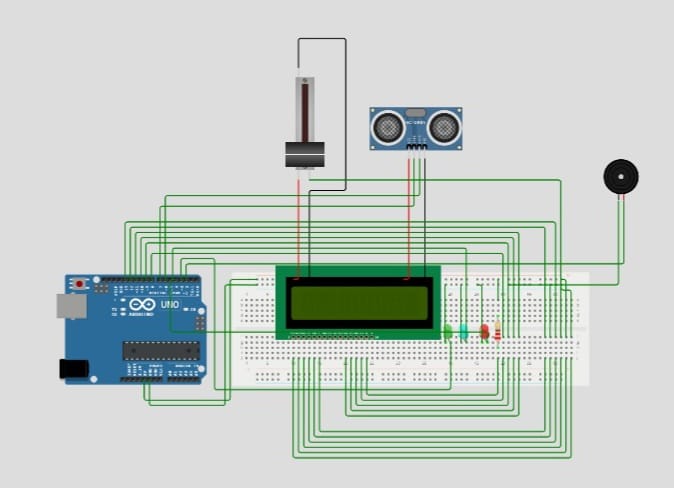
Flood Monitoring & Early Warning

Phase-3

Design for Flood Monitoring & early warning



C++ code for flood monitoring & Early warning

#include <LiquidCrystal.h>

LiquidCrystal lcd(13, 12, 11, 10,  9, 8);

int trig = 7;

int echo = 6;

int red = 5;

int blue = 4;

int green = 3;

int buz = 2;

int distance;

long duration;

void setup()

{

 pinMode(red,OUTPUT);

 pinMode(blue,OUTPUT);

 pinMode(green,OUTPUT);

 pinMode(buz,OUTPUT);

 pinMode(trig,OUTPUT);

 pinMode(echo,INPUT);

**Serial**.begin(9600);

 lcd.begin(16,2);

 lcd.setCursor(1,0);

 lcd.print("Distance:");

}

void loop()

{

 digitalWrite(trig,LOW);

 digitalWrite(trig,HIGH);

 delayMicroseconds(10);

 digitalWrite(trig,LOW);

 float duration = pulseIn(echo,HIGH);

 // Distance = speed \* Time

 float distance = (0.0343 \* duration)/2 ;

 lcd.setCursor(10,0);

 lcd.print(distance);

  if(distance>250)

  {

   lcd.setCursor(3,1);

   lcd.print("Safe Zone  ");

   digitalWrite(red,LOW);

   digitalWrite(green,HIGH);

   digitalWrite(blue,LOW);

   digitalWrite(buz,LOW);

   delay(1000);

  }

   else if(distance>200&&distance>100)

  {

   lcd.setCursor(3,1);

   lcd.print("Medium Zone");

   digitalWrite(red,LOW);

   digitalWrite(green,LOW);

   digitalWrite(blue,HIGH);

   digitalWrite(buz,LOW);

   delay(1000);

  }

  else if(distance<100)

  {

   lcd.setCursor(3,1);

   lcd.print("Danger Zone");

   digitalWrite(red,HIGH);

   digitalWrite(green,LOW);

   digitalWrite(blue,LOW);

   digitalWrite(buz,HIGH);

  }

}

Python code for flood monitoring & Early warning

# Flood Detection System

import RPi.GPIO as GPIO

import time

GPIO.setmode(GPIO.BOARD)

lcd\_rs = 13

lcd\_en = 12

lcd\_d4 = 11

lcd\_d5 = 10

lcd\_d6 = 9

lcd\_d7 = 8

trig = 7

echo = 6

red = 5

blue = 4

green = 3

buz = 2

GPIO.setup(red, GPIO.OUT)

GPIO.setup(blue, GPIO.OUT)

GPIO.setup(green, GPIO.OUT)

GPIO.setup(buz, GPIO.OUT)

GPIO.setup(trig, GPIO.OUT)

GPIO.setup(echo, GPIO.IN)

def setup():

GPIO.output(red, GPIO.LOW)

GPIO.output(blue, GPIO.LOW)

GPIO.output(green, GPIO.LOW)

GPIO.output(buz, GPIO.LOW)

GPIO.output(trig, GPIO.LOW)

GPIO.output(trig, GPIO.HIGH)

time.sleep(0.00001)

GPIO.output(trig, GPIO.LOW)

lcd\_init()

print\_distance()

def lcd\_init():

lcd = LiquidCrystal(lcd\_rs, lcd\_en, lcd\_d4, lcd\_d5, lcd\_d6, lcd\_d7)

lcd.begin(16, 2)

lcd.setCursor(1, 0)

lcd.print("Distance:")

def print\_distance():

while True:

GPIO.output(trig, GPIO.LOW)

GPIO.output(trig, GPIO.HIGH)

time.sleep(0.00001)

GPIO.output(trig, GPIO.LOW)

duration = pulseIn(echo, GPIO.HIGH)

distance = (0.0343 \* duration) / 2

lcd.setCursor(10, 0)

lcd.print(distance)

if distance > 250:

lcd.setCursor(3, 1)

lcd.print("Safe Zone ")

GPIO.output(red, GPIO.LOW)

GPIO.output(green, GPIO.HIGH)

GPIO.output(blue, GPIO.LOW)

GPIO.output(buz, GPIO.LOW)

time.sleep(1)

elif distance > 200 and distance > 100:

lcd.

**C++ Code Explaination:**

**1. #include <LiquidCrystal.h>**

Include the LiquidCrystal library for controlling the LCD display.

**2. LiquidCrystal lcd(13, 12, 11, 10, 9, 8);**

Initialize an instance of the LiquidCrystal class to control the LCD display, specifying the pins connected to the LCD module.

**3. int trig = 7; int echo = 6; int red = 5; int blue = 4; int green = 3; int buz = 2;**

Define variables to store pin numbers for the ultrasonic sensor's trigger (trig), echo (echo), and various LEDs and a buzzer.

**4. int distance; long duration;**

Declare two variables to store the distance measured and the duration of the ultrasonic pulse.

**5.** **void setup()**

**{**

**pinMode(red, OUTPUT);**

**pinMode(blue, OUTPUT);**

**pinMode(green, OUTPUT);**

**pinMode(buz, OUTPUT);**

**pinMode(trig, OUTPUT);**

**pinMode(echo, INPUT);**

**Serial.begin(9600);**

**lcd.begin(16, 2);**

**lcd.setCursor(1, 0);**

**lcd.print("Distance:");**

**}**

In the **setup()** function:

* Set the pins for the LEDs (red, blue, green) and buzzer as OUTPUT.
* Set the trigger pin for the ultrasonic sensor as OUTPUT and the echo pin as INPUT.
* Initialize the Serial communication at a baud rate of 9600 for debugging purposes.
* Initialize the LCD with 16 columns and 2 rows, and display the initial message on the LCD.

**6.** **void loop()**

**{**

**digitalWrite(trig, LOW);**

**digitalWrite(trig, HIGH);**

**delayMicroseconds(10);**

**digitalWrite(trig, LOW);**

**float duration = pulseIn(echo, HIGH);**

**float distance = (0.0343 \* duration) / 2;**

**lcd.setCursor(10, 0);**

**lcd.print(distance);**

In the **loop()** function:

* Trigger the ultrasonic sensor to measure distance by sending a pulse and then listening for the echo.
* Calculate the distance using the formula: **Distance = speed \* Time**.
* Display the measured distance on the LCD.

**7. if (distance > 250)**

**{**

**lcd.setCursor(3, 1);**

**lcd.print("Safe Zone ");**

**digitalWrite(red, LOW);**

**digitalWrite(green, HIGH);**

**digitalWrite(blue, LOW);**

**digitalWrite(buz, LOW);**

**delay(1000);**

**}**

**else if (distance > 200 && distance > 100)**

**{**

**lcd.setCursor(3, 1);**

**lcd.print("Medium Zone");**

**digitalWrite(red, LOW);**

**digitalWrite(green, LOW);**

**digitalWrite(blue, HIGH);**

**digitalWrite(buz, LOW);**

**delay(1000);**

**}**

**else if (distance < 100)**

**{**

**lcd.setCursor(3, 1);**

**lcd.print("Danger Zone");**

**digitalWrite(red, HIGH);**

**digitalWrite(green, LOW);**

**digitalWrite(blue, LOW);**

**digitalWrite(buz, HIGH);**

**}**

**}**

Based on the measured distance, the code changes the message on the LCD and the LED color:

* If the distance is greater than 250 units, it's considered a "Safe Zone," and the LED is set to green.
* If the distance is between 200 and 100 units, it's a "Medium Zone," and the LED is set to blue.
* If the distance is less than 100 units, it's a "Danger Zone," and the LED is set to red, and the buzzer is turned on.

**Finally:**

This code essentially creates a distance measuring system using an ultrasonic sensor and provides visual and auditory feedback based on the measured distance. The LCD displays the current distance, and the LED color and buzzer indicate the safety zone.