LAPORAN PRAKTIKUM INTERNET OF THINGS (IoT)

Fakultas Vokasi, Universitas Brawijaya

Praktik Pembuatan Sistem Sensor Jarak Ultrasonic Menggunakan Simulator Wokwi ESP 32

M Bimo Amarulloh
Fakultas Vokasi, Universitas Brawijaya
Email: bimoamar@gmail.com

Abstrak

Perkembangan teknologi mikrokontroler memungkinkan integrasi berbagai sensor untuk aplikasi pemantauan dan pengukuran jarak secara real-time. Penelitian ini bertujuan untuk merancang dan mengimplementasikan sistem sensor jarak berbasis sensor ultrasonik HC-SR04 menggunakan simulator Wokwi dengan mikrokontroler ESP32. Wokwi dipilih karena kemampuannya dalam mensimulasikan perangkat keras tanpa memerlukan perangkat fisik, sehingga memudahkan proses pengujian dan pengembangan sistem. Sistem ini dirancang untuk membaca data jarak dari sensor HC-SR04, mengolahnya, dan menampilkannya melalui serial monitor pada lingkungan pengembangan Arduino IDE. Proses perancangan mencakup pemrograman ESP32 menggunakan bahasa C++ dengan library yang sesuai serta pengaturan koneksi antarmuka sensor. Hasil simulasi menunjukkan bahwa sistem mampu membaca dan menampilkan data jarak secara akurat serta responsif terhadap perubahan jarak yang diberikan dalam skenario simulasi. Implementasi ini memberikan wawasan mengenai cara kerja sensor ultrasonik, komunikasi data dengan ESP32, serta efektivitas platform Wokwi dalam simulasi perangkat IoT. Penelitian ini dapat menjadi dasar bagi pengembangan sistem pemantauan jarak berbasis IoT dengan perangkat fisik yang sesungguhnya.

Kata kunci: Sensor ultrasonik, HC-SR04, ESP32, Wokwi, simulasi

Abstract

The rapid development of microcontroller technology enables the integration of various sensors for real-time distance measurement and monitoring applications. This study aims to design and implement a distance sensor system using the HC-SR04 ultrasonic sensor on the Wokwi simulator with an ESP32 microcontroller. Wokwi was chosen for its ability to simulate hardware without requiring physical components, making the testing and development process more efficient. The system is designed to read distance data from the HC-SR04 sensor, process it, and display the results through the serial monitor in the Arduino IDE development environment. The design process includes programming the ESP32 using C++ with the appropriate library and configuring the sensor interface connections. The simulation results show that the system can accurately read and display distance data while responding effectively to changes in distance under simulated conditions. This implementation provides insights into the operation of ultrasonic sensors, data communication with ESP32, and the effectiveness of the Wokwi platform in IoT device simulation. This study serves as a foundation for developing distance monitoring systems using physical IoT devices.

Keywords: Ultrasonic sensor, HC-SR04, ESP32, Wokwi, simulation

1. Introduction (Pendahuluan)

1.1 Latar belakang

Sistem pemantauan jarak memiliki peran penting dalam berbagai bidang, seperti robotika, otomasi, dan keamanan. Dengan kemajuan teknologi, penggunaan simulasi berbasis mikrokontroler semakin berkembang sebagai metode pembelajaran dan pengujian sebelum implementasi di dunia nyata. Salah satu perangkat yang dapat digunakan untuk tujuan ini adalah ESP32, sebuah mikrokontroler dengan kemampuan pemrosesan tinggi dan konektivitas luas. Pada praktik ini, sistem sensor jarak berbasis sensor ultrasonik HC-SR04 dikembangkan dan diuji melalui simulator Wokwi. Wokwi memungkinkan pengguna untuk mensimulasikan rangkaian elektronik dan menjalankan kode tanpa memerlukan perangkat keras fisik, sehingga lebih efisien dalam tahap pengembangan. Selain itu, Arduino IDE digunakan sebagai lingkungan pengembangan untuk menulis program menggunakan bahasa pemrograman C++. Keunikan dari praktik ini terletak pada penerapan sistem pemantauan jarak secara virtual, memungkinkan analisis performa sensor tanpa penggunaan perangkat fisik. Praktik ini bertujuan untuk mengamati bagaimana ESP32 berinteraksi dengan sensor HC-SR04 dalam lingkungan simulasi serta mengevaluasi kestabilan dan akurasi data yang dihasilkan. Hasil dari praktik ini memberikan wawasan mengenai efektivitas Wokwi sebagai alat simulasi sistem pemantauan jarak dan potensi penerapannya dalam pengembangan perangkat IoT. Praktikum ini diharapkan dapat memberikan pemahaman lebih dalam bagi mahasiswa dan praktisi mengenai integrasi sensor dengan mikrokontroler serta manfaat simulasi dalam pengembangan sistem berbasis IoT.

1.2 Tujuan eksperimen

1. Memahami cara kerja sensor ultrasonik HC-SR04 dalam mengukur jarak serta bagaimana data tersebut diproses oleh mikrokontroler ESP32.

2. Mempelajari integrasi sensor HC-SR04 dengan ESP32 melalui simulasi menggunakan Wokwi, tanpa memerlukan perangkat keras fisik.

3. Mengembangkan program berbasis Arduino (C++) untuk membaca dan menampilkan data jarak pada serial monitor.

4. Menguji efektivitas simulator Wokwi sebagai alat bantu dalam pengujian dan pengembangan sistem pemantauan jarak.

5. Menganalisis kestabilan dan akurasi data yang dihasilkan dalam simulasi serta mengevaluasi potensi penerapannya pada perangkat fisik di dunia nyata.

2. Methodology (Metodologi)

2.1 Tools & Materials (Alat dan Bahan)

Alat

1. **Komputer/Laptop** – Digunakan untuk menjalankan simulator dan menulis program.
2. **Koneksi Internet** – Diperlukan untuk mengakses simulator Wokwi secara online.
3. **Simulator Wokwi** – Digunakan untuk mensimulasikan rangkaian elektronik berbasis ESP32.
4. **Visual Studio Code (VS Code)** – Sebagai code editor untuk menulis dan mengedit program Arduino (C++).

Bahan (Komponen Virtual di Wokwi)

1. **Mikrokontroler ESP32** – Sebagai pusat pemrosesan dan pengendali sistem.
2. **Sensor Ultrasonik HC-SR04 –** Digunakan untuk mengukur jarak berdasarkan gelombang ultrasonik.

3. **Kabel Virtual (Wiring dalam Wokwi)** – Untuk menghubungkan ESP32 dengan sensor HC-SR04 secara simulasi.

Karena praktik ini berbasis simulasi di Wokwi, resistor tidak digunakan, dan pengujian tetap dapat dilakukan tanpa risiko kerusakan perangkat keras.

2.2 Implementation Steps (Langkah Implementasi)

1. Persiapan Alat dan Bahan

- Pastikan komputer/laptop sudah terinstall Visual Studio Code (VS Code).
- Akses Wokwi Simulator melalui browser untuk membuat diagram rangkaian.
- Siapkan komponen virtual di Wokwi, yaitu ESP32 dan sensor ultrasonik HC-SR04.

2. Membuat Rangkaian di Wokwi

- Buka Wokwi Simulator dan pilih proyek baru dengan ESP32.
- Tambahkan sensor ultrasonik HC-SR04 untuk membaca jarak.
- Hubungkan HC-SR04 ke ESP32 dengan konfigurasi yang sesuai (VCC, GND, Trig, Echo).
- Simpan diagram rangkaian untuk referensi.

3. Menulis Kode Program di Visual Studio Code

- Buka Visual Studio Code dan buat file baru dengan ekstensi .ino.
- Tulis kode untuk membaca data jarak dari sensor ultrasonik HC-SR04 dan menampilkannya di Serial Monitor.

4. Kode Program Arduino (C++)

Berikut adalah contoh kode untuk mengatur sensotr suhu dan kelembapan:

```
const int trigPin = 5;

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() {
```

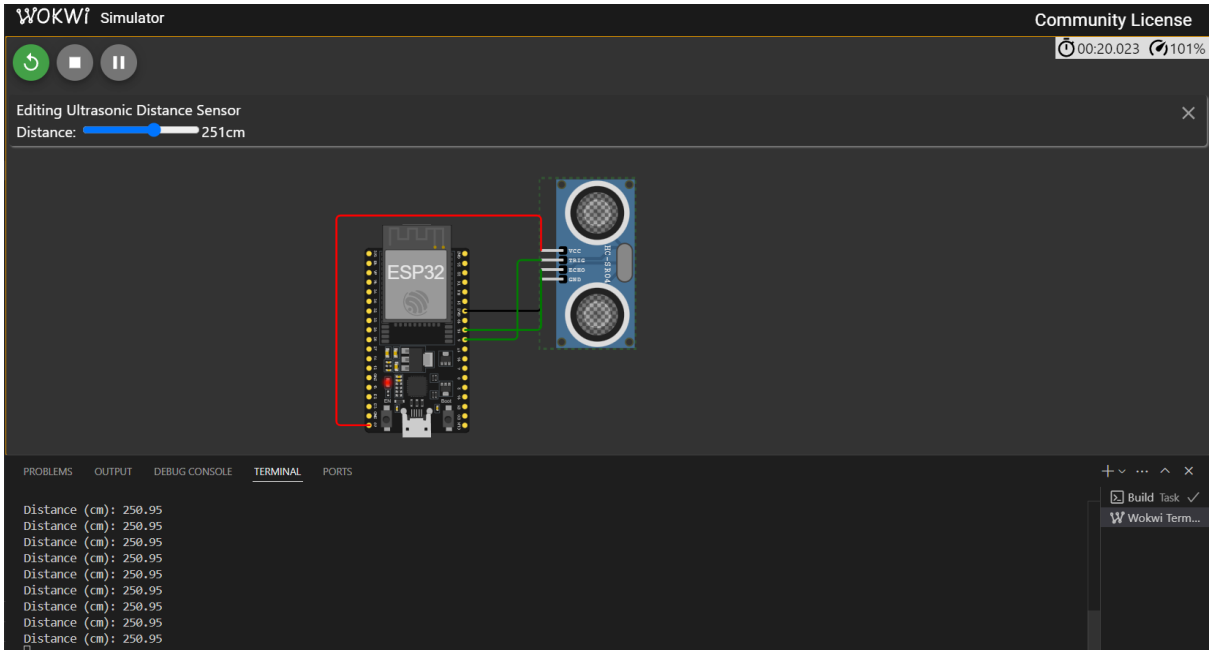
```
 Serial.begin(115200); // Starts the serial communication
 pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
 pinMode(echoPin, INPUT); // Sets the echoPin as an Input
}

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. Mengunggah dan Menjalankan Kode di Wokwi

- Copy kode program ke editor kode di Visual Studio Code
- Klik "Start Simulation" untuk menjalankan simulasi.
- Amati apakah sensor berjalan dengan baik

3. Results and Discussion (Hasil dan Pembahasan)

3.1 Experimental Results (Hasil Eksperimen)



Kode berjalan dengan baik sesuai dengan keinginan peneliti, sensor berjalan dengan melakukan output pemberitahuan panjang/jarak dan terdapat indicator manual untuk menyesuaikan apakah output berubah, dan hasilnya berubah.

4. Appendix (Lampiran)

diagram.json > ...

```
{
  "version": 1,
  "author": "Anonymous maker",
  "editor": "wokwi",
  "parts": [
    { "type": "board-esp32-devkit-c-v4", "id": "esp", "top": 0, "left": 0, "attrs": {} },
    {
      "type": "wokwi-hc-sr04",
      "id": "ultrasonic1",
      "top": -8.1,
      "left": 139.9,
      "rotate": 90,
      "attrs": {}
    }
  ],
  "connections": [
    [ "esp:TX", "$serialMonitor:RX", "", [] ],
    [ "esp:RX", "$serialMonitor:TX", "", [] ],
    [ "ultrasonic1:GND", "esp:GND.3", "black", [ "v31.2", "h-76.64" ] ],
    [ "ultrasonic1:ECHO", "esp:18", "green", [ "v60.4", "h-76.64" ] ],
    [ "ultrasonic1:TRIG", "esp:5", "green", [ "h-23.8", "v80" ] ],
    [ "ultrasonic1:VCC", "esp:5V", "red", [ "v-34.8", "h-206.2", "v211.2" ] ]
  ],
  "dependencies": {}
}
```

Gambar 1. 1 Syntax Diagram ESP 32

src > main.cpp > ...

```cpp
#include <Arduino.h>

const int trigPin = 5;
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): ");
```

Gambar 1. 2 Syntax main.cpp