Abertay University

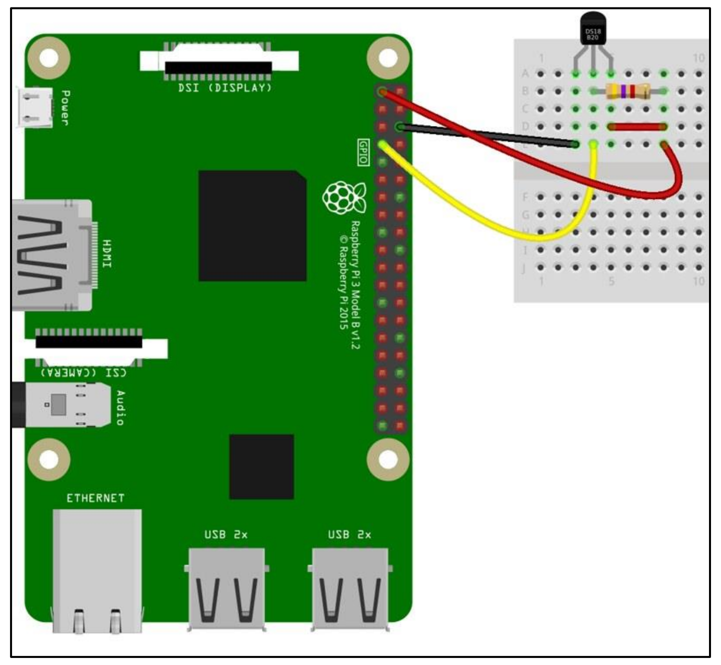
RPi Temperature Sensor

CMP408

Introduction

The aim of this project was to develop an intuitive IoT device that displays and stores the current air temperature using hardware connected to an Raspberry Pi Zero Wireless. The air temperature was recorded using a small thermometer called a DS18B20, which was connected to the RPi via a small breadboard (bigl.es, 2017). To address the aim of the project, the following objectives had to be met:

- Organises and connect the hardware to the RPi e.g., the Adafruit OLED display, and DS18B20 sensor.
- Develop a script using python that reads data from the DS18B20 sensor, then either illuminates three LEDs /displays the data on an display or sends the data to DynamoDB
- Develop an LKM using C that starts the python script when the button is pressed.
- Create table in DynamoDB that stores temperature readings

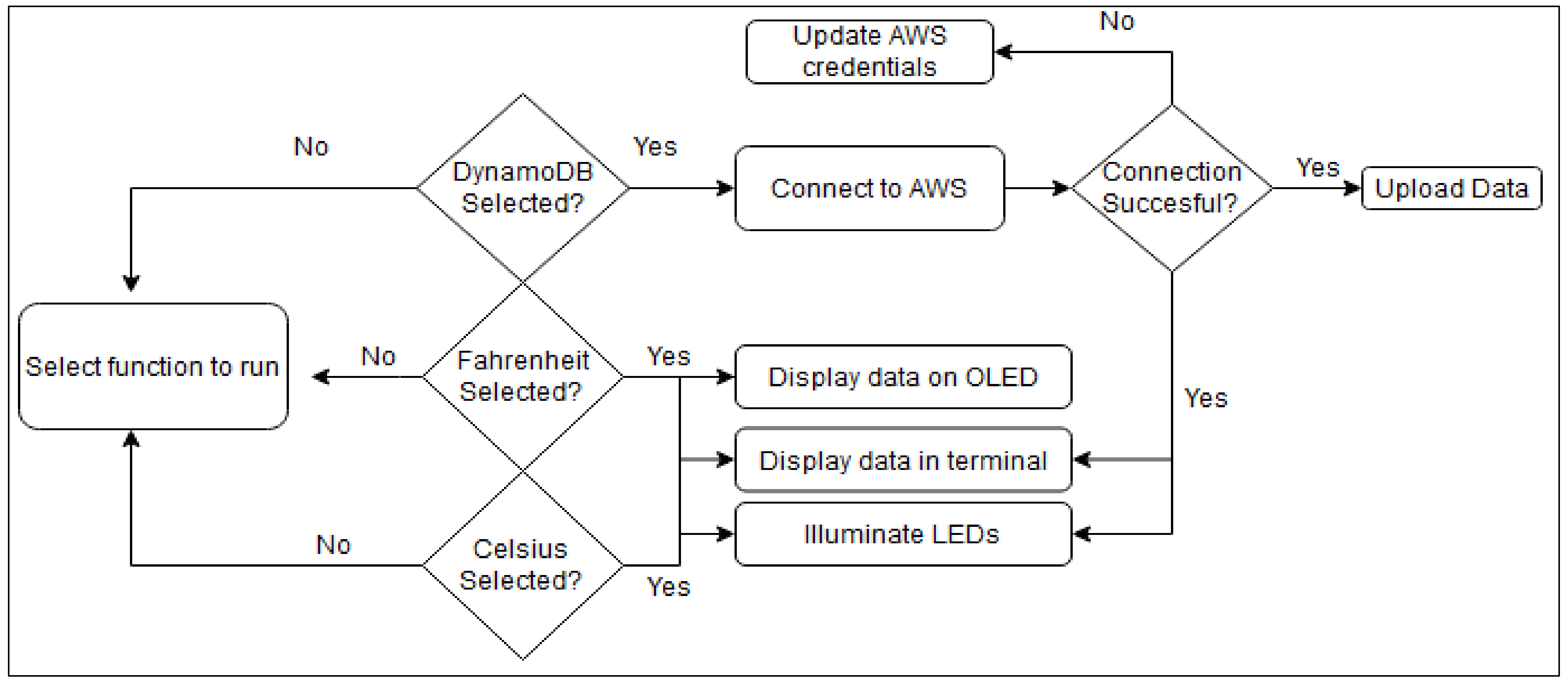


Methodology

To fulfil one of the software requirements of the project, a python script was created that reads temperature data from the DS18B20 sensor, then scripts the output to be displayed on either the terminal on the RPi, a small OLED display, or be uploaded to a DynamoDB. The python script also configures/ sets the values of relevant GPIO pins for three LED lights connected to the RPi. These LEDs will light up depending on what the current temperature value is, e.g. red for warm temperatures and blue for cold temperatures. A useful python module was used to configure the Adafruit OLED display, which allows the user to view the temperature data using a physical device (System, 16). The code also gives the user the opportunity to select which functions they would like to run and what output format they want. They can have the results be displayed in Celsius, Fahrenheit, or immediately upload all sensor data to DynamoDB.

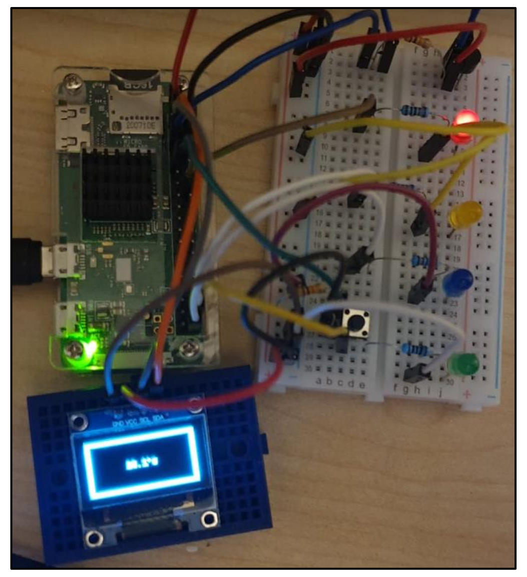
A LKM was created, in order to cover the low level Kernel programming portion of the project. In the LKM an interrupt handler was created to trigger an interrupt service routine when a button is pressed, and a char device driver was made to communicate between the userspace and the kernel space. The LKM included an interrupt handler function and a read function for a char device driver. In short, when the interrupt handler fires, it writes to global variable that indicates that the button has been pressed. The read function (dev_read) is called whenever device is being read from user space i.e., data that is being sent from the device to the user.

The cloud element of the project included the use of a DynamoDB table hosted on AWS. To connect AWS, the project made use of AWS CLI, which was installed onto the RPi along with the python module Boto3 (boto3.amazonaws.com, n.d). When connecting to DynamoDB it was important to use HTTPs to prevent data from being intercepted and/ or changed.



Project Highlights

Overall, the project was a success because all the objectives declared in the aim were met. The python script and LKM both worked during testing and the code managed to upload the data in a reasonable time. However, there is a noticeable delay when starting the python script which could affect the overall user experience.



Future Work

- Better integrate the LKM python script with the main python code.
- Reduce overall latency of the code at startup.
- Implement additional cloud infrastructure e.g., host HTML interface on AWS.

References

System, A. L, 2021. Monochrome OLED Breakouts. [Online] Available at: https://learn.adafruit.com/monochrome-oled-breakouts/python-wiring [Accessed 3. 1 2022]

boto3.amazonaws.com, n.d. Boto3 documentation — Boto3 Docs 1.16.56 documentation. [Online] Available: https://boto3.amazonaws.con/v1/documentation/api/latest/index.html [Accessed 10 1 2022].

bigl.es. (2017). *DS18B20 Temperature Sensor With Python (Raspberry Pi)*. [online] Available at: https://bigl.es/ds18b20-temperature-sensor-with-python-raspberry-pi/ [Accessed 18 Jan. 2022].