Prototype Report

Paul Cooper 220597978

Overview

Background

As a parent of a young child with asthma, a particularly vexing challenge is trying to figure out how the environmental conditions in their sleep space impact their asthma symptoms, and controlling the environment accordingly. I know that temperature and humidity can both be triggers for my child's asthma symptoms and can contribute to illness more generally. I believe it would help considerably in this challenge to know more about the environmental conditions in the sleep space and to be warned if these conditions moved outside an optimal range. To this end, I have tried to design a prototype embedded system to perform these tasks.

Problem Statement

Ideally, babies and children should sleep in a comfortable, healthy space. Parents and caregivers have tools to manage their child's sleep environment such as heaters, fans and air conditioners. However, caregivers often lack accurate information about the environment, often making decisions based on outside air temperature or even just on "feel".

Requirements

1. As a user, I would like the system to tell me the real-time temperature and humidity of a given room so that I can take actions to maintain a healthy environment. The system should
    a. Take readings of the temperature and humidity
    b. Display high level status via a "traffic light" style dashboard of LED lights
2. As a user, I would like to notified if the temperature moves outside of normal temperature and humidity thresholds. In particular, the system should:
    a. Warn me that the threshold has been reached by sending a notification to my smart device
    b. Update the traffic light LED dashboard to reflect the current environment state.

Design Principles

The dashboard design should provide affordance: it should help the user to understand the environment status at a glance by using an already known 'traffic light' colour LED system

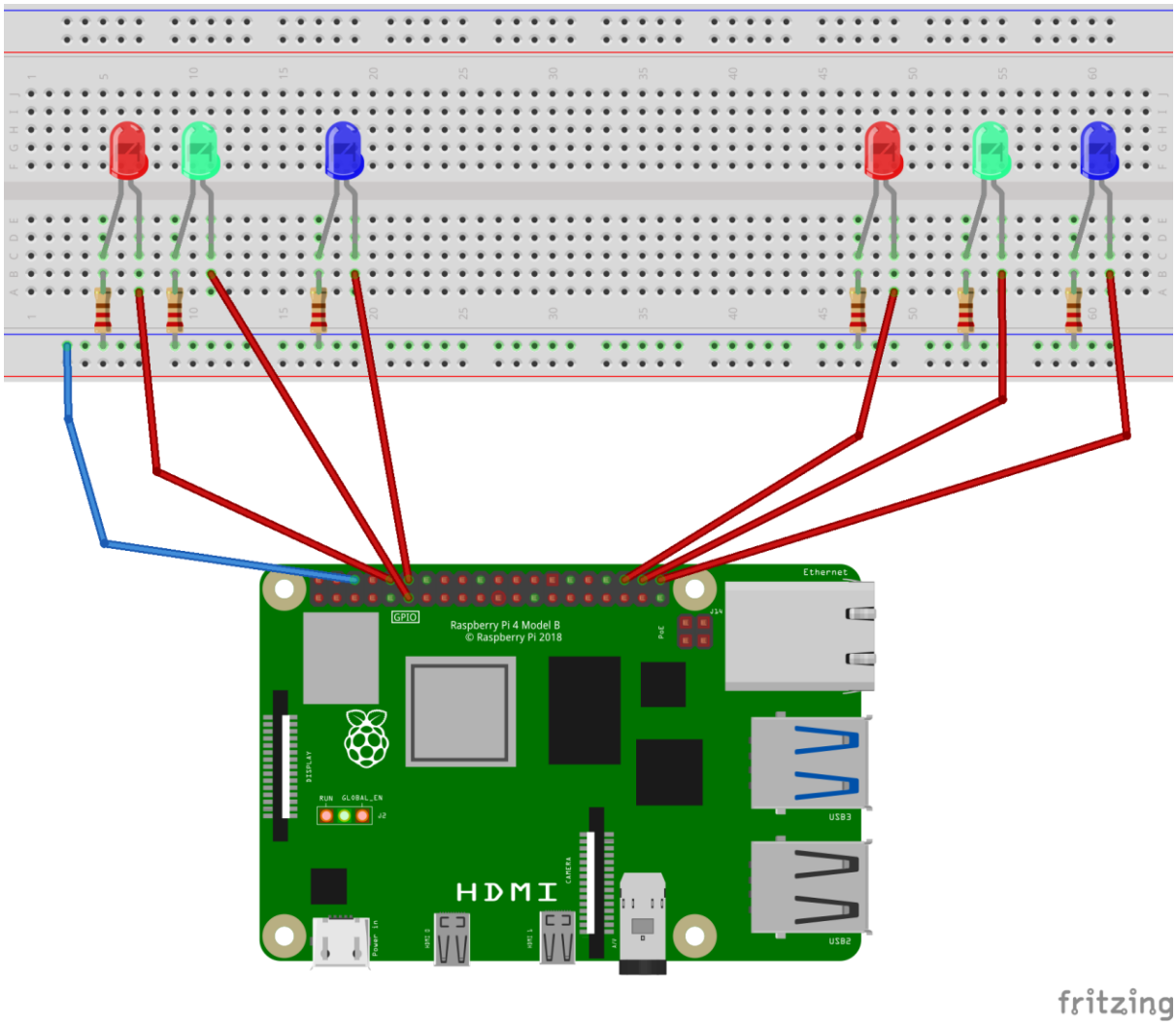
The notification system should provide only the most useful information to the user. Only threshold breach alerts will be communicated, rather than the temperature and humidity levels.

Prototype Architecture

Particle Argon

A DHT22 Temperature and Humidity sensor is connected to a Particle Argon in order to take regular temperature and humidity readings. These readings are then published to an MQTT broker.

Raspberry Pi



A Python program running on a Raspberry Pi subscribes to the relevant MQTT topics to listen to the temperature and humidity information from the Argon.

The program compares the temperature and humidity to normal boundaries which are set as thresholds in the program. If the temperature or humidity readings fall outside the boundaries, the program will trigger 2 actions:

1. An LED dashboard will be updated to give visual feedback
2. A message will be published to the MQTT broker with the details of the threshold breached

MQTT Push Client Android App

A third party Android app will be installed on the user's smart device, and subscribed to the relevant topics of the MQTT broker. When an alert is received via MQTT, a notification with sound will be triggered on the user's device.

Source Code

GitHub link

Testing

A high-level suite of functional tests is laid out below:

Name	Goal	Expected Result
Temperature and humidity GUI	Validate that temperature and humidity data are accurately rendered via the Raspberry Pi GUI	Temperature and humidity data should be displayed to the user via GUI on the RPi.
Set traffic light thresholds	Validate that a user can configure a temperature and humidity threshold via the RPi GUI	Threshold can be entered and saved by the user. The threshold is updated accordingly.
Threshold operation	Validate that traffic light LEDs are triggered by appropriate thresholds	Traffic light LEDs illuminate and extinguish according to the thresholds configured in the system as environmental conditions change
Notification operation	Validate that a notification is sent to a nominated smart device when a threshold is reached	Notification received on target smart device according to the thresholds configured in the system

User Manual

Step 1 – Collect your materials

You will need:

- A Particle Argon
- A Raspberry Pi (assumption – you also have a monitor, keyboard and mouse)
- 2 breadboards
- 1 DHT22 Temperature and Humidity Sensor
- 2 x Green, 2 x Red and 2 x Blue LEDs
- A laptop computer
- Assumption – you have access to male to male, male to female cables and resistors for routing circuits

Step 2 – Set up your Argon

Connect the DHT22 sensor to the argon as follows (from left to right with the device facing you):

- VCC to 3v3 (on my prototype I have connected VCC>breadboard + rail>3v3
- Data to D2
- Pin 3 is not connected
- GND to GND (on my prototype I have connected GND> - rail >GND)

Then connect your Argon to a laptop and flash it with the Argon source code linked above via the Particle web IDE. Confirm the program compiles successfully before moving on

Step 3 – Set up your RPi and Dashboard

Insert the 6 LEDs into the breadboard you will be using for your dashboard. Connect one side of a 220 ohm resistor to the negative power rail, and the other to the same terminal strip as the short leg of each LED.

Use male to female cables to make the following connections between your RPi pinout and breadboard:

- GND (pin 6) to negative power rail
- GPIO15 (pin 10) to red temperature LED long leg terminal strip
- GPIO17 (pin 11) to green temperature LED long leg terminal strip
- GPIO18 (pin 12) to blue temperature LED long leg terminal strip
- GPIO16 (pin 36) to red humidity LED long leg terminal strip
- GPIO20 (pin 38) to green humidity LED long leg terminal strip
- GPIO21 (pin 40) to blue humidity LED long leg terminal strip

Once this is done, compile and run the python program from the source code above on your RPi. You should notice at least one of the following and possibly both depending on environmental conditions around your sensor:

1. A terminal should open on your monitor and you should begin to receive temperature and humidity updates.
2. Your LED dashboard should light up according to the temperature and humidity thresholds.

Step 3 – Set up your Android smart device

On the Google Play store, download MQTT Push Client app. Add a new server (the + icon) with the following configuration:

Push notification server

- host: push.radioshuttle.de

MQTT server:

- host: test.mosquitto.org
- port: 1883

all other options can be left alone.

Then, select MQTT Topics and then again the + icon. Add the following topics:

snugnsound/temp/alerts

snugnsound/humidity/alerts

and save.

Once this is completed, you should start to receive notifications whenever the sensor detects temperature or humidity outside the threshold levels.

Conclusion

The project was a moderate success in meeting the requirements I set for the prototype. I feel I gained experience in a couple of interesting areas – especially coordinating multiple devices with MQTT servers and clients.

I faced a number of significant issues in getting the prototype completed. Firstly, I had initially planned to handle communication between devices using webhooks from the Particle Argon, but I quickly realised that it made more sense to use the RPi as a “controller” and the Argon more as a sensor. This led me to use MQTT, which I had no initial knowledge of. After a very steep learning curve, I found it very useful. Secondly, I had initially planned to also make a UI with some basic dashboard stats, and importantly some ability to set and update the thresholds in the device. However, this proved too much for the timeframe, and I made the decision to focus on the dashboard and notifications as the most useful features of my prototype.

Given a second chance at the project, I definitely would have spent more initial time gaining clarity on the hardware and software I planned to use and making sure I had some background on key functions, design patterns, and where to find support and documentation for the chosen items.