CSE 321

Real Time and Embedded Operating Systems

Project **3, Fall 2022**

Brian Leavell

University at Buffalo
Department of Computer Science and Engineering
School of Engineering and Applied Sciences

Table of Contents

Project Overview

It's time to save the world! To do this, this project utilizes several techniques that have been developed throughout the semester: Interrupts, threads, ISRs, and bitwise control

New techniques have also been implemented: Watchdog timer, thread synchronization, and critical section protection

Project Purpose: Environmental monitoring system for Greenhouses and other plant life facilities

- The user can set specific ranges of temperature and humidity based on what is being monitored
- The system will read the current levels and have specific LEDs light up based on the conditions
- A matrix keypad is used with an LCD to display the current temperature/humidity
- A dot matrix array will display the conditions as "Ideal", "Moderate", or "Bad" to the user

Real-World Applications

- Botanical Gardens
- Research Facilities
- Grade school Science Classes (Small projects)
- Farms (Crop Yield Monitoring)

Safety Concerns

Main Issues To Be Addressed: **Environmental Protection**

Constant expansion of cities have put many valuable plants in danger of extinction

Sweetbay Magnolia

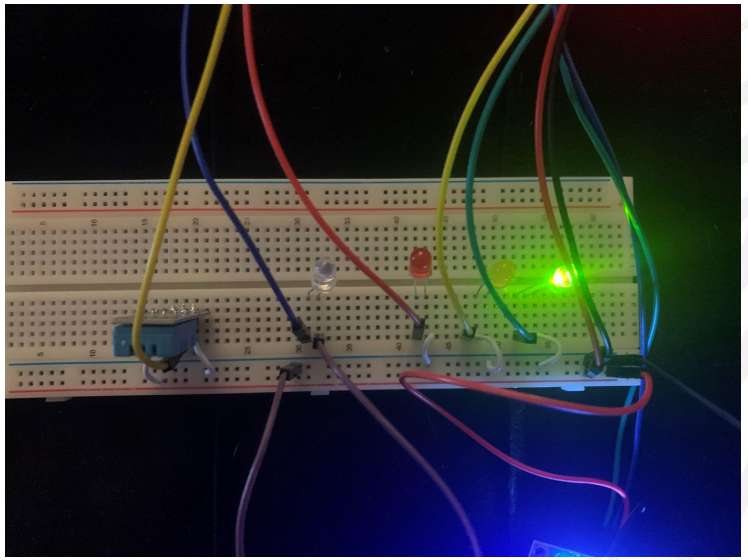
Goldenseal

BOM

- Resistors
- Matrix Keypad
- Dot Matrix Array
- 1602 LCD Display Screen
- DHT11 Sensor
- Nucleo L4R5ZI
- MBed Studio
- 4 LEDs (Red, Yellow, Green Required)
- Dot matrix array
- Breadboard
- Wiring Kit

Features


System Running
Humidity(%): 50
Temp(F): 62.5999




PLAY
www.icstation.com

Watchdog Timer

What is a WatchDog?

It is responsible for recovering your program after a malfunction

How to use it?

1.) Give the timer a time limit to count down from
2.) If the timer hits zero (aka the system malfunctions for example), the Watchdog resets the system
3.) Kick (Feed to be more PC) the dog whenever the program does what it is supposed to do (This will reset the timer)

How was it used in this system?

LED Status Function verifies the temperature sensor is working and currently taking in data

If conditions are not met, the timer is not reset, therefore resetting the system

Threads & Synchronization

Threads Used: Main Thread, Thread for the Interrupt

Synchronization Method Used: Event Queue

Thread responsible for the Interrupt starts the Event Queue

Interrupts ".rise" and the ISR that gets called (wanted address) are loaded into the EventQueue

ISR Contains a non-reentrant function (printf), so threads are needed

EventQueue ensures elements in the program are safely executed in the required order

Bitwise Control

Using AND and OR Logic to set specific pins to be “outputs”

Pins used for connecting the matrix keypad to the Nucleo

Example Code designed by:
Brian Leavell

<u>Setup:</u>

1.) Activate the bus (clock) for the port being used

2.) Set specific bits to be 1 or 0 depending on which pin is being used

|= for OR logic, only changing one bit to 1 without affecting others

&=~ for AND Logic to set all bits to a 0

```
//activating bus for port c for columns
RCC -> AHB2ENR |= 0x4;

//activating 4 pins for inputs PC8-11
//PC8
GPIOC -> MODER |= 0x10000; //sets 1
GPIOC -> MODER &=~ (0x20000); //sets 0
//PC9
GPIOC -> MODER |= 0x40000;
GPIOC -> MODER &=~ (0x80000);
//PC10
GPIOC -> MODER |= 0x100000;
GPIOC -> MODER &=~ (0x200000);
//PC11
GPIOC -> MODER |= 0x400000;
GPIOC -> MODER &=~ (0x800000);
```

Critical Section Protection

What is the Critical Section?

Memory that is being shared among multiple other threads

The Solution: Mutexes

When a thread attempts to access shared memory, its mutex locks and prevents other threads from entering

When it is done, the mutex unlocks other threads can now access the memory\

How it was implemented:

When the temperature/humidity functions are called, the mutex **LOCKS**

This is due to the function accessing the DHT object that holds the temperature/humidity values

After the LCD was updated and the object no longer needs to be accessed, mutex **UNLOCKS**

Interrupts

What is an Interrupt?

Essentially, a triggered action that causes something to happen (Hardware or software)

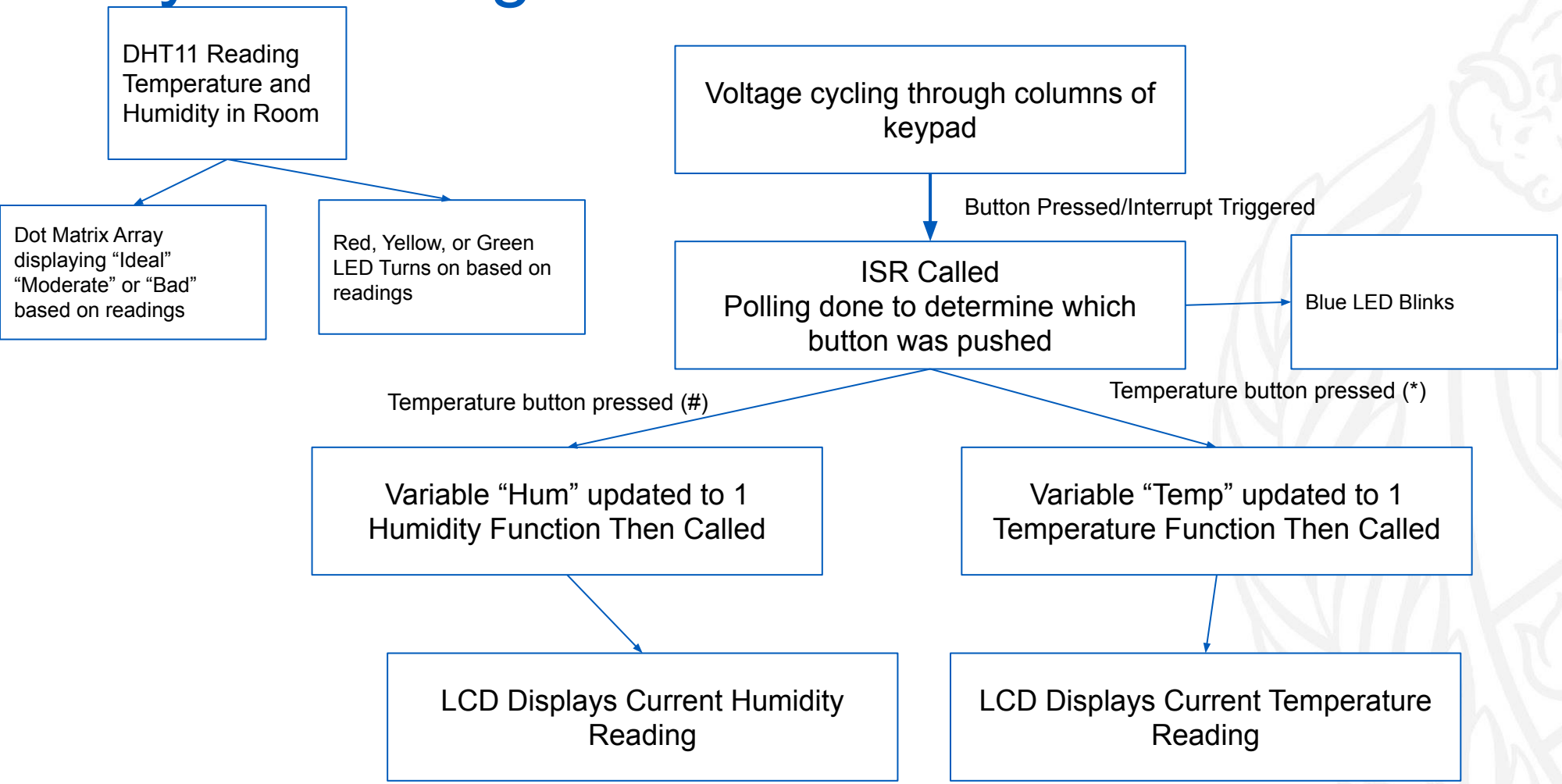
Unlike polling, this action only occurs once this trigger is detected

Interrupt then calls an ISR (A function) to execute some behavior

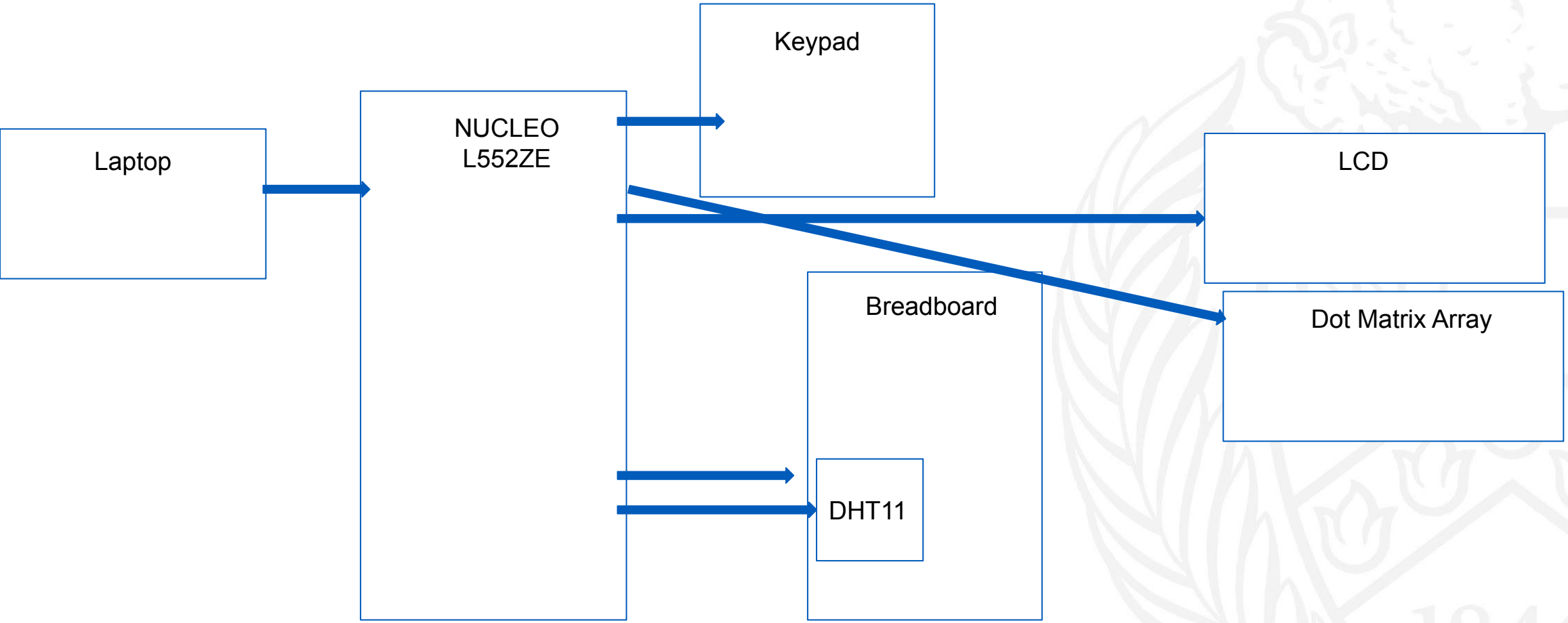
System Implementation

1.) A button is pushed on the matrix keypad
2.) The interrupt detects a rising edge, and then calls the ISR
3.) ISR uses AND logic to find which button was pushed on the keypad

System Diagram



Block Diagram



Instructions

1.) Connect the 5V and GND pins from the Nucleo into their dedicated rails on the solderless breadboard

2.) Only one row is being used on the keypad, so connect port 4 (from left to right) on the keypad to pin D6 on the Nucleo

3.) Although only two columns are being “checked”, connect ports 5-8 from the keypad into pins C 8,9,10 and 11 of the Nucleo respectively

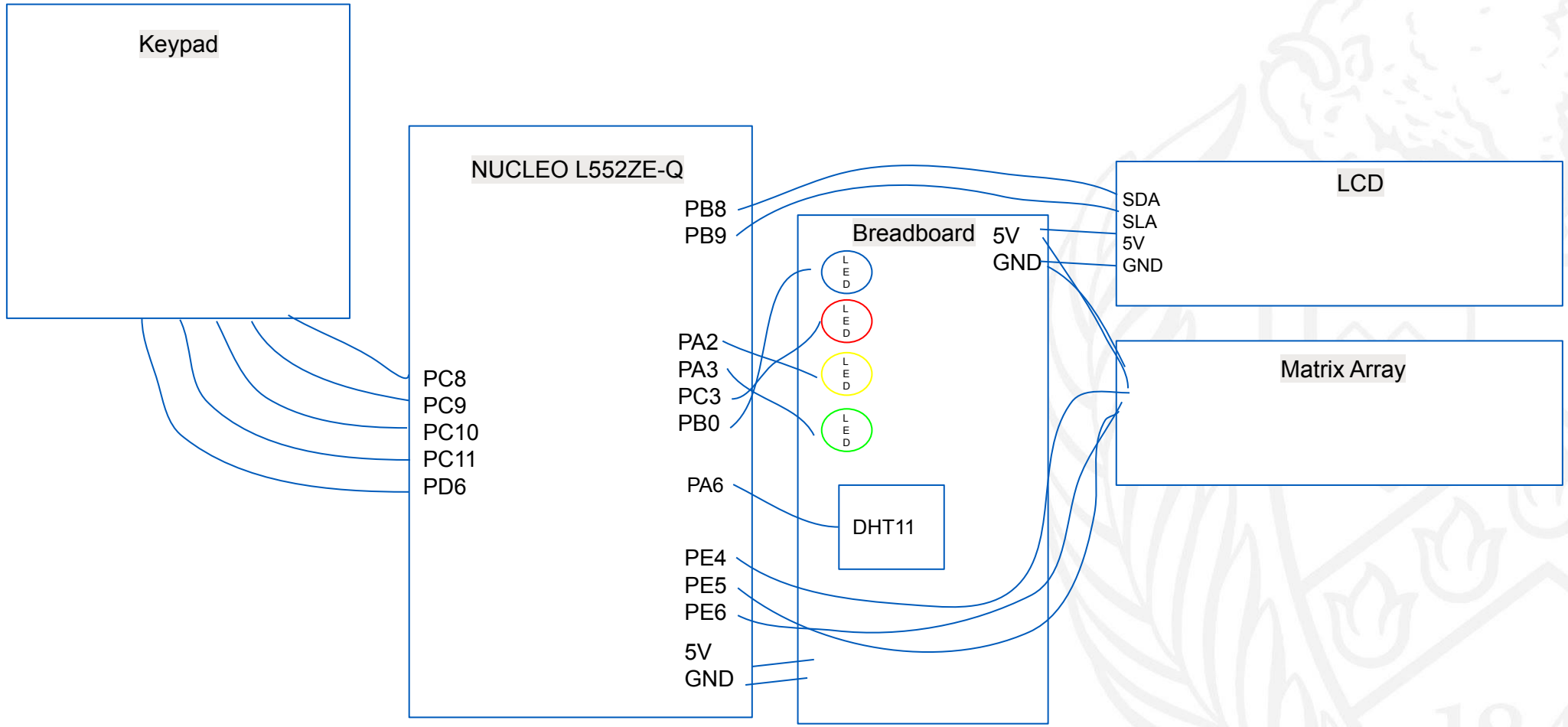
4.) Place the DHT11 anywhere on the breadboard. Connect the left pin to the power rail of the breadboard and the right to ground. Connect a wire from pin A6 of the Nucleo into the column that the middle pin of the DHT11 is connected to

5.) Place four LEDs on the breadboard next to each other (Red, yellow, green, blue). Place a resistor to the positive probe and ground the negative probe. Connect a wire from pin B0 of the Nucleo to the resistor of the blue LED, pin C3 to the resistor of the red LED, pin A2 to the resistor of the yellow LED, and pin A3 to the resistor of the green LED

6.) Connect the 5V and GND pins of the LCD to their respective rails on the breadboard. Connect the SDA and SLA pins to pins B8 and B9 on the Nucleo respectively

7.) Connect the 5V and GND pins of the Dot Matrix Array to their respective rails on the breadboard. Connect the DIN pin to pin E4 of the Nucleo, the CS pin to E5, and the CLK pin to E6

Schematic


Keypad
NUCLEO L552ZE-Q
PC8
PC9
PC10
PC11
PD6
PB8
PB9
PA2
PA3
PC3
PB0
PA6
PE4
PE5
PE6
5V
GND
Breadboard
LED
LED
LED
LED
DHT11
5V
GND
LCD
SDA
SLA
5V
GND
Matrix Array

User Instructions

1.) Correctly wire all components to the NUCLEO
2.) Set the specific variables for temperature and humidity to be their required values depending on what plant is being monitored in the marked section of code

```
//*********************IMPORTANT INFORMATION BELOW*******************************
//vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
//-------USER ADJUST THESE NUMBERS BEFORE RUNNING PROGRAM-------------------------//
double tempGood = 68.0;
double tempMod = 40.0;
double tempBad = 30.0;

double humGood = 40.0;
double humMod = 0.0;
double humBad = 0.0;
```

Look for this section in the code above “main”

3.) Set up the system in a designated spot that can receive accurate temperature and humidity readings that the plants are receiving
4.) Place the Dot Matrix Array in a noticeable place. This ensures the user will be able to see if the environmental conditions are ideal or not
5.) Press * to see current Temperature or # to see current humidity (Printed to the LCD)

Testing & Expected Outcomes

- Once the temperature and humidity values have been set, the system can be powered on and ran
- The LCD should display the message “System Running”
- Once the * button is pushed, the LCD should display the current temperature (In Fahrenheit)
- Once the # button is pushed, the LCD should display the current Humidity Percentage
- The Blue LED will flash whenever one of these two buttons is pressed
- The green, yellow, or red LED should stay lit based on the temperature and humidity readings and if they match the conditions’ ranges
- The dot matrix array should display the message “Ideal”, “Moderate”, or “Bad” depending on which LED is lit up

Future Considerations/Improvements

- Allow the user to activate/deactivate the system
- Have a buzzer go off if the red LED activates (signaling the conditions are dangerous)
- Allow the user to enter the desired temperature and humidity ranges from the keypad rather than the code itself (Shown to user on LCD)
- Find working code for the Dot Matrix Array