FLOOD DETECTION AND ALERTING SYSTEM: AN IOT PROJECT

DIY Course Project-1: CS-578 Internet of Things. *Instructor Dr. Manas Khatua*

Group5 Members: 1. Anurag (200102012) 2. Taruna (200102095) 3. Uday (200102097)

Introduction to Project

Flooding is a significant problem in many parts of India. Each monsoon season, considerable loss of life and property is caused by floods in many areas. With timely monitoring of the situation and analysis of the data, we can take preparatory measures and prevent the loss. The remote location and unavailability of experts and computing facilities at the observation site make it challenging to draw valuable insights and make a guess about emerging situations. With the help of IoT systems, such obstacles can be overcome.

This project aims to implement a basic IoT system that senses water level data at the site and sends it to the server. From there, actuators, e.g., dam gates or alarm alerts, can be manipulated to control dangerous situations. This project demonstrates that IoT systems can be helpful for the said purpose, and scalable and more complex systems can be built following the methodologies used here.

Main Objectives of the Project

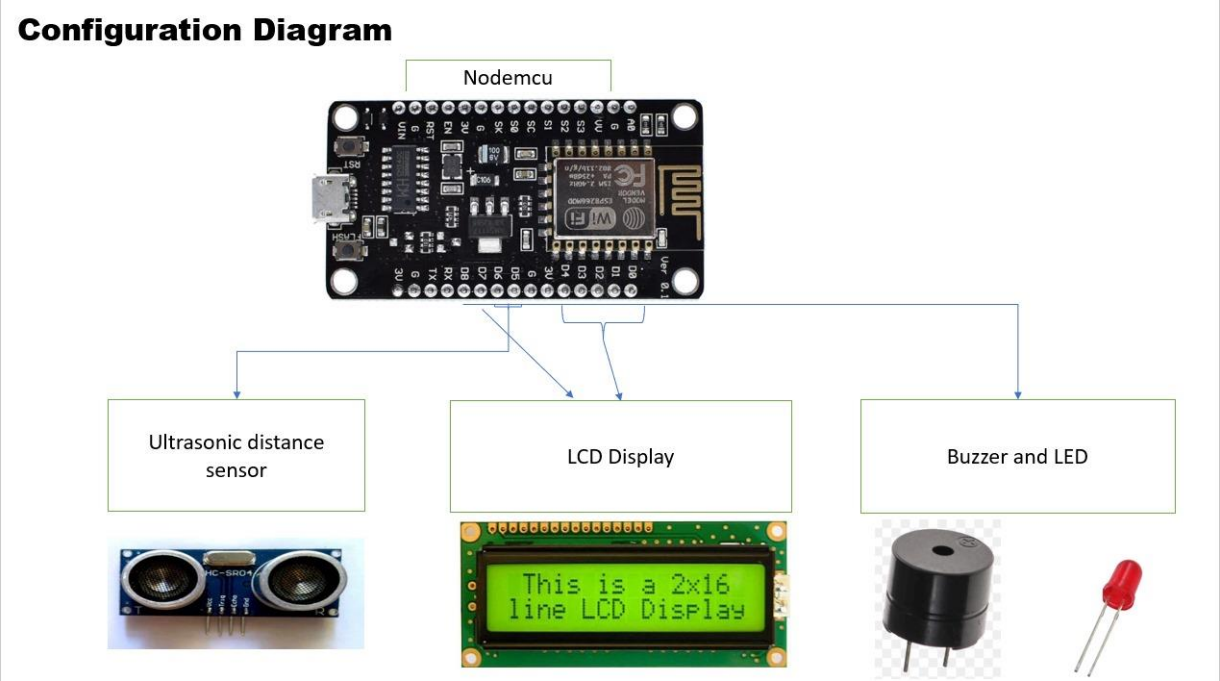
We aim to implement an IoT system to demonstrate its working and to understand the IoT ecosystem better. We design a Flood Detecting System to sense water level and send this data to the ThingSpeak Cloud server. On server we write a script to analyse this data, and based on the water level we send a command to an alarm that alerts people we also implement an actuator as an LED that symbolizes an actuator and in real life scenario it can be replaced by water pump-to drain excess water- or motors-to open or close sluice gate of a dam. We can see data trend using tools available on the ThingSpeak server.

Implemented Attributes

We have implemented the following attributes:

1. Sensing of data at site: Water level is read at site, e.g., reservoir or river.
2. Visualising sensed data on an LCD Screen at the site.
3. Sending data to ThingSpeak channel.
4. Analysing data of ThingSpeak channel and based on that sending command to actuators.
5. Visualisation of data using visualisation tools available on ThingSpeak. We can see time v/s water surface distance value on the server.
6. Actuators acting on commands received from server.

Configuration Diagram:



DATA FLOW DIAGRAMS

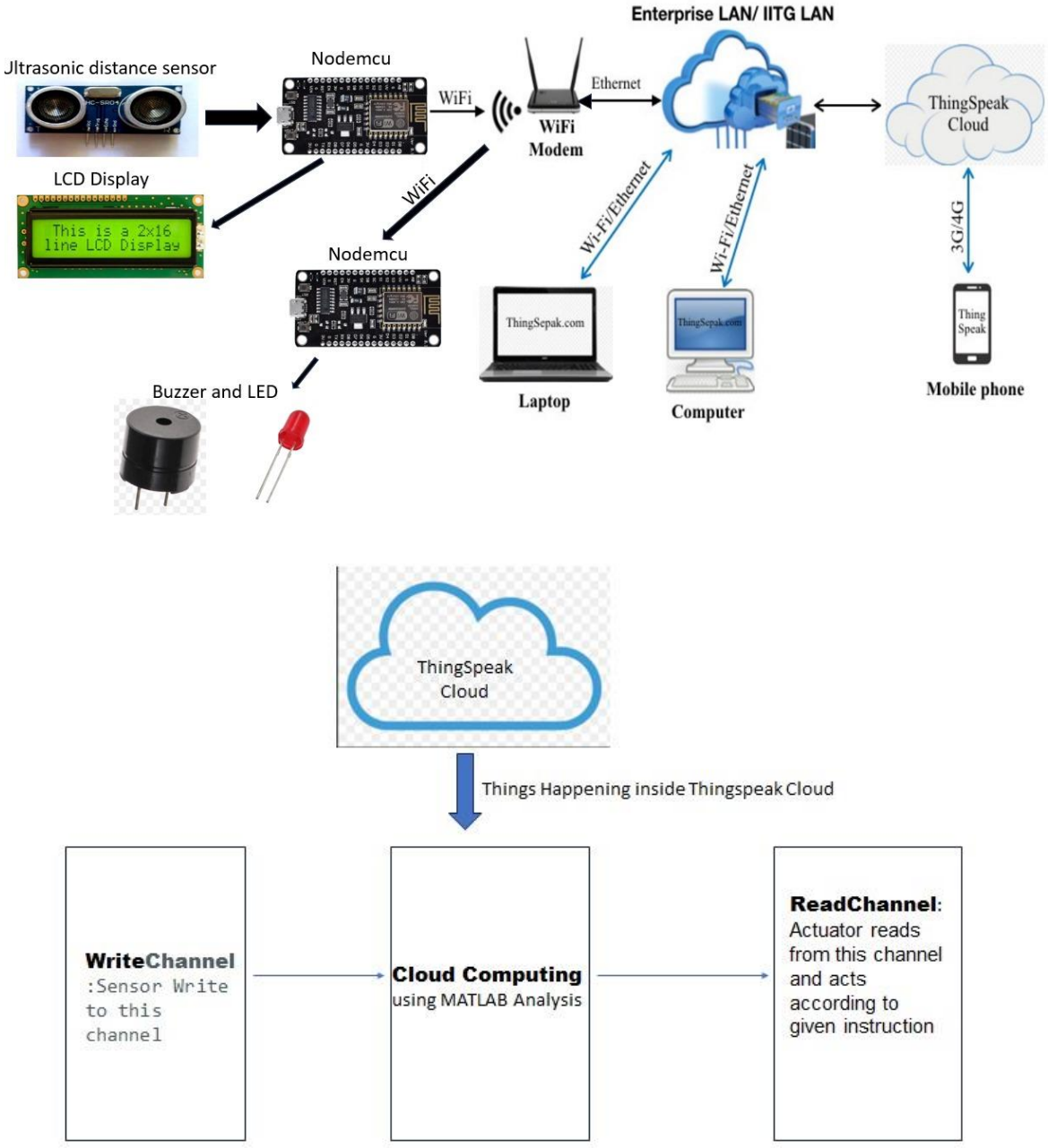


Figure 1: Data flow inside ThingSpeak cloud Server

Sample Outputs

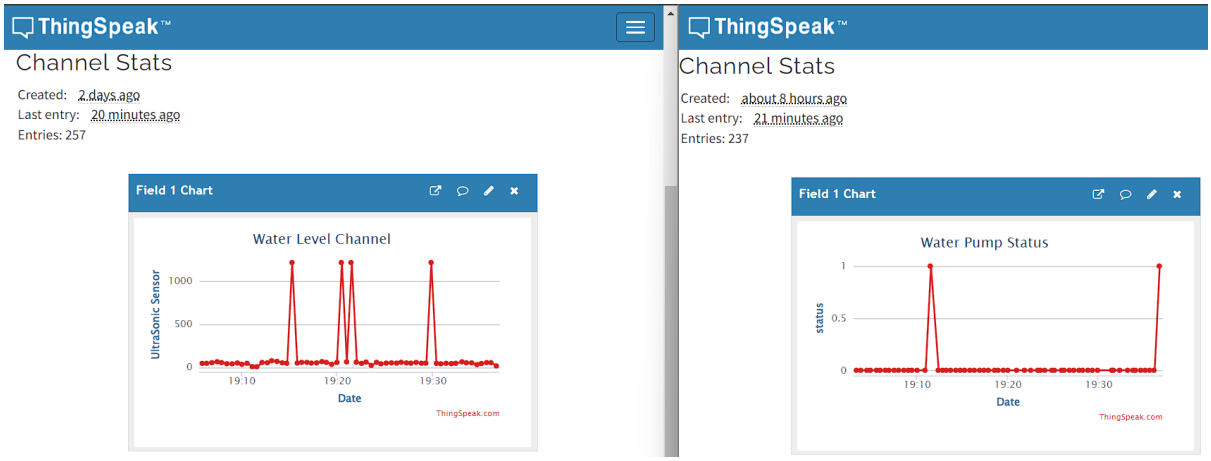


Figure 2 Left to Right: Water Level Data on ThingSpeak Cloud, Commands Given to Actuator from ThingSpeak cloud

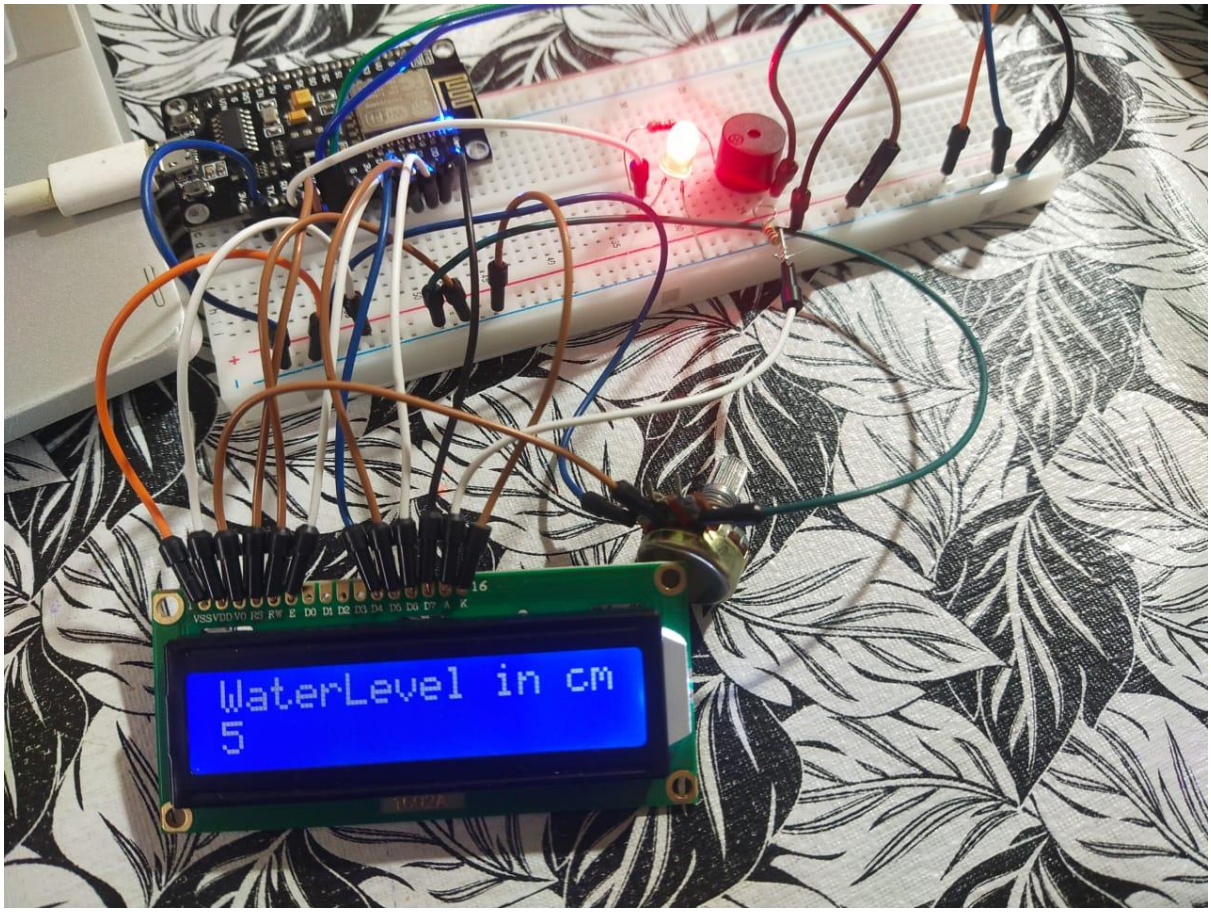


Figure 3: When Water level rises too high then red light glows and alarm buzzes. Note that water level is with respect to sensor placed at the top of dam.

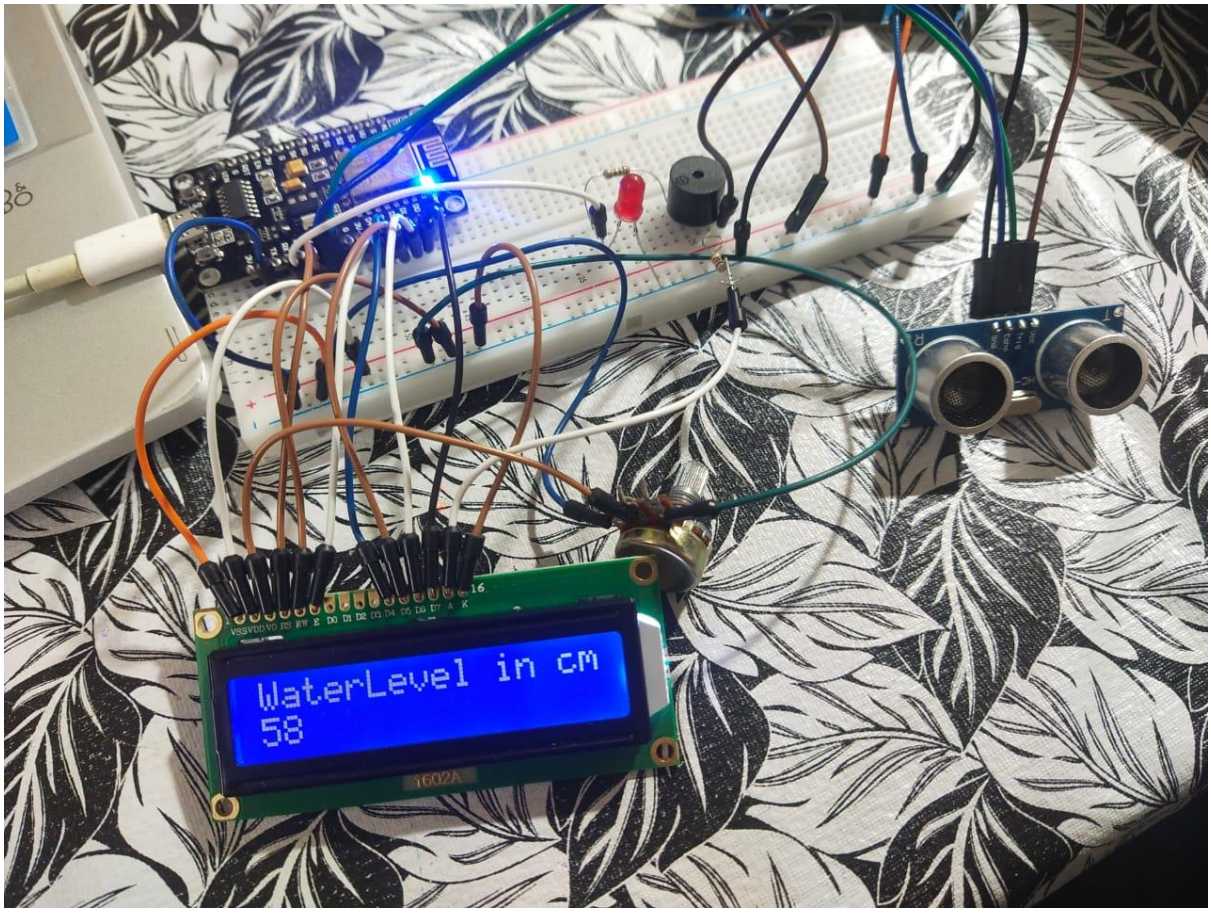


Figure 4: When water surface is far away from sensor light doesn't glow and alarm doesn't buzz.

Programming Code Used

Code for NodeMCU:

```
#include <ESP8266WiFi.h>
#include <ThingSpeak.h>
#include <LiquidCrystal.h>
#include <LiquidCrystal_I2C.h>
// Initialize the LiquidCrystal library with your LCD pin numbers
LiquidCrystal lcd(D7,D4, D3, D2,D1,D0);

#define pingPin D5
#define echoPin D6
#define alarm D8
WiFiClient client;
```

```
long myChannelNumber = 2285158;
const char myWriteAPIKey[] = "ZJ3D1EQW0MI724CJ";
const char* ssid = "Malicious Network";
const char* password = "goodbye!";

long myChannelNumber2 = 2287036;
const char myreadAPIKey[] = "58L7R6HV9F41D9SG";

void setup() {
  Serial.begin(9600);
  delay(1000);
  pinMode(echoPin, INPUT);
  pinMode(pingPin, OUTPUT);
  pinMode(alarm, OUTPUT);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(200);
    Serial.print(".");
  }
  Serial.println();
  Serial.println("NodeMCU is connected!");
  Serial.println(WiFi.localIP());
  ThingSpeak.begin(client);
  lcd.begin(16, 2);
  delay(1000);
  lcd.print("Distance in cm:");
}

void loop() {
  long duration, cm=7;
```

```
digitalWrite(pingPin, LOW);
delayMicroseconds(2);
digitalWrite(pingPin, HIGH);
delayMicroseconds(10);
digitalWrite(pingPin, LOW);

duration = pulseIn(echoPin, HIGH);
cm = microsecondsToCentimeters(duration);

Serial.print("WaterLevel in cm: ");
Serial.println(cm);
  // Display distance on LCD
lcd.setCursor(0, 1);
lcd.print("               "); // Clear the previous reading
lcd.setCursor(0, 1);
lcd.print(cm);
// delay(2000);
ThingSpeak.writeField(myChannelNumber, 1, cm, myWriteAPIKey);
delay(15000);
bool danger=false;
danger=ThingSpeak.readLongField(myChannelNumber2, 1,myreadAPIKey);
if (danger) {
  Serial.println("Danger ");
  digitalWrite(alarm,HIGH);
  tone(alarm, 5000); // Send 5KHz sound signal...
}
else{
  Serial.println("Safe ");
  digitalWrite(alarm,LOW);
  noTone(alarm);     // Stop sound...
```

```
  }
  delay(15000);
}

long microsecondsToCentimeters(long microseconds) {
  return microseconds / 29 / 2;
}
```

Code for MATLAB analysing of data on ThingSpeak:

```
readChannelID = 2285158;
writeChannelID = 2287036;
writeAPIKey = '1394LXB2V07DK9KU';

[data, time] = thingSpeakRead(readChannelID);

threshold = 20;
pumpStatus = false;

if data < threshold
   pumpStatus = true;
end

%% Write Data to channel%%
thingSpeakWrite(writeChannelID, pumpStatus, 'WriteKey', writeAPIKey);
```

Instruction to Users (User Manual) and Comments

Ensure proper power supply to NodeMCU (esp8266). Often power supplied through laptop USB port is not enough to power it. Keep in mind voltage and power ratings of different components. ESP8266 operates at 3.3V and ultrasonic sensor has range 3.3V-5V. If we connect Vin of distance sensor directly to nodemcu we may not get desired result as esp8266 may not provide enough power. It is a good idea to power external components separately.

We are using the actuator and sensor on the same NodeMCU but they are not communicating with each other directly. We have used single MCU to save resources. In practice sensor and actuator can be situated at any two different places in the world. To implement this cut channel reading part and actuator manipulating part from above code and upload it to another MCU that has actuators.

Acknowledgment

We thank our instructor, Dr. Manas Khatua, for his valuable instruction that enabled us to work on this project. We are also grateful to our friends who lent us their equipment for this project.