Brajan Gąbka

POLITECHNIKA LUBELSKA

Wydział Elektrotechniki i Informatyki

Projekt Inżynierski

Zastosowanie mikrokontrolerów i sygnałów bezprzewodowych do zdalnego sterowania oświetleniem LED w sieci AC 230 V

Wersja: 1.20v
Brajan Gąbka

Spis treści

1 Opis ogólny

Niniejsza dokumentacja techniczna stanowi opis projektu inżynierskiego pod tytułem: *„Zastosowanie mikrokontrolerów i sygnałów bezprzewodowych do zdalnego sterowania oświetleniem LED w sieci AC 230 V”*.

1.1 Cel projektu

Celem projektu jest stworzenie systemu sterowania oświetleniem LED, który można zastosować w budynku/mieszkaniu jednorodzinnym bez konieczności dodania przewodów do już istniejącej instalacji elektrycznej.

1.2 Zarys problemów

W związku z wzrastającymi wymaganiami mieszkańców i niepoprawnym oszacowaniu planu budynku/mieszkania przez inwestora, zastosowanie inteligentnego oświetlenia, w którym sygnały sterujące byłyby przenoszone przez przewód, powodowałoby wysokie koszty implementacji danego rozwiązania w już istniejącym budynku. [1]

1.3 Zakres projektu

Projekt obejmuje projektowanie, implementację, testowanie i przedstawienie systemu sterowania oświetleniem LED, w miniaturowym domu zasilanym z gniazdka elektrycznego 230 V 50 Hz. Do przesyłu informacji została zastosowana technologia Bluetooth. Jest stosowana do komunikacji między serwerem a sterownikami wykonawczymi, którymi są mikrokontrolery, oraz między telefonem a serwerem.

2 Specyfikacja wymagań

2.1 Wymagania funkcjonalne

- System powinien być w stanie komunikować się z smartfonem za pomocą Bluetooth oraz WiFi
- System powinien umożliwiać wyłączenie lub włączenie oświetlenia LED
- System powinien umożliwiać ściemnianie oświetlenia LED przez wielu użytkowników
- System powinien umożliwić sprawdzenie aktualnego poziomu oświetlenia
- System powinien kontrolować oświetlenie z poziomu pojedynczej oprawy jak i wszystkich dostępnych opraw

2.2 Wymaganie niefunkcjonalne

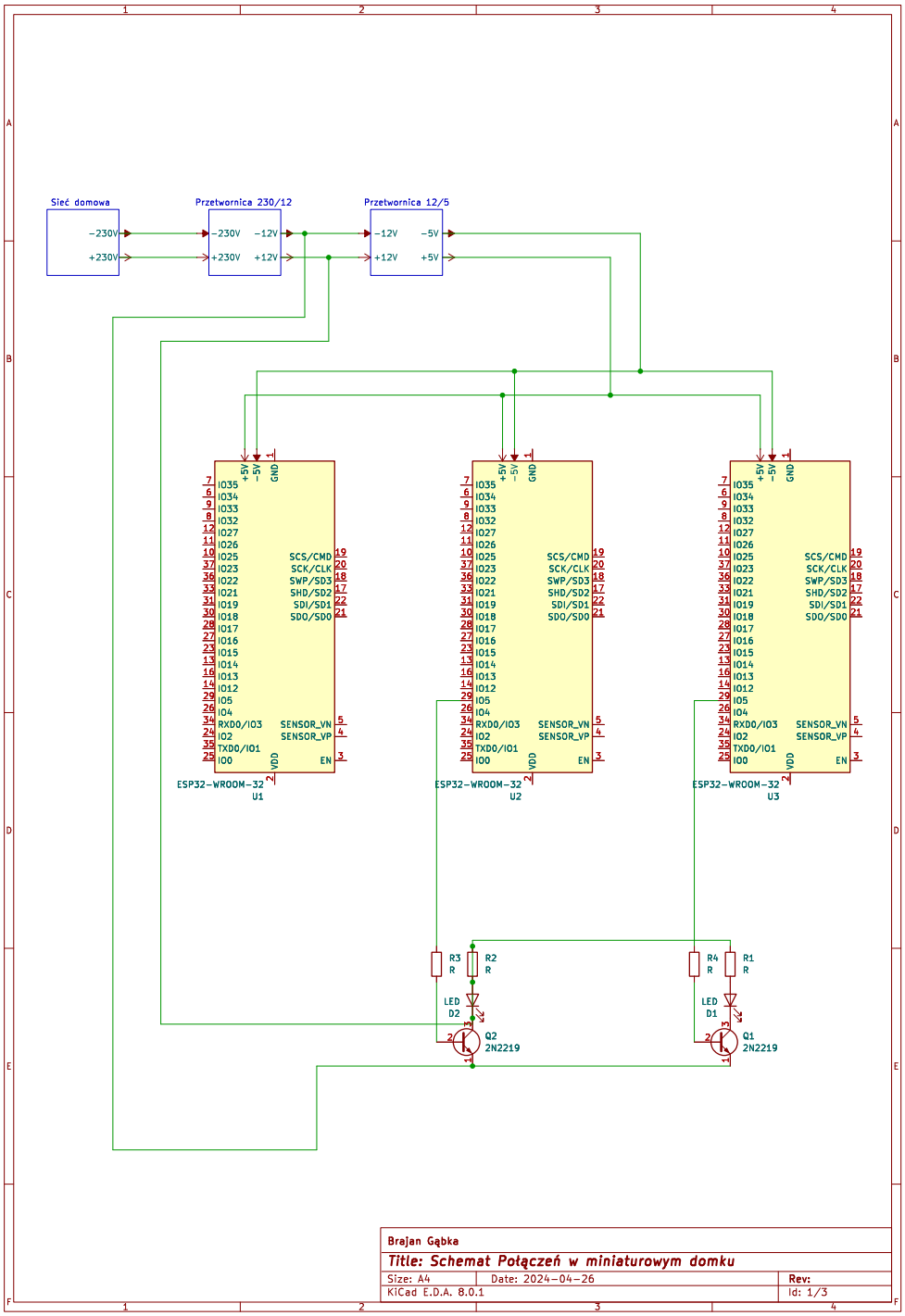
- System powinien nie wymagać remontu pokoju/budynku, w którym ma funkcjonować
- System powinien być łatwy w implementacji

2.3 Założenia i ograniczenia

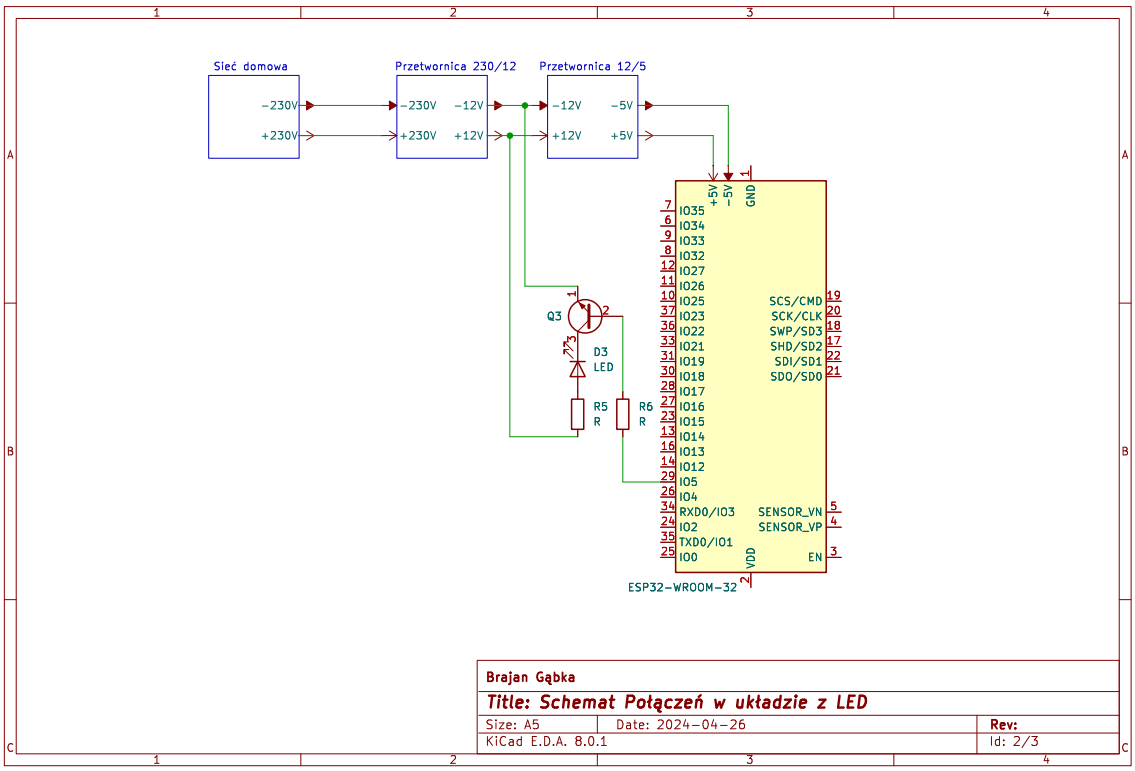
- System będzie wykorzystywał bezprzewodową transmisje danych Bluetooth oraz WiFi do komunikacji między urządzeniami
- Użytkownik będzie musiał zainstalować na telefon aplikację umożliwiającą komunikację z urządzeniami sterującymi w przypadku sterowania za pomocą Bluetooth
- System będzie przystosowany do sieci 230 V 50 Hz
- LED zastosowany w oświetleniu będzie przystosowany do 12 V

3 Projekt elektryczny

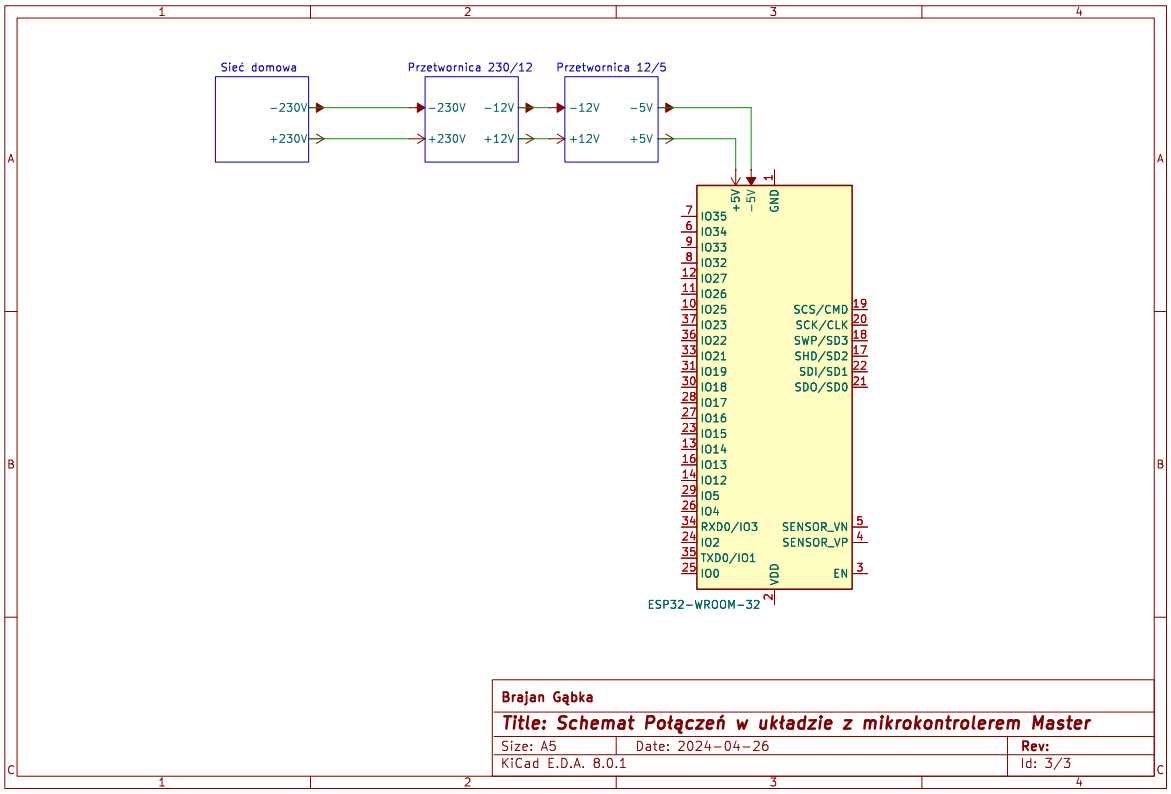
3.1 Schemat połączeń



Rys 1: Schemat Połączeń w miniaturowym domku



Rys 2: Schemat Połączeń w układzie z LED

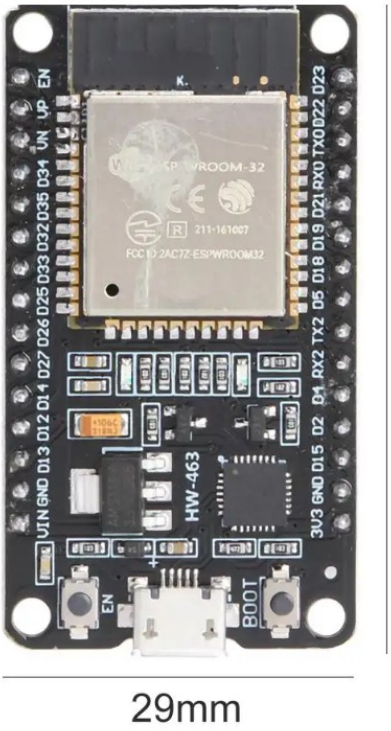


Rys 3: Schemat Połączeń w układzie z mikrokontrolerem Master

3.2 Lista komponentów

- ESP-WROOM-32 [2]
- Przetwornica 240/12 V AC/DC [3]
- Przetwornica 12/5 V DC/DC [4]
- Tranzystor BC547B [5]

3.3 Opis komponentów



ESP-WROOM-32

ID FCC: 2AC7Z-espwroom32

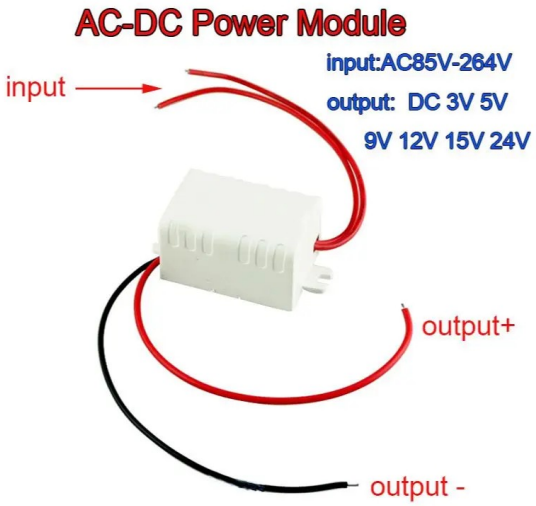
Moduł 802.11b/g/WiFi + BT

Bluetooth: Zgodne ze standardami Bluetooth 4.2 BR/EDR i BLE

Szybkość portu szeregowego: 115200

Dwurdzeniowy 32-bitowy procesor

Pobór mocy: 300 mA 3,3 V



Przetwornica

Model: LS-3S-WA

Napięcie wejściowe: AC 100-240 V 50/60 Hz

Napięcie wyjściowe: DC 12 V

Prąd: 0,3 A

Tętnienie: <150mV

Dokładność napięcia wyjściowego: ± 3%

Przetwornica

Napięcie wejściowe: DC 12 V

Napięcie wyjściowe: DC 5 V

Prąd: 3 A

Wydajność: 96%

Rozmiar: 46 mm x 27 mm x 14 mm

Producent: Diymore

Tranzystor

Model: BC547B

Napięcie maksymalne kolektor-emiter: 50 V

Prąd maksymalny kolektora: 0,1 A

Konfiguracja wyprowadzeń: CBE

Obudowa: TO92 (THT)

4 Oprogramowanie

4.1 Zaimplementowane biblioteki

WiFi.h

Biblioteka „WiFi.h” umożliwia mikrokontrolerowi ESP32 nawiązywanie połączenia z sieciami WiFi.

AsyncTCP.h

Biblioteka „AsyncTCP.h”zapewnia wsparcie dla asynchronicznej komunikacji TCP na ESP32, co pozwala na bardziej efektywne zarządzanie zasobami i obsługę wielu połączeń jednocześnie bez blokowania głównego wątku programu.

ESPAsyncWebServer.h

Biblioteka „ESPAsyncWebServer.h” jest rozszerzeniem do „AsyncTCP” i umożliwia tworzenie asynchronicznych serwerów HTTP na ESP32. Dzięki temu możliwe jest obsługiwanie żądań HTTP (GET, POST, itp.) bez blokowania głównego wątku.

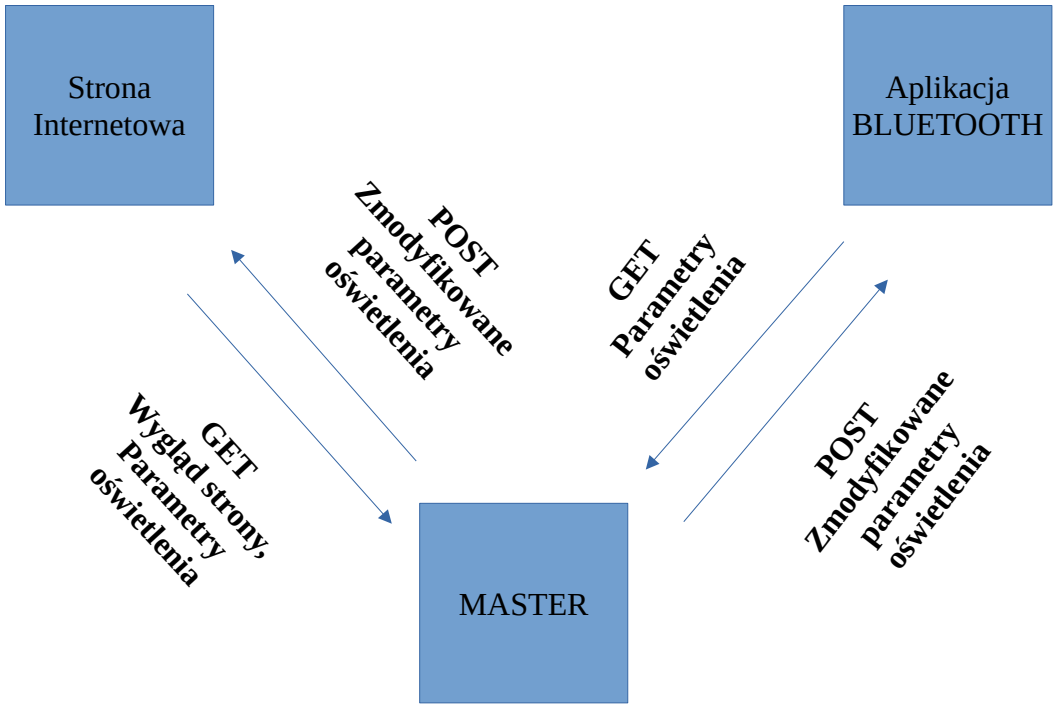
SPIFFS.h

Biblioteka „SPIFFS.h” (SPI Flash File System) pozwala na zarządzanie systemem plików na pamięci flash ESP32.

BluetoothSerial.h

Biblioteka „BluetoothSerial.h” umożliwia komunikację przez Bluetooth na ESP32.

4.2 Projekt systemu sterowania



Rys 4: Schemat Systemu Sterowania za pomocą ESP32 jako Master dla Aplikacji Webowej i Bluetooth

Użytkownik ma możliwość modyfikowania wartości oświetlenia przez stronę internetową oraz aplikację Bluetooth. Nie jest wymagane wysłanie zapytania o aktualne parametry oświetlenia. Aktualizacja parametrów oświetlenia dla wszystkich użytkowników następuje po modyfikacji.

4.3 BLUETOOTH

```
#include "BluetoothSerial.h"

bool MasterConnected;
#if !defined(CONFIG_BT_ENABLED) || !
defined(CONFIG_BLUEDROID_ENABLED)
#error Bluetooth is not enabled! Please run make menuconfig to and
enable it
#endif
#if !defined(CONFIG_BT_SPP_ENABLED)
#error Serial Bluetooth not available or not enabled. It is only
available for the ESP32 chip.
#endif

//Inicjalizacja obiektu BluetoothSerial
BluetoothSerial SerialBT;

void Bt_Status (esp_spp_cb_event_t event, esp_spp_cb_param_t
*param) {

  if (event == ESP_SPP_SRV_OPEN_EVT) {
    Serial.println ("Client Connected");
    SerialBT.print("Client Connected");
    MasterConnected = true;
  }
  else if (event == ESP_SPP_CLOSE_EVT ) {
    Serial.println ("Client Disconnected");
    MasterConnected = false;
  }
}
```

Te linie kodu sprawdzają, czy Bluetooth i stos BluDroid są włączone, a także czy funkcja Serial Bluetooth Profile (SPP) jest dostępna i włączona. W przypadku niespełnienia tych warunków, kompilator zgłosi błąd.

Następnie zostaje stworzony obiekt SerialBT klasy BluetoothSerial, który będzie używany do komunikacji przez Bluetooth.

Funkcja „Bt_Status” jest wywoływana, gdy następuje zdarzenie związane z SPP. Służy do wyświetlenia zmiany statusu połączenia Bluetooth.

```
SerialBT.begin("ESP32");
SerialBT.register_callback (Bt_Status);
Serial.printf("Serwer jest uruchomiony!\nMożna sparować z BLUETOOTH!\
n");
```

W Setup następuje inicjalizacja modułu Bluetooth na ESP32 i ustawienie nazwy urządzenia na "ESP32", oraz rejestrację funkcji „Bt_Status” jako funkcji zwrotnej (callback) dla zdarzeń związanych z Serial Port Profile (SPP).

```
if (SerialBT.available()) {
    char receivedChar = SerialBT.read();
    if (message == ""){
      timeForMessage = millis();
    }
    message += receivedChar;
  }
```

Ten fragment kodu odpowiada za odczyt danych przychodzących przez interfejs Bluetooth (SerialBT) na mikrokontrolerze ESP32. Warunek sprawdza, czy są dostępne jakieś dane do odczytu z interfejsu Bluetooth. Jeśli tak, to kod wewnątrz tego warunku zostanie wykonany.

Z powodu otrzymywania pojedynczych znaków w transmisji BT, przed przesłaniem informacji o odczycie należy scalić znaki w string.

4.4 Strona internetowa

```
WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
        delay(1000);
        Serial.print(".");
    }
    Serial.println("\nPołączono z WiFi!");
```

W sekcji Setup następuje połączenie z routerem. Tworzona jest pętla, która będzie się wykonywać, dopóki mikrokontroler nie połączy się z siecią WiFi.

```
server.on("/", HTTP_GET, [](AsyncWebServerRequest *request){

        request->send(SPIFFS, "/index.html", "text/html");
    });

    // Ustawienie routingu dla pobierania parametru
server.on("/pobierz-parametr", HTTP_GET, [](AsyncWebServerRequest
     *request){

        request->send(200, "text/plain", wartosc);
        Serial.print("Wartość parametru: ");
        Serial.println(wartosc);
    });

    // Ustawienie routingu dla ustawiania parametru
server.on("/ustaw-parametr", HTTP_POST, [](AsyncWebServerRequest
     *request){

        if (request->hasParam("parametr", true)) {

            String newValue = request->getParam("parametr", true)-
                >value();

            wartosc = newValue;
            Serial.print("Nowa wartość parametru: ");
            Serial.println(wartosc);
            request->send(200, "text/plain", "Wartość parametru została
                zaktualizowana");

        } else {
            request->send(400, "text/plain", "Nieprawidłowe żądanie");
        }
    });

    // Start serwera
    server.begin();
    Serial.println("Serwer uruchomiony!");
```

Ten fragment kodu dotyczy konfiguracji i obsługi prostego serwera HTTP na mikrokontrolerze ESP32, wykorzystującego bibliotekę AsyncWebServer do obsługi zapytań HTTP. Obsługuje zapytania HTTP GET na adresie "/" i "/pobierz-parametr", oraz zapytania HTTP POST na adresie "/ustaw-parametr".

4.5 Inicjalizacja SPIFFS

```
if (!SPIFFS.begin(true)) {
        Serial.println("Błąd inicjalizacji SPIFFS");
        return;
    }
    Serial.println("SPIFFS zainicjalizowane!");
```

Ten fragment kodu inicjalizuje system plików SPIFFS. W przypadku niepowodzenia inicjalizacji, co może się zdarzyć na przykład w przypadku uszkodzenia systemu plików lub braku dostępu do niego, wyświetlany jest komunikat “Błąd inicjalizacji SPIFFS”.

5 Projekt miniaturowego domku

Rys 5: Model: Welcome Home - Home

Na potrzeby prezentacji projektu, został wydrukowany model użytkownika NeatoBrian udostępniony na licencji „CC0”.[6]

Rozmieszczenie elektroniki:

- Na lewym i prawym dachu zostały umieszczone 2 ESP LED (Rys 2)
- Na lewym boku modelu został umieszczony ESP MASTER (Rys 3)
- Z tyłu zostały umieszczone przetwornice
- LED-y zostały umieszczone w środku, po lewej i prawej stronie oddzielonych między sobą ścianą

6 Implementacja

Układ związany ze sterowaniem LED (Rys 2) powinien zostać umieszczony w oprawie oświetleniowej. Wszystkie elementy z wyjątkiem LED należy umieścić nad oprawą oświetleniową w celu ukrycia elementów elektronicznych.

Układ odpowiedzialny za komunikację (Rys 3) wymaga dostępu do sieci zasilającej 230 V AC lub 5 V DC. Zastosowanie zasilania 5 V pozwala na pominięcie stosowania przetwornicy.

Układ Rys 1: Schemat Połączeń w miniaturowym domku jest stosowany wyłącznie w pokazowym przedstawieniu niniejszego projektu i zawiera się w konstrukcji, która do zasilenia wymaga dostępu do gniazdka elektrycznego.

7 Instrukcja użytkownika

7.1 Wprowadzenie

Inteligentne oświetlenie zostało zaprojektowane w celu zwiększenia komfortu użytkowania oświetlenia i umożliwienia zdalnego sterowania ściemnianiem LED.

7.2 Pierwsze uruchomienie

Bluetooth

Aby móc korzystać z produktu, należy zainstalować na smartfonie z dostępem do Bluetooth aplikację Serial Bluetooth Terminal. Następnie należy połączyć się z mikrokontrolerem Master i w terminalu wpisać komendę „SEARCH”. Po upływie 1 minuty należy ponownie połączyć się i

wpisać komendę „SHOW”. Zostaną wyświetlone wszystkie dostępne LED-y. Jeżeli zostały znalezione wszystkie zainstalowane LED-y, można korzystać z produktu.

WiFi

Aby móc korzystać z produktu należy sprawdzić do jakiego adresu został przypisany mikrokontroler Master, a następnie należy wpisać jego adres w przeglądarce internetowej. Po wyświetleniu się strony, należy przejść do zakładki „Dodaj” i wpisać adres lub nazwę Bluetooth oprawy LED.

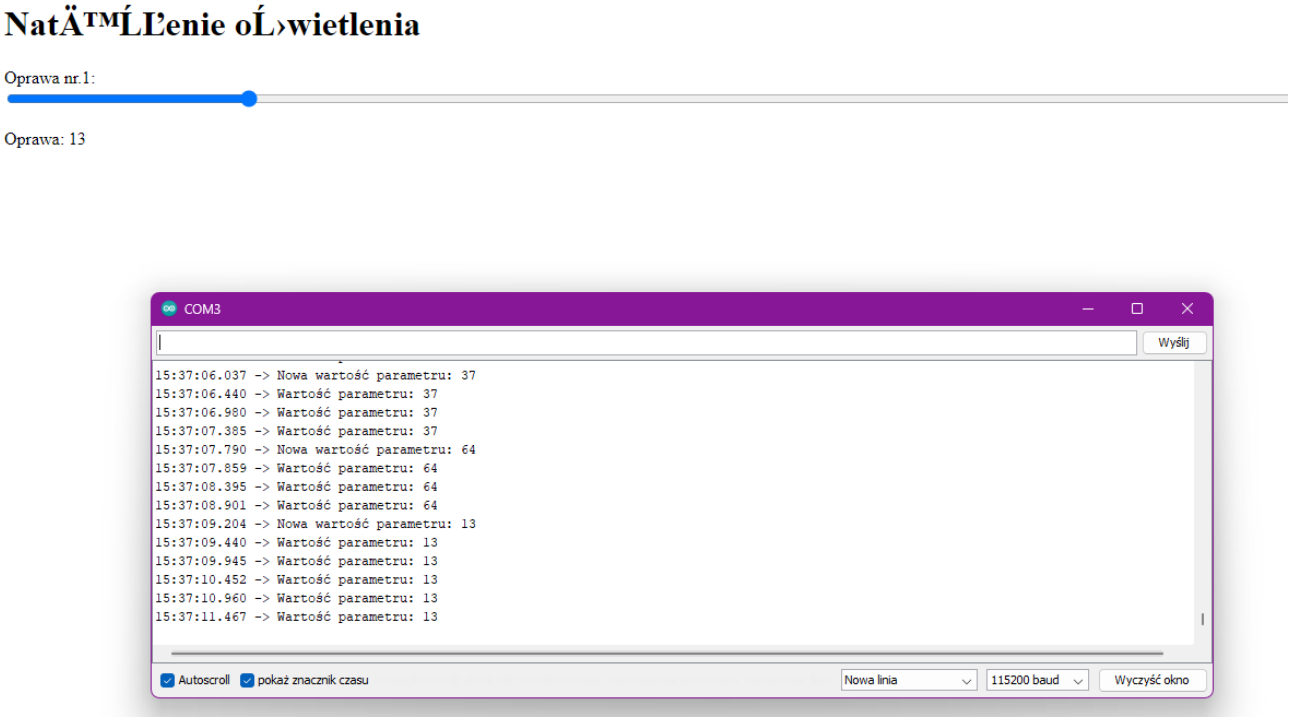
7.3 Ściemnianie oprawy LED

Aby ściemnić wybraną oprawę przez Bluetooth, należy połączyć się z Masterem a następnie wpisać ID oprawy ,znak „u” i poziom oświetlenia wyrażonego w procentach.

Przykład:

Chcę, aby oprawa o ID „01” osiągnęła docelowy poziom oświetlenia „30” %. W tym celu napiszę komendę „02u30”.

Aby ściemnić wybraną oprawę przez WiFi, należy wejść na stronę z listą opraw i suwakiem zmienić aktualny parametr oświetlenia.



Rys 6: Wygląd Strony internetowej i zmiana wartości oświetlenia

7.4 Dodanie nowych opraw do programu

Bluetooth

Aby dodać nowe urządzenie do listy opraw należy połączyć się z mikrokontrolerem Master i w terminalu wpisać komendę „SEARCH”, po 1 minucie należy ponownie połączyć się i wpisać komendę „SHOW”. Do listy powinna zostać dodana nowa oprawa. Jeżeli nie została dodana należy przejść do 7.5 Nie wszystkie urządzenia zostały znalezione.

WiFi

Należy na stronie z listą opraw przejść do zakładki „Dodaj” i wpisać adres lub nazwę Bluetooth oprawy LED.

7.5 Nie wszystkie urządzenia zostały znalezione

Jeżeli dodanie nowego urządzenia nie powiodło się, należy sprawdzić czy:

Instalacja oprawy została połączona według schematu Rys 2: Schemat Połączeń w układzie z LED,

Czy sygnał Bluetooth między oprawą LED a mikrokontrolerem Master jest dostateczny,

Czy sygnał WiFi między oprawą LED a routerem jest dostateczny,

Czy mikrokontroler LED nie jest uszkodzony.

7.6 Master komunikuje się z oprawą ale LED się nie świeci

Jeżeli komunikacja zachodzi prawidłowo, to należy sprawdzić czy oświetlenie LED nie uległo spaleniu.

8 Kosztorys

Nazwa	Ilość	Cena/opakowanie, zł
ESP-WROOM-32	3	18,63
Przetwornica 240/12 V AC/DC	1	17,19
Przetwornica 12/5 V DC/DC	1	10,27
Tranzystor BC547B x200	1	16,46
LED 12 V x10	1	14,10
Total		113,91

9 Podsumowanie i wnioski

Upload stron internetowych do pamięci SPIFFS jest możliwy wyłącznie na starszej wersji oprogramowania „Arduino IDE 1.8.19”. Aktualna wersja „Arduino IDE 2.3.2” nie jest kompatybilna z biblioteką „SPIFFS.h”.[7]

Domyślny schemat wielkość partycji pamięci na miktrokontrolerze „ESP-WROOM-32” jest za mały aby zapisać wszystkie biblioteki i kod do transmisji Bluetooth i WiFi. Aby rozwiązać ten problem należy zmienić schemat partycji na „Huge APP (3MB No OTA/1MB SPIFFS)”.[8]

Projekt inżynierski jest dostępny pod linkiem GitHub.[9]

10 Załączniki

Bibliografia

1: SmartSpace, Instalacja tradycyjna vs inteligentna - którą wybrać?, https://www.smartspace.pl/blog/blog-smartspace-1/instalacja-tradycyjna-vs-inteligentna-ktora-wybrac-50
2: fcc.report, 2AC7Z-ESPWROOM32, https://fcc.report/FCC-ID/2AC7Z-ESPWROOM32
3: AliExpress, Przetwornica LS-3S-WA (240/12 V), https://pl.aliexpress.com/item/1005005778264197.html?spm=a2g0o.order_list.order_list_main.79.21ef1c24FQTKjH&gatewayAdapt=glo2pol
4: Diymore, Przetwornica (12/5 V), https://www.diymore.cc/products/12v-to-5v-3a-15w-dc-dc-buck-converter-step-down-module-car-monitor-power-supply-car-power-converter-regulator-adapter-for-car
5: Botland, Tranzystor BC547B, https://botland.com.pl/tranzystory-npn/254-tranzystor-bipolarny-npn-bc547b-50v-01a-5szt-5904422308025.html
6: NeatoBrian, Welcome Home - Home, https://www.printables.com/pl/model/473105-welcome-home-home
7: randomnerdtutorials, Install ESP32 Filesystem Uploader in Arduino IDE, https://randomnerdtutorials.com/install-esp32-filesystem-uploader-arduino-ide/
8: robotzero.one, Partition Schemes in the Arduino IDE, https://robotzero.one/arduino-ide-partitions/
9: Brajan Gąbka, Zastosowanie mikrokontrolerów i sygnałów bezprzewodowych do zdalnego sterowania oświetleniem LED w sieci AC 230 V, https://github.com/Duke-Axer/LED-lighting_NKN