sparkfun_a111_r2c
example_get_next_by_reference_rpi_sparkfun_a111_r2c
example_service_envelope_rpi_sparkfun_a111_r2c
example_service_iq_rpi_sparkfun_a111_r2c
example_service_power_bins_rpi_sparkfun_a111_r2c
example_service_sparse_rpi_sparkfun_a111_r2c
lib
obj
ref_app_rf_certification_test_rpi_sparkfun_a111_r2c
ref_app_smart_presence_rpi_sparkfun_a111_r2c
pi@raspberrypi:~/Downloads/rpi_sparkfun/out $ ./example_detector_distance_basic_
rpi_sparkfun_a111_r2c
```

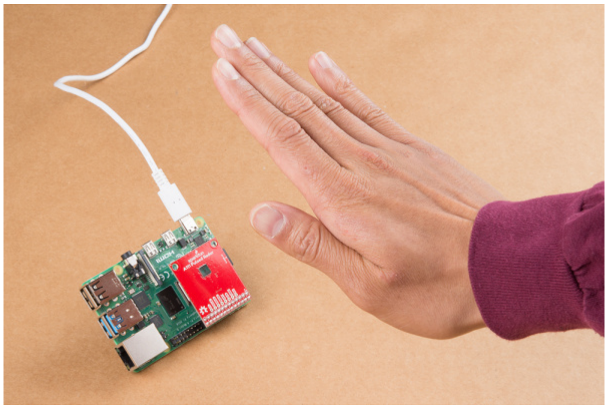
This will run our modified distance-detector example application. For simplicity, it will show the distance of an object that is in front of the antennae.



```
example_detector_presence_rpi_sparkfun_a111_r2c
example_get_next_by_reference_rpi_sparkfun_a111_r2c
example_service_envelope_rpi_sparkfun_a111_r2c
example_service_iq_rpi_sparkfun_a111_r2c
example_service_power_bins_rpi_sparkfun_a111_r2c
example_service_sparse_rpi_sparkfun_a111_r2c
lib
obj
ref_app_rf_certification_test_rpi_sparkfun_a111_r2c
ref_app_smart_presence_rpi_sparkfun_a111_r2c
pi@raspberrypi:~/Downloads/rpi_sparkfun/out $ ./example_detector_distance_basic_
rpi_sparkfun_a111_r2c
Acconeer software version v2.1.0
212 mm (230)
213 mm (230)
212 mm (222)
296 mm (228)
296 mm (226)
294 mm (224)
293 mm (228)
297 mm (236)
295 mm (218)
295 mm (222)
pi@raspberrypi:~/Downloads/rpi_sparkfun/out $
```

By placing an object in front of the sensor, the distance should vary based on what the A111 can detect. The application will continue running in the terminal window until the loop is finished. In this case, the example took 10 samples. Try adjusting the code to obtain more samples!





Hand placed in front the A111 v1.1 and v1.0 with the Pi remotely connected via VNC.

Running Other Examples

In addition to the distance detector basic, the SDK includes a few extra examples. To execute these files, run them from the **out** directory:

- **example_detector_presence_rpi_sparkfun_a111_r2c** — If there is an object moving in front of the sensor, it will output a true. Otherwise, the output is false.
- **example_detector_obstacle_rpi_sparkfun_a111_r2c** — Checks to see if there is an object in the way.
- **example_service_iq_rpi_sparkfun_a111_r2c** — Advanced version of the envelope service, which includes phase information for very small variations in distance. The output will be in polar coordinates.
- **example_service_power_bins_rpi_sparkfun_a111_r2c** — Demonstrates how to use A111 "power bins," which are still, kind of, a mystery to us...
- **example_detector_distance_peak_rpi_sparkfun_a111_r2c** — This application will begin by calculating raw peak-distances, with a maximum of ten reflections. The example will briefly estimate minimum and maximum threshold.

Make sure to read through the comments in the code (located in the **source** folder), datasheet, user guide, and associated documents from Acconeer for more information before writing your own custom code for the A111.

Troubleshooting

For the best support, we recommend checking the FAQ or submitting a support ticket with Acconeer since they handle the software SDK packages.

ACCONEEER > FAQ AND SDK SUPPORT

Resources and Going Further

Now that you've successfully got your A111 pulsed radar sensor up and running, it's time to incorporate it into your own project! For more information, check out the resources below:

- v1.1
  - Schematic (PDF)
  - Eagle Files (ZIP)
  - Board Dimensions (PNG)
  - SFE Showcase
- v1.0
  - Schematic (PDF)
  - Eagle Files (ZIP)
  - Board Dimensions (PNG)
  - SFE Showcase
- Acconeer
  - Developer - SW Download, datasheet, user guide, FAQ
  - Read the Docs: Setting Up Your Raspberry PI EVK - Documentation for the Python examples that might be useful in visualizing the data taken with the A111 examples.
- GitHub Repos
  - Acconeer Example Code
  - *SparkFun Example Code (Archived)*
  - Hardware Repo

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

Getting Started with Walabot

See through walls, track objects, monitor breathing patterns, and more using the power of radio frequency with the Walabot! In this tutorial, we will explore Walabot’s features using the Software Demo Kit (SDK) on Windows and the Application Programming Interface (API) on Linux-based distributions for embedded projects.

Qwiic Kit for Raspberry Pi Hookup Guide

Get started with the CCS811, BME280, VCNL4040, and microOLED via I2C using the Qwiic system and Python on a Raspberry Pi! Take sensor readings from the environment and display them on the microOLED, serial terminal, or the cloud with Cayenne!