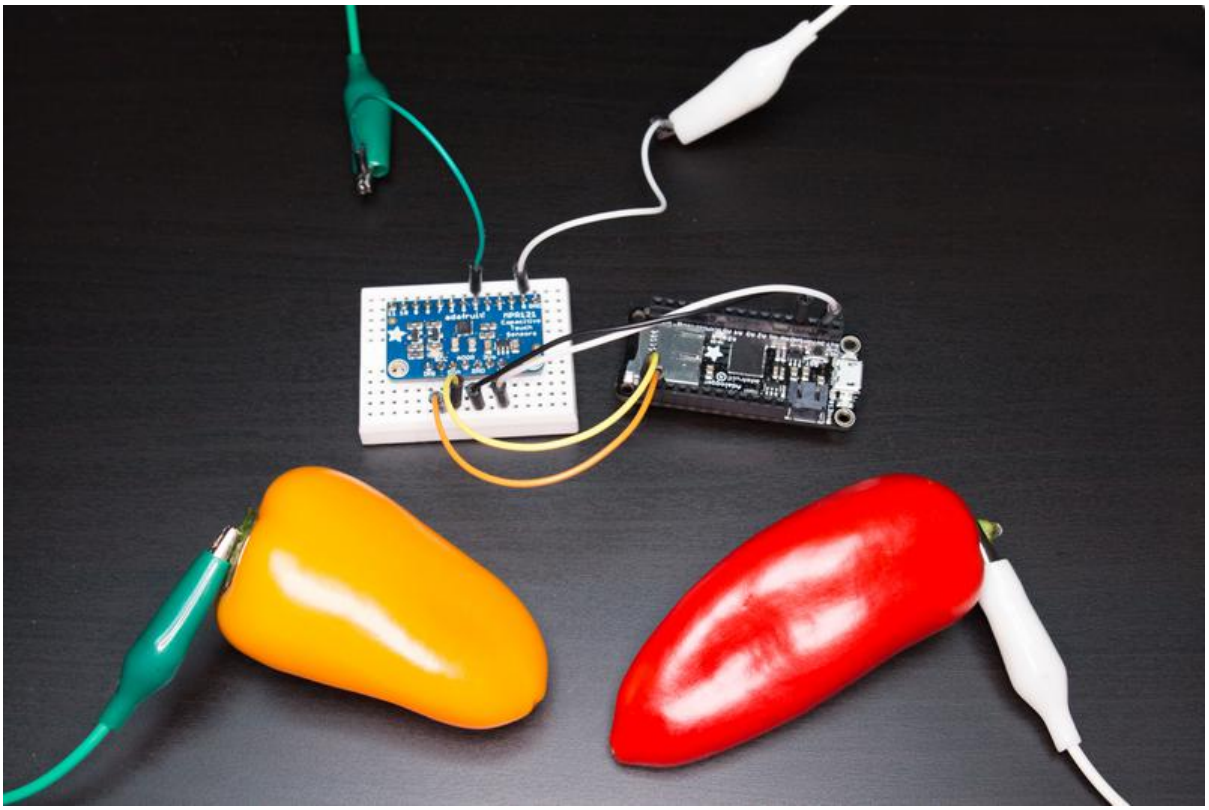
CircuitPython Hardware: MPR121 Capacitive Touch Breakout

Created by Tony DiCola



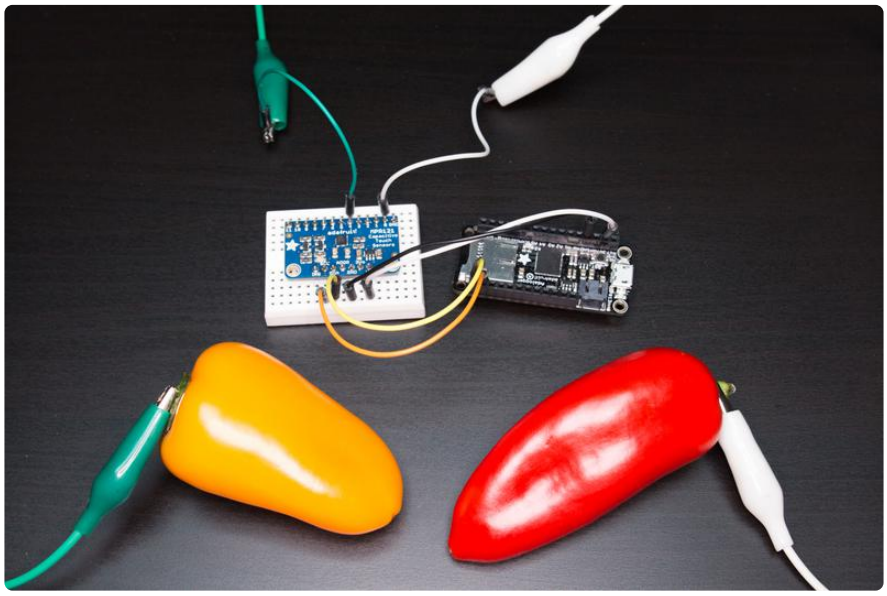
https://learn.adafruit.com/circuitpython-hardware-mpr121-capacitive-touch-breakout

Last updated on 2023-08-29 03:20:22 PM EDT

Table of Contents

Overview

The examples in this guide are no longer supported. Check out the MPR121 sensor guide for CircuitPython and Python usage: https://learn.adafruit.com/adafruit-mpr121-12-key-capacitive-touch-sensor-breakout-tutorial/overview



Reach out and touch somebody or some thing with capacitive touch sensing and the MPR121 capacitive touch breakout board (). Capacitive touch sensing means detecting when something conductive is touched by a large object or person. For example detecting when a person touches a piece of foil, conductive ink, or even a piece of fruit or a vegetable. It almost seems like magic but capacitive touch sensing is based on a very simple principle--by touching an object your body slightly changes the capacitance of it and that change can be detected with special circuitry.

The MPR121 board is a dedicated capacitive touch sensing chip that can check up to 12 inputs independently for touches. Since the MPR121 uses a simple I2C interface you can use it with almost any development board like an Arduino, MicroPython or CircuitPython board. This guide shows how to use the MPR121 capacitive touch sensing breakout with a CircuitPython board.

Before you follow this guide it will help to familiarize yourself with the following other guides:

- MicroPython Basics: What is MicroPython? ()
- MicroPython Basics: How to Load MicroPython on a Board ()
- MicroPython Basics: Load Files & Run Code ()

- Adafruit MPR121 Capacitive Touch Sensor Breakout ()

Continue on to learn about the hardware needed to follow this guide.

CircuitPython Wiring

The examples in this guide are no longer supported. Check out the MPR121 sensor guide for CircuitPython and Python usage: https://learn.adafruit.com/adafruit-mpr121-12-key-capacitive-touch-sensor-breakout-tutorial/overview

Parts

You'll need the following hardware to follow this guide:

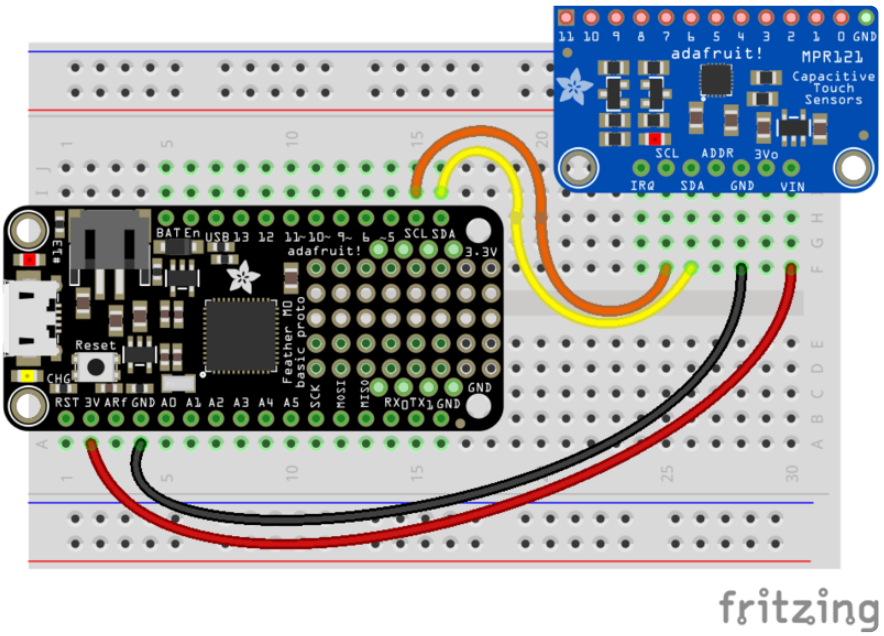
- Board running CircuitPython. Currently ESP8266 or SAMD21-based boards are supported by CircuitPython. The Feather HUZZAH ESP8266 () or Feather M0 boards (http://adafru.it/2772) are great options that can easily be connected to an accelerometer. See the guide on loading MicroPython & CircuitPython firmware () on a board for details on loading CircuitPython.
- MPR121 capacitive touch breakout board () or MPR121 capacitive touch Arduino shield ().
- Breadboard (), hookup wires (http://adafru.it/153), and soldering tools (). You'll need to solder headers to the accelerometer breakout, be sure to see the guide to excellent soldering () if you're new to it.
- Alligator clip wires (http://adafru.it/1008) or alligator clip to male jumper wires (http://adafru.it/3255), these are useful for connecting conductive objects to the MPR121 breakout inputs. For example you can connect the inputs to pieces of fruit to make a funky, fruity musical instrument!

Start by following the MPR121 breakout guide () to assemble and test the board. Then continue on below to learn how to wire it to a Feather for use with CircuitPython.

For the MPR121 Arduino shield just solder headers to the shield (don't forget to solder the 2x3 SPI header in the center of the board too) and connect it to a compatible Arduino (like the Arduino Zero flashed with CircuitPython firmware).

Wiring

Connect the MPR121 to your board using its I2C interface as follows:



- MPR121 VIN to board 3V (or 5V) output - red wire.
- MPR121 GND to board GND/ground - black wire.
- MPR121 SCL to board SCL (I2C clock) - orange wire.
- MPR121 SDA to board SDA (I2C data) - yellow wire.

Continue on to learn how to install a CircuitPython module to control the MPR121 board.

CircuitPython Software

The examples in this guide are no longer supported. Check out the MPR121 sensor guide for CircuitPython and Python usage: https://learn.adafruit.com/adafruit-mpr121-12-key-capacitive-touch-sensor-breakout-tutorial/overview

Adafruit CircuitPython Module Install

To use the MPR121 with your Adafruit CircuitPython () board you'll need to install the Adafruit_CircuitPython_MPR121 () module on your board.

First make sure you are running the latest version of Adafruit CircuitPython () for your board.

Next you'll need to install the necessary libraries to use the hardware--carefully follow the steps to find and install these libraries from Adafruit's CircuitPython library bundle (). For example the Circuit Playground Express guide has a great page on how to install the library bundle () for both express and non-express boards.

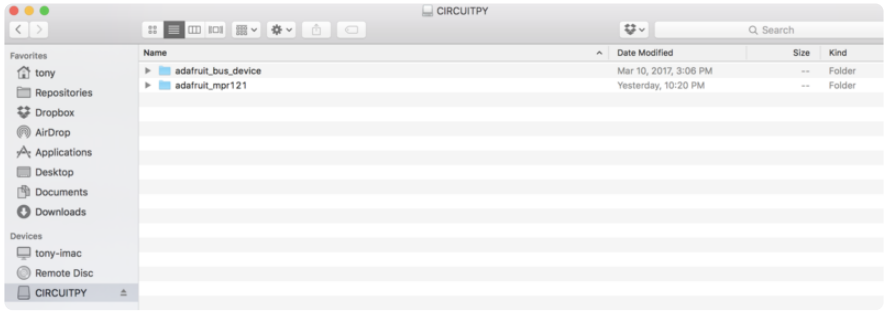
Remember for non-express boards like the Trinket M0, Gemma M0, and Feather/Metro M0 basic you'll need to manually install the necessary libraries from the bundle:

- adafruit_mpr121
- adafruit_bus_device

You can also download the adafruit_mpr121.zip which contains the adafruit_mpr121 folder from its releases page on Github ().

If your board doesn't support USB mass storage, like the ESP8266, then use a tool like ampy to copy the file to the board ().

Before continuing make sure your board's lib folder or root filesystem has the adafruit_mpr121, and adafruit_bus_device folders copied over.



Example

To learn how to use the MPR121 module code you can look at the simpletest.py example () included in the library. Save this as a main.py file on your board:

```
# SPDX-FileCopyrightText: 2017 Tony DiCola for Adafruit Industries
# SPDX-License-Identifier: MIT

# Simple test of the MPR121 capacitive touch sensor library.
# Will print out a message when any of the 12 capacitive touch inputs of the
# board are touched.  Open the serial REPL after running to see the output.
# Author: Tony DiCola
import time
```

```
import board
import busio

# Import MPR121 module.
import adafruit_mpr121

# Create I2C bus.
i2c = busio.I2C(board.SCL, board.SDA)

# Create MPR121 object.
mpr121 = adafruit_mpr121.MPR121(i2c)

# Note you can optionally change the address of the device:
# mpr121 = adafruit_mpr121.MPR121(i2c, address=0x91)

# Loop forever testing each input and printing when they're touched.
while True:
    # Loop through all 12 inputs (0-11).
    for i in range(12):
        # Call is_touched and pass it then number of the input.  If it's touched
        # it will return True, otherwise it will return False.
        if mpr121[i].value:
            print("Input {} touched!".format(i))
    time.sleep(0.25)  # Small delay to keep from spamming output messages.
```

Once the example is running open the serial REPL for your board. With your finger try pressing any of the 0-11 input pins on the MPR121. You should see a message printed every time an input is touched:

If you don't see any messages when you touch the inputs you might need to ground yourself to the board by touching the GND pin on the board with one finger and then touching the input pads with another finger.

Also make sure nothing is touching the pins when you first run the script or else it might confuse the MPR121's touch detection (unmount the board's file system from your operating system, then press the board's reset button to reset the script and run it again with nothing touching the pins).

Usage

Examine the simpletest.py code () to see how to use the MPR121 module.

```
# SPDX-FileCopyrightText: 2017 Tony DiCola for Adafruit Industries
# SPDX-License-Identifier: MIT
```

```
# Simple test of the MPR121 capacitive touch sensor library.
# Will print out a message when any of the 12 capacitive touch inputs of the
# board are touched.  Open the serial REPL after running to see the output.
# Author: Tony DiCola
import time
import board
import busio

# Import MPR121 module.
import adafruit_mpr121

# Create I2C bus.
i2c = busio.I2C(board.SCL, board.SDA)

# Create MPR121 object.
mpr121 = adafruit_mpr121.MPR121(i2c)

# Note you can optionally change the address of the device:
# mpr121 = adafruit_mpr121.MPR121(i2c, address=0x91)

# Loop forever testing each input and printing when they're touched.
while True:
    # Loop through all 12 inputs (0-11).
    for i in range(12):
        # Call is_touched and pass it then number of the input.  If it's touched
        # it will return True, otherwise it will return False.
        if mpr121[i].value:
            print("Input {} touched!".format(i))
    time.sleep(0.25)  # Small delay to keep from spamming output messages.
```

First the module is imported with code like:

```
# Import MPR121 module.
import adafruit_mpr121
```

This code initializes the I2C bus:

```
# Create I2C bus.
import board
import busio
i2c = busio.I2C(board.SCL, board.SDA)
```

Remember for boards without hardware I2C like the ESP8266 you might need to use the bitbangio module to create the I2C bus:

```
# Create I2C bus for software I2C boards (ESP8266)
import board
import bitbangio
i2c = bitbangio.I2C(board.SCL, board.SDA)
```

Once the I2C bus is intialized the MPR121 class can be created by passing it an instance of the I2C bus:

```
# Create MPR121 class.
mpr121 = adafruit_mpr121.MPR121(i2c)
```

```
# Note you can optionally change the address of the device:
#mpr121 = adafruit_mpr121.MPR121(i2c, address=0x91)
```

Notice the MPR121 class initializer takes an optional address parameter if you changed the board's I2C address.

Now you're ready to start checking if inputs are touched. Notice the main loop for the example calls the MPR121 class is_touched function for every input:

```
# Loop forever testing each input and printing when they're touched.
while True:
    # Loop through all 12 inputs (0-11).
    for i in range(12):
        # Call is_touched and pass it then number of the input.  If it's touched
        # it will return True, otherwise it will return False.
        if mpr121.is_touched(i):
            print('Input {} touched!'.format(i))
    time.sleep(0.25)  # Small delay to keep from spamming output messages.
```

Just pass a value 0 to 11 to the is_touched function and it will return a boolean True or False value depending on if the input is currently being touched or not. In the example it prints a small message when an input is touched.

The example doesn't show its usage but if you want to check all of the inputs at once you can call the touched function. This function returns a 12-bit value where each bit represents the touch state of an input. So bit 0 is input 0 and will be 1 if it's touched and 0 if it's not touched, bit 1 would be input 1, etc. For example to test if input 0 and 11 are being touched with one call you could run code like:

```
# Call touched to get a 12-bit value with the state of each MPR121 input.
touch = mpr121.touched()
# Test if exactly bit 0 and 11 are set to 1, i.e. input 0 and 11 are touched.
if touch == 0b100000000001:
    print('Input 0 and 11 touched!')
# Or test if either bit 0 or 11 are set to 1, i.e. input 0 or 11 are touched:
if touch &amp;amp; 0b100000000000 &amp;gt; 0 or touch &amp;amp; 0b000000000001 &amp;gt; 0:
    print('Input 0 or 11 touched!')
```

That's all there is to using the MPR121 module with CircuitPython!