

---

# **AdafruitTLC5947 Library Documentation**

***Release 1.0***

**Tony DiCola**

**Nov 30, 2018**



---

## Contents

---

<b>1</b>	<b>Dependencies</b>	<b>3</b>
<b>2</b>	<b>Usage Example</b>	<b>5</b>
<b>3</b>	<b>Contributing</b>	<b>7</b>
<b>4</b>	<b>Building locally</b>	<b>9</b>
4.1	Sphinx documentation . . . . .	9
<b>5</b>	<b>Table of Contents</b>	<b>11</b>
5.1	Simple test . . . . .	11
5.2	adafruit_tlc5947 . . . . .	12
5.2.1	Implementation Notes . . . . .	12
<b>6</b>	<b>Indices and tables</b>	<b>15</b>
	<b>Python Module Index</b>	<b>17</b>



CircuitPython module for the TLC5947 12-bit 24 channel LED PWM driver.



# CHAPTER 1

---

## Dependencies

---

This driver depends on:

- [Adafruit CircuitPython](#)

Please ensure all dependencies are available on the CircuitPython filesystem. This is easily achieved by downloading the [Adafruit library and driver bundle](#).



## CHAPTER 2

---

### Usage Example

---

See `examples/tlc5947_simpletest.py` for a demo of the usage.



## CHAPTER 3

---

### Contributing

---

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



## CHAPTER 4

---

### Building locally

---

To build this library locally you'll need to install the `circuitpython-build-tools` package.

```
python3 -m venv .env
source .env/bin/activate
pip install circuitpython-build-tools
```

Once installed, make sure you are in the virtual environment:

```
source .env/bin/activate
```

Then run the build:

```
circuitpython-build-bundles --filename_prefix adafruit-circuitpython-tlc5947 --
↳library_location .
```

### 4.1 Sphinx documentation

Sphinx is used to build the documentation based on rST files and comments in the code. First, install dependencies (feel free to reuse the virtual environment from above):

```
python3 -m venv .env
source .env/bin/activate
pip install Sphinx sphinx-rtd-theme
```

Now, once you have the virtual environment activated:

```
cd docs
sphinx-build -E -W -b html . _build/html
```

This will output the documentation to `docs/_build/html`. Open the `index.html` in your browser to view them. It will also (due to `-W`) error out on any warning like Travis will. This is a good way to locally verify it will pass.



## 5.1 Simple test

Ensure your device works with this simple test.

Listing 1: examples/tlc5947\_simpletest.py

```
1  # Simple demo of controlling the TLC5947 12-bit 24-channel PWM controller.
2  # Will update channel values to different PWM duty cycles.
3  # Author: Tony DiCola
4
5  import board
6  import busio
7  import digitalio
8
9  import adafruit_tlc5947
10
11 # Define pins connected to the TLC5947
12 SCK = board.SCK
13 MOSI = board.MOSI
14 LATCH = digitalio.DigitalInOut(board.D5)
15
16 # Initialize SPI bus.
17 spi = busio.SPI(clock=SCK, MOSI=MOSI)
18
19 # Initialize TLC5947
20 tlc5947 = adafruit_tlc5947.TLC5947(spi, LATCH)
21 # You can optionally disable auto_write which allows you to control when
22 # channel state is written to the chip. Normally auto_write is true and
23 # will automatically write out changes as soon as they happen to a channel, but
24 # if you need more control or atomic updates of multiple channels then disable
25 # and manually call write as shown below.
26 # tlc5947 = adafruit_tlc5947.TLC5947(spi, LATCH, auto_write=False)
27
```

(continues on next page)

(continued from previous page)

```

28 # There are two ways to channel channel PWM values. The first is by getting
29 # a PWMOut object that acts like the built-in PWMOut and can be used anywhere
30 # it is used in your code. Change the duty_cycle property to a 16-bit value
31 # (note this is NOT the 12-bit value supported by the chip natively) and the
32 # PWM channel will be updated.
33
34 # With an RGB LED hooked up to pins 0, 1, and 2, cycle the red, green, and
35 # blue pins up and down:
36
37 red = tlc5947.create_pwm_out(0)
38 green = tlc5947.create_pwm_out(1)
39 blue = tlc5947.create_pwm_out(2)
40
41 step = 10
42 start_pwm = 0
43 end_pwm = 32767 # 50% (32767, or half of the maximum 65535):
44
45 while True:
46     for pin in (red, green, blue):
47         # Brighten:
48         print("Brightening LED")
49         for pwm in range(start_pwm, end_pwm, step):
50             pin.duty_cycle = pwm
51
52         # Dim:
53         print("Dimming LED")
54         for pwm in range(end_pwm, start_pwm, 0 - step):
55             pin.duty_cycle = pwm
56
57 # Note if auto_write was disabled you need to call write on the parent to
58 # make sure the value is written (this is not common, if disabling auto_write
59 # you probably want to use the direct 12-bit raw access instead shown below).
60 # tlc5947.write()
61
62 # The other way to read and write channels is directly with each channel 12-bit
63 # value and an item accessor syntax. Index into the TLC5947 with the channel
64 # number (0-23) and get or set its 12-bit value (0-4095).
65 # For example set channel 1 to 50% duty cycle.
66 # tlc5947[1] = 2048
67 # Again be sure to call write if you disabled auto_write.
68 #tlc5947.write()

```

## 5.2 adafruit\_tlc5947

CircuitPython module for the TLC5947 12-bit 24 channel LED PWM driver. See examples/simpletest.py for a demo of the usage.

- Author(s): Tony DiCola

### 5.2.1 Implementation Notes

#### Hardware:

- Adafruit 24-Channel 12-bit PWM LED Driver - SPI Interface - TLC5947 (Product ID: 1429)

**Software and Dependencies:**

- Adafruit CircuitPython firmware for the ESP8622 and M0-based boards: <https://github.com/adafruit/circuitpython/releases>

**class** `adafruit_tlc5947.TLC5947` (*spi, latch, \*, auto\_write=True*)

TLC5947 12-bit 24 channel LED PWM driver. Create an instance of this by passing in at least the following parameters:

**Parameters**

- **spi** – The SPI bus connected to the chip (only the SCK and MOSI lines are used, there is no MISO/input).
- **latch** – A DigitalInOut instance connected to the chip's latch line.

Optionally you can specify:

**Parameters** **auto\_write** – This is a boolean that defaults to True and will automatically write out all the channel values to the chip as soon as a single one is updated. If you set to false to disable then you MUST call write after every channel update or when you deem necessary to update the chip state.

**class** `PWMOut` (*tlc5947, channel*)

Internal PWMOut class that mimics the behavior of CircuitPython's PWMOut class but is associated with a channel on the TLC5947. You can get and set the instance's `duty_cycle` property as a 16-bit PWM value (note there will be quantization errors as the TLC5947 is a 12-bit PWM chip, instead use the TLC5947 class item accessor notation for direct 12-bit raw PWM channel access). Note you cannot change the frequency as it is fixed by the TLC5947 to ~2.4-5.6 mhz.

**duty\_cycle**

Get and set the 16-bit PWM duty cycle value for this channel.

**frequency**

Frequency of the PWM channel, note you cannot change this and cannot read its exact value (it varies from 2.4-5.6 mhz, see the TLC5947 datasheet).

**create\_pwm\_out** (*channel*)

Create an instance of a PWMOut-like class that mimics the built-in CircuitPython PWMOut class but is associated with the TLC5947 channel that is specified. This PWMOut class has a `duty_cycle` property which you can read and write with a 16-bit value to control the channel. Note there will be quantization error as the chip only supports 12-bit PWM, if this is problematic use the item accessor approach to update the raw 12-bit channel values.

**write** ()

Write out the current channel PWM values to the chip. This is only necessary to call if you disabled `auto_write` in the initializer, otherwise write is automatically called on any channel update.



## CHAPTER 6

---

### Indices and tables

---

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



### a

adafruit\_tlc5947, [12](#)



### A

`adafruit_tlc5947` (module), [12](#)

### C

`create_pwm_out()` (`adafruit_tlc5947.TLC5947` method), [13](#)

### D

`duty_cycle` (`adafruit_tlc5947.TLC5947.PWMOut` attribute), [13](#)

### F

`frequency` (`adafruit_tlc5947.TLC5947.PWMOut` attribute), [13](#)

### T

`TLC5947` (class in `adafruit_tlc5947`), [13](#)

`TLC5947.PWMOut` (class in `adafruit_tlc5947`), [13](#)

### W

`write()` (`adafruit_tlc5947.TLC5947` method), [13](#)