Learning Objectives:

- Learn to apply Android to control and monitor a device remotely using the internet
- Gain experience with Google Android Things and Firebase
- Gain experience with embedded systems hardware using an off-the-shelf single-board computer and commonly available components

Project 3

Project 3 demos will be Wed, 14-Nov-2018 and Thu, 15-Nov-2018. Deliverables are due to GitHub by 10:00 PM on Sat, 17-Nov-2018

You will work in teams of two on this project (Please self-enroll in teams on D2L)

Project: Building an IoT Home Automation system

Although ECE 558 is entitled *Embedded Systems Programming* all we have produced so far has been software running on an Android device. That's about to change. Project #3 has both a hardware and a software component to it and provides a chance for you to gain experience building an application for the Internet of Things (IoT). For hardware we will be using a Raspberry PI 3 (RPI3), a few resistors, a tri-color (RGB) LED, a PIC microcontroller, an analog temperature sensor and DC hobbyist motor. You will use your Android device to run an app that shares information with the RPI3 across a WiFi or Bluetooth link. Both the control app running on the RPI3 and the user interface app running on the Android device will be programmed in Java using Android Studio and the Android framework.

How is this possible? Introducing Android Things – Google's attempt to bring the Android ecosystem to the embedded system world. Android Things (https://developer.android.com/things/index.html) leverages the existing Android development tools, API's, resources, and a thriving developer community to enable developers to build connected devices for a wide variety of consumer, retail, and industrial applications.

The project involves creating two apps and prototyping hardware and integrating/debugging the apps and the hardware to create your very own *connected device*. Deliverables for this project are the Android Studio projects for both apps and a design report that provides insight into how your implementation works. You will also do a live demo demonstrating the functionality of your IoT device. You do not need to submit a working `.apk` file since it wouldn't be that useful without your hardware to connect to.

All of the external components (RPI3, PIC, temperature sensor, motor, etc.) are available at the EPL store in EB. The cost of the RPI3 and microSD card is about $50.00; the other components add a few additional dollars. We are hoping that working in teams will keep the cost reasonable and that you will find further use for the components as you proceed through the Embedded Systems track. The RPI3 is one of the most cost effective, popular SBC's in the marketplace. It features a Broadcom ARM-based CPU, 1GB of RAM, HDMI, audio, and an expansion connector. The Operating System and files are stored on a microSD card which is programmed (e.g. an image is downloaded to the microSD card) on a PC using a microSD card reader and then installed into the microSD slot on the RPI3. There are many RPI3 starter kits available on amazon.com, digikey.com, adafruit.com, etc. but if you take into account the shipping time and costs you will find the EPL store to be fairly competitive.

Step-by-step instructions for installing the image and bringing up Android Things on the RPI3 are included in the `project3_hardware` document. The `project3_hardware` document also includes a Bill of Materials and instruction for prototyping the IoT circuit and connecting it to the expansion connector on the RPI3. Details on creating the Firebase database project and on the two apps are given in the `project3_software` document. The `project3_software document` also describes the apps. We have left nearly all of the design and implementation details to you. You will do this project in teams of two since there is both a hardware and software component to the project and because time is short. Please self-assign into project teams using the *People/Group/Project #3 Teams* functionality in D2L.

We will be using GitHub classroom for this assignment, only we will enable the Team functionality in GitHub classroom. You will do your development in a local Git repository on your PC that will be linked to a private repository on GitHub that can also be accessed by the instructor and T/A for the course. Each team will have a single shared repository so you may want to view Will Willis' video tutorials on Merging:

- Simple Merges:
  https://www.youtube.com/watch?v=flUN7MA6Yg&index=5&list=PLhO3X757HABdao5hv4HGQSWPX0tnN4lCO

- Complex Merges:
  https://www.youtube.com/watch?v=i-55813ItEM&index=6&list=PLhO3X757HABdao5hv4HGQSWPX0tnN4lCO

Will has several other Git and GitHub-related tutorial that may be worth viewing:
https://www.youtube.com/playlist?list=PLhO3X757HABdao5hv4HGQSWPX0tnN4lCO

Functionality Requirements:

The major functionality for this project is broken down into hardware and software components and include the following:

Embedded System (RPI3 hardware and software):

- Continuously read the ADC channels of the PIC microcontroller from your RPI using I^2C protocol and update the values in the Firebase database and in the mobile app.

- Control the colors of an RGB LED connected to the PIC by adjusting the duty cycle of three of the PIC PWM channels. The duty cycles will be set in the mobile app and written to the Firebase database. The RPI reads the values from Firebase and tells the PIC to adjust the PWM values according to slider values in the mobile app.

- Based on the current ambient temperature (TMP36), set the duty cycle of the remaining PWM channel of the PIC to adjust the speed of the 6V hobbyist motor.

  - [15°C] < TMP36 <= [18°C] ——-——> 30% DUTY
  - [18°C] < TMP36 <= [22°C] ——-——> 50% DUTY
  - [22°C] < TMP36 <= [25°C] ——-——> 70% DUTY
  - TMP36 < [15°C] ——-——> 80% DUTY
  - DEFAULT ————-> 40% DUTY

- Toggle one of the LEDs if the app doesn't throw any I^2C exceptions. If it does, then set the LED (stops toggling). You could use one of the GPIOs of the RPI.

Software (Mobile App):

The Android app is a straightforward single activity app. It should have the following components to it:

- Three sliders to control the intensity of each of the LEDs in the RGB LED
- A progress bar which indicates the current duty cycle of the DC motor
- A Text view & Increment/Decrement buttons for the DAC output
- A Text view indicating the current ambient temperature of the base station
- Three Text views indicating the raw ADC values. The three ADC channels should be connected to the DAC output on the PIC.

The Firebase database should have the following

- Four child values for the PWM
- Four child values for the ADC
- One child value for the DAC
- One child value for the TIME STAMP

A sample screenshot of the user interface in attached in the `project2_software` document.

Demo:

You will give a live demo showing that your IoT application is functioning as intended. The demo should show both the user interface app and the IoT circuit. Changes to the controls in the mobile app should be reflected on the remote

station (RPI3 application) and visa versa. Include enough test cases so that we can determine that the app is working. We will provide a WiFi hotspot to connect your Android mobile device and RPI3 to (it is difficult to get onto the PSU WiFi networks for a device without a user interface.

Deliverables:

- A live demo for your interconnected system.
- Source code for both apps. Your source code should be organized and documented well. Both apps are straightforward with limited opportunities for original design. Be sure your apps handle exceptions gracefully. Make use of Javadocs where appropriate. Since you are using the Android Studio integrated support for Git and GithHub it will be easier to push both (RPI3 and Mobile) Android Studio projects to the shared repository.
- Design report. The purpose of the design report is to provide insight into how your implementation works. Granted, much of the code in these apps will be based on examples you find and will be fairly straightforward it is useful to describe your approach and explain small sections of code that may not be easy to follow. Include information on lessons learned and challenges faced. Explain your methodology for handling exceptions. Describe your test strategy. List the contributions of each of the team members.

Our plan is to assign/grade this project as a team project using GitHub classroom, but this is a new experience for us so we'll see how it goes. You must register your team in D2L before you do your first push so that I can form the teams in GitHub Classroom. In fact, it is not clear that I can even create the assignment until I know all of the teams so please form and register your teams ASAP. Your team will be given a link to a shared private repository for all of your work. Please push/merge your final submission to the master branch of that repository.

Grading Rubric:

You will be graded on the following:

• Functionality of your app	60 points (35 pts hardware, 25 pts software)
• Design report	15 points
• Code quality	25 points
• Total	**100 points**

Extra Credit Opportunities:

There are no opportunities for extra credit on this project. Time is short and the learning curve is non-trivial. Save the extra features for your final project.

Useful Links:

Android Things :

1. https://developer.android.com/things/sdk/pio/pwm.html
2. https://developer.android.com/things/sdk/pio/gpio.html
3. https://developer.android.com/samples/?technology=iot
4. https://www.hackster.io/google/products/android-things

Hardware Debugging Links

1. https://stackoverflow.com/questions/41126915/connect-to-raspberry-pi-3-using-adb
2. https://stackoverflow.com/questions/41145671/how-do-i-connect-my-raspberry-pi-3-running-android-things-%20to-a-wifi-network

Firebase Links

1. https://firebase.google.com/docs/database/security/
2. https://howtofirebase.com/firebase-security-rules-88d94606ce4a

Acknowledgements

This project would not exist in its current form without the hard work and dedicated support of Kiyasul Arif Abdul Majeeth. Kiyasul did the vast majority of the technical development for the project and produced much of the documentation. He created the PIC code, prototyped the hardware and added the features and capabilities that make this project a truly interesting and challenging learning experience. Thanks also to Hiral Borot and Dheeraj Chand Vummidi for their groundbreaking work in bringing the IoT experience to ECE 558.