nodeMCU-esp8266 & Raspberry Pi Multi Irrigation (PART 1)

Introduction:

Project was requested by a college student seeking to learn and develop an IoT projects. This project consists of 2 nodeMCU esp8266 v1.0 and a raspberry pi zero w as it’s processing units. Aim of the project is to water multiple (3) indoor plant beds all located at different regions of the apartment. Along with watering the plant beds we will also monitor temperature, light intensity, soil moisture levels, security camera and have a remote monitor & control website with user login. Bonus with telegram bot.

Hardware Components:

1. Raspberry Pi Zero W x1
2. 32GB Micro SD x1
3. Raspberry Pi Zero W Case x1 (with camera mount)
4. Amazon Basics micro USB x1
5. Raspberry Pi 5MP Camera x1
6. 12V 5A Power Supply x1
7. nodeMCU esp8266 V1.0 x2
8. Soil Moisture Sensor x6
9. LM35 x3
10. LDR R10k x4
11. HCSR-04 x1
12. 12V Water Pump x3 (can be switched with regular AC Pump (max 240v))
13. 5V Relay x3
14. 74HC4067 Analog MUX x1
15. Wires
16. Prototype PCB 4x4 x2

Software Tools & Technologies:

All this tools/software are available for Windows, Mac and Linux

1. Raspbian Buster OS (you can also use lite version)
2. SSH
3. Python
4. MariaDB
5. Apache2
6. PHPMYadmin
7. PHP 7
8. Remote.it Service
9. Arduino CC
10. easyEDA

Basic Setup:

To setup your Raspberry Pi: [RPi Setup](https://projects.raspberrypi.org/en/projects/raspberry-pi-setting-up/2)

To setup ArduinoCC with nodeMCU: [nodeMCU Setup](https://www.instructables.com/id/Steps-to-Setup-Arduino-IDE-for-NODEMCU-ESP8266-WiF/)

To setup MariaDB server on Raspberry Pi: [Server Setup](https://randomnerdtutorials.com/raspberry-pi-apache-mysql-php-lamp-server/)

Designing the circuit (step 1):

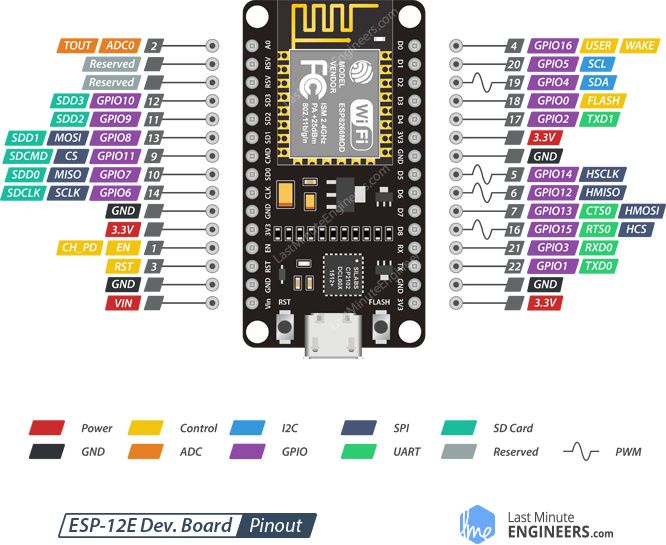
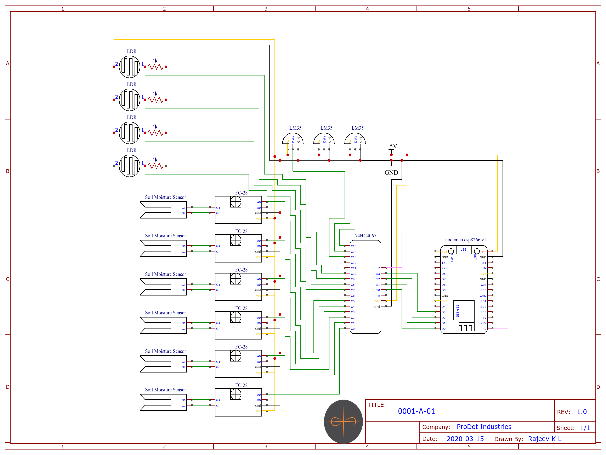
Before designing let us have a look at the nodeMCU esp8266, we notice in ‘Figure – 01’ that it has only one analog pin and we have 13 analog sensors. To connect so many sensors we will use an analog 16:1 multiplexer.

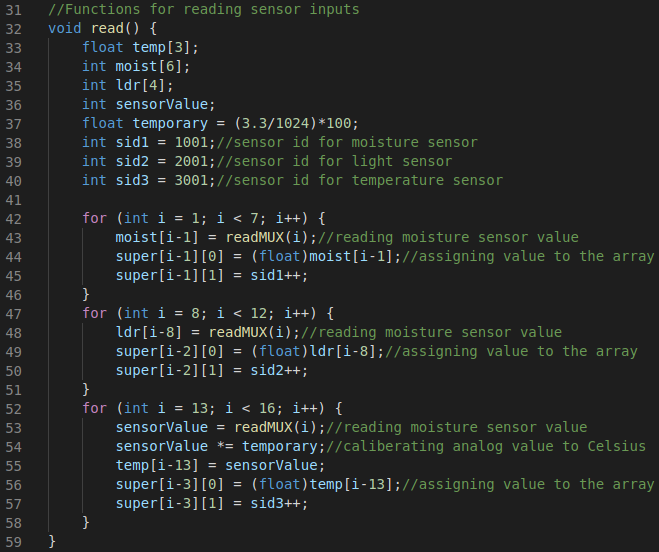
Figure - 01

We will design the circuit in easyEDA. Create a new project. In the libraries search for nodemcu, this should give you a list of existing designs. Select one and make the connections as shown in ‘Circuit Design – 01’. Save the project and build the prototype PCB accordingly.

Circuit Design - 01

Programming the nodemcu (step 2):

Let us program the nodemcu to read the sensors through the multiplexer. Observe in the ‘Code Snippet – 01’, I have used 3 arrays to store values of different sensors. To read the values I am using a ‘readMUX( )’ function which modifies the control pins to access channel pins. The 3 loops are used to traverse through the MUX/DEMUX and read sensor values.

Upon executing this code without connecting any sensors, you should get all sensor values as zero. After connecting all the sensors with the MUX, new set of values will be visible in the serial monitor.

Code Snippet - 01

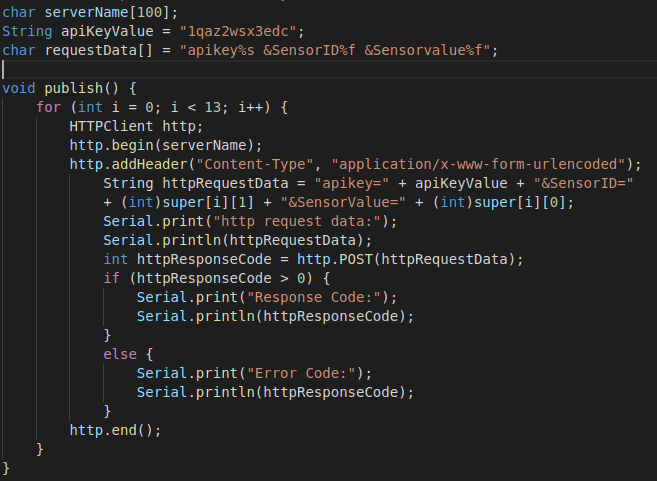
Programming on Raspberry Pi (step 3):

In the raspberry pi, open the phpMyAdmin page to create a MariaDB schema. Create a table ‘EventTable’ with 4 attributes – Event ID, Sensor ID, Sensor Value, EventTime.

Now let us create a php script to accept the values from the nodemcu and insert the data to the table that we created. Create a new file named ‘postEspData.php’. Initialize the server name, DB username and password. To accept values from a client application/device we will use the ‘POST’ request method. To verify that no random user can update data, we will use a key which is known only to our devices and server. Once all the necessary values are acquired, we connect to MariaDB. When the connection is established, we insert the acquired data into the ‘EventTable’.

Code Snippet - 02

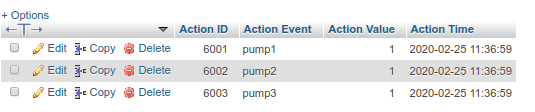
Connecting the nodemcu to Server (step 4):

To connect we need to use WiFiClient and ESP8266WiFi libraries. Start the server connection with http.begin(serverName), define the type of content in the header. Append the ‘Sensor ID’, ‘Sensor Value’ and ‘apiKey’ to the httpRequestData. POST this request data to the server, if the upload is successful, you will get a response code of 200.

Code Snippet - 03

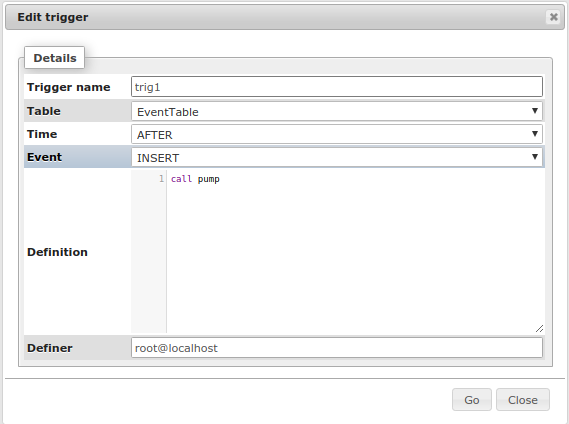
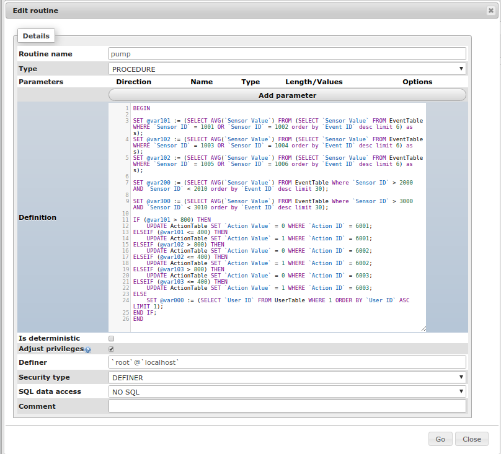
Programming MariaDB (step 5):

Now we will write a procedure to identify low soil moisture levels and high moisture levels. With this we can control the state of the pumps.

First create a new table ‘ActionTable’ with 4 attributes – Action ID, Action Event, Action Value, Action Time. Add 3 records for 3 pumps, set Action Value as 0.

SQL - 01

Now go to ‘EventTable’ and create a trigger with ‘Time’ as ‘AFTER’ and ‘Event’ as ‘INSERT’. In the trigger write ‘call pump()’.

Now let’s create a routine with ‘Type’ as ‘Procedure’. Select the average of last 3 readings of pair of sensors. Using this average value, we will update the ‘ActionTable’ accordingly. I have used lower limit as 400 (min 0) to switch ON the pump and a upper limit of 800 (max 1023) to switch OFF the pump.

SQL – 03

SQL – 02

To verify if this procedure is working, insert values manually into the ‘EventTable’ and check the ‘ActionTable’ for changes.

This should complete our Main Objective of watering 3 separate soil beds when the soil moisture levels are low and automatically turn off after watering.