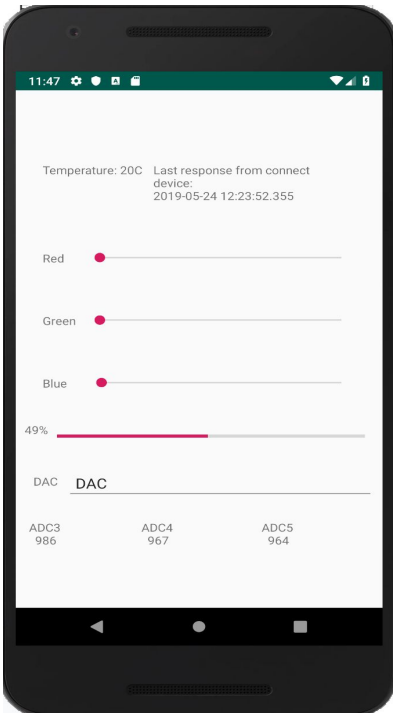
ECE 544 - Spring 2019
Project 2: Android things App
Dhakshayini Koppad
Surya Ravikumar

Project: Android Things



Project Summary

The project involves creating two apps and prototyping hardware and integrating/debugging the apps and the hardware to create your very own connected device. Goal is to create an android application that controls LEDs by setting the intensity. The LEDs will be driven using Raspbery Pi and will also have other components like motor, Temp Sensor etc. Firebase cloud storage is used to publish and access values between the hardware and the application.

Hardware

The Things application communicates with a Raspberry Pi B, which then communicates with a PIC16F15325 via I2C. To communicate with the PIC, the things apps get an I2C peripheral in the app using the peripheral manager. The things app is also connect to Firebase Realtime Database. Once the user changes values on the phone app, firebase is updated. As soon as the value changes on the database, the things app is notified of the change, and takes necessary actions by writing the new values to respective registers. The things app also reads temperature information and ADC values continuously, and updates them on the database on regular intervals with a timestamp.

Once the app is created, it gets a reference to the database, and creates a value event listener that is notified on data change. When this listener is created, it reads the values on the database first time and writes to the registers. Once data changes on the database, this method is called, which reads and writes information to the PIC. Specifically, this method only reads RGB (pwm0, pwm1, pwm2) and DAC1 values, and writes these values to the PIC using the SDKs built in I2C methods.

Two runnables are created to read temperature and ADC values from the PIC respectively. Temperature runnable reads the 10-bit value and converts it to a temperature is celsius using the given reference voltage. It then updates the database with the new temperature. A handler is used to invoke this runnable every minute, meaning the temperature is read and updated every minute. After reading the temperature, an 8-bit value is written to pwm3 port on the PIC that controls a motor. Value written to the pwm3 port depends on the temperature read - different speeds for different temperature ranges. This pwm3 value is also written to the database.

The second runnable reads, ADC3, ADC4 and ADC. These values are dependent on DAC1 value written in the data change listener and all 3 values should be the same. These values should be between 0 and 1024, while DAC value should be in the range 0 - 31. These values are written to the database once read with a timestamp child being written with it. Another handler is used to invoke this method every second.

If there is any issue communicating with the I2C peripheral, a GPIO LED is turned on. Else, the LED toggles every second showing there are no issues communicating with the I2C slave.

Hardware - Physical Circuit

Before everything, a Android Things image had to be flashed to a SD card which the RPI used. The RPI had to be connected to the internet once it booted up. Then, the I2C ports on the RPI had to connected with the PICs I2C ports, along with the 3.3V and GND pins. Since the motor used 5V, the 5V pin on RPI had to be connected to a diode which was connected to a NPN transistor. Once the connections between RPI and PIC were made, pwm ports, adc and dac ports on the PIC had to be connected to the motor, RGB LED, temperature sensor. DAC port was connected to 3 ADC ports.

Since a GPIO LED was used to show I2C communication failure, a GPIO port on the RPI was used for this purpose.

RASPBERRY PI3 MODEL B W/ MICRO SD CARD **(Note: Android Things has NOT be ported to the RPI3 B+ and does not work. Use a Model RPI3 B)**	1
Pre-programmed PIC16F15325	1
TMP36	1
DC Motor 6V 11.5krpm	1
270 ohm resistor	6
330 ohm resistor	1
1N4148 General Purpose Diode	1
2N3904 NPN Transistor	1
Common Cathode RGB LED	2
Prototyping board	1
M-F & M-M jumper wires	As required

Figure 1. Materials required for the project

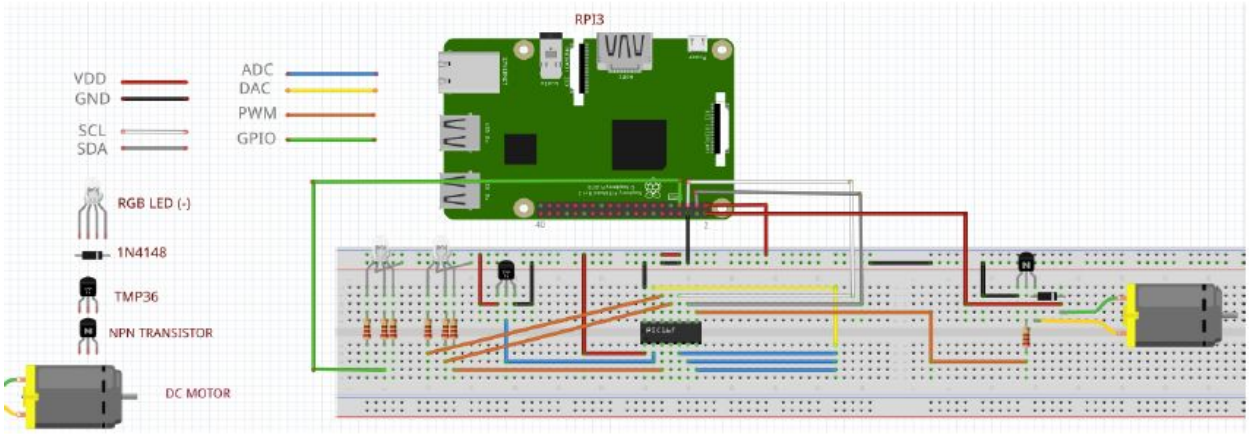


Figure 2. Circuit

Software - Android App

The Android app is a 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.

When building the app for the Software portion of the project, the MainActivity.java consisted of an onCreate and onSavedInstanceState. Inside the onCreate() function, the program set up the seek bars, TextViews and progress so that they would talk with Firebase.

In android studio, the Seekbar has its own “on click listener” called onSeekListener. Inside the listener, there are 3 functions that control the seek bar. onProgressChanged(), OnStartTrackingTouch(), and OnStopTrackingTouch(). The onProgressChanged tracks progress of the seek bar. The onStopTracking is where the data is displayed. Inside the onStopTracking is where the data for the red light and blue light were sent to Firebase.

The function getDataInit() reads the data from the Firebase and sets the data in the text views on the app.

Data Exchange information:

Raspberry Pi Firebase Application

Motor Speed
Time Stamp
ADC1
ADC2
ADC3
Temp

Red intensity
Blue Intensity
Green intensity
DAC

References:

https://androidthings.withgoogle.com/#!/kits/raspberry-pi-3-starter-kit
https://developer.android.com/studio/write/firebase

Contribution:

Dhakshayini Koppad: Worked on the android application and Firebase.
Surya Ravikumar: Worked on the hardware and the Firebase.