EE 629 – IoT using Raspberry Pi

Lesson 3 – Exercises - **Done**

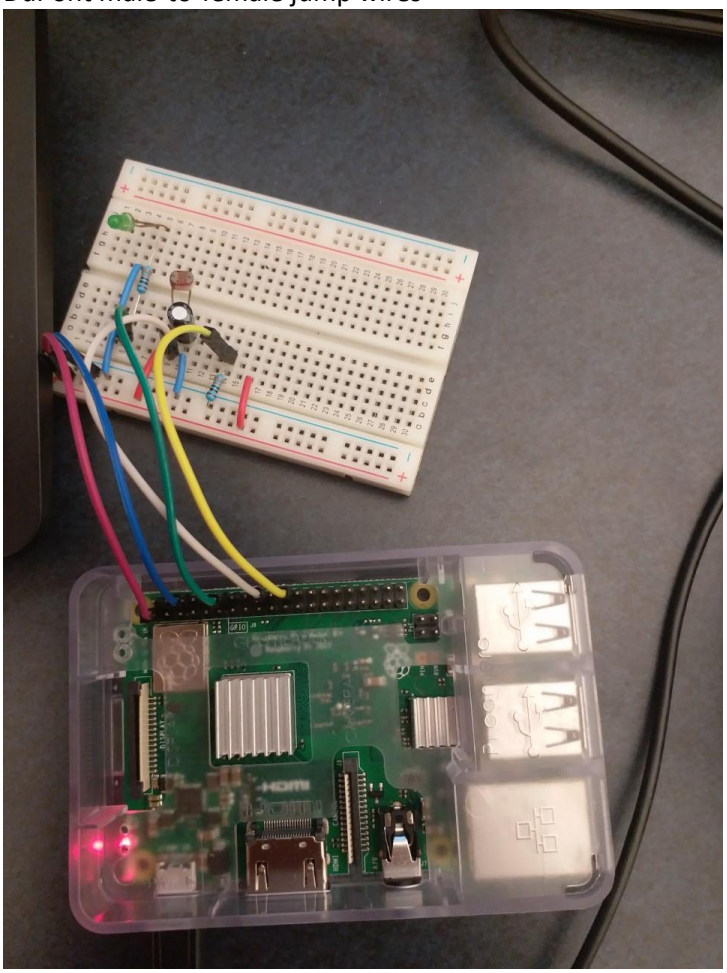
Lab A: Python

On a Raspberry Pi, installed the "jdcal astral geopy" packages, updated the IoT repository, and ran the provided python programs. We have used the respective API's in the program to collect the data and displayed it using the python code.

2. Ran socket server from a Terminal, and ran socket client from another Terminal of the same Raspberry Pi

Lab B: Breadboard

1. Connected the breadboard to Raspberry Pi 3V3, GND, GPIO 18, GPIO 24, and GPIO 25 using five DuPont male-to-female jump wires



2. Ran the blink.py , manual.py and auto.py and noticed the changes in LED

3. Ran the test_email.py and hello.py python programs to learn how to send an email using python programs from raspberry pi.

Lab C: Remote GPIO

1. We need to launch the pigpio daemon using the −n flag to allow connections from a specific IP address of the controlling computer.

2. Since my controlling computer uses Ubuntu Linux distributions,, installed the GPIO Zero and pigpio, and cloned the iot repository and ran the Python program with the environment variable PIGPIO_ADDR and had set to the IP address of the controlled Raspberry Pi

Lab D: 1-Wire

Connect DS18B20 to Raspberry Pi and run the Python program, note that how comfortable it is easy to use the python to get the required data from these devices compared to the previous Lab exercise in Lesson2

Lab E: Pypy

Ran the test.py in pypy, python and python3 and analyzed the time taken by each of the to complete the same process

Lab F: Doxygen

The executable **doxygen** is the main program that parses the sources and generates the documentation. The executable **doxytag** is only needed if you want to generate references to external documentation (i.e. documentation that was generated by doxygen) for which you do not have the sources. Optionally, the executable **doxywizard** can be used, which is a graphical front-end for editing the configuration file that is used by doxygen.

1. **Doxygen Installation:**

Installed the doxygen and followed the below steps:

sudo apt-get install doxygen

cd demo

cp ~/iot/lesson3/pyexample.py .

2. **Creating a configuration file**

Doxygen uses a configuration file to determine all of its settings. Each project should get its own configuration file. A project can consist of a single source file, but can also be an entire source tree that is recursively scanned.

To simplify the creation of a configuration file, doxygen can create a template configuration file for you. To do this call doxygen from the command line with the -g option:

doxygen -g doxygen.config

3. **Ran the Doxygen**

Doxygen will create a html, rtf, latex and/or man directory inside the output directory. As the names suggest these directories contain the generated documentation in HTML, RTF, Latex and Unix-Man page format.

nano doxygen.config

doxygen doxygen.config

Note: This Doxygen makes life easy by creating a template for HTML, RTF and other format files using the python