

Real-Time Clock Setup — Adafruit PiRTC DS3231

Sets up a DS3231 battery-backed real-time clock on the Mustang 2 Raspberry Pi 5, so the Pi keeps accurate time in the field with no internet connection — needed for FT8, which requires the Pi’s clock to be within about 1 second of correct.

Works with the Adafruit PiRTC DS3231 board (Adafruit #4282), or any other DS3231 module wired to the Pi’s I2C pins.

Alternative: the Pi 5’s own onboard RTC. The Pi 5 (unlike earlier Pi boards) has a built-in RTC, but it only keeps time through a power-off if you add a battery to its J5 connector — a small 2-pin JST header near the USB-C power input. That connector is a *charging* circuit for a rechargeable cell (such as a LIR2032), not a plain coin-cell holder — don’t plug a non-rechargeable CR2032 into it. A Pi-5-specific RTC battery pack (25-64 mAh, sold by Waveshare, Pi Hut, etc.) needs no wiring and no setup script at all. The tradeoff: that small rechargeable cell self-discharges faster than a coin cell, so it holds time less well through long periods in storage between field days — the DS3231’s replaceable CR1220 holds accurate time for months to years unpowered, against days to a few weeks for the J5 battery. If this radio is powered up often enough to keep the J5 battery topped off, the onboard RTC alone is simpler and needs none of the rest of this document. The DS3231 below is the better choice for a radio that sits unused between trips.

Parts

- Adafruit PiRTC DS3231 (or equivalent DS3231 breakout)
- CR1220 coin cell (backup battery for the RTC)
- 4 jumper wires (female-to-female, if not using a HAT-style direct connection)

1. Identifying the board’s pins

Adafruit’s own documentation for this board never labels its 6-pin connector individually — by design, the board is meant to be pushed straight onto the Pi’s GPIO header with its mounting hole aligned, no wiring needed at all. Since this project wires it with loose jumpers instead (to avoid the Touch Display 2’s power pins and the PiLink UART pins elsewhere on the header), the 6 pins were identified by hand and confirmed two ways (2026-10-14): continuity from the coin-cell holder’s negative tab to one pin (that’s ground), and by physically laying the board’s header against the Pi 5’s own GPIO pinout chart with the board flipped so its bottom row lines up with the Pi’s pins 1/2 — all six positions then matched straight across with no contradictions:

```

      (toward the DS3231SN chip)
+-----+-----+
|      SCL      | GND (Neg of |   <- matches Pi pins 5, 6
|                | Battery tab) |
+-----+-----+
|      SDA      |   VCC (5V)  |   <- matches Pi pins 3, 4
+-----+-----+
| VCC (3.3V)    |   VCC (5V)  |   <- matches Pi pins 1, 2
+-----+-----+
(away from the chip; SQW/32K row below this, unused)
```

A multimeter check backs this up rather than contradicting it: 3.3V-to-SDA and 3.3V-to-SCL both read several megohms (not a pull-up’s few kOhm, and not a dead short either) — this board carries no onboard I2C pull-up resistors, it’s designed to rely on the Raspberry Pi’s own built-in pull-ups on GPIO2/GPIO3, so an “open” reading here is expected and correct, not a sign of a wrong pin.

Only VCC, GND, SDA, SCL are used here (there are two VCC holes on this board's header - either works, use whichever jumper routing is more convenient). The SQW/32K row is left unconnected.

If you have a different DS3231 board (not Adafruit's), its 4 pins may be labeled directly on the silkscreen (VCC/GND/SDA/SCL or similar) - check that first rather than assuming this same layout.

The board as labeled during this identification, and the Pi 5 GPIO chart it was matched against:

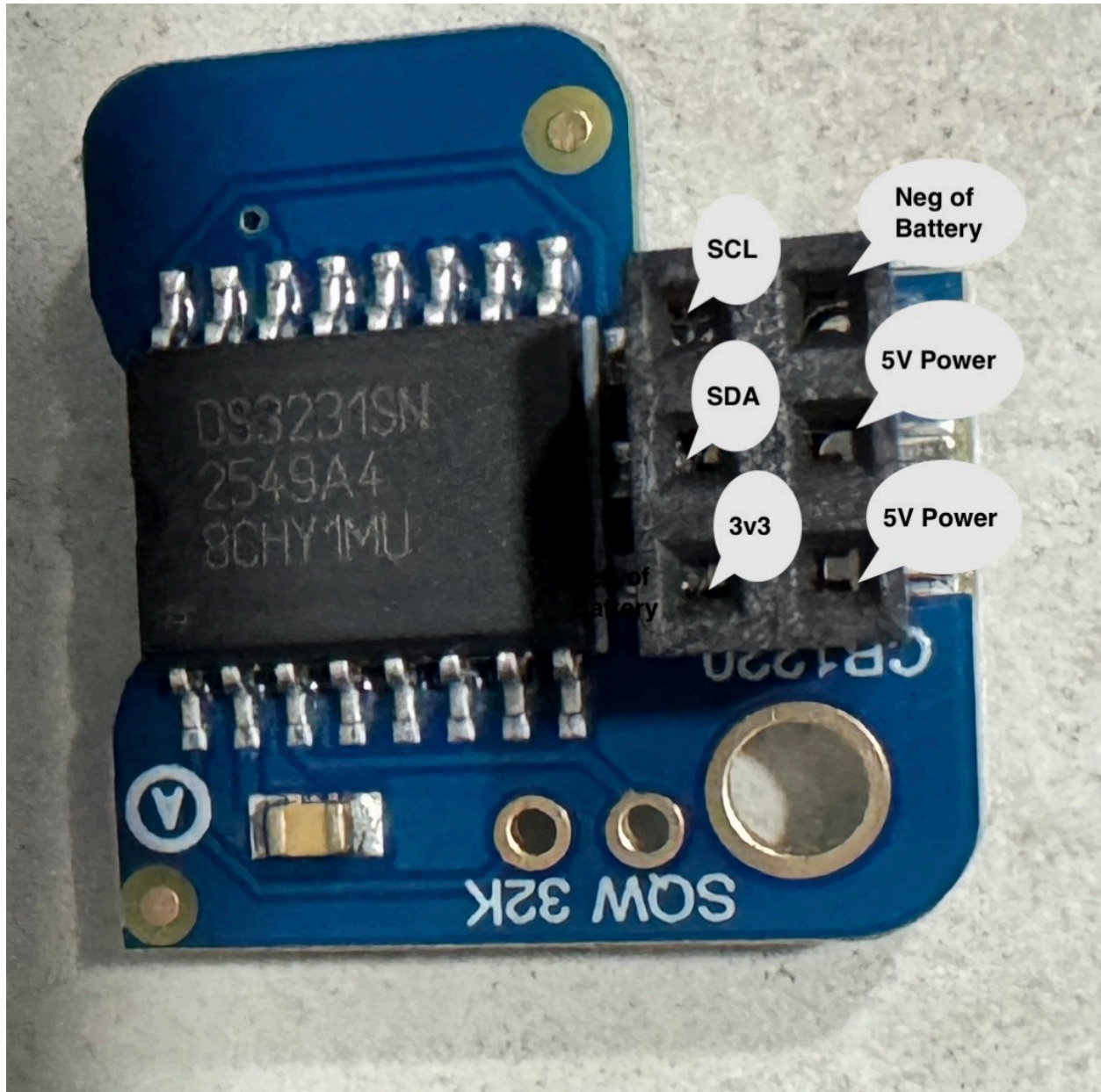


Figure 1: DS3231 board with pins labeled

2. Wiring

Connect with jumper wires to the Pi 5's 40-pin header:

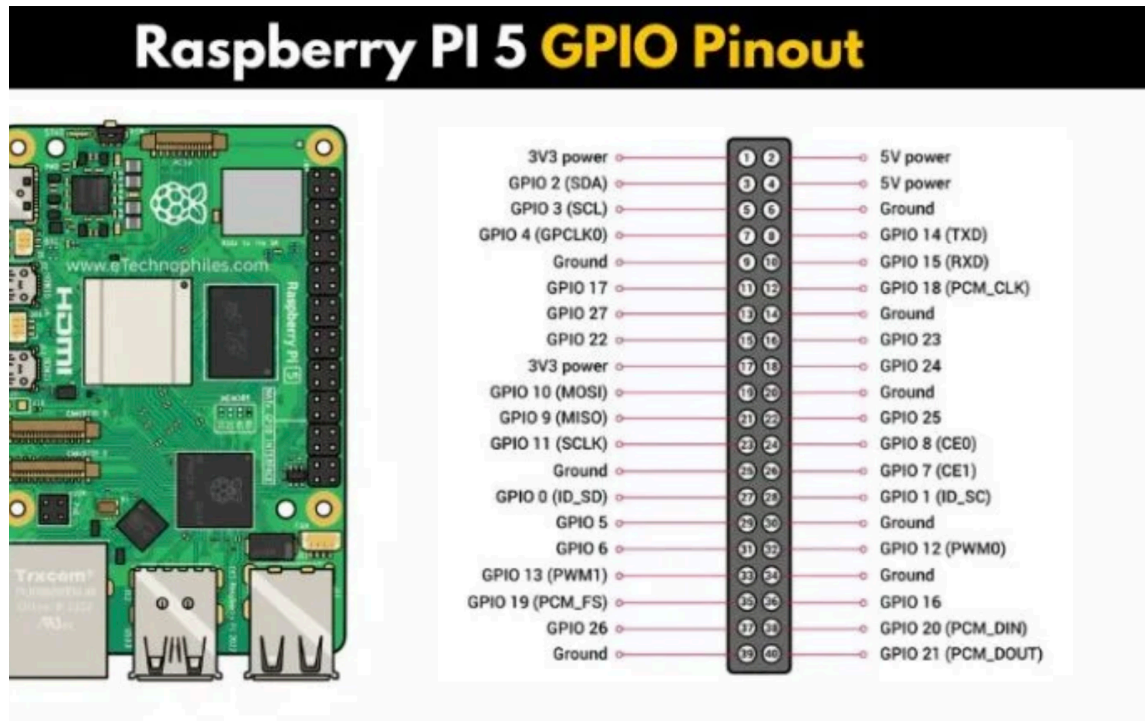


Figure 2: Raspberry Pi 5 GPIO pinout

RTC pin	Pi 5 header pin
VCC	Pin 1
SDA	Pin 3
SCL	Pin 5
GND	Pin 9

Don't plug the board straight onto the corner of the header. On this build, pins 2 and 6 already carry the Touch Display 2's power, and pins 8 and 10 carry the PiLink UART to the Teensy (see `INTEGRATION_PROCEDURE.md` Section 6). Use jumper wires to the four pins above instead of a stacked HAT connection.

Fit the CR1220 coin cell in the RTC board before powering up, so it keeps time through a power loss.

3. Run the setup script

Copy `PiDisplay/rtc/setup_ds3231.sh` to the Pi (already included in a normal `Sync_PiDisplay.command` sync), then on the Pi:

```
sudo bash ~/PiDisplay/rtc/setup_ds3231.sh
sudo reboot
```

What this does:

1. Turns on I2C and adds the DS3231 device-tree overlay to `/boot/firmware/config.txt`.
2. Installs `/usr/local/sbin/ds3231-sync.sh` plus two systemd units:

- `ds3231-boot.service` — at boot, sets the system clock from the DS3231 (unless network time has already set it first).
 - `ds3231-save.timer` — every 15 minutes, while network time is in sync, copies the system clock into the DS3231.
3. Turns off `fake-hwclock` if it's installed (it would otherwise put back a stale time at boot and mask the DS3231).

Note: the Pi 5 also has its own onboard RTC (`rtc0`, needs the separate J5 battery described above). The DS3231 shows up as another `/dev/rtcN` device; the script finds it by driver name, so either numbering works correctly, and the two RTCs don't conflict if both happen to be present.

4. First-time clock save

Once, after the reboot, with the Pi connected to the network so it has synced its clock over NTP:

```
sudo /usr/local/sbin/ds3231-sync.sh save
```

Check it any time with:

```
sudo /usr/local/sbin/ds3231-sync.sh show
```

This prints the DS3231's own time, the system clock, and whether NTP is currently synchronized.

After setup

From here on, the Pi:

- Sets its clock from the DS3231 at every boot (if network time isn't already available).
- Copies network time back into the DS3231 every 15 minutes, whenever it has internet — so the DS3231 stays current for the next time the Pi boots without a network connection.

No further action is needed under normal use; this only needs re-running if the DS3231 is ever replaced or the Pi's SD card is re-imaged.