Report 5/3/2019

By William Kerr

After last week’s meeting, I immediately ported my code to the Raspberry Pi, and it immediately worked after installing modbus-tk, since I wrote my code to change the port format based upon the system it’s running on using the standard sys module.

This week, I decided to go a different direction, since I could not get my hands on the BMS system.

I decided to work on Bluetooth communication. I learned everything I can about Bluetooth within a week. Bluetooth is a wireless networking system. It requires no Internet connection (there are many technologically illiterate people who believe otherwise). The Raspberry Pi 3 Model B+ comes equipped with it. Bluetooth is a wireless way to communicate data between two computers.

I plan on using Bluetooth to communicate data between the slave device, and the master device. I would also like to use Bluetooth to communicate with a PC and write a program to harvest data using Bluetooth.

Getting Bluetooth on my PC was a real hassle. I stayed home for that job. I needed to get the right Visual C++ distributable and Windows SDK installed on my PC (Windows SDK v10.0a and Visual C++ version 14.0), and used the PyBluez-win10 distribution, even though I run Windows 7 on my laptop.

For the Raspberry Pi, I need to run these commands to install PyBluez:

sudo apt-get install libbluetooth-dev python-dev libglib2.0-dev libboost-python-dev libboost-thread-dev

pip3 download gattlib

tar xvzf ./gattlib-0.20150805.tar.gz

cd gattlib-0.20150805/

sed -ie 's/boost\_python-py34/boost\_python-py35/' setup.py

pip3 install .

sudo python3 -m pip install pybluez pybluez\[ble\]

These commands set up a certain distribution of pybluez.

To use PyBluez (either distribution), have an ‘import bluetooth’ statement at the top of any python module that uses it. The PyBluez library

Bluetooth programming relies on Socket I/O programming. This is easily accommodated in Python using the standard socket library.

The Tristar MPPT programming might also go into a different direction, as well. As I was researching Bluetooth, I came across the asyncio library. This is a library that allows you to do asynchronous programming in python. Asynchronous means “concurrent”, not “multi-threading”. For our situation, it means while the Raspberry Pi waits for the Tristar MPPT to send us data, we can tell the Raspberry Pi to request BMS data, send data to our PC via Bluetooth, or do something else that needs to be done within the Linux kernel. If we didn’t have an asynchronous library, we would have to wait for the Tristar MPPT to send us data (which takes a while) before we can even think about asking the BMS to send us data. We can initialize the connection to the server while we wait for the Tristar and BMS’s response to our requests. Once the Tristar and BMS requests are fulfilled, we can send the data over the internet using our GRPS service.

Next week, I plan on continuing to learn how to work with Bluetooth using Python (specifically, sending data), and then I will try using our FONA device to send data to our server. I’ve got the Hayes Command Set PDF on standby. I think our FONA device will be the one to use the I/O pins on the Master Raspberry Pi. FONA uses 2G, which should be enough bandwidth.

I will have to learn about GPRS, TCP, and HTTP protocols that the FONA uses.

I would also like to ask permission to buy and use another GPS-GPRS module that works better for the Raspberry Pi, and is specifically fabricated to fit onto a Raspberry Pi. Here’s a picture:



References:

An Introduction to Bluetooth Programming, Albert Huang. Web. © 2005-2008

[https://people.csail.mit.edu/albert/bluez-intro/c212.html#findmyphone.py](https://people.csail.mit.edu/albert/bluez-intro/c212.html" \l "findmyphone.py)

Beej’s Guide to Network Programming Using Internet Sockets, Brian “Beej Jorgensen” Hall”.

Web. Version 3.0.21 © June 8, 2016. PDF Format.

<https://beej.us/guide/bgnet/pdf/bgnet_USLetter.pdf>

FONA Commands:

<https://cdn-learn.adafruit.com/downloads/pdf/adafruit-fona-mini-gsm-gprs-cellular-phone-module.pdf>

Alternative GSM-GPRS Shield:

<https://sixfab.com/product/gsmgprs-shield/>