





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

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

```
const int ledPin = 13; // Pin for the LED
```

```
volatile bool measureDistance = false; // Flag to indicate when to measure distance
```

```
volatile float measuredDistance = 0.0; // Store the measured distance
```

```
void setup() {
```

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

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

```
    pinMode(ledPin, OUTPUT); // Set LED pin as an 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
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");
    }
}
```

```

// Determine if the measured distance is approximately 113.4 cm
if (abs(measuredDistance - 113.4) < 1.0) { // Acceptable error margin of 1 cm
    // Activate the LED with a blink pattern
    digitalWrite(ledPin, HIGH);
    delay(500); // LED on for 500 milliseconds
    digitalWrite(ledPin, LOW);
    delay(500); // LED off for 500 milliseconds
} else {
    // Ensure the LED is turned off if distance does not match
    digitalWrite(ledPin, LOW);
}
}

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

// Timer1 interrupt service routine
ISR(TIMER1_COMPA_vect) {
    measureDistance = true; // Set flag to measure distance
}

```

<https://github.com/s223200581/module1-task-1.2p/upload>