UNIWERSYTET ZIELONOGÓRSKI

Wydział Informatyki, Elektrotechniki i Automatyki

PRACA DYPLOMOWA

Kierunek: Informatyka

OPRACOWANIE WYŚWIETLACZA WYKORZYSTUJĄCEGO EFEKT „PERSISTENCE OF VISION”

Filip Chmielewski

Promotor:

dr inż. Mirosław Kozioł

Pracę akceptuję:

……..

(data i podpis promotora)

Zielona Góra, luty 2024

Streszczenie

Praca dotyczy opracowania wyświetlacza wykorzystującego efekt „persistence of vision”. Omówiono w niej teoretyczne podstawy, trudności techniczne i istniejące rozwiązania. Projekt zakłada stworzenie prototypu wyświetlacza oraz odpowiedniego oprogramowania. Implementacja obejmuje budowę części sprzętowej i programowej, z uwzględnieniem testów funkcjonalnych. Wnioski wskazują na możliwości dalszego rozwoju, takie jak zwiększenie częstotliwości odświeżania i eksploracja trójwymiarowych wersji wyświetlaczy.

Słowa kluczowe: praca dyplomowa, wyświetlacz widmowy, persistence of vision, systemy wbudowane

Spis treści

Spis rysunków

1. Wstęp

1.1. Wprowadzenie

Technologia wyświetlaczy osiągnęła niebywały postęp w ostatnich kilku dekadach. Współczesne rozwiązania oferują gęstość pikseli przekraczającą możliwości ludzkiego oka i doskonałe kąty widzenia, a nowe technologie, takie jak OLED czy mini LED, zapewniają niespotykany dotąd kontrast. Zauważalnym trendem jest również rozwój wyświetlaczy micro LED i micro OLED, które ze względu na ogromne zagęszczenie pikseli znajdują zastosowanie w rozwiązaniach wirtualnej i rozszerzonej rzeczywistości. W tej pracy opisano inną technologię wyświetlania, która jest znacznie bardziej niszowa od standardów na rynku, a także proces wykonania urządzenia ją wykorzystującego. Mowa o wyświetlaczu widmowym wykorzystującym efekt bezwładności wzroku („persistence of vision”). Takie wyświetlacze cechują się z reguły niższą jakością obrazu od tradycyjnych rozwiązań, jednak posiadają również szereg unikalnych zalet, takich jak możliwość wyświetlania obrazów z przezroczystym tłem oraz możliwość budowy wolumetrycznych wyświetlaczy trójwymiarowych.

1.2. Cel pracy

Celem pracy jest zaprojektowanie, wykonanie i oprogramowanie modelu wyświetlacza widmowego wykorzystującego efekt „persistence of vision” w oparciu o system mikroprocesorowy bazujący na mikrokontrolerze.

1.3. Zakres pracy

- Rozpoznanie trudności technicznych związanych z realizacją wyświetlacza widmowego.
- Opracowanie założeń dla praktycznej realizacji modelu wyświetlacza widmowego.

- Zaprojektowanie i wykonanie części sprzętowej i programowej funkcjonalnego modelu wyświetlacza widmowego.

1.4. Struktura pracy

1. Wprowadzenie

Praca rozpoczyna się od omówienia postępu technologicznego w dziedzinie wyświetlaczy, a także przedstawiany jest główny cel pracy, którym jest stworzenie projektu i implementacja wyświetlacza widmowego.

2. Podstawy teoretyczne

W tej sekcji wyjaśniany jest efekt "persistence of vision" oraz zasada działania wyświetlaczy widmowych, aby czytelnik mógł zrozumieć podstawy teoretyczne, na których opiera się niniejszy projekt.

3. Analiza techniczna

Analizowane są tutaj trudności techniczne związane z wyświetlaczami widmowymi oraz istniejące rozwiązania, takie jak Voxon VX1, co umożliwia zrozumienie kontekstu niniejszej pracy.

4. Projektowanie

W tej sekcji szczegółowo opisywane są założenia projektowe, poszczególne komponenty sprzętowe i programowe, oraz plan realizacji projektu.

5. Implementacja

W tej sekcji opisywany jest proces realizacji projektu, włączając w to budowę części sprzętowej oraz pisanie oprogramowania, a także ewentualne problemy i sposoby ich rozwiązania.

6. Ocena funkcjonalności wyświetlacza widmowego

Przedstawiane są wyniki testów prototypu i osiągnięte rezultaty.

7. Podsumowanie

Praca kończy się podsumowaniem wniosków wynikających z przeprowadzonych prac, przedstawieniem możliwości dalszego rozwoju projektu oraz sugestiami dotyczącymi potencjalnych usprawnień i zastosowań wyświetlacza widmowego.

2. Podstawy teoretyczne

2.1. Efekt „persistence of vision”

Efekt „persistence of vision”, znany również jako „trwałość wzroku”, to zjawisko, które dotyczy percepcji ludzkiego oka. Zmysł wzroku u człowieka ma pewną inercję, która powoduje, że widziany obraz pozostaje widoczny przez krótki czas po tym, jak źródło tego obrazu zostało usunięte. Innymi słowy, percepcja obrazu nie znika natychmiast po zniknięciu bodźca, który go wywołał.

Dzięki temu właśnie efektowi nasze oczy potrafią przetwarzać szybkie sekwencje obrazów jako płynny ruch. Przykładem tego jest oglądanie filmów. Kiedy na ekranie wyświetlane są kolejne klatki filmowe, nasze oczy utrzymują obraz poprzedniej klatki przez krótki czas, co sprawia, że wydaje się nam, że obraz się porusza. W rzeczywistości jest to iluzja ruchu, stworzona poprzez szybkie wyświetlanie kolejnych obrazów.

Efekt „persistence of vision” jest wykorzystywany w różnych dziedzinach, nie tylko w kinematografii. Na przykład, jest wykorzystywany w animacji, telewizji, grach wideo i wielu innych formach wizualnej komunikacji. Zrozumienie tego zjawiska pozwala projektantom i twórcom na wykorzystanie go w celu stworzenia płynnych i realistycznych doznań wizualnych dla widza.

2.2. Wyświetlacze widmowe

Wyświetlacze widmowe wykorzystują efekt „persistence of vision”, aby wyświetlać obraz poprzez wyświetlanie niewielkiego jego fragmentu w danym momencie. Jeśli wszystkie fragmenty obrazu zostaną wyświetlone w wystarczająco krótkim czasie (poniżej około 100 milisekund [1]), to ludzkie oko dzięki efektowi „persistence of vision” zarejestruje je jako jeden spójny obraz. Przykładem mogą być urządzenia, które poruszając diodami LED z dużą częstotliwością i odpowiednio nimi sterując

tworzą iluzję stabilnego obrazu [2]. Istnieją też wyświetlacze wykorzystujące ten efekt, które nie mają ruchomych części. Są to na przykład multipleksowane macierze LED, w których w danym momencie aktywny jest tylko jeden rząd diod, ale dzięki szybkiemu przełączaniu i efektowi „persistence of vision” użytkownik ma wrażenie stabilnego obrazu [3].

3. Analiza techniczna

3.1. Trudności techniczne związane z wyświetlaczem widmowym

Opracowanie urządzenia zdolnego wyświetlać obraz wykorzystując efekt „persistence of vision” i ruchomą linijkę diod LED wiąże się z wieloma wyzwaniami technicznymi. Po pierwsze, taki wyświetlacz wymaga częstotliwości odświeżania o wiele większej, niż rzeczywista częstotliwość odświeżania całego wyświetlanego obrazu. Dla przykładu, załóżmy, że chcemy wyświetlić 30 klatek całego obrazu na sekundę, gdzie każda klatka składa się ze 100 fragmentów, które są wyświetlane jeden po drugim, a efekt „persistence of vision” sprawia, że dla oka widoczne są one jako jeden obraz. Aby to osiągnąć, potrzebujemy częstotliwości odświeżania o wartości co najmniej 3000 Hz w elemencie wyświetlającym fragmenty obrazu.

Rozwiązać należy również problem synchronizacji wyświetlania fragmentów obrazu z ruchem wyświetlacza. W tym celu należy wyświetlać odpowiednie fragmenty obrazu precyzyjnie w momencie, kiedy ruchomy element wyświetlający znajduje się w odpowiednim miejscu. Brak prawidłowej synchronizacji może skutkować pogorszeniem jakości wyświetlanego obrazu i artefaktami wizualnymi.

Kolejnym wyzwaniem jest konstrukcja mechanicznej części wyświetlacza. Urządzenia tego typu często wykorzystują ruchomy element wyświetlający, który poprzez szybki ruch i wyświetlanie odpowiednich pikseli w odpowiednim czasie sprawia wrażanie posiadania dodatkowego wymiaru przestrzennego. Dla przykładu, obracający się wyświetlacz jednowymiarowy stanie się wyświetlaczem dwuwymiarowym, a dwuwymiarowy - trójwymiarowym. W przypadku umieszczenia elektroniki na ruchomej części należy także rozwiązać problem zasilania i przesyłania danych między ruchomą częścią a resztą systemu.

3.2. Istniejące rozwiązania techniczne

Na rynku dostępnych jest wiele wyświetlaczy widmowych. Podzielić je można na dwie kategorie - dwuwymiarowe i trójwymiarowe.

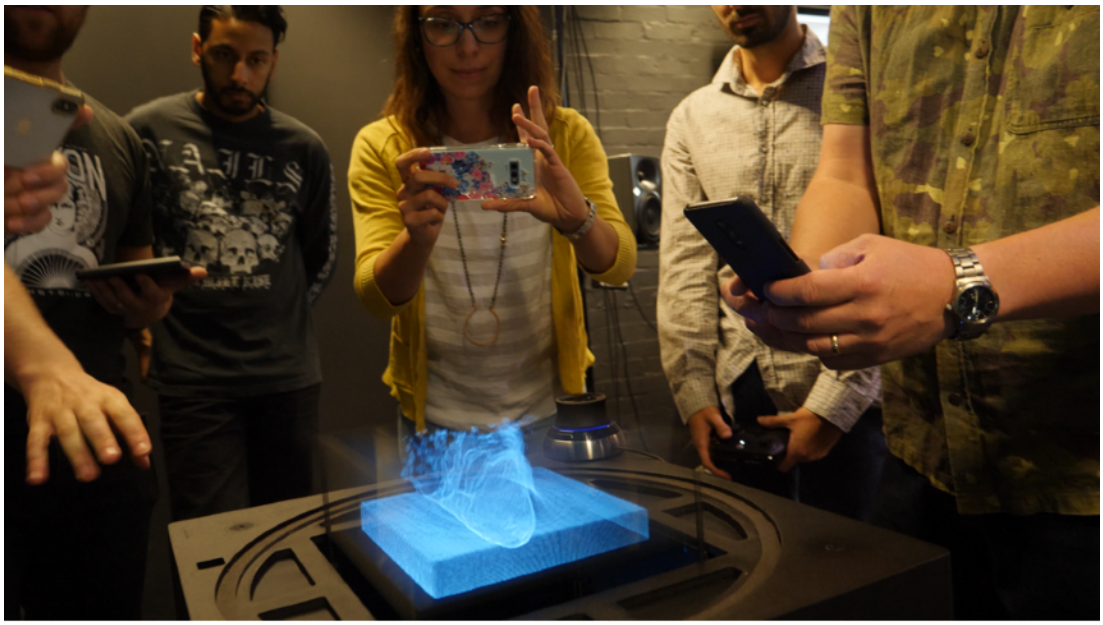
Wyświetlacze dwuwymiarowe najczęściej wykorzystują jedną lub więcej linijek diod LED znajdujących się na ruchomym elemencie. Przykładowe komercyjne urządzenia tego typu produkowane są przez firmę HYPERVSN. Wykorzystują one konstrukcję na kształt wiatraka, na którym umieszczone są diody [4]. Kolejnym ciekawym, komercyjnym zastosowaniem wyświetlaczy widmowych jest system reklamowy zainstalowany w metrze w Pradze. W tym przypadku same diody LED używane do wyświetlania obrazu znajdujące się na ścianie tunelu są nieruchome, ale obserwowane z perspektywy pasażerów jadącego pociągu pozwalają na wyświetlanie animacji [5].

Rys. 3.1. System reklamowy LED w metrze [5]

Przykładem jednego z niekomercyjnych rozwiązań jest ten projekt [6], w którym diody zostały umieszczone na obracającym się łuku, dzięki czemu wyświetlany obraz ma kształt kuli.

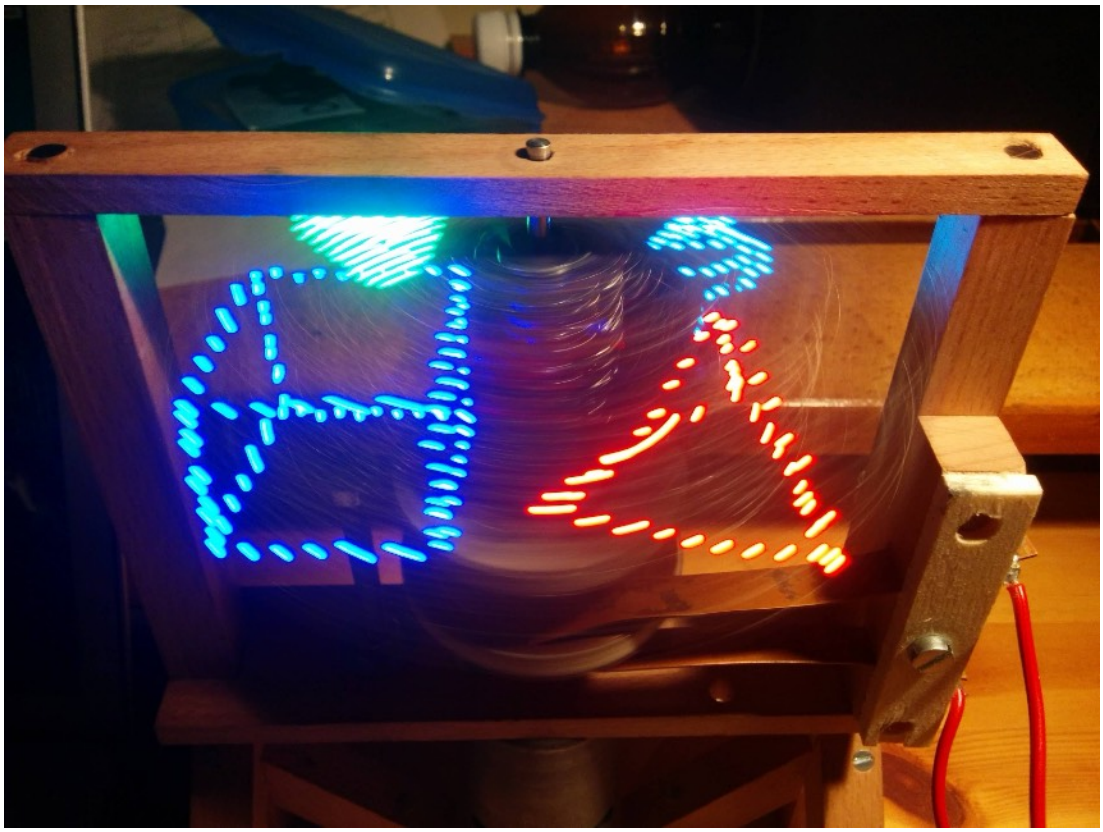
Rys. 3.2. Wyświetlacz widmowy w kształcie kuli [6]

Trójwymiarowe wyświetlacze natomiast wyświetlają trójwymiarowy obraz poprzez cykliczne wyświetlanie jego dwuwymiarowych fragmentów. Przykładem komercyjnej implementacji tej idei jest wyświetlacz Voxon VX1 stworzony przez firmę Voxon Photonics. Jest to wyświetlacz widmowy, który wykorzystuje projektor rzucający obraz na poruszający się ekran, co dzięki precyzyjnej synchronizacji ruchu ekranu z obrazem wyświetlanym przez projektor pozwala na wyświetlanie trójwymiarowych obrazów. W danym momencie wyświetlana jest tylko jedna, dwuwymiarowa warstwa obrazu, ale dzięki efektowi „persistence of vision" widoczny jest obraz trójwymiarowy, będący połączeniem wszystkich dwuwymiarowych warstw wyświetlonych podczas danego cyklu [7].



Rys. 3.3. Widmo obrazu 3D prezentowanego przez wyświetlacz Voxon VX1 [8]

Przykładem niekomercyjnego rozwiązania tego typu jest ten projekt [9], który wykorzystuje diody LED ułożone w spiralę, które podczas obrotu wyświetlają trójwymiarowy obraz.



Rys. 3.4. Widmo obrazu 3D prezentowanego przez wyświetlacz 3D LED [9]

Synchronizacja sterowania diodami z ruchem w wyświetlaczach widmowych z ruchomym elementem wyświetlającym może być realizowana na różne sposoby, np. przy pomocy czujnika Halla i magnesu lub czujnika podczerwieni i diody IR. Oba te czujniki mają to samo zadanie - poinformować układ sterujący o tym, że ruchomy element znalazł się w wyznaczonej pozycji. Dzięki temu możliwe jest obliczenie szybkości ruchu wyświetlacza i oszacowanie pozycji, w jakiej się znajduje w danym momencie.

W kwestii zasilania również istnieją różne rozwiązania. Do tego celu można wykorzystać slip ring, jak w [9]. Wadą takiego rozwiązania jest zwiększone tarcie i dodatkowy element mechaniczny, który podatny jest na zużycie. Inną opcją jest umieszczenie baterii na ruchomej części, jak w [6], jednak w tym przypadku rośnie masa ruchomego elementu, a w przypadku rozładowania baterii praca urządzenia zostaje przerwana. Kolejnym rozwiązaniem problemu zasilania ruchomej części jest zastosowanie indukcyjnego zasilania bezprzewodowego. Pozwala ono uniknąć stosowania części mechanicznych zwiększających tarcie czy baterii zwiększających masę części ruchomej.

Kwestia przesyłania danych do wyświetlenia do diod LED umieszczonych na ruchomej części może zostać rozwiązana poprzez umieszczenie układu sterującego na ruchomej części. W przypadku przesyłania danych z i do urządzenia podczas jego pracy można wykorzystać układ sterujący wyposażony w interfejs bezprzewodowy, taki jak Wi-Fi czy Bluetooth.

3.3. Potencjalne zastosowania

Wyświetlacze widmowe potrafią przyciągnąć uwagę, ponieważ wiele osób nigdy nie widziało tego typu urządzenia, a efekt przezroczystego tła sprawia wrażenie, że obraz nie jest wyświetlany na ekranie, lecz unosi się w powietrzu. Ten aspekt sprawia, że wyświetlacze widmowe mają duży potencjał w przemyśle reklamowym.

Kolejnym potencjalnym zastosowaniem wyświetlaczy widmowych, a zwłaszcza ich trójwymiarowego wariantu jest edukacja. Trójwymiarowy wyświetlacz widmowy może pozwolić na oglądanie modeli 3D tak, jakby były one rzeczywistymi obiektami, co ma szansę pomóc w nauce m.in. biologii czy inżynierii poprzez tworzenie trójwymiarowych wizualizacji.

Takie urządzenia mogą również znaleźć zastosowanie w komunikacji, umożliwiając wideorozmowy z wykorzystaniem trójwymiarowych awatarów wyświetlanych w rzeczywistej przestrzeni, co może być wykorzystane do zapewnienia niespotykanego wcześniej poziomu interakcji.

Trójwymiarowe, wolumetryczne wyświetlacze widmowe mogą także być wykorzystane w celach rozrywkowych, umieszczając wirtualne światy gier komputerowych w rzeczywistej przestrzeni. Przy odpowiedniej wierności obrazu i kreatywności twórców gier, tego typu urządzenia mogłyby zapewnić unikalne doznania z rozgrywki, które byłyby niemożliwe do osiągnięcia na konwencjonalnym wyświetlaczu.

Obecnie wyświetlacze widmowe są uznawane za urządzenia niszowe, ale możliwe, że dzięki rozwojowi technologii w przyszłości podobne konstrukcje będą w stanie umożliwić wyświetlanie prawdziwie trójwymiarowych, nieodróżnialnych od rzeczywistości obrazów, które będzie można oglądać z każdej strony.

4. Projektowanie

4.1. Założenia projektowe

W ramach realizacji pracy dyplomowej podjęto się zaprojektowania i wykonania dwuwymiarowego wyświetlacza widmowego, który używa 36 diod RGB obracających się po okręgu aby wyświetlać kolorowe obrazy składające się łącznie z 8136 pikseli (36 diod odświeżanych 226 razy podczas każdego obrotu). Aby wykorzystać efekt „persistence of vision", jeden obrót musi trwać co najwyżej 100 milisekund, gdzie szybsze obroty oznaczają mniej migoczący i bardziej stabilny obraz.

4.2. Projekt części sprzętowej

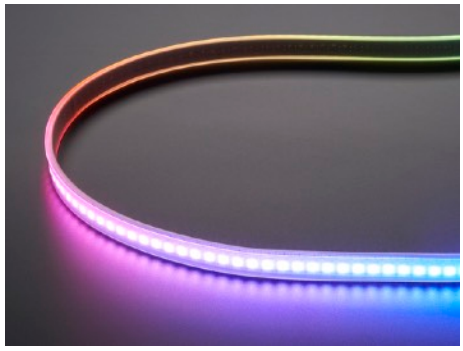
Urządzenie składa się z dwóch zasadniczych części. Pierwszą częścią jest część nieruchoma, na którą składa się bezszczotkowy silnik elektryczny Redox DC 12V 12000RPM (rys. 4.1.) [12], sterownik umożliwiający regulację obrotów silnika (rys. 4.2.) [10], moduł nadawczy zasilania bezprzewodowego, zasilacze DC i magnes. Użyty w projekcie sterownik to bardzo prosty, jednokanałowy sterownik silnika prądu stałego z gałką do regulacji prędkości obrotowej. Regulacja obrotów odbywa się przy pomocy sygnału PWM [10]. Silnik został umieszczony w stojaku, którego podstawa ma wymiary: szerokość 152 mm, wysokość 18 mm i długość 333 mm. Do podstawy przymocowana jest pionowa część, której wymiary to: szerokość 96 mm, wysokość 410 mm i długość 18 mm. W pionowej części został wywiercony otwór o średnicy 36 mm pozwalający na montaż silnika.

Rys. 4.1. Silnik Redox DC 12V 12000RPM [12]

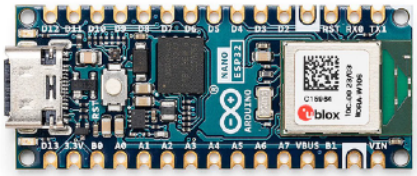
Rys. 4.2. Sterownik silnika [10]

Druga część to wirnik wprawiany w ruch przez silnik. Na tej części znajdują się diody LED (rys. 4.3.) [13], układ sterujący (rys. 4.4.) [14], czujnik Halla i moduł odbiorczy zasilania bezprzewodowego. Wybrane przeze mnie diody DotStar nadają się do zastosowania w wyświetlaczu widmowym, ze względu na wysoką jasność i interfejs SPI pozwalający na szybki przesył danych. Każda dioda ma wbudowany sterownik, który po otrzymaniu danych steruje diodą w odpowiedni sposób, regulując kolor i jasność przy pomocy sygnału PWM o częstotliwości 20 kHz, co pozwala na prawidłowe wyświetlanie kolorów podczas szybkiego ruchu. [11] Jako układ sterujący wybrałem moduł Arduino Nano ESP32. Wybór ten podyktowany został przez kilka czynników:

- mały rozmiar tego modułu ułatwia zamontowanie go na obracającym się elemencie,
- zastosowany mikrokontroler ESP32-S3 zapewnia odpowiednią wydajność dla tego zastosowania,
- 16 MB pamięci flash pozwala na przechowywanie wielu obrazów do wyświetlenia.

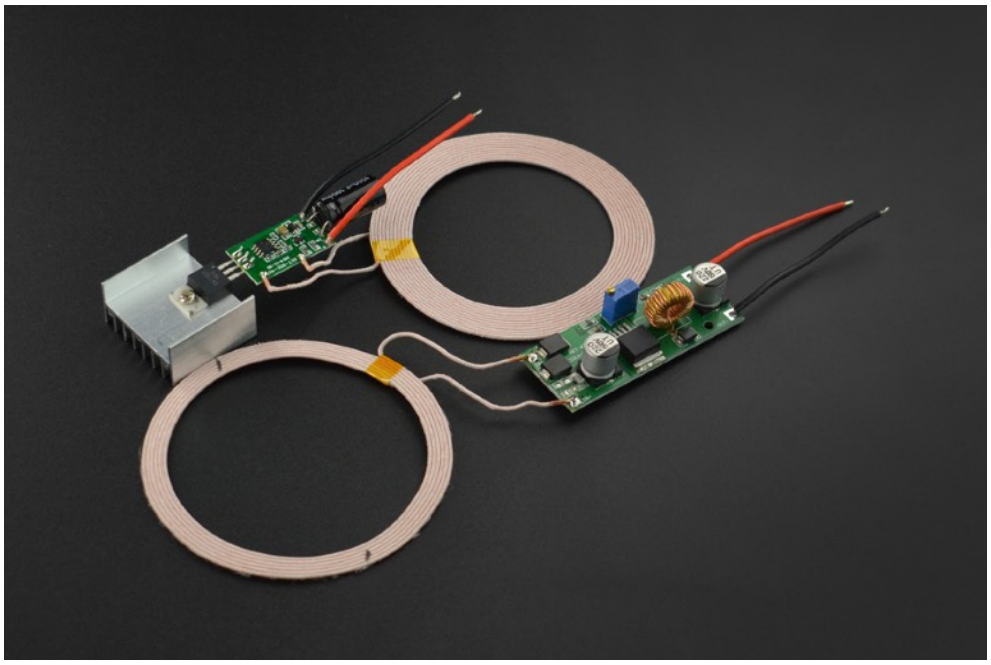


Rys. 4.3. Pasek LED DotStar [13]



Rys. 4.4. Arduino Nano ESP32 [14]

W kwestii zasilania części ruchomej wybrano moduł DFRobot Wireless Charging Module 5V/5A (rys. 4.5.) [16]. Rozważono również inne opcje, takie jak montaż baterii na części ruchomej czy slip ring, ale ostatecznie zasilanie bezprzewodowe wygrało z uwagi na możliwość zmniejszenia masy ruchomej części oraz mniejsze tarcie.

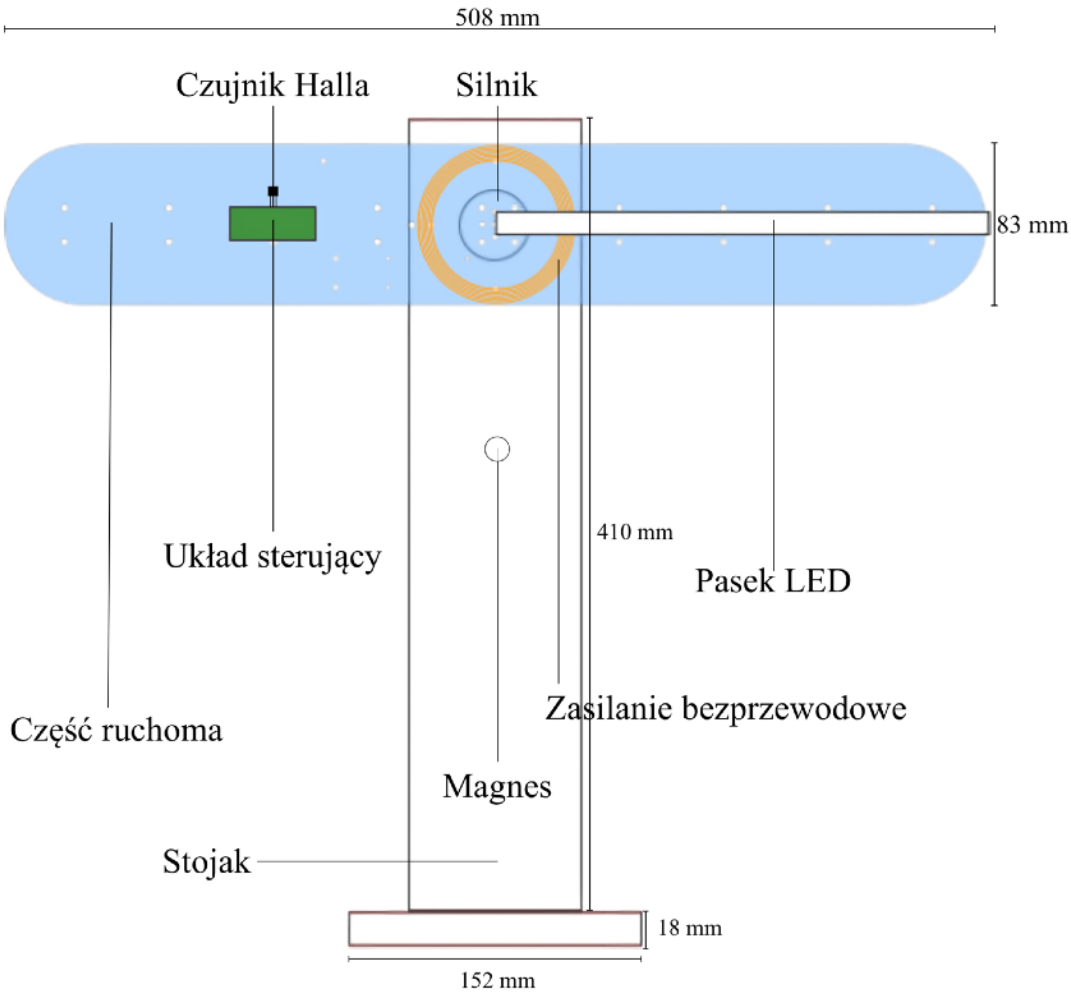


Rys. 4.5. Moduł zasilania bezprzewodowego [16]

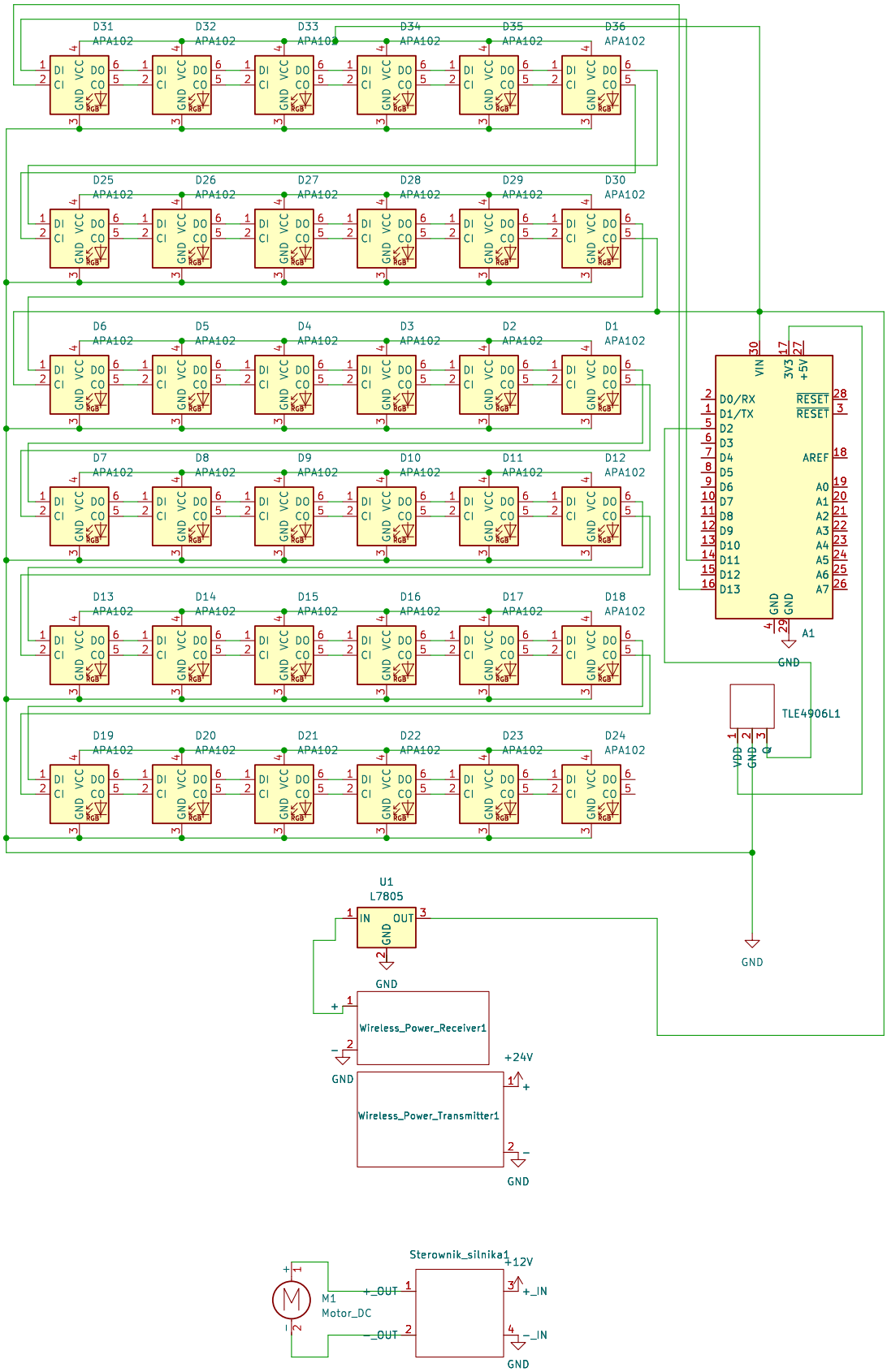
Za synchronizację wyświetlanego obrazu z ruchem wyświetlacza odpowiada czujnik Halla umieszczony na części ruchomej. Przy każdym obrocie wykrywa on magnes umieszczony na części nieruchomej. Wybrany czujnik TLE4906L (przedstawiony na rys. 4.6) jest przeznaczony do zastosowań motoryzacyjnych i przemysłowych wymagających wysokiej precyzji pomiaru [15], co czyni go odpowiednim do zastosowania w tym projekcie.

Rys. 4.6. Czujnik Halla TLE4906L [17]

Na rys. 4.7. przedstawiono projekt sprzętowej części wyświetlacza widmowego, a na rys. 4.8. schemat ideowy urządzenia.



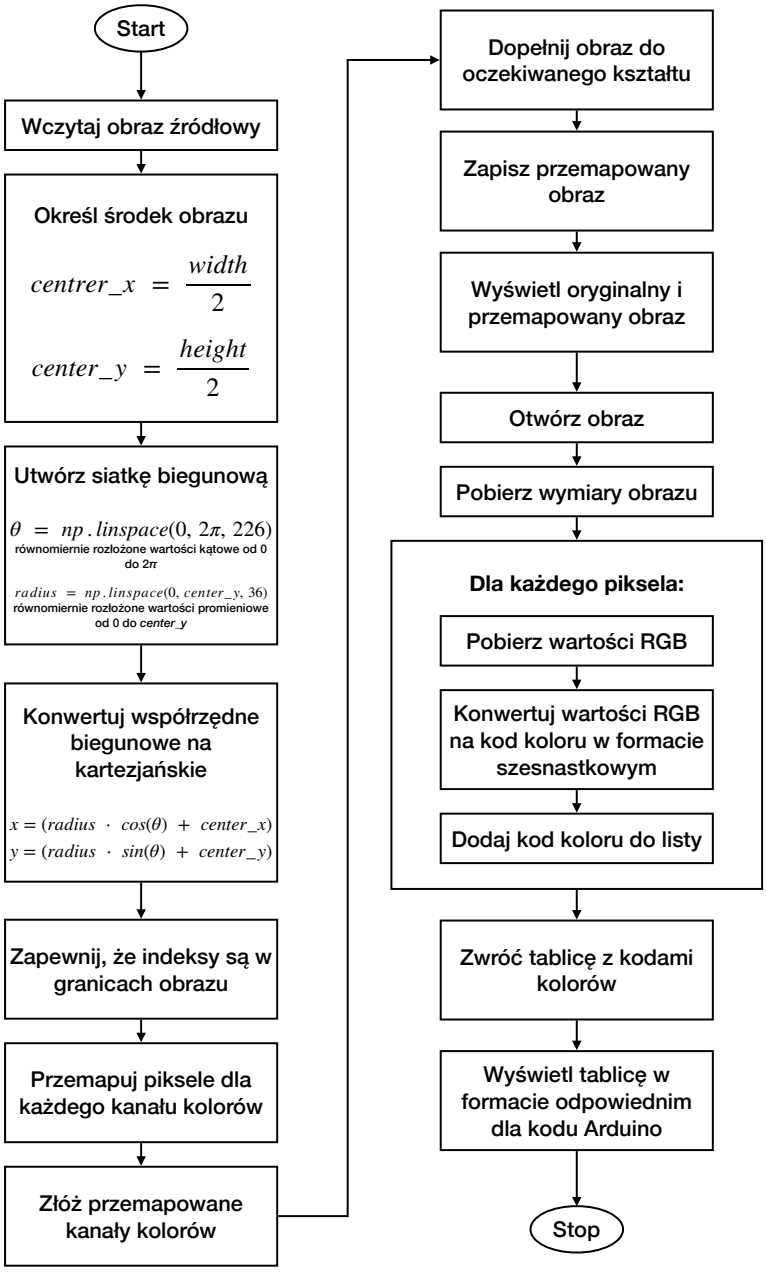
Rys. 4.7. Projekt sprzętowej części wyświetlacza widmowego



Rys. 4.8. Schemat ideowy wyświetlacza widmowego

4.3. Projekt części programowej

Projekt oprogramowania został podzielony na dwie części. Pierwszą częścią jest konwerter - program uruchamiany na komputerze, mający za zadanie konwersję pliku graficznego na dane umożliwiające sterowanie diodami LED w celu wyświetlenia obrazu. Druga część to program uruchamiany na mikrokontrolerze wyświetlacza, odpowiedzialny za wyświetlanie obrazu. Schemat blokowy algorytmu konwertera obrazu został przedstawiony na rys. 4.9.



Rys. 4.9. Schemat blokowy konwertera obrazu

Oprogramowanie mikrokontrolera odpowiada za sterowanie diodami w taki sposób, aby w odpowiednim momencie wyświetlały daną część obrazu. Program wykorzystuje czujnik Halla w celu obliczania prędkości obrotów wyświetlacza, a także w celu obliczenia pozycji, w jakiej znajduje się wyświetlacz w danym momencie. W momencie wykrycia magnesu przez czujnik rozpoczynany jest cykl wyświetlania, podczas którego diody wyświetlają po kolei 226 fragmentów obrazu. Liczba 226 została wybrana ze względu na to, że wyświetlany obraz ma 36 pikseli w promieniu, co daje w przybliżeniu 226 pikseli na obwodzie. Zastosowanie mniejszej liczby pikseli na obwodzie spowodowałoby, że piksele znajdujące się dalej od środka obrazu byłyby rozciągnięte wzdłuż krawędzi okręgu. Czas, przez jaki wyświetlany jest każdy fragment zależny jest od zmierzonej prędkości obrotu.

Konwerter natomiast odczytuje plik graficzny, a następnie zwraca reprezentację tego pliku przeznaczoną do wyświetlenia na wyświetlaczu widmowym. Ponieważ wyświetlany obraz ma kształt koła, konwerter wycina piksele wewnątrz największego okręgu w obrazie wejściowym (rys. 4.10.).

Rys. 4.10. Zaznaczona część obrazu wejściowego przeznaczona do wyświetlenia

5. Implementacja

5.1. Wykonanie części sprzętowej

Budowę części sprzętowej rozpoczęto od zamówienia elementu, na którym umieszczone zostały wszystkie elementy wchodzące w skład ruchomej części urządzenia. Element ten to wycięty według projektu wirnik ze szkła akrylowego. W projekcie uwzględnione zostały otwory ułatwiające montaż elektroniki na wirniku, a także przymocowanie wirnika do wału silnika (rys. 5.1.).

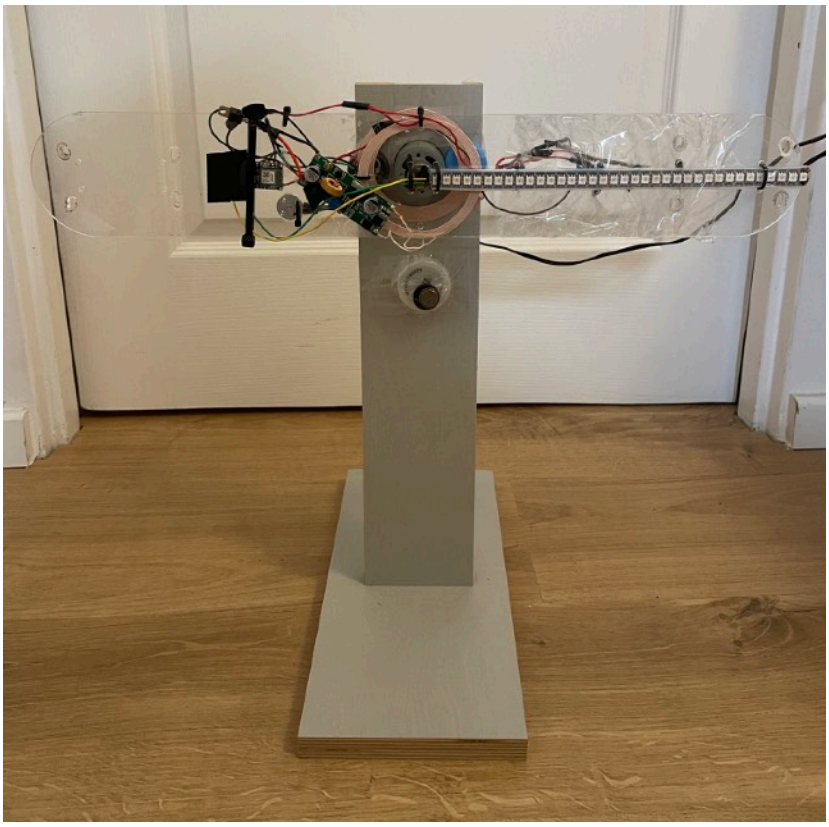
Rys. 5.1. Projekt ruchomej części ze szkła akrylowego

Podczas wstępnych testów modułu zasilania bezprzewodowego napotkano na problem w postaci niestabilnego napięcia na wyjściu modułu, co uniemożliwiło wykorzystanie go w celu zasilenia elektroniki. W związku z tym postawiono na tymczasowe rozwiązanie w postaci zasilania elektroniki z baterii umieszczonych na ruchomej części. Ograniczenie napięcia przy pomocy stabilizatora liniowego L7805CV pozwoliło na wykorzystanie modułu zasilania bezprzewodowego i zrezygnowanie z baterii.

Dzięki zastosowaniu gotowego paska LED Adafruit DotStar udało się ograniczyć potrzebę lutowania. Konieczne było jedynie przylutowanie modułu zasilania, stabilizatora liniowego i linii sterujących diodami (zegar i dane). Na ruchomej części umieszczono również element mocujący ją do wału silnika oraz czujnik Halla.

Na nieruchomą część urządzenia składa się stojak, który wykonany został na zamówienie w lokalnym zakładzie stolarskim, a także silnik, sterownik silnika i moduł nadawczy zasilania bezprzewodowego. Silnik i moduł zasilania

bezprzewodowego zasilane są przy pomocy zasilaczy DC o napięciu odpowiednio 12 V i 24 V. Gotowy prototyp wyświetlacza widmowego przedstawiono na rysunku 5.2.

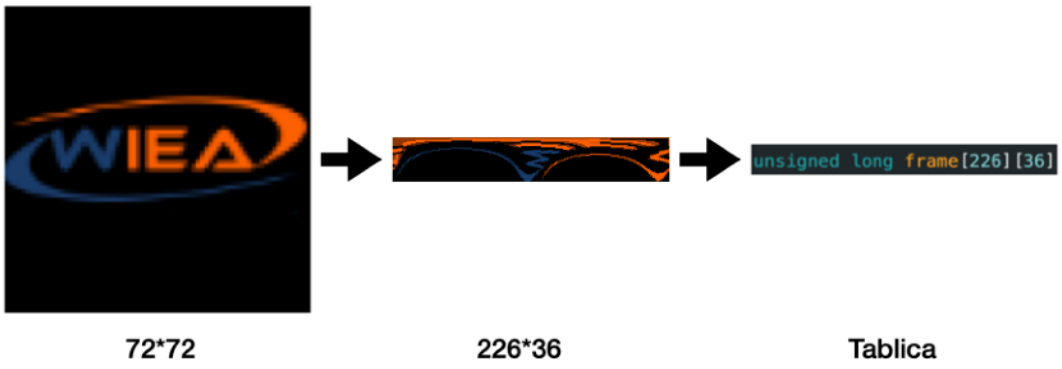


Rys. 5.2. Gotowy prototyp wyświetlacza widmowego

5.2. Wykonanie części programowej

Część programowa składa się z dwóch części - konwertera obrazu i oprogramowania mikrokontrolera. Konwerter obrazu to skrypt napisany w języku programowania Python, który ma za zadanie przygotowanie pliku graficznego do wyświetlenia na wyświetlaczu widmowym. Kod skryptu przedstawiono w dodatku A. Działa on w dwóch etapach: w pierwszym wykonywane jest mapowanie pikseli znajdujących się wewnątrz środkowego okręgu na obrazie wejściowym o wymiarach 72x72 piksele (ta wartość wynika z faktu, że wycięte koło ma w promieniu 36 pikseli, czyli tyle, ile diod ma wyświetlacz) na obraz o wymiarach 226x36 pikseli (36 diod wyświetlających 226 fragmentów obrazu). Następnie otrzymany obraz jest zamieniany na tablicę wartości RGB pikseli, która może zostać umieszczona w

pamięci układu sterującego wyświetlacza widmowego poprzez umieszczenie tablicy w oprogramowaniu mikrokontrolera, a następnie przesłanie go do układu sterującego za pomocą przewodu USB. Działanie konwertera zostało przedstawione na rys. 5.3.



Rys. 5.3. Etapy konwersji obrazu

Oprogramowanie mikrokontrolera natomiast zostało wykonane w środowisku Arduino IDE przy użyciu języka programowania Arduino, będącego wariantem języka C++ i przedstawione w dodatku B. Sterowanie diodami odbywa się za pośrednictwem biblioteki *Adafruit_DotStar*. Główną funkcją odpowiedzialną za wyświetlanie jest funkcja *displayLoop()*. Jest ona uruchamiana w głównej pętli programu. Sprawdza ona aktualny czas przy pomocy funkcji *micros()* (pomiar jest wykonywany w mikrosekundach z uwagi na wymaganą wysoką precyzję sterowania diodami).

Zmierzony czas porównywany jest ze zmienną *nextSliceTime*, która przechowuje czas, w którym wyświetlony ma zostać następny fragment obrazu. Jeśli zmienna jest mniejsza lub równa zmierzonemu czasowi, do diod wysyłany jest następny fragment obrazu, a zmienna *nextSliceTime* jest inkrementowana o czas trwania wyświetlania

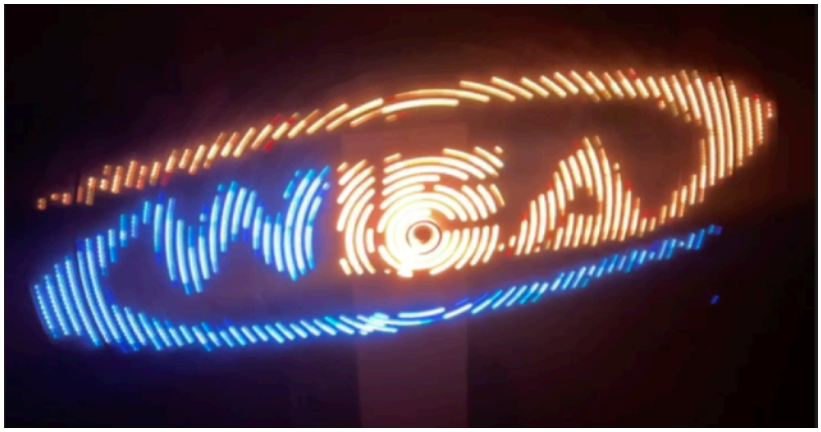
jednego fragmentu obrazu obliczanego według wzoru $\frac{revolutionTime}{226}$, gdzie *revolutionTime* to czas obrotu zmierzony przy pomocy czujnika Halla. Inkrementowana jest również zmienna *sliceIndex* przechowująca indeks fragmentu obrazu do wyświetlenia. W przypadku osiągnięcia przez zmienną *sliceIndex* maksymalnej wartości (225) jest ona resetowana do wartości 0. Aktualnie wyświetlany obraz jest przechowywany w postaci zmiennej *frame* będącej wskaźnikiem do tablicy o wymiarach $226^{x}36$ (226 fragmentów, z których każdy składa się z 36 pikseli).

Czas obrotu obliczany jest przy pomocy przerwania wywoływanego przez czujnik Halla. W przypadku wykrycia zbocza opadającego na wyjściu czujnika po raz pierwszy, aktualny czas w mikrosekundach jest zapisywany do zmiennej *startTime*. Gdy zbocze opadające zostanie wykryte drugi raz, czas zapisywany jest do zmiennej *endTime*, od której odejmowana jest wartość zmiennej *startTime*, wynik przypisywany jest do zmiennej *revolutionTime*, a cały cykl się resetuje. Oprócz obliczania czasu obrotu czujnik Halla pełni również funkcję kalibracji orientacji wyświetlacza, co eliminuje problem przechylania obrazu. Zostało to wykonane poprzez ustawianie zmiennej *sliceIndex* na predefiniowaną wartość między 0 a 225, która jest zależna od miejsca umieszczenia magnesu na nieruchomej części urządzenia.

Zastosowanie zmiennej globalnej *frame* będącej wskaźnikiem do aktualnie wyświetlanej klatki obrazu umożliwiło odseparowanie logiki wyświetlania od samej wyświetlanej zawartości, dzięki czemu możliwa jest zmiana wyświetlanej klatki w trakcie jej wyświetlania.

6. Ocena funkcjonalności wyświetlacza widmowego

Wykonany przeze mnie prototyp umożliwia wyświetlanie kolorowych obrazów składających się łącznie z 8136 pikseli (promień 36, obwód 226). Podczas testów udało się wyświetlać obraz z częstotliwością 12 Hz, co spełnia założenie, że klatka obrazu powinna trwać krócej niż 100 milisekund, aby wykorzystać efekt „persistence of vision”. Poprzez zaimplementowanie mechanizmu przeplotu, który z każdym obrotem naprzemiennie wyświetla połowę pikseli udało się zwiększyć częstotliwość odświeżania obrazu do 16 Hz, gdzie ograniczeniem okazały się zbyt niskie obroty silnika. Prawdopodobnie poprzez zwiększenie obrotów możliwe byłoby osiągnięcie odświeżania na poziomie 24 Hz. Podczas testów zaimplementowano kilka obrazów wyświetlanych jeden po drugim, których przełączanie odbywało się poprzez zmianę wartości wskaźnika *frame*. Przykładowy obraz wyświetlany przez urządzenie widoczny jest na rys. 6.1.



Rys. 6.1. Obraz wyświetlany na wyświetlaczu widmowym

7. Podsumowanie

7.1. Wnioski

W ramach pracy zaprojektowany i wykonany został prototyp wyświetlacza widmowego wykorzystującego efekt „persistence of vision”. Urządzenie składa się z dwóch głównych części: ruchomej, zawierającej wirnik z diodami LED, układem sterującym, czujnikiem Halla i modułem odbiorczym zasilania bezprzewodowego, oraz nieruchomej, na której umieszczony jest silnik, sterownik silnika, moduł nadawczy zasilania bezprzewodowego, zasilacze DC i magnes. Poprzez synchronizację wyświetlanego obrazu z obrotami wirnika udało się uzyskać płynne efekty wizualne.

Podczas wykonania tego projektu zaprojektowano i zaimplementowano sprzętową i programową część wyświetlacza widmowego. Przeanalizowano techniczne trudności, które należy brać pod uwagę podczas tworzenia podobnych projektów, a także rozwiązano wiele problemów, które pojawiły się podczas jego wykonania. Jednym z pierwszych napotkanych problemów było zbyt wysokie napięcie na wyjściu modułu odbiorczego zasilania bezprzewodowego. Zostało ono ustabilizowane przy pomocy stabilizatora liniowego. Kolejną napotkaną przeszkodą była zbyt mała częstotliwość odświeżania paska LED, która nie pozwalała na wyświetlanie więcej niż 12 klatek na sekundę całego obrazu, składającego się z 226 fragmentów. To ograniczenie zostało ominięte dzięki implementacji mechanizmu przeplotu, który podczas jednego obrotu wyświetla jedynie 113 z 226 fragmentów obrazu. Pozwala to na zwiększenie częstotliwości odświeżania całego obrazu przy zachowaniu jego czytelności.

Problem ze zbyt małą przepustowością danych między układem sterującym a paskiem LED może być również rozwiązany na inne sposoby. Jednym z nich byłoby zmniejszenie rozdzielczości wyświetlanego obrazu, przez co więcej klatek mogłoby być wyświetlonych w tym samym czasie. Drugą możliwością jest zmiana elektroniki

na rozwiązanie pozwalające na szybszy przesył danych. Możliwe, że zmiana mikrokontrolera na szybszy lub podzielenie paska LED na kilka części sterowanych przez osobne mikrokontrolery pozwoliłoby na osiągnięcie wyższej częstotliwości. Inną opcją byłaby modyfikacja sterowania diodami w taki sposób, aby przesyłać kolory z mniejszą precyzją, niż obecnie stosowane 8 bitów na piksel. Takie rozwiązanie umożliwiłoby zwiększenie częstotliwości odświeżania kosztem dokładności odwzorowania barw.

Po modyfikacji oprogramowania w taki sposób, żeby z każdym obrotem wyświetlana była połowa pikseli udało mi się zwiększyć częstotliwość do 16 Hz. Tym razem głównym ograniczeniem okazała się prędkość obrotowa silnika, który w tym przypadku osiąga maksymalnie 16 obrotów na sekundę. Prawdopodobnie zwiększenie prędkości obrotowej silnika pozwoliłoby na zwiększenie częstotliwości do około 24 Hz przy obecnej konfiguracji sprzętu i oprogramowania na ruchomej części urządzenia.

7.2. Możliwości dalszego rozwoju

Wykonany w ramach pracy prototyp stanowi przykład tego, jak efekt „persistence of vision” może być wykorzystany do wyświetlania obrazu. Dzięki zdobytemu doświadczeniu jestem w stanie wymienić potencjalne usprawnienia, które mogą zostać zaimplementowane w podobnych konstrukcjach.

Po pierwsze, najbardziej oczywistą kwestią jest zwiększenie parametrów takich jak częstotliwość odświeżania i rozdzielczość, co pozwoliłoby uzyskać bardziej wyraźny i płynny obraz. Programowa część projektu również ma potencjał do usprawnień, takich jak wykorzystanie interfejsów bezprzewodowych znajdujących się w układzie sterującym w celu przekazywania obrazu do urządzenia.

Kolejnym etapem ewolucji urządzenia mogłoby być stworzenie wolumetrycznego wyświetlacza trójwymiarowego działającego na tej samej zasadzie. Budowa trójwymiarowych wyświetlaczy widmowych stanowi ciekawy kierunek do eksploracji, jednak wiąże się także z większymi kosztami i wyzwaniami technicznymi.

Dodatek A

Kod źródłowy konwertera obrazu

```
import numpy as np
import cv2
from PIL import Image

def remap_pixels_color(source_image, target_shape):
    # Get the center of the source image
    center_x, center_y = source_image.shape[1] // 2,
source_image.shape[0] // 2

    # Create a polar grid
    theta, radius = np.meshgrid(np.linspace(0, 2*np.pi,
target_shape[1]),
                                np.linspace(0, center_y,
target_shape[0]))

    # Convert polar coordinates to Cartesian coordinates
    x = (radius * np.cos(theta) + center_x).astype(int)
    y = (radius * np.sin(theta) + center_y).astype(int)

    # Ensure indices are within bounds
    x = np.clip(x, 0, source_image.shape[1] - 1)
    y = np.clip(y, 0, source_image.shape[0] - 1)

    # Remap pixels for each color channel
    remapped_channels = [source_image[:,:,i][y, x] for i
in range(source_image.shape[2])]

    # Stack the remapped channels to get the final
remapped image
    remapped_image = np.stack(remapped_channels,
axis=-1)

    # Reshape the remapped image to the target shape
    remapped_image =
remapped_image.reshape(target_shape[0], target_shape[1],
source_image.shape[2])

    return remapped_image

# Load an input image
```

```
source_image = cv2.imread('input.png', cv2.IMREAD_COLOR)

# Set the target shape
target_shape = (36, 226)

# Remap pixels in color
remapped_image_color = remap_pixels_color(source_image,
target_shape)

# Save the remapped color image
output_path = 'remapped_image_color.png'
cv2.imwrite(output_path, remapped_image_color)

print(f"Remapped image saved to {output_path}")


# Display the original and remapped images
cv2.imshow('Original Image', source_image)
cv2.imshow('Remapped Image (Color)',
remapped_image_color)
cv2.waitKey(0)
cv2.destroyAllWindows()


def convert_image_to_array(image_path):
    # Open the image
    img = Image.open(image_path)

    # Get the size of the image
    width, height = img.size

    # Initialize an empty list to store the Arduino
array values
    arduino_array = []

    # Loop through each pixel in the image
    for x in range(width):
        row = []
        for y in range(height):
            # Get the RGB values of the pixel
            r, g, b = img.getpixel((x, y))

            # Convert RGB values to a hexadecimal color
code
            color_code = (r << 16) | (g << 8) | b

            # Append the color code to the row list
            row.append(color_code)

        # Append the row list to the Arduino array
```

```
            arduino_array.append(row)

        return arduino_array

def print_arduino_array(arduino_array):
    # Print the Arduino array
    print("{")
    for row in arduino_array:
        print("  {" + ", ".join(f"0x{color:06X}" for
color in row) + "},")
    print("};")

if __name__ == "__main__":
    image_path = 'remapped_image_color.png'

    # Convert the image to an Arduino array
    arduino_array = convert_image_to_array(image_path)

    # Print the Arduino array
    print_arduino_array(arduino_array)
```

Dodatek B

Kod źródłowy oprogramowania mikrokontrolera

```
#include <Adafruit_DotStar.h>

Adafruit_DotStar strip(36, DOTSTAR_BGR);

const PROGMEM unsigned long slideshow[9][226][36] = {{
  // Paste the RGB values generated by the converter
here.
}}


unsigned long timeToNextSlide;
int slideIndex = 0;

const int hallEffectPin = 2;  // Hall effect sensor
input pin
volatile unsigned long startTime = 0;
volatile unsigned long endTime = 0;
unsigned long revolutionTime = 1883333;

unsigned long nextSliceTime = 0;
int sliceIndex = 0;

const unsigned long (*frame)[226][36];

void setup() {
  pinMode(hallEffectPin, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(hallEffectPin),
handlePulse, FALLING);

  frame = &slideshow[0];
  timeToNextSlide = millis() + 3000;

  strip.begin(); // Initialize pins for output
  strip.setBrightness(30);
  strip.show();  // Turn all LEDs off ASAP
}

void loop() {
```

```
  displayLoop();

  if (endTime != 0) {
    // Calculate time for a single revolution in
microseconds
    revolutionTime = endTime - startTime;

    // Reset variables
    startTime = 0;
    endTime = 0;
  }

  // Change slides in the slideshow
  if (millis() >= timeToNextSlide) {
    timeToNextSlide = millis() + 3000;
    if (slideIndex < 7) {
      slideIndex += 1;
    } else {
      slideIndex = 0;
    }

    frame = &slideshow[slideIndex];
  }
}

void handlePulse() {
  // Display orientation calibration
  sliceIndex = 177;

  if (startTime == 0) {
    // Record start time on the first pulse
    startTime = micros();
  } else {
    // Record end time on the second pulse
    endTime = micros();
  }
}

void displayLoop() {
  unsigned long currentTime = micros();
  if (currentTime >= nextSliceTime) {
    // Implemented interlacing, only half of the pixels
are displayed in a single rotation
      nextSliceTime = currentTime + (revolutionTime /
113);
      for (int i = 0; i < 36; i++) {
        strip.setPixelColor(i, (*frame)[sliceIndex][i]);
      }

      sliceIndex += 2;
```

```
      strip.show();

      if (sliceIndex == 226) {
        sliceIndex = 1;
        nextSliceTime += (revolutionTime / 113);
      } else if (sliceIndex > 226) {
        sliceIndex = 0;
      }
    }
}
```

Bibliografia

[1] J. E. Farrell, „Visible Persistence of Moving Objects”.

[2] A. Patel, A. K. Khandual, K. Kumar, i S. Kumar, „Persistence Of Vision Display-A Review”.

[3] „Introduction to Driving LED Matrices”, Avago Technologies. Dostęp: 8 luty 2024. [Online]. Dostępne na: https://docs.broadcom.com/doc/AV02-3697EN

[4] „HYPERVSN is a 3D Integrated Holographic System for advertising, digital signage, events.” Dostęp: 8 luty 2024. [Online]. Dostępne na: https://hypervsn.com/

[5] „Prague unveils innovative new LED advertising system in metro tunnels”. Dostęp: 8 luty 2024. [Online]. Dostępne na: https://www.expats.cz/czech-news/article/prague-unveils-innovative-new-led-advertising-in-metro-tunnels

[6] GreatScottLab, „Make Your Own POV LED Globe”, Instructables. Dostęp: 8 luty 2024. [Online]. Dostępne na: https://www.instructables.com/Make-Your-Own-POV-LED-Globe/

[7] „Technology”, Voxon Photonics. Dostęp: 7 luty 2024. [Online]. Dostępne na: https://voxon.co/technology/

[8] „Community / Developers”, Voxon Photonics. Dostęp: 8 luty 2024. [Online]. Dostępne na: https://voxon.co/community/

[9] Balduin, „mbjd/3DPOV”. 20 lipiec 2022. Dostęp: 7 luty 2024. [Online]. Dostępne na: https://github.com/mbjd/3DPOV

[10] „Prosty sterownik silnika DC 28V/3A - moduł z gałką”, BOTLAND. Dostęp: 8 luty 2024. [Online]. Dostępne na: https://botland.com.pl/sterowniki-silnikow-dc/6906-prosty-sterownik-silnika-dc-28v3a-modul-z-galka-5904422359898.html

[11] „Adafruit DotStar LEDs”, Adafruit Industries. Dostęp: 7 luty 2024. [Online]. Dostępne na: https://cdn-learn.adafruit.com/downloads/pdf/adafruit-dotstar-leds.pdf

[12] „Silnik Redox DC 12V 12000RPM”, BOTLAND. Dostęp: 8 luty 2024. [Online]. Dostępne na: https://botland.com.pl/silniki-dc-bez-przekladni/18443-silnik-redox-dc-12v-12000rpm-5903754000331.html

[13] A. Industries, „Adafruit DotStar Digital LED Strip - White 144 LED/m - 0.5 Meter”. Dostęp: 8 luty 2024. [Online]. Dostępne na: https://www.adafruit.com/product/2329

[14] „Arduino® Nano ESP32 — Arduino Official Store”. Dostęp: 8 luty 2024. [Online]. Dostępne na: https://store.arduino.cc/products/nano-esp32

[15] „TLE4906K / TLE4906L”, Infineon Technologies. Dostęp: 7 luty 2024. [Online]. Dostępne na: https://www.infineon.com/dgdl/Infineon-TLE4906L-DataSheet-v02_10-EN.pdf?fileId=db3a304316f66ee8011754425fe50642

[16] „Wireless Charging Module 5V/5A”. Dostęp: 8 luty 2024. [Online]. Dostępne na: https://www.dfrobot.com/product-2086.html

[17] „TLE4906L INFINEON TECHNOLOGIES - Czujnik: Halla | unipolarny; 150Gs; Uzas: 2,7÷18VDC; 20mA | TME - Części elektroniczne (WFS)”. Dostęp: 8 luty 2024. [Online]. Dostępne na: https://www.tme.eu/pl/details/tle4906l/czujniki-halla/infineon-technologies/