

---

# **AdafruitMCP4725 Library Documentation**

*Release 1.0*

**Tony DiCola**

**Dec 21, 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_mcp4725 - MCP4725 digital to analog converter . . . . .	12
5.2.1	Implementation Notes . . . . .	12
<b>6</b>	<b>Indices and tables</b>	<b>13</b>
	<b>Python Module Index</b>	<b>15</b>



CircuitPython module for the MCP4725 digital to analog converter.



# 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/max4725_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-mcp4725 --
↳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/mcp4725\_simpletest.py

```
1  # Simple demo of setting the DAC value up and down through its entire range
2  # of values.
3  # Author: Tony DiCola
4  import board
5  import busio
6
7  import adafruit_mcp4725
8
9
10 # Initialize I2C bus.
11 i2c = busio.I2C(board.SCL, board.SDA)
12
13 # Initialize MCP4725.
14 dac = adafruit_mcp4725.MCP4725(i2c)
15 # Optionally you can specify a different address if you override the A0 pin.
16 #amp = adafruit_max9744.MAX9744(i2c, address=0x63)
17
18 # There are a three ways to set the DAC output, you can use any of these:
19 dac.value = 65535 # Use the value property with a 16-bit number just like
20                  # the AnalogOut class. Note the MCP4725 is only a 12-bit
21                  # DAC so quantization errors will occur. The range of
22                  # values is 0 (minimum/ground) to 65535 (maximum/Vout).
23
24 dac.raw_value = 4095 # Use the raw_value property to directly read and write
25                     # the 12-bit DAC value. The range of values is
26                     # 0 (minimum/ground) to 4095 (maximum/Vout).
27
```

(continues on next page)

(continued from previous page)

```

28 dac.normalized_value = 1.0 # Use the normalized_value property to set the
29                             # output with a floating point value in the range
30                             # 0 to 1.0 where 0 is minimum/ground and 1.0 is
31                             # maximum/Vout.
32
33 # Main loop will go up and down through the range of DAC values forever.
34 while True:
35     # Go up the 12-bit raw range.
36     print('Going up 0-3.3V...')
37     for i in range(4095):
38         dac.raw_value = i
39     # Go back down the 12-bit raw range.
40     print('Going down 3.3-0V...')
41     for i in range(4095, -1, -1):
42         dac.raw_value = i

```

## 5.2 adafruit\_mcp4725 - MCP4725 digital to analog converter

CircuitPython module for the MCP4725 digital to analog converter. See `examples/mcp4725_simpletest.py` for a demo of the usage.

- Author(s): Tony DiCola

### 5.2.1 Implementation Notes

#### Hardware:

- Adafruit MCP4725 Breakout Board - 12-Bit DAC w/I2C Interface (Product ID: 935)

#### Software and Dependencies:

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

**class** `adafruit_mcp4725.MCP4725` (*i2c*, \*, *address=98*)

MCP4725 12-bit digital to analog converter. This class has a similar interface as the CircuitPython `AnalogOut` class and can be used in place of that module.

#### Parameters

- **i2c** (*I2C*) – The I2C bus.
- **address** (*int*) – The address of the device if set differently from the default.

#### **normalized\_value**

The DAC value as a floating point number in the range 0.0 to 1.0.

#### **raw\_value**

The DAC value as a 12-bit unsigned value. This is the the true resolution of the DAC and will never perform scaling or run into quantization error.

#### **value**

The DAC value as a 16-bit unsigned value compatible with the `AnalogOut` class.

Note that the MCP4725 is still just a 12-bit device so quantization will occur. If you'd like to instead deal with the raw 12-bit value use the `raw_value` property, or the `normalized_value` property to deal with a 0...1 float value.

## CHAPTER 6

---

### Indices and tables

---

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



**a**

`adafruit_mcp4725`, 12



## A

`adafruit_mcp4725` (module), 12

## M

`MCP4725` (class in `adafruit_mcp4725`), 12

## N

`normalized_value` (`adafruit_mcp4725.MCP4725` attribute), 12

## R

`raw_value` (`adafruit_mcp4725.MCP4725` attribute), 12

## V

`value` (`adafruit_mcp4725.MCP4725` attribute), 12