Laboratorium Optoelektroniki i Fotoniki

Politechnika Wrocławska

Katedra Metrologii Elektronicznej i Fotonicznej

Detektor składowych RGB

(Projekt wykonano w ramach przedmiotu Optoelektronika 2-projekt)

Nr gr. 23

Skład grupy

Radosław Mierzwa 263675

Pavlo Kostushevych 251190

Mateusz Gwioździk 263658

WROCLAW 2024

Spis treści

1. Wstęp

Niniejszy projekt detektora składowych RGB został zrealizowany w ramach kursu Optoelektronika 2 w ciągu jednego semestru. Raport stanowi kompleksową dokumentację projektu detektora składowych RGB, będącego przedmiotem naszych badań. W treści raportu zgłębiamy główne aspekty projektu, zgodnie z założeniami przyjętymi na etapie jego planowania.

Projekt ten ma na celu pomiar składowych kolorów RGB w otoczeniu oraz praktyczne wykorzystanie tych danych. W ramach raportu przedstawiamy proces projektowania, zastosowane technologie, kluczowe komponenty oraz osiągnięte rezultaty.

Niniejszy raport stanowi pełne zagłębienie w projekt detektora składowych RGB, stworzonego z myślą o połączeniu teorii z praktyką.

Przedmiot Raportu: Raport ten skupia się na kompleksowym omówieniu procesu projektowania, implementacji i testowania detektora składowych RGB. Od teoretycznych fundamentów po szczegółowy opis części sprzętowej i programowej.

Zawartość Raportu:

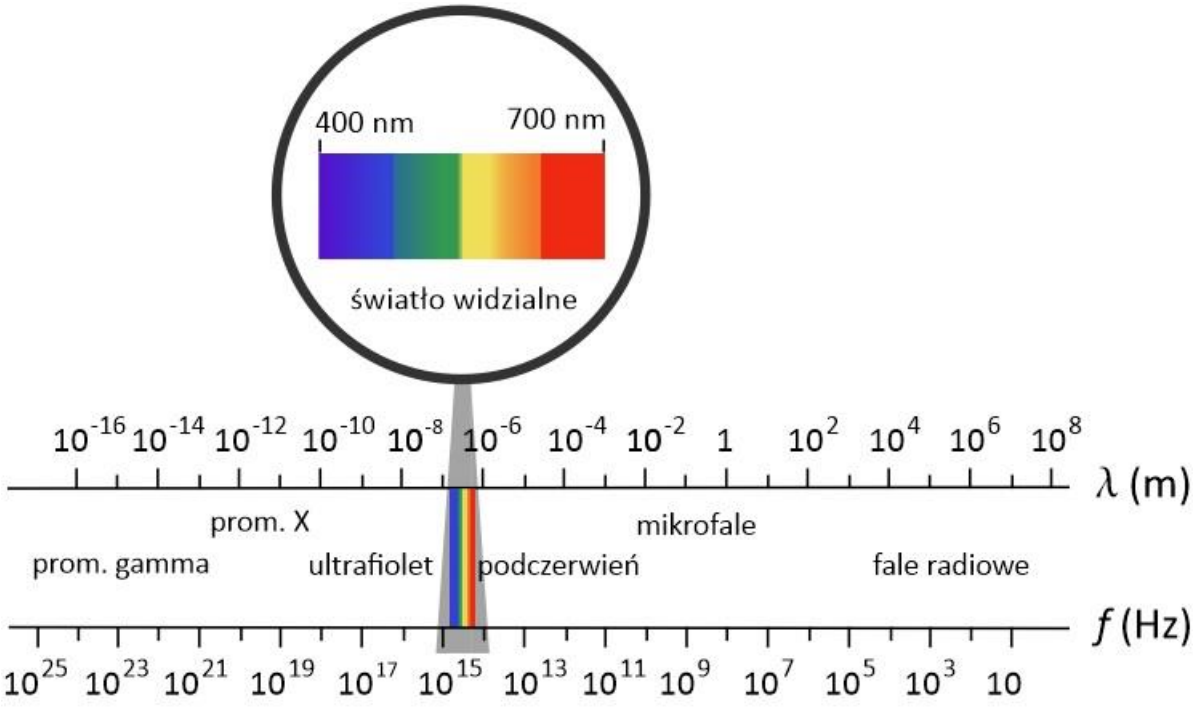
- **Wprowadzenie Teoretyczne:** Zapoznaj się z fundamentalnymi koncepcjami związanymi z detekcją kolorów RGB oraz zrozumienie teoretyczne zastosowanych technologii.
- **Założenia Projektowe:** Poznaj główne cele i założenia projektu, które kierowały procesem jego tworzenia.
- **Opis Części Sprzętowej:** Przejrzyj szczegółowy przegląd elementów składowych detektora, zanalizuj architekturę elektroniczną i połączenia między poszczególnymi komponentami.
- **Opis Części Programowej:** Dowiedz się, jakie platformy programistyczne zostały wykorzystane, jakie algorytmy programowe wspierają funkcjonowanie detektora.
- **Uruchomienie oraz Testy Urządzenia:** Śledź proces uruchomienia detektora, prześledź etapy testów, a także analizuj wyniki i dostosuj urządzenie do oczekiwań.
- **Specyfikacja Urządzenia:** Zdobądź pełen obraz technicznych parametrów detektora, zrozum jego charakterystyki działania i precyzji pomiarów.
- **Podsumowanie:** Oceń osiągnięcia projektu w kontekście założeń, zastanów się nad praktycznymi zastosowaniami detektora, oraz rozwijaniem podobnych projektów w przyszłości.

2. Wprowadzenie teoretyczne

Światło to rodzaj promieniowania elektromagnetycznego, które jest widoczne dla ludzkiego oka. Jest to forma energii, która manifestuje się w postaci fal elektromagnetycznych o określonych długościach fal. Widmo elektromagnetyczne obejmuje różne rodzaje fal, takie jak promieniowanie radiofalowe, mikrofalowe, podczerwone, światło widzialne, ultrafioletowe, rentgenowskie i gamma .

Światło widzialne, które jest postrzegane przez nasze oczy, zajmuje tylko niewielki fragment całego spektrum elektromagnetycznego. Widmo światła widzialnego obejmuje zakres długości fal około 380 do 750 nanometrów (Rysunek 1). W tym zakresie długości fal różne kolory są widoczne dla ludzkiego oka, przy czym fioletowy ma krótszą długość fali, a czerwony ma dłuższą.

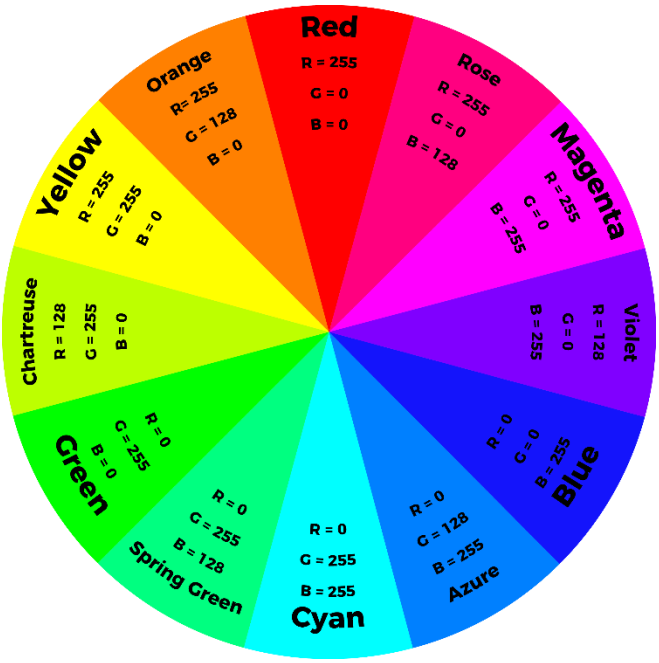
Światło może mieć różne źródła, takie jak Słońce, źródła sztuczne, lampy fluorescencyjne, świetlówki, diody elektroluminescencyjne (LED) itp. Istnieje wiele zastosowań światła w codziennym życiu i różnych dziedzinach, takich jak oświetlenie, fotografia, medycyna (terapia światłem), telekomunikacja (światłowody), fizyka (badania fal elektromagnetycznych), czy technologia wyświetlaczy.



Rysunek 1 Widmo fal elektromagnetycznych [1].

Przestrzenie barw to jednoznaczny opis barwy przedstawiony w układzie współrzędnych. W zależności od wybranego modelu układ może być dwu lub trójwymiarowy.

RBG to addytywny model przestrzeni barw, w którym kolor opisywany jest jako suma składowych koloru czerwonego, zielonego i niebieskiego (Rysunek 2). W modelu RGB każdy parametr może osiągać wartości z zakresu 0-255 czyli jest opisywana na 8 bitach. Ten sposób zapisu pozwala na opisanie ponad 16 milionów kolorów. Wartości RGB często zapisywane są w systemie heksadecymalnym, np. biały, który występuje gdy wszystkie 3 składowe osiągają maksymalną wartość, w systemie heksadecymalnym będzie opisany jako #FFFFFF.



Rysunek 2 Model barw RGB [2].

3. Założenia projektowe

Projekt zakładał stworzenie detektora składowych RGB, którego wyniki pomiarów miały być przedstawiane na wyświetlaczu ciekłokrystalicznym. Całość urządzenia miała być zasilana z baterii, co zapewniało przenośność i niezależność od źródła zasilania. Dodatkowo, zaplanowano, aby dioda RGB emitowała światło o barwie zbliżonej do koloru analizowanego, co dodatkowo ułatwiało wizualne monitorowanie wyników pomiarów. W ten sposób projekt nie tylko spełniał funkcję detektora kolorów, ale również zapewniał czytelność i praktyczność użytkowania dzięki zastosowaniu wyświetlacza LCD oraz mobilności dzięki zasilaniu bateryjnym.

Projekt zakładał rozbudowę detektora składowych RGB, celem, którego było przedstawienie wyników pomiarów zarówno na monitorze szeregowym w platformie Arduino, jak i na dodatkowym wyświetlaczu LCD. Kluczowym elementem koncepcji było zasilanie całego urządzenia z baterii o napięciu 9V, co miało zapewnić przenośność i niezależność od zewnętrznego źródła energii.

W dążeniu do doskonałości funkcjonalnej i estetycznej, wprowadziliśmy innowacyjne elementy, takie jak dioda RGB, emitująca światło o barwie zbliżonej do koloru analizowanego. To zaawansowane rozwiązanie nie tylko podnosi czytelność wyników pomiarów, ale także ułatwia wizualne monitorowanie procesu analizy.

3.1 Założenia funkcjonalne

* Pomiar Składowych RGB: Głównym celem projektu jest możliwość precyzyjnego pomiaru składowych kolorów RGB w badanym otoczeniu.

*Wyświetlanie Wyników Pomiarów na Monitorze Szeregowym Arduino: Projekcja rezultatów pomiarów na monitorze szeregowym w platformie Arduino, umożliwiająca użytkownikowi bieżące monitorowanie i analizę danych.

*Dodatkowy Wyświetlacz LCD: Integracja dodatkowego wyświetlacza LCD dla zwiększenia czytelności wyników pomiarów oraz zwiększenia interakcji użytkownika z detektorem.

*Zasilanie Bateryjne dla Przenośności: Zastosowanie zasilania bateryjnego (9V) w celu zapewnienia przenośności i niezależności od stałego źródła energii.

*Dodatkowa Dioda RGB do Wizualizacji Wyników: Wykorzystanie diody RGB do emitowania światła o barwie zbliżonej do koloru analizowanego, co ułatwi wizualne monitorowanie wyników pomiarów.

*Stabilność i Precyzja Pomiarów: Zapewnienie stabilności i precyzji w procesie pomiaru składowych RGB dla skutecznej identyfikacji kolorów.

*Czas Działania na Baterii: Zgodnie z założeniem, aby detektor był przenośny, zapewnienie odpowiedniego czasu działania na baterii.

*Odporność na Zmienne Warunki Otoczenia: Projektowanie detektora w taki sposób, aby był odporny na zmienne warunki otoczenia, takie jak oświetlenie.

*Możliwość Rozszerzeń Funkcjonalnych: Zaprojektowanie systemu z myślą o możliwości dodawania rozszerzeń funkcjonalnych w przyszłości, umożliwiających ewentualne udoskonalenia i rozbudowę detektora składowych RGB.

*Ważnym aspektem jest, że wyświetlacz LCD został zasilany bezpośrednio z platformy Arduino, a ta z kolei pobierała energię z baterii 9V. To zintegrowane podejście do zasilania eliminuje konieczność dodatkowego źródła energii dla wyświetlacza LCD, jednocześnie zwiększając mobilność i praktyczność detektora składowych RGB.

*Dzięki zastosowaniu monitora szeregowego w Arduino, wyświetlacza LCD zasilanego bezpośrednio z platformy Arduino, diody RGB oraz zasilaniu bateryjnemu, projekt nie tylko skutecznie spełnia

założenia detektora kolorów, ale również wyróżnia się profesjonalnym designem, doskonałą czytelnością wyników pomiarów, praktycznością użytkowania oraz wysokim stopniem mobilności.

3.2 Założenia konstrukcyjne projektu detektora składowych RGB:

*Dioda RGB z Rezystorami na Kolor 220ohm Czerwony i Zielony, Rezystor 440 Ohm dioda Niebieska:

Wykorzystanie diody RGB wraz z rezystorami o wartości 220 Ohm dla kolorów czerwonego i zielonego oraz rezystorem o wartości 440 Ohm dla koloru niebieskiego w celu precyzyjnej kontroli jasności i intensywności poszczególnych składowych kolorów kolor niebieski wprowadza bardzo duży rozstrzał więc zastosowaliśmy większy rezystor, żeby ograniczyć prąd.

*Platforma Arduino jako Układ Sterujący:

Zastosowanie platformy Arduino jako centralnego układu sterującego, umożliwiającego programowalność detektora i integrację z różnymi modułami.

*Zasilanie Baterią 9V:

Użycie baterii o napięciu 9V jako źródła zasilania, co zapewnia przenośność detektora i uniezależnia go od stałego źródła energii.

*Moduł Czujnika Koloru TCS3200:

Wykorzystanie modułu czujnika koloru TCS3200 do dokładnego pomiaru składowych RGB w otoczeniu, co zapewnia precyzję i niezawodność wyników pomiarów.

*Odpowiednie Połączenia Elektryczne:

Staranne zaplanowanie i wykonanie połączeń elektrycznych pomiędzy poszczególnymi elementami, zapewniające stabilną pracę detektora.

*Odpowiednia Ochrona Elektroniki:

Zabezpieczenie układu elektronicznego przed potencjalnymi przepięciami i zakłóceniami, aby zapewnić trwałość i niezawodność detektora.

*Integracja Zespołu Diody RGB z Wyświetlaczem LCD:

Sprawne połączenie diody RGB z wyświetlaczem LCD w celu umożliwienia wizualizacji wyników pomiarów w czasie rzeczywistym.

*Zastosowanie Interfejsu Monitora Szeregowego w Arduino:

Implementacja interfejsu monitora szeregowego w Arduino w celu bieżącej prezentacji wyników pomiarów i ułatwienia analizy przez użytkownika.

*Optymalizacja Zużycia Baterii:

Optymalizacja programu detektora oraz implementacja funkcji uśpienia w celu minimalizacji zużycia energii, co przekłada się na dłuższy czas pracy na baterii.

Współpraca Z Modułem TCS3200:

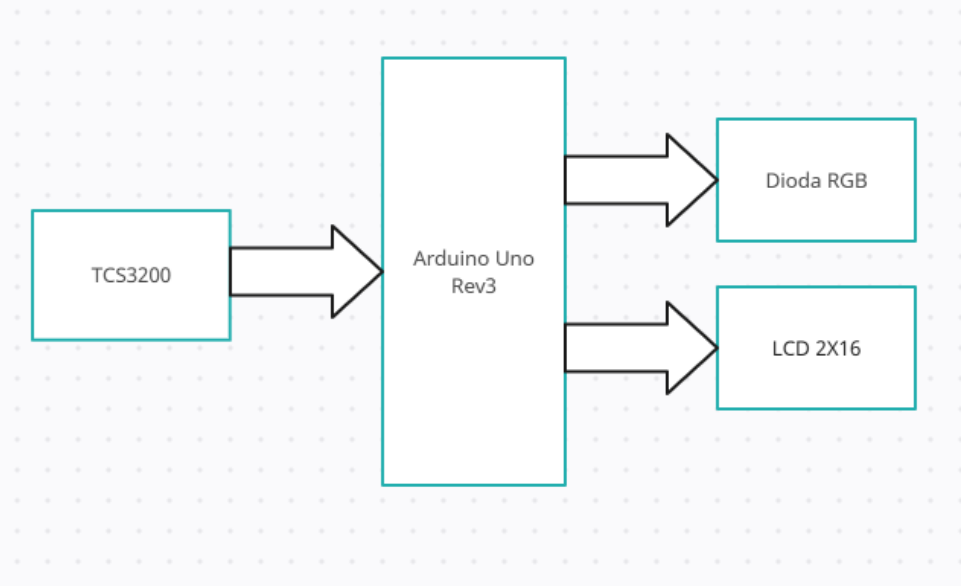
Skoordynowana współpraca między platformą Arduino a modułem TCS3200, umożliwiająca precyzyjne sterowanie i odczytywanie danych z czujnika koloru.

*Prosta i Wytrzymała Konstrukcja Mechaniczna:

Stworzenie prostego i wytrzymałego fizycznego obudowania detektora, które jednocześnie będzie łatwe w obsłudze oraz zabezpieczy wszystkie komponenty przed uszkodzeniami.

4. Opis części sprzętowej

Przedstawiony schemat blokowy (Rysunek 3) jest ukazaniem wizualnej reprezentacji poszczególnych komponentów i ich wzajemnych relacji w projekcie detektora składowych RGB. Ten zgrabny schemat blokowy służy jako czytelne mapowanie struktury części sprzętowej, ukazując harmonijną integrację poszczególnych elementów, co pozwala na łatwiejsze zrozumienie złożoności urządzenia.



Rysunek 3 Schemat blokowy detektora składowych RGB.

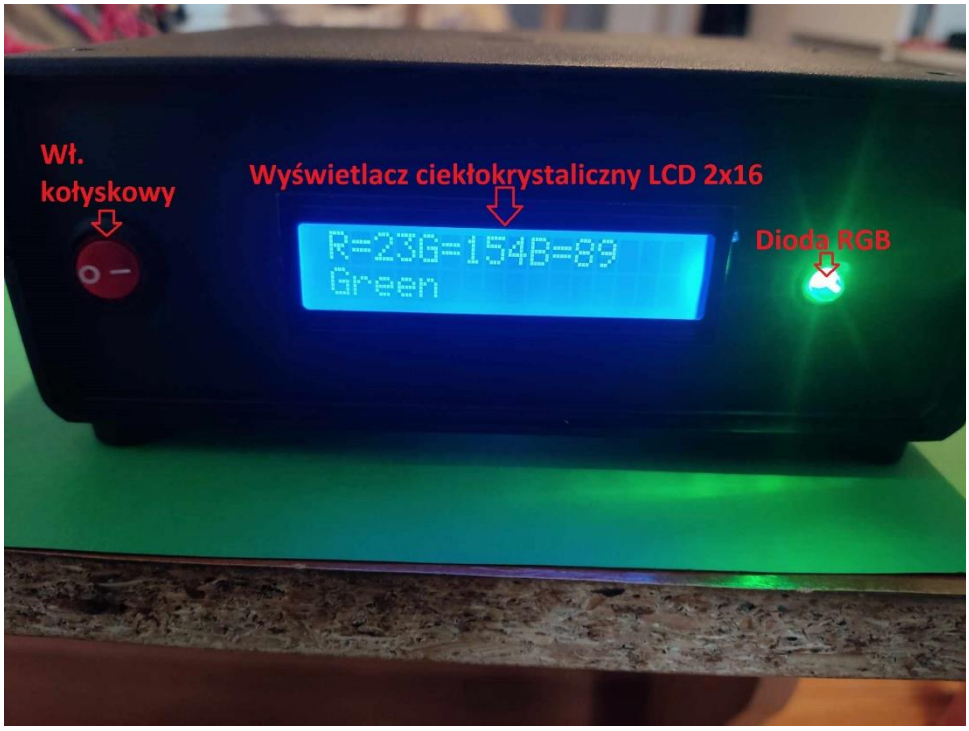
Urządzenie można podzielić na 4 bloki, jakimi są czujnik, mikrokontroler, LCD oraz dioda RGB.

a) Opis schematu blokowego:

Czujnik koloru TCS3200 przekazuje dane o intensywności RGB do mikrokontrolera Arduino, który poprzez sygnały PWM precyzyjnie reguluje diodę RGB, emitującą światło zbliżone do analizowanego koloru. Jednocześnie mikrokontroler przekazuje gotowe wyniki pomiarów do wyświetlacza LCD 2x16, umożliwiając czytelną prezentację danych.

b) Fotografie rzeczywistego układu:

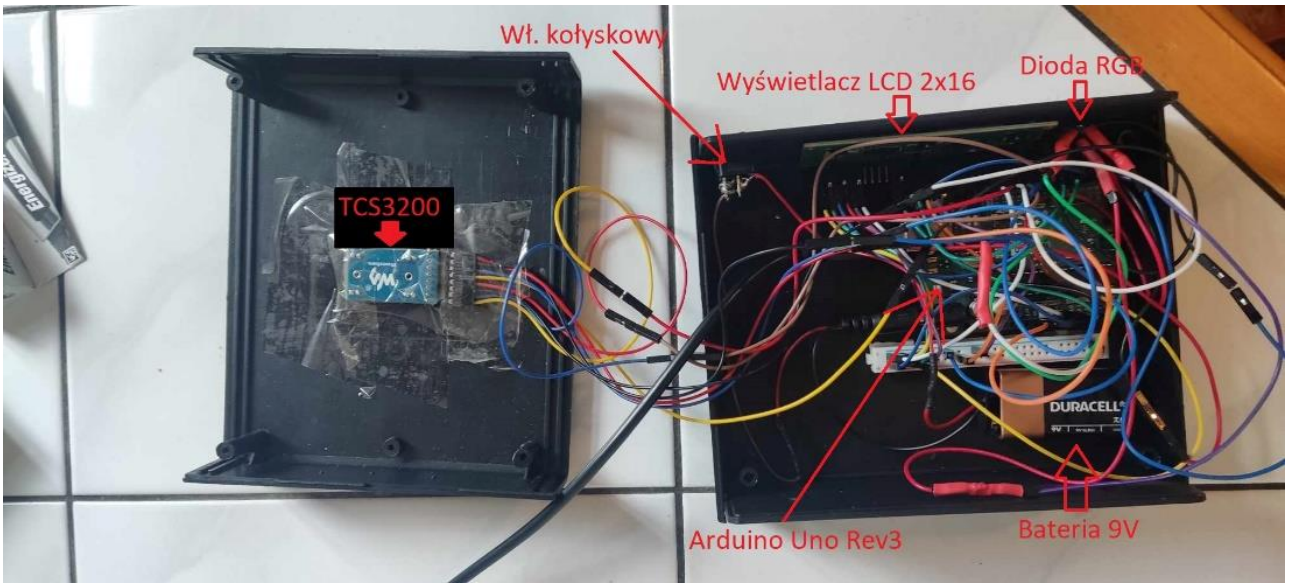
Rysunki 4-7 prezentują kolejno: Widok przodu urządzenia (Rysunek 4), Widok urządzenia z góry (Rysunek 5), Widok spodu urządzenia (Rysunek 6), oraz Widok wnętrza urządzenia (Rysunek 7).



Rysunek 4 Widok przodu urządzenia

Rysunek 5 Widok urządzenia z góry

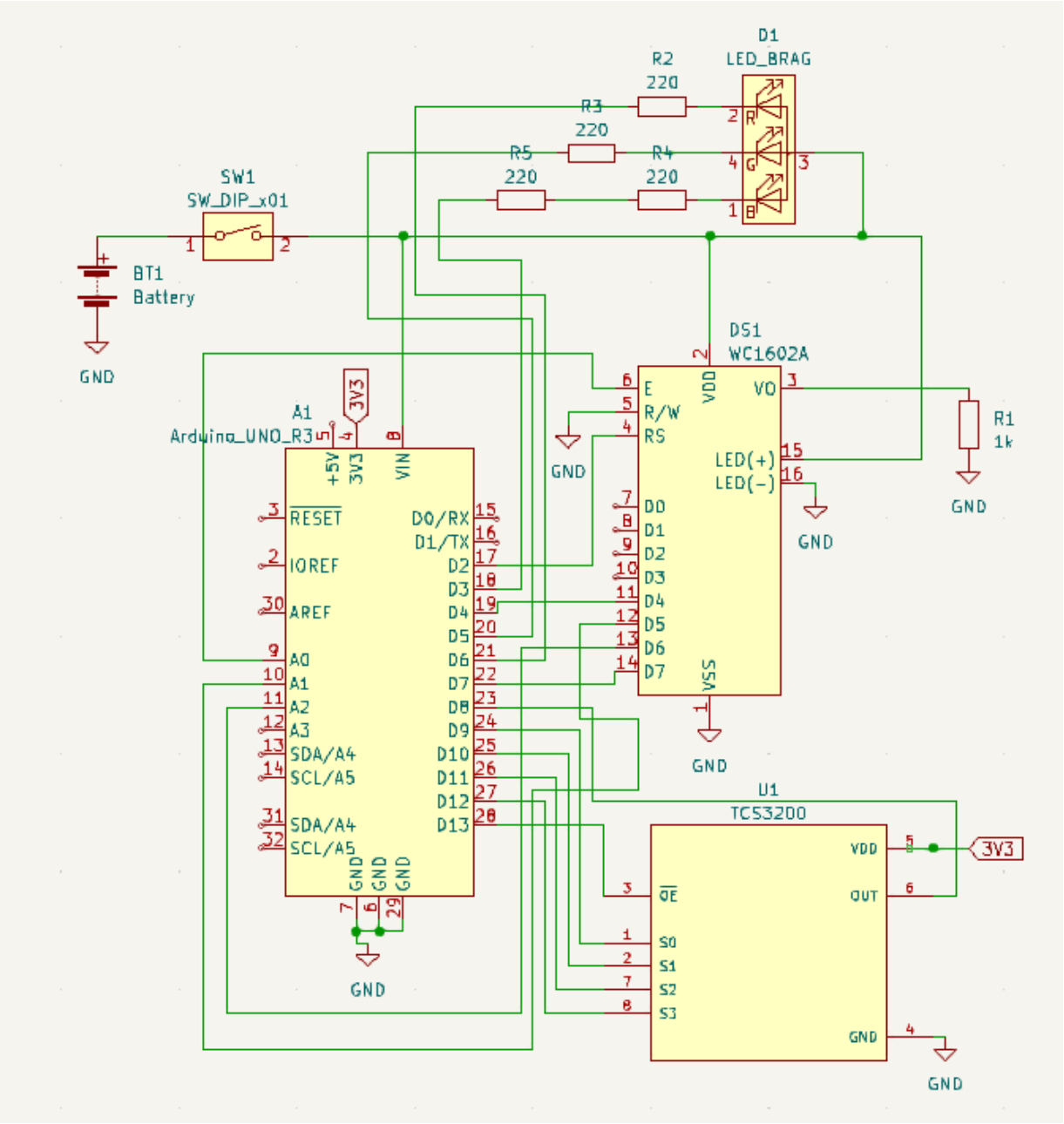
Rysunek 6 Widok spodu urządzenia



Rysunek 7 Widok środka urządzenia

4.1 Schemat ideowy

Przedstawiony schemat ideowy (Rysunek 8) ilustruje kompleksową strukturę detektora składowych RGB. Zestawienie elementów obejmuje m.in. moduł czujnika koloru TCS3200, diodę RGB oraz wyświetlacz LCD. Schemat ukazuje, jak poszczególne komponenty elektroniczne są ze sobą połączone, prezentując integralną architekturę urządzenia.



Rysunek 8 Schemat ideowy detektora składowych RGB

a) Opis schematu ideowego:

Arduino UNO rev3 (A1):

Odbiera i przetwarza dane z czujnika koloru TCS3200 oraz steruje diodą RGB.

Działa jako centralna jednostka sterująca.

TCS3200 (U1):

Odbiera światło z otoczenia, dokonując pomiaru składowych kolorów: czerwonej, zielonej i niebieskiej.

Wykorzystuje piny S0-S3 do ustawienia trybu pomiaru i generuje sygnał wyjściowy na pinie OUT.

LCD (DS1):

Wykorzystuje bibliotekę LiquidCrystal do komunikacji z Arduino.

Odbiera dane od Arduino i prezentuje wyniki pomiarów oraz nazwę koloru na dwuwierszowym wyświetlaczu LCD.

Dioda RGB (D1):

Składa się z trzech diod (czerwona, zielona, niebieska), emitujących światło o różnych barwach.

Odbiera sygnały od Arduino i reguluje intensywność emisji światła w zależności od wyników pomiarów.

Połączenia Elektryczne:

Poszczególne piny Arduino (A1) i komponentów (U1, DS1, D1) są połączone zgodnie z opisem podłączenia, zapewniając stabilną pracę układu.

Piny Arduino (A1) do Elementów:

A1_VIN: Zasilanie Arduino od 5V.

A1_3V3: Zasilanie czujnika TCS3200.

A1_A0, A1_A1, A1_A2: Sterowanie wyświetlaczem LCD.

A1_D2, A1_D4, A1_D5, A1_D6, A1_D7: Połączenia z wyświetlaczem LCD.

A1_D8: Odbiór sygnału z czujnika TCS3200.

A1_D9, A1_D10, A1_D11, A1_D12: Sterowanie trybem pomiaru czujnika TCS3200.

A1_GND: Wspólna masa układu.

Piny LCD (DS1):

DS1_E: Sterowanie wyświetlaczem.

DS1_D4, DS1_D5, DS1_D6, DS1_D7: Połączenia danych z wyświetlaczem.

DS1_RS: Kontrola trybu wyświetlania.

Piny TCS3200 (U1):

U1_OUT: Sygnał wyjściowy czujnika koloru.

U1_S0, U1_S1, U1_S2, U1_S3: Sterowanie trybem pomiaru.

U1_OE: Wyłączanie wyjścia czujnika.

U1_VDD: Zasilanie czujnika.

Piny Dioda RGB (D1):

RED, GREEN, BLUE: Sterowanie intensywnością emisji poszczególnych diod RGB.

b) Spis elementów

Tabela 1 Spis elementów wykorzystanych w projekcie.

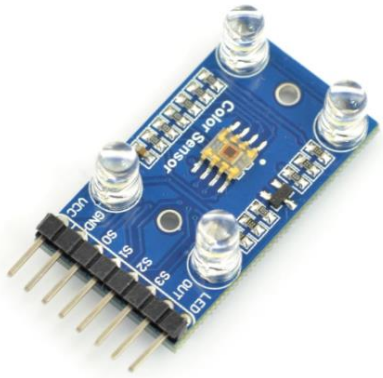
Symbol	Opis elementu	sztuk	wartość	jednostka
DS1	LCD 2x16	1	-	-
A1	Arduino uno rev.3	1	-	-
D1	Dioda RGB	1	-	-
BT1	Bateria alkaliczna 6LR61	1	9	V
R2,R3,R4,R5	Rezystor	4	220	Ω
R1	Rezystor	1	1	kΩ
U1	TCS3200	1	-	-
SW1	Włącznik kołyskowy	1	-	-

4.2 Kluczowe elementy

W niniejszym podrozdziale szczegółowo przedstawione są kluczowe elementy projektu detektora składowych RGB, a także ich istotna rola w zapewnieniu funkcjonalności i efektywności całego systemu. Profesjonalne podejście do opisu poszczególnych komponentów układu umożliwia zrozumienie ich funkcji oraz współpracy, co przekłada się na precyzję i niezawodność pomiarów kolorów.

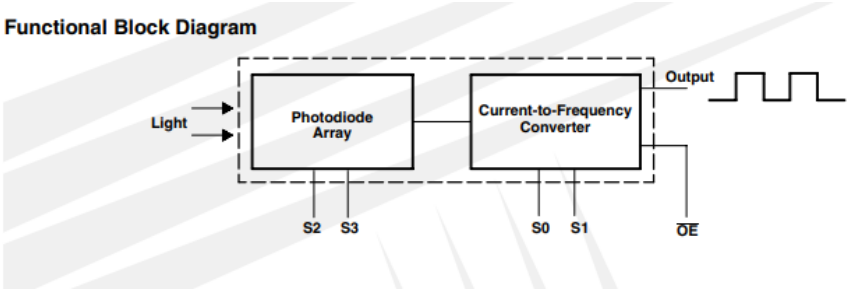
4.2.1 Czujnik koloru

Moduł z czujnikiem koloru umożliwia pomiar składowych RGB, wyjściem jest częstotliwość zależna od natężenia światła wybranej barwy. Obsługiwany przy pomocy dowolnego mikrokontrolera lub zestawu uruchomieniowego, w tym Arduino. W projekcie wykorzystano czujnik składowych RGB TCS3200 (Rysunek 9).



Rysunek 9 Czujnik składowych RGB TCS3200 [3].

Czujnik koloru TCS3200 to programowalny konwerter światła na częstotliwość, który łączy konfigurowalne fotodiody krzemowe i przetwornik prądu na częstotliwość w jednym monolitycznym układzie scalonym CMOS. Wyjście to fala kwadratowa (o współczynniku wypełnienia 50%), a częstotliwość jest proporcjonalna do natężenia światła (jego natężenia).



Rysunek 10 Blok funkcyjny czujnika [4].

*Specyfikacja czujnika koloru TCS3200

- Napięcie zasilania: 2,7 V do 5,5 V
- Wyprowadzenia: goldpin
- Prosta komunikacja z mikrokontrolerem (odczyt częstotliwości)
- Zintegrowane 4 diody LED oświetlające badany przedmiot
- Wymiary: 36 x 20 mm (bez złącz)

Moduł zasilany jest napięciem z zakresu 2,7 V do 5,5 V, dzięki temu sprawdza się zarówno w systemach 3,3 V i jak i 5 V. Czujnik można podłączyć do dowolnego zestawu uruchomieniowego z mikrokontrolerem wyposażonym w licznik (Timer) z wejściem częstotliwościowym, np. STM32Discovery lub Arduino. Wyjściem jest sygnał, którego częstotliwość zależna jest od natężenia światła, wybranej za pomocą wyprowadzeń S2 i S3, barwy.

*Podłączenie modułu

Szczegółowy opis wyprowadzeń dostępny jest w tabelach w dalszej części opisu. Wyprowadzeniami są popularne złącza goldpin pozwalające na połączeniu czujnika z modułem głównym za pomocą przewodów połączeniowych (w zestawie). Opis wyprowadzeń znajduje się w tabeli.

Podłączenie modułu

PIN	OPIS
VCCIO	Zasilanie od 2,7 V do 5,5 V.
GND	Masa układu.
LED	Kontrola 4 diod LED.
OUT	Wyjście częstotliwości koloru.
S0/S1	Wejścia służące do skalowania częstotliwości wyjściowej. Opis poniżej.
S2/S3	Wejścia służące do wyboru typu fotodiody. Opis poniżej.

Rysunek 11 Opis pinów modułu z czujnikiem TCS3200 [5].

*Konfiguracja czujnika koloru

Za pomocą wejść konfiguracyjnych S0 i S1 wybierany jest preskaler (dzielnik) wyjściowego sygnału częstotliwościowego, dzięki któremu możemy dobrać odpowiedni zakres częstotliwości dla wykorzystywanego mikrokontrolera. Wejścia S2 i S3 służą do wyboru rodzaju fotodiody pomiarowej: czerwonej, zielonej, niebieskiej lub opcji clear czyli bez filtracji (mierzone będą wszystkie składowe RGB jednocześnie).

Wybór dzielnika częstotliwości wyjściowej

S0	S1	SKALA CZĘSTOTLIWOŚCI WYJŚCIOWEJ W STOSUNKU DO FO
L		Czujnik wyłączony
L	H	2 % 10-12 kHz
H	L	20 % 100-120 kHz
H	H	100 % 500-600 kHz

Rysunek 12 Wybór skalowania częstotliwości sygnału wyjściowego czujnika [6].

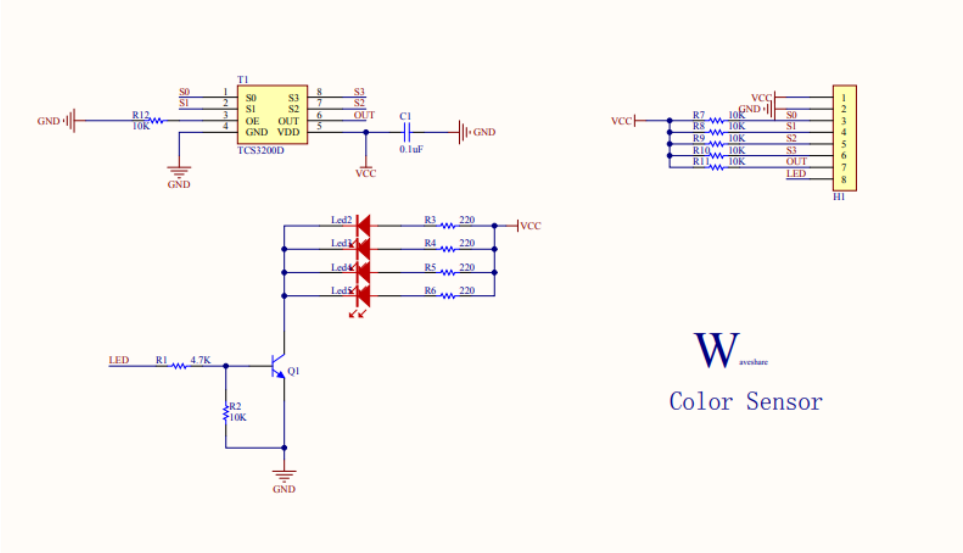
Wybór typu fotodiody (filtracji)

S0	S1	RODZAJ FOTODIODY
L	L	czerwona (R)
L	H	niebieska (B)
H	L	clear (RGB) (bez filtracji)
H	H	zielona (G)

L (Low) - stan niski

H (High) - stan niski

Rysunek 13 Wybór typu fotodioy (odczytywanego koloru) [7].



Rysunek 14 Schemat ideowy czujnika [8].

*Rola czujnika w projekcie.

Rola czujnika koloru TCS3200 w naszym projekcie detektora składowych RGB sprowadza się do precyzyjnego pomiaru intensywności światła w trzech fundamentalnych składowych kolorów: czerwonym (R), zielonym (G) i niebieskim (B). Główne zadania i rola czujnika obejmują:

1.Pomiar Intensywności Kolorów:

Czujnik TCS3200 jest odpowiedzialny za akwizycję danych dotyczących intensywności światła w poszczególnych składowych RGB. To umożliwia precyzyjne określenie proporcji i intensywności poszczególnych kolorów na analizowanej powierzchni.

2.Filtracja Kolorów:

Dzięki zastosowaniu filtrów kolorów, czujnik selektywnie rejestruje długości fal światła, co pozwala na skoncentrowane pomiaru składowych RGB. Filtry te są kluczowe do dokładnej identyfikacji barw.

3.Konwersja Analogowo-Cyfrowa (ADC):

Czujnik wykorzystuje wbudowany przetwornik analogowo-cyfrowy (ADC) do konwersji odczytów fotodiod na sygnały cyfrowe. To ułatwia dalsze przetwarzanie danych przez platformę Arduino.

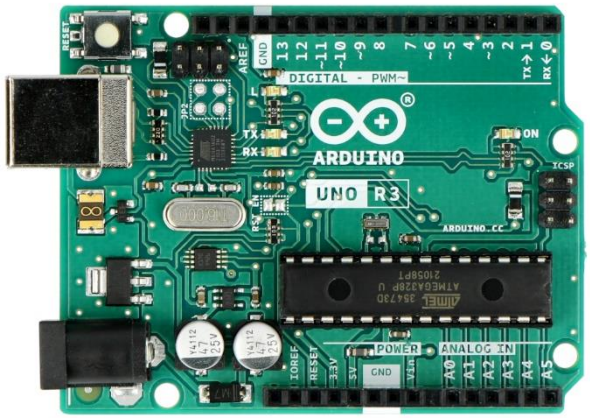
4.Synchronizacja z Diodą RGB:

Czujnik TCS3200 jest synchronizowany z diodą RGB, co zapewnia spójność pomiarów. Oznacza to, że pomiary składowych RGB są wykonywane jednocześnie, co przekłada się na precyzyjne wyniki pomiarów.

W skrócie, rola czujnika TCS3200 w naszym projekcie polega na zbieraniu dokładnych danych dotyczących intensywności składowych RGB, co umożliwia efektywną identyfikację kolorów i spełnienie założeń funkcjonalnych detektora składowych RGB.

4.2.2 Mikrokontroler

W projekcie wykorzystano mikrokontroler ATmega328P wbudowany w płytkę ewaluacyjną Arduino Uno rev3 (Rysunek 15).



Rysunek 15 Płytka ewaluacyjna Arduino UNO rev3 z procesorem ATmega328P [9].

***Specyfikacje techniczne:**

-Częstotliwość taktowania: Standardowo pracuje z częstotliwością taktowania 16 MHz.

-Pamięć programu: Posiada 32 KB pamięci Flash na programy użytkownika.

-Pamięć RAM: Wyposażony jest w 2 KB pamięci RAM do przechowywania danych w trakcie działania programu.

-Pamięć EEPROM: Posiada 1 KB pamięci EEPROM do przechowywania danych na stałe.

-Ilość pinów: Dostępnych jest 28 programowalnych pinów wejścia/wyjścia.

-Komunikacja szeregowa: Posiada interfejsy UART, SPI i I2C, co umożliwia komunikację z innymi urządzeniami.

-Maksymalny prąd 50mA.

***Czym jest:**

Blok mikrokontrolera odpowiada za odebranie częstotliwości natężenia koloru poszczególnych fotodiod od czujnika przez wejście cyfrowe lub konwerter analogowo cyfrowy. Mikrokontroler użyty w urządzeniu to ATmega328P wbudowany w płytkę ewaluacyjną Arduino Uno rev3. Jest to 8-bitowy mikrokontroler taktowany częstotliwością 16MHz. ATmega328P to popularny mikrokontroler produkowany przez firmę Microchip Technology, a wcześniej przez Atmel Corporation. Jest szeroko stosowany w projektach elektronicznych i prototypach, szczególnie w zastosowaniach związanych z platformą Arduino. Oto kilka kluczowych informacji dotyczących ATmega328P:

***Jak Działa:**

Mikrokontroler działa na zasadzie programowania, co oznacza, że jego zachowanie jest kontrolowane przez wcześniej napisany program, który jest wgrany do pamięci mikrokontrolera. Program ten wykonuje szereg instrukcji, reagując na dane wejściowe i wykonując odpowiednie akcje na wyjściach.

***Funkcje Bloku Arduino w Projekcie Detektora RGB:**

1. Zbieranie Danych: Arduino odbiera sygnały z modułu TCS3200, analizuje składowe kolorów (czerwoną, zieloną i niebieską) i przetwarza je na wartości numeryczne.

2. Sterowanie Działaniem Diod RGB: Mikrokontroler kontroluje diodę RGB, dostosowując intensywność poszczególnych składowych kolorów na podstawie odczytanych danych, co umożliwia wizualne przedstawienie wyników pomiarów.

3.Komunikacja z Wyświetlaczem LCD: Arduino zarządza wyświetlaczem LCD, prezentując użytkownikowi bieżące wyniki pomiarów oraz nazwę koloru w czytelnej formie.

4. Optymalizacja Zużycia Energii: Implementacja funkcji uśpienia oraz optymalizacja programu mają na celu minimalizację zużycia energii, co przekłada się na dłuższy czas pracy detektora na baterii.

5. Decyzje na Podstawie Wyników: Na podstawie odczytanych wartości składowych kolorów, Arduino podejmuje decyzje dotyczące nazwy koloru, co pozwala na bardziej złożoną analizę spektrum barw.

***Obsługa Wejść/Wyjść:**

Blok Arduino obsługuje wejścia od czujnika koloru TCS3200 oraz diody RGB, kontrolując sygnały i przekazując dane do odpowiednich pinów.

***Komunikacja z Wyświetlaczem LCD:**

Arduino jest odpowiedzialne za przesyłanie danych do wyświetlacza LCD, co pozwala na wizualizację wyników pomiarów i interakcję z użytkownikiem.

***Zasilanie Modułu Czujnika Koloru:**

Mikrokontroler zarządza zasilaniem modułu TCS3200, inicjuje pomiary oraz kontroluje cykl pracy detektora składowych RGB.

***Interfejs Monitora Szeregowego:**

Blok Arduino umożliwia korzystanie z interfejsu monitora szeregowego, co pozwala na monitorowanie i debugowanie pracy detektora poprzez komunikaty wysyłane do portu szeregowego komputera.

***Sczytywanie Danych:**

Arduino szczytuje dane z różnych źródeł, takich jak moduł TCS3200 Te dane są następnie przetwarzane według określonych algorytmów programu.

***Przetwarzanie Danych:**

Mikrokontroler przetwarza zebrane dane na podstawie wcześniej napisanego programu. W przypadku detektora składowych RGB, przetwarzanie obejmuje analizę pomiarów intensywności kolorów i ewentualne podejmowanie decyzji na podstawie tych danych.

***Sterowanie Wyjściami:**

Na podstawie wyników analizy, Arduino steruje diodą RGB oraz przesyła informacje do wyświetlacza LCD, umożliwiając wizualizację wyników dla użytkownika.

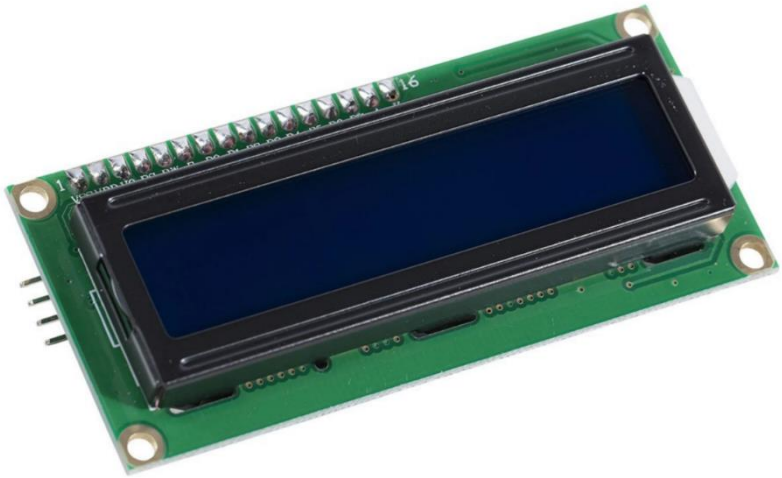
***Zarządzanie Zasilaniem:**

Mikrokontroler zarządza zasilaniem całego układu, w tym diody RGB, modułu TCS3200 oraz innych podłączonych elementów, aby zoptymalizować zużycie energii.

Blok mikrokontrolera Arduino stanowi centralny element sterujący w projekcie detektora składowych RGB, decydujący o współpracy poszczególnych komponentów i realizacji założeń funkcjonalnych projektu.

4.2.3 Wyświetlacz LCD 2x16

W projekcie wykorzystano Alfanumeryczny wyświetlacz LCD marki justPi. Prezentuje wartości w dwóch liniach, po 16 znaków w każdej. Zasilany jest napięciem 5 V. Charakteryzuje się wyjątkowo prostą obsługą, wysoką dostępnością oraz licznym wsparciem dla wielu mikrokontrolerów. Znaki wyświetlane są na niebieskim tle.



Rysunek 16 Wyświetlacz LCD [10].

***Czym jest:**

Wyświetlacz szeregowy LCD (Liquid Crystal Display) 2x16 to ekran ciekłokrystaliczny o wymiarach 2 linii po 16 znaków, z wbudowanym kontrolerem umożliwiającym komunikację szeregową. W

projekcie detektora RGB pełni funkcję interfejsu użytkownika, prezentując wyniki pomiarów oraz informacje diagnostyczne.

***Jak działa:**

Wyświetlacz odbiera sygnały szeregowe od mikrokontrolera Arduino i na ich podstawie steruje pikselami ciekłokrystalicznymi, tworząc znaki i symbole na ekranie.

***Funkcje Bloku Wyświetlacza LCD w Projekcie Detektora RGB:**

1. Prezentacja Wyników Pomiarów: Wyświetlacz LCD jest odpowiedzialny za prezentację wyników pomiarów składowych RGB w czasie rzeczywistym, umożliwiając użytkownikowi bieżącą ocenę analizowanych barw.

2. Komunikaty Diagnostyczne: W przypadku wystąpienia błędów lub sytuacji wyjątkowych, wyświetlacz informuje użytkownika o stanie systemu poprzez wyświetlanie stosownych komunikatów diagnostycznych.

3. Interakcja z Użytkownikiem: Umożliwia interakcję z użytkownikiem, na przykład przez wyświetlanie instrukcji kalibracji, statusu baterii lub innych komunikatów dotyczących działania detektora składowych RGB.

***Sczytywanie Danych:**

Wyświetlacz odbiera dane od mikrokontrolera poprzez protokół komunikacyjny szeregowy, prezentując informacje w postaci tekstowej na ekranie.

***Przetwarzanie i Formatowanie Tekstu:**

Zajmuje się formatowaniem danych i tekstu, aby prezentować wyniki pomiarów w czytelnej i zrozumiałej formie. Może również formatować komunikaty diagnostyczne, aby ułatwić zrozumienie informacji przez użytkownika.

***Sterowanie Podświetleniem:**

Wyposażony w kontrolę podświetlenia, co umożliwia zapewnienie odpowiedniego kontrastu i czytelności w różnych warunkach oświetleniowych (wykorzystaliśmy rezystor 1 kΩ).

***Integracja z Arduino:**

Komunikuje się z mikrokontrolerem Arduino poprzez przewody szeregowe, co umożliwia przesyłanie danych i instrukcji do wyświetlacza.

***Podświetlanie Niebieskie:**

W przypadku detektora składowych RGB, wyświetlacz charakteryzuje się podświetleniem w kolorze niebieskim, co harmonizuje z ogólnym projektem i estetyką detektora.

***Prosta i Efektywna Prezentacja Informacji:**

Stanowi efektywny sposób prezentacji danych, szczególnie tam, gdzie czytelność i natychmiastowa dostępność informacji są kluczowe.

Blok wyświetlacza szeregowego LCD 2x16 odgrywa zatem istotną rolę w interakcji z użytkownikiem, prezentując wyniki pomiarów oraz umożliwiając skuteczną diagnostykę i obsługę detektora składowych RGB.

4.2.4 Dioda RGB

W projekcie wykorzystano diodę RGB ze wspólną anodą.

Rysunek 17 Dioda RGB [11].

***Czym Jest:**

Dioda LED 5 mm RGB ze wspólną anodą to trójbarwna dioda emitująca światło w kolorach czerwonym, zielonym i niebieskim. Posiada wspólną anodę, co oznacza, że anoda diody RGB jest połączona wspólnie, a sterowanie poszczególnymi kolorami odbywa się poprzez zmianę poziomu napięcia na katodach.

***Jak Działa:**

Działa na zasadzie elektroluminescencji półprzewodników. Każdy kolor (czerwony, zielony, niebieski) jest generowany poprzez pobudzenie odpowiedniego materiału półprzewodnikowego, co powoduje emisję światła.

***Funkcje Diody RGB w Projekcie Detektora RGB:**

Mieszanie Kolorów: Dioda RGB umożliwia mieszanie kolorów podstawowych (czerwonego, zielonego, niebieskiego) w różnych proporcjach, co pozwala uzyskać szeroką gamę kolorów.

Sterowanie Jasnością: Poprzez regulację napięcia podawanego na diodę (PWM), można kontrolować jasność emitowanego światła, co jest istotne w projekcie detektora składowych RGB.

***Parametry Kolorów:**

Kolor Czerwony:

Długość Emitowanej Fali: 625 nm

Jasność: 100 mcd

Kąt Świecenia: 30°

Parametry Pracy:

Prąd If: 25 mA

Napięcie Vf: 2,0 V

Kolor Zielony:

Długość Emitowanej Fali: 525 nm

Jasność: 800 mcd

Kąt Świecenia: 30°

Parametry Pracy:

Prąd If: 25 mA

Napięcie Vf: 3,5 V

Kolor Niebieski:

Długość Emitowanej Fali: 470 nm

Jasność: 20 mcd

Kąt Świecenia: 30°

Parametry Pracy:

Prąd If: 25 mA

Napięcie Vf: 3,5 V

4.2.5 Bateria

W projekcie wykorzystano baterie alkaliczną energizer max plus 9V 6LR61 (650 mAh)

Rysunek 18 Bateria użyta w urządzeniu [12].

***Czym Jest:**

Bateria 9V jest źródłem zasilania dla całego projektu detektora składowych RGB. Zapewnia energię elektryczną niezbędną do pracy mikrokontrolera Arduino, diod RGB, wyświetlacza LCD oraz innych komponentów.

*** Rodzaj Baterii:**

Typ Baterii: 9V (Zakładamy, że to bateria zasilająca cały układ).

*** Jak Działa:**

Bateria 9V dostarcza stałe napięcie elektryczne potrzebne do zasilania wszystkich komponentów w projekcie. W przypadku detektora RGB, bateria jest źródłem przenośnym, co umożliwia niezależność od źródła zasilania.

*** Dlaczego Jest Potrzebna:**

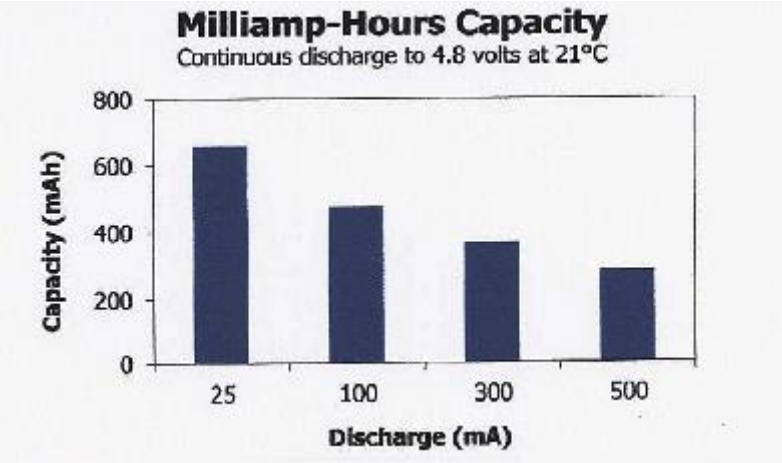
Bateria 9V jest niezbędna, aby umożliwić przenośne i niezależne od źródła zasilania działanie detektora składowych RGB. Zapewnia energię elektryczną dla mikrokontrolera Arduino, diod RGB, wyświetlacza LCD oraz innych podłączonych elementów.

*** Parametry Baterii:**

Napięcie: 9V (Średnie napięcie baterii 9V w trakcie użytkowania).

Typ Baterii: Alkaline lub inny odpowiedni typ.

Pojemność: 650 mAh przy poborze 100 mA pojemność około 500 mAh (Rysunek 19)



Rysunek 19 Zależność pojemności baterii od prądu rozładowania [13].

Typ: alkaliczna

Bezpieczny maksymalny pobór prądu: 100-150 mAh.

4.2.6 Włącznik kołyskowy

W projekcie wykorzystano włącznik kołyskowy.

Rysunek 20 Włącznik kołyskowy wykorzystany w projekcie [14].

***Czym Jest:**

Włącznik kołyskowy 250V 6A to element, który pełni funkcję przełącznika w obwodzie elektrycznym projektu detektora składowych RGB. Pozwala na kontrolowane włączanie i wyłączanie zasilania urządzenia.

***Specyfikacja Techniczna:**

Napięcie Pracy: 250V

Prąd Pracy: 6A

Typ Włącznika: Kołyskowy (mechaniczny, z włącznikiem/wyłącznikiem w formie dźwigni).

***Jak Działa:**

Włącznik kołyskowy jest urządzeniem mechanicznym, którego dźwignia pozwala na fizyczne połączenie lub rozłączenie obwodu elektrycznego. Włączanie i wyłączanie urządzenia odbywa się poprzez ruch dźwigni.

***Dlaczego Jest Potrzebny:**

Włącznik kołyskowy jest niezbędny do umożliwienia kontrolowanego włączania i wyłączania zasilania detektora składowych RGB. Zapewnia łatwy dostęp do kontroli nad urządzeniem oraz umożliwia oszczędność energii, gdy urządzenie nie jest w użyciu.

***Zastosowanie w Projekcie Detektora RGB:**

Włącznik kołyskowy stanowi kluczowy element projektu detektora składowych RGB, umożliwiając łatwe zarządzanie zasilaniem urządzenia. Jego umieszczenie na obudowie detektora ułatwia dostęp i obsługę.

5. Opis części programowej

5.1 Środowisko programistyczne

Do napisania kodu, jego kompilacji oraz zaimplementowania programu na mikrokontrolerze wykorzystano język programowania C w środowisku Arduino IDE, stworzonym przez producenta mikrokontrolera. Arduino IDE to zaawansowane narzędzie, które oferuje użytkownikowi szeroki zakres możliwości, w tym dostęp do gotowych szkiców, bibliotek oraz rozbudowanego forum, gdzie można uzyskać pomoc w razie potrzeby. Listing kodu został umieszczony w dodatkach.

5.2 Algorytm działania programu

Na poniższym diagramie (Rysunek 21) przedstawiono algorytm, na którym opiera się kod projektu.

Rysunek 21 Diagram przedstawiający algorytm programu

5.3 Kod

W niniejszym rozdziale zostaną szczegółowo opisane kolejne bloki programu:

```
1   #include <LiquidCrystal.h>
2   LiquidCrystal lcd(2, A0, 4, A1, A2, 7);
3   #define S0 9
4   #define S1 10
5   #define S2 11
6   #define S3 12
7   #define OUT 8
8   #define LED 13
9
10  #define RED 6
11  #define GREEN 5
12  #define BLUE 3
```

Rysunek 22 Fragment kodu odpowiedzialny za definiowanie pinów.

W przedstawionym fragmencie kodu (Rysunek 22) zdefiniowano piny mikrokontrolera Arduino UNO rev3, które są używane w projekcie. Dla obsługi wyświetlacza LCD wykorzystano piny cyfrowe 2, 4 i 7, a także piny analogowe A0, A1 i A2. Czujnik koloru TCS3200 jest sterowany za pomocą pinów cyfrowych od 8 do 13, natomiast do kontrolowania diody RGB używane są piny cyfrowe PWM 3, 5 i 6.

```
14  long int HALF_PERIOD = 0;
15  int RED_VALUE = 0;
16  int GREEN_VALUE = 0;
17  int BLUE_VALUE = 0;
18  int RED_LED_VALUE = 0;
19  int GREEN_LED_VALUE = 0;
20  int BLUE_LED_VALUE = 0;
21  String COLOR_NAME;
```

Rysunek 23 Fragment kodu odpowiedzialny za deklarację zmiennych.

Następnie (Rysunek 23) zadeklarowano zmienne użyte w projekcie. Do składowania danych odczytanych z czujnika została utworzona zmienna **HALF_PERIOD**, do której w dalszej części kodu zapisywana jest wartość odczytana z czujnika za pomocą funkcji **pulseIn()**. Zmienne typu int **RED_VALUE**, **GREEN_VALUE** oraz **BLUE_VALUE** będą przechowywać składowe RGB badanego koloru, natomiast zmienne **RED_LED_VALUE**, **GREEN_LED_VALUE** oraz **BLUE_LED_VALUE** będą zawierały wartości, które później będą dostarczane do diody RGB. String **COLOR_NAME** będzie przechowywać nazwę koloru na podstawie zawartości składowych RGB.

```
23  void setup() {
24
25    pinMode(S0, OUTPUT);
26    pinMode(S1, OUTPUT);
27    pinMode(S2, OUTPUT);
28    pinMode(S3, OUTPUT);
29    pinMode(LED, OUTPUT);
30    pinMode(OUT, INPUT);
31    lcd.begin(16, 2);
32
33    digitalWrite(S0, HIGH);
34    digitalWrite(S1, LOW);
35
36    digitalWrite(LED,HIGH );
37
38    Serial.begin(9600);
39  }
```

Rysunek 24 Fragment kodu przy inicjalizacji programu.

W funkcji **setup()** (Rysunek 24) przydzielono pinom podłączanym do czujnika tryb (**OUTPUT** lub **INPUT**) oraz wywołano funkcję rozpoczynającą działanie wyświetlacza. Do pinu S0 przypisano wartość **HIGH** a do S1 **LOW** w celu ustawienia skalowania częstotliwości sygnału zwracanego przez czujnik na 20%. Przypisanie stanu wysokiego do pinu LED powoduje zapalenie się ledów dołączonych do modułu czujnika. Rozpoczynamy również przesyłanie informacji do monitora szeregowego z prędkością bitową 9600 bps.

```
43    digitalWrite(S2, LOW);
44    digitalWrite(S3, LOW);
45    HALF_PERIOD = pulseIn(OUT, LOW);
46    RED_VALUE = 255-HALF_PERIOD;
47    if (RED_VALUE < 0) {
48      RED_VALUE = 0;
49    }
50
51    Serial.print("R = ");
52    Serial.print(RED_VALUE);
53    Serial.print("   ");
54
55    delay(100);
```

Rysunek 25 Fragment kodu odpowiadający za odczyt składowej czerwonej.

Zgodnie z dokumentacją czujnika TCS3200 ustawienie wartości LOW na pinach S2 oraz S3 ustawia czujnik na odczyt wartości czerwonej. Czujnik zwraca sygnał prostokątny (duty cycle = 50%) o częstotliwości zależnej od natężenia badanej składowej. Za pomocą funkcji **pulseIn()** odczytujemy czas trwania sygnału niskiego, czyli ½ okresu badanego sygnału. (Rysunek 25)

- **digitalWrite(S2, LOW); oraz digitalWrite(S3, LOW);**
    - Ustawiają konkretny tryb pracy czujnika. W tym przypadku, oba ustawienia na LOW oznaczają, że czujnik działa w trybie odczytu czerwonego koloru.

- **HALF_PERIOD = pulseIn(OUT, LOW);**
    - Mierzy czas trwania impulsów o niskim stanie logicznym na pinie OUT. Ta wartość jest następnie używana do określenia intensywności odbieranego światła, co w rezultacie odpowiada czerwonej części widma.
- **RED_VALUE = 255 - HALF_PERIOD;**
    - Odwraca odczytaną wartość, ponieważ czujnik TCS3000 zwraca większą wartość dla ciemniejszych kolorów. Następnie wartość ta jest przypisywana do zmiennej **RED_VALUE**.
- **if (RED_VALUE < 0) { RED_VALUE = 0; }**
    - Zapobiega sytuacji, w której odczytana wartość jest poniżej zera. Jeśli wartość była ujemna, zostaje ustawiona na zero.
- **Serial.print("R = "); Serial.print(RED_VALUE); Serial.print(" ");**
    - Wypisuje odczytaną wartość czerwonego koloru na monitor szeregowy (Serial Monitor) w środowisku Arduino.
- **delay(100);**
    - Krótka przerwa, aby uniknąć zbyt szybkiego odczytu kolejnych kolorów.

Powtórzone są te same kroki dla zielonego (**GREEN_VALUE**) (Rysunek 26) i niebieskiego (**BLUE_VALUE**) (Rysunek 27) koloru, z odpowiednimi zmianami w ustawieniach pinów (S2 i S3) czujnika.

```
59  digitalWrite(S2, HIGH);
60  digitalWrite(S3, HIGH);
61
62  HALF_PERIOD = pulseIn(OUT, LOW);
63  GREEN_VALUE = 255-HALF_PERIOD;
64  if (GREEN_VALUE < 0) {
65    GREEN_VALUE = 0;
66  }
67
68  Serial.print("G = ");
69  Serial.print(GREEN_VALUE);
70  Serial.print("   ");
71  delay(100);
```

Rysunek 26 Fragment kodu odpowiadający za odczyt składowej zielonej.

```
74  digitalWrite(S2, LOW);
75  digitalWrite(S3, HIGH);
76
77  HALF_PERIOD = pulseIn(OUT, LOW);
78  BLUE_VALUE = 255-HALF_PERIOD;
79  if (BLUE_VALUE < 0) {
80    BLUE_VALUE = 0;
81  }
82
83  Serial.print("B = ");
84  Serial.print(BLUE_VALUE);
85  Serial.println("   ");
86  delay(100);
```

Rysunek 27 Fragment kodu odpowiadający za odczyt składowej niebieskiej.

```
88    RED_LED_VALUE=RED_VALUE;
89    GREEN_LED_VALUE=GREEN_VALUE;
90    BLUE_LED_VALUE=BLUE_VALUE;
```

Rysunek 28 Fragment kodu odpowiadający za zmienne używane do sterowania diodami LED RGB.

W tym fragmencie kodu (Rysunek 28) przypisywane są wartości odczytane z czujnika kolorów **(RED_VALUE, GREEN_VALUE, BLUE_VALUE**) do zmiennych, które będą używane do sterowania diodą LED RGB (**RED_LED_VALUE, GREEN_LED_VALUE, BLUE_LED_VALUE**). Te zmienne są odpowiedzialne za ustawienie intensywności światła diody LED RGB, co w rezultacie wpływa na kolor emitowany przez diodę.

Następny fragment kodu (Rysunki 29-40) zawiera serię warunków if-else, które sprawdzają wartości odczytane z czujnika kolorów (czerwony, zielony i niebieski) i przypisują im odpowiednie nazwy kolorów. Dodatkowo, w niektórych przypadkach, są dokonywane pewne modyfikacje wartości dla diod LED RGB, aby uzyskać pożądany efekt kolorystyczny. Oto opis poszczególnych warunków:

```
93    if(RED_VALUE < 30 && GREEN_VALUE < 30 && BLUE_VALUE < 30)
94    {
95      COLOR_NAME = "Black";
96    }
```

Rysunek 29 Fragment kodu odpowiadający za warunek dla koloru czarnego.

```
97    else if(RED_VALUE > 165 && GREEN_VALUE > 190 && BLUE_VALUE > 190)
98    {
99      COLOR_NAME = "White";
100   }
```

Rysunek 30 Fragment kodu odpowiadający za warunek dla koloru białego.

```
101   else if(RED_VALUE >= 0 && RED_VALUE < 20 && GREEN_VALUE > 130 && GREEN_VALUE < 180 && BLUE_VALUE > 180 && BLUE_VALUE < 255)
102   {
103     COLOR_NAME = "Blue";
104   }
```

Rysunek 31 Fragment kodu odpowiadający za warunek dla koloru niebieskiego.

```
105     else if(RED_VALUE > 45 && RED_VALUE < 160 && GREEN_VALUE > 170 && GREEN_VALUE < 210 && BLUE_VALUE > 195 && BLUE_VALUE < 225)
106   {
107     COLOR_NAME = "Light Blue";
108   }
```

Rysunek 32 Fragment kodu odpowiadający za warunek dla koloru jasnoniebieskiego.

```
109     else if(RED_VALUE > 140 && RED_VALUE < 225 && GREEN_VALUE >= 0 && GREEN_VALUE < 20 && BLUE_VALUE >= 0 && BLUE_VALUE < 28)
110   {
111     COLOR_NAME = "Red";
112     BLUE_LED_VALUE=0;
113   }
```

Rysunek 33 Fragment kodu odpowiadający za warunek dla koloru czerwonego, a także modyfikację wartości dla diody RGB.

```
114   else if(RED_VALUE > 160 && RED_VALUE < 205 && GREEN_VALUE > 60 && GREEN_VALUE < 115 && BLUE_VALUE > 25 && BLUE_VALUE < 90)
115   {
116     COLOR_NAME = "Orange";
117     RED_LED_VALUE=160;
118     GREEN_LED_VALUE=37;
119     BLUE_LED_VALUE=0;
120   }
```

Rysunek 34 Fragment kodu odpowiadający za warunek dla koloru pomarańczowego, a także modyfikację wartości dla diody RGB.

```
121   else if(RED_VALUE > 40 && RED_VALUE < 90 && GREEN_VALUE >= 0 && GREEN_VALUE < 25 && BLUE_VALUE >= 0 && BLUE_VALUE < 20)
122   {
123     COLOR_NAME = "Brown";
124     RED_LED_VALUE=70;
125     GREEN_LED_VALUE=12;
126     BLUE_LED_VALUE=0;
127   }
```

Rysunek 35 Fragment kodu odpowiadający za warunek dla koloru brązowego, a także modyfikację wartości dla diody RGB.

```
128   else if(RED_VALUE > 0 && RED_VALUE < 50 && GREEN_VALUE > 130 && GREEN_VALUE < 170 && BLUE_VALUE > 60 && BLUE_VALUE < 115)
129   {
130     COLOR_NAME = "Green";
131     RED_LED_VALUE=0;
132     GREEN_LED_VALUE=75;
133     BLUE_LED_VALUE=0;
134   }
```

Rysunek 36 Fragment kodu odpowiadający za warunek dla koloru zielonego, a także modyfikację wartości dla diody RGB.

```
135   else if(RED_VALUE > 120 && RED_VALUE < 170 && GREEN_VALUE > 170 && GREEN_VALUE < 210 && BLUE_VALUE > 100 && BLUE_VALUE < 150)
136   {
137     COLOR_NAME = "Light Green";
138     RED_LED_VALUE=54;
139     GREEN_LED_VALUE=130;
140     BLUE_LED_VALUE=18;
141   }
```

Rysunek 37 Fragment kodu odpowiadający za warunek dla koloru jasnozielonego, a także modyfikację wartości dla diody RGB.

```
142   else if(RED_VALUE > 175 && RED_VALUE < 205 && GREEN_VALUE > 120 && GREEN_VALUE < 165 && BLUE_VALUE > 160 && BLUE_VALUE < 200)
143   {
144     COLOR_NAME = "Pink";
145     RED_LED_VALUE=148;
146     GREEN_LED_VALUE=7;
147     BLUE_LED_VALUE=50;
148   }
```

Rysunek 38 Fragment kodu odpowiadający za warunek dla koloru różowego, a także modyfikację wartości dla diody RGB.

```
149   else if(RED_VALUE > 180 && RED_VALUE < 220 && GREEN_VALUE > 180 && GREEN_VALUE < 220 && BLUE_VALUE > 100 && BLUE_VALUE < 140)
150   {
151     COLOR_NAME = "Yellow";
152     RED_LED_VALUE=66;
153     GREEN_LED_VALUE =55;
154     BLUE_LED_VALUE=0;
155   }
```

Rysunek 39 Fragment kodu odpowiadający za warunek dla koloru żółtego, a także modyfikację wartości dla diody RGB.

```
156   else
157   {
158     COLOR_NAME = "Different Color";
159   }
```

Rysunek 40 Fragment kodu odpowiadający za warunek domyślny (Different Color).

```
161   analogWrite(RED, 255-RED_LED_VALUE);
162   analogWrite(GREEN, 255-GREEN_LED_VALUE);
163   analogWrite(BLUE, 255-BLUE_LED_VALUE);
```

Rysunek 41 Fragment kodu odpowiadający za sterowanie diodą RGB.

W tym fragmencie kodu (Rysunek 41) wykorzystywane są funkcje **analogWrite()** do sterowania diodami LED RGB w zależności od wykrytego koloru. Oto jak to działa:

analogWrite(RED, 255-RED_LED_VALUE);

Sterowanie intensywnością czerwonej diody LED RGB. Wartość **RED_LED_VALUE** została wcześniej obliczona na podstawie odczytu koloru. Funkcja **analogWrite()** przyjmuje wartość od 0 (brak światła) do 255 (pełna jasność). Ponieważ zakłada się, że 0 oznacza pełną jasność, a 255 brak światła, wartość **255 - RED_LED_VALUE** jest używana, aby odwrócić tę logikę.

analogWrite(GREEN, 255-GREEN_LED_VALUE);

Analogicznie jak w przypadku czerwonej diody, ta linia kodu kontroluje intensywność zielonej diody LED RGB.

analogWrite(BLUE, 255 - BLUE_LED_VALUE);

Analogicznie jak w przypadku czerwonej i zielonej diody, ta linia kodu kontroluje intensywność niebieskiej diody LED RGB.

```
166   lcd.setCursor(0, 0);
167   lcd.print("R=");
168   lcd.print(RED_VALUE);
169   lcd.print("G=");
170   lcd.print(GREEN_VALUE);
171   lcd.print("B=");
172   lcd.print(BLUE_VALUE);
173   lcd.setCursor(0, 1);
174   lcd.print(COLOR_NAME);
175   delay(300);
176   lcd.clear();
```

Rysunek 42 Fragment kodu odpowiadający za wyświetlanie informacji na wyświetlaczu LCD.

W tym fragmencie kodu (Rysunek 42) zostały użyte następujące funkcje:

- **lcd.setCursor(0, 0);**

Ustawia kursor na ekranie LCD w pozycji (0, 0), co oznacza pierwszy znak pierwszej linii ekranu.

- **lcd.print("R=");**

Wyświetla na ekranie LCD tekst "R=".

- **lcd.print(RED_VALUE);**

Wyświetla na ekranie LCD wartość czerwonego koloru (**RED_VALUE**), która została wcześniej odczytana z czujnika.

- **lcd.print("G=");**

Wyświetla na ekranie LCD tekst "G=".

- **lcd.print(GREEN_VALUE);**

Wyświetla na ekranie LCD wartość zielonego koloru (**GREEN_VALUE**), odczytaną z czujnika.

- **lcd.print("B=");**

Wyświetla na ekranie LCD tekst "B=".

- **lcd.print(BLUE_VALUE);**

Wyświetla na ekranie LCD wartość niebieskiego koloru (**BLUE_VALUE**), odczytaną z czujnika.

- **lcd.setCursor(0, 1);**

Przesuwa kursor na ekranie LCD do pozycji (0, 1), co oznacza pierwszy znak drugiej linii ekranu.

- **lcd.print(COLOR_NAME);**

Wyświetla na ekranie LCD nazwę zidentyfikowanego koloru (**COLOR_NAME**), która została wcześniej przypisana w warunkach if-else.

- **delay(300);**

Powoduje opóźnienie na ekranie LCD przez 300 milisekund, co pozwala użytkownikowi zobaczyć informacje na ekranie przez pewien czas.

- **lcd.clear();**

Czyści ekran LCD, przygotowując go do wyświetlenia kolejnej porcji danych. To jest używane, aby nowe dane nie nakładały się na poprzednie.

6. Pomiary testowe

6.1 Warunki bezpiecznego użytkowania urządzenia

Aby uniknąć uszczerbku na zdrowiu, w tym poparzenia, porażenia prądem, należy przestrzegać poniższych zaleceń.

1. Temperatura:
   - Unikaj ekstremalnych warunków temperaturowych, które mogą wpływać na dokładność pomiarów.
2. Ochrona przed wilgocią:
   - Upewnij się, że detektor jest chroniony przed wilgocią, zwłaszcza jeśli jest stosowany w środowisku o wysokiej wilgotności.
3. Czystość:
   - Zachowaj czystość detektora, aby uniknąć zakłóceń wynikających z zanieczyszczeń na powierzchni detektora.
4. Bezpieczeństwo elektryczne:
   - Sprawdź, czy detektor jest bezpieczny elektrycznie i czy spełnia wszystkie standardy bezpieczeństwa elektrycznego.
5. Transport i przechowywanie:
   - Podczas transportu i przechowywania, dbaj o zabezpieczenie detektora przed wstrząsami i uszkodzeniami mechanicznymi.

6.2 Testy urządzenia

Korzystając z urządzenia zmierzono wartości składowych RGB dla kilku podstawowych kolorów. Wyniki pomiarów opisuje poniższa tabela.

Tabela 2 Kolory uzyskane przy pomocy detektora.

Kolor	R	G	B	Kolor Uzyskany	Kolor Rzeczywisty	R	G	B
Czarny	0	0	0			0	0	0
Niebieski	0	162	198			87	172	247
Brązowy	73	19	0			88	63	38
Zielony	23	154	89			52	137	69
Jasny Niebieski	93	198	210			103	196	215
Jasny Zielony	155	191	133			173	218	83
Pomarańczowy	180	81	31			245	138	67
Różowy	191	143	186			245	155	197
Czerwony	152	0	0			198	56	41
Biały	197	212	218			255	255	255
Żółty	198	192	114			230	208	6

Na podstawie otrzymanych wyników można stwierdzić, że wartości otrzymane przez czujnik są zbliżone do kolorów oryginalnych, jednak są one przyciemnione.

Na zdjęciach przedstawione zostały odczyty kolorów, które zostały uzyskane za pomocą naszego urządzenia na kartkach o różnych kolorach. Z rysunku 43 do 53 przedstawiono kolejno odczyty dla koloru czarnego, niebieskiego, brązowego, zielonego, jasnoniebieskiego, jasnozielonego, pomarańczowego, różowego, czerwonego, białego i żółtego. Należy zwrócić szczególną uwagę na kolor diody, który odzwierciedla kolor kartki

Rysunek 43 Odczyt dla koloru czarnego.

Rysunek 44 Odczyt dla koloru niebieskiego.

Rysunek 45 Odczyt dla koloru brązowego.

Rysunek 46 Odczyt dla koloru zielonego.

Rysunek 47 Odczyt dla koloru jasnoniebieskiego

Rysunek 48 Odczyt dla koloru jasnozielonego

Rysunek 49 Odczyt dla koloru pomarańczowego

Rysunek 50 Odczyt dla koloru różowego

Rysunek 51 Odczyt dla koloru czerwonego.

Rysunek 52 Odczyt dla koloru białego.

Rysunek 53 Odczyt dla koloru żółtego.

Specyfikacja urządzenia

Przy założeniu, że układ pobiera 90 mA, a bateria ma pojemność 500 mAh przy napięciu 9V, możemy szacować, że układ powinien działać ciągle przez około 5,56 godziny. Ta kalkulacja wynika z podziału pojemności baterii przez pobór prądu:

$$\frac{500mAh}{90mA} \approx 5{,}56\ h.$$

Warto jednak pamiętać, że są to teoretyczne obliczenia, a rzeczywisty czas działania może się różnić w zależności od wielu czynników, takich jak efektywność konwersji energii, temperatura otoczenia czy stan baterii.

Tabela 3 Parametry znamionowe urządzenia.

Parametr	Symbol	Min.	Typ.	Max.	Jednostka
Napięcie zasilania	Ucc	5	5	12	V
Pobór prądu	Icc	60	85	90	mA
Długość	X	-	138	-	mm
Szerokość	Y	-	155	-	mm
Wysokość	Z	-	58	-	mm
Wartość składowej czerwonej	R	0	-	255	-
Wartość składowej zielonej	G	0	-	255	-
Wartość składowej niebieskiej	B	0	-	255	-

7. Instrukcja Obsługi Detektora Składowych RGB:

Uruchomienie Urządzenia

Aby uruchomić detektor składowych RGB, wykonaj następujące kroki:

Ustawienie Klucza: Zlokalizuj przełącznik urządzenia i ustaw go w pozycję "ON". Ten klucz odpowiedzialny jest za włączenie detektora.

Badanie Koloru: Umieść detektor na badanym kolorze. Detektor składowych RGB automatycznie zbada i zarejestruje składowe koloru, prezentując wyniki na wyświetlaczu LCD.

Wyłączenie Urządzenia

Aby bezpiecznie wyłączyć detektor składowych RGB, postępuj zgodnie z poniższymi wskazówkami:

Ustawienie Klucza: Znajdź przełącznik urządzenia i przenieś go w pozycję "OFF". Ten krok zabezpiecza detektor przed dalszym zużyciem baterii i wyłącza urządzenie.

Rysunek 1 Zdjęcie przodu urządzenia

8. Podsumowanie

W trakcie realizacji projektu zdobyliśmy liczne wartościowe umiejętności oraz poszerzyliśmy naszą wiedzę w obszarze optoelektroniki, co było głównym celem tego semestralnego przedsięwzięcia. Wszystkie założenia projektowe zostały pomyślnie zrealizowane, a prace nad nim przebiegały sprawnie i systematycznie przez cały okres semestru.

Spełnione założenia obejmują detektor składowych RGB, wykorzystanie wyświetlacza LCD do czytelnej prezentacji wyników, zasilanie bateryjne dla mobilności i niezależności energetycznej oraz diody RGB emitującej światło zbliżone do koloru analizowanego, zapewniającej precyzyjne pomiary. Dodatkowo, projekt wkomponowuje ergonomiczny design, dostosowany do komfortowego użytkowania.

W trakcie realizacji projektu napotkaliśmy kilka istotnych wyzwań, z którymi musieliśmy się skonfrontować, mając na uwadze skuteczność detektora składowych RGB. Jednym z głównych problemów był proces dostosowywania diody RGB do naszych oczekiwań. Okazało się, że uzyskanie pożądanej intensywności i barwy światła w trzech składowych kolorów wymagało precyzyjnego dostrajania parametrów, co stanowiło wyzwanie techniczne.

Kolejnym istotnym zagadnieniem było dostosowanie czujnika koloru do otoczenia. Umieszczenie czujnika bezpośrednio pod obudową detektora miało na celu zminimalizowanie wpływu zewnętrznych czynników. Dodatkowym problemem okazało się określenie optymalnej odległości czujnika od badanego elementu. Diody oświetlające, jeśli były zbyt blisko, wprowadzały znaczące błędy, natomiast zbyt duże odległości sprawiały, że pomiary stawały się nieskuteczne.

Rozwiązanie tych problemów wymagało wielokrotnych eksperymentów, testów i iteracji. Konieczne było zastosowanie różnych kombinacji parametrów, a także dostosowanie układu elektronicznego i mechanicznego detektora.

W kontekście możliwych modyfikacji, rozważamy potencjalne ulepszenia. Mogą to być dodanie łączności bezprzewodowej, takiej jak Bluetooth lub Wi-Fi, umożliwiającej zdalne monitorowanie i kontrolowanie detektora za pomocą smartfona lub komputera. Rozbudowa interfejsu użytkownika na wyświetlaczu pozwoliłaby na dostęp do różnych trybów pracy, ustawień czy historii pomiarów. Ponadto, projekt mógłby zostać zabezpieczony przed warunkami atmosferycznymi, co umożliwiłoby bezpieczne użytkowanie w różnych warunkach środowiskowych. Wprowadzenie panelu słonecznego lub innych źródeł energii odnawialnej do ładowania baterii zwiększyłoby niezależność energetyczną detektora.

Bibliografia

*Wiedza:

1. https://www.rapidtables.com/web/color/RGB_Color.html
2.https://botland.com.pl/czujniki-swiatla-i-koloru/4932-czujnik-koloru-przetwornik-swiatlo-czestotliwosc-tcs3200-waveshare-9520-5904422374211.html
3.https://botland.com.pl/index.php?controller=attachment&id_attachment=185
4.https://www.arduino.cc/reference/en/language/functions/math/map/
5.https://audiodesign.info.pl/diody-led-przewlekane/DIODA-LED-5mm-RGB-WA-.html
6.https://www.piekarz.pl/40247-bateria-energizer-6lr61-energizer-max-plus-9v-blister/
7.https://botland.com.pl/wyswietlacze-alfanumeryczne-i-graficzne/19732-wyswietlacz-lcd-2x16-znakow-niebieski-ze-zlaczami-justpi-5903351243131.html?cd=18298825138&ad=&kd=&gad_source=1&gclid=CjwKCAiAkp6tBhB5EiwANTCx1H-GfvNyHQdA5LCNkZ9w4_wtmfIEy_63p6gFy2m4YMoWI2nRjxxbARoCMSAQAvD_BwE
8.https://efizyka.net.pl/fale-elektromagnetyczne-widmo-fal-elektromagnetycznych
9.https://botland.com.pl/czujniki-swiatla-i-koloru/4932-czujnik-koloru-przetwornik-swiatlo-czestotliwosc-tcs3200-waveshare-9520-5904422374211.html
10.https://forbot.pl/blog/kurs-arduino-silniki-pwm-serwomechanizm-zewnetrzne-biblioteki-id3913
11.https://html.alldatasheet.com/html-pdf/785114/ENERGIZER/6LR61/484/1/6LR61.html
12.https://www.bing.com/ck/a?!&&p=31ddef02894de022JmltdHM9MTcwNTYyMjQwMCZpZ3VpZD0zMTIzMTBlMC1iNjg4LTY0MWItMGY5Zi0wMmM1YjdmMDY1YjcmaW5zaWQ9NTIyNw&ptn=3&ver=2&hsh=3&fclid=312310e0-b688-641b-0f9f-02c5b7f065b7&psq=wikipedia+rgb&u=a1aHR0cHM6Ly9wbC53aWtpcGVkaWEub3JnL3dpa2kvUkdC&ntb=1

*Zdjęcia:

[1] – https://dmfizyka.online/?p=347

[2] –https://www.pikpng.com/pngvi/xhhoio_alt-text-rgb-led-color-mixing-chart-clipart/

[3] –https://botland.com.pl/czujniki-swiatla-i-koloru/4932-czujnik-koloru-przetwornik-swiatlo-czestotliwosc-tcs3200-waveshare-9520-5904422374211.html

[4] – https://www.mcselec.com/index.php?option=com_content&task=view&id=329&Itemid=105

[5] – https://botland.com.pl/czujniki-swiatla-i-koloru/4932-czujnik-koloru-przetwornik-swiatlo-czestotliwosc-tcs3200-waveshare-9520-5904422374211.html

[6] –https://botland.com.pl/czujniki-swiatla-i-koloru/4932-czujnik-koloru-przetwornik-swiatlo-czestotliwosc-tcs3200-waveshare-9520-5904422374211.html

[7] –https://botland.com.pl/czujniki-swiatla-i-koloru/4932-czujnik-koloru-przetwornik-swiatlo-czestotliwosc-tcs3200-waveshare-9520-5904422374211.html

[8] –https://botland.com.pl/czujniki-swiatla-i-koloru/4932-czujnik-koloru-przetwornik-swiatlo-czestotliwosc-tcs3200-waveshare-9520-5904422374211.html

[9] –https://allegro.pl/oferta/arduino-uno-rev3-a000066-oryginal-8203566183

[10] –https://archiwum.allegro.pl/oferta/wyswietlacz-lcd-1602-2x16-hd44780-blue-arduino-i7384561661.html

[11] –https://botland.com.pl/diody-led-rgb/543-dioda-led-5mm-rgb-wsp-anoda-5-szt-5903351244176.html

[12] –https://www.piekarz.pl/40247-bateria-energizer-6lr61-energizer-max-plus-9v-blister/

[13] –https://electronics.stackexchange.com/questions/382584/running-12v-pc-fan-with-9v-battery

[14] –https://www.masterled.pl/wlacznik-kolyskowy-okragly-12v-podswietlany-duplikat-2.html

*Materiały własne:

1.Zdjęcia testów urządzenia Rysunek (43-53)
2.Schemat blokowy - Rysunek 3
3.Schemat ideowy - Rysunek 8
4.Algorytm programu -Rysunek 21
5.Fotografie urządzenia Rysunek (4-7)

Dodatki

Listing kodu projektu:

```
1 /*
2 * Kod źródłowy do projektu "Detektor składowych RGB".
3 * Wersja 19.01.2024
4 * Autorzy: Radosław Mierzwa 263675, Pavlo Kostushevych 251190, Mateusz Gwioździk 263658
5  Opis podłączenia układu:
7  A1 - Arduino UNO rev3
8  U1 - TCS300
9  DS1 - LCD
10 D1 - dioda RGB
11
12  +5V A1_VIN
13  A1_3V3 U1_VDD
14  A1_A0 DS1_E
15  A1_A1 DS1_D5
16  A1_A2 DS1_D6
17  A1_D2 DS1_RS
18  A1_D3 D1_B
```

```
19  A1_D4 DS1_D4
20  A1_D5 D1_G
21  A1_D6 D1_R
22  A1_D7 DS1_D7
23  A1_D8 U1_OUT
24  A1_D9 U1_S0
25  A1_D10 U1_S1
26  A1_D11 U1_S2
27  A1_D12 U1_S3
28  A1_D12 U1_OE
29  A1_GND GND
30  */
31
32  // Deklaracja pinów podłączonych do LCD
33  #include <LiquidCrystal.h>
34  LiquidCrystal lcd(2, A0, 4, A1, A2, 7);
35
36  // Deklaracja pinów podłączonych do czujnika TCS3200
37  #define S0 9
38  #define S1 10
39  #define S2 11
40  #define S3 12
41  #define OUT 8
42  #define LED 13
43
44  // Deklaracja pinów podłączonych do diody RGB
45  #define RED 6
46  #define GREEN 5
47  #define BLUE 3
48
49  // Deklaracja zmiennych użytych w projekcie
50  long int HALF_PERIOD = 0;
51  int RED_VALUE = 0;
52  int GREEN_VALUE = 0;
53  int BLUE_VALUE = 0;
```

```
54  int RED_LED_VALUE = 0;
55  int GREEN_LED_VALUE = 0;
56  int BLUE_LED_VALUE = 0;
57  String COLOR_NAME;
58
59  void setup() {
60
61  // Deklaracja trybu działania pinów podłączanych do czujnika
62  pinMode(S0, OUTPUT);
63  pinMode(S1, OUTPUT);
64  pinMode(S2, OUTPUT);
65  pinMode(S3, OUTPUT);
66  pinMode(LED, OUTPUT);
67  pinMode(OUT, INPUT);
68
69  // Inicjalizacja wyświetlacza oraz podanie jego wymiarów
70  lcd.begin(16, 2);
71
72  // Ustawienie skalowania częstotliwości sygnału wyjściowego z czujnika
na 20%
73  digitalWrite(S0, HIGH);
74  digitalWrite(S1, LOW);
75
76  // Zapalenie diod umieszczonych w module z czujnikiem
77  digitalWrite(LED,HIGH );
78
79  // Rozpoczeńcie przesyłania szeregowego z prędkością bitową 9600 bps
80  Serial.begin(9600);
81  }
82
83  void loop() {
84  // ROZPOCZĘCIE POMIARÓW SKŁADOWEJ CZERWONEJ
85
86  // Ustawienie odczytu na składową czerwoną
87  digitalWrite(S2, LOW);
```

```
88  digitalWrite(S3, LOW);
89
90  // Odczyt połowy okresu sygnału wyjściowego czujnika
91  HALF_PERIOD = pulseIn(OUT, LOW);
92
93  // Odwrócenie otrzymanej wartości
94  RED_VALUE = 255-HALF_PERIOD;
95
96  // Ucięcie wartości poniżej 0
97  if (RED_VALUE < 0) {
98    RED_VALUE = 0;
99  }
100
101  // Wypisanie wartości składowej czerwonej na monitorze szeregowym
102  Serial.print("R = ");
103  Serial.print(RED_VALUE);
104  Serial.print("   ");
105
106  // Odczekanie 100 ms
107  delay(100);
108
109  // ROZPOCZĘCIE POMIARÓW SKŁADOWEJ ZIELONEJ
110
111  // Ustawienie odczytu na składową zieloną
112  digitalWrite(S2, HIGH);
113  digitalWrite(S3, HIGH);
114
115  // Odczyt połowy okresu sygnału wyjściowego czujnika
116  HALF_PERIOD = pulseIn(OUT, LOW);
117
118  // Odwrócenie otrzymanej wartości
119  GREEN_VALUE = 255-HALF_PERIOD;
120
121  // Ucięcie wartości poniżej 0
122  if (GREEN_VALUE < 0) {
```

```
123    GREEN_VALUE = 0;
124  }
125
126  // Wypisanie wartości składowej zielonej na monitorze szeregowym
127  Serial.print("G = ");
128  Serial.print(GREEN_VALUE);
129  Serial.print("   ");
130
131  // Odczekanie 100 ms
132  delay(100);
133
134  // ROZPOCZĘCIE POMIARÓW SKŁADOWEJ NIEBIESKIEJ
135
136  // Ustawienie odczytu na składową niebieską
137  digitalWrite(S2, LOW);
138  digitalWrite(S3, HIGH);
139
140  // Odczyt połowy okresu sygnału wyjściowego czujnika
141  HALF_PERIOD = pulseIn(OUT, LOW);
142
143  // Odwrócenie otrzymanej wartości
144  BLUE_VALUE = 255-HALF_PERIOD;
145
146  // Ucięcie wartości poniżej 0
147  if (BLUE_VALUE < 0) {
148    BLUE_VALUE = 0;
149  }
150
151  // Wypisanie wartości składowej niebieskiej na monitorze szeregowym
152  Serial.print("B = ");
153  Serial.print(BLUE_VALUE);
154  Serial.println("   ");
155
156  // Odczekanie 100 ms
157  delay(100);
```

```
158
159  // Przypisanie zmiennym odpowiedzialnym za kolor świecenia diody RGB
uzyskanych wartości
160  RED_LED_VALUE=RED_VALUE;
161  GREEN_LED_VALUE=GREEN_VALUE;
162  BLUE_LED_VALUE=BLUE_VALUE;
163
164  // Określenie nazwy otrzymanego koloru na podstawie szacunkowych
wartości granicznych oraz zmiana wartości pinów diody RGB dla trudnych do
otrzymania kolorów
165  if(RED_VALUE < 30 && GREEN_VALUE < 30 && BLUE_VALUE < 30)
166  {
167    COLOR_NAME = "Black";
168  }
169  else if(RED_VALUE > 165 && GREEN_VALUE > 190 && BLUE_VALUE > 190)
170  {
171    COLOR_NAME = "White";
172  }
173  else if(RED_VALUE >= 0 && RED_VALUE < 20 && GREEN_VALUE > 130 &&
GREEN_VALUE < 180 && BLUE_VALUE > 180 && BLUE_VALUE < 255)
174  {
175    COLOR_NAME = "Blue";
176  }
177    else if(RED_VALUE > 45 && RED_VALUE < 160 && GREEN_VALUE > 170 &&
GREEN_VALUE < 210 && BLUE_VALUE > 195 && BLUE_VALUE < 225)
178  {
179    COLOR_NAME = "Light Blue";
180  }
181   else if(RED_VALUE > 140 && RED_VALUE < 225 && GREEN_VALUE >= 0 &&
GREEN_VALUE < 20 && BLUE_VALUE >= 0 && BLUE_VALUE < 28)
182  {
183    COLOR_NAME = "Red";
184    BLUE_LED_VALUE=0;
185  }
186  else if(RED_VALUE > 160 && RED_VALUE < 205 && GREEN_VALUE > 60 &&
GREEN_VALUE < 115 && BLUE_VALUE > 25 && BLUE_VALUE < 90)
187  {
```

```
188    COLOR_NAME = "Orange";
189    RED_LED_VALUE=160;
190    GREEN_LED_VALUE=37;
191    BLUE_LED_VALUE=0;
192  }
193  else if(RED_VALUE > 40 && RED_VALUE < 90 && GREEN_VALUE >= 0 &&
GREEN_VALUE < 25 && BLUE_VALUE >= 0 && BLUE_VALUE < 20)
194  {
195    COLOR_NAME = "Brown";
196    RED_LED_VALUE=70;
197    GREEN_LED_VALUE=12;
198    BLUE_LED_VALUE=0;
199  }
200  else if(RED_VALUE > 0 && RED_VALUE < 50 && GREEN_VALUE > 130 &&
GREEN_VALUE < 170 && BLUE_VALUE > 60 && BLUE_VALUE < 115)
201  {
202    COLOR_NAME = "Green";
203    RED_LED_VALUE=0;
204    GREEN_LED_VALUE=75;
205    BLUE_LED_VALUE=0;
206  }
207  else if(RED_VALUE > 120 && RED_VALUE < 170 && GREEN_VALUE > 170 &&
GREEN_VALUE < 210 && BLUE_VALUE > 100 && BLUE_VALUE < 150)
208  {
209    COLOR_NAME = "Light Green";
210    RED_LED_VALUE=54;
211    GREEN_LED_VALUE=130;
212    BLUE_LED_VALUE=18;
213  }
214  else if(RED_VALUE > 175 && RED_VALUE < 205 && GREEN_VALUE > 120 &&
GREEN_VALUE < 165 && BLUE_VALUE > 160 && BLUE_VALUE < 200)
215  {
216    COLOR_NAME = "Pink";
217    RED_LED_VALUE=148;
218    GREEN_LED_VALUE=7;
219    BLUE_LED_VALUE=50;
```

```
220  }
221  else if(RED_VALUE > 180 && RED_VALUE < 220 && GREEN_VALUE > 180 &&
GREEN_VALUE < 220 && BLUE_VALUE > 100 && BLUE_VALUE < 140)
222  {
223    COLOR_NAME = "Yellow";
224    RED_LED_VALUE=66;
225    GREEN_LED_VALUE =55;
226    BLUE_LED_VALUE=0;
227  }
228  else
229  {
230    COLOR_NAME = "Different Color";
231  }
232
233  // Ustawienie wartości pinów podłączonych do diody RGB
234  analogWrite(RED, 255-RED_LED_VALUE);
235  analogWrite(GREEN, 255-GREEN_LED_VALUE);
236  analogWrite(BLUE, 255 - BLUE_LED_VALUE);
237
238  // Wyświetlenie wartości składowych RGB oraz nazwy koloru na
wyświetlaczu
239  lcd.setCursor(0, 0);
240  lcd.print("R=");
241  lcd.print(RED_VALUE);
242  lcd.print("G=");
243  lcd.print(GREEN_VALUE);
244  lcd.print("B=");
245  lcd.print(BLUE_VALUE);
246  lcd.setCursor(0, 1);
247  lcd.print(COLOR_NAME);
248
249  // Odczekanie 300 ms
250  delay(300);
251
252  // Wyczyszczenie wyświetlacza
```

```
253  lcd.clear();
254  }
255
```