Seattle Raspberry Jam

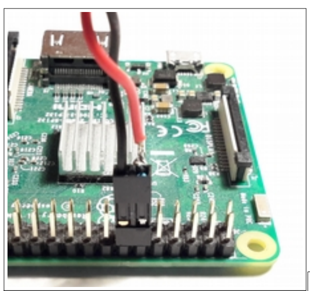
A Raspberry Pi Meetup for Beginners to Experts

RASPBERRY JAM

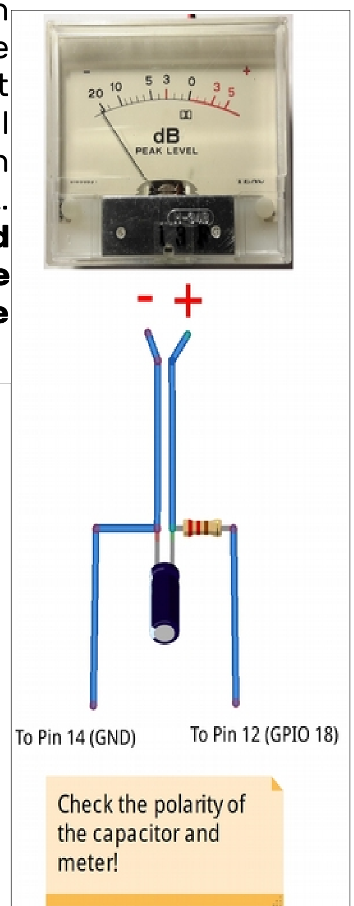
Hardware Project #8: Analog Meter

Assembly

In this project, we will be creating an analog display from an inexpensive galvanometer. If you are not at the meeting, order one of the meters (you will want a "1mA analog meter"), assorted resistors, a 10uF capacitor, and a 2-pin socket. Then, see the wiring diagram to the right and use the following instructions to assemble it. Solder the circuit together with the resistor connecting the positive terminal (don't solder this yet – you will need to determine which value of resistor to use first) of the meter to the first pin on the connector and the second pin on the connector directly connecting to the negative terminal. Then, solder the capacitor across the meter terminals. The capacitor's negative terminal must be connected to the meter's negative terminal. The final step is to plug the pin socket into the Raspberry Pi's pin header on pins 12 to 14 as shown in the picture to the left. **WARNING! Incorrectly installing this socket could damage the Raspberry Pi. Please have one of the organizers check that it is plugged in correctly before applying power.**



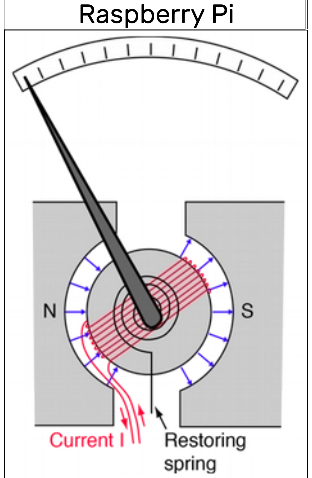
Connection to a Raspberry Pi



How to wire the meter to the Pi. *Created with Fritzing*

Inside our Analog Meter: How a d'Arsonval Galvanometer Works

For this project, we are using a galvanometer, which a current measuring device. This specific type, the d'Arsonval galvanometer (so called because of its inventor, Jacques-Arsène d'Arsonval) is meant to provide linear movement proportional to the current supplied. It also has a returning spring so that the needle points to "0" when the meter is not active. In general, galvanometers work by using the a wire coil (the two ends are the positive and negative terminals) and a permanent magnet. When current flows from the positive terminal to the negative terminal, a a magnetic field is created by the coil. This counteracts the magnetic field of the permanent magnet, which in turn makes the needle rotate. See the diagram above for a graphic representation of how this works.



A simplified diagram of a galvanometer. *Photo credits: HyperPhysics*

 Continued overleaf

Software Setup

After the Raspberry Pi is booted up and you are logged in, we are ready to control the meter! We will be using Pulse Width Modulation (PWM) to simulate a variable voltage level. This will the, in conjunction with the resistor, be used to generate a current through the meter, thereby making the needle move. See the side boxes for a more complete explanation. First, run this command to set up the PWM channel on GPIO pin 18 (hardware pin 12):

```
sudo gpio -g mode 18 pwm
```

Once that is done, continue on to the next section.

Moving the Meter!

Now, we can finally move the needle! Run the following command (replacing VALUE with 100):

```
sudo gpio -g pwm 18 VALUE
```

And needle should move! Next, change the value to 500, and run the command again. The meter's needle should be at the far edge of the scale. If not, try increasing this value (to no more than 1023). If you need to, change the value of the resistor (lower the resistance if the meter is not at the end of the range when the PWM is set to 1023; raise it if the meter is at the end of its range when PWM is less than 500). Once you have the maximum and minimum values, continue onto the next section.

How the Circuit Works

Essentially, we are turning the galvanometer, which measures small amounts of current, into a voltmeter, which measures voltage. This is easily accomplished by attaching a high-resistance resistor in series with the meter, as this will reduce the amount of current flowing through the galvanometer into a value that is in the range of the meter's display. Since the amount of current is directly proportional to the amount of voltage, the Raspberry Pi's output voltage of 0-3.3V is able to use the full range of the meter.

Final Step: Writing a Program

Since we now know how to control the meter, we can implement it in a project. We created some example code that uses the meter to display the Raspberry Pi's current CPU temperature. You can download the fully commented version of this code from https://github.com/polarpiberry/Seattle-Raspberry-Jam (in the Hardware Project #8 folder).

```
#!/bin/bash
#Copyright (c) 2019, PolarPiBerry - Licensed under the CC-BY-SA 2.0

min=0; #Enter your max and min values here
max=500;
sudo gpio -g mode 18 pwm
while :
    do
        raw=`vcgencmd measure_temp`;
        temp=${raw:5:2};
        let temp="(($temp - 30)*100)/69";
        let pwmv="(($max-$min)*$temp)/100 + $min";
        sudo gpio -g pwm 18 $pwmv
        sleep 5;
    done
```

Challenge: You may have noticed that the display's movement is not linear. To correct this, we can use curve fitting. First, collect around 10 data points (evenly spaced between the minimum and maximum PWM values of the meter). Next, use a program such as https://mycurvefit.com/ to fit these points using a 4^{th} degree polynomial. Finally, implement this function in the script and/or in a different programming language so as to preserve the degree of precision.