
Adafruit DS1307 Library Documentation

Release 1.0

Phillip Moyer

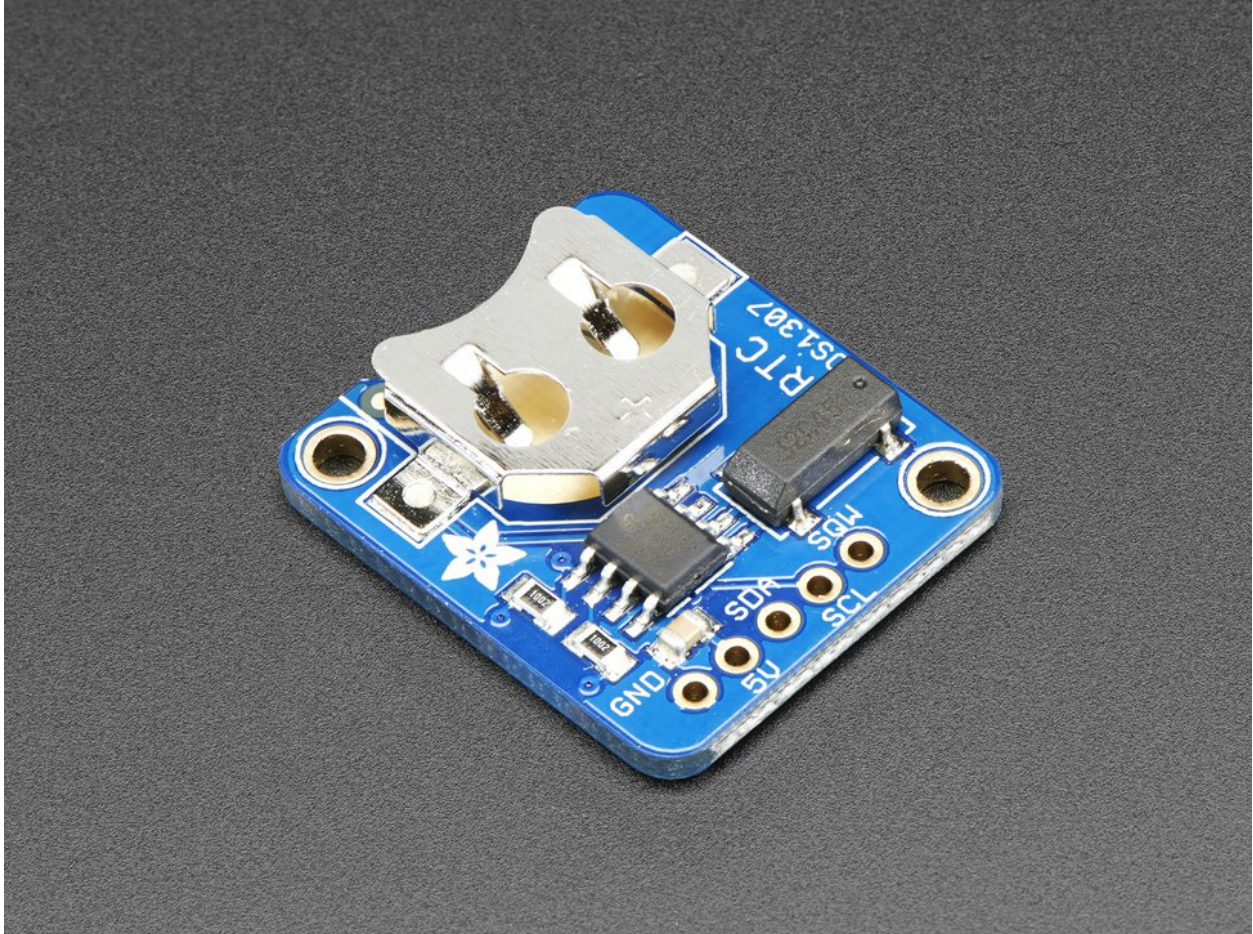
Feb 10, 2021

Contents

1	Dependencies	3
2	Installing from PyPI	5
3	Usage Notes	7
4	Contributing	9
5	Documentation	11
6	Table of Contents	13
6.1	Simple test	13
6.2	adafruit_ds1307 - DS1307 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. The DS1307 is the most popular RTC - but it requires 5V power to work.

The DS1307 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 do not need a DS1307, or you need a 3.3V-power/logic capable RTC please check out our affordable [PCF8523 RTC breakout](#).



CHAPTER 1

Dependencies

This driver depends on:

- [Adafruit CircuitPython](#)
- [Bus Device](#)
- [Register](#)

Please ensure all dependencies are available on the CircuitPython filesystem. This is easily achieved by downloading the [Adafruit 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-ds1307
```

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

```
sudo pip3 install adafruit-circuitpython-ds1307
```

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-ds1307
```


CHAPTER 3

Usage Notes

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

```
import busio
import adafruit_ds1307
import time
```

All the Adafruit RTC libraries take an instantiated and active I2C object (from the `busio` 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:

```
from board import *
```

You can also use pins defined by the onboard microcontroller through the `microcontroller.pin` module.

Now, to initialize the I2C bus:

```
myI2C = busio.I2C(SCL, SDA)
```

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

```
rtc = adafruit_ds1307.DS1307(myI2C)
```

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)
```


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 Simple test

Ensure your device works with this simple test.

Listing 1: examples/ds1307_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 DS1307 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
12 # For hardware I2C (M0 boards) use this line:
13 import busio as io
14
15 # Or for software I2C (ESP8266) use this line instead:
16 # import bitbangio as io
17
18 import adafruit_ds1307
19
20 # Change to the appropriate I2C clock & data pins here!
21 i2c_bus = io.I2C(board.SCL, board.SDA)
22
23 # Create the RTC instance:
24 rtc = adafruit_ds1307.DS1307(i2c_bus)
25
26 # Lookup table for names of days (nicer printing).
27 days = ("Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday")
```

(continues on next page)

(continued from previous page)

```

28
29
30 # pylint: disable-msg=using-constant-test
31 if False: # change to True if you want to set the time!
32     # year, mon, date, hour, min, sec, wday, yday, isdst
33     t = time.struct_time((2017, 10, 29, 15, 14, 15, 0, -1, -1))
34     # you must set year, mon, date, hour, min, sec and weekday
35     # yearday is not supported, isdst can be set but we don't do anything with it at
    ↪ this time
36     print("Setting time to:", t) # uncomment for debugging
37     rtc.datetime = t
38     print()
39 # pylint: enable-msg=using-constant-test
40
41 # Main loop:
42 while True:
43     t = rtc.datetime
44     # print(t) # uncomment for debugging
45     print(
46         "The date is {} {}/{}/{}{}".format(
47             days[int(t.tm_wday)], t.tm_mday, t.tm_mon, t.tm_year
48         )
49     )
50     print("The time is {}:02:02".format(t.tm_hour, t.tm_min, t.tm_sec))
51     time.sleep(1) # wait a second

```

6.2 adafruit_ds1307 - DS1307 Real Time Clock module

CircuitPython library to support DS1307 Real Time Clock (RTC).

This library supports the use of the DS1307-based RTC in CircuitPython.

Beware that most CircuitPython compatible hardware are 3.3v logic level! Make sure that the input pin is 5v tolerant.

- Author(s): Philip R. Moyer and Radomir Dopieralski for Adafruit Industries

6.2.1 Implementation Notes

Hardware:

- Adafruit [DS1307 RTC breakout](#) (Product ID: 3296)

Software and Dependencies:

- **Adafruit CircuitPython firmware (0.8.0+) for the ESP8622 and M0-based boards:** <https://github.com/adafruit/circuitpython/releases>
- 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. Alarms and timers are not supported by this RTC.
3. Datasheet: <https://datasheets.maximintegrated.com/en/ds/DS1307.pdf>

class adafruit_ds1307.**DS1307** (*i2c_bus*)

Interface to the DS1307 RTC.

datetime

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

datetime_register

Current date and time.

disable_oscillator

True if the oscillator is disabled.

CHAPTER 7

Indices and tables

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

a

adafruit_ds1307, [14](#)

A

`adafruit_ds1307` (*module*), [14](#)

D

`datetime` (*adafruit_ds1307.DS1307 attribute*), [15](#)

`datetime_register` (*adafruit_ds1307.DS1307 attribute*), [15](#)

`disable_oscillator` (*adafruit_ds1307.DS1307 attribute*), [15](#)

`DS1307` (*class in adafruit_ds1307*), [14](#)