
Adafruit PCF8523 RTC Library Documentation

Release 1.0

Philip Moyer

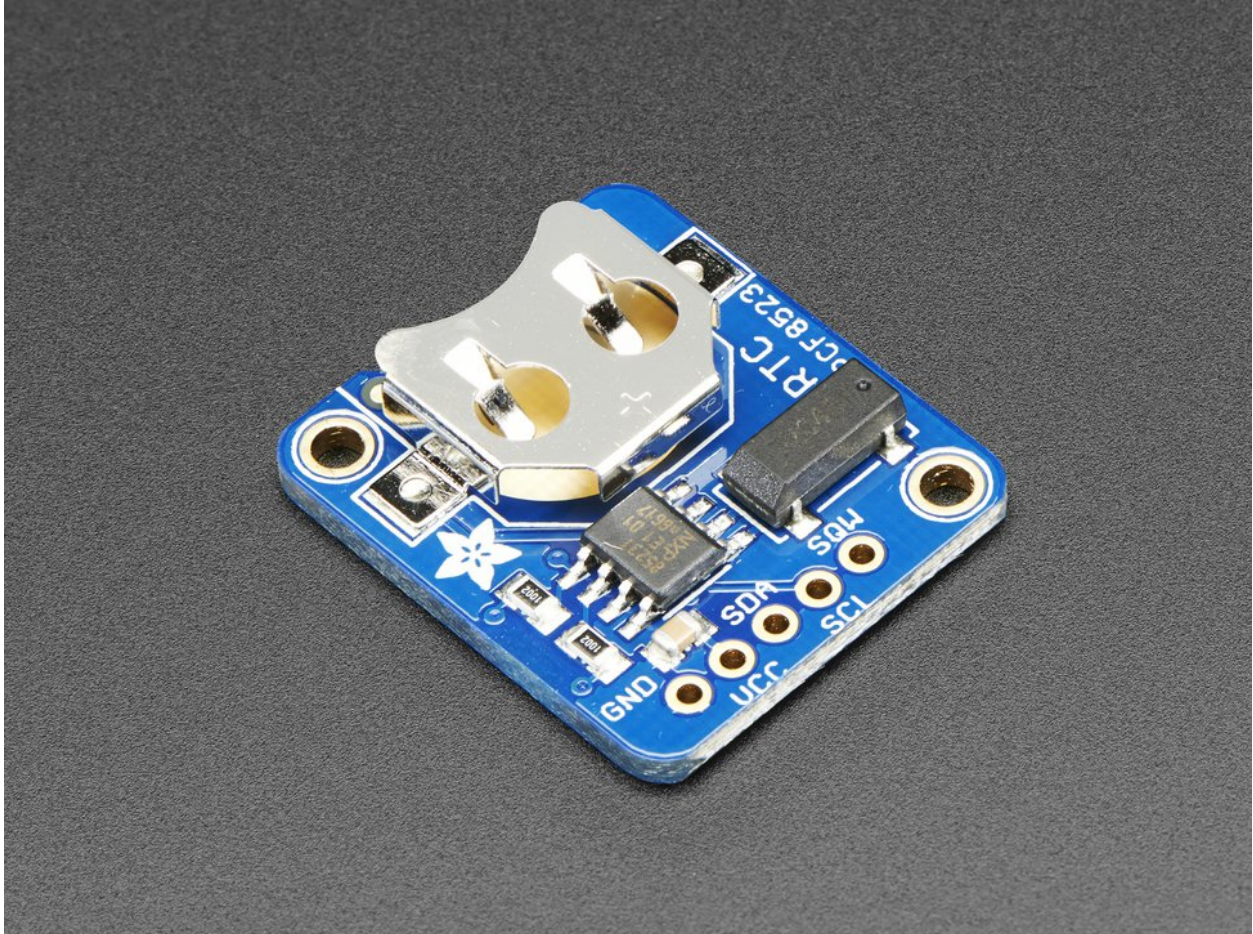
Apr 26, 2021

Contents

1	Dependencies	3
2	Installing from PyPI	5
3	Usage Notes	7
3.1	Basics	7
3.2	Date and time	7
3.3	Alarm	8
4	Contributing	9
5	Documentation	11
6	Table of Contents	13
6.1	Demo	13
6.2	adafruit_pcf8523 - PCF8523 Real Time Clock module	14
6.2.1	Implementation Notes	14
7	Indices and tables	17
	Python Module Index	19
	Index	21

This is a great battery-backed real time clock (RTC) that allows your microcontroller project to keep track of time even if it is reprogrammed, or if the power is lost. Perfect for datalogging, clock-building, time stamping, timers and alarms, etc. Equipped with PCF8523 RTC - it can run from 3.3V or 5V power & logic!

The PCF8523 is simple and inexpensive but not a high precision device. It may lose or gain up to two seconds a day. For a high-precision, temperature compensated alternative, please check out the [DS3231 precision RTC](#). If you need a DS1307 for compatibility reasons, check out our [DS1307 RTC breakout](#).



CHAPTER 1

Dependencies

This driver depends on the [Register](#) and [Bus Device](#) libraries. Please ensure they are also available on the CircuitPython filesystem. This is easily achieved by downloading [a library and driver bundle](#).

CHAPTER 2

Installing from PyPI

On supported GNU/Linux systems like the Raspberry Pi, you can install the driver locally [from PyPI](#). To install for current user:

```
pip3 install adafruit-circuitpython-pcf8523
```

To install system-wide (this may be required in some cases):

```
sudo pip3 install adafruit-circuitpython-pcf8523
```

To install in a virtual environment in your current project:

```
mkdir project-name && cd project-name
python3 -m venv .env
source .env/bin/activate
pip3 install adafruit-circuitpython-pcf8523
```


3.1 Basics

Of course, you must import the library to use it:

```
import time
import adafruit_pcf8523
```

All the Adafruit RTC libraries take an instantiated and active I2C object (from the `board` library) as an argument to their constructor. The way to create an I2C object depends on the board you are using. For boards with labeled SCL and SDA pins, you can:

```
import board
```

Now, to initialize the I2C bus:

```
i2c = board.I2C()
```

Once you have created the I2C interface object, you can use it to instantiate the RTC object:

```
rtc = adafruit_pcf8523.PCF8523(i2c)
```

3.2 Date and time

To set the time, you need to set `datetime` to a `time.struct_time` object:

```
rtc.datetime = time.struct_time((2017, 1, 9, 15, 6, 0, 0, 9, -1))
```

After the RTC is set, you retrieve the time by reading the `datetime` attribute and access the standard attributes of a `struct_time` such as `tm_year`, `tm_hour` and `tm_min`.

```
t = rtc.datetime
print(t)
print(t.tm_hour, t.tm_min)
```

3.3 Alarm

To set the time, you need to set *alarm* to a tuple with a `time.struct_time` object and string representing the frequency such as “hourly”:

```
rtc.alarm = (time.struct_time((2017,1,9,15,6,0,0,9,-1)), "daily")
```

After the RTC is set, you retrieve the alarm status by reading the *alarm_status* attribute. Once True, set it back to False to reset.

```
if rtc.alarm_status:
    print("wake up!")
    rtc.alarm_status = False
```

CHAPTER 4

Contributing

Contributions are welcome! Please read our [Code of Conduct](#) before contributing to help this project stay welcoming.

CHAPTER 5

Documentation

For information on building library documentation, please check out [this guide](#).

6.1 Demo

Listing 1: examples/pcf8523_simpletest.py

```
1  # SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
2  # SPDX-License-Identifier: MIT
3
4  # Simple demo of reading and writing the time for the PCF8523 real-time clock.
5  # Change the if False to if True below to set the time, otherwise it will just
6  # print the current date and time every second. Notice also comments to adjust
7  # for working with hardware vs. software I2C.
8
9  import time
10 import board
11 import adafruit_pcf8523
12
13 i2c = board.I2C()
14 rtc = adafruit_pcf8523.PCF8523(i2c)
15
16 # Lookup table for names of days (nicer printing).
17 days = ("Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday")
18
19
20 # pylint: disable-msg=using-constant-test
21 if False: # change to True if you want to set the time!
22     # year, mon, date, hour, min, sec, wday, yday, isdst
23     t = time.struct_time((2017, 10, 29, 10, 31, 0, 0, -1, -1))
24     # you must set year, mon, date, hour, min, sec and weekday
25     # yearday is not supported, isdst can be set but we don't do anything with it at
26     ↪ this time
27     print("Setting time to:", t) # uncomment for debugging
28     rtc.datetime = t
29     print()
```

(continues on next page)

(continued from previous page)

```

29 # pylint: enable-msg=using-constant-test
30
31 # Main loop:
32 while True:
33     t = rtc.datetime
34     # print(t)      # uncomment for debugging
35     print(
36         "The date is {} {}/{}/{}{}".format(
37             days[int(t.tm_wday)], t.tm_mday, t.tm_mon, t.tm_year
38         )
39     )
40     print("The time is {}:02:02".format(t.tm_hour, t.tm_min, t.tm_sec))
41     time.sleep(1)  # wait a second

```

6.2 adafruit_pcf8523 - PCF8523 Real Time Clock module

This library supports the use of the PCF8523-based RTC in CircuitPython. It contains a base RTC class used by all Adafruit RTC libraries. This base class is inherited by the chip-specific subclasses.

Functions are included for reading and writing registers and manipulating datetime objects.

Author(s): Philip R. Moyer and Radomir Dopieralski for Adafruit Industries. Date: November 2016 Affiliation: Adafruit Industries

6.2.1 Implementation Notes

Hardware:

- Adafruit Adalogger FeatherWing - RTC + SD Add-on (Product ID: 2922)
- Adafruit PCF8523 RTC breakout (Product ID: 3295)

Software and Dependencies:

- Adafruit CircuitPython firmware for the supported boards: <https://circuitpython.org/downloads>
- Adafruit's Register library: https://github.com/adafruit/Adafruit_CircuitPython_Register
- Adafruit's Bus Device library: https://github.com/adafruit/Adafruit_CircuitPython_BusDevice

Notes:

1. Milliseconds are not supported by this RTC.
2. Datasheet: http://cache.nxp.com/documents/data_sheet/PCF8523.pdf

class `adafruit_pcf8523.PCF8523` (*i2c_bus*)
Interface to the PCF8523 RTC.

Parameters `i2c_bus` (*I2C*) – The I2C bus the device is connected to

Quickstart: Importing and using the device

Here is an example of using the `PCF8523` class. First you will need to import the libraries to use the sensor

```
import time
import board
import adafruit_pcf8523
```

Once this is done you can define your `board.I2C` object and define your sensor object

```
i2c = board.I2C() # uses board.SCL and board.SDA
rtc = adafruit_pcf8523.PCF8523(i2c)
```

Now you can give the current time to the device.

```
t = time.struct_time((2017, 10, 29, 15, 14, 15, 0, -1, -1))
rtc.datetime = t
```

You can access the current time accessing the `datetime` attribute.

```
current_time = rtc.datetime
```

alarm

Alarm time for the first alarm.

alarm_interrupt

True if the interrupt pin will output when alarm is alarming.

alarm_status

True if alarm is alarming. Set to False to reset.

battery_low

True if the battery is low and should be replaced.

calibration

Calibration offset to apply, from -64 to +63. See the PCF8523 datasheet figure 18 for the offset calibration calculation workflow.

calibration_schedule_per_minute

False to apply the calibration offset every 2 hours (1 LSB = 4.340ppm); True to offset every minute (1 LSB = 4.069ppm). The default, False, consumes less power. See datasheet figures 28-31 for details.

datetime

Gets the current date and time or sets the current date and time then starts the clock.

datetime_register

Current date and time.

high_capacitance

True for high oscillator capacitance (12.5pF), otherwise lower (7pF)

lost_power

True if the device has lost power since the time was set.

power_management

Power management state that dictates battery switchover, power sources and low battery detection. Defaults to BATTERY_SWITCHOVER_OFF (0b000).

CHAPTER 7

Indices and tables

- `genindex`
- `modindex`
- `search`

a

adafruit_pcf8523, [14](#)

A

`adafruit_pcf8523` (*module*), 14
`alarm` (*adafruit_pcf8523.PCF8523 attribute*), 15
`alarm_interrupt` (*adafruit_pcf8523.PCF8523 attribute*), 15
`alarm_status` (*adafruit_pcf8523.PCF8523 attribute*), 15

B

`battery_low` (*adafruit_pcf8523.PCF8523 attribute*), 15

C

`calibration` (*adafruit_pcf8523.PCF8523 attribute*), 15
`calibration_schedule_per_minute` (*adafruit_pcf8523.PCF8523 attribute*), 15

D

`datetime` (*adafruit_pcf8523.PCF8523 attribute*), 15
`datetime_register` (*adafruit_pcf8523.PCF8523 attribute*), 15

H

`high_capacitance` (*adafruit_pcf8523.PCF8523 attribute*), 15

L

`lost_power` (*adafruit_pcf8523.PCF8523 attribute*), 15

P

`PCF8523` (*class in adafruit_pcf8523*), 14
`power_management` (*adafruit_pcf8523.PCF8523 attribute*), 15