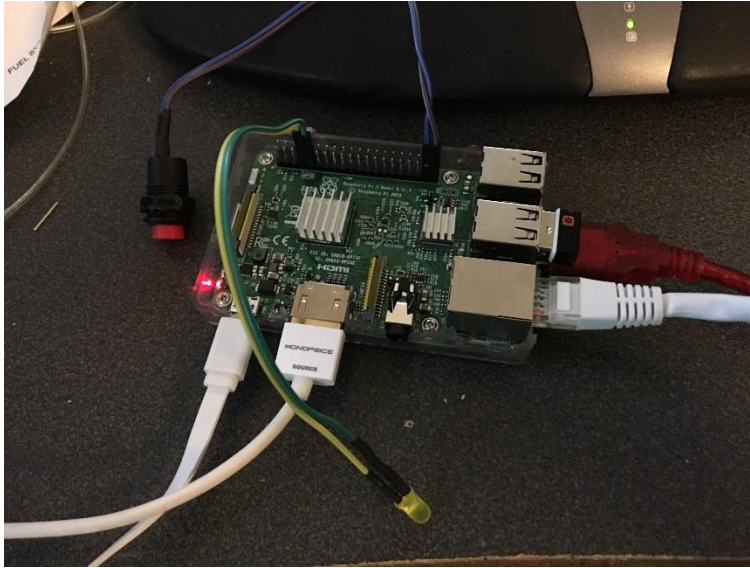


MEGR3241 Advanced Motorsports Instrumentation. Raspberry pi GPS logger project.

V0.21 11/14/21



Logger build guide:

Parts:

Raspberry pi 3 or 4

UBLOX Neo 7 or 8 chipset USB GPS

HDMI LCD

Logitech K400 or similar keyboard/trackpad

USB micro cable for pi 3.

USB C cable for pi 4

Micro HDMI to HDMI for pi 4

HDMI to HDMI for pi 3

**** High quality micro USB cables required, low quality cables can result in random reboots. Shorter cables generally have less voltage drop and are more reliable. A yellow lightning bolt on the upper right hand screen indicates low voltage. Undersized USB power supplies can also be problematic.**

You will likely find this project easier with a HDMI display and a usb keyboard. These are not required. You can also use a video input into a radio head unit but text can be difficult to read.

If those are not available, you should be able to login via SSH using a program called Putty. The Pi supports Bonjour by default. This means you can access it by typing the hostname.local from FTP, SSH or other client. Your hostname is likely obdpi.local if you have modified it. If you have not, it will default to raspberrypi.local

For Example, use filezilla to download you logs over a network connection. Hostname:Sftp://obdpi.local
User:pi Pw:gofast You cannot recover this password if you forget and you must start over. Proceed with caution.

You can use realVNC to remotely login via a phone or PC once your pi is setup and the VNC is enabled.

You will need to know the IP address or the hostname of the pi (as discussed above) to continue and login to your pi via putty over Ethernet.

Install NOOBS to your SD card first. Download NOOBS to a PC, then copy the unzipped files to the blank SD card.

<https://www.raspberrypi.org/downloads/noobs/>

Connect pi to hdmi, micro usb power and attach keyboard. A TV with HDMI works great as a monitor.

Install raspbian with pixel if you installed noobs.

(20-30 minutes)

Login the first time.

Open terminal. You now will see a "console"

Run "sudo raspi-config"

Change password. For the class images will have a password of "gofast"

"sudo passwd root"

"Sudo passwd pi"

Using sudo raspi-config under advanced, Change hostname to "obdpi"

Change keyboard to US, English.

Turn on rdp (remote desktop)

Sudo raspi-config and open advanced settings.

Turn on RDP (remote desktop protocol)

Turn on SSH

Enable SPI

Enable I2C

If you have a custom HDMI LCD, you need to edit the config.txt file in /boot

This file can also be edited from a PC on the SD card. Do not reformat your card on the PC if prompted.

Use this tutorial and install and configure the pi for data logger usage:

From the root directory, use

```
# sudo git clone https://github.com/macsbost/pyobd-pi.git
```

from the pi directory run this command to give write permissions and access to every user

```
# sudo chmod 777 pyobd-pi
```

Change the Directory to go into the pyobd-pi folder

```
# cd pyobd-pi
```

Make a log directory

```
# sudo mkdir log
```

Give the log directory write permissions

```
# sudo chmod 777 log
```

Install GPSD (GPS Daemon) 11/1/2017

Your ublox Neo 7 chipset GPS when plugged into your Pi will show up as /dev/ttyACM0

Linux devices show up as "files" under the device /dev folder.

If you are using a different device, the hardware may have a different file name. You will need to look in the python code and find references to this hardware and change it to match your hardware.

```
# sudo apt-get install gpsd gpsd-clients python-gps
```

```
# sudo nano /etc/default/gpsd
```

Make sure the following lines are there:

```
START_DAEMON="true"
```

```
GPSD_OPTIONS=""
```

```
DEVICES="/dev/ttyACM0"
```

```
USB_AUTO="true"
```

Save, then

```
# sudo /etc/init.d/gpsd restart
```

You can view the raw gps data by typing

```
# cat /dev/ttyACM0
```

You should see some GPRMC and other messages fly by rapidly.

Press Control C to exit.

You can run

```
# gpsmon
```

```
# xgps
```

```
# cgps
```

To view parsed gps data.

For example: #gpsmon

```
pi@raspberrypi: ~
localhost:2947: Generic NMEA>
Time: 2014-12-26T22:01:14.000Z Lat: 40 43' 34.176" N Lon: 74 00' 17.291" W
Cooked PVT

GPGLA GPGLA GPRMC GPVTG GPGSV
Sentences

Ch PRN Az El S/N Time: 220114.000 Time: 220114.000
0 22 194 77 22 Latitude: 4043.5696 N Latitude: 4043.5696
1 14 323 66 25 Longitude: 07400.2882 W Longitude: 07400.2882
2 18 148 41 23 Speed: 0.25 Altitude: 33.5
3 25 125 38 24 Course: 145.29 Quality: 2 Sats: 09
4 31 225 34 36 Status: A FAA: D HDOP: 0.91
5 12 74 34 38 MagVar: RMC Geoid: -34.2
6 51 225 31 32 Mode: A 3 UTC: RMS:
7 4 296 23 23 Sats: 22 12 31 24 32 25 4 1 MAJ: MIN:
8 24 49 22 32 DOP: H=0.91 V=0.86 P=1.26 ORI: LAT:
9 32 313 16 26 LON: ALT:
10 1 323 9 21 GST
11 11 302 8 0 GSV

(68) $GPGSV,3,3,12,24,22,049,32,32,16,313,26,01,09,323,21,11,08,302,*73\x0d\x0a
(72) $GPRMC,220113.000,A,4043.5697,N,07400.2883,W,0.25,138.90,261214,,,D*70\x0d\x0a
(39) $GPVTG,138.90,T,M,0.25,N,0.47,K,D*3F\x0d\x0a
(82) $GPGLA,220114.000,4043.5696,N,07400.2882,W,2,09,0.91,33.5,M,-34.2,M,0000,00
00*69\x0d\x0a
(60) $GPGLA,A,3,22,12,31,24,32,25,04,18,14,,,1.26,0.91,0.86*08\x0d\x0a
(72) $GPRMC,220114.000,A,4043.5696,N,07400.2882,W,0.25,145.29,261214,,,D*7F\x0d\x0a
x0a
```

By using GPSD, many programs can access the gps hardware at one time. GPS can even be accessed remotely over an ethernet connection.

https://cdn-learn.adafruit.com/assets/assets/000/021/936/original/adafruit_products_gpsmon.png?1419631337

If you would like to log your data run:

sudo python gpslogger3241mclaren.py

You can easily copy files from this directory to your google drive using the pi's web browser. Once on your drive they can be viewed from a PC.

You can also use a usb stick. Copy using the command line or using the file explorer.

data can be analyzed with excel, megalogviewer, dataloglab or McLaren Atlas.

Racerender can be used to overlay logged CSV data with video.

To update your operating system periodically run the following commands.

```
sudo apt-get update  
sudo apt-get upgrade  
sudo rpi-update
```

Helpful info:

<http://www.makeuseof.com/tag/15-useful-commands-every-raspberry-pi-user-should-know/>

<http://www.cowfishstudios.com/blog1.html>

<http://www.cowfishstudios.com/blog/obd-pi-raspberry-pi-displaying-car-diagnostics-obd-ii-data-on-an-aftermarket-head-unit>

to start pi gui # startx

to configure options: # sudo raspi-config

the software is installed in “/home/pi/pyobd-pi”

Our USB adapters show up as “ttyUSB0” under the root folder /dev

Change directory # cd pyobd-pi

To go back a directory # cd ..

To go back to your user home director # cd ~

To make a directory # mkdir

To remove a directory # rmdir

To close a program <Ctl C>

To start the gui # startx

Use “sudo” to increase your permission level to allow a program to execute. i.e. # sudo reboot

Display raspi IP address:

ifconfig eth0

Or

ip a

Iphone tools:

Fing – network scanner

Or, use raspberrypi.local or obdpi.local, depending on your hostname.

Webssh – ssh terminal

/boot/config.txt Remoter pro \$4.99 for graphical login over wifi to the xrdp server.

Connect your pi to your phone and use your phone as a remote control.

GIT:

To update our software, go into the pyobd-pi folder and execute

```
# sudo git pull
```

To configure git,

```
#sudo git config --global user.email email@uncc.edu
```

To save your changes:

```
# sudo git commit
```

When you are ready to push to the git server

```
# sudo git commit
```

To add a file to the repository

```
# sudo git add filename to add.example
```

Getting the python script to auto run on boot:

I have created a new script that responds to GPIO input to trigger logging.

This script is called obdlog.py. This does not require a logging switch. I recommend plugging in the OBD2 adapter to the car, boot the pi, then run this python script to log.

Alternatively, the latest script that uses a switch is called obdgpslog2.py The obdgpslog2 script can be used without a gps. You must use a logging switch to start logging with this version.

(follow the steps in this guide for auto load on boot, but look at the line below for the actual crontab command)

<http://www.instructables.com/id/Raspberry-Pi-Launch-Python-script-on-startup/step3/Add-logs-directory/>

To edit the crontab script use the following command

```
# sudo crontab -e
```

add one of the following lines to crontab script

Without GPS:

```
@reboot sleep 20 & sh /home/pi/pyobd-pi/launcher.sh >/home/pi/logs/cronlog 2>&1
```

With GPS:

```
@reboot sleep 20 & sh /home/pi/pyobd-pi/launchergps.sh >/home/pi/logs/cronlog 2>&1
```

You will need to create a /logs directory under /home/pi using the mkdir logs command.

** the sleep command is required to allow adequate time for devices to connect. It prevents the program from generating an exception if started too early in the boot process.

To stop your program after boot, should you need to, open the terminal and type in

```
# sudo killall python
```

You may want to do this to kill the program, modify it so that you can reload it without a reboot.

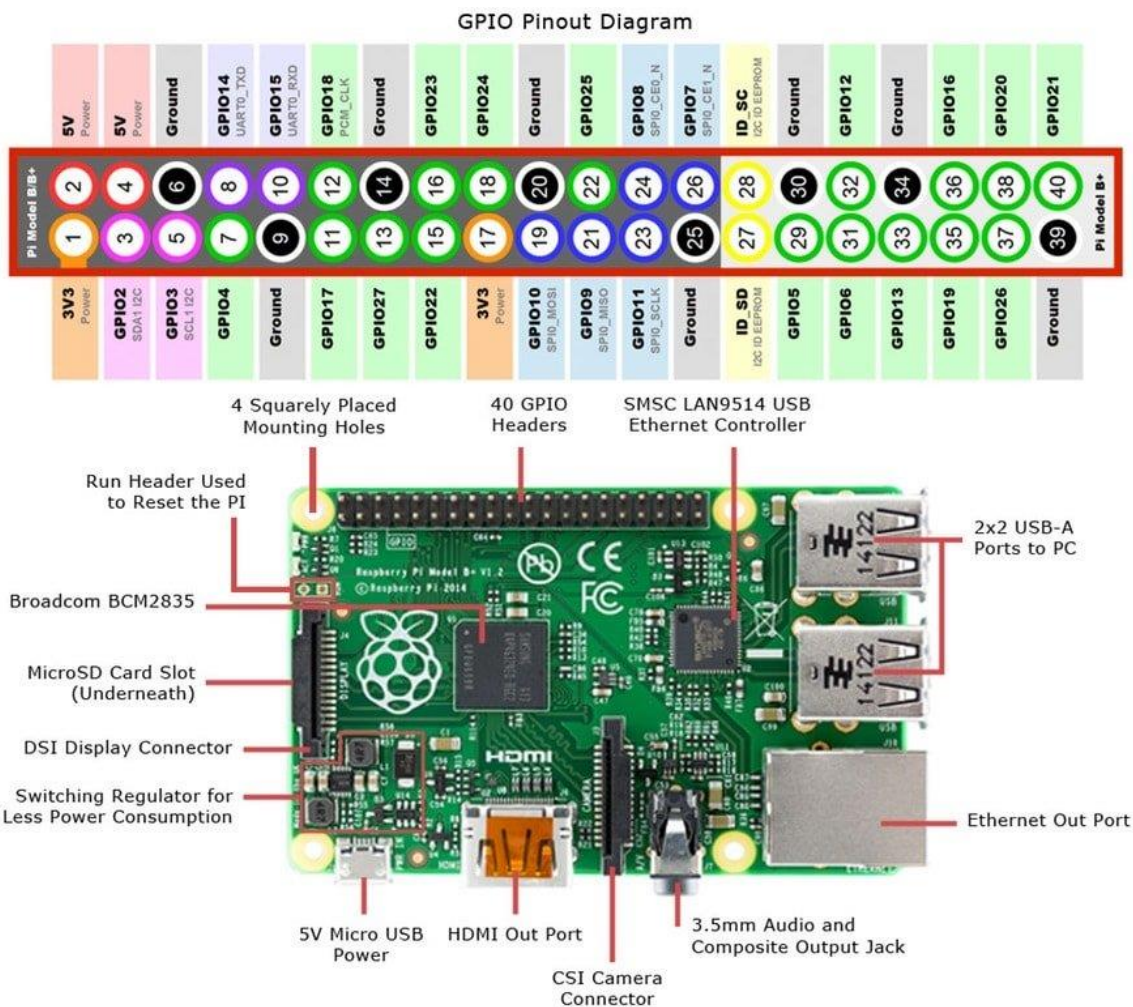
When auto started, you can view the terminal output by pressing ctl-alt-F2. You can get back to the gui by pressing ctl-alt-F7

Logging triggers:

The obdgpslog2.py script will by default log when pins 37 and 39 on the pi are shorted. You need an SPST switch if you want to use a switch. Wire each leg of the switch to each pin. The pin is setup with an internal pull up. The switch shorts out the two pins and pulls the sense pin to ground. The script waits for the transition and then starts logging. If the system boots with the switch pressed in it will not log. It requires being unswitched and then rearmed.

If you wish to add an LED for indicating the python script is logging, it can be added between pins 33, 34 along with the appropriate resistor. . A 220-500 ohm resistor should work.

The positive side of the LED should point to pin 33.



Default behavior can be changed for either script by editing the script files with a text editor.

If a HDMI monitor is not attached to the pi at boot, it defaults to the RCA or headphone jack video output.

To force the HDMI on at boot you can modify this file

`/boot/config.txt`

and add these two lines:

`hdmi_force_hotplug=1`

`hdmi_drive=2`

To have permissions to write this file use

`# cd /`

`# cd boot`

`# sudo nano config.txt`

Use PS to determine if your program is running

`# sudo ps aux | grep python`

You will see a list of 3 processes, two with python if it is running properly.

Adding a GPS Receiver

<https://blog.retep.org/2012/06/18/getting-gps-to-work-on-a-raspberry-pi/>

setup the port correctly

<http://catb.org/gpsd/installation.html>

For UBLOX ttyACM0

Fixes

<https://learn.adafruit.com/adafruit-ultimate-gps-on-the-raspberry-pi/setting-everything-up>

```
sudo dpkg-reconfigure gpsd
```

```
sudo service ntp restart
```

add to autostart

gpsd /dev/ttyACM0

GPS Fixes:

by [amrbekhit](#) » Wed Apr 16, 2014 11:10 am

I know this is quite an old post, but it shows up frequently in search results related to `gpsd` problems on bootup.

Here's a simpler solution than messing around with `udev`.

As brodieh rightly pointed out, running `ps aux | grep gps` shows that `gpsd` is not being started up with the correct parameters, despite the `/etc/defaults/gpsd` file being set up "correctly" via **`dpkg-reconfigure gpsd`**.

A simple way to fix that is to modify `/etc/default/gpsd` so that rather than the device name being placed under the `DEVICES` option, place it under the `GPSD_OPTIONS` option instead. For example, do this:

CODE: SELECT ALL

```
# Default settings for gpsd.

# Please do not edit this file directly - use `dpkg-reconfigure gpsd' to

# change the options.

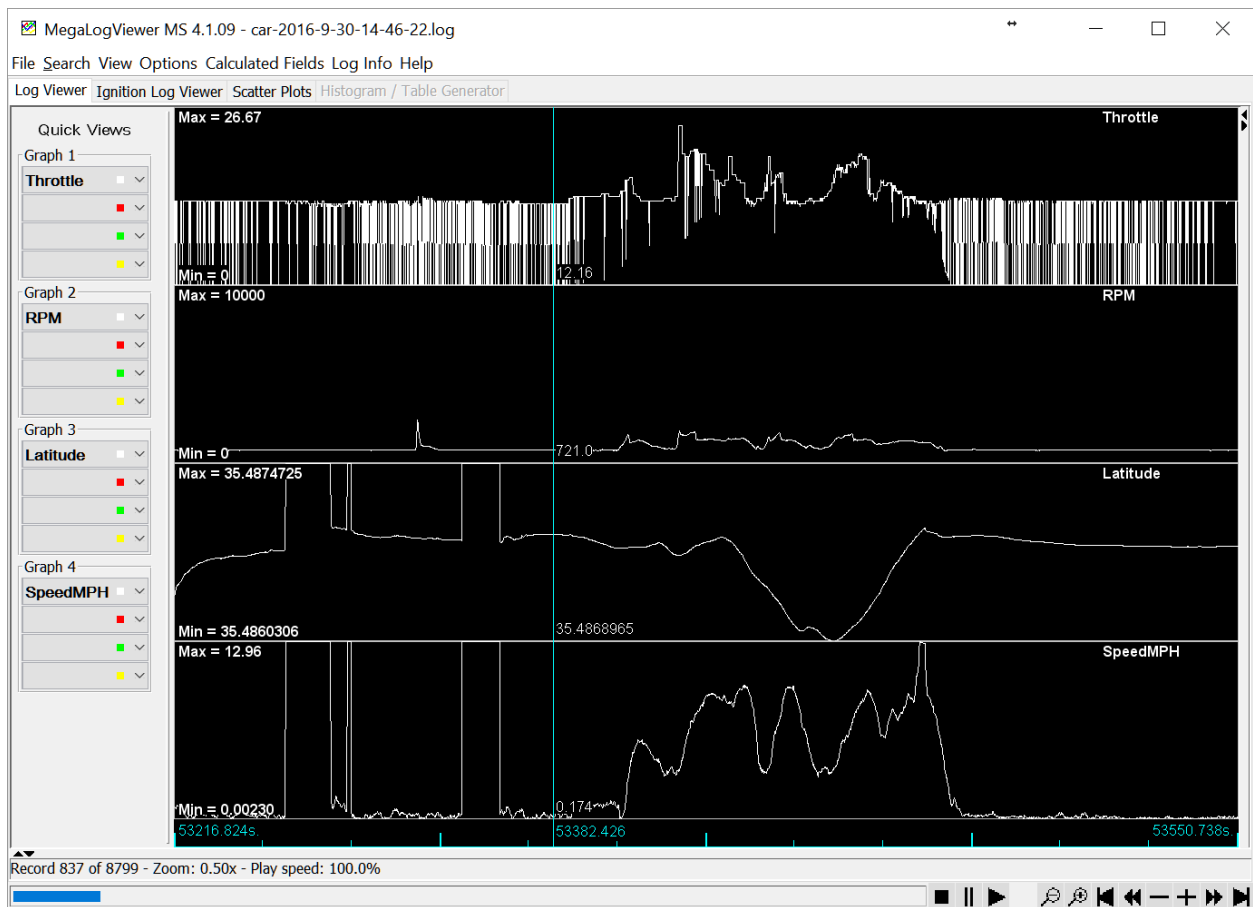
START_DAEMON="true"

GPSD_OPTIONS="/dev/ttyACM0"

DEVICES=""

USB AUTO="true"

GPSD_SOCKET="/var/run/gpsd.sock"
```



Files can be opened by the megalogviewer

Copy with a usb stick, google drive, or SFTP Into the pi and get the log. You use this windows viewer to look at the data.