



**Politechnika
Śląska**

PROJEKT INŻYNIERSKI

System wizualizacji danych webowego interfejsu Maszyny W z użyciem
mikrokontrolera ESP32

Bartosz FARUGA

Nr albumu: 305985

Kierunek: Informatyka

Specjalność: Bazy danych i inżynieria systemów

PROWADZĄCY PRACĘ

dr. inż. Tomasz Rudnicki

KATEDRA SYSTEMÓW CYFROWYCH

Wydział Automatyki, Elektroniki i Informatyki

OPIEKUN, PROMOTOR POMOCNICZY

⟨stopień naukowy imię i nazwisko⟩

Gliwice 2025

Tytuł pracy

System wizualizacji danych webowego interfejsu Maszyny W z użyciem mikrokontrolera ESP32

Streszczenie

(Streszczenie pracy – odpowiednie pole w systemie APD powinno zawierać kopię tego streszczenia.)

Słowa kluczowe

(2-5 słów (fraz) kluczowych, oddzielonych przecinkami)

Thesis title

Thesis title in English

Abstract

(Thesis abstract – to be copied into an appropriate field during an electronic submission – in English.)

Key words

(2-5 keywords, separated by commas)

Spis treści

1	Wstęp	1
1.1	Cel pracy	1
1.2	Analiza rozwiązań literaturowych	1
2	Przykładowa realizacja systemu wizualizacji	3
2.1	Założenia realizacyjne	3
2.2	Schemat blokowy	4
3	Część projektowa	5
3.1	System mikroprocesorowy	5
3.2	Adresowalne diody LED	6
3.3	Przyciski	6
3.4	Podświetlenie LED	7
3.5	Enkoder	7
3.6	Przełącznik trybów	7
4	[Właściwy dla kierunku – np. Specyfikacja zewnętrzna]	9
5	[Właściwy dla kierunku – np. Specyfikacja wewnętrzna]	11
6	Weryfikacja i walidacja	13
7	Podsumowanie i wnioski	15
	Bibliografia	17
	Spis skrótów i symboli	19
	Źródła	21
	Lista dodatkowych plików, uzupełniających tekst pracy	23
	Spis rysunków	25

Rozdział 1

Wstęp

1.1 Cel pracy

Celem niniejszej pracy jest opracowanie systemu wizualizacji danych pochodzących z webowego interfejsu Maszyny W z wykorzystaniem mikrokontrolera ESP32-S3.

Opracowane w ramach pracy dyplomowej urządzenie umożliwia odwzorowanie w czasie rzeczywistym wartości liczbowych oraz sygnałów sterujących symulatorem Maszyny W na adresowalnych diodach RGB. Ponadto system pozwala na zdalne sterowanie funkcjami symulatora zarówno za pomocą przycisków fizycznych, jak i interfejsu internetowego. System obsługuje hosting interfejsu webowego bezpośrednio na mikrokontrolerze oraz zapewnia komunikację pomiędzy ESP32 a interfejsem. Wykorzystany w tym celu protokół WebSocket umożliwia natychmiastową synchronizację zmian między stroną internetową a wyświetlaczem, co pozwala użytkownikowi na intuicyjny i interaktywny monitoring oraz kontrolę stanu Maszyny W.

1.2 Analiza rozwiązań literaturowych

Rozdział 2

Przykładowa realizacja systemu wizualizacji

2.1 Założenia realizacyjne

Pierwszym etapem budowy systemu wizualizacji danych jest projektowanie, które powinno być poprzedzone przemyśleniami dotyczącymi możliwych realizacji oraz wyborem tych, które są optymalne i spełniają przyjęte założenia. Założenia dodatkowe nie są bezwzględnie wymagane, ale mogą poszerzyć tworzone rozwiązanie o alternatywne sposoby działania.

Założenia podstawowe:

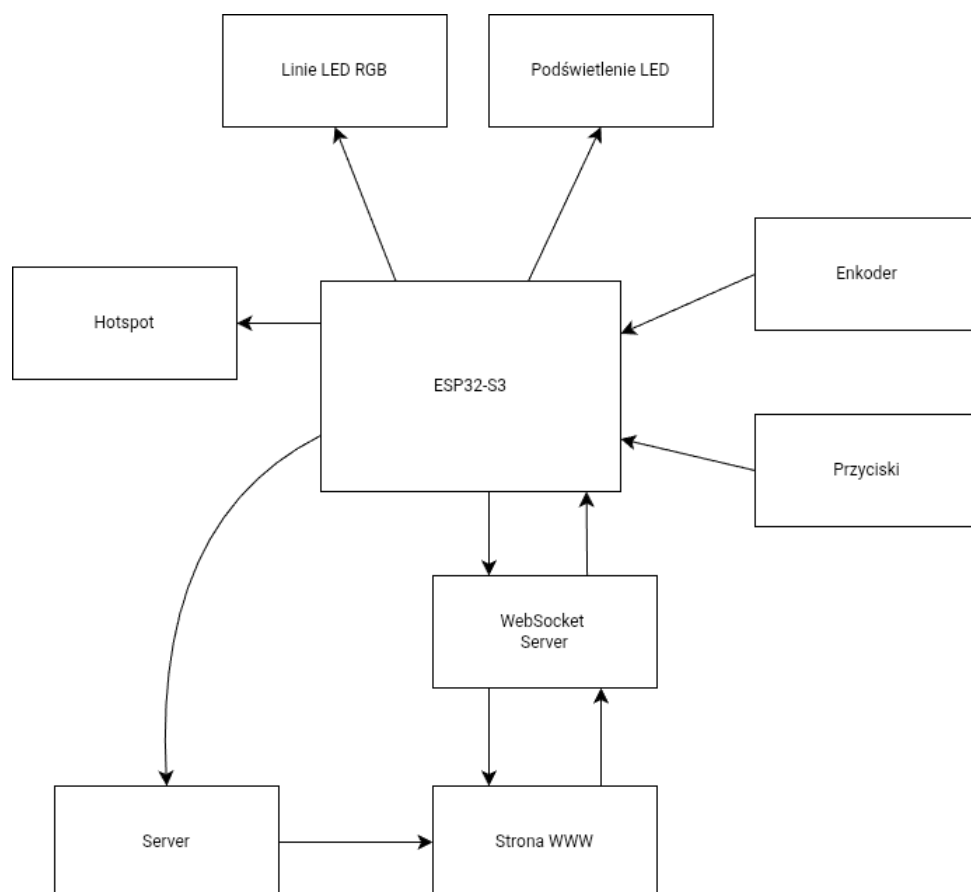
- dedykowany układ dla mikrokontrolera z odpowiednimi wyprowadzeniami,
- dedykowany układ dla LED-owych wyświetlaczy siedmiosegmentowych,
- konstrukcja kompletnego układu elektronicznego z układów mikrokontrolera, wyświetlaczy LED oraz przycisków,
- hosting strony internetowej bezpośrednio na urządzeniu wraz z przesyłem danych pomiędzy urządzeniem a stroną,
- wizualizacja informacji pobieranych ze strony internetowej na adresowalnych diodach LED,
- przesył informacji o wciśniętych przyciskach do strony internetowej,
- obsługa podświetlenia modelu Maszyny W diodami LED.

Założenia dodatkowe:

- lokalna symulacja Maszyny W na mikrokontrolerze,
- obsługa enkodera do wprowadzania wartości liczbowych.

2.2 Schemat blokowy

Na rysunku poniżej (Rys. 2.1) przedstawiono przykładowy schemat blokowy systemu wizualizacji interfejsu maszyny (SWIM). Schemat uwzględnia zarówno bloki spełniające założenia podstawowe, jak i dodatkowe.



Rysunek 2.1: Schemat blokowy SWIM

Rozdział 3

Część projektowa

3.1 System mikroprocesorowy

Ze względu na wymaganą komunikację sieciową projektu, zdecydowano się na użycie prostego mikrokontrolera z rodziny ESP32. Głównymi zadaniami użytego mikrokontrolera jest obsługa dwóch linii adresowalnych diod LED, obsługa 17 przycisków wraz z enkoderem, kontrola podświetlenia LED oraz hosting strony internetowej.

Ze względu na dostępność, popularność, bogatą dokumentację oraz niską cenę zdecydowano się na użycie mikrokontrolera *ESP32-S3-WROOM-1* firmy *Espressif Systems*

Własności układu

- dwurdzeniowy 32-bitowy mikroprocesor *Xtensa LX7*,
- 16MB pamięci flash,
- 16MB pamięci PSRAM,
- 512MB pamięci SRAM,
- 36 portów GPIO,
- komunikacja 2.4GHz WiFi (802.11b/g/n) oraz Bluetooth 5,
- wbudowana antena,
- wsparcie dla: I2C, LED PWM, USB Serial,
- zasilanie napięciem 3.3V.

3.2 Adresowalne diody LED

Projekt zakładał wizualizację wielu informacji na znaczących rozmiarów modelu Maszyny W. Przewidywana liczba pojedynczych diod LED przekracza 600. Oznaczać to może znaczące spadki napięcia na długiej linii LED.

Podczas wstępnych testów linii 5V zaobserwowano zauważalny spadek jakości odwzorowania kolorów.

Biorąc pod uwagę wyżej wymienione wymagania zdecydowano się na użycie adresowalnych diod *WS2815B-V1* firmy WORLDSEMI. Wybrany model diod pozwala na indywidualne zadawanie wartości Red, Green oraz Blue każdej z nich. Wyższe napięcie zasilania eliminuje powstałe w wyniku spadków napięcia zmiany barwy.

Własności elementu

- napięcie pracy 12V,
- obudowa 5050,
- montaż SMD,
- częstotliwość odświeżania 4kHz.

3.3 Przyciski

W celu zapewnienia pełnej obsługi Maszyny W potrzebne jest wykorzystanie 17 monostabilnych przycisków. Symulator w wersji podstawowej udostępnia użytkownikowi 16 możliwych sygnałów: *czyt*, *wys*, *wei*, *il*, *wyad*, *weak*, *pisz*, *przep*, *wel*, *wyl*, *dod*, *ode*, *wes*, *weja*, *wyak*, *wea*. Do wymienionych sygnałów należy również wliczyć przycisk *takt*, który zapewni możliwość wykonania zadanego rozkazu.

Zdecydowano się na wykorzystanie przełączników *MX Blue* firmy *Cherry*. Według dokumentacji producenta wyżej wymieniony przełącznik klawiaturowy pozwala na 50 milionów kliknięć. Pozwoli to na długoletnią, niezawodną działalność. Wykorzystanie przycisków typu *Blue* oferujących słyszalny odgłos kliknięcia przy wciskaniu. Daje to użytkownikowi informację zwrotną o stanie przełącznika.

Wykorzystanie tak dużej liczby przycisków wymaga użycia multiplexera 16:4. Z uwagi na niską cenę, łatwą dostępność oraz niewielki wymiar wybrano model *CD74HC4067M* firmy *Texas Instruments*.

3.4 Podświetlenie LED

Chcąc zapewnić przyjazny wizualnie wygląd modelu SWIM zdecydowano się na podświetlenie następujących elementów:

- licznik rozkazów,
- akumulator,
- rejestr instrukcji,
- rejestry A oraz S,
- logo Politechniki Śląskiej.

Wykorzystano do tego 12V linie diod LED.

TODO skończyć tekst PODŚWIETLENIA LED

3.5 Enkoder

Do wprowadzania danych liczbowych w lokalnym trybie działania urządzenia zastosowano enkoder obrotowy z przyciskiem. Wykorzystanie enkodera:

- zwiększanie/zmniejszanie wartości liczbowych w zależności od kierunku obrotu,
- aktywacja trybu wprowadzania danych poprzez przytrzymanie przycisku,
- przełączanie między polami wprowadzania danych przy użyciu szybkich kliknięć.

Enkoder pozwala użytkownikowi w prosty i intuicyjny sposób sterować parametrami urządzenia bez korzystania z interfejsu sieciowego.

3.6 Przełącznik trybów

Jednym z założeń dodatkowych projektu jest możliwość przełączania SWIM między trybami lokalnym oraz sieciowym. Aby zapewnić taką możliwość bezpośrednio z poziomu modelu wykorzystano przełącznik bistabilny.

Aby zapewnić taką możliwość bezpośrednio z poziomu modelu wykorzystano przełącznik bistabilny. Aby zapewnić taką możliwość bezpośrednio z poziomu modelu wykorzystano przełącznik bistabilny. Aby zapewnić taką możliwość bezpośrednio z poziomu modelu wykorzystano przełącznik bistabilny. Aby zapewnić taką możliwość bezpośrednio z poziomu modelu wykorzystano przełącznik bistabilny. Aby zapewnić taką możliwość bezpośrednio z poziomu modelu wykorzystano przełącznik bistabilny. Aby zapewnić taką możliwość bezpośrednio z poziomu modelu wykorzystano przełącznik bistabilny.

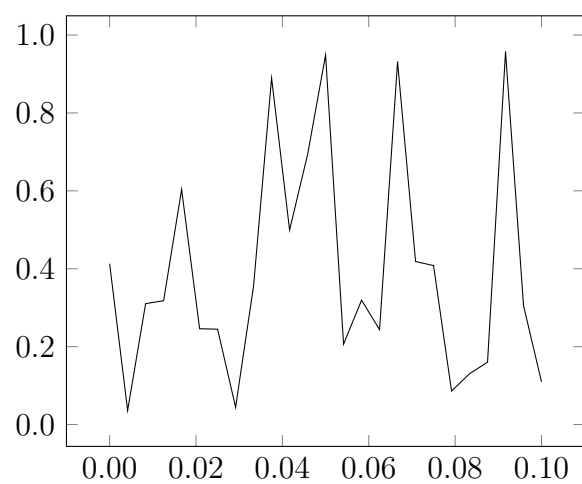
Aby zapewnić taką możliwość bezpośrednio z poziomu modelu wykorzystano przełącznik bistabilny. Aby zapewnić taką możliwość bezpośrednio z poziomu modelu wykorzystano przełącznik bistabilny. Aby zapewnić taką możliwość bezpośrednio z poziomu modelu wykorzystano przełącznik bistabilny. Aby zapewnić taką możliwość bezpośrednio z poziomu modelu wykorzystano przełącznik bistabilny. Aby zapewnić taką możliwość bezpośrednio z poziomu modelu wykorzystano przełącznik bistabilny. Aby zapewnić taką możliwość bezpośrednio z poziomu modelu wykorzystano przełącznik bistabilny. Aby zapewnić taką możliwość bezpośrednio z poziomu modelu wykorzystano przełącznik bistabilny. Aby zapewnić taką możliwość bezpośrednio z poziomu modelu wykorzystano przełącznik bistabilny. Aby zapewnić taką możliwość bezpośrednio z poziomu modelu wykorzystano przełącznik bistabilny.

Rozdział 4

[Właściwy dla kierunku – np. Specyfikacja zewnętrzna]

Jeśli „Specyfikacja zewnętrzna”:

- wymagania sprzętowe i programowe
- sposób instalacji
- sposób aktywacji
- kategorie użytkowników
- sposób obsługi
- administracja systemem
- kwestie bezpieczeństwa
- przykład działania
- scenariusze korzystania z systemu (ilustrowane zrzutami z ekranu lub generowanymi dokumentami)



Rysunek 4.1: Podpis rysunku po rysunkiem.

Rozdział 5

[Właściwy dla kierunku – np. Specyfikacja wewnętrzna]

Jeśli „Specyfikacja wewnętrzna”:

- przedstawienie idei
- architektura systemu
- opis struktur danych (i organizacji baz danych)
- komponenty, moduły, biblioteki, przegląd ważniejszych klas (jeśli występują)
- przegląd ważniejszych algorytmów (jeśli występują)
- szczegóły implementacji wybranych fragmentów, zastosowane wzorce projektowe
- diagramy UML

Krótką wstawka kodu w linii tekstu jest możliwa, np. **int a;** (biblioteka `listings`). Dłuższe fragmenty lepiej jest umieszczać jako rysunek, np. kod na rys 5.1, a naprawdę długie fragmenty – w załączniku.

```
1 class test : public basic
2 {
3     public:
4         test (int a);
5         friend std::ostream operator<<(std::ostream & s,
6                                         const test & t);
7     protected:
8         int _a;
9
10 };
```

Rysunek 5.1: Pseudokod w `listings`.

Rozdział 6

Weryfikacja i walidacja

- sposób testowania w ramach pracy (np. odniesienie do modelu V)
- organizacja eksperymentów
- przypadki testowe zakres testowania (pełny/niepełny)
- wykryte i usunięte błędy
- opcjonalnie wyniki badań eksperymentalnych

Tabela 6.1: Nagłówek tabeli jest nad tabelą.

ζ	metoda						
	alg. 1	alg. 2	alg. 3			alg. 4, $\gamma = 2$	
			$\alpha = 1.5$	$\alpha = 2$	$\alpha = 3$	$\beta = 0.1$	$\beta = -0.1$
0	8.3250	1.45305	7.5791	14.8517	20.0028	1.16396	1.1365
5	0.6111	2.27126	6.9952	13.8560	18.6064	1.18659	1.1630
10	11.6126	2.69218	6.2520	12.5202	16.8278	1.23180	1.2045
15	0.5665	2.95046	5.7753	11.4588	15.4837	1.25131	1.2614
20	15.8728	3.07225	5.3071	10.3935	13.8738	1.25307	1.2217
25	0.9791	3.19034	5.4575	9.9533	13.0721	1.27104	1.2640
30	2.0228	3.27474	5.7461	9.7164	12.2637	1.33404	1.3209
35	13.4210	3.36086	6.6735	10.0442	12.0270	1.35385	1.3059
40	13.2226	3.36420	7.7248	10.4495	12.0379	1.34919	1.2768
45	12.8445	3.47436	8.5539	10.8552	12.2773	1.42303	1.4362
50	12.9245	3.58228	9.2702	11.2183	12.3990	1.40922	1.3724

Rozdział 7

Podsumowanie i wnioski

- uzyskane wyniki w świetle postawionych celów i zdefiniowanych wyżej wymagań
- kierunki ewentualnych danych prac (rozbudowa funkcjonalna ...)
- problemy napotkane w trakcie pracy

Dodatki

Spis skrótów i symboli

DNA kwas deoksyrybonukleinowy (ang. *deoxyribonucleic acid*)

MVC model – widok – kontroler (ang. *model-view-controller*)

N liczebność zbioru danych

μ stopnień przyleżności do zbioru

\mathbb{E} zbiór krawędzi grafu

\mathcal{L} transformata Laplace’a

Źródła

Jeżeli w pracy konieczne jest umieszczenie długich fragmentów kodu źródłowego, należy je przenieść w to miejsce.

```
1 if (_nClusters < 1)
2     throw std::string ("unknown_number_of_clusters");
3 if (_nIterations < 1 and _epsilon < 0)
4     throw std::string ("You_should_set_a_maximal_number_of_
        iteration_or_minimal_difference_epsilon.");
5 if (_nIterations > 0 and _epsilon > 0)
6     throw std::string ("Both_number_of_iterations_and_minimal_
        epsilon_set—you_should_set_either_number_of_iterations
        or_minimal_epsilon.");
```

Lista dodatkowych plików, uzupełniających tekst pracy

W systemie do pracy dołączono dodatkowe pliki zawierające:

- źródła programu,
- dane testowe,
- film pokazujący działanie opracowanego oprogramowania lub zaprojektowanego i wykonanego urządzenia,
- itp.

Spis rysunków

2.1	Schemat blokowy SWIM	4
4.1	Podpis rysunku po rysunkiem.	10
5.1	Pseudokod w <code>listings</code>	12

Spis tabel

6.1	Nagłówek tabeli jest nad tabelą.	14
-----	--	----