# Table of Contents

## Overview

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
</tr>
</tbody>
</table>

## Pinouts

- Power Pins
- I2C Logic Pins
- Wii Nunchuck Connector

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
</tr>
</tbody>
</table>

## Arduino Use

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
</tr>
</tbody>
</table>

## Python & CircuitPython

- CircuitPython Microcontroller Wiring
- Python Computer Wiring
- CircuitPython Installation of Nunchuck Library
- Python Installation of Nunchuck Library
- CircuitPython & Python Usage
- Full Example Code
- Nunchuck Accelerometer and Tap Detection

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
</tr>
</tbody>
</table>

## Python Docs

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
</tr>
</tbody>
</table>

## Downloads

- Files:
- Schematic
- Fab Print

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
</tr>
</tbody>
</table>

## Sample Project: Nunchuck NeoPixel Ring

- Parts
- Wiring
- Code

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
</tr>
</tbody>
</table>
Overview

Dig out that old Wii controller and use it as a sleek controller for your next robot if you like. The Adafruit Adafruit Wii Nunchuck Breakout Adapter fits snugly into the Wii connector, and performs the level shifting and power regulation needed to use the controller with any microcontroller or microcomputer.

The Wii controllers use a standard I2C interface, and there's existing code for both Arduino (https://adafruit.com) and CircuitPython/Python for quick integration (https://adafruit.com) with an Arduino UNO, Feather, or even a Raspberry Pi. We like to use these with the Wii Nunchuck, as you can get an X-Y joystick, two buttons and an accelerometer all in one hand-held
package. All data is transmitted over I2C address 0x52, and the address cannot be changed.

We use extra thicc 2.0mm PCBs for this breakout, and made cut-outs for the grabber-notches, so that the controller connection is snug, and won't rattle or come loose!

To make using it as easy as possible, we've created this breakout in Stemma QT form factor. You can either use a breadboard or the SparkFun qwiic compatible STEMM A QT connectors, and compatibility with 5V voltage levels as commonly found on Arduinos, as well as 3.3V logic used by many other boards like the Raspberry Pi or our Feathers. QT Cable is not included, but we have a variety in the shop for quick plug-and-play support.
Doesn't come with the Wii Nunchuck controller, that's sold separately (or just look in that plastic bin in your parent's attic next time you visit).

---

**Pinouts**

---

**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 5V microcontroller like Arduino, use 5V
• 3V - This is the 3.3V output from the voltage regulator, you can grab up to 100mA from this if you like.
• GND - common ground for power and logic

I2C Logic Pins

Default address is 0x52 - this cannot be changed.

• SCL - I2C clock pin, connect to your microcontroller I2C clock line. This pin is level shifted so you can use 3-5V logic, and there's a 10K pullup on this pin.
• SDA - I2C data pin, connect to your microcontroller I2C data line. This pin is level shifted so you can use 3-5V logic, and there's a 10K pullup on this pin.
• STEMMA QT () - These connectors allow you to connectors to dev boards with STEMMA QT connectors or to other things with various associated accessories ()

Wii Nunchuck Connector

Wii controllers use a standard I2C interface. We use extra thick 2.0mm PCBs for this breakout, and made cut-outs for the grabber-notches, so that the controller connection is snug, and wont rattle or come loose!

See the GIF for how to install. The notched side of the connector U goes on the side that says Notch Up!
Arduino Use

Connect the Wii Nunchuck Breakout Adapter as shown below using the STEMMA QT connector or a solderless breadboard.

Connect board VIN (red wire) to Arduino 5V if you are running a 5V board Arduino (Uno, etc.). If your board is 3V, connect to that instead.

Connect board GND (black wire) to Arduino GND

Connect board SCL (yellow wire) to Arduino SCL

Connect board SDA (blue wire) to Arduino SDA

There are existing Arduino libraries you can use with the Wii series of devices. We'll be using the popular WiiChuck library (https://github.com/adafruit/WiiChuck) which has support for the following devices:

- Nunchuk
- Classic Controller
- Guitar Hero Guitar
- Guitar Hero Drums
- DJ Hero
- Drawesome Tablet
- Taiko Drums

Install it by searching the Arduino Library Manager for WiiChuck
You can use this sketch, again it barely fits on a 32u4 or 328 Arduino but it will connect and display data from a Nunchuk type controller!

```cpp
#include <WiiChuck.h>

Accessory nunchuck;

void setup() {
  Serial.begin(115200);
  nunchuck.begin();
  if (nunchuck.type == Unknown) {
    nunchuck.type = NUNCHUCK;
  }
}

void loop() {
  nunchuck.readData();    // Read inputs and update maps
  Serial.print("X: ");  Serial.print(nunchuck.getAccelX());
  Serial.print(" Y: ");  Serial.print(nunchuck.getAccelY());
  Serial.print(" Z: ");  Serial.println(nunchuck.getAccelZ());
  Serial.print("Joy: (");
  Serial.print(nunchuck.getJoyX());
  Serial.print(", ");
  Serial.print(nunchuck.getJoyY());
  Serial.println("") ;
  Serial.print("Button: ");
  if (nunchuck.getButtonZ()) Serial.print(" Z ");
  if (nunchuck.getButtonC()) Serial.print(" C ");
  Serial.println();
  delay(100);
}
```

Open the serial console at 115200 baud to see data streaming out, you can see the X Y Z accelerometer data (ranges from 0 to 1023), X and Y from the thumbstick (ranges from 0-255, with ~127 in the center position), and the two trigger buttons Z and C
It's easy to use the Wii Nunchuck Breakout Adapter with CircuitPython and the Adafruit CircuitPython Nunchuk module. This module allows you to easily write Python code that reads controls from the Wii Nunchuck.

You can use this sensor 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 a Wii Nunchuck Adapter Breakout to your board exactly as follows. Here is an example of the Nunchuck Adapter wired to a Feather using I2C:
Board 3V to sensor VIN (red wire)
Board GND to sensor GND (black wire)
Board SCL to sensor SCL (yellow wire)
Board SDA to sensor SDA (blue wire)

Python Computer Wiring

Since there’s 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:
CircuitPython Installation of Nunchuck Library

You’ll need to install the Adafruit CircuitPython Nunchuk () library on your CircuitPython 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 (). Our CircuitPython starter guide has a great page on how to install the library bundle ()

Copy the following files from the bundle to the lib folder on your CIRCUITPY drive:

- adafruit_nunchuk.mpy
- adafruit_bus_device
Before continuing make sure your board's lib folder or root filesystem has the adafruit_nunchuk.mpy, and adafruit_bus_device file and folder copied over.

Next connect to the board's serial REPL so you are at the CircuitPython >>> prompt.

Python Installation of Nunchuck 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:

```
ip3 install adafruit-circuitpython-nunchuk
```

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 & Python Usage

To demonstrate the usage of the sensor we'll initialize it and read from the controller in the board's Python REPL.

Run the following code to import the necessary modules and initialize the I2C connection with the adapter:

```python
import time
import board
import adafruit_nunchuk
nc = adafruit_nunchuk.Nunchuk(board.I2C())
```

Now you're ready to read values from the controller using these properties:

- joystick - An (x, y) tuple of the joystick position.
- button_C - The pressed state of button C.
- button_Z - The pressed state of button Z.
- acceleration - An (x, y, z) tuple of acceleration data.

For example, to read the joystick position:
```python
x, y = nc.joystick
print("joystick = {},{}\".format(x, y))
```

To read button presses on the+ buttons:

```python
print(nc.button_C)
print(nc.button_Z)
```

To read the acceleration data:

```python
ax, ay, az = nc.acceleration
print("acceleration ax={}, ay={}, az={}\".format(ax, ay, az))
```

That's all there is to using the Adafruit Wii Nunchuk Breakout Adapter with Python and CircuitPython!

## Full Example Code

This example reads all the data in a loop.

```python
# SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
# SPDX-License-Identifier: MIT

import time
import board
import adafruit_nunchuk

i2c = board.I2C()  # uses board.SCL and board.SDA
# i2c = board.STEMMA_I2C()  # For using the built-in STEMMA QT connector on a
# microcontroller
nc = adafruit_nunchuk.Nunchuk(i2c)

while True:
    x, y = nc.joystick
    ax, ay, az = nc.acceleration
    print("joystick = {},{}\".format(x, y))
    print("acceleration ax={}, ay={}, az={}\".format(ax, ay, az))

    if nc.buttons.C:
        print("button C")
    if nc.buttons.Z:
        print("button Z")
    time.sleep(0.5)
```

## Nunchuck Accelerometer and Tap Detection

This section and code come from notes by CGrover used to develop tap detection for potential use in the PowerWash Simulator Controller project.
The accelerometer is very securely mounted, so the tap detection scheme only monitors the more sensitive x-axis (horizontal).

Nunchuck tap detection works best if the side of the controller handle is tapped.

Double-Tap Detection Code

```
# Set the Nunchuck tap detector parameters and initialize tap event history list
TAP_THRESHOLD = 350  # Tap sensitivity threshold; depends on the physical sensor mount
TAP_DEBOUNCE = 0.2  # Time for accelerometer to settle after tap (seconds)
TAP_TIMEOUT = 1500  # Remove tap event from history timeout (milliseconds)
tap_events = []  # Initialize the tap event history list

# The project's main while loop
while True:
    # Detect a tap on the x-axis of the Nunchuck accelerometer
    accel_sample_1 = nunchuck.acceleration  # Read one sample
    accel_sample_2 = nunchuck.acceleration  # Read the next sample
    if abs(accel_sample_1 - accel_sample_2) >= TAP_THRESHOLD:
        # The difference between two consecutive samples exceeded the threshold
        # (equivalent to a high-pass filter)
        print(f"SINGLE tap detected {ticks_ms()}")
        tap_events.append(ticks_ms() + TAP_TIMEOUT)  # save tap expiration time in event stack
        time.sleep(TAP_DEBOUNCE)  # Debounce delay

    # Clean up tap event history after timeout period expires
    if len(tap_events) > 0:
        # Check for expired events
        if tap_events[0] <= ticks_ms():
            # The oldest event has expired
            tap_events = tap_events[1:]  # Remove the oldest event

    # Check see if two taps are in the event history list
    if len(tap_events) == 2:
        # Double-tap: execute the task and clear event history
        print(f"DOUBLE tap detected {ticks_ms()}")
        # Perform the double-tap task here
        tap_events = []  # Clear event history
```
Python Docs

Downloads

Files:

- Fritzing object in the Adafruit Fritzing Library
- EagleCAD PCB files on GitHub
Sample Project: Nunchuck NeoPixel Ring

Here's an example project you can do with your Nunchuck breakout to control some NeoPixels with joystick, button, and motion from a Wii Nunchuck.

Parts

Adafruit Wii Nunchuck Breakout Adapter
Dig out that old Wii controller and use it as a sleek controller for your next robot if you like. The Adafruit Adafruit Wii Nunchuck Breakout Adapter fits snugly into the Wii connector...  
https://www.adafruit.com/product/4836

Wii controller (Nunchuck / Wiichuck)
This is a generic Wii Nunchuck controller, we haven't tried it with a Wii but it does work great with the Video Game shield, and all the microcontroller code we tried. May come in...  
https://www.adafruit.com/product/342

You can use pretty much any CircuitPython-capable board for this, I had a Metro M4 Airlift handy.

Adafruit Metro M4 Express AirLift (WiFi) - Lite
Give your next project a lift with AirLift - our witty name for the ESP32 co-processor that graces this Metro M4. You already know about the Adafruit Metro...  
https://www.adafruit.com/product/4000
This will work with any NeoPixel strand, strip, or ring.

**NeoPixel 1/4 60 Ring - 5050 RGBW LED w/ Integrated Drivers**
What is better than smart RGB LEDs?
Smart RGB+White LEDs! These NeoPixels now have 4 LEDs in them (red, green, blue and white) for excellent lighting effects. Round and round...
https://www.adafruit.com/product/2874

**STEMMA QT / Qwiic JST SH 4-pin to Premium Male Headers Cable**
This 4-wire cable is a little over 150mm / 6" long and fitted with JST-SH female 4-pin connectors on one end and premium Dupont male headers on the other.
Compared with the...
https://www.adafruit.com/product/4209

**Wiring**

Use the STEMMA QT/qwiic to male header pin adapter to plug the Nunchuck breakout into your microcontroller's 3.3V, GND, SDA, & SCL lines.

Wire your NeoPixels into the board's 5V, GND, and D6 (to NeoPixel DIN) pins (if your board only has 3.3V that's usually fine, too, but NeoPixels prefer 5V when available).
Then, plug your Nunchuck controller into the adapter, making sure to follow the silkscreened "NOTCH UP" guideline.

**Code**

Set up your board as shown on the previous page, then download this code onto the board as code.py.

```python
# SPDX-FileCopyrightText: 2021 John Park for Adafruit Industries
# SPDX-License-Identifier: MIT

import time
import board
import adafruit_nunchuk
import neopixel
import simpleio

i2c = board.I2C()  # uses board.SCL and board.SDA
# i2c = board.STEMMA_I2C()  # For using the built-in STEMMA QT connector on a microcontroller
nc = adafruit_nunchuk.Nunchuk(i2c)

# create neopixel object
NEOPIN = board.D6
NEOLENGTH = 60
NEOORDER = neopixel.GRBW  # set to GRB for 'regular' RGB NeoPixels
pixels = neopixel.NeoPixel(
    NEOPIN, NEOLENGTH, brightness=0.1, auto_write=False, pixel_order=NEOORDER
)

RED = (220, 0, 0)
PURPLE = (80, 0, 160)
PINK = (100, 0, 80)
GREEN = (0, 180, 0)
CYAN = (0, 80, 100)
BLUE = (0, 0, 255)
BLACK = (0, 0, 0)
COLORS = [RED, PURPLE, PINK, GREEN, CYAN, BLUE]
pix = 0  # selected pixel
```

© Adafruit Industries
Now, you can control your LEDs with the nunchuck!

- move the joystick up and down to choose how many pixels are lit
- tap the Z button to change colors
- hold the C button while tilting the controller from side to side to adjust brightness. Let go of the C button to lock in that brightness level