Final Project

Building Management System

EECS 113

Created by:

Daisuke Otagiri

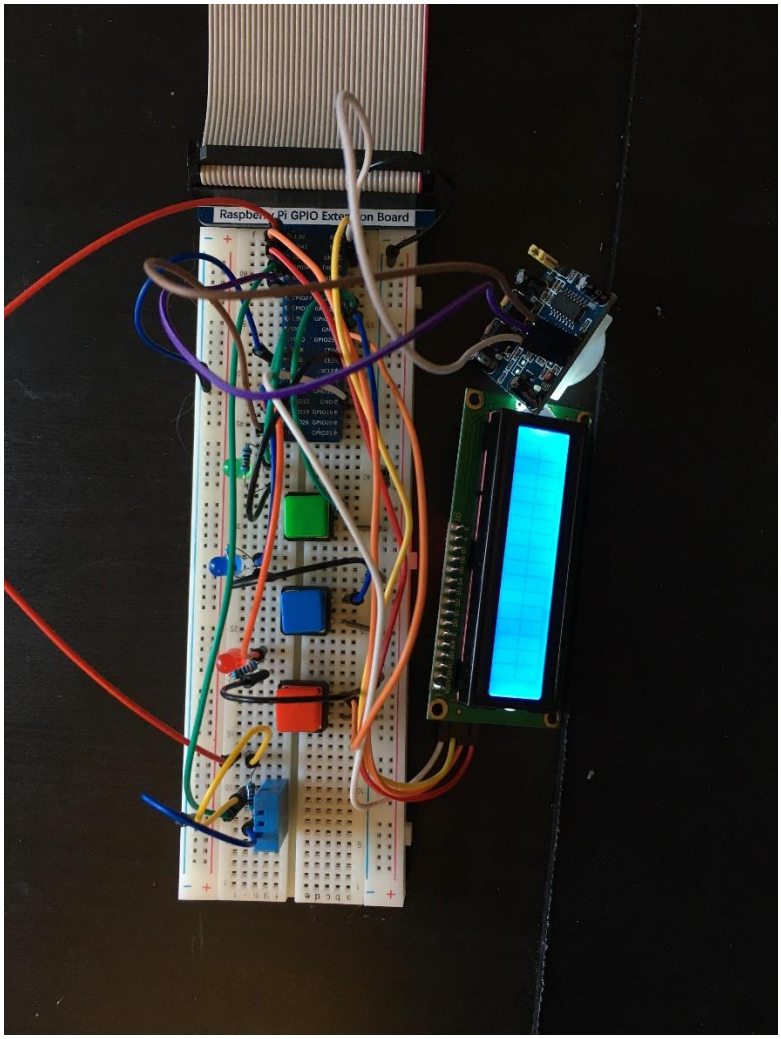
June 9, 2021

SID: 12823053

Introduction

The goal of this project is to create a computer-based control system for buildings that controls and monitors the building's mechanical equipment such as ventilation, lighting, power systems, fire systems, and security systems. This Building Management System (BMS) will complete the following functions: displaying a status on the LCD, control and display room temperature (HVAC), control ambient lighting, and monitor human presence in the room. This project aims to provide accurate information of temperature, humidity and weather index as much as possible by utilizing the California Irrigation Management Information System (CIMIS) database.

Schematic



Schematic Description

GPIO Pin List:

- DHT-11 – Pin 4
- Red Button – Pin 25
- Blue Button – Pin 24
- Green Button – Pin 23
- Red LED – Pin 19
- Blue LED – Pin 13
- Green LED – Pin 6
- PIR Infrared Motion Sensor – Pin 17

LCD Display:

There are 4 pins to plug in for the LCD display which are GND, VCC, SDA and SCL. The red jumper wire connects the SCL to the SCL1 port located on the top left of the GPIO extension board. The orange jumper wire connects SDA to the SDA1 port locates above the SCL1 port on the GPIO. The white wire is for GND which is plugged directly into the GPIO GND port located at the bottom right. And lastly, the yellow wire connects VCC to the 5V power port at the top right of the GPIO.

PIR Motion Sensor:

There are 3 pins leading out of the sensor. The white wire plugs directly into the 5V port at the top right of the GPIO board. The purple wire leads into GPIO pin #17. And lastly the brown wire is for GND at the bottom left of GPIO.

Adafruit DHT-11 - Temperature and Humidity Sensor:

The DHT-11 is grounded via the blue jumper wire from the 4th pin to the left ground rail. The ground rail is connected to the GND on the GPIO with the blue wire in the top left section of the GPIO. The 3.3V at the top left port of the GPIO is plugged into the 1st pin of the DHT-11 with a red jumper wire going across the breadboard. Lastly there is a 10kΩ resistor connecting to the 2nd pin on DHT-11 where the GPIO pin #4 is also connected to. The 10kΩ resistor connects the 3.3V source in series with the 2nd pin of the DHT-11.

Red Button

Uses GPIO pin #25 (orange jumper wire) and is grounded to the ground rail on the right. The ground rail is powered by the top right ground port on the GPIO

Blue Button

Uses GPIO pin #24 (blue jumper wire) and is connected to the ground rail on the right (same as red button)

Green Button

Uses GPIO pin #23 (green jumper wire) and is connected to the ground rail on the right (same as red button)

Red LED

Positive pin of the LED is connected to GPIO pin #19 (orange wire). Negative pin is connected to the 220Ω resistor which is then grounded using the right ground rail (black wire)

Blue LED

Positive pin of LED is connected to GPIO pin #13 (blue wire). Negative pin is connected to the 220Ω resistor which is then grounded using the right ground rail (black wire)

Green LED

Positive pin of the LED is connected to GPIO pin #6 (green wire). Negative pin is connected to the 220Ω resistor which is then grounded using the right ground rail (black wire)

Equipment List

- Raspberry Pi Model 4
- DHT-11 (Temperature and Humidity sensor)
- x3 220Ω Resistor
- x1 10kΩ Resistor
- x3 Big Push Buttons
- x1 Red Push Button Cap
- x1 Blue Push Button Cap
- x1 Green Push Button Cap
- x1 Red LED
- x1 Blue LED
- x1 Green LED
- I2C LCD 1602
- 40 pin GPIO Cable
- GPIO Extension Board
- Breadboard
- Jumper wires
- Infrared Motion Sensor

cimis.py

This file acquires the humidity and time of day from the CIMIS database by using the URL directly. The cimis_data class holds the information for humidity and time retrieved from the CIMIS database. The target and appkey was found using the CIMIS website. Under “Data”, the station number for Irvine was 75 and the appkey was created after registering for an account. From these information alone, we can generate a URL with information regarding humidity.

Functions:

- `def get_cimis data_for (current hour):`

  This function takes in the current_hour of the day to obtain startDate and endDate for URL that will be created later on. The goal of this function is to return a cimis_data object where the date, hour and humidity is returned to the main file.

- `def retrieve_cimis_data(url, target):`

  The function takes in the URL from run_cimis function that calls it and does urlopen to retrieve the data from the website. Depending on the polling speed, this will be the main function that will affect the time of the entire program. (Note: sometimes the website is slow so the polling will be also slow)

- `def run_cimis(appKey, target, start, end):`

  Creates the URL for the CIMIS website using my personal appkey, the Irvine weather station ID (75) and the dates from the get_cimis_data_for function. Once it creates the URL it will call retrieve_cimis_data and put the retrieved information into the variable data. If data is not None, then it will return the information back to the program.

main.py

This is the main file which handles the entire program. Includes functions for checking temperature, opening/closing the door, PIR motion sensor, displaying things on the LCD, and controlling the AC and heater when the blue and red buttons are pushed. The entire file uses global flags to indicate the state of the system so the LCD can display accurate stages that the system changes to. The external libraries used in this file include Adafruit_DHT and drivers which are third party libraries provided by Freenove.

Functions:

- **`def setupGPIO():`**

  Sets GPIO warnings to False, sets the GPIO mode to BCM and also sets up all the buttons and LEDs used in the program

- **`def checkTemp():`**
  - Threaded function that is created in main.
  - While the event for checking the temperature is set, 3 temperatures over the course of 1 second will be recorded to take the average of.
  - Function will also use the **`get_cimis_data_for`** function from the **`cimis.py`** file to retrieve humidity from the CIMIS website. If the data is None from the CIMIS website, the program will attempt to retry retrieving the data again in 1 hour.
    - Otherwise if data was retrieved, then the weather index will be calculated and **`displayLCD()`** function will be called.
  - This function will use the global `door_flag` variable to decide to retrieve temperatures or not. If the door is open, then `door_flag = 1`, so the function will not retrieve any data from the website.

  (Note: Because the function is directly polling the website, if the website is slow then the function will also take time to move on to **`displayLCD()`**)

- **`def checkDoor(channel):`**
  - Controlled by the green button.
  - Uses the global `door_flag` variable
    - `door_flag = 0` indicates that the door is closed
    - `door_flag = 1` indicates it is open.

        - If the button is first pushed when door_flag is 0
            - Raises the flag and display on the LCD that the "DOOR/WINDOW OPEN".
        - Otherwise, if the door is already open and the button is pushed again, the flag will be put down and the LCD will display "DOOR CLOSED".
- **`def ambientLightaControl(channel):`**
    - controlled by the PIR motion sensor event detector declared in **`main`**
    - if the sensor detects something then the green LED will be turned on and the `ambient_flag` will be set to 1 for the **`displayLCD()`** function
    - Otherwise a thread for the **`timer()`** function will start
- **`def timer():`**
    - threaded function where it will start if no motion is detected by the PIR
    - makes sure that the moment the PIR stops detecting movement then the timer for 10 seconds will start
    - if 10 seconds has passed, then the green LED will turn off and the global `ambient_flag` variable will be set to 0 for the **`displayLCD()`** function
- **`def displayLCD():`**
    - uses global flag variables to decide what to print
    - For example, if `ambient_flag` is set to 0 then the lights are "OFF" so it will display it on the LCD
    - Converts float values into string and also displays that onto the LCD
    - This function displays the home screen for the program
- **`def heater(channel):`**
    - Controlled by the red button
    - Only goes into the function if the door is initially closed
    - When the button is pressed, the `desired_temp` will increase by 1 up to a limit of 85
    - Also checks if the `weather_index` is greater than or equal to the `desired_temp` + 3.
        - If it is, then the blue LED will turn on, the AC flag will be raised and the LCD will display "HVAC AC ON".
            - Uses mutex to ensure that it does not use the previous state of the AC and display the wrong state on the LCD

        - If weather index is less than or equal to the `desired_temp` – 3 then the red LED will turn on and the heater flag will be raised
            - Displays “HVAC HEATER ON” and uses mutex to ensure that it doesn’t use the previous state of the heater and display wrong things on the LCD
        - Else, if the weather index is within the range of 3 degrees above and below the `desired_temp`, then both LEDs are turned off and the `hvac_flag` is set to 0.
            - If the AC was on previously, then it will display “HVAC AC OFF” using mutex
            - If the heater was on previously, it will display “HVAC HEATER OFF” using mutex

- **`def AC(channel):`**
    - just like the heater function, where it checks if the `weather_index` is in range of the `desired_temp` and prints accordingly on the LCD if AC/HEATER is ON.
    - Controlled by the blue button
    - If the `desired_temp` is not below 65, then decrease the `desired_temp` by 1 per blue button push

- **`def cleanup():`**
    - Called at the very end of the program
    - Clears the LCD, turns of LEDs
    - Calls **`GPIO.cleanup()`**
- **`if __name__ == “__main__”:`**
    - initially calls **`setupGPIO()`**
    - Adds four event detectors
        - PIR Sensor
            - Callback goes to **`ambientLightControl`** function
        - Green button
            - Callback goes to **`checkDoor`** function to open/close the door

        - Blue button
            - Callback goes to **`AC`** function to decrease `desired_temp` and check the range between `weather_index` and `desired_temp`
        - Red button
            - Callback goes to **`heater`** function to increase `desired_temp` and check the range between `weather_index` and `desired_temp`
    - Has two events
        - `event()`
            - declared for the PIR sensor timer function
            - sets the event as soon as the PIR stops detecting movement
        - `temp_event()`
            - set for the **`checkTemp`** function
            - keeps running the checkTemp function as long as the event is set
    - `starting_hour`
        - **VERY IMPORTANT** variable for obtaining the data from the CIMIS file
        - Captures the current hour time and subtracts by 2
        - This way, the CIMIS can grab the data from 2 hours ago
            - Easier for testing
    - Thread created for **`checkTemp`** function to keep running in the background.

`drivers directory`

This directory is a third party directory provided by the raspberry-pi-guy. Simply used to display strings onto the LCD display by indicating which line (line 1 or 2).

LCD Screens During Program Runtime

Home Screen (image1 on the project pdf):

- Weather Index/Desired temperature – displayed at the top left portion of the LCD. Temperature is obtained by the DHT-11 and humidity is retrieved from the CIMIS website. Weather index is calculated with the following equation: weather index = temperature + 0.05 * humidity
- D: SAFE/OPEN – security system displayed at the top right section of LCD. Shows the state of the door if it is open (OPEN) or closed (SAFE).
- H:OFF/AC/HEATER – displayed on the bottom left of the LCD. Indicates the state of the AC and heater. Displayed on the LCD as "H:OFF" if neither are on, "H:HEATER" if the heater is on, and "H:AC" if the AC is on.
- L:OFF/ON – displayed at the bottom right of the LCD. Indicates if the sensor detected movement or not by displaying "ON" or "OFF" respectively.

Fig. 1 Home Screen (Neutral Status)

Security System Screen:

- Displayed on the LCD for 3 seconds whenever the green button is pushed
- If the door is opened then it will display "DOOR/WINDOW OPEN HVAC HALTED"
- If the door is closed then it will display "DOOR CLOSED HVAC START"

Fig 2. Door Open display

Fig. 3 Door Closed display

HVAC Screen

- Displayed for 3 seconds whenever AC or heater turns on
- If AC turns on, displays “HVAC AC ON” and vice versa
- If heater turns on, displays “HVAC HEATER ON” and vice versa

Fig. 4 HVAC Heater On display

Fig. 5 HVAC Heater Off display

Fig 6. HVAC AC on display

Fig. 7 HVAC AC off display

References

CIMIS Website:

https://www.cimis.water.ca.gov

LCD Setup Tutorial:

https://www.youtube.com/watch?v=3XLjVChVgec

CIMIS Learning how to create the URL:

https://cimis.water.ca.gov/

Setting up PIR and DHT11

http://www.freenove.com/tutorial.html