Adafruit PiCowbell Proto for Pico

Created by Kattni Rembor

https://learn.adafruit.com/picowbell,proto

Last updated on 2023-08-29 04:52:05 PM EDT
Table of Contents

Overview 3

Pinouts 6
  • Power and Power-Related Pins
  • Power Pins
  • Ground Pins
  • Power-Related Pins
  • I2C Logic Pins
  • Reset Button
  • GPIO Pins
  • Proto Area

Assembly 12

Pico 13
  • Assembly Steps

Stacking Headers 15
  • Assembly Steps

Socket Headers 18
  • Assembly Steps

Shorty Socket Headers 21
  • Assembly Steps

Direct to Pico Headers 24
  • Assembly Steps

CircuitPython I2C Scan 27
  • Wiring
  • I2C Scan Code
  • I2C Object Creation using board.STEMMA_I2C()

Arduino I2C Scan 29
  • Wiring
  • I2C Scan Code

Downloads 31
  • Files:
  • Schematic and Fab Print
Overview

Ding dong! Hear that? It's the PiCowbell ringing, letting you know that the new Adafruit PiCowbell Proto is finally in stock and ready to assist your Raspberry Pi Pico and Pico W project with handy hardware and practical prototyping.

The PiCowbell Proto is the same size and shape as a Pico, and is intended to socket underneath to make programming and sensor connectivity easy. Reset button? Yes! STEMMA QT / Qwiic connector for fast I2C? Indeed.
Please Note! To keep things flexible, this PiCowbell does not come with headers: there’s a lot of possible configurations and we stock various headers depending on how you want to solder and attach. Especially if you want the Pico on top, so that the BOOTSEL button and LED are accessible.

1. **Use the Pico Stacking Headers** () if you want to be able to plug into a breadboard or other accessory with sockets.
2. **Use the Pico Socket Headers** () if you want to plug directly in and have a nice solid connection that doesn’t have any poking-out-bits.
3. **Use the Short Socket Headers** () for a very slim but pluggable design, note that you’ll want to trim down the Pico’s headers or use the short plug headers on the Pico () to have a skinny sandwich.
4. Solder the PCB directly to the Pico headers - of course this is very compact and inexpensive but you won’t be able to remove the PiCowbell.

The PiCowbell Proto provides you with:

- Right angle reset button that sticks out the end
- Right angle JST SH connector for I2C / Stemma QT / Qwiic connection. Or can use it for plain GPIO wiring if you don’t have any I2C devices to attach. Provides 3V, GND, IO4 (SDA), and IO5 (SCL)
- Extra set of 4 breakout holes next to the JST SH in case you want more I2C connection or want to re-assign the I2C port.
- 13 rows of 4-hole connected strips in the center area. You can cut the traces between the holes but they’re intended to be treated like a mini-mini breadboard
- Every pad on the Pico has a duplicate hole pad next to it for solder-jumpering
• The ground pads have white silkscreen rectangles to easily identify, plus one long ground strip near the reset button
• One long strip of connected holes for 3.3V power
• Gold plated pads for easy soldering

Note that we do not have I2C pullups on the board, but your Qwiic/QT breakout board or accessory likely has them already on-board. If using the Philhower Arduino core, the Wire peripheral is already setup to use IO4 and IO5. If using CircuitPython or MicroPython you'll need to let the code know to look at 4+5 for SDA+SCL pins.
The two rows of pins along the sides of the PiCowbell Proto are connected to each other, to provide access to all of the pins, even when you solder headers onto the PiCowbell. Both rows of pins along the two sides of the PiCowbell match up exactly with Raspberry Pi Pico, Pico W, etc. The major difference from the Pico is the inclusion of the STEMMA QT connector which is connected to GPIO4 and GPIO5 (as well as power and ground). Let's take a look!

**Power and Power-Related Pins**

The following pins are either power or power related.
Power Pins

These are the available 3V power pins.

- 3V pins - The pair of pins highlighted in the image, along the top of the PiCowbell, are the standard Pico 3V out pins. These are used to power external devices, etc.
- 3V proto power rail - The 3V out Pico pin connects to a power rail that extends into the proto area. These pins all provide 3V power and are available to power anything you've connected to the proto area.
- 3V I2C pin - This pin next to the STEMMA QT connector is the 3V power pin for the broken out I2C pins. It can power a manually wired I2C sensor.

Ground Pins

These are the available GND pins. They are common ground for power and logic.
All GND pins are highlighted in white on the silk, except for the I2C GND pin. The GND pins that match the Pico exactly have square pads as well, as they do on the Pico.

- GND pins - There are pairs of GND pins along the top and bottom of the board, dispersed evenly throughout the other pairs of pins.
- GND proto rail - The right-most pairs of GND pins are connected to a GND rail that extends into the proto area. These pins are available to connect to anything you wire up in the proto area.
- GND I2C pin - This single G pin, at the upper right corner of the STEMMA QT connector is the GND pin for the broken out I2C pins. It can ground a manually wired I2C sensor.

**Power-Related Pins**

These pins are power-related.

- **VB (VBUS)** - This is the micro-USB input voltage, connected to the micro-USB port on Pico. It is nominally 5V.
- **VS (V_SYS)** - This is the main system input voltage, can range from 1.8V to 5.5V, and is used to generate the 3.3V needed for the RP2040 and the GPIO pins.
- **3V3_EN** - This connects to the enable pin, and is pulled high (to VSYS) via a 100kΩ resistor. To disable the 3.3V power, which also disables power to the Pico’s RP2040, short this pin low (to GND).
- **VR (ADC_VREF)** - This is the ADC power supply and reference voltage. It is generated on the Pico by filtering the 3.3V supply. It can be used with an external reference when ADC performance is required.
I2C Logic Pins

- C (SCL) - I2C clock pin, labeled C on the PiCowbell. It is connected to your microcontroller I2C clock line, which is GPIO5 on the Pico.
- D (SDA) - I2C data pin, labeled D on the PiCowbell. It is connected to your microcontroller I2C data line, which is GPIO4 on the Pico.
- STEMMA QT () - The PiCowbell Proto comes with a built in STEMMA QT connector! Located on the left end of the board, this connector means you can connect up all sorts of I2C sensors and breakout (), with no soldering required!

The I2C logic pins and STEMMA QT logic pins correspond with GPIO4 for I2C data (SDA), and GPIO5 for I2C clock (SCL).

When connected to the Pico, in CircuitPython, you can talk to the STEMMA connector using `board.STEMMA_I2C()`. In Arduino it is `Wire0`.

The STEMMA QT connector uses GPIO4 for I2C data (SDA), and GPIO5 for I2C clock (SCL). Make sure if you're calling the individual pins, that you are using the right ones!
Reset Button

- The PiCowbell Proto comes with a built in RESET button! Located on the right end of the board, it allows you to reset the Pico microcontroller and everything attached, without needing to unplug the USB connector. Use the reset button in conjunction with the BOOTSEL button on the Pico to enter the UF2 bootloader.

- The reset button is wired to the RUN pin, located towards the center of the top of the board. As with the other pins along the top and bottom, there are a connected pair of RUN pins available. This is the enable pin for the RP2040 on the Pico, and has an internal pullup to 3.3V. To reset the Pico, you can also short this pin low (to GND).

GPIO Pins

Along the top and bottom of the board are pairs of GPIO pins. The outside set matches the Pico pinout exactly, and is intended for soldering headers. The inside set is intended to make the pins available to connect up sensors and so on once you've attached the PiCowbell to a Pico.
GPIO Pins - Bottom Row

- GPIO0 through GPIO15 - Along the bottom of the board, from left to right, the pin pairs are GPIO0 through GPIO15 (with four pairs among them being GND pins). Note that all pin pairs are labeled on the silk, except for GPIO10, which is located to the left of the third GND pair, and to the right of GPIO11.

GPIO Pins - Top Row

- GPIO16 through GPIO22 - Along the top-right of the board, from right to left, the pin pairs are GPIO17 through GPIO22 (with two pairs among them being GND pins). Note that all pin pairs are labeled on the silk, except for GPIO21, which is located to the right of the second GND pair, and to the left of GPIO20.
- A0 (ADC0) through A2 (ADC2) - Along the top of the board, to the left of center, the pins are A0 through A2 (with one pair among them being GND pins). Note that all pin pairs are labeled on the silk, except for A1, which is located to the right of the GND pair, and to the left of A0.

Proto Area

In the center of the board, you'll find the proto area.

- The single column on the far left is four separate pins. They are not connected in any way.
- In the highlighted block more towards the right, there are twelve columns that are connected vertically. You can identify these columns by the white silk between each of the pins in a given column. Similar to a breadboard, wherein the rows on either half are connected, this allows you to wire something to one pin in a column, and use another pin in the same column to wire something else to the same pin.
Assembly

There are four ways to get your PiCowbell board working with your Pico. To keep things flexible, PiCowbells do not come with headers: there's a lot of possible configurations and we stock various headers depending on how you want to solder and attach. Especially since you want the Pico on top, so that the BOOTSEL button and LED are accessible.

The options are as follows.

1. Use the Pico Stacking Headers () if you want to be able to plug into a breadboard or other accessory with sockets.
2. Use the Pico Socket Headers () if you want to plug directly into the Pico and have a nice solid connection that doesn't have any poking-out-bits.
3. For some PiCowbells: Use the Short Socket Headers () for a very slim but pluggable design, note that you'll want to trim down the Pico's headers or use the short plug headers on the Pico () to have a skinny sandwich.
4. For some PiCowbells: Solder the PiCowbell directly to the standard headers already soldered to your Pico. Of course this is very compact and inexpensive but you won't be able to remove the PiCowbell. However, this method is not possible for some PiCowbell variants depending on the clearance of the components on the PiCowbell (i.e. the PiCowbell Adalogger and its coin cell battery holder).
The next page shows how to solder standard headers onto a Pico board. The following four pages walk you through each type of PiCowbell assembly so you can choose the one that will work best for you!

You MUST solder all of the pins for the PiCowbell to work! Soldering only a few pins, or not soldering at all are not sufficient!

If you're unsure about soldering up the Pico and PiCowbell, check out our FAQ on soldering.

Pico

Three out of four of the assembly methods included in this guide assume you have a Raspberry Pi Pico soldered up with standard male headers in preparation for using it with the PiCowbell Proto. This page will show you how to solder a set of standard headers to a Pico.

(The shorty header assembly method uses short male headers on the Pico. The soldering concept is exactly the same, but use the shorty male headers on the Pico instead of standard ones. You can follow these instructions with the shorty headers and you'll be set for that.)

Follow the steps below to solder the standard male headers to a Pico. The process is the same for all flavors of Pico, such as Pico W.
Assembly Steps

Use the Pico to line up the headers on a breadboard. This is the easiest way to ensure the headers are soldered on straight.

Solder the pins on each end of the two header strips, so the four corners of the Pico are soldered. This ensures the Pico and headers are attached properly while you continue to solder the rest of the pins.
Solder the rest of the pins.

Remove it from the breadboard. You're done!

For a bit more detail on the process of soldering standard male headers to a board, check out the How to Solder Headers' Male Headers page (!).

Stacking Headers

The first PiCowbell assembly method uses stacking headers, which allows you to use a breadboard with your PiCowbell-Pico sandwich. This is super helpful when you're still prototyping other parts of your project, or simply want jumper-wire access to the Pico pins in addition to the PiCowbell.

This page assumes you have already soldered standard male headers to your Pico. If you have not, please return to the Pico assembly page (!) and follow the steps there.

Follow the steps below to solder stacking headers to your PiCowbell.

Although these pages show the PiCowbell Proto, the soldering instructions are applicable for all PiCowbell boards.
Assembly Steps

Place a standard-header-soldered Pico upside down on the table, so the long side of the header pins are facing up. Press the female sockets of each stacking header onto one of the rows of standard headers attached to the Pico, until they are fully attached.

Ensure the PiCowbell is oriented correctly before beginning assembly. The PiCowbell should be top-down, so that you are looking at the bottom of the PiCowbell. The STEMMA QT connector should be on the same end as the Pico USB connector, and the reset button should be on the opposite end with the Pico debug pins.

The PiCowbell pins must match the pinout on the Pico.

Remember, the pins are labeled on the bottom of the Pico. In this case, that works well because they are labeled on both sides of the PiCowbell, allowing for direct comparison before attaching the PiCowbell to the stacking header assembly.

Ensure the PiCowbell is oriented properly before beginning soldering! If you solder it on upside down or backwards, it will not function properly!
Press the PiCowbell onto the male pins sticking up from the stacking headers. You may need to push the stacking header pins in or out a bit to get the PiCowbell attached.

With the stacking header male pins sticking up, the bottom of the PiCowbell should be facing up as well.

Solder the pins on each end of each stacking header, so that the opposite four corners of the PiCowbell are soldered on.

Solder the rest of the pins onto the PiCowbell.
You're done! Now you can attach the whole sandwich to a breadboard, have access to the pins via the breadboard, and still be able to use the PiCowbell as well.

Socket Headers

This PiCowbell assembly method uses female socket headers on the PiCowbell to create a standalone sandwich when attached to a Pico with standard male headers.

This page assumes you have already soldered standard male headers to your Pico. If you have not, please return to the [Pico assembly page](#) and follow the steps there.

Follow the steps below to solder socket headers to your PiCowbell.

> Although these pages show the PiCowbell Proto, the soldering instructions are applicable for all PiCowbell boards.
Assembly Steps

Place a standard-header-soldered Pico upside down on the table, so the long side of the header pins are facing up. Press the female sockets onto one of the rows of standard headers attached to the Pico, until both are fully attached.

Ensure the PiCowbell is oriented correctly before beginning assembly. The PiCowbell should be top-down, so that you are looking at the bottom of the Cowbell. The STEMMA QT connector should be on the same end as the Pico USB connector, and the reset button should be on the opposite end with the Pico debug pins.

The PiCowbell pins must match the pinout on the Pico.

Remember, the pins are labeled on the bottom of the Pico. In this case, that works well because they are labeled on both sides of the PiCowbell, allowing for direct comparison before attaching the PiCowbell to the stacking header assembly.
Ensure the PiCowbell is oriented properly before beginning soldering! If you solder it on upside down or backwards, it will not function properly!

Press the PiCowbell onto the pins sticking up from the socket headers. You may need to push the stacking header pins in or out a bit to get the PiCowbell attached.

Solder the pins on each end of each socket header, so that the opposite four corners of the PiCowbell are soldered on.
Solder the rest of the pins onto the PiCowbell.

That's it, you're done!

---

**Shorty Socket Headers**

This PiCowbell assembly method uses shorty female socket headers on the PiCowbell to create a standalone sandwich when attached to a Pico with shorty male headers.

This page assumes you have already soldered shorty male headers to your Pico. If you have not, please return to the [Pico assembly page](#) and follow the steps there. The page shows how to solder standard male headers to the Pico, but the concept is identical with the shorty headers.

Follow the steps below to solder shorty socket headers to your PiCowbell.

*Although these pages show the PiCowbell Proto, the soldering instructions are applicable for all PiCowbell boards.*
Assembly Steps

Solder the short male headers to the Pico. See the Pico assembly page for instructions on soldering headers to the Pico.

Place a shorty-header-soldered Pico upside down (headers up) on the table. Press the each of the short female sockets onto one of the rows of short headers attached to the Pico, until both are fully attached.

Ensure the PiCowbell is oriented correctly before beginning assembly. The PiCowbell should be top-down, so that you are looking at the bottom of the Cowbell. The STEMMA QT connector should be on the same end as the Pico USB connector, and the reset button should be on the opposite end with the Pico debug pins.

The PiCowbell pins must match the pinout on the Pico.

Remember, the pins are labeled on the bottom of the Pico. In this case, that works well because they are labeled on both sides of the PiCowbell, allowing for direct comparison before attaching the Cowbell to the stacking header assembly.
Press the PiCowbell onto the pins sticking up from the shorty female headers. You may need to push the shorty header pins in or out a bit to get the PiCowbell attached.

Solder the pins on each end of each female header, so that the opposite four corners of the PiCowbell are soldered on.

Try not to use too much solder on these four pins! The solder can wick into the associated female header socket, onto the inserted male pin, and permanently attach the two boards.

Do not use too much solder when tacking the four corners! It can wick into the female header and permanently attach the two boards!

CAREFULLY remove the partially soldered PiCowbell from the Pico, before continuing to solder the rest of the PiCowbell pins.

As stated above, too much solder on the shorty female header pins can wick into the associated header socket, onto the inserted male pin, and permanently attach the two boards.
Direct to Pico Headers

The shorty socket header assembly method has so far been the most compact option. What if there was an even skinnier option? Turns out there is!

This page assumes you have already soldered standard male headers to your Pico. If you have not, please return to the [Pico assembly page](#) and follow the steps there.

Once you have the standard male headers soldered to the Pico, it is possible to solder the PiCowbell Proto directly to the Pico. This results in a super compact little sandwich. Be aware, however, that this method means you cannot remove the PiCowbell without desoldering it.

Note that while this is the most compact option, it also means you cannot remove the PiCowbell. Only use this method if you want the connection to be permanent!

Follow the steps below to solder your PiCowbell Proto directly to your Pico.
Assembly Steps

Place a standard-header-soldered Pico upside down (headers up) on the table.

Ensure the PiCowbell is oriented correctly before beginning assembly. The PiCowbell should be top-down, so that you are looking at the bottom of the Cowbell. The STEMMA QT connector should be on the same end as the Pico USB connector, and the reset button should be on the opposite end with the Pico debug pins.

The PiCowbell pins must match the pinout on the Pico.

Remember, the pins are labeled on the bottom of the Pico. In this case, that works well because they are labeled on both sides of the PiCowbell, allowing for direct comparison before attaching the Cowbell to the stacking header assembly.

Ensure the PiCowbell is oriented properly before beginning soldering! If you solder it on upside down or backwards, it will not function properly!
Press the PiCowbell onto the pins sticking up from the standard headers on the bottom of the Pico. There will be a small gap on both sides due to the height of the reset button and STEMMA QT connector being taller than the plastic spacer on the Pico's standard male headers.

Solder the pins on each end of each standard header, so that the opposite four corners of the PiCowbell are soldered on. Ensure that you maintain an equal gap on both sides so the PiCowbell is horizontally straight.

Solder the rest of the pins onto the PiCowbell.
CircuitPython I2C Scan

Now that you've assembled your PiCowbell Proto and Pico, you're ready to connect one of many breakouts to the STEMMA QT connector and get going with no further soldering needed!

This page shows you how to use CircuitPython to scan I2C to show the address of any connected devices, which will indicate whether your breakout is connected and your PiCowbell is assembled correctly.

Wiring

This diagram uses the MCP9808. You can replace it with any STEMMA QT breakout and get similar results; the I2C address returned below may be different with a different breakout.

Simply connect your STEMMA QT cable (from the MCP9808 STEMMA QT connector to the PiCowbell Proto STEMMA QT connector.)
I2C Scan Code

You'll need to load the example below onto your Raspberry Pi Pico, and connect to the serial console to see the results printed out. To do so, 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 i2c_scan.zip zip file, and navigate to the directory PiCowbell_Proto/i2c_scan. Then open on the directory within that matches the version of CircuitPython you're using and copy the contents of that directory to your CIRCUITPY drive.

Once everything is saved to the CIRCUITPY drive, connect to the serial console to see the data printed out!

The MCP9808 default I2C address is 0x18. You've now verified your PiCowbell Proto and Pico are assembled properly, and that CircuitPython is detecting the I2C device you connected! Now you're ready to jump into whatever project you have planned!

I2C Object Creation using `board.STEMMA_I2C()`

One thing worth noting about this code, that will help you in all of your future STEMMA QT project endeavors, is the I2C object creation.

```python
i2c = board.STEMMA_I2C()
```
To create the I2C object, you assign `board.STEMMA_I2C()` to a `i2c` variable for use later in the code. Simple! But, how does it work?

As noted previously in this guide, the STEMMA QT connector on the PiCowbell Proto is connected to GPIO5 (SCL) and GPIO4 (SDA). In CircuitPython, we have assigned `board.STEMMA_I2C()` to use GPIO5 and GPIO4 to allow you a simpler way to create the I2C object when using the STEMMA QT connector.

**Arduino I2C Scan**

Now that you've assembled your PiCowbell Proto and Pico, you're ready to connect one of many breakouts to the STEMMA QT connector and get going with no further soldering needed!

This page shows you how to use Arduino to scan I2C to show the address of any connected devices, which will indicate whether your breakout is connected and your PiCowbell is assembled correctly.

**Wiring**

This diagram uses the MCP9808. You can replace it with any STEMMA QT breakout and get similar results; the I2C address returned below may be different with a different breakout.

Simply connect your STEMMA QT cable () from the MCP9808 STEMMA QT connector to the PiCowbell Proto STEMMA QT connector.

**I2C Scan Code**

```cpp
// SPDX-FileCopyrightText: 2021 Carter Nelson for Adafruit Industries
// SPDX-FileCopyrightText: 2022 Kattni Rembor for Adafruit Industries
// SPDX-License-Identifier: MIT
// -----------------------------
```
// PiCowbell Proto and Pico I2C scan code.
// Modified from https://playground.arduino.cc/Main/I2cScanner/
#include <Wire.h>
// Wire uses GPIO4 (SDA) and GPIO5 (SCL) automatically.
#define WIRE Wire
void setup() {
  WIRE.begin();
  Serial.begin(9600);
  while (!Serial)
    delay(10);
  Serial.println("PiCowbell Proto Pico I2C Scanner");
}
void loop() {
  byte error, address;
  int nDevices;
  Serial.println("Scanning...");
  nDevices = 0;
  for(address = 1; address < 127; address++)
  {
    // The i2c_scanner uses the return value of
    // the Write.endTransmission to see if
    // a device did acknowledge the address.
    WIRE.beginTransmission(address);
    error = WIRE.endTransmission();
    if (error == 0)
    {
      Serial.print("I2C device found at address 0x");
      if (address<16)
        Serial.print("0");
      Serial.print(address,HEX);
      Serial.println(" !");
      nDevices++;
    }
    else if (error==4)
    {
      Serial.print("Unknown error at address 0x");
      if (address<16)
        Serial.print("0");
      Serial.println(address,HEX);
    }
  }
  if (nDevices == 0)
    Serial.println("No I2C devices found");
  else
    Serial.println("done");
  delay(5000);           // wait 5 seconds for next scan
}

Upload the sketch to your board and open up the Serial Monitor (Tools -> Serial Monitor) at 115200 baud. You should see something like the following.
The MCP9808 default I2C address is 0x18. You've now verified your PiCowbell Proto and Pico are assembled properly, and that Arduino is detecting the I2C device you connected! Now you're ready to jump into whatever project you have planned!

Note that `Wire` represents both of the pins used by the STEMMA QT connector, which are GPIO4 (SDA) and GPIO5 (SCL). For more details on how `Wire` works, check out the [Arduino Pico Wire documentation](https://www.arduino.cc/en/Reference/Wire).

---

**Downloads**

**Files:**

- [EagleCAD PCB files on GitHub](https://github.com/)
- [3D models on GitHub](https://github.com/)
- [Fritzing object in the Adafruit Fritzing Library](https://www.arduino.cc/en/Reference/Wire)