  
  
const int triggerPin = 9; // Pin for HC-SR04 trigger

const int echoPin = 10; // Pin for HC-SR04 echo

const int tempPin = A1; // Pin for the temperature sensor (analog)

const int ledPin1 = 13; // Pin for the first LED (for HC-SR04)

const int ledPin2 = 12; // Pin for the second LED (for temperature sensor)

volatile bool measureDistance = false; // Flag to measure distance

volatile bool readTemperature = false; // Flag to read temperature

volatile float measuredDistance = 0.0; // Store the distance

volatile float temperature = 0.0; // Store the temperature

void setup() {

pinMode(triggerPin, OUTPUT); // Set trigger pin as OUTPUT

pinMode(echoPin, INPUT); // Set echo pin as INPUT

pinMode(tempPin, INPUT); // Set temperature pin as INPUT

pinMode(ledPin1, OUTPUT); // Set LED1 pin as OUTPUT

pinMode(ledPin2, OUTPUT); // Set LED2 pin as OUTPUT

Serial.begin(9600); // Initialize serial communication

// Configure Timer1 to trigger an interrupt every 100 milliseconds

noInterrupts(); // Disable interrupts while configuring timer

TCCR1A = 0;

TCCR1B = 0;

TCNT1 = 0; // Initialize counter value to 0

OCR1A = 15624; // Set compare match register for 100ms (assuming 16MHz clock)

TCCR1B |= (1 << WGM12); // Turn on CTC mode

TCCR1B |= (1 << CS12) | (1 << CS10); // Set prescaler to 1024

TIMSK1 |= (1 << OCIE1A); // Enable Timer1 compare interrupt

// Configure Timer2 to trigger an interrupt every 50 milliseconds for temperature sensor

TCCR2A = 0;

TCCR2B = 0;

TCNT2 = 0; // Initialize counter value to 0

OCR2A = 124; // Set compare match register for 50ms (assuming 16MHz clock)

TCCR2A |= (1 << WGM21); // Turn on CTC mode

TCCR2B |= (1 << CS22) | (1 << CS21); // Set prescaler to 256

TIMSK2 |= (1 << OCIE2A); // Enable Timer2 compare interrupt

interrupts(); // Enable interrupts

}

void loop() {

// Check if it's time to measure distance

if (measureDistance) {

measureDistance = false; // Reset flag

// Send a signal to the trigger pin

digitalWrite(triggerPin, LOW);

delayMicroseconds(2);

digitalWrite(triggerPin, HIGH);

delayMicroseconds(10);

digitalWrite(triggerPin, LOW);

// Read the duration of the pulse from the echo pin

long pulseDuration = pulseIn(echoPin, HIGH);

// Convert the pulse duration to distance in centimeters

measuredDistance = (pulseDuration / 2.0) \* 0.0344;

// Display the measured distance on the Serial Monitor

Serial.print("Measured Distance: ");

Serial.print(measuredDistance);

Serial.println(" cm");

// Control LED1 based on distance

if (abs(measuredDistance - 113.4) < 1.0) { // Acceptable error margin of 1 cm

digitalWrite(ledPin1, HIGH);

delay(500); // LED1 on for 500 milliseconds

digitalWrite(ledPin1, LOW);

delay(500); // LED1 off for 500 milliseconds

} else {

digitalWrite(ledPin1, LOW);

}

}

// Check if it's time to read temperature

if (readTemperature) {

readTemperature = false; // Reset flag

// Read the temperature sensor value

int sensorValue = analogRead(tempPin);

// Convert the sensor value to temperature in Celsius

temperature = (sensorValue \* 5.0 / 1024.0) \* 100.0;

// Display the temperature on the Serial Monitor

Serial.print("Temperature: ");

Serial.print(temperature);

Serial.println(" C");

// Check both conditions and blink LED2 if both are met

if (abs(measuredDistance - 113.4) < 1.0 && abs(temperature - 74.1) < 1.0) { // Acceptable error margin

blinkLED(ledPin2, 3); // Blink LED2 thrice

}

}

delay(100); // Short pause before the next measurement

}

// Function to blink an LED a specified number of times

void blinkLED(int pin, int times) {

for (int i = 0; i < times; i++) {

digitalWrite(pin, HIGH); // Turn LED on

delay(500); // LED on for 500 milliseconds

digitalWrite(pin, LOW); // Turn LED off

delay(500); // LED off for 500 milliseconds

}

}

// Timer1 interrupt service routine for HC-SR04

ISR(TIMER1\_COMPA\_vect) {

measureDistance = true; // Set flag to measure distance

}

// Timer2 interrupt service routine for temperature sensor

ISR(TIMER2\_COMPA\_vect) {

readTemperature = true; // Set flag to read temperature

}

In this setup, the Arduino board is configured to handle two sensors using interrupts for effective real-time monitoring. The HC-SR04 ultrasonic distance sensor is connected with its trigger pin on digital pin 9 and echo pin on digital pin 10. The temperature sensor is connected to analog pin A1. The system uses Timer1 to trigger interrupts every 100 milliseconds for distance measurements and Timer2 to trigger interrupts every 50 milliseconds for temperature readings. The flags measureDistance and readTemperature, set by the interrupt service routines (ISRs) associated with Timer1 and Timer2 respectively, indicate when to process the sensor data. The loop function checks if the distance measured by the HC-SR04 (using pin 10) is approximately 113.4 cm and if the temperature read from the analog pin A1 is approximately 74.1°C. If both conditions are satisfied, the LED connected to digital pin 12 blinks thrice to signal the result. Additionally, another LED connected to digital pin 13 provides visual feedback based on the distance measurement. This setup efficiently integrates multiple sensors and outputs, utilizing interrupts to ensure timely and accurate responses based on the collected data.

<https://github.com/s223200581/Module-1-sit-315-task-1.3c>