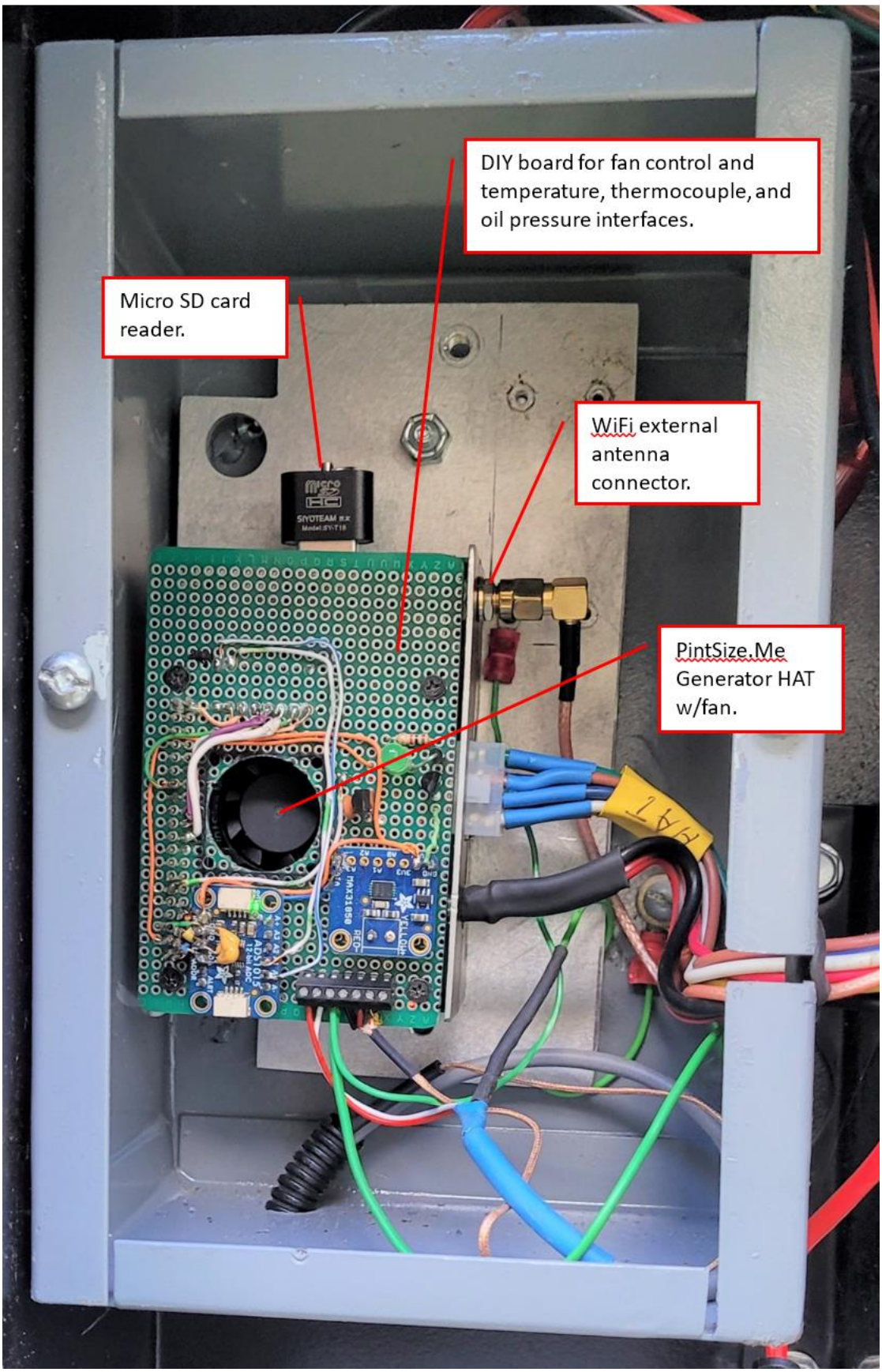
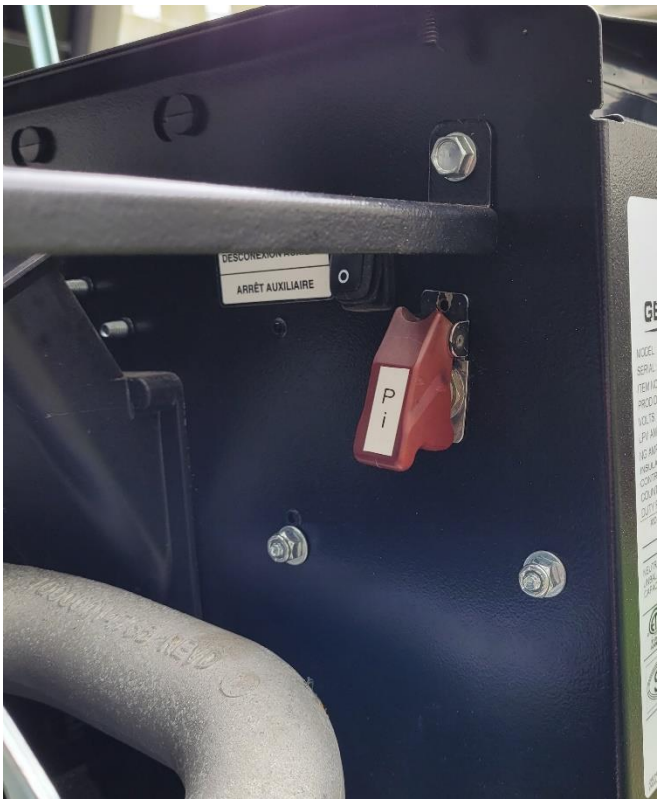
My experience with Genmon began after purchasing a Generac 18kW, air-cooled, natural gas fueled, standby generator. I wanted to be able to monitor the operation of the generator and be alerted to potential problems before they became critical. The Mobile Link app that Generac offers is terribly inadequate and even the paid plans do not give the information that I wanted. Fortunately, I stumbled across Genmon (https://github.com/jgyates/genmon/wiki). In addition to what is offered in Genmon, I wanted to be able to monitor oil temperature and oil pressure. Oil temperature is important because it gives an accurate reading of the operating temperature of the air-cooled engine. Generac only includes a high temperature cut-off that shuts the generator down at 310° F. I want to know about a potential overheating condition before it gets to that critical point. The oil pressure is also important to know. High oil pressure could indicate clogged oil flow and low oil pressure could indicate low oil volume. Again, Generac only provides a low oil pressure engine cut-off which shuts down the generator at the critical point, no warning that problems are developing. I thought I'd describe my solutions to overcome these shortcomings and share my experience so others may benefit.

I am not electronics engineer or software writer. My coding days were 30 years ago and although I am an avid do-it-yourselfer, I am not a circuit designer. As someone else stated, I am a cook, not a chef. All of the circuits that I used were adapted from what others have done and are just one way of accomplishing what I wanted. They may not be the best solutions and I give no guarantees that they will perform over the long run. Suggestions for improvements are always welcome. A big shout out goes to @jgyates for the countless hours he has put into developing and supporting Genmon. Also, thanks to @lmamakos for his work on a thermocouple interface board and @skipfire for the advice and handholding getting my system working. And of course, none of this would have happened without my friend Joe who introduced me to the Raspberry Pi, developed the oil pressure sensor circuit and wrote the code to get and display accurate readings.

My system is built around a Raspberry Pi 4 Model B running Debian v. 11, Bullseye software. The Pi is mounted in a modified aluminum case mainly to help support the interface connectors. The case is left open in order to allow better cooling. The assembly was then mounted on an aluminum plate for installation in an enclosed, steel box. The aluminum plate has a magnet attached to mount the Pi assembly in the steel box which allows easy removal for maintenance. This assembly is located inside a generator cover that requires tools to open so I included a remote, guarded power switch for the Pi so I can easily cut the power in case I need to reboot. The switch is easily accessible when the top of the generator is opened.

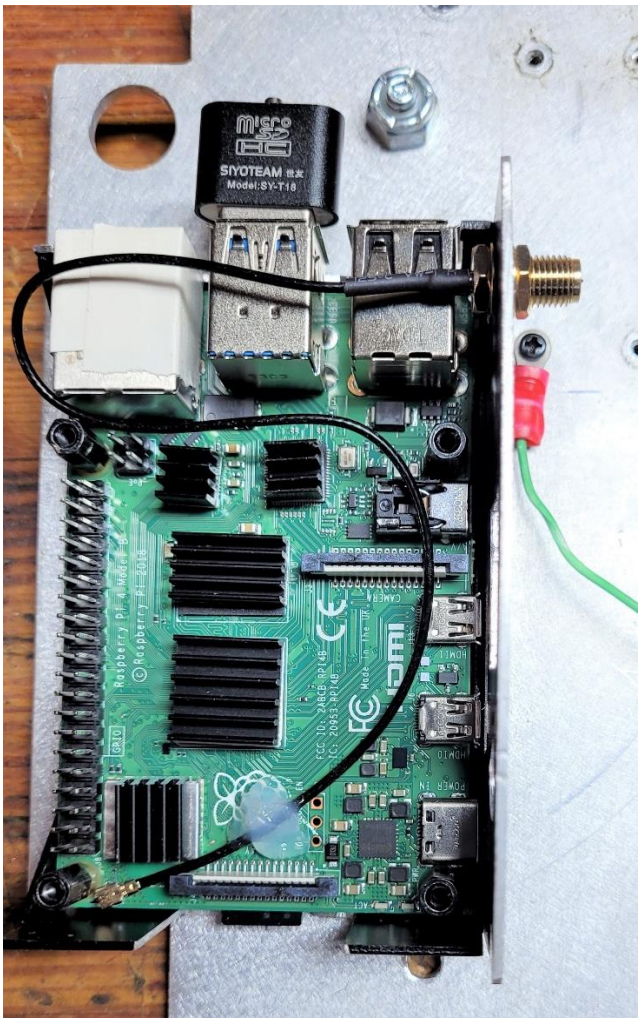

DIY board for fan control and temperature, thermocouple, and oil pressure interfaces.
Micro SD card reader.
WiFi external antenna connector.
PintSize.Me Generator HAT w/fan.



Guarded power switch.

Communication with the generator is through a generator HAT from PintSize.Me using their data and battery power cables. I shortened the data cable to about 16” and installed a new Molex connector. The HAT came with extended GPIO pins so I could add another HAT on top with my home-brew circuits. It also has a built-in fan to provide CPU cooling. I didn’t think the fan was required but, since I planned to install everything in a small, enclosed box, I wanted to provide extra cooling if needed. The fan came set up to run continuously and I felt this was unnecessary so I added a circuit to control the fan with the Pi’s preferences setup. When the CPU gets above 75° C, the Pi turns on GPIO pin 18 triggering a transistor which controls the fan.

I needed to use an external antenna outside of the generator’s metal enclosure for Wi-Fi connection so I disabled the on-board Wi-Fi and added a USB Wi-Fi adapter. Although I had a good Wi-Fi signal, occasionally I would lose my connection and it wouldn’t automatically reconnect. I tried a number of software suggestions that were supposed to force a reconnect but was not successful. I purchased a Raspberry Pi 4B with an external antenna connector from an ebay seller. No more Wi-Fi problems.



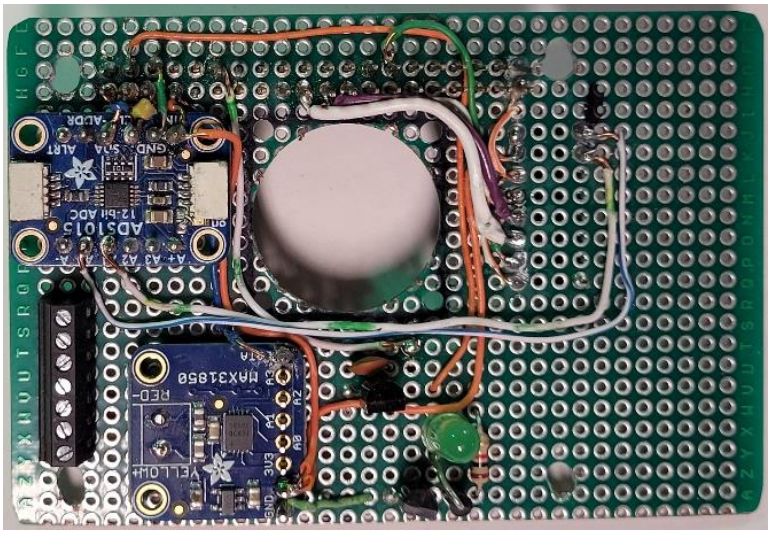
Raspberry Pi 4B with Wi-Fi antenna connector

Remote Wi-Fi antenna.

I also included a USB, Micro SD card reader so I could easily back up the Raspberry Pi's software and operating system. An old laptop was converted to a dual boot Windows XP/Linux machine so I could communicate with the Pi using SSH or RealVNC remote access computing software across my home Wi-Fi network. The laptop also allows me to become more familiar with the new-to-me, Linux operating system leaving the Pi dedicated to the generator. Now I could run Genmon on the Pi inside the generator enclosure and control everything from indoors.

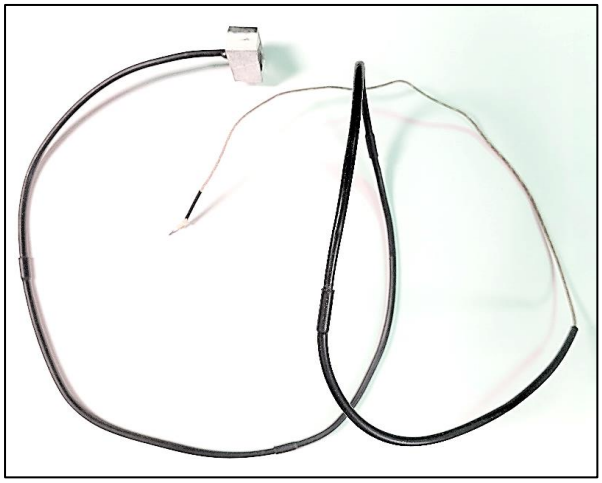
I set up my home computer and phone with a shortcut to easily access the Genmon Status page. I found that after a re-boot, the Raspberry Pi would pick a new I.P. address for communications with my home network. While Genmon reports the I.P. address when starting, I would have to search around, find the new address, and change my shortcuts. A static I.P. address would solve this. The Raspberry Pi has the ability to assign a specific I.P. address but I wasn't happy with this. If that address wasn't available on my network, The Pi would change to another available address and I would still have to update my shortcuts. Instead, I set up my router to assign a specific I.P. address to the Pi. Now my shortcuts would not need to be changed.

Next step was setting up temperature monitoring. @lmamakos has a great add-in board plan that permits multiple thermocouples (https://github.com/lmamakos/RPi-pHat-Thermocouple). Unfortunately, I did not feel comfortable assembling the tiny, surface mount components. I noticed that his circuit was based on MAX31850 thermocouple amplifiers so I purchased a preassembled amplifier that uses the MAX31850 chip from Adafruit along with a 3' Type K thermocouple. Adafruit provides great documentation and links to all libraries needed to get things running. The thermocouple amplifier board was mounted on a piece of printed circuit prototype board along with a DS18B20 digital thermometer, analog to digital converters, and the circuit for controlling the fan on the PintSize.Me HAT. This board plugs into the extended GPIO connector on the HAT – kind of a double HAT.

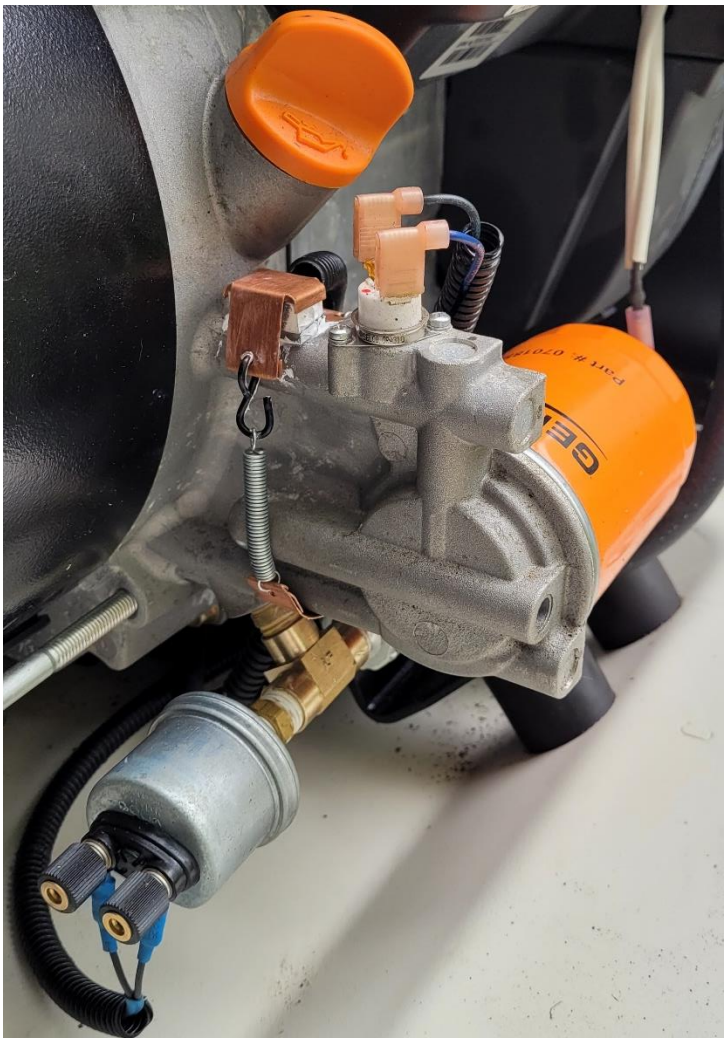


My DIY circuit board for interfaces & fan control.

I covered the thermocouple cable with high temperature, heat shrink sleeving to add extra protection and added a thin coat of high temperature epoxy to the junction to prevent short circuits. I wanted to measure the oil temperature but not risk an oil leak so my first attempt was to thread the thermocouple down the oil drain tube. This worked great except that I found the oil sump provides cooling for the oil and does not indicate engine oil temperature accurately. Using an infrared thermometer, I found that the aluminum housing that provides a mount for the oil filter and oil pressure sensors transmits heat directly related to the engine oil temperature. In fact, this is where Generac connects their high oil temperature cut-out switch. Still leery of the possibility of creating an oil leak, I didn't want to tap into the oil passages so I epoxied the thermocouple tip to a piece of glass which was then clamped to the aluminum housing. The glass (with the addition of some heat sink paste) provides good thermal transmission while insulating the thermocouple circuit from the engine ground. My oil temperature sensor now provides real time readings and could be adapted to measure other temperatures if desired.



The thermocouple is epoxied between a sandwich of glass and plastic.



The thermocouple is held under a copper strap to hold it against the oil filter adapter just below the dip stick. The oil pressure sensor is at the bottom left, teed into the engine's oil passage.

The DS18B20 digital thermometer was added just because I could. While the temperature of the Pi enclosure is not very important, this device can be used to monitor any temperature from -67° F to 257° F. It is interesting to see on a freezing day, the Pi case maintains 70° F just from the heat of the CPU. The thermocouple and digital thermometer are enabled and configured via the Add-Ons page in the Genmon web interface.

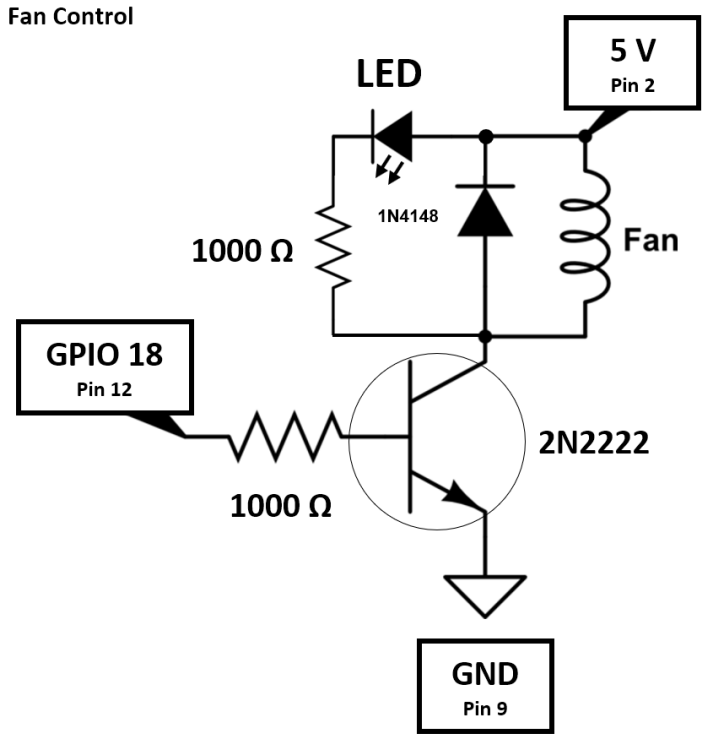
Reading the oil pressure was the next project. I could not find information online by anyone who has done this but my research indicated that the Raspberry Pi along with an analog to digital converter was easily able to take an analog voltage and convert it to a computer friendly, digital signal. My friend Joe was sure he could write a Python program to take that information and output a pressure reading. Once again, I started at Adafruit and purchased 10-bit and 12-bit analog to digital converters. The 10-bit converter is just a chip which runs on the SPI bus. The 12-bit ADC circuit board uses the I2C bus and has amplifying capabilities. My initial experiments had problems with the I2C communications so I included

both converters on my DIY board so I could have the option to switch between them. After encountering data interference problems on the I2C bus, my final version uses the SPI bus circuit and ignores the I2C data.

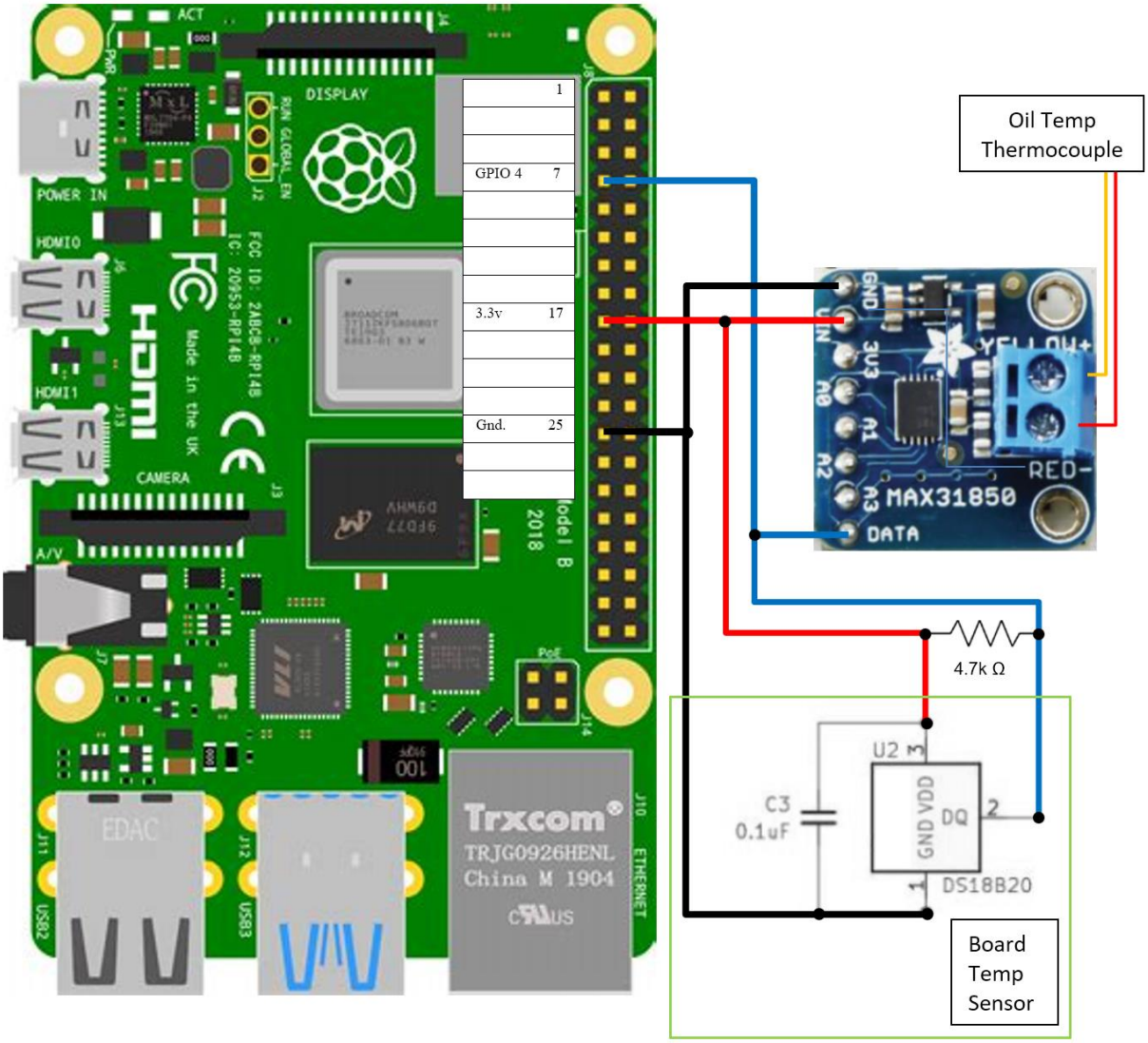
I felt it would be best to have the oil pressure sending unit isolated from ground so I chose a VDO #360-410, 0 – 80 PSI sending unit. This sending unit is teed off of the engine's oil system where the factory, low oil pressure switch is located. Adafruit's documentation and sources were invaluable in figuring out the circuitry and developing the code. Many hours were spent pressurizing the oil pressure sensor with air, comparing the pressure change to the voltage change, and writing a program to accurately display the voltage change as oil pressure. Joe's programming writes the oil pressure readings to a userdefined.json file which is displayed on the Monitor page of the Genmon display. I'm hoping that if there is enough interest, @jgyates will add this as a user option to Genmon and provide a gauge, warning alerts, and have the data included in email notifications.

Circuits:

Below are the individual circuits that I used. My final version combined all of these onto one PC board and the wiring was modified for ease of assembly. The individual circuits shown below should work for you or give you a starting point to make your own design.

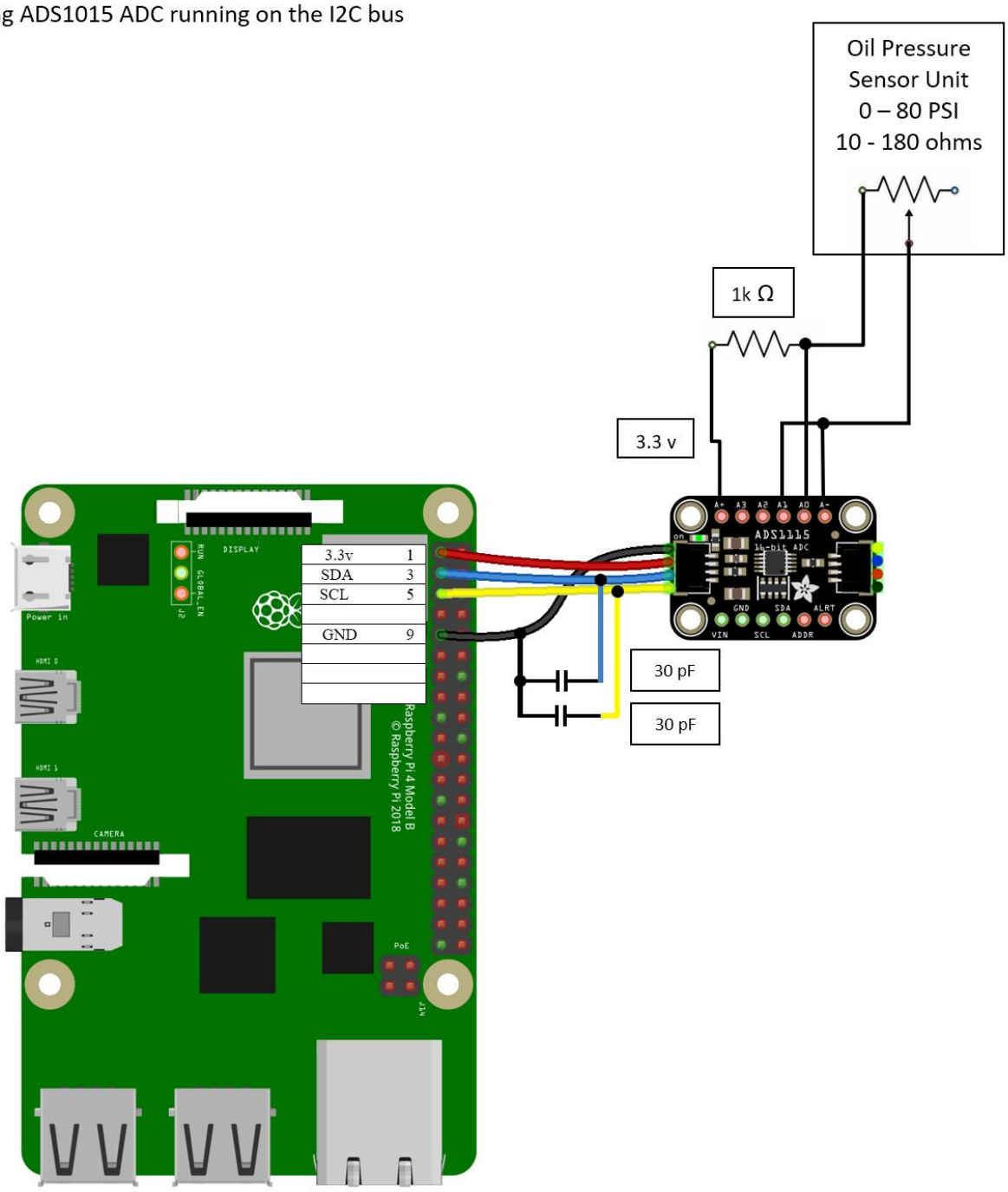


MAX31850K thermocouple amplifier & 1-wire board temperature sensor



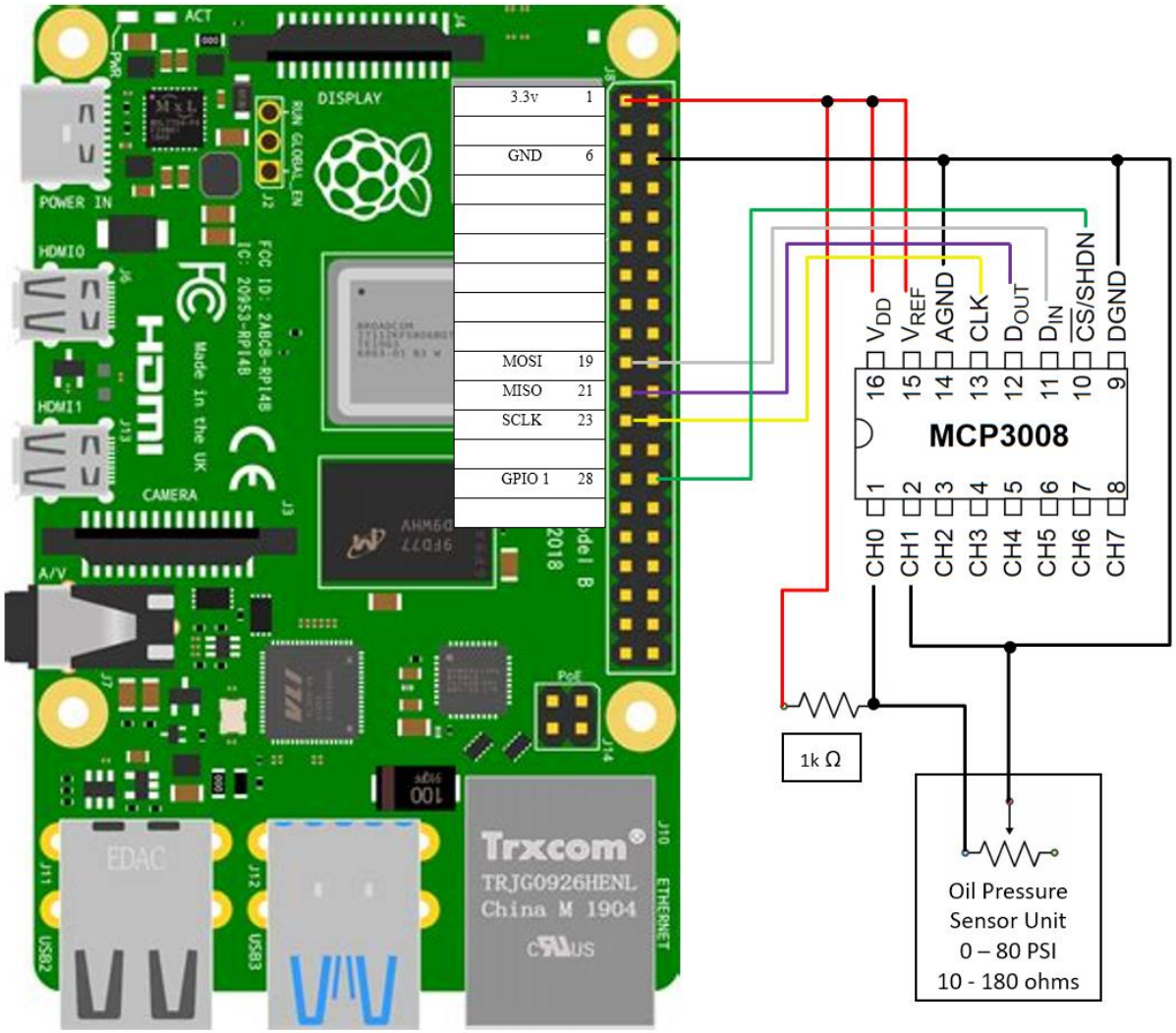
Oil Pressure sensor

Using ADS1015 ADC running on the I2C bus



Oil Pressure sensor

Using MCP3008 chip and running on SPI bus



Things I learned:

1. During initial assembly and testing I used a plastic box and everything seemed to be working well. I installed the system in the generator and found that when the generator was running, the oil pressure ADC using the I2C bus received a lot of interference causing errors and the program to crash. We re-designed the DIY board to include the ability to run the oil pressure sensor on either the I2C or SPI bus. I kept the data lines as short as possible but, in spite of the shielding provided by the steel box and filtering by a pair of capacitors, the interference continued while using the I2C bus. I have switched to using the SPI bus and the problems seem to have disappeared.

2. My first attempt had the Pi screwed down to the box which made it very difficult to change connections or SD cards. Using a magnetic mount as suggested by others made access to the Pi much easier.
3. The Thermocouple didn't give accurate readings when immersed in the oil sump due to the sump providing cooling and the large volume of oil that needed to be heated to effect a change. I connected it to the oil filter adapter and get faster responses to changes in oil temperature. While it does not directly measure the oil temperature, changes in the temperature of the oil filter housing closely follows the oil temperature. Further testing will determine what is "normal" oil temperature as read from the housing and what should be considered critical.
4. Controlling the Raspberry Pi in "Headless" mode via RealVNC works great until communication is lost. When the Wi-Fi went out, I had to remove access covers to get to the power supply for the Pi in order to do a power off reboot. To make this easier, I installed an easy to access Pi power switch.
5. If you are setting up your Pi to run in headless mode, make sure you set the headless display resolution in Raspberry Pi Configuration to the resolution of the screen that you will use for viewing. Otherwise, with no monitor connected, the Pi defaults to composite video and will be almost impossible to use via RealVNC.

This has been a very interesting and worthwhile project. I am now able to monitor important information from my generator and take action before problems become critical. I hope my experience encourages others to take on this project.

In addition to raiding my personal collection of parts, here is a list of sources for most of the more unusual parts and supplies:

Genmon:

https://github.com/jgyates/genmon/wiki

PintSize.Me:

Generator HAT w/power & data cables, tall header, & fan
https://pintsize.me

Thermocouple concept:

https://github.com/lmamakos/RPi-pHat-Thermocouple

Adafruit:

MAX31850K Thermocouple amplifier Product ID #1727
Type-K 3' thermocouple Product ID #270

MCP3008 10-Bit ADC With SPI Interface Product ID #856
ADS1015 12-Bit analog to digital converter (I2C interface) Product ID #1083
Assorted Pi GPIO headers, hardware & connectors
https://www.adafruit.com

ebay:

Raspberry Pi 4 Model B with External WiFi Antenna Connector
VDO #360-410 Pressure Sender Unit 0 – 80 PSI, 10 – 180 ohms
USB 2.0 - Micro SD Memory Card Reader
DS18B20 1-Wire Digital Thermometer
Micro Connectors aluminum Raspberry Pi case
6dBi 2.4GHz 5GHz Dual Band WiFi RP-SMA Antenna + 35cm U.fl / IPEX Cable
RP-SMA MALE RIGHT ANGLE to RP-SMA Female 36" RF Extension Cable
Molex 8-pin connector with male & female housings w/pins - Mini-Fit Jr ™
Double Sided Prototype Printed Circuit Board

Grainger:

INSULTAB Heat Shrink Tubing, 0.13 in I.D. Item #4WND1
8" X 6" X 4" metal box Item #32FG50
(In order to fit in my generator, I had to trim 1" off the side of this box and weld it back together making an 8" X 5" X 4" box.)
https://www.grainger.com

Real VNC:

Remote access computing
https://www.realvnc.com