Project 3
Temperature Alarm

University at Buffalo

2021 Fall

CSE321 – Real Time Embedded System

Author: Ray Chen

rchen63@buffalo.edu

Data: 12/05/2021

Table of Contents

Introduction:

In this project, a temperature alarm is designed and implemented. The user can use the 4x4 matrix keypad to set a temperature range, and 1602 LCD is used to display the temperature range and the current temperature, if the measured temperature by the DHT11 sensor is out of the temperature range, the buzzer alerts.

Specifications:

- User can use matrix keypad to input the temperature range.
- Current temperature will be displayed on the LCD screen.
- The default temperature range is 10 ℃ to 35 ℃.
- The valid temperature range is between 0 ℃ and 50 ℃.
- The lower bound of the temperature range should be lower than the upper bound of the temperature range.
- If the temperature is not in the set range, the buzzer outputs sound.
- If the buzzer alarms, user can press C on the matrix keypad to stop the buzzer for 1 minute. After 1 minute, if the current temperature is still out of rang, the buzzer keeps alerting.
- The LCD displays *Range:* followed by the set temperature range on the top row and displays *Current Temp:* followed by the current temperature in Celsius on the bottom row.
- When inputting time, *Enter new range:* is displayed on the top row of LCD and the user can input the range on the second row of the LCD.
- The alarm runs “forever”

Features:

- Fast response time of the keypad
- Easy to setup.
- 0 ℃ to 50 ℃. Temperature range
- Low power consumption
- Watchdog is set up to reset the system in the case of system failures or malfunctions.
- EventQueue is used for critical section protection
- Mutex is used for synchronization
- Multiple threads are used to allow the different peripherals to run at the same time.

Constraints:

- The temperature range can only be between 0 ℃ and 50 ℃ because the DHT11 sensor can only measure temperatures in that range.
- It can only be used to measure the air temperature.
- Temperature measurement error is ±2 ℃.
- It takes time to measure the current temperature.

Applications:

The user can input a range of temperature that is suitable for the environment, if the temperature is higher or lower than the range, the buzzer will output sound to notify people around it. For example, the temperature alarm can be placed in a bedroom, the user can set a range of room temperature between 10 ℃ and 27 ℃, if the room temperature is not in that range, the buzzer outputs sound to notify the user. It can be also placed in a green house, if a certain plant can live in a range between 20 ℃ and 35 ℃, the user can set that range in the temperature alarm, and if the temperature is not in that range, the buzzer alerts to notify people around.

Project Requirements:

This project is fit in the area of safety. The purpose of the temperature alarm is to notify people that the temperature of the environment is now lower or higher than the set temperature range.

Explanation of how the required internal features are integrated:

- **Watchdog:**

Watchdog is set up to reset the system in the case of system failures or malfunctions. If the system runs properly, the watchdog is being 'kicked' constantly to stay awake. If the system runs into a deadlock or when the user is not entering any key while inputting for 15 seconds, the system reset to the default state.

- **Synchronization:**

Mutex is used for synchronization to protects the shared resource by locking it from the other threads. In the temperature alarm, mutex is used to when the data is sent to the LCD for print, therefore only one thread can print text on the LCD. If two threads send data to LCD together, it will make the data distorted and cause the LCD to print garbled string.

- **Bitwise driver control:**

Bitwise driver control is used to control the output of rows of the keypad. Bitwise driver control is used instead of API because there are four rows of keys, bitwise masking can change the power of all four row by changing the GPIO once and API need to change the GPIO for four times, which waste time.

- **Critical section protection:**

Critical section protection can avoid the situation that multiple functions modify the same memory at the same time. In the temperature alarm system, the EventQueue is used for critical section protection. EventQueue is set up to line up the interrupts so that multiple interrupts do not happen at the same time. There are four interrupts that is used to read from the keypad and change the row and column that read from the keypad, if multiple interrupt happen, the row and column might be wrong. EventQueue makes the interrupt run once at a time, so that they don't change row and column at the same time.

- **Threads/Tasks:**

Multi-thread allows different tasks to run at the same time, also called parallel tasks. In the project, 4 extra threads are created. Two threads are used for the 4x4 matrix keypad, one is used for the DHT11 temperature humidity sensor and the last one is used for the buzzer. The two threads for the keypad are called polling and interrupts, thread polling changes the power of the rows of the keypad and thread interrupts is used for interrupts triggered by the key pressed. Thread readTemp is used to read the current temperature and display the temperature on the LCD. Thread buzzerIO is used to monitor the current temperature and if current temperature if out of range, change the state of the buzzer. main() runs in its own thread in the OS, it sets up the whole system and do different function when keys are pressed.

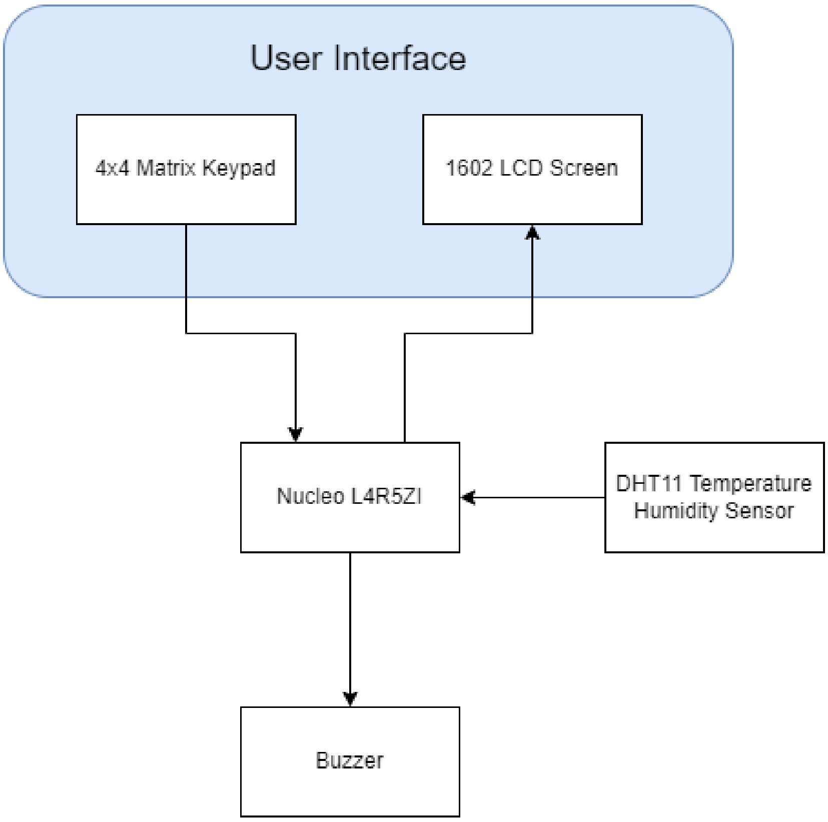
- **Interrupts:**

Interrupt is used to trigger an event when a digital input pin changes. In this project, four interrupts are created to react to the four columns of the matrix keypad. All four interrupts are rising edge triggered. Each interrupt has a corresponding column and a corresponding ISR. When a key is pressed, the corresponding interrupt is triggered to change the row and column so that the main thread can perform different functions.

Solution development:

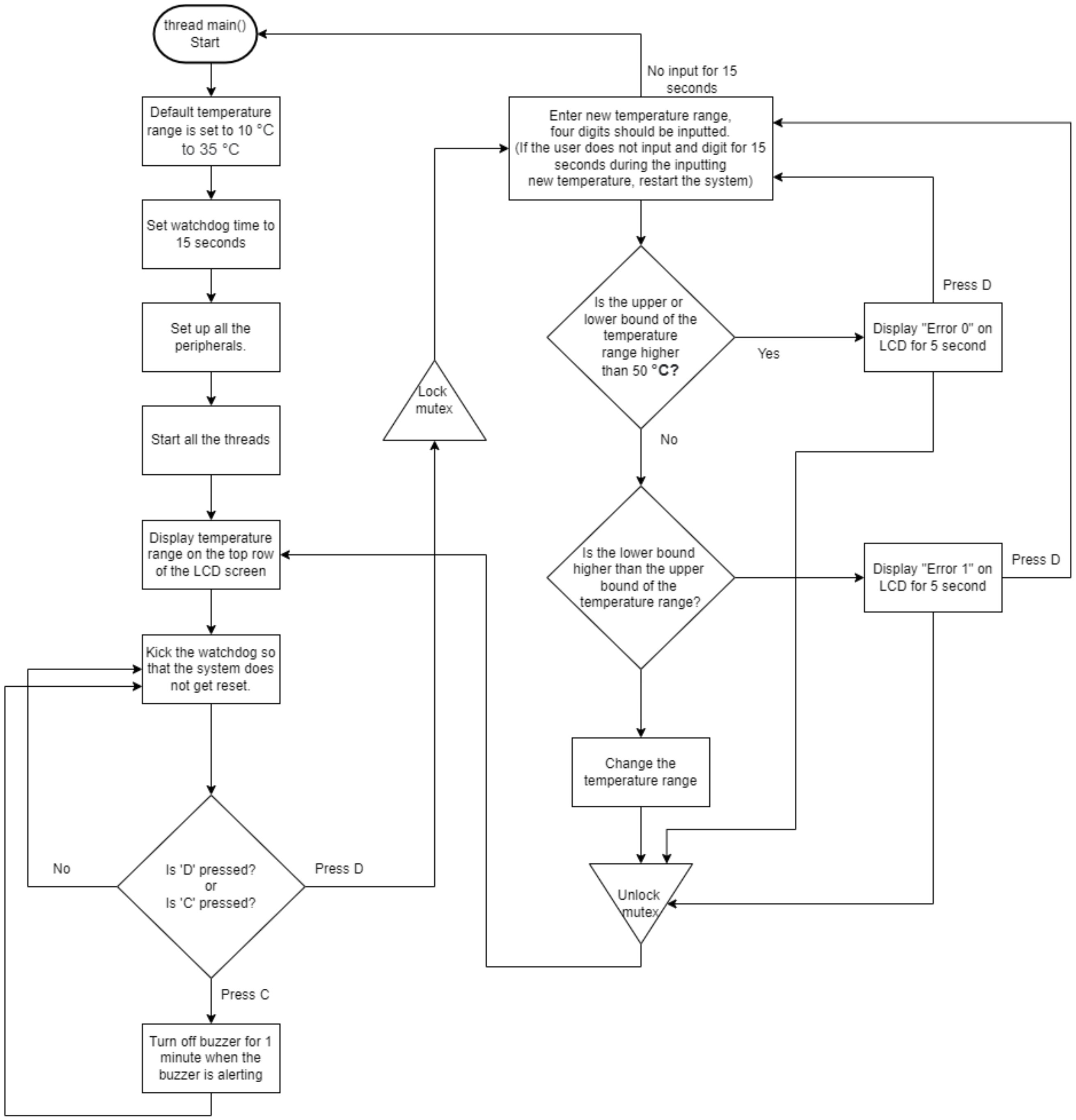
The final design is similar to the initial design, the only difference is the function of the buzzer. When the temperature alarm is set up in a new environment with a new temperature, the sensor takes time to measure the accurate temperature. If the current temperature is out of the set temperature range while the sensor is still trying to measure the accurate temperature, the buzzer would keep alerting until the temperature falls within the range. There should be a way to stop the buzzer, I design that key ‘C’ on the keypad is use to stop the buzzer for 1 minute because the sensor takes at most 1 minute to measure the correct temperature.

Block Diagram:

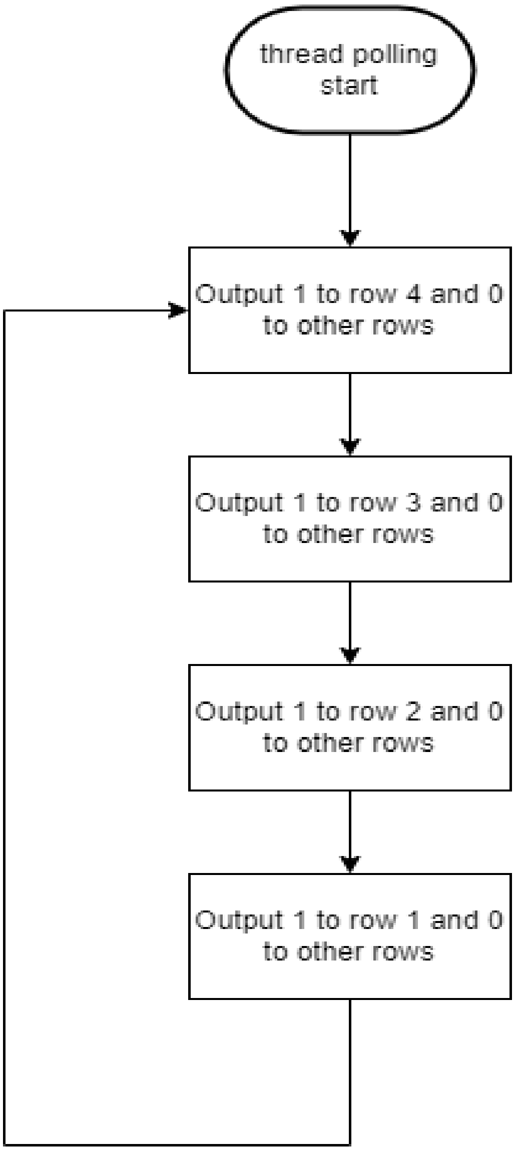


Functionality Diagram:

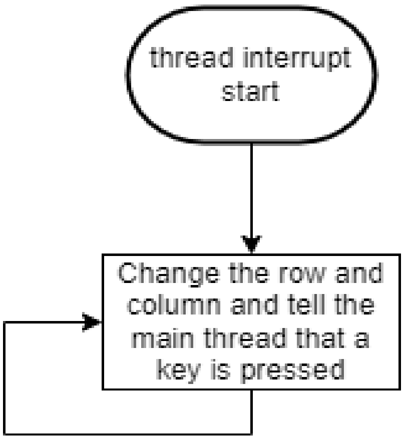
- **Thread Main():**


thread main()
Start
Default temperature range is set to 10 °C to 35 °C
Set watchdog time to 15 seconds
Set up all the peripherals.
Start all the threads
Display temperature range on the top row of the LCD screen
Kick the watchdog so that the system does not get reset.
Is 'D' pressed? or Is 'C' pressed?
No
Press D
Press C
Turn off buzzer for 1 minute when the buzzer is alerting
Lock mutex
No input for 15 seconds
Enter new temperature range, four digits should be inputted. (If the user does not input and digit for 15 seconds during the inputting new temperature, restart the system)
Is the upper or lower bound of the temperature range higher than 50 °C?
Yes
No
Display "Error 0" on LCD for 5 second
Press D
Is the lower bound higher than the upper bound of the temperature range?
Display "Error 1" on LCD for 5 second
Press D
Change the temperature range
Unlock mutex

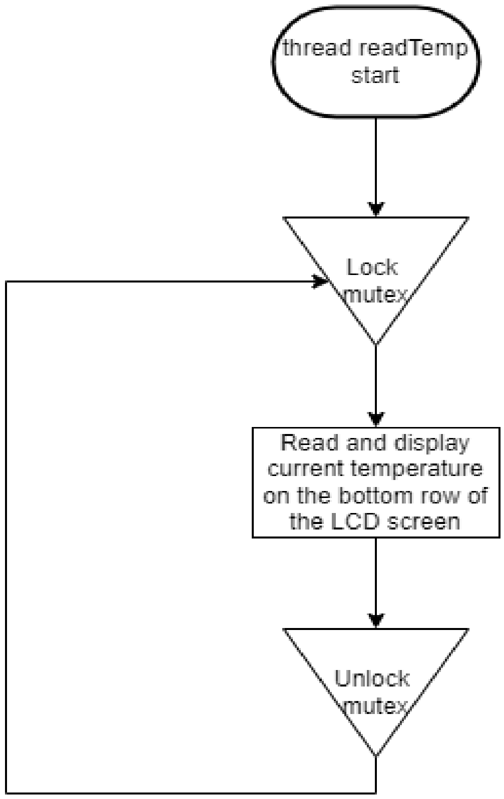
- **Thread polling:**


thread polling start
Output 1 to row 4 and 0 to other rows
Output 1 to row 3 and 0 to other rows
Output 1 to row 2 and 0 to other rows
Output 1 to row 1 and 0 to other rows

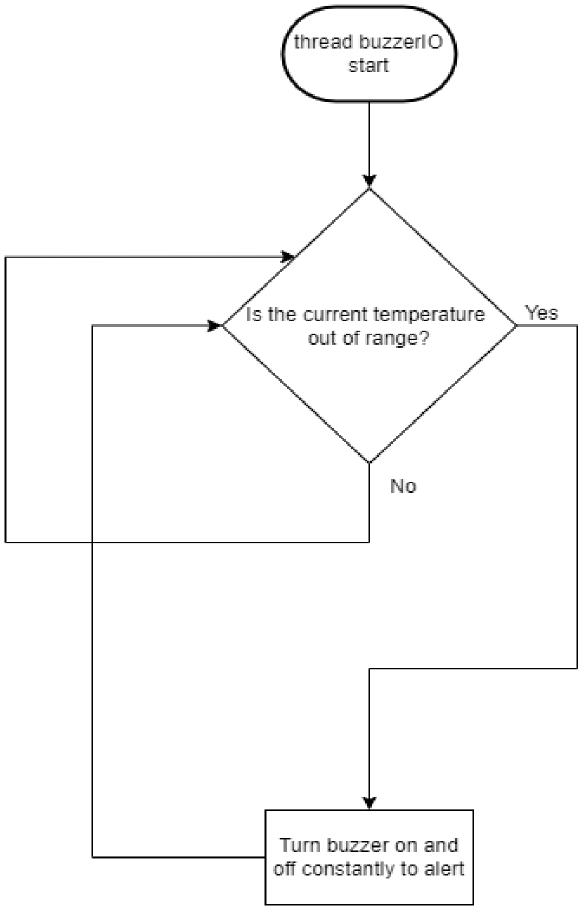
- **Thread interrupts:**


thread interrupt start
Change the row and column and tell the main thread that a key is pressed

- **Thread readTemp:**


thread readTemp start
Lock mutex
Read and display current temperature on the bottom row of the LCD screen
Unlock mutex

- **Thread buzzerIO:**

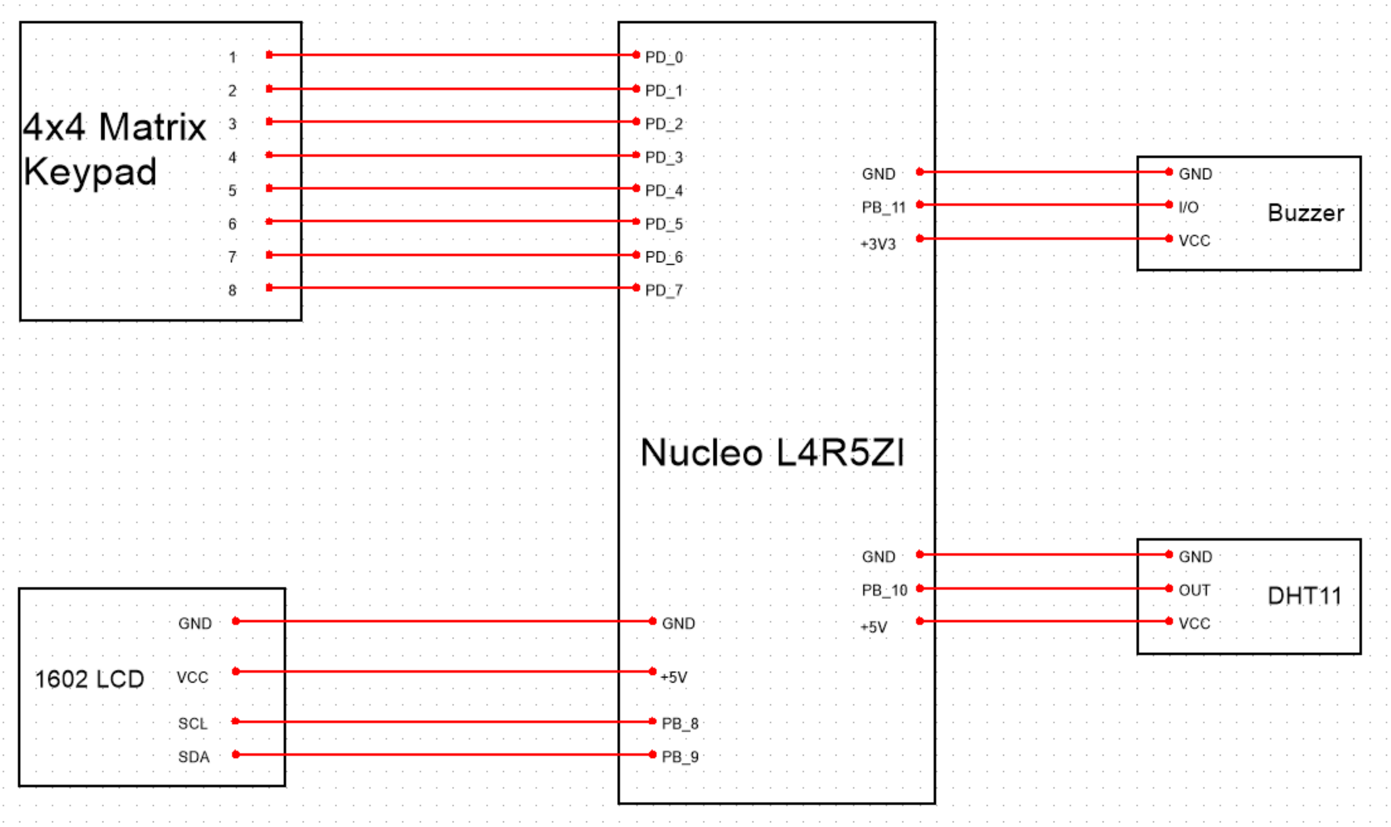

thread buzzerIO start
Is the current temperature out of range?
Yes
No
Turn buzzer on and off constantly to alert

Bill of Material:

- DHT11 temperature humidity sensor
- 4x4 membrane matrix keypad
- 1602 LCD screen
- ARCELI Passive Low Level Trigger Buzzer
- Nucleo L4R5ZI
- Jumper wires
    - At least 10 Male to Female jumper wires
    - At least 20 Male to Male jumper wires
- Breadboard.

User instructions:

Schematic:



Instruction to build

1. Mbed Studio is needed to compile the codes into Nucleo L4R5ZI.
2. Create an empty Mbed OS program.
3. Download the files listed below into the program folder
    - CSE321_project3_rchen63_main.cpp
    - DHT.cpp
    - DHT.h

- ■ lcd1602.cpp
- ■ lcd1602.h
- ■ mbed_app.json

4. Connect Nucleo L4R5ZI to the computer.
5. Click “Run program” to compile the code into Nucleo L4R5ZI.
6. Connect the 4x4 membrane matrix keypad directly to Nucleo L4R5ZI
   - ■ Connect pin 8 to PB_7
   - ■ Connect pin 7 to PB_6
   - ■ Connect pin 6 to PB_5
   - ■ Connect pin 5 to PB_4
   - ■ Connect pin 4 to PB_3
   - ■ Connect pin 3 to PB_2
   - ■ Connect pin 2 to PB_1
   - ■ Connect pin 1 to PB_0
7. Connect +5V to the power rail with ‘+’ sign on the breadboard, the rail is called VCC rail.
8. Connect GND to the power rail with ‘-’ sign on the breadboard, the rail is called GND rail.
9. Connect PB_8, PB_9, PB_10, PB 11 and +3V3 each to a different column
10. Buzzer
    - ■ Connect GND to the GND rail
    - ■ Connect VCC to the column with +3V3
    - ■ Connect I/O to the column with PB_11
11. DHT11
    - ■ Connect GND to the GND rail
    - ■ Connect VCC to the VCC rail
    - ■ Connect OUT to the column with PB_10
12. 1602 LCD
    - ■ Connect GND to the GND rail
    - ■ Connect VCC to the VCC rail
    - ■ Connect SDA to the column with PB_9
    - ■ Connect SCL to the column with PB_8

Instruction to use

- ● Press ‘D’ to set a new temperature range.
- ● Press ‘C’ to turn off the buzzer for 1 minute.
- ● The temperature is displayed and measured in degree Celsius.
- ● Do not enter a lower bound or upper bound higher than 50 °C, or “Error 0” will be display for 5 seconds.
- ● Do not enter a lower bound higher than upper bound, or “Error 1” will be display for 5 seconds.
- ● The DHT11 sensor can only measure temperature range between 0 °C and 50 °C
- ● If current temperature is out of range, the buzzer is turned on and off periodically at 0.5 second intervals.

Test Plan

1. Press ‘D’, LCD should display “Enter new range”

2. Set a temperature range so that the current temperature is not in that range, buzzer should be turn on and off periodically at a 0.5 second intervals.
3. While the buzzer is alerting, press ‘C’, the buzzer should stop alerting, and then wait for 1 minute, the buzzer should start alerting again.
4. Take the system to an environment with different temperature, the current temperature should be changing.
5. Input a new temperature range, input a lower bound higher than the upper bound, the LCD should display “Error 1”, and then wait for 5 second, the LCD should return to display the previous temperature range and current temperature.
6. Input a new temperature range, input a lower bound or upper bound higher than 50 C, the LCD should display “Error 0”, and then wait for 5 second, the LCD should return to display the previous temperature range and current temperature.
7. Press ‘A’, ‘B’, ‘#’ and ‘*’ on the keypad, nothing should happen.
8. Press any key other than numerical keys while inputting a new temperature range, nothing should happen.
9. While inputting new temperature range, do not press any numerical key for 15 second, the system should reset to default, the default temperature range should be 10 °C to 35 °C

Outcome of implementation

1. After pressing ‘D’, the LCD displayed “Enter new range”.
2. When current temperature is out of range, buzzer alerted.
3. When buzzer was alerting, pressing ‘C’ stopped the buzzer for 1 minute.
4. The current temperature changed when in different environment
5. Inputting a lower bound higher than the upper bound caused LCD to display “Error 1”, and then it displayed temperature range and current temperature after 5 seconds
6. Inputting a lower bound and upper bound caused LCD to display “Error 1”, and then it displayed temperature range and current temperature after 5 seconds
7. Nothing happened when ‘A’, ‘B’, ‘#’ and ‘*’ were pressed
8. Nothing happened when none-numerical keys were pressed while inputting new temperature range
9. The system reset by itself by not entering any numerical keys for 15 seconds while inputting new range.

The outcome of implementation is the same as the project design and all the tests passed as expected.

Future Considerations:

- Now the temperature can only be displayed and measured in degree Celsius, for future improvement, user can press a key to change the unit to Fahrenheit degree.
- DHT11 can also measure humidity of the environment, the system can also be implemented to display the current humidity.