Buku Petunjuk Penggunaan Aplikasi (User Manual)
IR AC REMOTE WITH WEB INFTEFACE AND ESP32

Dibuat Oleh:
Hansen Brian

EMMERICH RESEARCH CENTER
PLUIT
2023

BAB I
PENDAHULUAN

1.1 Tujuan Pembuatan Dokumen

Dokumen ini dibuat dengan tujuan utama untuk memberikan panduan komprehensif kepada pengguna mengenai penggunaan aplikasi IR AC Remote dengan Antarmuka Web dan ESP32. Tujuannya meliputi:

- Panduan Langkah demi Langkah: Memberikan instruksi yang jelas dan terperinci tentang cara mengistal, mengonfigurasi, dan menggunakan aplikasi dengan efektif.

- Pemahaman tentang Konteks Proyek: Menjelaskan latar belakang proyek kerja praktik serta konteksnya, sehingga pengguna dapat memahami tujuan dan manfaat aplikasi.

- Mengatasi Masalah: Menyediakan panduan troubleshooting untuk membantu pengguna mengatasi masalah yang mungkin terjadi saat menggunakan aplikasi, meminimalkan hambatan dalam penggunaan.

1.2 Informasi Teknis dan Fungsionalitas

Dokumen ini memberikan informasi teknis yang mendalam tentang komponen yang digunakan dalam proyek, termasuk perangkat ESP32, protokol ESP-NOW, antarmuka web, dan sinyal IR. Informasi ini bertujuan untuk:

- Memahami Teknologi yang Digunakan: Memberikan pemahaman tentang bagaimana komponen bekerja bersama untuk menciptakan aplikasi yang fungsional.

- Meningkatkan Penggunaan Efektif: Menjelaskan fungsionalitas aplikasi secara rinci, termasuk tombol-tombol dan fitur-fitur antarmuka web, sehingga pengguna dapat memaksimalkan manfaat dari aplikasi ini.

- Mendefinisikan Burst Protocol: Menjelaskan penggunaan Burst Protocol untuk mengirim sinyal IR dengan berbagai protokol, memastikan bahwa perangkat dapat berkomunikasi dengan berbagai jenis AC.

1.3 Deskripsi Dokumen

Dokumen ini dibuat untuk memberikan panduan penggunaan aplikasi **IR AC Remote Web Interface**. Dokumen ini berisikan informasi sebagai berikut:

1. BAB I.
   Berisi informasi umum yang merupakan bagian pendahuluan, yang meliputi tujuan pembuatan dokumen, deskripsi umum sistem serta deskripsi dokumen.
2. BAB II
   Berisi perangkat yang dibutuhkan untuk penggunaan aplikasi meliputi perangkat lunak dan perangkat hardware
3. BAB III
   Berisi *user manual* aplikasi

BAB II
PERANGKAT YANG DIBUTUHKAN

Dalam bab ini, Anda akan memahami perangkat lunak dan perangkat keras yang diperlukan untuk mengimplementasikan aplikasi IR AC Remote dengan Antarmuka Web dan ESP32.

2.1 Perangkat Lunak

Pada bagian ini, Anda perlu mempersiapkan perangkat lunak yang diperlukan untuk mengembangkan dan menjalankan aplikasi di laptop. Beberapa perangkat lunak penting termasuk:

2.1.1 Arduino IDE/Visual Studio Code

Anda memerlukan perangkat lunak pengembangan seperti Arduino IDE atau Visual Studio Code (VS Code) untuk menulis, mengedit, dan mengunggah kode program ke perangkat ESP32.

2.1.2 Web Browser (Google Chrome, EDGE, dll.)

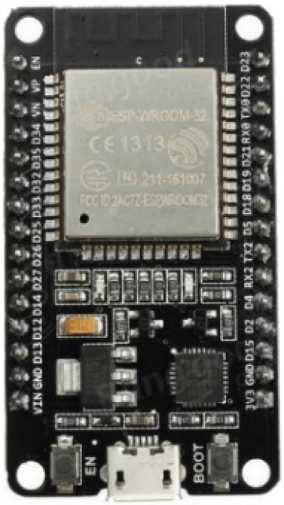
Sebuah web browser seperti Google Chrome atau Microsoft Edge akan digunakan untuk mengakses antarmuka web yang di-host oleh perangkat ESP32. Melalui web browser, Anda dapat berinteraksi dengan perangkat dan mengontrol perangkat AC melalui antarmuka yang telah dibuat.

2.2 Perangkat Keras

Dalam pengembangan aplikasi ini, Anda akan memerlukan perangkat keras tertentu untuk menjalankan sistem IR AC Remote dengan Antarmuka Web. Beberapa perangkat keras yang diperlukan meliputi:

2.2.1 ESP32 Board

Perangkat utama yang digunakan adalah modul ESP32. ESP32 akan bertindak sebagai pusat kendali yang menghubungkan antarmuka web dengan perangkat AC melalui sinyal IR. Pastikan Anda memiliki 2 buah ESP32 board yang kompatibel dengan proyek ini.

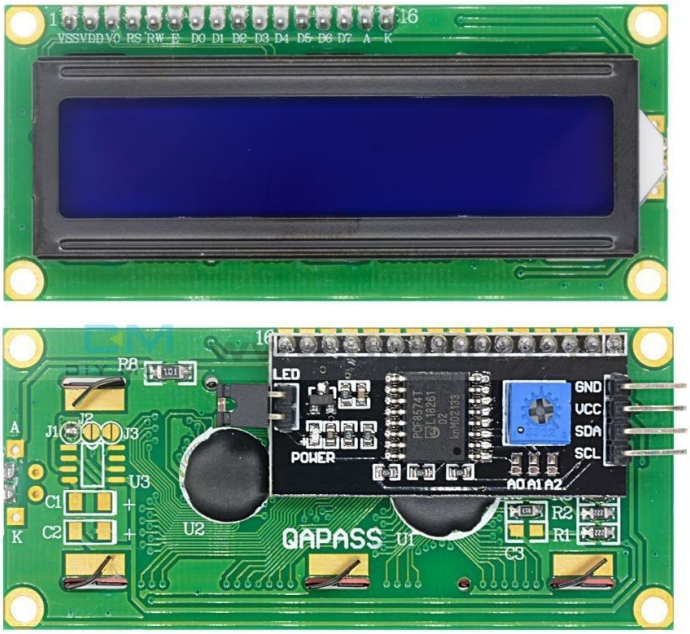


2.2.2 IR LED

IR LED digunakan untuk menghasilkan sinyal inframerah yang diperlukan untuk mengontrol perangkat AC. Pastikan Anda memiliki IR LED yang cocok untuk penggunaan ini.

2.2.3 I2C LCD 16x2

LCD 16x2 dengan koneksi I2C akan digunakan untuk menampilkan informasi dan status dari sistem. Pastikan Anda memiliki I2C LCD 16x2 yang sesuai dengan proyek ini.



2.2.4 Komponen Elektronik Lainnya

Selain ESP32, IR LED, dan I2C LCD 16x2, Anda mungkin juga memerlukan komponen tambahan seperti resistor, kabel jumper, breadboard, dan lainnya untuk merakit perangkat keras proyek ini.

Catatan:
Sebelum memulai proyek, pastikan Anda telah mengumpulkan semua perangkat lunak dan perangkat keras yang disebutkan di atas. Dengan memiliki semua peralatan yang diperlukan, Anda akan siap untuk memulai pengembangan dan mengikuti panduan yang diberikan dalam buku petunjuk ini.

BAB III
TEORI SINGKAT

Dalam bab ini, Anda akan diajarkan tentang cara menggunakan aplikasi IR AC Remote dengan Antarmuka Web dan ESP32. Bagian pertama akan menjelaskan tentang sistem aplikasi secara keseluruhan.

3.1 Alur Sistem

Dalam bagian ini, Anda akan mendapatkan pemahaman tentang alur sistem dari aplikasi IR AC Remote dengan Antarmuka Web dan ESP32. Alur sistem ini menjelaskan bagaimana komponen-komponen saling berhubungan dan beroperasi bersama untuk mengontrol perangkat AC melalui antarmuka web. Alur sistem ini mencakup:

1. **Konektivitas Sistem**: Bagian ini menjelaskan bagaimana ESP32-Master terhubung ke internet melalui koneksi Wi-Fi, dan bagaimana ESP-Slave terhubung ke ESP32-Master melalui jaringan lokal yang dibentuk saat ESP32-Master beroperasi sebagai Access Point.

2. **Komunikasi Antar Komponen**: Bagian ini menjelaskan tentang penggunaan protokol ESP-NOW untuk komunikasi antara ESP32-Master dan ESP-Slave. Protokol ini memungkinkan komunikasi langsung dan cepat antara perangkat ESP tanpa memerlukan jaringan Wi-Fi.

3. **Aliran Data dalam Sistem**: Bagian ini menjelaskan bagaimana pengguna mengakses Web Server yang di-host oleh ESP32-Master melalui local IP untuk berinteraksi dengan sistem. Saat pengguna menekan tombol pada antarmuka Web Server, data perintah dikirimkan dari ESP32-Master ke ESP-Slave melalui protokol ESP-NOW. ESP-Slave menerima data perintah dan memprosesnya sesuai dengan fungsi yang telah ditentukan. Setelah data diproses, ESP-Slave akan menggunakan sinyal Infrared (IR) untuk mengontrol perangkat AC melalui IR LED yang terhubung kepadanya.

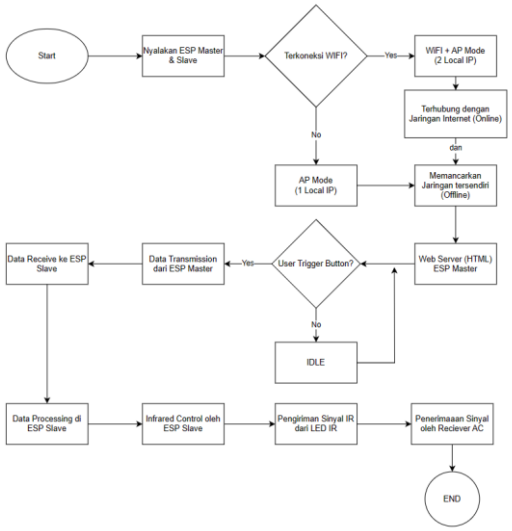


Figure 1 Diagram Garis Besar Proyek IR AC Remote

3.2 WiFi dan Mode Akses Point

Dalam bagian ini, Anda akan mempelajari tentang penggunaan koneksi Wi-Fi dan mode akses point dalam aplikasi IR AC Remote dengan Antarmuka Web dan ESP32.

3.2.1 Pengenalan WiFi dan Mode Akses Point

Pertama-tama, Anda akan memahami pentingnya koneksi Wi-Fi dan mode akses point dalam konteks aplikasi ini. Koneksi Wi-Fi memungkinkan ESP32-Master untuk terhubung ke internet dan berkomunikasi dengan perangkat lain melalui jaringan. Mode akses point memungkinkan ESP32-Master beroperasi sebagai titik akses nirkabel, memungkinkan perangkat lain terhubung dan berkomunikasi langsung dengan ESP32-Master.

3.2.2 Library WiFi dan Penggunaannya

Dalam bagian ini, Anda akan mengenal penggunaan library WiFi pada ESP32 untuk mengelola koneksi Wi-Fi dan mode akses point. Anda akan melihat cara mendeklarasikan library WiFi dan bagaimana menggunakan fungsi-fungsi terkait untuk menghubungkan ESP32-Master ke jaringan Wi-Fi yang ada atau memulai mode akses point.

```
// Load Wi-Fi library
#include <WiFi.h>
```

3.2.3 Menghubungkan ke Jaringan Wi-Fi

Langkah selanjutnya adalah memahami cara menghubungkan ESP32-Master ke jaringan Wi-Fi yang ada. Anda akan melihat bagaimana mendeklarasikan nama dan kata sandi Wi-Fi dalam kode, serta bagaimana menginisiasi proses koneksi ke jaringan Wi-Fi tersebut.

3.2.4 Mode Akses Point

Dalam bagian ini, Anda akan mempelajari cara mengaktifkan mode akses point pada ESP32-Master. Mode akses point memungkinkan ESP32-Master menciptakan jaringan lokal sendiri dengan nama dan kata sandi yang telah ditentukan. Anda akan melihat bagaimana mengatur konfigurasi jaringan pada mode akses point dan memberikan petunjuk pada pengguna tentang status mode akses point.

```
//============================================================
// Wifi network credenials
const char *wifi_ssid = "Zereixus"; //--> wifi name
const char *wifi_password = "z4x2ds2z"; //--> wifi password

// Access Point Declaration and Configuration.
const char* soft_ap_ssid = "ESP32_WS";  //--> access point name
const char* soft_ap_password = "helloesp32WS"; //--> access point password

IPAddress local_ip(192,168,1,1);
IPAddress gateway(192,168,1,1);
IPAddress subnet(255,255,255,0);
//============================================================
```

3.3 ESP-NOW, HTML User Interface, dan Pengelolaan Data

Dalam bagian ini, Anda akan mempelajari tentang komunikasi melalui protokol ESP-NOW, pembuatan antarmuka web menggunakan HTML, dan pengelolaan data dari dan ke perangkat.

3.3.1 Komunikasi melalui Protokol ESP-NOW

Anda akan menjelajahi protokol komunikasi nirkabel ESP-NOW yang memungkinkan komunikasi langsung dan efisien antara perangkat ESP. Anda akan melihat bagaimana menginisiasi komunikasi, mendaftarkan perangkat sebagai peer, dan mengirim serta menerima data antara ESP32-Master dan ESP-Slave.

```
// Load Master-Slave library
#include <esp_now.h>
#include <esp_wifi.h>
```

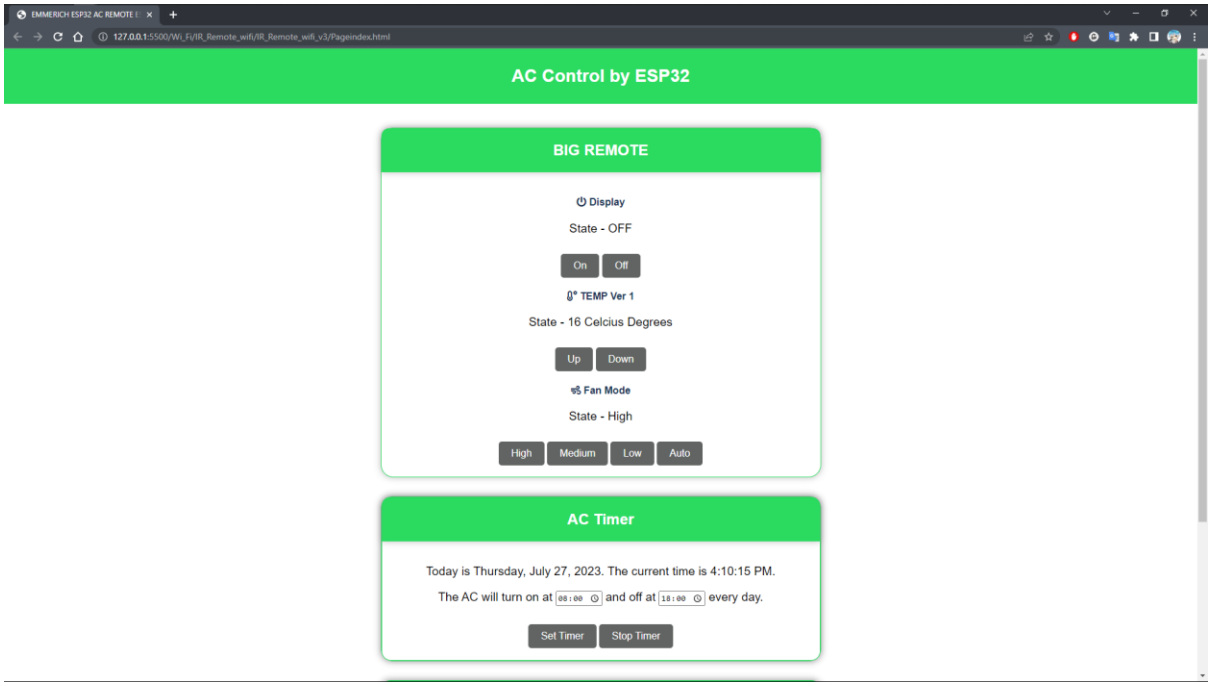
```
// Create a variable of type "esp_now_peer_info_t" to store information about the peer.
esp_now_peer_info_t peerInfo;
```

3.3.2 Pembuatan Antarmuka Web dengan HTML

Bagian ini akan membahas pembuatan antarmuka web menggunakan HTML. Anda akan memahami bagaimana memasukkan kode HTML ke dalam program Anda, serta mengatur tampilan dan interaksi antarmuka web. Antarmuka web ini akan digunakan oleh pengguna untuk mengontrol perangkat AC.

```
// Include the contents of the User Interface Web page, stored in the same folder as the .ino file
#include "PageIndex.h"
```

```
const char MAIN_page[] PROGMEM = R"=====(
<!DOCTYPE HTML>
<html>
  <head>
    <title>ESP32 ESP-NOW & WEB SERVER (CONTROLLING)</title>
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.2
    <link rel="icon" href="data:,">
    <style>
      html {font-family: Arial; display: inline-block; text-align: center;}
      p {font-size: 1.2rem;}
```



3.3.3 Pengelolaan Data dari dan ke Perangkat

Anda akan belajar tentang pengelolaan data yang dikirim dari antarmuka web ke perangkat ESP32-Master. Data perintah yang diterima dari pengguna akan diolah dan dikirimkan ke ESP-Slave melalui protokol ESP-NOW. Selain itu, Anda juga akan melihat bagaimana perangkat ESP-Slave memproses data dan mengirim respons kembali ke ESP32-Master.

```
//WEB PARAM
// ======================================== The variables used to check the parameters passed in the URL.
// Look in the "PageIndex.h" file.
// "set_LED?board="+board+"&gpio_output="+gpio+"&val="+value
// For example :
// set_LED?board=ESP32Slave1&gpio_output=13&val=1
// PARAM_INPUT_1 = ESP32Slave1
// PARAM_INPUT_2 = 13
// PARAM_INPUT_3 = 1
const char* PARAM_INPUT_1 = "board";
const char* PARAM_INPUT_2 = "gpio_output";
const char* PARAM_INPUT_3 = "val";
// ========================================
```

```
// ---------------------------------------------------------------- XMLHttpRequest to submit data.
function send_cmd(board,gpio,value) {
  var xhr = new XMLHttpRequest();
  xhr.open("GET", "set_LED?board="+board+"&gpio_output="+gpio+"&val="+value, true);
  xhr.send();
}
// ----------------------------------------------------------------
```

```
// ======================================== Structure to send data
// Send Data by struct
typedef struct struct_message_send {
  int send_GPIO_num;
  int send_Val;
} struct_message_send;

struct_message_send send_Data; //--> Create a struct_message to send data.
// ========================================
```

```
// ======================================== Structure example to receive data.
// Must match the sender structure
typedef struct struct_message_receive {
  int receive_GPIO_num;
  int receive_Val;
} struct_message_receive;

// Create a struct_message to receive data.
struct_message_receive receive_Data;
// ========================================
```

3.3.4 Burst Protocol

Dalam bagian ini, Anda akan diajarkan tentang Burst Protocol, yaitu cara untuk menguji protokol sinyal inframerah (IR) yang sesuai dengan perangkat AC yang berbeda-beda. Anda akan melihat bagaimana Burst Protocol digunakan untuk mencoba setiap protocol yang ada pada library IR hingga sinyal diterima dengan benar oleh perangkat AC yang tepat.

```
for (int i = 1; i < kLastDecodeType; i++) {
  decode_type_t protocol = (decode_type_t)i;
  // If the protocol is supported by the IRac class ...
  if (ac.isProtocolSupported(protocol)) {
    Serial.println("Protocol " + String(protocol) + " / " +
                   typeToString(protocol) + " is supported.");
    ac.next.protocol = protocol;  // Change the protocol used.
    Serial.println("Sending a message to the A/C unit.");
    lcd.clear();
    lcd.print("Sending");
    lcd.setCursor(0,1);
    lcd.print(typeToString(protocol));
    ac.sendAc();  // Have the IRac class create and send a message.
    delay(1000);  // Wait 1 seconds.
  }
}
```

Catatan:
Penting untuk memahami bagaimana ESP-NOW digunakan untuk komunikasi antara perangkat ESP, serta bagaimana data dikirim dan diterima. Selain itu, pahami bagaimana membuat antarmuka web yang mudah digunakan oleh pengguna, serta cara mengelola data yang masuk dan keluar dari perangkat. Terakhir, Burst Protocol akan membantu Anda menentukan protocol yang tepat untuk mengontrol perangkat AC.

BAB IV
PANDUAN PENGGUNAAN

4.1 Cara Mengunduh File dari GitHub

Langkah pertama dalam menggunakan aplikasi ini adalah mengunduh file proyek dari repositori GitHub. Repositori GitHub berisi semua kode dan file yang diperlukan untuk mengembangkan aplikasi ini. Berikut adalah langkah-langkahnya:

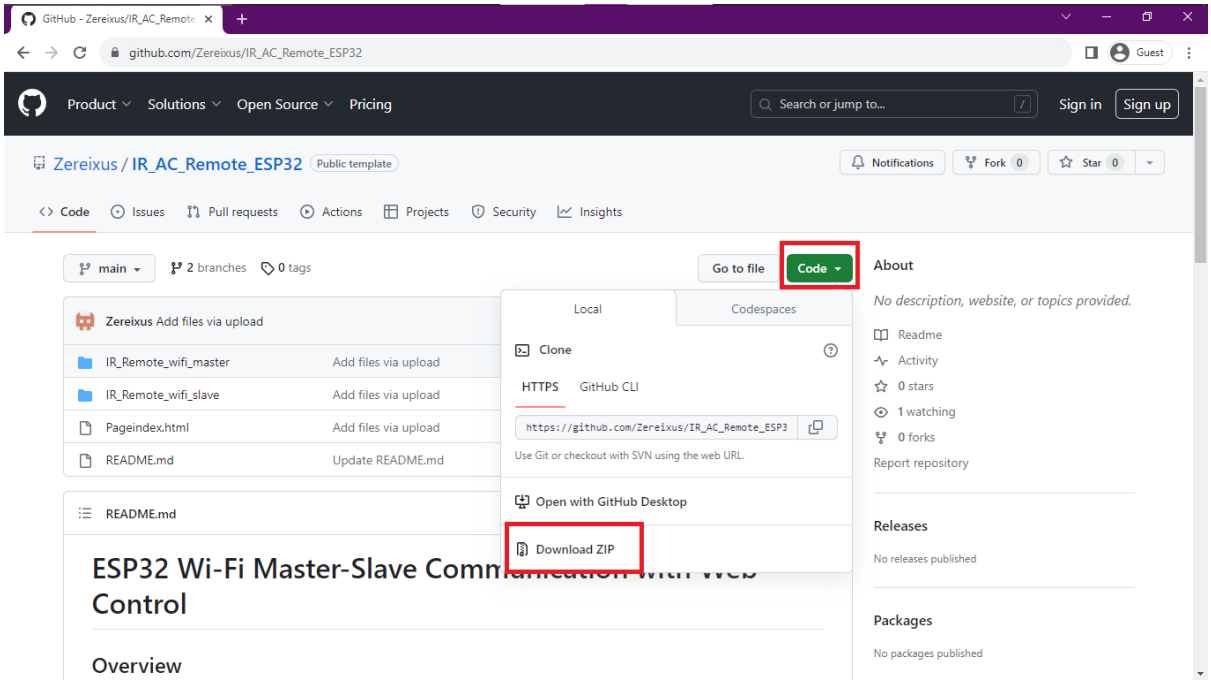
1. Buka Navigasi Web: Buka browser web di komputer Anda dan arahkan ke alamat GitHub repositori proyek yaitu https://github.com/Zereixus/IR_AC_Remote_ESP32.

2. Pilih Branch: Pilih branch Main.

3. Klik Tombol "Code": Di halaman repositori, cari tombol yang disebut "Code" atau "Download". Klik tombol ini untuk membuka opsi unduhan.

4. Pilih Opsi Download: Dalam opsi unduhan, Anda dapat memilih untuk mengunduh repositori dalam bentuk ZIP atau menggunakan Git untuk mengkloning repositori. Pilih opsi yang sesuai dengan preferensi Anda.

5. Ekstrak ZIP File: Jika Anda memilih opsi ZIP, unduh dan ekstrak file ZIP yang diunduh ke direktori yang diinginkan di komputer Anda.



4.2 Konfiguasi ESP

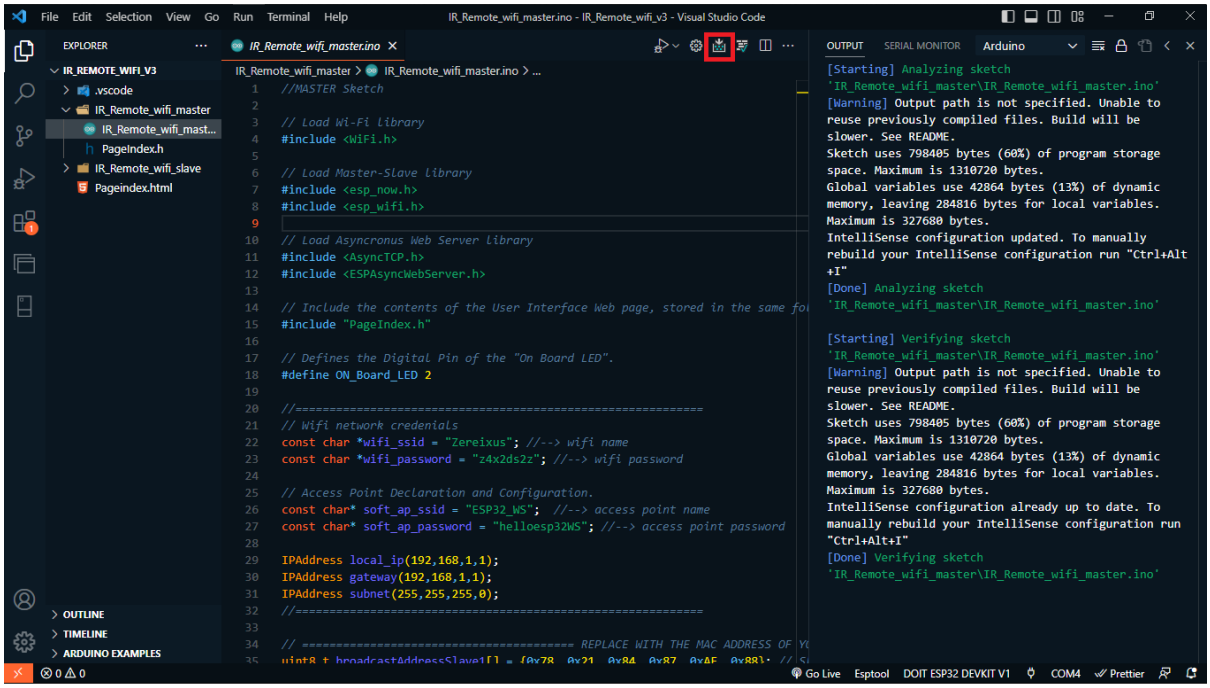
Dalam bagian ini, Anda akan dipandu tentang bagaimana melakukan konfigurasi ESP32 untuk aplikasi IR AC Remote dengan Antarmuka Web dan ESP32. Konfigurasi ini melibatkan mengunggah kode ke ESP32-Master dan ESP-Slave, serta melakukan beberapa pengaturan khusus.

4.2.1 Master ESP

4.2.1.1 Mengunggah Kode Melalui Visual Studio Code

Pertama, Anda perlu mengunggah kode program ke ESP32-Master menggunakan Visual Studio Code:

1. **Buka Proyek:** Buka proyek yang telah Anda unduh pada Visual Studio Code.
2. **Hubungkan ESP32:** Hubungkan ESP32-Master ke komputer Anda melalui kabel USB.
3. **Compile dan Unggah:** Buka berkas kode untuk ESP32-Master. Kemudian, kompile kode dengan mengklik tombol "Build" (ikon palu) di bagian bawah jendela. Setelah kompilasi selesai, unggah kode ke ESP32-Master dengan mengklik tombol "Upload" (ikon panah mengarah ke atas).
4. **Monitor Serial:** Setelah unggah selesai, buka "Serial Monitor" pada Visual Studio Code untuk melihat keluaran pesan dari ESP32-Master. Pastikan tidak ada pesan kesalahan.



4.2.1.2 Pengenalan Peer dan MAC Address

Anda perlu melakukan pengenalan peer untuk ESP32-Slave pada ESP32-Master:

1. **Dapatkan MAC Address Slave:** Buka kode ESP32-Slave dan temukan MAC address yang diberikan pada bagian "Variable penyimpan informasi Peer".
2. **Deklarasikan MAC Address:** Di kode ESP32-Master, temukan bagian "Variable pernyimpanan MAC ADDRESS untuk Receivers". Deklarasikan MAC address ESP32-Slave dengan nama yang relevan.
3. **Daftarkan Peer:** Pada kode ESP32-Master, temukan fungsi "registerPeer()" dan daftarkan ESP32-Slave sebagai peer menggunakan MAC address yang telah dideklarasikan.

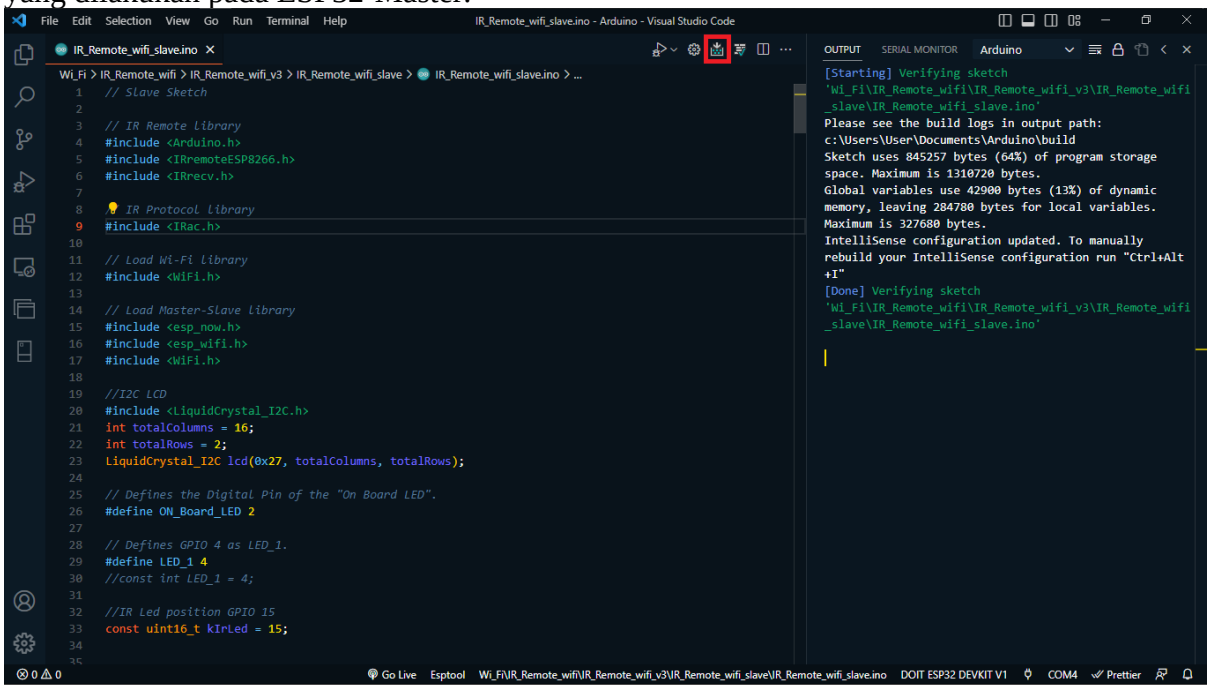
```
// ========================================= REPLACE WITH THE MAC ADDRESS OF YOUR SLAVES / RECEIVERS / ESP32 RECEIVERS.
uint8_t broadcastAddressSlave1[] = {0x78, 0x21, 0x84, 0x87, 0xAF, 0x88}; // Slave 1 MAC Address 78:21:84:87:AF:88
uint8_t broadcastAddressSlave2[] = {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF}; // Slave 2 MAC Address ---
```

```ino
void registerPeer()
{
  Serial.println();
  Serial.println("-------------");
  Serial.println("Register peer");
  peerInfo.encrypt = false;
  // :::::::::::::::: register first peer
  Serial.println("Register first peer (ESP32 Slave 1)");
  memcpy(peerInfo.peer_addr, broadcastAddressSlave1, 6);
  if (esp_now_add_peer(&peerInfo) != ESP_OK){
    Serial.println("Failed to add first peer");
    return;
  }
  // ::::::::::::::::
  // :::::::::::::::: register second peer
  Serial.println("Register second peer (ESP32 Slave 2)");
  memcpy(peerInfo.peer_addr, broadcastAddressSlave2, 6);
  if (esp_now_add_peer(&peerInfo) != ESP_OK){
    Serial.println("Failed to add second peer");
    return;
  }
  // ::::::::::::::::
  Serial.println("-------------");
}
```

4.2.2 Slave ESP

4.2.2.1 Mengunggah Kode Melalui Visual Studio Code

Langkah-langkah untuk mengunggah kode program ke ESP32-Slave adalah serupa dengan yang dilakukan pada ESP32-Master.



4.2.2.2 AC Direct Protocol atau Burst Protocol

Anda perlu menentukan apakah protokol IR untuk perangkat AC sudah diketahui atau tidak. Jika protokolnya diketahui, Anda dapat menggantinya di kode. Namun, jika protokolnya tidak

diketahui, Anda dapat menggunakan Burst Protocol untuk mencoba berbagai protokol dan menemukan yang sesuai.

```
ac.next.protocol = decode_type_t::SHARP_AC;  // Set a protocol to use.
```

```
for (int i = 1; i < kLastDecodeType; i++) {
  decode_type_t protocol = (decode_type_t)i;
  // If the protocol is supported by the IRac class ...
  if (ac.isProtocolSupported(protocol)) {
    Serial.println("Protocol " + String(protocol) + " / " +
                   typeToString(protocol) + " is supported.");
    ac.next.protocol = protocol;  // Change the protocol used.
    Serial.println("Sending a message to the A/C unit.");
    lcd.clear();
    lcd.print("Sending");
    lcd.setCursor(0,1);
    lcd.print(typeToString(protocol));
    ac.sendAc();  // Have the IRac class create and send a message.
    delay(1000);  // Wait 1 seconds.
  }
}
```

4.3 Penambahan Coding Untuk Peer Baru

4.3.1 MAC Address di Master

1. **Dapatkan MAC Address Baru:** Jika Anda ingin menambahkan peer baru, pertama-tama Anda perlu mendapatkan MAC address perangkat ESP yang akan menjadi peer baru.
2. **Deklarasikan MAC Address:** Buka kode ESP32-Master dan temukan bagian "Variable pernyimpanan MAC ADDRESS untuk Receivers". Deklarasikan MAC address perangkat ESP yang akan ditambahkan sebagai peer baru dengan nama yang relevan.

```
// ======================================== REPLACE WITH THE MAC ADDRESS OF YOUR SLAVES / RECEIVERS / ESP32 RECEIVERS.
uint8_t broadcastAddressSlave1[] = {0x78, 0x21, 0x84, 0x87, 0xAF, 0x88}; // Slave 1 MAC Address 78:21:84:87:AF:88
uint8_t broadcastAddressSlave2[] = {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF}; // Slave 2 MAC Address ---
```

4.3.2 Register Peer Baru

1. **Buka Kode ESP32-Master:** Temukan fungsi "registerPeer()" pada kode ESP32-Master.

2. **Tambahkan Informasi Peer Baru:** Di dalam fungsi "registerPeer()", tambahkan blok kode untuk mendaftarkan peer baru. Gunakan MAC address yang telah Anda deklarasikan sebelumnya. Misalnya, jika MAC address perangkat baru adalah "AA:BB:CC:DD:EE:FF", maka kode akan terlihat seperti ini:

```
// ::::::::::::::::: register third peer
  Serial.println("Register third peer (ESP32 Slave 3)");
  memcpy(peerInfo.peer_addr, broadcastAddressSlave3, 6);
  if (esp_now_add_peer(&peerInfo) != ESP_OK){
    Serial.println("Failed to add third peer");
    return;
  }
  // ::::::::::::::::
```

3. **Tambahkan Kondisi Pengiriman:** Pada bagian pengiiman sinyal dai master ke slave kia memerlukan deklarasi pengiriman seperti “if (Board == “Nama Board”) send_data_to_slave(“Nomor Board”, LED_gpio_num.toInt(), LED_val.toInt());”

```
  if (request->hasParam(PARAM_INPUT_1) && request->hasParam(PARAM_INPUT_2) && request->hasParam(PARAM_INPUT_3)) {
    Board = request->getParam(PARAM_INPUT_1)->value();
    LED_gpio_num = request->getParam(PARAM_INPUT_2)->value();
    LED_val = request->getParam(PARAM_INPUT_3)->value();

    String Rslt = "Board : " + Board + " || GPIO : " + LED_gpio_num + " || Set to :" + LED_val;
    Serial.println();
    Serial.println(Rslt);
    // Conditions for sending data to All Slave.
    if (Board == "ESP32AllSlave") send_data_to_slave(0, LED_gpio_num.toInt(), LED_val.toInt());
    // Conditions for sending data to Slave 1.
    if (Board == "ESP32Slave1") send_data_to_slave(1, LED_gpio_num.toInt(), LED_val.toInt());
    // Conditions for sending data to Slave 2.
    if (Board == "ESP32Slave2") send_data_to_slave(2, LED_gpio_num.toInt(), LED_val.toInt());
  }
  else {
    Board = "No message sent";
    LED_gpio_num = "No message sent";
    LED_val = "No message sent";
  }
  request->send(200, "text/plain", "OK");
});
// ------------------------------------
```

4.3.3 HTML Code Baru (More Card)

1. **Buka Kode HTML:** Buka kode HTML antarmuka web pada aplikasi Anda.
2. **Tambahkan More Card:** Tambahkan elemen HTML baru, seperti "card" atau "button", untuk memungkinkan pengguna mengakses peer baru yang telah Anda tambahkan.
3. **Konfigurasi Aksi:** Tentukan aksi yang akan dilakukan saat pengguna menekan More Card. Anda dapat menghubungkan aksi ini dengan mengirim data ke peer baru, mengubah tampilan, atau melakukan tindakan lainnya sesuai dengan tujuan aplikasi.

```
<!-- Can Copy For Multiple Slave
<div class="card">
  <div class="card header">
    <h2>ESP32 BOARD #2</h2>
  </div>
  <br>

  <h4 class="LEDColor"><i class="fas fa-power-off"></i> POWER </h4>
  <label class="switch">
    <input type="checkbox" id="togPower1" onclick="send_LED_State_Cmd('ESP32Slave1','togPower1','1')">
    <div class="sliderTS"></div>
  </label>
  <br><br>

  <h4 class="LEDColor"><i class="fas fa-temperature-low"></i> TEMP </h4>
  <div class="slidecontainer">
    <input type="range" min="16" max="32" value="16" class="slider" id="tempSlider1"> <span id="tempTextSlider1">16</span><sup>o</sup>
  </div>
  <br>
 </div>
</div> -->
```

4.4 Penambahan Fitur Baru

Pada bagian button HTML, terdapat deklarasi fungsi yang akan dipanggil saat tombol tersebut ditekan. Fungsi ini menggunakan "**send_cmd('Slave ke berapa', 'Perintah yang dilakukan', 'valuenya')**" untuk mengirimkan data yang diinginkan ke perangkat ESP dengan format yang sesuai, seperti yang ditunjukkan pada Gambar … send_cmd()-nya berfungsi untuk ke semua Slave, kemudian perintah 1(ON/OFF), valuenya 1(ON)/0(OFF).

```
  <h2>BIG REMOTE</h2>
</div>
<br>
  <h4 class="LEDColor"><i class="fas fa-power-off"></i> Display</h4>
  <p id="State1">State - OFF</p>
  <button class="custom-button" onclick="change_display('State1','ON'); send_cmd('ESP32AllSlave','1','1')"> On </button>
  <button class="custom-button" onclick="change_display('State1','OFF');send_cmd('ESP32AllSlave','1','0')"> Off </button>
```