LAPORAN PRAKTIKUM INTERNET OF THINGS (IoT)

Fakultas Vokasi, Universitas Brawijaya

Latihan 4: Praktik Simulasi Sensor Jarak (Ultrasonic)

Author(s) (Najwa Firdaus Azkiyah)
Fakultas Vokasi, Universitas Brawijaya
Email: najwazkiyah@student.ub.ac.id

Abstract (Abstrak)

Praktik simulasi ini menggunakan platform Wokwi untuk mengemulasi dan memprogram mikrokontroler ESP32 dengan sensor jarak ultrasonik. Wokwi menyediakan lingkungan simulasi berbasis web yang memungkinkan pengguna untuk merancang dan menguji rangkaian elektronik secara virtual tanpa memerlukan perangkat keras fisik. Dalam simulasi ini, sensor jarak ultrasonik digunakan untuk mengukur jarak objek dengan memanfaatkan gelombang ultrasonik. Mikrokontroler ESP32 membaca data dari sensor dan mengolahnya untuk berbagai aplikasi, seperti deteksi objek atau pengukuran jarak. Simulasi ini bertujuan untuk memberikan pemahaman yang mendalam tentang prinsip kerja sensor ultrasonik, pengintegrasiannya dengan ESP32, serta penerapan logika pemrograman untuk memperoleh data jarak secara akurat. Melalui simulasi ini, pengguna dapat memperoleh pengalaman praktis dan meningkatkan keterampilan pemrograman mikrokontroler, yang sangat berguna dalam pengembangan proyek-proyek IoT dan sistem penginderaan jarak.

Keywords—Internet of Things, ESP32, ultrasonic, arduino

1. Introduction (Pendahuluan)

1.1 Latar belakang

Simulasi sensor jarak ultrasonik menggunakan ESP32 adalah metode praktis untuk memahami dan mengembangkan proyek IoT. Sensor ultrasonik menggunakan gelombang suara untuk mengukur jarak ke objek, yang kemudian diolah oleh mikrokontroler ESP32. Wokwi menyediakan platform simulasi berbasis web yang memungkinkan pengguna untuk merancang dan menguji rangkaian elektronik secara virtual tanpa memerlukan perangkat keras fisik. Simulasi ini membantu pengguna mempelajari prinsip kerja sensor ultrasonik dan integrasinya dengan ESP32 sebelum diimplementasikan dalam proyek nyata.

1.2 Tujuan eksperimen

- Memahami Prinsip Kerja Sensor Ultrasonik: Menjelajahi cara kerja sensor ultrasonik dalam mengukur jarak objek menggunakan gelombang suara.
- Mengembangkan Keterampilan Pemrograman ESP32: Meningkatkan kemampuan menulis kode yang mengolah data dari sensor ultrasonik.
- Menguji dan Memvalidasi Rangkaian: Memastikan bahwa sensor dan ESP32 berfungsi dengan baik dalam simulasi sebelum implementasi fisik.
- Mendapatkan Pengalaman Praktis: Memberikan pengalaman hands-on dalam merancang dan menyimulasikan sistem penginderaan jarak.
- Meningkatkan Efisiensi Pengembangan Proyek: Menggunakan simulasi untuk mengidentifikasi dan memperbaiki masalah sebelum perangkat keras fisik digunakan.

2. Methodology (Metodologi)

2.1 Tools & Materials (Alat dan Bahan)

> Mikrokontroler (ESP8266, Arduino, Raspberry Pi, dll.), sensor (DHT22, PIR, dsb.), software (Arduino IDE, MQTT Broker, dsb.), sensor ultrasonic

2.2 Implementation Steps (Langkah Implementasi)

Berikut adalah langkah-langkah implementasi praktis untuk simulasi sensor jarak ultrasonik menggunakan ESP32 di platform Wokwi:

1. **Persiapan Lingkungan Simulasi:**
   - Buka [Wokwi](https://wokwi.com/) di peramban web Anda.
   - Pilih **New Project** dan pilih **ESP32** sebagai mikrokontroler yang akan digunakan.

2. **Penambahan Komponen:**
   - Tambahkan komponen **sensor ultrasonik (HC-SR04)** dari pustaka komponen yang tersedia di Wokwi.

- Atur posisi komponen di canvas dan hubungkan pin-pin komponen sesuai dengan skema rangkaian.

3. **Koneksi Komponen:**
   - Sambungkan pin VCC sensor ultrasonik ke pin 5V ESP32.
   - Sambungkan pin GND sensor ultrasonik ke pin ground ESP32.
   - Sambungkan pin trig sensor ultrasonik ke salah satu pin GPIO ESP32 (misalnya, GPIO 5).
   - Sambungkan pin echo sensor ultrasonik ke pin GPIO ESP32 lainnya (misalnya, GPIO 18).

4. **Pemrograman ESP32:**
   - Klik **Code** di Wokwi untuk membuka editor kode.
   - Tulis kode untuk mengatur pin GPIO sebagai input dan output untuk sensor ultrasonik. Contoh kode selengkapnya:

```
#include <Arduino.h>

const int trigPin = 19;
const int echoPin = 18;

// Define sound speed in cm/uS
#define SOUND_SPEED 0.034
#define CM_TO_INCH 0.393701

long duration;
float distanceCm;
float distanceInch;

void setup() {
  Serial.begin(115200); // Starts the serial communication
  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
  pinMode(echoPin, INPUT); // Sets the echoPin as an Input
}

void loop() {
  // Clears the trigPin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  // Sets the trigPin on HIGH state for 10 micro seconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
```

```
  digitalWrite(trigPin, LOW);
   // Reads the echoPin, returns the sound wave travel time in
microseconds
  duration = pulseIn(echoPin, HIGH);
  // Calculate the distance
  distanceCm = duration * SOUND_SPEED / 2;
  // Convert to inches
  distanceInch = distanceCm * CM_TO_INCH;
  // Prints the distance in the Serial Monitor
  Serial.print("Distance (cm): ");
  Serial.println(distanceCm);
  // Serial.print("Distance (inch): ");
  // Serial.println(distanceInch);
  delay(1000);
}
```

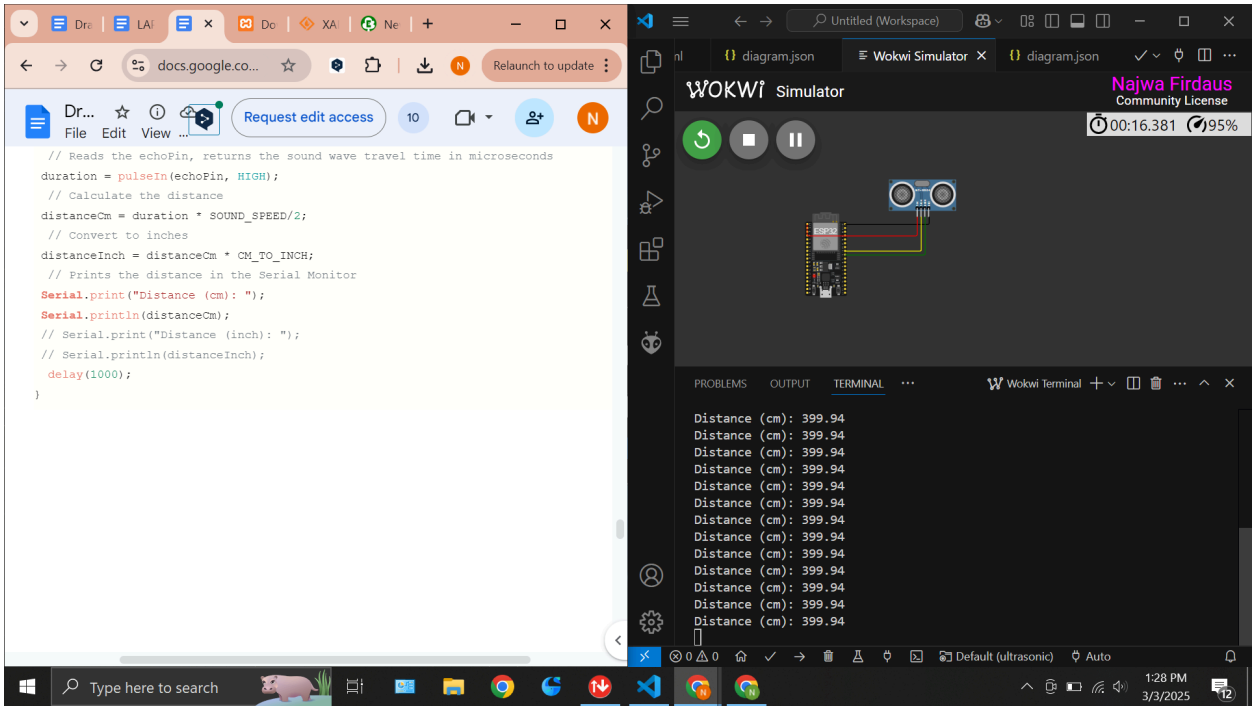
5. **Simulasi:**
   - Klik **Start Simulation** untuk menjalankan simulasi dan mengamati cara kerja sensor ultrasonik.
   - Perhatikan output pada Serial Monitor untuk melihat jarak yang terdeteksi oleh sensor.

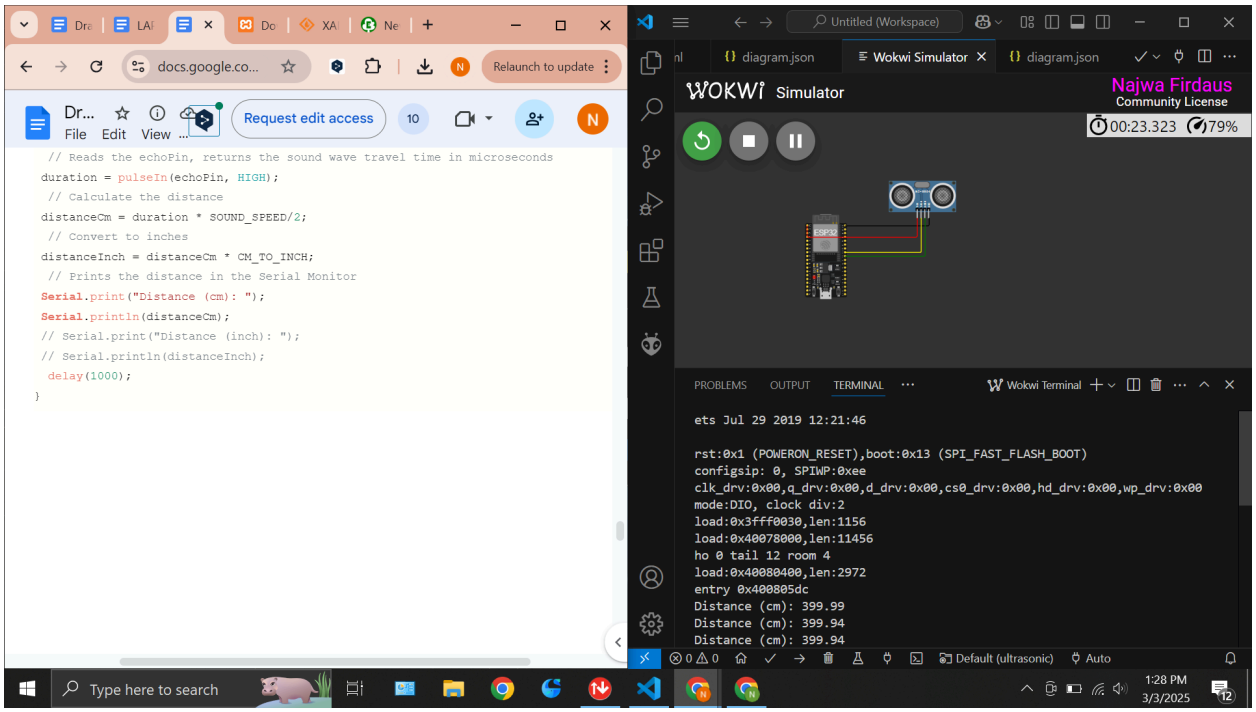
6. **Troubleshooting dan Optimasi:**
   - Uji dan pantau rangkaian untuk memastikan semuanya berfungsi dengan baik.
   - Lakukan penyesuaian pada rangkaian atau kode jika diperlukan untuk mencapai hasil yang diinginkan.

Langkah-langkah ini memberikan panduan singkat dan efektif untuk memahami dan mengimplementasikan simulasi sensor jarak ultrasonik menggunakan Wokwi dengan ESP32. Jika ada pertanyaan lebih lanjut atau butuh penjelasan lebih detail, jangan ragu untuk bertanya!

3. Results and Discussion (Hasil dan Pembahasan)

3.1 Experimental Results (Hasil Eksperimen)





4. Appendix (Lampiran, jika diperlukan)

>sketch.ino / main.cpp

```ino
#include <Arduino.h>

const int trigPin = 19;
const int echoPin = 18;

// Define sound speed in cm/uS
#define SOUND_SPEED 0.034
#define CM_TO_INCH 0.393701

long duration;
float distanceCm;
float distanceInch;

void setup() {
  Serial.begin(115200); // Starts the serial communication
  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
  pinMode(echoPin, INPUT); // Sets the echoPin as an Input
}

void loop() {
  // Clears the trigPin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  // Sets the trigPin on HIGH state for 10 micro seconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  // Reads the echoPin, returns the sound wave travel time in microseconds
  duration = pulseIn(echoPin, HIGH);
  // Calculate the distance
  distanceCm = duration * SOUND_SPEED / 2;
  // Convert to inches
  distanceInch = distanceCm * CM_TO_INCH;
  // Prints the distance in the Serial Monitor
  Serial.print("Distance (cm): ");
  Serial.println(distanceCm);
  // Serial.print("Distance (inch): ");
  // Serial.println(distanceInch);
```

```
  delay(1000);
}
```

>diagram.json

```
{
    "version": 1,
    "author": "Najwa Firdaus",
    "editor": "wokwi",
    "parts": [
      {
        "type": "board-esp32-devkit-c-v4",
        "id": "esp",
        "top": -115.2,
        "left": -129.56,
        "attrs": {}
      },
         { "type": "wokwi-hc-sr04", "id": "ultrasonic1", "top": -200.1,
"left": 82.3, "attrs": {} }
    ],
    "connections": [
      [ "esp:TX", "$serialMonitor:RX", "", [] ],
      [ "esp:RX", "$serialMonitor:TX", "", [] ],
      [ "ultrasonic1:GND", "esp:GND.2", "black", [ "v0" ] ],
      [ "ultrasonic1:TRIG", "esp:19", "yellow", [ "v0" ] ],
      [ "ultrasonic1:ECHO", "esp:18", "green", [ "v0" ] ],
      [ "ultrasonic1:VCC", "esp:3V3", "red", [ "v48", "h-278.55" ] ]
    ],
    "dependencies": {}
  }
```