**Simulasi Monitoring Cuaca Berbasis Internet of Things (IoT) Menggunakan ESP32, LCD I2C, dan API OpenWeather**

**Abstrak**

Praktikum ini mengembangkan sistem pemantauan cuaca berbasis Internet of Things (IoT) yang memanfaatkan mikrokontroler ESP32 untuk mengambil data cuaca secara real-time dari API OpenWeather. Data yang diperoleh meliputi suhu, suhu terasa, curah hujan, kecepatan dan arah angin, kelembapan, tutupan awan, tekanan udara, serta waktu matahari terbit dan terbenam. Informasi tersebut ditampilkan secara bergantian pada LCD 16x2 melalui komunikasi I2C, dengan navigasi menggunakan tombol Next dan Prev untuk melakukan scrolling antar data. Simulasi dilakukan di platform Wokwi, sehingga seluruh proses dapat diuji tanpa perangkat keras fisik. Hasil simulasi menunjukkan sistem mampu menampilkan data cuaca secara dinamis dan responsif sesuai urutan yang diatur.

Kata Kunci: IoT, ESP32, LCD I2C, API OpenWeather, Wokwi

**Pendahuluan**

**LatarBelakang**Internet of Things (IoT) telah membuka peluang besar dalam pengembangan sistem pemantauan lingkungan secara real-time. Salah satu penerapannya adalah monitoring data cuaca dengan memanfaatkan layanan API publik seperti OpenWeather. Dengan menggunakan ESP32 yang terhubung ke jaringan Wi-Fi, sistem dapat mengambil data cuaca terkini dan menampilkannya pada LCD 16x2. Pengguna dapat menavigasi data menggunakan tombol fisik, sehingga informasi yang ditampilkan dapat diakses sesuai kebutuhan.

**TujuanPraktikum**Praktikum ini bertujuan untuk:

* Memahami dasar pemanfaatan API dalam sistem IoT.
* Mengimplementasikan pengambilan dan penampilan data cuaca dari API OpenWeather pada LCD 16x2 menggunakan ESP32.
* Mengaplikasikan navigasi data menggunakan tombol Next dan Prev.
* Melakukan simulasi sistem monitoring cuaca menggunakan platform Wokwi.

**Metodologi**

Alat dan Bahan

* Laptop/komputer
* Simulator Wokwi
* Visual Studio Code dengan PlatformIO
* ESP32 Devkit V1 (virtual)
* LCD 16x2 I2C
* Dua tombol pushbutton (Next dan Prev)
* Koneksi Wi-Fi (Wokwi Guest)

Langkah Implementasi

* Membuat proyek baru di Wokwi dengan ESP32 Devkit V1.
* Menghubungkan LCD I2C ke pin D21 (SDA) dan D22 (SCL) ESP32.
* Menyambungkan tombol Next ke pin D12 dan tombol Prev ke pin D14, masing-masing ke GND.
* Menulis dan mengunggah kode program pada Wokwi Web.
* Memulai simulasi dan memastikan LCD menampilkan data cuaca serta tombol berfungsi untuk scrolling.
* (Opsional) Melanjutkan simulasi di Visual Studio Code dengan PlatformIO.

**Hasil dan Pembahasan**

Sistem monitoring cuaca berbasis ESP32 berhasil disimulasikan pada platform Wokwi. Setelah sistem menyala, LCD menampilkan identitas pembuat sebagai salam pembuka, kemudian melakukan koneksi ke jaringan Wi-Fi. Selanjutnya, data cuaca diambil dari API OpenWeather dan ditampilkan satu per satu pada LCD. Navigasi antar data dilakukan dengan tombol Next dan Prev, memungkinkan pengguna melihat seluruh informasi cuaca secara berurutan.

Data yang ditampilkan meliputi:

* Suhu udara
* Suhu terasa
* Curah hujan
* Kecepatan angin
* Arah angin
* Kelembapan
* Tutupan awan
* Tekanan udara
* Waktu matahari terbit
* Waktu matahari terbenam

Konversi waktu sunrise dan sunset dilakukan dari UNIX Timestamp ke waktu lokal (WIB) dengan penyesuaian zona waktu. Parsing data JSON dari API dilakukan secara manual tanpa library tambahan, sehingga program lebih ringan. Karena simulasi dilakukan di Wokwi, data yang diterima bersifat statis (dummy), sehingga nilai parameter cuaca tidak berubah-ubah seperti pada perangkat asli yang terhubung ke internet.

Secara keseluruhan, sistem mampu menampilkan data cuaca secara dinamis dan responsif, serta mendemonstrasikan integrasi antara ESP32, API OpenWeather, dan LCD I2C dalam aplikasi IoT monitoring lingkungan

**Lampiran**

**Kode Program (main.cpp)**

#include <Wire.h>

#include <LiquidCrystal\_I2C.h>

#include <WiFi.h>

#include <HTTPClient.h>

const char\* ssid = "Wokwi-GUEST";

const char\* password = "";

String apiKey = "0c854b30810ba8e3e1f55b2eeabe6da6";

String city = "Malang";

String units = "metric";

String server = "http://api.openweathermap.org/data/2.5/weather?q=" + city + "&units=" + units + "&appid=" + apiKey;

LiquidCrystal\_I2C lcd(0x27, 16, 2);

const int buttonNext = 12;

const int buttonPrev = 14;

int scrollIndex = 0;

unsigned long lastUpdate = 0;

unsigned long updateInterval = 60000;

// Data weather

String temp = "N/A";

String feels\_like = "N/A";

String precipitation = "0";

String windSpeed = "N/A";

String windDirection = "N/A";

String humidity = "N/A";

String clouds = "N/A";

String pressure = "N/A";

String sunrise = "N/A";

String sunset = "N/A";

// Function prototypes

void fetchWeather();

void displayData(int index);

String getValue(String data, String key);

String getOptionalValue(String data, String key);

String formatTime(String unixTimeStr);

void setup() {

Serial.begin(115200);

lcd.init();

lcd.backlight();

pinMode(buttonNext, INPUT\_PULLUP);

pinMode(buttonPrev, INPUT\_PULLUP);

// Tampilan awal identitas

lcd.setCursor(0, 0);

lcd.print("Created by");

lcd.setCursor(0, 1);

lcd.print("TB Rangga Gilang");

delay(2000);

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Yanuari");

delay(2000);

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Connecting...");

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Connected!");

delay(1000);

lcd.clear();

fetchWeather(); // Fetch pertama

displayData(scrollIndex);

}

void loop() {

unsigned long currentMillis = millis();

// Update data tiap 60 detik

if (currentMillis - lastUpdate > updateInterval) {

lastUpdate = currentMillis;

fetchWeather();

displayData(scrollIndex);

}

// Tombol Next

if (digitalRead(buttonNext) == LOW) {

scrollIndex++;

if (scrollIndex > 9) scrollIndex = 0;

displayData(scrollIndex);

delay(300); // Debounce

}

// Tombol Prev

if (digitalRead(buttonPrev) == LOW) {

scrollIndex--;

if (scrollIndex < 0) scrollIndex = 9;

displayData(scrollIndex);

delay(300); // Debounce

}

}

void fetchWeather() {

if (WiFi.status() == WL\_CONNECTED) {

HTTPClient http;

http.begin(server);

int httpCode = http.GET();

if (httpCode > 0) {

String payload = http.getString();

Serial.println(payload);

temp = getValue(payload, "temp");

feels\_like = getValue(payload, "feels\_like");

humidity = getValue(payload, "humidity");

windSpeed = getValue(payload, "speed");

windDirection = getValue(payload, "deg");

pressure = getValue(payload, "pressure");

clouds = getValue(payload, "all");

precipitation = getOptionalValue(payload, "\"rain\":{\"1h\":");

sunrise = formatTime(getValue(payload, "sunrise"));

sunset = formatTime(getValue(payload, "sunset"));

} else {

Serial.println("Error on HTTP request");

}

http.end();

}

}

void displayData(int index) {

lcd.clear();

lcd.setCursor(0, 0);

switch(index) {

case 0:

lcd.print("Temp: " + temp + "C");

break;

case 1:

lcd.print("Feels: " + feels\_like + "C");

break;

case 2:

lcd.print("Precip: " + precipitation + "mm");

break;

case 3:

lcd.print("Wind: " + windSpeed + "m/s");

break;

case 4:

lcd.print("Dir: " + windDirection + (char)223);

break;

case 5:

lcd.print("Humidity: " + humidity + "%");

break;

case 6:

lcd.print("Clouds: " + clouds + "%");

break;

case 7:

lcd.print("Pressure: " + pressure + "hPa");

break;

case 8:

lcd.print("Sunrise:");

lcd.setCursor(0, 1);

lcd.print(sunrise);

break;

case 9:

lcd.print("Sunset:");

lcd.setCursor(0, 1);

lcd.print(sunset);

break;

}

}

// Fungsi parsing data biasa

String getValue(String data, String key) {

int startIndex = data.indexOf(key);

if (startIndex == -1) return "N/A";

startIndex = data.indexOf(":", startIndex) + 1;

int endIndex = data.indexOf(",", startIndex);

if (endIndex == -1) endIndex = data.indexOf("}", startIndex);

String value = data.substring(startIndex, endIndex);

value.trim();

return value;

}

// Fungsi parsing rain opsional

String getOptionalValue(String data, String key) {

int start = data.indexOf(key);

if (start == -1) return "0";

start += key.length();

int end = data.indexOf(",", start);

if (end == -1) end = data.indexOf("}", start);

String value = data.substring(start, end);

value.trim();

return value;

}

// Format UNIX Time ke WIB jam:menit

String formatTime(String unixTimeStr) {

unsigned long unixTime = unixTimeStr.toInt();

unixTime += 7 \* 3600;

int hours = (unixTime % 86400L) / 3600;

int minutes = (unixTime % 3600) / 60;

char buffer[6];

sprintf(buffer, "%02d:%02d", hours, minutes);

return String(buffer);

}

Diagram JSON untuk simulasi sistem monitoring cuaca berbasis ESP32 dengan LCD I2C dan tombol Next/Prev yang digunakan di platform Wokwi adalah sebagai berikut:

{

"version": 1,

"author": "awikwaok",

"editor": "wokwi",

"parts": [

{ "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": 0, "left": 0, "attrs": {} },

{

"type": "wokwi-lcd1602",

"id": "lcd1",

"top": 35.2,

"left": 255.2,

"attrs": { "pins": "i2c" }

},

{

"type": "wokwi-pushbutton",

"id": "btn1",

"top": 303.8,

"left": -67.2,

"attrs": { "color": "red", "xray": "1", "label": "Prev" }

},

{

"type": "wokwi-pushbutton",

"id": "btn2",

"top": 303.8,

"left": 115.2,

"attrs": { "color": "blue", "xray": "1", "label": "Next" }

}

],

"connections": [

[ "esp:TX0", "$serialMonitor:RX", "", [] ],

[ "esp:RX0", "$serialMonitor:TX", "", [] ],

[ "lcd1:SCL", "esp:D22", "green", [ "h-124.8", "v-66.9" ] ],

[ "lcd1:VCC", "esp:3V3", "red", [ "h-124.8", "v86.5" ] ],

[ "lcd1:GND", "esp:GND.1", "black", [ "h-144", "v57.6" ] ],

[ "lcd1:SDA", "esp:D21", "green", [ "h-134.4", "v-28.6" ] ],

[ "btn2:2.l", "esp:GND.2", "black", [ "h-28.8", "v-96", "h-172.8" ] ],

[ "btn2:1.l", "esp:D12", "blue", [ "h-19.2", "v-86.4", "h-105.6", "v-105.6" ] ],

[ "btn1:1.l", "esp:D14", "red", [ "h0", "v-28.8", "h48", "v-163.2" ] ],

[ "btn1:2.l", "esp:GND.2", "black", [ "h-9.6", "v-67", "h86.4" ] ]

],

"dependencies": {}

}