ECE-331 Datalogger, Josh Andrews

A project to read sensor values from the ECE331 S18 Expansion board via serial, store it in a sql database, and serve up a plot of the data on a publice facing webpage.

Enable ttyAMA0 on Raspberry Pi 3B

To enable normal use of ttyAMA0 some configuration is needed. Bluetooth and the console login for ttyAMA0 must be disabled.

First start by editing raspi-config,

```console
# sudo raspi-config
```

For Rasbian image older than April-2018

- Enable serial in the **Interfacing Options** tab.

For Rasbian April-2018 and newer

- Select the **Interfacing Options** tab and then select **Serial**. On the next window, select NO to disable login over serial, and then YES to enable the serial port hardware on the next window.

Disable Bluetooth

In a terminal window type:

```console
# sudo vim /boot/config.txt
```

All the way at the bottom should be **enable_uart=1**, if not add that line. Also add **dtoverlay=pi3-disable-bt** to swap normal access back to ttyAMA0 and disable bluetooth. Set all changes with a reboot.

```console
# reboot
```

The steps below are not needed for Raspbian April-2018 and above

As a final, two part step, serial console login must be turned off. First enter:

```console
# sudo vim /boot/cmdline.txt
```

and remove **console=ttyAMA0,115200** or **console=serial0,115200** if present. Be careful not to split the line. To stop getty from running perform the following systemctl commands and then set all the changes by restarting the pi.

```console
# sudo systemctl disable serial-getty@ttyAMA0.service
# sudo systemctl mask serial-getty@ttyAMA0.service
# reboot
```

Install Minicom to Test Serial

If Minicom is not installed, use the following commands to install and setup Minicom. We will use it to test we have proper communication with the device.

```console
# sudo apt-get install minicom -y
```

To determine the baudrate of the device connected to ttyAMA0 use:

```console
# stty -F /dev/ttyAMA0 -a
```

I found the baudate of my device to be 9600. Now we can configure minicom to use our devic
e at the specified baudrate everytime. Run the command:

```console
# sudo minicom -s
```

and select **Serial Port Setup**. Change the Serial Device to **/dev/ttyAMA0** and the **Bp
s/Par/Bits** to the correct baudrate and parity (9600 8N1 found above).

At the main Minicom setup screen, select **Save setup as dfl** and then exit. Hitting ente
r should now show 1 set of sensor values on the screen. The expansion hat should now be set
 up and able to be interfaced in python.

For Raspbian older than April-2018, Install pySerial

Install pySerial, if not already, as it will be needed later on to grab the expansion hat d
ata in **hat.py**

```console
# sudo apt-get install python-serial
```

Install Lighttpd

Install lighttpd:

```console
# sudo apt-get install lighttpd -y
```

The default lighttpd webpage can now be seen by entering the pi's local IP address into a b
rowser. To find the IP address, use:

```console
# hostname -I
```

Now copy over the files found in Prof. Sheaff's tlogger repo (index.html and json-data) to
/var/www/html/. A black, blank page should now appear at the same IP address. Well con
figure our new webserver in the next sections.

Enable PHP and SQLite3 Support

Some additional configuration is needed to get PHP and SQLite3 to work with our webserver.

First install PHP and SQLite3, if needed, and then the modules we need.

```console
# sudo apt-get install php7.0 sqlite3 -y
# sudo apt-get install php7.0-cgi php-sqlite3 -y
```

The php7.0 package probably threw out some warnings about Apache, which is fine. We'll be removing Apache in the steps below.

```console
# sudo apt-get remove apache2 -y
```

Retest the default index page shows up to verify lighttpd is still running correctly. If n ot verify Apache2 is removed by using:

```console
# dpkg --get-selections | grep -i apache
```

If the above command returns any results, **sudo apt-get remove/purge** those as well to pe rmanently remove them.

Enable Modules

The cgi modules are not installed by default and must be enabled with the first two command s. Reload lighttpd to set the changes with the last command.

```console
# sudo lighttpd-enable-mod fastcgi
# sudo lighttpd-enable-mod fastcgi-php
# sudo service lighttpd force-reload
```

In order to test everything is set up correctly, navigate to **/var/www/html/** and create **index.php**.

```console
# sudo vim index.php
```

Insert the following code into the file and save it.

```php
<?php
    phpinfo()
?>
```

Going to the Pi's IP address should now show details about PHP and it modules (may need to add /index.php at the end of the IP address). If for some reason the page does not appear, ensure lighttpd is running. The command below should display that lighttpd is both active and running.

```console
# systemctl status lighttpd
```

Also check the error log for more detailed information.

```console
# tail /var/log/lighttpd/error.log
```

Once the webpage displays correctly, you can delete **index.php** as it was only needed for testing purposes.

SQLite3 Database

Verify Sqlite3 is installed and configured (see previous section) before proceding.

In **/var/www/html/** create the database and the table we will be using. We'll worry abou
t permissions later.

```console
# sudo sqlite3 sensor_data.db
sqlite> CREATE TABLE sens_data(timestamp text, temp_hiRes real, IR_int integer,
   ...> full_int integer, vis_int integer, lux integer, temp real, pres real,
   ...> humd real);
```
All done for now, but dont forget to set up permissions

Grab and Store Serial Data

I chose to use a python script for this. I was able to grab the sensor data on **ttyAMA0**
and store the data in a database all in one script. It also removes data that is over 24 ho
urs old.

The script **hat.py** is well documented and should be referred to for more information.

There should be no specific permission requirements on this (besides +x to execute python)
and can live anywhere but preferably not in **/var/www/html**. The database needs to allow
 the python script to to write to the database but prevent writing by the web user. Those
permissions will be setup later.

Note: My expansion hat has not received a firmware update.

Note: Temperature reads high (reads 90F when actual is 70F).

Schedule Cronjob to update database every minute

In a terminal window type:

```console
# crontab -e
```

Add the following line at the bottom

```console
* * * * * /usr/bin/python <full path of script>
```

I used **/home/pi/ece331/datalogger/hat.py** in place of the script path above. The databas
e will now grab and store a new data set every minute as well as delete any data over 24 ho
urs old.

Well it will when we get the permissions set up according to the permissions section below.

Convert database data to json type

The script **sql24.php** queries the sensor database and returns the contents in JSON forma
t. The script should be located in **/var/www/html/** to allow for minimal changes to index
.html.

The script is well documented and should be refered to for additional information.

Permissions should be 644 on this and will be set up in the Permissions section.

Plot the Data

To plot the data we will need to edit **index.html** that was provided by Prof. Sheaff.

At the bottom of the file is a line that looks like this:

```console
var jsondata= $.ajax({url:'sql24.php', dataType: 'json', }).done<omitted>;
```

Ensure where it says **url**, the following is the name of the php script.

We also need to download the javascript modules listed at the top of index.html. Navigate
to your home directory (or one of you chosing besides **/var/www/html**) and run the follow
ing commands. First npm is installed which then allows the modules to be downloaded.

```console
# sudo apt-get install npm -y
# npm install moment --save
# npm install jquery --save
# npm install chart.js --save
```

We only need to copy 3 of the many files downloaded over to **/var/www/html**. Use the foll
owing commands to move the files.

```console
# mv ~/node_modules/Chart.js/dist/chart.min.js /var/www/html
# mv ~/node_modules/jquery/dist/jquery.min.js /var/www/html
# mv ~/node_modules/moment/min/moment-with-locales.js /var/www/html
```

A plot of the sensor data should now appear at index.html. This index file can be editted f
urther, without much effort, to plot multiple data sets.

I was able to add multiple datasets (provided I modified the sql php script to echo the cor
rect data) and alter the fonts/colors.

The html files included in this repo provide more explanation to what was changed over the
default index.html provided by Prof. Sheaff. At very the least, axis labels, title, and da
ta labels were changed to match the data.

Permissions

The database and the directory containing it must be write accessible by the **hat.py** scr
ipt and read only for the web-user (**www-data** by default).

With everything set up, use

```console
```

```
# sudo chown -R pi:pi /var/www/html
```

to recursively change the owner and group of the **/var/www/html** directory and all the fi les it contains. All files in the directory should also be set to 644 permissions, with th e html directory having 755, which can be done using the following.

```console
# sudo chmod -R 644 /var/www/html
# sudo chmod 755 /var/www/html
```

This allows the **hat.py** to be able to access and write to the database without root, pro vided it is owned by pi (which is should be). It also allows the cronjob to execute the sc ript without root permissions. The web user is only allowed Read-Only privileges to all fi les and therefore cannot edit/delete any files on our webserver.

This solution to permissions found at https://www.raspberrypi.org/forums/viewtopic.php?t=15 5067

Setup Port Forwarding and Webpage

This final step is actually perform through your router. In order for the webserver to hav e a public IP, we have to set up port forwarding. The steps below worked for my netgear AC1 200 router.

Login to your router's admin page, at the router's IP address and go to advanced options. There should be an option for port forwarding. Configure the settings on the router to matc h those lsited below. Solution taken from https://thepi.io/how-to-set-up-a-web-server-on-th e-raspberry-pi/

```
Service Port: 80

Internal Port: 80

IP Address: [your Pi's IP address]

Protocol: TCP

Common Service Port: HTTP
```

If you have wireless connection issues you can check the wireless chip logs

```console
# dmesg | grep brcm
```

You can also run the following command to see the status of your connections

```console
# iwconfig
```

and ensure the internet adapter used for this project is running. Connecting to the intern et via WiFi is not reccomended as timeouts and hangups can cause server connectivity issues . Directly connecting the pi to the router via an ethernet cable is the preferred method.M y wlan0 seemed to go into idle mode if the Pi sat untouched for any significant amount of t ime. I tried to disable **power management** using

```console
# sudo iwconfig wlan0 power off
```

This will only disable power management on wlan0, not the wireless adapter. It also did not work to stop the wifi access issues.

Static IP

It is **HIGHLY** recommended that you connect your Pi directly to the router via an ethernet cable and configure the Pi to have a static IP address. This will keep port forwarding always configured to the correct IP address, and is used to increase security and maintain accessibility.

To do so enter the command below. These instructions found at https://raspberrypi.stackexchange.com/questions/37920/how-do-i-set-up-networking-wifi-static-ip-address/74428#74428

```console
# sudo vim /etc/dhcpcd.conf
```

At the bottom should be an example of setting a static IP address. Uncomment it and either choose your Pi's currently connected IP address, or one that is available on your router.

Also we won't worry about ipv6 at the moment so you can delete that line. The **ifconfig** command will also give the current ipv6 address if you want to configure it. Also delete the ipv6 adress in the domain name server line. An example is given below.

```
# Example static IP configuration:
interface eth0
static ip_address=192.168.1.15/24
static routers=192.168.1.1
static domain_name_servers=192.168.1.1
```

The namesever can be found by using the first command. The netmask and current eth0 IP address is found using the second.

```console
# cat /etc/resolv.conf
# ifconfig
```

Now the Pi will remain connected to the router via the specified IP address (this can also be performed for wlan0).

However if the WiFi is not disbaled, it will come back on and try to connect on reboot. To disable WiFi enter,

```console
# sudo vim /boot/config.txt
```

and where we entered **dtoverlay=pi3-disable-bt** earlier, enter **dtoverlay=pi3-disable-wifi** right below it. Now the wifi will be disabled on reboot.

My router also had the option to setup a dynamic DNS through no-ip. I created an account with no-ip and then logged in through my router which allowed my to choose my own URL (to a degree). My website can be found at:

gertrude.mynetgear.com