Web Based Function Generator

Introduction

This app demonstrates the use of Python, Javascript, PHP, and HTML to implement a web based function generator. A function generator outputs an electrical signal with various standard, user selectable waveforms such as sine wave, triangle wave, or square wave. Usually the output voltage and frequency is also user selectable. This app also demonstrates how to hookup an I2C serial i/o device to the Raspberry Pi's GPIO.

Referring to figure 1, the user interacts with the Raspberry Pi web service via a browser running on computer or mobile unit. Whenever the user turns on the function generator, or changes one of the controls on the web page, a PHP scripts executes a Python script as a shell command running as a process in the background. The Python script sends real-time digital waveform data to a digital to analog converter (DAC) device connected to its GPIO. The script calculates the data based on parameters supplied with the scripts gets executed as a shell command.

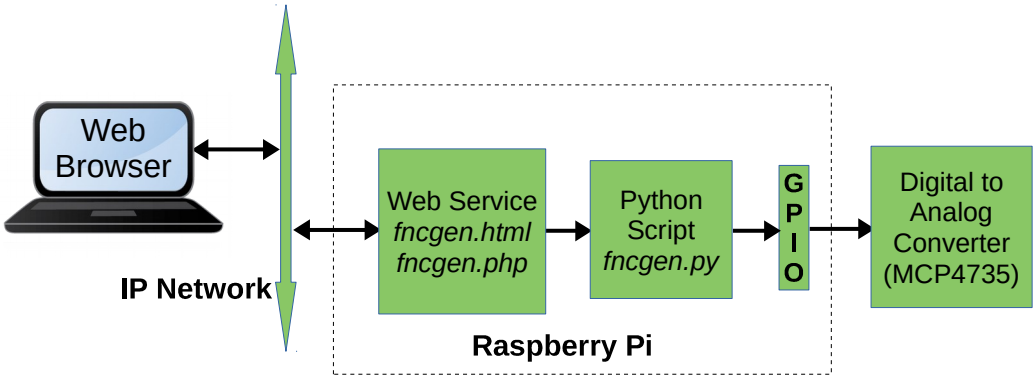

Figure 1. Block diagram showing logical relationship between web client, Raspberry Pi, and the digital to analog converter.

Hardware Hookup

Referring to figure 2, the MCP4735 breakout board VDD is connected to 3.3 Volts via header pin 1. The breakout GND is connected to ground via header pin 6. Breakout serial clock (SCL) is connected to GPIO SCL, which is GPIO pin 3, or pin 5 on the header; and breakout serial data (SDA) is connected to GPIO SDA, which is GPIO pin 2, or pin 3 on the header. If you also have the oscilloscope app hooked up, you can use the function generator as an input to the oscilloscope. The dotted line from breakout OUT shows where the input to the oscilloscope ADC would be connected.

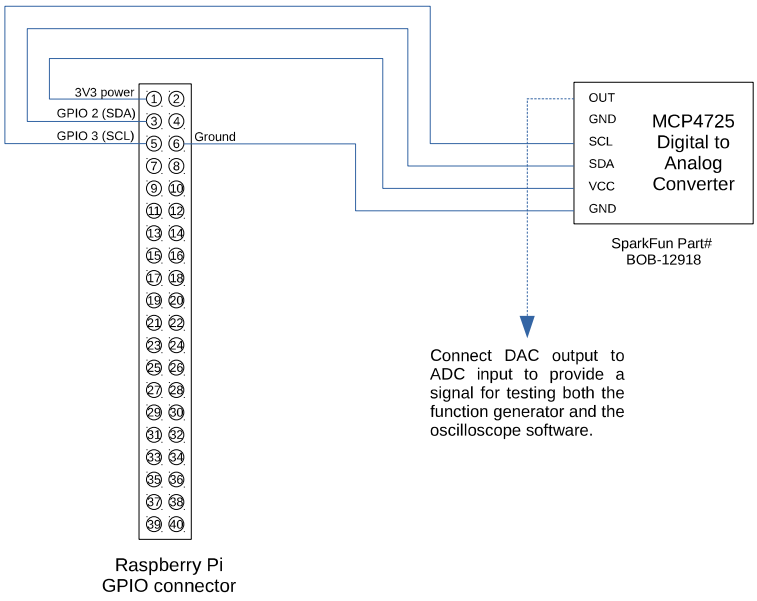

Figure 2. Schematic diagram of the digital to analog converter connected to the Raspberry Pi GPIO.

Software Description

The Python script *fncgen.py* is at the heart of the function generator app. The script can produce several different waveforms: sine wave, square wave, triangle wave, and sawtooth wave. Additionally the script supports user selectable frequency, amplitude, and duty cycle (square wave). The function generator can be run from the command line in a terminal. The command syntax is

```
fncgen.py [-kv] [-w dc|sin|sqr|tri|saw] [-a AMPLITUDE] [-f FREQUENCY] [-d DUTYCYCLE]
```

where

-w one of five waveform types: dc, sine, square, triangle, and sawtooth;
-a amplitude, a number between 0 and 3.3 Volts;
-f frequency, a number between 0 and 100 Hertz;
-d duty cycle, a number between 0% and 100%;
-k kill all running instances of *fncgen.py*, including this instance
-v run in verbose debug mode

When started up, the script first kills any previously launched instances of itself. Once the script has been started in the background, it then continues to run in the background until killed. As long as the script continues to run, the function generator produces an output. Executing the script with the -k option kills this previously running instance and then exits the current instance. That is, to stop the function generator when running in the background, execute the following on the command line

```
fncgen.py -k
```

When the user opens the HTML document *fncgen.html* in the browser, Javascript submits a form to be acted upon by the PHP script *fncgen.php*. Whenever the user changes a control on the function generator web page, the web page submits the run state and waveform parameters to the PHP script. The PHP script executes the

Python script *fncgen.py* in a shell with the appropriate arguments depending on the run state submitted by *fncgen.html*.

The above procedure may be summarized in the following steps

1. Open the function generator web page *fncgen.html* in the browser. Javascript submits the *off* runstate to the PHP script *fncgen.php*.
2. The PHP script executes the Python script *fncgen.py* as a shell command with the -k option to kill any perviously running instances of itself.
3. If the user selects the ON button, Javascript will submit the ON run state along with all the necessary waveform parameters: waveform type, amplitude, frequency, and duty cycle.
4. Else if the function generator is in the ON run state and the user changes a waveform parameter on the web page, Javascript submits these new values to the PHP script.
5. Return to step 3.

The MCP4725 Module

The *MCP4725.py* module acts as a hardware abstraction layer providing a software interface between higher level calling routines and the MCP4725 device. The module defines a class containing all the necessary class methods to read and write the MCP4725 digital to analog converter. Class methods handle conversion and digital formatting of input and output data.

Software Installation

Important notes:

1. The function generator application should be installed on a Linux host which meets the following requirements:

    - The server software should be installed on a recent distribution such as Debian or Raspberry Pi OS.
    - Apache 7 should be installed and configured to allow serving HTML documents from the user's public_html folder.
    - PHP 7 should be installed and configured to allow running PHP scripts from the user pi public_html folder.
    - The Python I2C smbus library should be installed.
    - Python 3 usually comes per-installed in virtually all Linux distributions. Type "python3" at a command line prompt to verify Python has been installed.

2. See *Setting Up the Raspberry Pi for Internet of Things* hookup guide for detailed notes on how to set up a Raspberry Pi server, and how to install and configure the components in note 1.

Software Inventory

The following software items from the repository need to be installed on the Raspberry Pi

fncgen html folder:
- index.html
- fncgen.html
- fncgen.php

fncgen bin folder:
- fncgen.py
- mcp4725.py

Installing on a Raspberry Pi

Note that the following installation procedure assumes that the document root for HTML documents will be the user's public_html folder. Typically the full path name to this folder will be something like **/home/{user}/public_html**, where **{user}** is the name of the user account hosting node power. The following steps will assume the function generator app is running under the **pi** user account. The **pi** account is the default account when Raspberry Pi OS is first installed on a Raspberry Pi.

1. Follow the instructions in *Setting Up the Raspberry Pi for Internet of Things* hookup guide to install and configure web services on your Raspberry Pi.

2. If it doesn't already exist, use **mkdir** to create a folder **public_html** in the **pi** user's home folder. The full path should look like **/home/pi/public_html**.

3. In the p**ublic_html** folder, use **mkdir** to create a folder called **fncgen** to contain the pushbutton HTML and PHP files. The full path should look like **/home/pi/public_html/fncgen/**.

4. Move all the contents of the repository **fncgen/html** sub-folder to the **fncgen** folder created in step 3.

5. Move all the contents of the repository **fncgen/bin** sub-folder to the **bin** folder in the user pi home folder. The full path to this folder should look like **/home/pi/bin/**.

6. Acting as superuser open **/etc/sudoers** in a text editor. Add the following line to the end of the file

   ```
   www-data ALL=(ALL) NOPASSWD: /home/pi/bin/fncgen.py
   ```

   Adding this line gives the Apache www-data user permission to execute the fncgen.*py* script in the background from the *fncgen.php* script.

7. Open up the *fncgen.html* document in the web browser. You should be able to view the DAC output change on an oscilloscope when interacting with the various controls on the web page. Also, the Python script *fncgen.py* can be started anytime from the command line by entering on the command line `fncgen.py` with the appropriate arguments.

This completes installation of the function generator software.