FLOOD MONITORING AND EARLY WARNING SYSTEM USING IOT

Term member

411521104012:Bhuvanesh.G

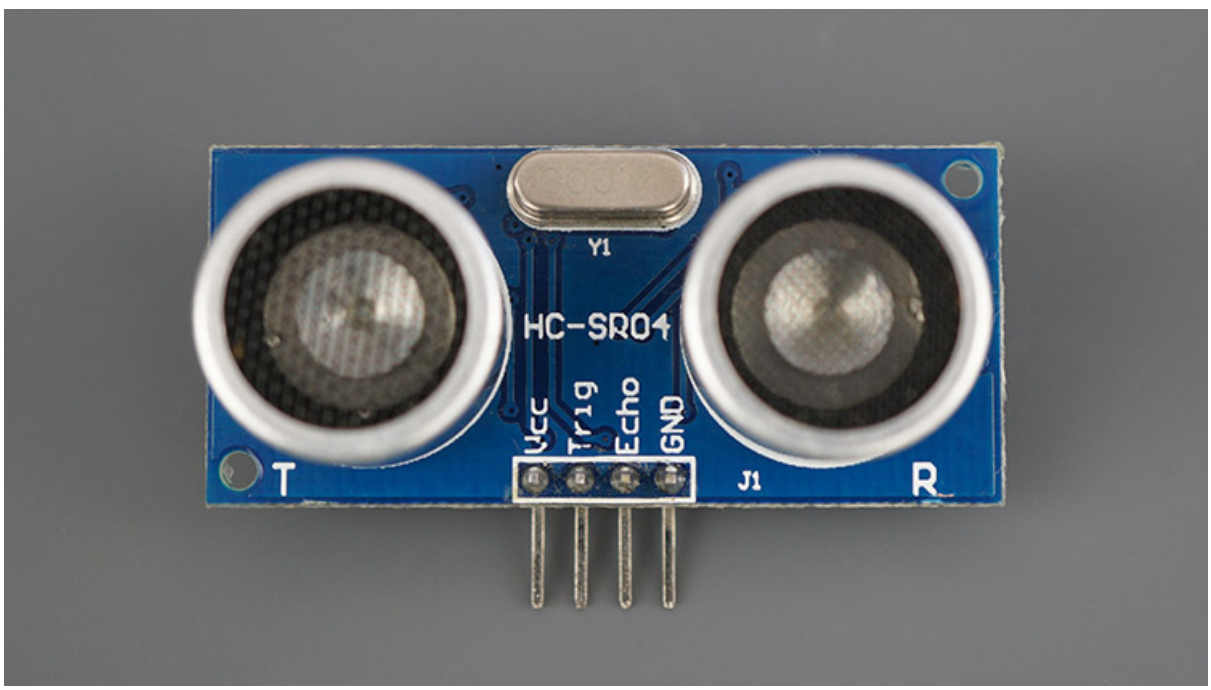
Phase 3 Document Submission

HC-SR04 ULTRASONIC SENSOR

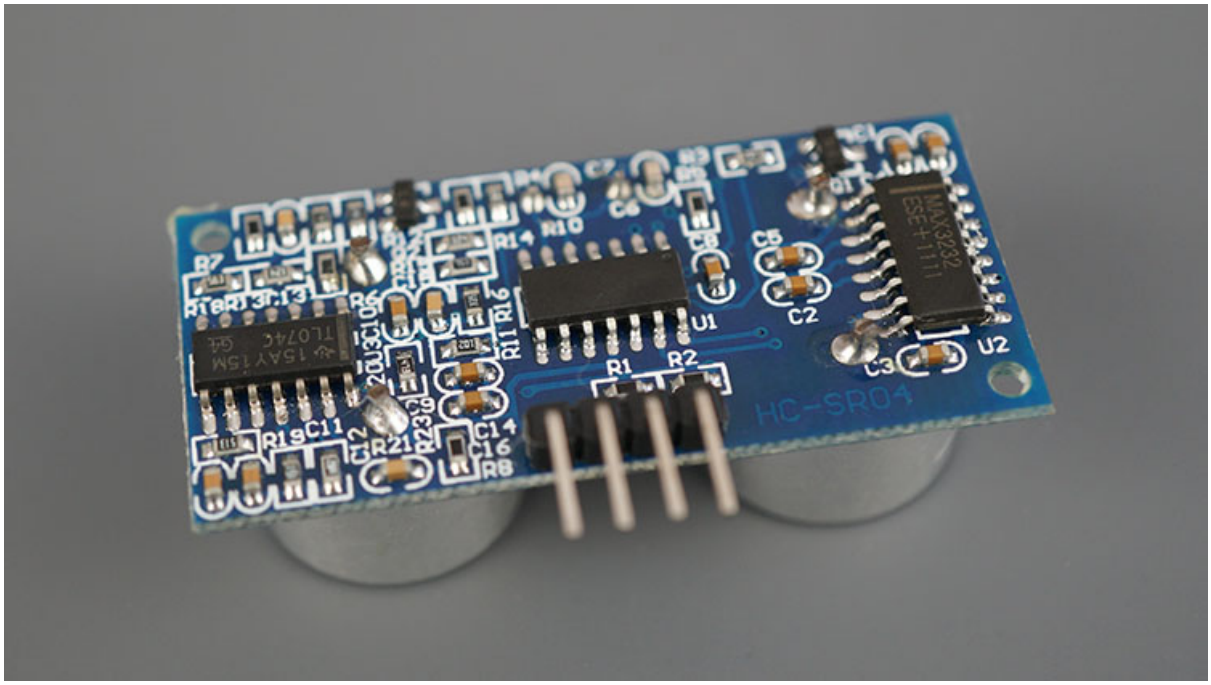
DESCRIPTION

The HC-SR04 ultrasonic sensor uses sonar to determine the distance to an object. This sensor reads from 2cm to 400cm (0.8 inch to 157 inch) with an accuracy of 0.3cm (0.1inches), which is good for most hobbyist projects. In addition, this particular module comes with ultrasonic transmitter and receiver modules.

The following picture shows the HC-SR04 ultrasonic sensor.



The next picture shows the other side of the sensor.



FEATURES

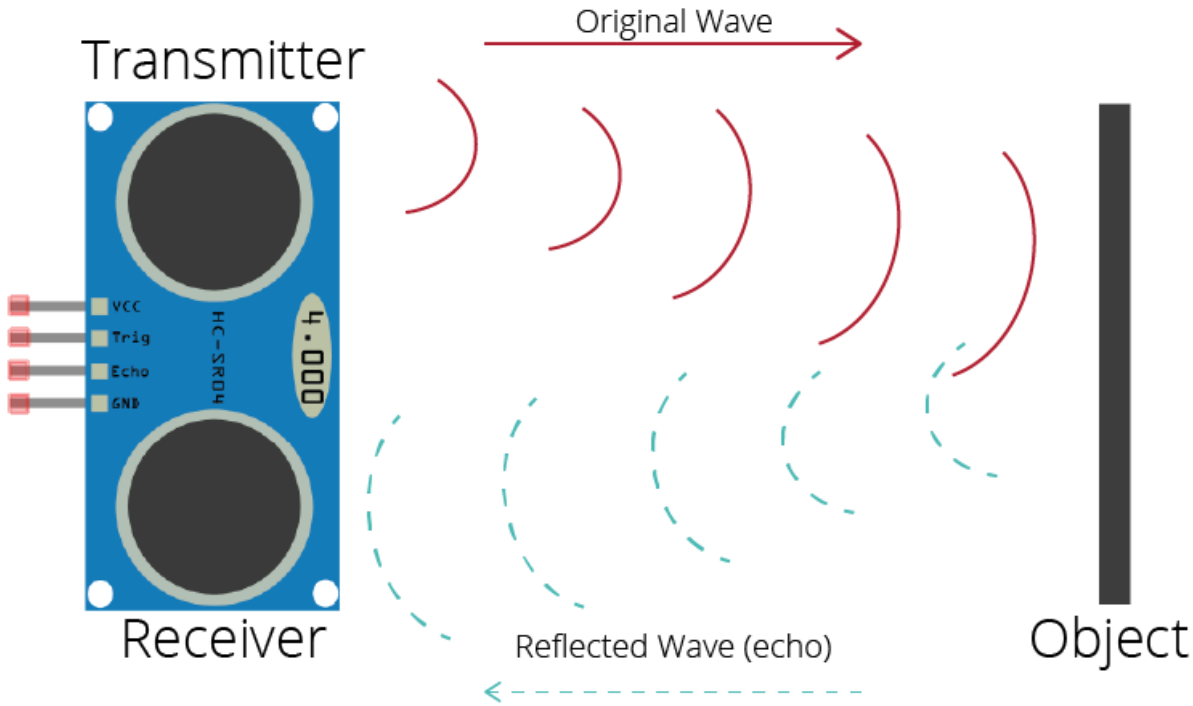
Here's a list of some of the HC-SR04 ultrasonic sensor features and specs—for more information, you should consult the sensor's datasheet:

- Power Supply :+5V DC
- Quiescent Current : <2mA
- Working Current: 15mA
- Effectual Angle: <15°
- Ranging Distance : 2cm – 400 cm/1" – 13ft
- Resolution : 0.3 cm
- Measuring Angle: 30 degree
- Trigger Input Pulse width: 10uS TTL pulse
- Echo Output Signal: TTL pulse proportional to the distance range
- Dimension: 45mm x 20mm x 15mm

HOW DOES IT WORK

The ultrasonic sensor uses sonar to determine the distance to an object. Here's what happens:

1. The ultrasound transmitter (trig pin) emits a high-frequency sound (40 kHz).
2. The sound travels through the air. If it finds an object, it bounces back to the module.
3. The ultrasound receiver (echo pin) receives the reflected sound (echo).



The time between the transmission and reception of the signal allows us to calculate the distance to an object. This is possible because we know the sound's velocity in the air. Here's the formula:

PYTHON PROGRAM

```
pip install paho-mqtt RPi.GPIO
import paho.mqtt.client as mqtt
import RPi.GPIO as GPIO
import time

# MQTT Broker Details
broker_address = "your_broker_address"
broker_port = 1883  # Default MQTT port
username = "your_username"
password = "your_password"
# GPIO Pin for Water Level Sensor
water_level_pin = 17  # Change this to match your hardware setup
# Water Level Threshold
water_level_threshold = 50  # Modify this as needed
# Initialize GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setup(water_level_pin, GPIO.IN)
# Callback when the MQTT client is connected to the broker
def on_connect(client, userdata, flags, rc):
    print("Connected to MQTT Broker with result code " + str(rc))
# Create an MQTT client
client = mqtt.Client("WaterLevelSensor")
```

```
client.username_pw_set(username, password)
client.on_connect = on_connect
# Connect to the MQTT broker
client.connect(broker_address, broker_port)
try:
    while True:
        # Read the water level sensor status
        water_level = GPIO.input(water_level_pin)
        # Publish water level data to MQTT broker
        topic = "water_level"
        message = "High" if water_level == GPIO.HIGH else "Low"
        client.publish(topic, message)
        # You can add more logic here to send the data only when the level changes significantly
        time.sleep(10)  # Delay between readings (adjust as needed)
except KeyboardInterrupt:
    print("Script terminated by user.")
# Disconnect from the MQTT broker and cleanup GPIO on exit
client.disconnect()
GPIO.cleanup()
```