Forest Fire Detection using LoRa technology

K. Varshitha,
Dept. of Physics & Electronics
Bhavan's Vivekananda College,
Secunderabad, Telangana, India
varshithareddy464@gmail.com

M.Prasanna
Dept. of Physics & Electronics
Bhavan's Vivekananda College,
Secunderabad, Telangana, India
prasanna.elec@bhavansvc.ac.in

ABSTRACT

Forest fires are more frequent than ever today. The Amazon and many woodlands have been burnt. The source of the fire is unknown, but every year fires that threaten people's lives, homes, and property as well as devour and destroy thousands of hectares of forest land impact the forest. Despite investments and advancements in firefighting methods and tools, fires frequently cause more destruction than prevention because of delayed discovery. To prevent extensive harm to natural resources, fire detection should occur at the appropriate moment. With the help of the LoRa communication technology, this project intends develop a simple IoT application to quickly identify the occurrence of fire in forests using Arduino device and sensors.

Keywords: Internet-of-Things (IoT), LoRa technology, Arduino UNO, sensors

I. INTRODUCTION

The Arduino Uno, which is based on WSN, can be used to monitor and forecast forest fires, which are getting worse over time. In order to detect the temperature and gases created by the fire, a temperature sensor and an IR sensor are interfaced to the Arduino in this project. The sensor collects the values, which are then sent to the LoRa. The system must identify the fire as soon as possible, and it is crucial to pinpoint its specific location and notify the firefighting units.

II. SYSTEM DESIGN

The existing fire alarm systems with a buzzer have expensive monitoring systems, some of them require more technical man power for maintenance The proposed system with Arduino UNO uses LoRa devices for monitoring and alerting when fire is detected, this system can as well be used for any parameter measurement or disaster management. Both local and global fire alert is provided, Very low cost implementation. Also camera based detection can be provided as failsafe in case sensors malfunction.

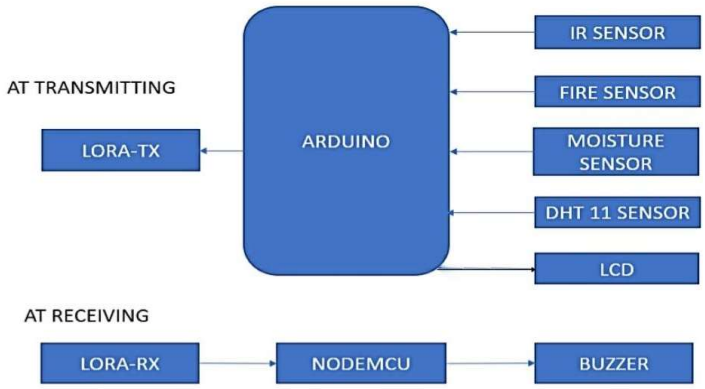


Fig 1. BLOCKDIAGRAM

SYSTEMREQUIREMENTS

- Arduino Uno
- LoRa Module
- NodeMCU
- Moisture Sensor
- IR Sensor
- Fire Sensor
- Buzzer
- I2C LCD
- DHT 11
- Arduino IDE

III. HARDWARE DESCRIPTIONS

A. LoRa Module

LoRa devices have IoT applications in many tough areas such as: energy management, natural resources protection, pollution management, infrastructure efficiency, and disaster management. Semtech's LoRa devices have several use cases for wise cities, homes and buildings, communities, metering, supply chain and provision, agriculture, and many more. Semtech corporation is the leader in LoRa wireless technology and have introduced LoRa RF modules for the market. especially, the SX127x family of RF transceivers for the IoT/M2M markets. These RF modules operate between 860-1000 rate and 137- 960MHz. LoRa devices with 865.0625 MHz, 865.4025 MHz, 865.985 rate frequency channels are used in Asian countries. LoRa wireless technology plays a key role in the IoT market in interconnecting devices to form wise cities, and industrial solutions, whereas reducing the restrictions from other wireless technologies like power and other overheads.

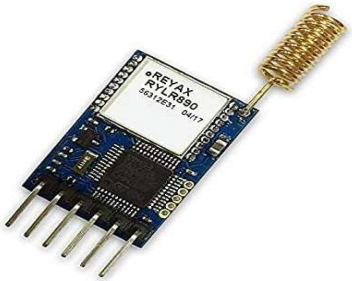


Fig 2: LoRa module

B. Arduino UNO

Fig 3: Arduino Uno

The Arduino Uno hasMicrochipATmega328P microcontroller and is developed by Arduino.cc. The board is supplied with sets of digital and analog input/output (I/O) pins which can be interfaced to various growth boards (shields) and totally different circuits. The board has fourteen Digital pins, six analog pins, and programmable with the Arduino IDE (Integrated Development Environment) via USB cable. It can be powered by the USB cable or external 9-volt battery, though it accepts voltages between seven and twenty volts. The board is typically programmed for any application through a bunch of directions to the microcontroller.

C**. DHT11**

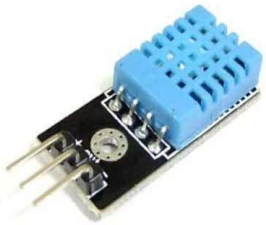


Fig 4:DHT 11 sensor

DHT11 is an inexpensive digital detector module for sensing temperature and wetness(humidity).This detector unit can typicallybe interfaced with any micro-controller like Arduino, Raspberry Pi etc… to measure wetness and temperature. This detector is utilized here to monitor the wetness variation of the environment where the fire. This detector uses a thermistorand a physical phenomenon wetness detector.

D. NodeMCU

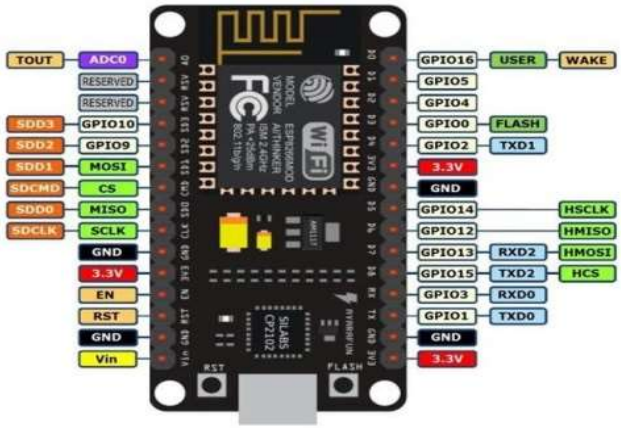


Fig 5:NodeMCU

The NodeMCU (Node MicroController Unit) is hardware development board designed around an inexpensive System-on-a-Chip (SoC) called the ESP8266. The ESP8266, designed and created by

Espressif Systems, contains the crucial parts of a computer hardware like RAM, networking (WiFi), and even an up to date software package and SDK.

E. IR sensor

IR sensors use radar technology, they emit and receive infrared radiation, this radiation hits the objects nearby and bounces back to the receiver of the device. The sensor not only detects movement in an environment but measures how far the object is from the sensor.

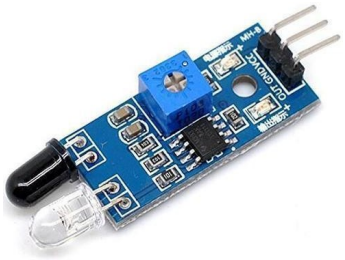


Fig 6 IR sensor

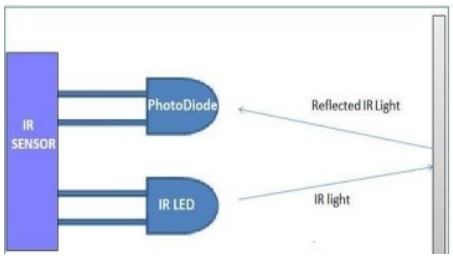


Fig 7 working principle

IV SOFTWARE

A. ARDUINO IDE

Connect the Arduino board to the PC via the USB cable, and next follow the below steps

* Board Setup
* COM Port Setup
* Uploading of sketch

Download and install Arduino IDE(https://www.arduino.cc/en/Main/Software)

1. Introduce your Arduino Board
2. Select the proper board at intervals the IDE (Tools>Boards>Arduino Uno)
3. Select the proper COM port (Tools>Port>COMx (Arduino Uno))
4. Open the "Blink" sketch(File>Examples>Basics>01.Blink)
5. Press the transfer button to transfer the program to the board

B. THINGSPEAK

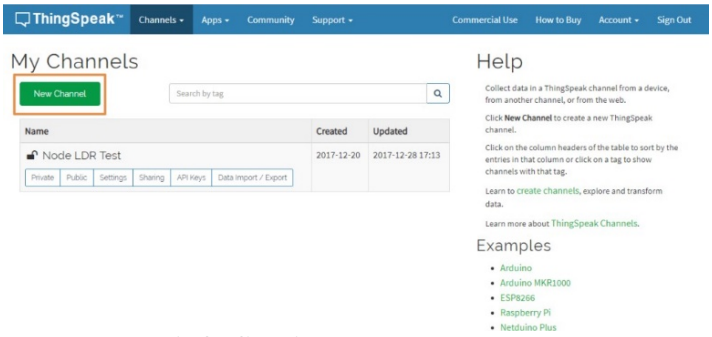


Fig 8 . Creating channel

Thing Speak API is used to store and retrieve info from hardware devices and sensors and develop IoT applications. Also, this platform provides apps to analyse and visualize information. It uses protocol over internet or network for its communication. The MATLAB analytics is encircled to analyse and visualize the data received from Hardware or detector Devices. Channels are created for each and every detector information.

Create account in thingspeak and add channels with description for each parameter

Step 1:https://thingspeak.com/ Login to Your Account.

Step 2:add a Channel by clicking 'New Channel'.

Step 3: Enter the channel details.

Name: Any Name

Description: optional

Step 4: presently you will be ready to see the channels. Click on the 'API Keys'tab. Here you'll get the Channel ID and APIKeys. Note this down.

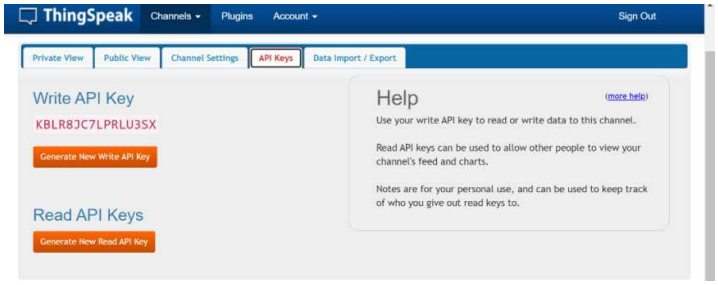


Fig9. API Keys

Step 5: Install ThingSpeak library.

to do this visit Sketch>Include Library>Manage Libraries. seek for ThingSpeak and install the library.

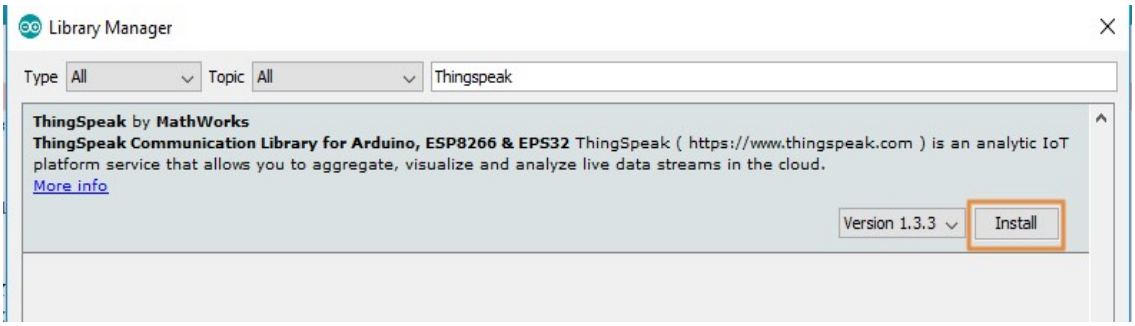


Fig10: library installing

Step 6: Click **Save Channel** at the bottom of the settings.

V. SOURCE CODE

Source code to Arduino

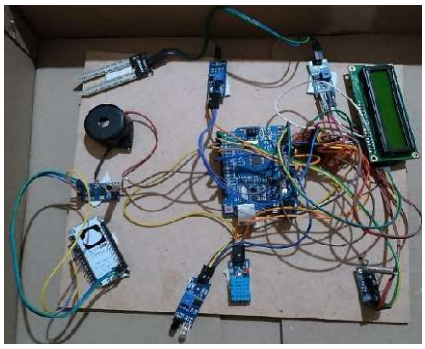
```
#include"DHT.h"
#include<Wire.h>
#include <LiquidCrystal_I2C.h>
// Set the LCD address to 0x27 for a 16 chars and 2 line display LiquidCrystal_I2C lcd(0x27, 16, 2);
#define DHTPIN 6
#define DHTTYPE DHT11 int ir1 = 8;
int fir = 7; Software Serial lora(2,3);
DHT dht (DHTPIN, DHTTYPE);

void setup()
{
Serial.begin(9600); pinMode(7, INPUT);

lora.begin(9600); dht.begin();
lcd.begin();

// Turn on the blacklight and print a message. lcd.backlight();
lcd.print("Hello, world!");
}
void loop()
{
float h = dht.readHumidity();
float t = dht.readTemperature(); float s = h + 40;
float r = t + 63;
float f = (t*1.8+32)-72; if (isnan(h) || isnan(t) ) {
Serial.println(F("Failed to read from DHT sensor!")); return;
}
int moisture = analogRead(A0);
int ir = digitalRead(ir1);
int fire = digitalRead(fir);
Serial.print("fire");
Serial.println(fire);
lcd.clear();
lcd.print("fire");
lcd.print(fire);
delay(1000);
Serial.print("ir");
Serial.println(ir);
lcd.clear();
lcd.print("ir");
lcd.print(ir);
delay(1000);
Serial.print("moisture");
Serial.println(moisture);
lcd.clear();
lcd.print("moisture");
lcd.print(moisture);
delay(1000);
Serial.print(" Humidity: ");
Serial.print(h);
lcd.clear();
lcd.print(" Humidity: ");
lcd.print(h);
delay(1000);
```

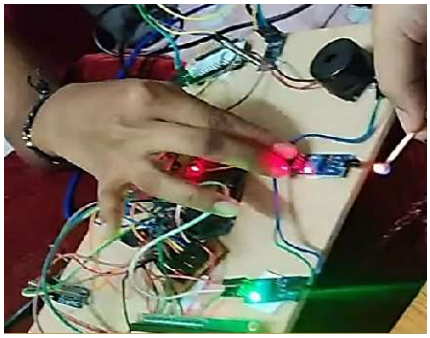
```
delay(1000);
Serial.print(" Temperature: ");
Serial.print(f);
lcd.clear();
lcd.print(" Temperature: ");
lcd.print(f);
delay(1000);
String data = String(fire) + ":" + String(ir) + ":" + String(moisture) + ":" + String(h) + ":" + String(f);
lora.println(data);
//Serial.println(data);
delay(1000);
```

A. SOURCE CODE FOR NODEMCU

```
#define SW_VERSION " ThinkSpeak.com" // SW version will appears at innitial LCD Display
#include <SoftwareSerial.h>
#include <ESP8266WiFi.h>
// The TinyGPS++ object
SoftwareSeriallora(D2, D3); // The serial connection to the GPS device const char* ssid = "Social Network";
const char* password = "PanProEdHyd&";
const char* TS_SERVER ="api.thingspeak.com";
String TS_API_KEY ="KBLR8JC7LPRLU3SX";//

WiFiClient client1;
String fire,ir,moisture,h,f;
String getStringPartByNr(String data, char separator, int index)
{
// spliting a string and return the part nr index
// split by separator
int stringData= 0; //variable to count data part nr
String dataPart="";//variable to hole the return text
for(int i = 0; i<data.length()-1;i++){
//Walk through the text one letter at a time if(data[i]==separator){
//Count the number of times separator character appears in the text stringData++;
}
else if(stringData==index) {
//get the text when separator is the rignt one dataPart.concat(data[i]);
}else if(stringData>index) {
//return text and stop if the next separator appears - to save CPU-time return dataPart;
break;
}
}
//return text if this is the last part return dataPart;
}
void sendDataTS(void)
{
if (client1.connect(TS_SERVER, 80))
{
String postStr = TS_API_KEY; postStr +="&field1=";
postStr += String(f); postStr += "&field2="; postStr += String(ir); postStr +="&field3=";
postStr += String(moisture); postStr += "&field4="; postStr += String(fire); postStr +="&field5="; postStr +=
String(h); postStr +="\r\n\r\n";
client1.print("POST /update HTTP/1.1\n"); client1.print("Host: api.thingspeak.com\n");
client1.print("Connection: close\n");
client1.print("X-THINGSPEAKAPIKEY: " + TS_API_KEY + "\n");
client1.print("Content-Type: application/x-www-form-urlencoded\n");

client1.print("Content-Length: "); client1.print(postStr.length()); client1.print("\n\n"); client1.print(postStr);
```

```
delay(1000);
}
client1.stop();
}
void setup()
{
pinMode(D5,OUTPUT); Serial.begin(9600); lora.begin(9600); Serial.println(); Serial.print("Connecting to ");
 Serial.println(ssid);
WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED)
{
delay(500); Serial.print(".");
}
Serial.println("");
Serial.println("WiFi connected");
Serial.println("Server started");
// Print the IP address Serial.println(WiFi.localIP());
 Serial.print("Connected! IP address: ");
}
void loop()
{
if(lora.available()>0)
{
String rcv=lora.readString();
 Serial.println(rcv);
fire=(getStringPartByNr(rcv,':',0));
 ir=(getStringPartByNr(rcv,':',1));
moisture=(getStringPartByNr(rcv,':',2));
h=(getStringPartByNr(rcv,':',3));
f=(getStringPartByNr(rcv,':',4));
}
Serial.print((" fire Value: ")); Serial.println(fire);Serial.print((" hum: ")); Serial.println(h); Serial.print((" moisture: ")); Serial.println(moisture);
Serial.print(("object: ")); Serial.println(ir);
// Serial.println(("C ")); Serial.print(("temperature: ")); Serial.println(f);
float f1 =f.toInt();
int h = moisture.toInt(); int g =fire.toInt();
int t = ir.toInt();
 if(f1>30){
Serial.println(F("::::::::temperature Detected::::::::")); digitalWrite(D5,HIGH);
delay(1000);
}
elseif(g==0){

Serial.println(F("::::::::fire Detected::::::::")); digitalWrite(D5,HIGH);
delay(1000);
}
else{
digitalWrite(D5,LOW); delay(1000);
}
sendDataTS();
delay(1000);
```





VI. RESULTS

Fig11 **Fig12**

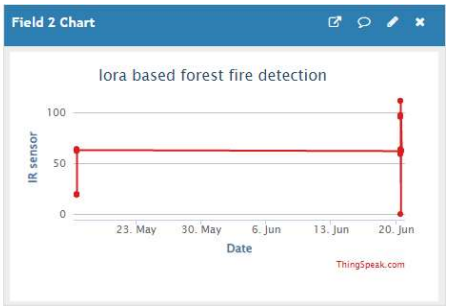


Fig13 **Fig14**

VI. CONCLUSION & FUTURESCOPE

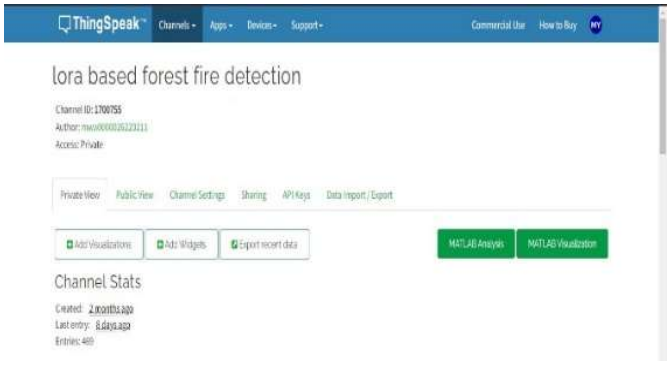
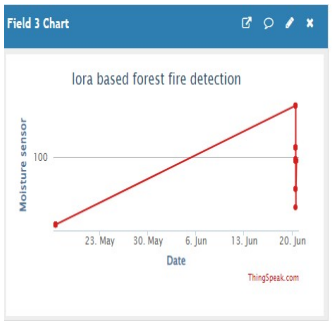
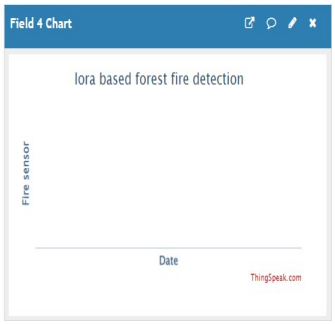


Fig15

This study described as fire monitoring system uses a wireless sensor network to inform the user remotely. This system was successfully created and implemented. The technology has been tested in a simulated fire disaster environment and has proven to be quite responsive. Efficiency can be improved by using more accurate sensors and GPS receivers and to predict the disaster. This system can be used in schools, colleges, offices and industries for any fire, gas leakage, etc.





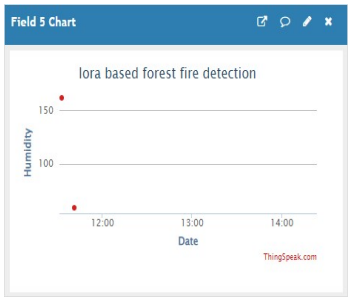


Fig16 **Fig17**

IX REFERENCES

[1]. K Ram Prasanna, J.M. Mathana, T. Anne Ramya et al., "LoRa network based high performance forest fire detection system", Materials Today: Proceedings,https://doi.org/10.1016/j.matpr.2021.05.656

[2]. K.MaheshBabu, R. Priyakanth, G. Roshini, V. Saisri, BH. Keerhipriya, N. Srujana, M. Taruni "Forest Fire Detection Using LoRa" Dept. of ECE 2020 JETIR May 2020, Volume 7, Issue 5

[3]. Aksamovic, M. Hebibovic, and D. Boskovic, "Forest fire early detection system design utilising the WSN simulator," in *2017 XXVI International Conference on Information, Communication and Automation Technologies (ICAT)*, 2017, pp. 1-5.

[4]. Y. Liu, Y. Liu, H. Xu, and K. L. Teo, "Forest fire monitoring, detection and decision making systems by wireless sensor network," in *2018 Chinese Control And Decision Conference (CCDC)*, 2018, pp. 5482-5486.

[5] Pant, S. Verma, and P. Dhuliya, "A study on disaster detection and management using WSN in Himalayan region of Uttarakhand," in *2017 3rd International Conference on Advances in Computing, Communication & Automation (ICACCA) (Fall)*, 2017, pp.1-6.

[6]https://circuitdigest.com/microcontroller-projects/iot-based-forest-fire-detection-using-arduino-and-gsm-module

[7] Kia C. Wiklundh "Understanding the IoT technology LoRa and its interference vulnerability"*September 2– 6, 2019.Proc. of the 2019 International Symposium on Electromagnetic Compatibility (EMC Europe 2019), Barcelona, Spain,*