Adafruit PC Joystick to seesaw I2C Adapter

Created by Liz Clark

https://learn.adafruit.com/adafruit-pc-joystick-to-seesaw-i2c-adapter

Last updated on 2023-09-05 12:52:12 PM EDT
# Table of Contents

**Overview** 3

**Pinouts** 5
- Power Pins
- I2C Logic Pins
- Interrupt Pin and LED
- UPDI Pin
- 5V LED
- DE-15 Port

**CircuitPython and Python** 6
- CircuitPython Microcontroller Wiring
- Python Computer Wiring
- Python Installation of seesaw Library
- CircuitPython Usage
- Python Usage
- Example Code

**Python Docs** 11

**Arduino** 11
- Wiring
- Library Installation
- Example Code

**Arduino Docs** 15

**Downloads** 15
- Files
- Schematic and Fab Print
Ladyada writes: Back in my day, we used to have these big chunky PC joysticks that would plug into the back of your PC. No, not through USB, they had a DE-15 port and they'd go into the Game Port your sound effects card. Now, of course, we have tons of USB ports and quality audio is built into every computer. But what if you want to relive your Commander Keen days?

This board has a 15 pin D-Sub socket port, that you can plug any classic PC joystick into. Onboard is a ATtiny816 helper chip that acts as a I2C peripheral using our ’seesaw’ library. Two STEMMA QT ports mean you can plug this board in using a
4-pin JST SH cable to any microcontroller or microcomputer with I2C support and read the joystick X & Y plus the two buttons, even if your main chip doesn't have analog inputs. There's even a 5V power generator on-board since joysticks are expecting that voltage.

Great for converting old controllers without needing to cut any cables. We have example code for Arduino and CircuitPython / Python to make your next pew pew project a breeze.

Ships fully assembled and tested. If you prefer working on a breadboard, each order comes with a small piece of header. You'll need to solder the header onto the PCB, but it's fairly easy and takes only a few minutes even for a beginner.
Pinouts

The default I2C address is 0x49.

Power Pins

- VIN - This is the power pin. To power the board, give it the same power as the logic level of your microcontroller - e.g. for a 3V microcontroller like a Feather RP2040, use 3V, or for a 5V microcontroller like Arduino, use 5V. There is a 5V boost converter onboard to supply an attached PC joystick with 5V.
- GND - This is common ground for power and logic.

I2C Logic Pins

The default I2C address is 0x49.

- SCL - I2C clock pin, connect to your microcontroller I2C clock line. There's a 10K pullup on this pin.
- SDA - I2C data pin, connect to your microcontroller I2C data line. There's a 10K pullup on this pin.
- STEMMA QT () - These connectors allow you to connect to development boards with STEMMA QT / Qwiic connectors or to other things with various associated accessories ().
Interrupt Pin and LED

- **IRQ** - This is the interrupt output pin. It can be configured to pulse low whenever an input from an attached PC joystick is detected so you do not have to spam-read the I2C port to detect input.
- **Interrupt LED** - On the front of the board, directly above the IRQ pin, is the interrupt LED. It is the red LED and turns on whenever an interrupt is detected.

UPDI Pin

- **UPDI** - This is the single-pin Unified Program and Debug Interface. This pin is for external programming or on-chip-debugging for the ATtiny816 running the seesaw firmware. We have a page in the ATtiny Breakouts with seesaw Learn Guide detailing how to reprogram these chips with your own firmware (at your own risk). We don't provide any support for custom builds of seesaw - we think this is cool and useful for the Maker community.

5V LED

- **5V** - On the front of the board, next to the IRQ LED, is the 5V LED, labeled 5V. It is the green LED. It acts as a power indicator LED that receives power from the 5V boost converter.

DE-15 Port

The large, chonky port on the adapter board is a 15 pin D-Sub socket port. Your vintage analog PC joystick will plug into this port, just like how you used to plug into your Game Port on your PC add-on card. You'll be able to read the joystick X & Y plus button inputs.

If you're curious what these analog joysticks look like inside, check out the teardown that Ladyada did with the Gravis Advanced Classic PC Joystick in the video below.

CircuitPython and Python

It's easy to use the PC Joystick to seesaw adapter with Python or CircuitPython, and the Adafruit_CircuitPython_seesaw module. This module allows you to easily write Python code that reads the movement of the joystick and button inputs over I2C.
You can use this adapter with any CircuitPython microcontroller board or with a computer that has GPIO and Python thanks to Adafruit_Blinka, our CircuitPython-for-Python compatibility library.

CircuitPython Microcontroller Wiring

First wire up an I2C adapter to your board exactly as follows. The following is the adapter wired to a Feather RP2040 using the STEMMA connector:

- Board STEMMA 3V to adapter VIN (red wire)
- Board STEMMA GND to adapter GND (black wire)
- Board STEMMA SCL to adapter SCL (yellow wire)
- Board STEMMA SDA to adapter SDA (blue wire)

The following is the adapter wired to a Feather RP2040 using a solderless breadboard:

- Board 3V to adapter VIN (red wire)
- Board GND to adapter GND (black wire)
- Board SCL to adapter SCL (yellow wire)
- Board SDA to adapter SDA (blue wire)

Python Computer Wiring

Since there are dozens of Linux computers/boards you can use, we will show wiring for Raspberry Pi. For other platforms, please visit the guide for CircuitPython on Linux to see whether your platform is supported.

Here’s the Raspberry Pi wired with I2C using the STEMMA connector:
Here's the Raspberry Pi wired with I2C using a solderless breadboard:

Python Installation of seesaw Library

You'll need to install the Adafruit_Blinka library that provides the CircuitPython support in Python. This may also require enabling I2C on your platform and verifying you are running Python 3. Since each platform is a little different, and Linux changes often, please visit the CircuitPython on Linux guide to get your computer ready!

Once that's done, from your command line run the following command:

```
• pip3 install adafruit-circuitpython-seesaw
```

If your default Python is version 3 you may need to run 'pip' instead. Just make sure you aren't trying to use CircuitPython on Python 2.x, it isn't supported!
CircuitPython Usage

To use with CircuitPython, you need to first install the Adafruit_CircuitPython_seesaw library, and its dependencies, into the lib folder on your CIRCUITPY drive. Then you need to update code.py with the example script.

Thankfully, we can do this in one go. In the example below, click the Download Project Bundle button below to download the necessary libraries and the code.py file in a zip file. Extract the contents of the zip file, and copy the entire lib folder and the code.py file to your CIRCUITPY drive.

Your CIRCUITPY/lib folder should contain the following folders and file:

- adafruit_bus_device/
- adafruit_seesaw/
- adafruit_pixelbuf.mpy

Python Usage

Once you have the library pip3 installed on your computer, copy or download the following example to your computer, and run the following, replacing code.py with whatever you named the file:

```
python3 code.py
```

Example Code

If running CircuitPython: Once everything is saved to the CIRCUITPY drive, connect to the serial console () to see the data printed out!

If running Python: The console output will appear wherever you are running Python.
import time
import board
from micropython import const
from adafruit_seesaw.seesaw import Seesaw

BUTTON_1 = const(3)
BUTTON_2 = const(13)
BUTTON_3 = const(2)
BUTTON_4 = const(14)

JOY1_X = const(1)
JOY1_Y = const(15)
JOY2_X = const(0)
JOY2_Y = const(16)

button_mask = const((1 << BUTTON_1) | (1 << BUTTON_2) | (1 << BUTTON_3) | (1 << BUTTON_4))

i2c_bus = board.STEMMA_I2C()  # The built-in STEMMA QT connector on the microcontroller
# i2c_bus = board.I2C()  # Uses board.SCL and board.SDA. Use with breadboard.
seesaw = Seesaw(i2c_bus, addr=0x49)

seesaw.pin_mode_bulk(button_mask, seesaw.INPUT_PULLUP)

last_x = 0
last_y = 0
x = 0
y = 0

while True:
    # These joysticks are really jittery so let's take 4 samples of each axis
    for i in range(4):
        x += seesaw.analog_read(JOY1_X)
        y += seesaw.analog_read(JOY1_Y)

    # take average reading
    x /= 4
    y /= 4

    # PC joysticks aren't true voltage divider because we have a fixed 10K
    # we dont know the normalized value so we're just going to give you
    # the result in 'Kohms' for easier printing
    x = 1024 / x - 1
    y = 1024 / y - 1

    if (abs(x - last_x) > 3) or (abs(y - last_y) > 3):
        print(x, y)
    last_x = x
    last_y = y

    buttons = seesaw.digital_read_bulk(button_mask)

    if not buttons & (1 << BUTTON_1):
        print("Button 1 pressed")

    if not buttons & (1 << BUTTON_2):
        print("Button 2 pressed")

    if not buttons & (1 << BUTTON_3):
        print("Button 3 pressed")
if not buttons & (1 << BUTTON_4):
    print("Button 4 pressed")
    time.sleep(0.01)

Plug in a PC joystick to the 15 pin connector socket port on the adapter. Then, start up the example code. The serial console will print out whenever a button is pressed or the joystick is moved.

The loop uses some math to take an average reading of the two analog pins for the joystick potentiometers since they can be really jittery. This provides a more stable reading from the controller.

**Python Docs**

- [Python Docs ()](#)

**Arduino**

Using the PC joystick to seesaw adapter with Arduino involves wiring up the adapter to your Arduino-compatible microcontroller, installing the [Adafruit_Seesaw ()](#) library, plugging in a PC joystick to the adapter and running the provided example code.

**Wiring**

Wire as shown for a 5V board like an Uno. If you are using a 3V board, like an Adafruit Feather, wire the board's 3V pin to the adapter VIN.

Here is an Adafruit Metro wired up to the adapter using the STEMMA QT connector:
Board 5V to adapter VIN (red wire)
Board GND to adapter GND (black wire)
Board SCL to adapter SCL (yellow wire)
Board SDA to adapter SDA (blue wire)

Here is an Adafruit Metro wired up using a solderless breadboard:

Library Installation

You can install the Adafruit_Seesaw library for Arduino using the Library Manager in the Arduino IDE.

Click the Manage Libraries ... menu item, search for Adafruit_Seesaw, and select the Adafruit seesaw Library library:
If asked about dependencies, click "Install all".

If the "Dependencies" window does not come up, then you already have the dependencies installed.

If the dependencies are already installed, you must make sure you update them through the Arduino Library Manager before loading the example!

Example Code

```cpp
#include "Adafruit_seesaw.h"

// creates seesaw on I2C0 port
Adafruit_seesaw ss = Adafruit_seesaw(&Wire);
// uncomment this for using I2C1, such as STEMMA port on QT Py RP2040
// Adafruit_seesaw ss = Adafruit_seesaw(&Wire1);

#define BUTTON_1         3
#define BUTTON_2         13
#define BUTTON_3         2
#define BUTTON_4         14

uint32_t button_mask = (1UL << BUTTON_1) | (1UL << BUTTON_2) |
(1UL << BUTTON_3) | (1UL << BUTTON_4);

#define JOY1_X            1
#define JOY1_Y           15
```
```c
#define JOY1_X 0
#define JOY1_Y 16
#define JOY2_X 0
#define JOY2_Y 16

#define IRQ_PIN 5

void setup() {
  Serial.begin(115200);
  while(!Serial) {
    delay(10);
  }
  Serial.println("PC Joystick QT example!");
  ss.begin(0x49);
  Serial.println("seesaw started");
  uint32_t version = ((ss.getVersion() >> 16) & 0xFFFF);
  if (version != 5753) {
    Serial.print("Wrong firmware loaded? ");
    Serial.println(version);
    while(1) delay(10);
  }
  Serial.println("Found Product 5753");
  ss.pinModeBulk(button_mask, INPUT_PULLUP);
  ss.setGPIOInterrupts(button_mask, 1);
  #if defined(IRQ_PIN)
    pinMode(IRQ_PIN, INPUT);
  #endif
}

float last_x = 0, last_y = 0;

void loop() {
  delay(10); // delay in loop to slow serial output
  float x = 0, y = 0;
  // These joysticks are really jittery so lets take 4 samples of each axis
  for (int s=0; s<4; s++) {
    x += ss.analogRead(JOY1_X);
    y += ss.analogRead(JOY1_Y);
  }
  x /= 4.0; // Take the average of the 4 samples
  y /= 4.0;
  // PC joysticks aren't "true" voltage divider, because we have a fixed 10K
  // we dont know the 'normalized' value so we're just going to give you
  // the result in 'Kohms' for easier printing
  x = (1024.0/(float)x - 1);
  y = (1024.0/(float)y - 1);
  if ( (fabs(x - last_x) > 0.1) || (fabs(y - last_y) > 0.1) ) {
    Serial.print(x); Serial.print(", "); Serial.println(y);
    last_x = x;
    last_y = y;
  }
  #if defined(IRQ_PIN)
    if(!digitalRead(IRQ_PIN)) {
      return;
    }
  #endif
}

uint32_t buttons = ss.digitalReadBulk(button_mask);
//Serial.println(buttons, BIN);
  if (! (buttons & (1UL << BUTTON_1))) {
    Serial.println("Button 1 pressed");
  }
  if (! (buttons & (1UL << BUTTON_2))) {
    //
    //
  }
```

Serial.println("Button 2 pressed");
}
if (! (buttons & (1UL << BUTTON_3))) {
    Serial.println("Button 3 pressed");
}
if (! (buttons & (1UL << BUTTON_4))) {
    Serial.println("Button 4 pressed");
}

Plug in a PC joystick to the 15 pin socket port on the adapter. Upload the sketch to your board and open up the Serial Monitor (Tools -> Serial Monitor) at 115200 baud. You'll see the seesaw firmware recognized by the code. Then, when you press the buttons or move the joystick on your connected controller it will print to the Serial Monitor.

Arduino Docs

Downloads

Files

- ATtiny816 Datasheet ()
- PC Game Port Pinout ()
- IBM Game Control Adapter manual () (PDF)
- EagleCAD PCB Files on GitHub ()
- Fritzing object in the Adafruit Fritzing Library ()