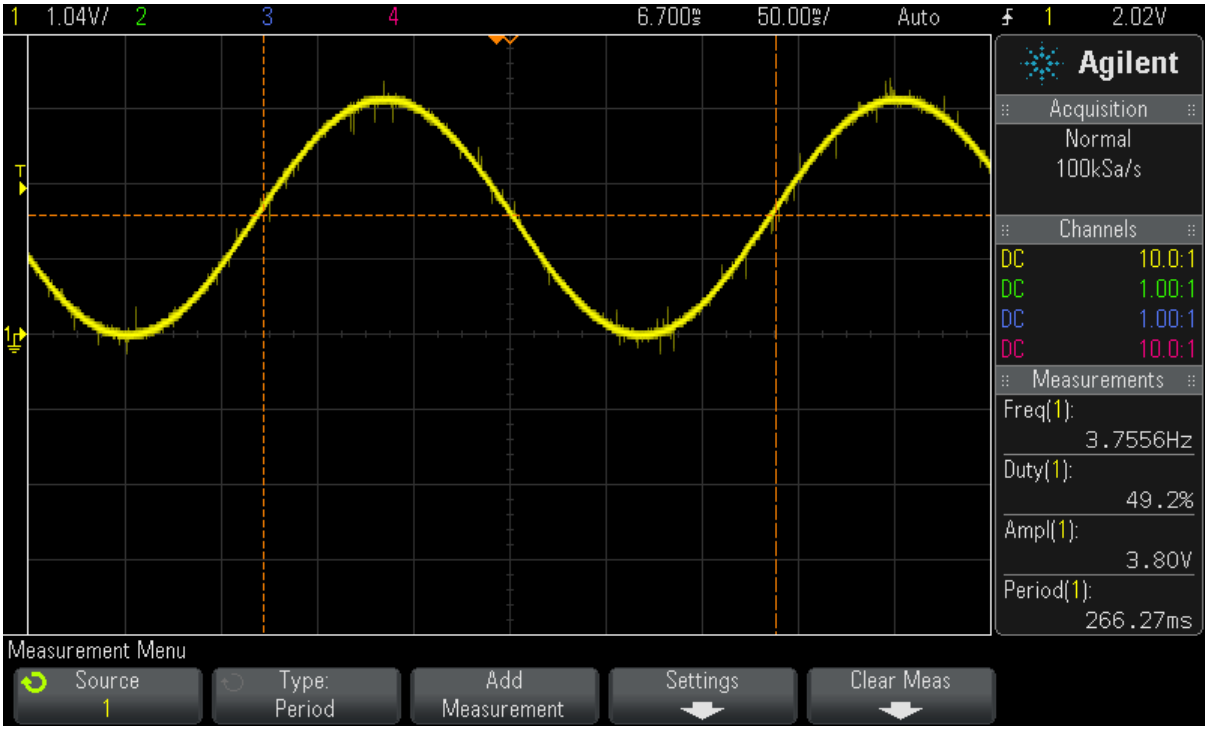
MCP4725 12-Bit DAC with Raspberry Pi

Created by Kevin Townsend



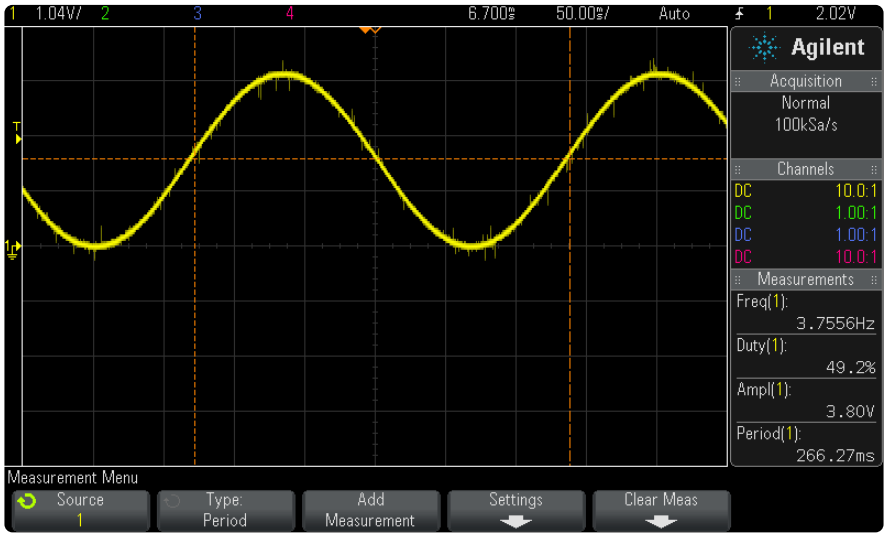
https://learn.adafruit.com/mcp4725-12-bit-dac-with-raspberry-pi

Last updated on 2023-08-29 02:10:57 PM EDT

Table of Contents

Overview

The examples in this guide are no longer supported. Please check out the MCP4725 12-Bit DAC Tutorial guide for CircuitPython and Python usage: https://learn.adafruit.com/mcp4725-12-bit-dac-tutorial



Already mastered Analog Inputs with the Pi (), and looking for a new challenge? How about: How can I generate an analog output on the Pi?!

There are several ways you can accomplish this, but one of the easiest and most flexible is to use a dedicated IC called a Digital to Analog Convertor (or DAC). A DAC allows you to specify a numeric value (0..255 for an 8-bit DAC, 0..4095 for a 12-bit DAC, etc.), and the IC will output a voltage based on the supply voltage, and relative to that numeric value. For example, using a 12-bit DAC like the MCP4725 (http://adafru.it/935) we'll be using here, setting the value to 2048 on a 3.3V system will results in ~1.65V output on the DAC.

This guide will show you everything you need to know to be able to generate precise analog outputs using your Pi and the MCP4725 12-Bit I2C DAC, from connecting everything up, to how to use our easy Python library.

What You'll Need

The following products are used in this tutorial:

- Adafruit MCP4725 12-Bit DAC (http://adafru.it/935)
- Pi Cobbler (http://adafru.it/914)

Configuring Your Pi for I2C

The examples in this guide are no longer supported. Please check out the MCP4725 12-Bit DAC Tutorial guide for CircuitPython and Python usage: https://learn.adafruit.com/mcp4725-12-bit-dac-tutorial

Before you can get started with I2C on the Pi, you'll need to run through a couple quick steps from the console.

If you are running Occidentalis and are familiar with Terminal commands, then the description below will be sufficient.

If not, then to learn more about how to setup I2C with either Raspbian or Occidentalis, then take a minor diversion to this Adafruit Tutorial: http://learn.adafruit.com/adafruits-raspberry-pi-lesson-4-gpio-setup/configuring-i2c ()

When you are happy to continue enter the following commands to add SMBus support (which includes I2C) to Python:

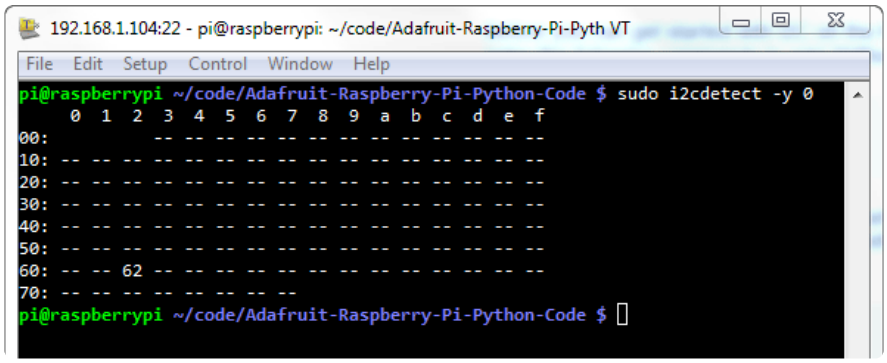
```
sudo apt-get install python-smbus
sudo apt-get install i2c-tools
```

i2c-tools isn't strictly required, but it's a useful package since you can use it to scan for any I2C or SMBus devices connected to your board. If you know something is connected, but you don't know it's 7-bit I2C address, this library has a great little tool to help you find it:

```
sudo i2cdetect -y 0
```

This will search /dev/i2c-0 for all address, and if an MCP4725 DAC breakout is properly connected and it's set to it's default address it should show up at 0x62.

If you are using a 512MB Raspberry Pi version 2, you'll want to use /dev/i2c-1 by running sudo i2cdetect -y 1 as the i2c port # changed from #0 to #1

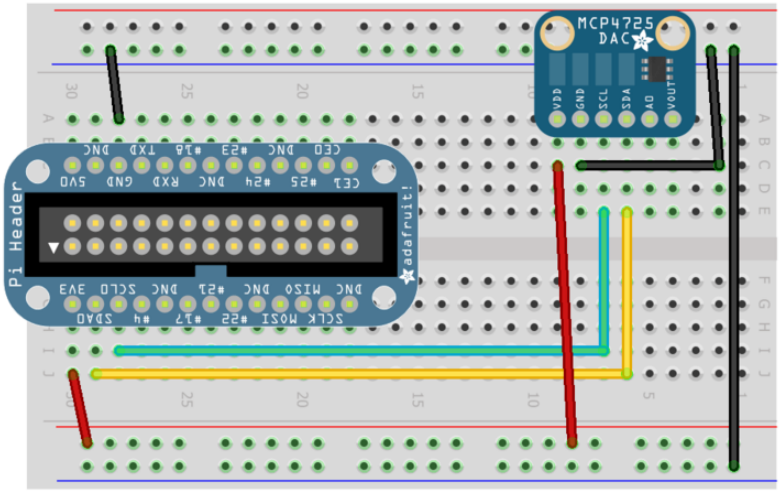


Once both of these packages have been installed, you have everything you need to get started accessing I2C and SMBus devices in Python.

Hooking It Up

The examples in this guide are no longer supported. Please check out the MCP4725 12-Bit DAC Tutorial guide for CircuitPython and Python usage: https://learn.adafruit.com/mcp4725-12-bit-dac-tutorial

The easiest way to hook the MCP4725 up to your Pi is with the Adafruit Pi Cobbler, as seen in the wiring diagram below:



The output voltage will be visible on VOUT, which you should connect to your target device, or to a multimeter (http://adafru.it/71) or oscilloscope (http://adafru.it/681) if you just want to see the results.

Using the Adafruit Library

The examples in this guide are no longer supported. Please check out the MCP4725 12-Bit DAC Tutorial guide for CircuitPython and Python usage: https://learn.adafruit.com/mcp4725-12-bit-dac-tutorial

The Python code for Adafruit's MCP4725 breakout on the Pi is available on Github at https://github.com/adafruit/Adafruit_Python_MCP4725 ()

This code should be a good starting point to understanding how you can access SMBus/I2C devices with your Pi, and getting things moving with your DAC breakout.

Downloading the Code from Github

The easiest way to get the code onto your Pi is to hook up an Ethernet cable, and clone it directly using 'git', which is installed by default on most distros. Simply run the following commands from an appropriate location (ex. "/home/pi"):

```
sudo apt-get install git build-essential python-dev
cd ~
git clone https://github.com/adafruit/Adafruit_Python_MCP4725.git
cd Adafruit_Python_MCP4725
sudo python setup.py install
```

Testing the Library

Once the code has be downloaded to an appropriate folder, and you have your MCP4725 breakout properly connected, you can test it out with the following command (the driver includes a simple demo program in the examples subfolder):

```
cd examples
sudo python simpletest.py
```

This will alternate the voltage of the output from VDDto VDD/2 to 0 and back. You can use a multimeter or oscilloscope to check the voltage on the output and see it changing between values.

Writing Your Own Code

The MCP4725 library is quite straight forward, and includes a single function: .set_voltage(voltage), with voltage having a value between 0 and 4095 (corresponding to a 12-

bit value). If you're new to Python, though, have a look at the simpletest.py example to see how you can instantiate an instance of the base DAC class, and change the voltage.

From the command-line you can edit this file in nano (vi is also included if you prefer that) with the following command:

```
sudo nano simpletest.py
```

This will result in something similar to the following, though we've scrolled down a few pages to show the actual instantiation code, etc.:

```
# Simple demo of setting the output voltage of the MCP4725 DAC.
# Will alternate setting 0V, 1/2VDD, and VDD each second.
# Author: Tony DiCola
# License: Public Domain
import time

# Import the MCP4725 module.
import Adafruit_MCP4725

# Create a DAC instance.
dac = Adafruit_MCP4725.MCP4725()

# Note you can change the I2C address from its default (0x62), and/or the I2C
# bus by passing in these optional parameters:
#dac = Adafruit_MCP4725.MCP4725(address=0x49, busnum=1)

# Loop forever alternating through different voltage outputs.
print('Press Ctrl-C to quit...')
while True:
    print('Setting voltage to 0!')
    dac.set_voltage(0)
    time.sleep(2.0)
    print('Setting voltage to 1/2 Vdd!')
    dac.set_voltage(2048)  # 2048 = half of 4096
    time.sleep(2.0)
    print('Setting voltage to Vdd!')
    dac.set_voltage(4096, True)
    time.sleep(2.0)
```