UNIWERSYTET ZIELONOGÓRSKI

Wydział Informatyki, Elektrotechniki i Automatyki

PRACA DYPLOMOWA

Kierunek: Automatyka i Robotyka

MODEL STEROWANIA RUCHEM DROGOWYM Z WYKORZYSTANIEM MIKROKONTROLERA

Szymon Pacholski

Promotor:

dr inż. Marcel Luzar

Pracę akceptuję

(data i podpis promotora)

Konsultant:

Zielona Góra, styczeń 2023

Streszczenie

Celemniniejszej pracy było wykonanie modelu sterowania ruchem drogowym z wykorzystaniem mikrokontrolera Arudino Uno. Przedstawione zostały możliwe rozwiązania dostępne na rynku, ich podział oraz uzasadnienie dlaczego zostało wybrane Arduino Uno jako jednostka sterująca. Przedstawiony został proces tworzenia od zaprojektowania makiety oraz schematu elektronicznego, aż po implementacje oprogramowania.

Słowa kluczowe: Arduino, model sterowania, mikrokontroler, skrzyżowanie drogowe

Spis treści

1. Wstęp

1.1. Wprowadzenie

Stale zwiększająca się liczba pojazdów na drogach staje się problemem wielu użytkowników dróg, ponieważ powoduje ogromne zatłoczenie ruchu, zwłaszcza na skrzyżowaniach. Korki na drogach przyczyniają się do niskiej produktywności, straty energii oraz zanieczyszczeń środowiska. Dodatkowo użytkownicy dróg marnują olbrzymią ilość czasu, a także są narażeni na dodatkowe niebezpieczeństwo wynikające z zablokowanych dróg. Wiele z obecnych skrzyżowań, nawet te w większych miastach, nie są dostosowane do ilości pojazdów. Wystarczy zminimalizować czas oczekiwania pojazdów oraz na drogach najbardziej zatłoczonych zmaksymalizować liczbę pojazdów, które będą w stanie pokonać skrzyżowanie.

1.2. Cel i zakres pracy

Celem niniejszej pracy było zaprojektowanie, zbudowanie, zaprogramowanie i uruchomienie, a także przetestowanie makiety skrzyżowania drogowego sterowanej poprzez mikrokontroler Arduino Uno.

Praca swoim zakresem obejmuje:

- zaprojektowanie oraz wykonanie obwodu elektronicznego;
- zaprogramowanie mikrokontrolera Arduino Uno;
- przetestowanie oprogramowania wraz z makietą.

1.3. Struktura pracy

Pracę podzielono na siedem rozdziałów, w rozdziale pierwszym opisano platformę Arduino. Przedstawione zostały możliwe rozwiązania oraz to, dlaczego do projektu został wykorzystany mikrokontroler Arduino Uno. Przedstawiony został mikrokontroler, jego wady i zalety. Dodatkowo opisany został edytor tekstu Visual Studio Code oraz wykorzystane biblioteki tworzące z edytora tekstu środowisko programistyczne.

Rozdział drugi przedstawia projekt oraz wykonanie makiety. Podzielony został na założenia projektowe oraz elementy składowe makiety. Opisane zostały części wykorzystane do projektu a także ich wady oraz zalety. Dodatkowo rozdział ten zawiera informacje dotyczące schematu połączenia elektronicznego, o przebiegu wykonania makiety a także opis jej działania.

Implementacja oprogramowania, które steruje Arduino Uno, jest opisana w rozdziale trzecim. Przedstawiony jest tam algorytm działania makiety oraz jego implementacja na platformie Arduino. Opisane zostały także problemy oraz trudności, napotkane podczas tworzenia oprogramowania.

Czwarty rozdział jest sprawozdaniem z realizacji testów. Opisane zostały w nim testy manualne całej makiety, wraz ze scenariuszami testującymi.

Podsumowanie przedstawia wnioski z realizacji makiety skrzyżowania opartej na platformie Arduino a także potwierdzenie wszystkich testów. Dodatkowo do pracy został dołączony spis pozycji, z których korzystano podczas pisania pracy.

2. Platforma Arduino

2.1. Przegląd rozwiązań

Systemy wbudowane w obecnych czasach to najbardziej powszechny dział informatyki i elektroniki. Najczęściej jako systemy wbudowane rozumie się miniaturowy komputer, którego zadaniem jest wykonywanie zadanych operacji logicznych, poprzez wykorzystanie układów peryferyjnych. Szczególną rolę odgrywają mikrokomputery jednopłytkowe inaczej mikrokontrolery.

Mikrokontroler, czyli system mikroprocesorowy zawierającą jednostkę centralną (CPU), pamięć RAM oraz rozbudowany układ wejść i wyjść. Głównym obszarem jego zastosowań jest sterowanie różnego rodzaju urządzeniami elektronicznymi. Biorąc pod uwagę dostępne rozwiązania na rynku wyróżnić, można przewodnie jednostki sterujące.

1. Raspberry Pi

    1.1. Historia Raspberry Pi

    Pierwszy mikrokontroler Raspberry Pi do sprzedaży trafił w 2012 roku. Pracę nad nim rozpoczęto sześć lat wcześniej w 2006 roku na uniwersytecie w Cambridge. Inspiracja do jego stworzenia stanowiły komputery BBC Micro z serii Acorn Computer. Głównym celem tego projektu były uczelnie techniczne, które mogłyby zapewnić studentom tanią platformę komputerową, dzięki czemu studenci mieliby możliwość w prosty sposób poznania systemów komputerowych od strony sprzętowo-programowej. Projekt Raspberry Pi osiągnął światowy sukces, który niezmiennie jest podtrzymywany. Do 2018 roku na całym świecie zostało sprzedanych 20 milionów sztuk owego mikrokontrolera oraz powstała olbrzymia społeczność majsterkowiczów o hobbystycznym podejściu jak i zawodowych inżynierów, którzy publikują swoje prace w internecie. Dzięki zrzeszeniu tak ogromnej ilości miłośników Raspberry Pi, każdy bez problemu uzyska pomoc oraz wsparcie techniczne niezbędne przy tworzeniu własnego projektu. Możliwości Raspberry Pi są nieograniczone i pomimo ponad dekaty na rynku w dalszym ciągu się rozrastają.

Rysunek 2.1. Pierwsza wersja Raspberry Pi

https://upload.wikimedia.org/wikipedia/commons/3/3d/RaspberryPi.jpg

Na rynku pojawiło się wiele rozwiązań Raspberry Pi, od bardziej do mniej rozbudowanych. Przeglądając najnowsze oraz te starsze rozwązania, szybko nasuwa się wniosek, że Raspberry Pi oraz jego możliwości zdecydowanie przerastają potrzeby oraz wykonywane operacje przez makiete skrzyżowania, kóre zostały opisane w rozdziale 3.1. Biorąc pod uwagę koszty, również owy mikrokontroler nie wygląda zachęcająco.

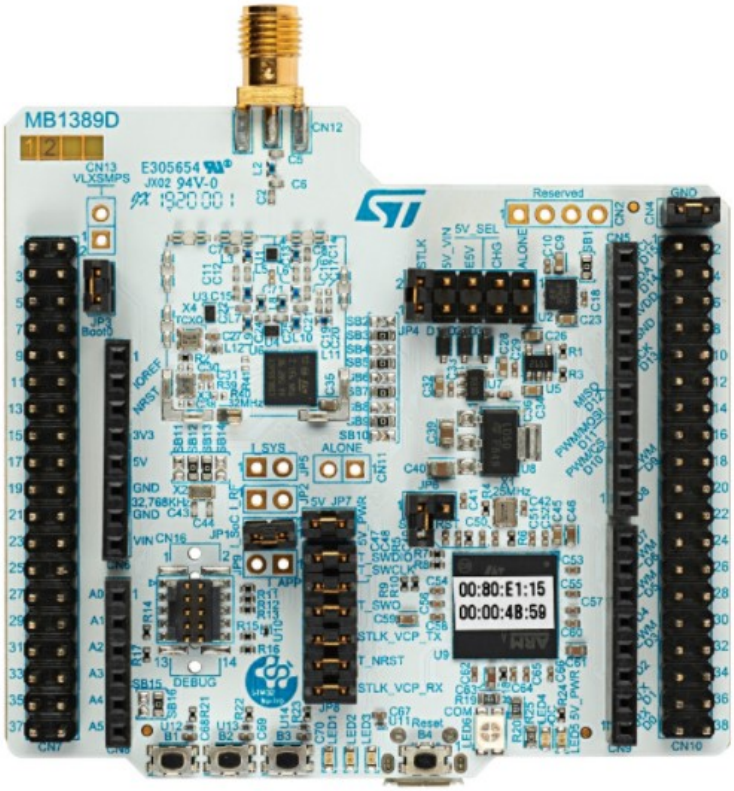
2. STM32

2.1. Historia STM32

Historia STM32 od firmy STMicroelectronics sięga już ponad dekatę, kiedy to w 2008 roku, na rynek został wprowadzony pierwszy mikrokontroler, noszący nazwę STM32F1 powstałego na bazie rdzenia Cortex-M3 stwożonego przez firmę ARM.

W następnych latach aż do 2018 roku firma ARM tworzyła kolejne rdzenie a ze stajni STMicroelectronics wychodziły kolejne mikrokontrolery. W roku 2018 została wprowadzona nowa kategoria układów SoC, która

łączy mikrokontroler z nadajnikiem-odbiornikiem radiowym. Pozwoliła ona na kominikacie w paśmie 2.4 GHz, koncentrując się głownie na aplikacjiach wykorzystujących technologie Bluetooth. Najnowsze rozwiązania od 2020 roku posiadają komunikacje w paśmie sub-GHz z modulacją LoRa, która to powala na daleką komunikacje wraz z niskim poborem energii. Minusem takiej modulacji jest niska przepustowość danych.



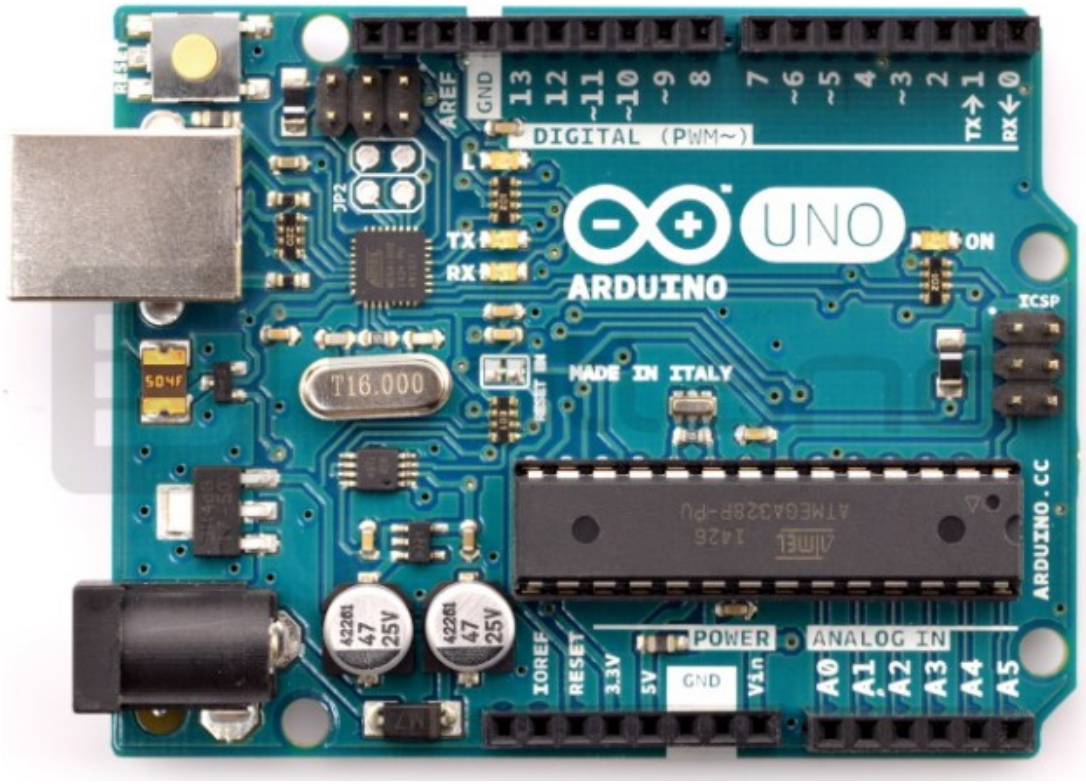
Rysunek 2.2. Mikrokontroler STM32WL
https://elektronikab2b.pl/produkty/produkcja-elektroniki/oprogramowanie-projektowe/14597-plytki-ewaluacyjne-nucleo-64-z-mikrokontrolerem-stm32wl-z-wbudowanym-transceiverem-lora

Rodzina mikrokontrolerów STM32, jest jest bardzo dobrym rozwiązaniem biorąc pod uwagę chęc komunikacji aplikacji oraz mikrokontrolera bezprzewodowo. Aczkolwiek wracając do założeń projektowych, nie jest to interesująca funkcjonalność.

3. Arduino

3.1. Historia Arduino

Historia Arduino sięga roku 2003 gdzie to jej pierwotnym założeniem było stwożenie platformy edukacyjnej do nauki elektroniki jak i programowania z użyciem układów scalonych zawierających procesor, pamięć oraz porty wyjść i wejść. Na rynek rozwiązanie Arduino zostało wprowadzone w roku 2008 pod nazwą Arduino Nano a następnie w 2010 roku, zaprezentowane zostaje rozwiązanie Arduino Uno, które podbiło rynek stając się zdecydowanie najczęściej wybieranym mikrokontrolerem i znajduje swoje zastosowanie do dnia dzisiejszego w wielu rozwązaniach oraz na uczelniach wyższych.



Rysunek 2.3. Mikrokontroler Arduino Uno
https://raspberrypivsarduino.com/historia-arduino/#:~:text=Pocz%C4%85tki%20rozwoju%20platformy%20Arduino%20si%C4%99gaj%C4%85,oraz%20zestaw%20port%C3%B3w%20wej%C5%9Bcia%2Fwyj%C5%9Bcia.

Arduino Uno wyposażone jest w 8 bitowy procesor, niewielką pamięć operacyjną oraz 14 portów wejść i wyjść. Takie rozwiązanie pozwala na dość

sporą ilość wykonywanych operacji. Niestety owy procesor posiada tylko jeden wątek co za tym idzie, niemożliwe jest zastosowanie wielowątkowości.

Przy wyborze mikrokontrolera kluczowe jest jego wykorzystanie. Na rynku występuje wiele rozwiązań z pełnym wachlarzem specyfikacji, zaczynając od mniej rozbudowanych jak Arduino, aż po bardziej zaawansowane jak Raspberry Pi, które dorównuje niemal komputerom osobistym. W przypadku makiety inteligentnego skrzyżowania sterowanej poprzez mikrokontroler, bardzo ważnym punktem jest zdecydowanie, jakie operacje logiczne będą wykonywane poprzez ten mikrokontroler. Biorąc pod uwagę założenia projektowe opisane w rozdziale 3.1, oraz mikrokontrolery STM32, Arduino oraz Raspberry Pi, a także ich cenę. Najbardziej rozsądną opcją, która spełnia wszstkie założenia jest mikrokontroler Arduino Uno również on zostanie wykorzystany podczas tworzenia projektu.

2.2. Czym jest Arduino Uno?

Arduino Uno czyli tani mikrokontroler, stosowany w wielu projektach akademickich jak i komercyjnych. Jak na jego niepozorną specyfikacje techniczną pozwala na realizacje dość zaawansowanych operacji logicznych.

Microcontroller	ATmega38P – 8 bit z rodziny AVR
Operating Voltage	5V
Recommended Input Voltage	7-12V
Input Voltage Limits	6-20V
Analog Input Pins	6 (A0-A5)
Digital I/O Pins	14 (Z czego 6 wspiera wyjścia analogowe)
DC Current on I/O Pins	40mA
DC Current on 3.3V Pin	50mA
Flash Memory	32 KB (0.5 KB zarezerwowane dla Bootloader'a)
SRAM	2kB
EEPROM	1kB
Frequency (Clock Speed)	16MHz

Tabela 2.1. Specyfikacja techniczna Arduino Uno

Analizując specyfikacje techniczną widać, że Arduino Uno, nie posiada zaawansowanej jednostki obliczeniowej oraz pamięci, nie ma również wiele wyjść i wejść.

2.3. Środowisko Visual Studio Code

Platforma Arduino wspiera język programowania C/C++, również posiada ona swoje własne wbudowane środowisko programistyczne ArduinoIDE, któremu niestety sporo brakuje do pełnoprawnego środowiska znanego z obecnych rozwiąń znanych z nowoczesnych jezyków programowania na przykład Visual Studio lub IntelliJ IDEA.

Wykorzystany zostanie edytor tekstowy Visual Studio Code wywodzący się z firmy Microsoft oraz będący narzędziem open source. Co oznacza, że jest to w pełni darmowe rozwiązanie. Dodatkowo Visual Studio Code, cieszy się wielką popularnością wsród programistów i programistek na całym świecie. Olbrzymia biblioteka wtyczek i rozszerzeń modyfukująca niepozorny edytor testów tak aby stał się pełnoprawnym środowiskiem programistycznym dla dowolnego języka, zaczynając od nowoczesnych typu Python lub Java a kończąc na programowaniu w ANSI C a nawet w Kobolcie czy Assemblerze.

Dodatkowym narzędziem które zostanie wykorzystane jest system kontroli wersji Git, który pozwoli na śledzenie i łatwe wprowadzanie zmian w napisanym kodzie.

2.3.1 Wykorzystane biblioteki

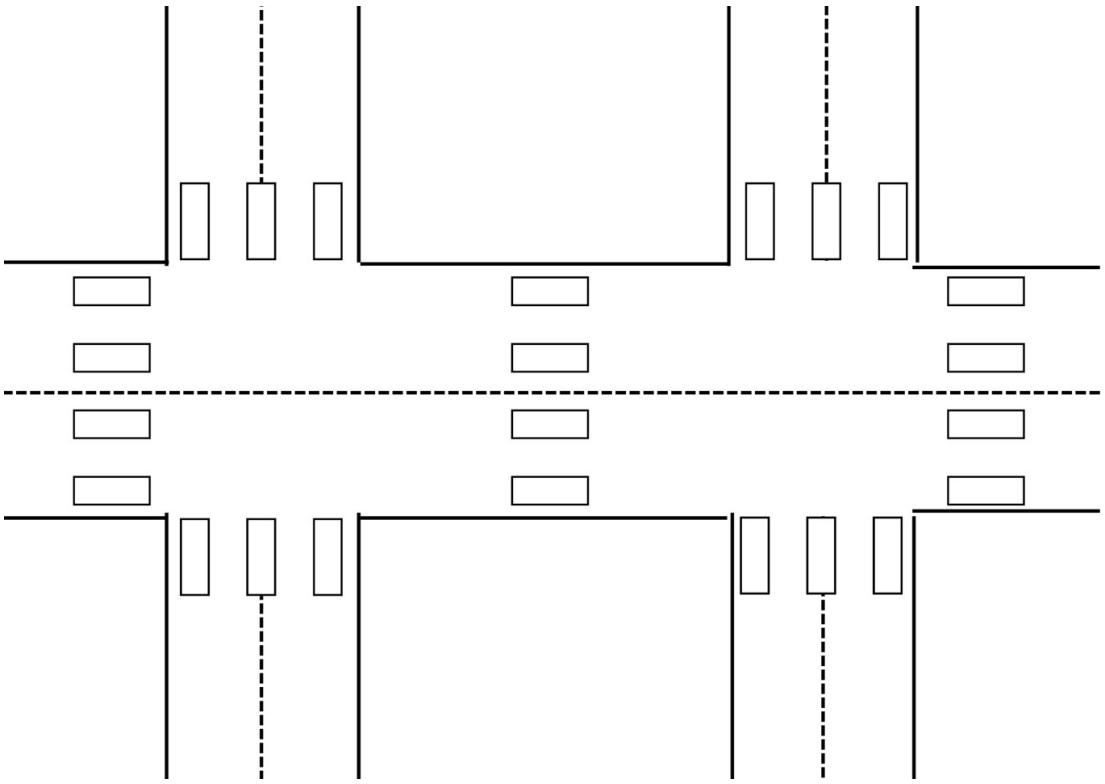
(OPISAĆ WYKORZYSTANE ROZSZERZENIE, ZE POWSTALO OD MICROSOFT ITD.)

3. Projekt i wykonanie makiety

3.1. Założenia projektowe makiety

Projektując makiete, najważniejszym jej aspektem jest oprogramowanie i operacje logiczne jakie będzie wykonywać. Ten rozdział przedstawi proces tworzenia makiety zaczynając na logice przez oprogramowanie, aż po fizyczne wykończenie.

Makieta działa w dwóch trybach dzień i noc. Pozwoli to na lepszą organizacje ruchem drogowym podczas dnia, w godzinach szczytu, a także w nocy, kiedy ruch drogowy jest bliski zeru.



Rysunek 3.1 Projekt skrzyżowania

Skrzyżowanie składa się z drogi głównej oraz dwóch dróg podporządkowanych. Podczas przejazdu poprzez drogę główną w trybie dziennym dochodzi do zielonej falii, co umożliwi sprawny przejazd pojadów drogowych. W czasie zielonej falii niemożliwy jest przejazd drogą podporządkowaną, w tym czasie zapalone jest światło czerwone. Natomiast przy przejściu dla pieszych, na drodze

podporządkowanej z jednej oraz drugiej strony, zapalone jest światło zielone, umożliwiające przejście pieszym.

Po okresie trwania zielonej falii, następuje przełączenie się świateł. Tak aby pojazdy z drogi podporządkowanej mogłby swobodnie i bezpiecznie przejechać. W tym momencie przejście pieszych jest niemożliwe. W tym czasie przejścia dla pieszych na drodzie głównej zmieniają swój stan z koloru czerwonego na kolor zielony.

Cykl zmiany świateł w trybie dziennym działa trwa określony okres czasu, a także działa w nieskończonej pętli.

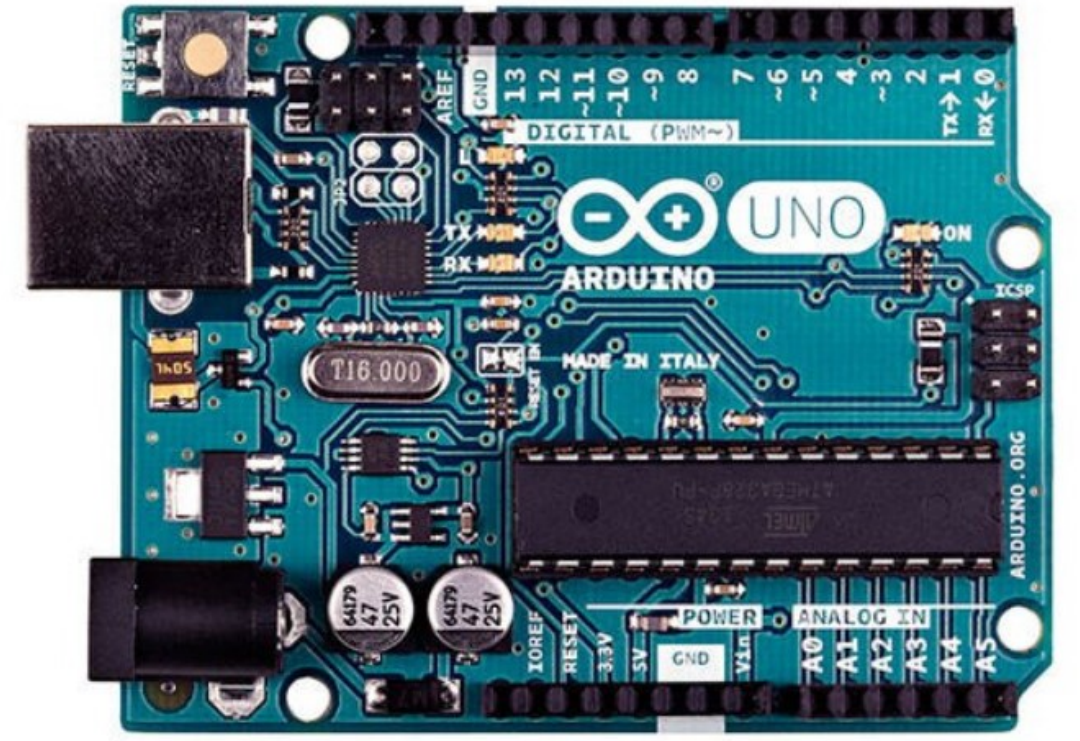
W trybie nocnym na drodze głównej, ze względu na założenie, że jest to najbardziej ruchliwa ulica, zapalone jest ciągle światło zielone. Pieszy podczas przejścia przez ulice może nacisnąć przycisk, co spowoduje zmianę świateł na całej ulicy i zmianę koloru świateł dla przejścia z czerwonego na zielony.

Droga podporządkowana w tym czasie jest w stanie światła czerwonego a przejście dla pieszych jest oznakowane kolorem zielonym. W momencie zbliżenia się pojazdu do skrzyżowania, zostaje on wykryty poprzez czujnik zbliżeniowy i w momencie wykrycia następuja zmiana świateł blokując przejście pieszym, tym samym umożliwiając przejazd pojazdu drogowego. W momencie zmiany światła na drodze podporządkowanej na kolor zielony dochodzi również do zmiany świateł na drodze głównej, co pozwoli na bezkolizyjne pokonanie skrzyżowania przez pojazd.

3.2. Elementy składowe makiety

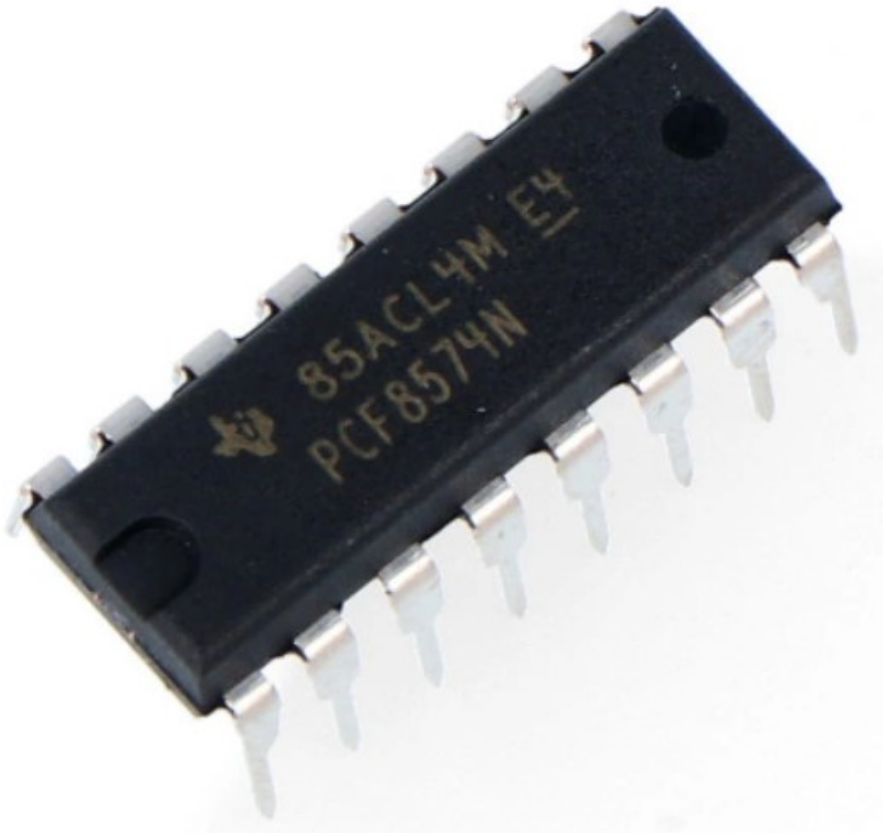
Projekt makiety skrzyżowania drogowego, sklada się z wielu elementów, które zostaną opisane w tym rozdziale.

Sercem całej makiety jest mikrokontroler Arduino Uno, głównym powodem jego zastosowania jest niska cena w stosunku do możliwości które może wykonać.



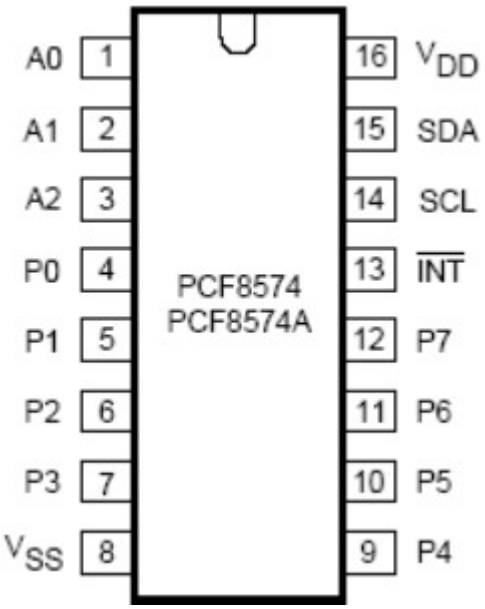
Rysunek 3.1 Mikrokontroler Arduino Uno http://technovade.pl/arduino-uno-r3.html

Ze względu na rozbudowany schemat połączeń oraz wykorzystanie dużej ilości diód niezbędnym aspektem było zaopatrzenie się w ekspandery wyprowadzeń dla mikrokontrolerów PCF8674N. Pozwoli on na zwiększenie ilości pinów Arduino Uno o 8.



Rysunek 3.2. Ekspander PCF8574N https://botland.com.pl/ekspandery-wyprowadzen/1210-pcf8574n-ekspander-wyprowadzen-mikrokontrolera-5904422303082.html

Ekspander PFC8574N komunikuje się z mikrokontrolerem za pomocą magistrali I2C (TWI). Możliwe jest podłączenie maksymalnie ośmiu takich ekspanderów w jednym momencie, wynika to z podłączenia go do Arduino Uno.



Rysunek 3.3 Schemat ekspandera PCF8574N

https://www.letscontrolit.com/wiki/index.php?title=File:PCF8574-pins.gif

Podłączenie ekspandera do płyki mikrokontrolera następuje poprzez podpięcie do masy oraz napięcia nóżek oznaczonych na schemacie poprzez 1, 2, 3 (Rysunek 3.3). Podłączenie polega na odzwierciedleniu zapisu binarnego. Oznacza to, że nóżka podpięta do masy informuje o tym, że podawany na tę nóżkę jest syngał 0. Analogicznie podpięcie do napięcia 5V, mówi o stanie 1. Każdy ekspander musi być podpięty w indywidualny sposób, czyli maksymalnie może zostać podpięte osiem ekspanderów. Dzięki czemu jest możliwość zwiększenia, pinów mikrokontrolera, aż o 56.

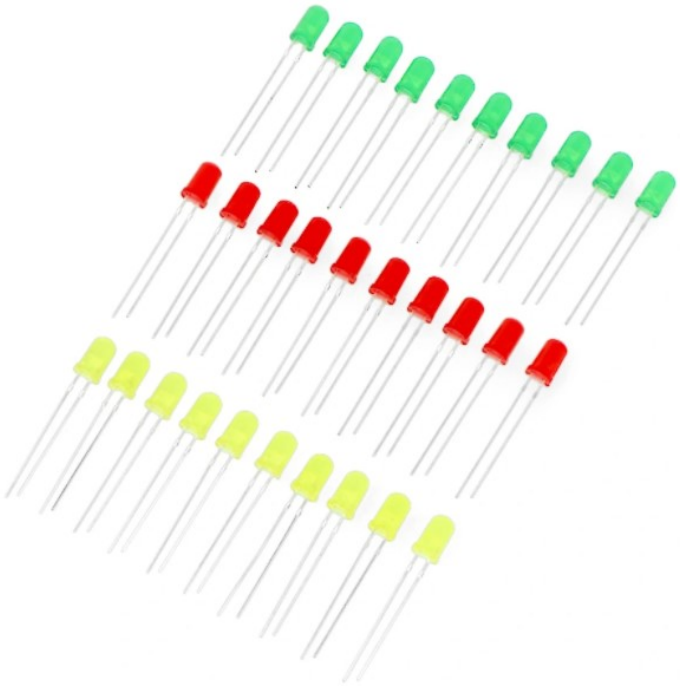
Schemat połączeniowy makiety, został wyposarzony o cztery ekspandery PFC8574N, co pozwoliło na zwiększenie potów I/O o 24 piny. Jest to zdecydowanie wystarczająca ilość.

Decyzja o użyciu aż czterech ekspanderów, została spowodowana tym, że element ten jest stosunkowo tani a większa ilość pozwoli na przypisanie kazdemu elementowi jednego zadania, co następnie ułatwi pisanie kodu, a także go uporządkuje oraz uprości poziom rozbudowania schematu elektronicznego.

Każdy ekspander ma oddzielne zadanie. Dwa z czterech sterują światłami skrzyżowania drogowego, jeden lewym drugi prawym a

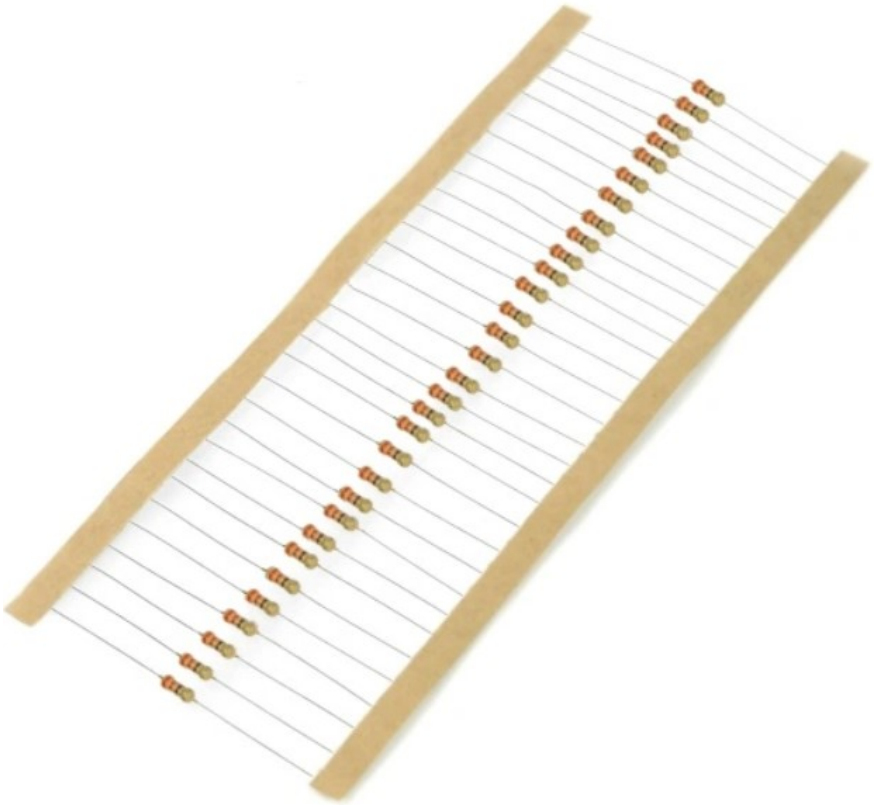
pozostałe dwa przejściem dla pieszych również lewym oraz prawym. Takie rozwiązanie sprawi, że układ oraz montaż będzie bardziej przejrzysty a co za tym idzie nastąpi minimalizacja pomyłek podczas lutowania poszczególnych elementów.

Wykorzytane zostały również diody czerwone, zielone i żółte. Symbolizujące oświetlenie uliczne.



Rysunek 3.4 Diody https://botland.com.pl/diody-led/19985-zestaw-diod-led-5mm-justpi-30szt-5904422329259.html

Diody (Rysunek 3.4) podpięte są szeregowo wraz z rezystorem 330Ohm (Rysunek 3.5). Spowoduje to uniknięcie przepłynięcie prądu maksymalnego przez diode, co spowodowałoby przepalenie diody.



Rysunek 3.5 https://cdn1.botland.com.pl/103426-pdt_540/rezystor-justpi-tht-cf-weglowy-14w-330-30szt.jpg

Przejścia dla pieszych wyposażone są w przyciski bez podtrzymania (Rysunek 3.6). Przyciski te, są jednymi z prostszych rozwiązań, aczkolwiek są również w pełni wystarczające.

Rysunek 3.6. Tact Switch 6x6mm https://botland.com.pl/tact-switch/3495-tact-switch-6x6mm-5mm-tht-2pin-5szt-5904422307639.html

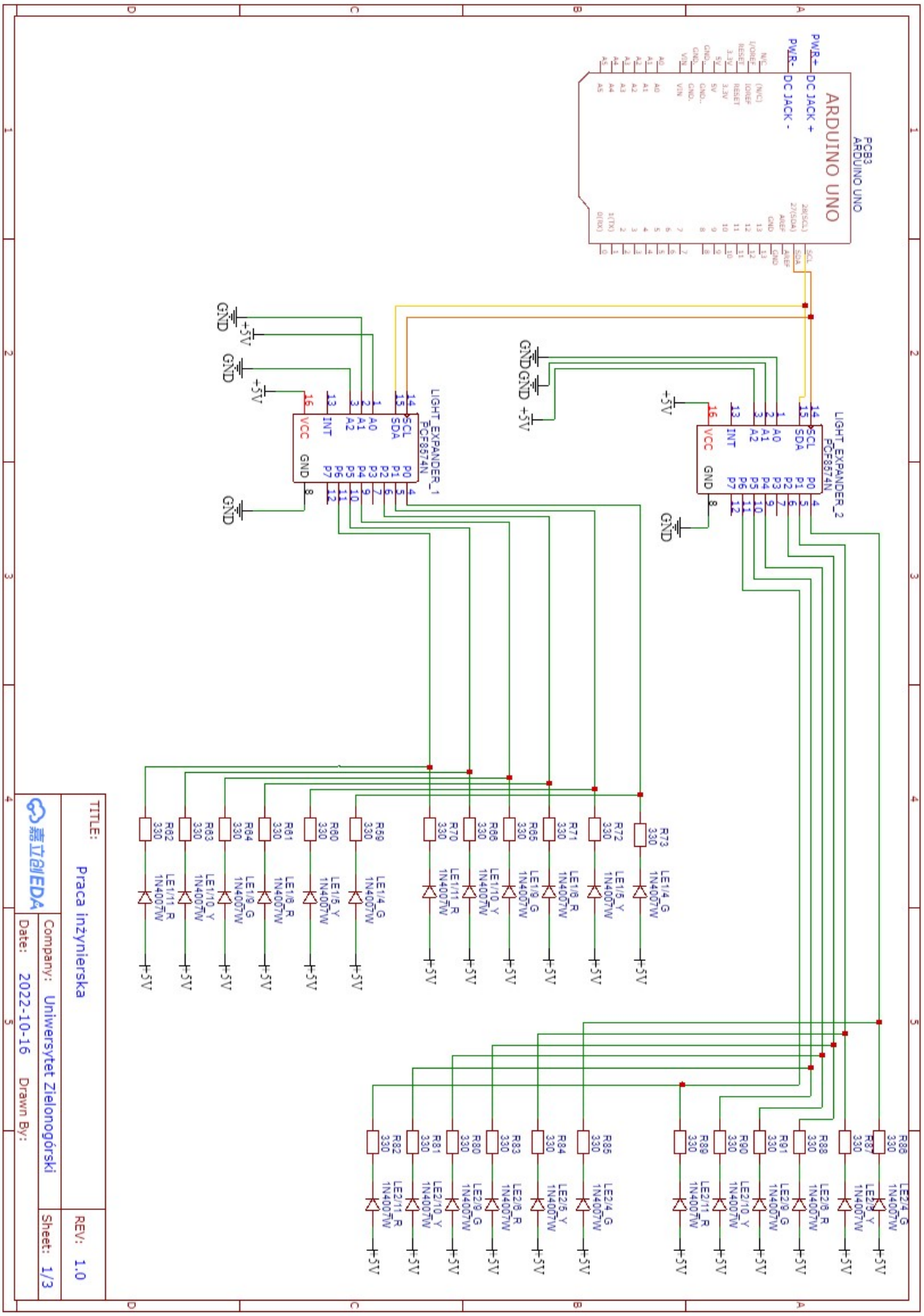
W trybie nocnym dla dróg podporządkowanych zastosowane są czujniki odległości. Podczas analizy rynkowej, wyróżnić można kilka rozwiązań tychże czujników. Zaczynając od ultradzwiękowych aż po czujniki na podczerwień.

Biorąc pod uwagę wymagania projektowe makiety oraz jej opis działania. Wybór padł na czujnik uktradźwiękowy HC-SR04 (Rysunek 3.7)

Rysunek 3.7 Czujnik ultradźwiękowy HC-SR04
https://botland.com.pl/ultradzwiekowe-czujniki-odleglosci/1420-ultradzwiekowy-czujnik-odleglosci-hc-sr04-2-200cm-5903351241366.html

3.3. Schemat połączenia

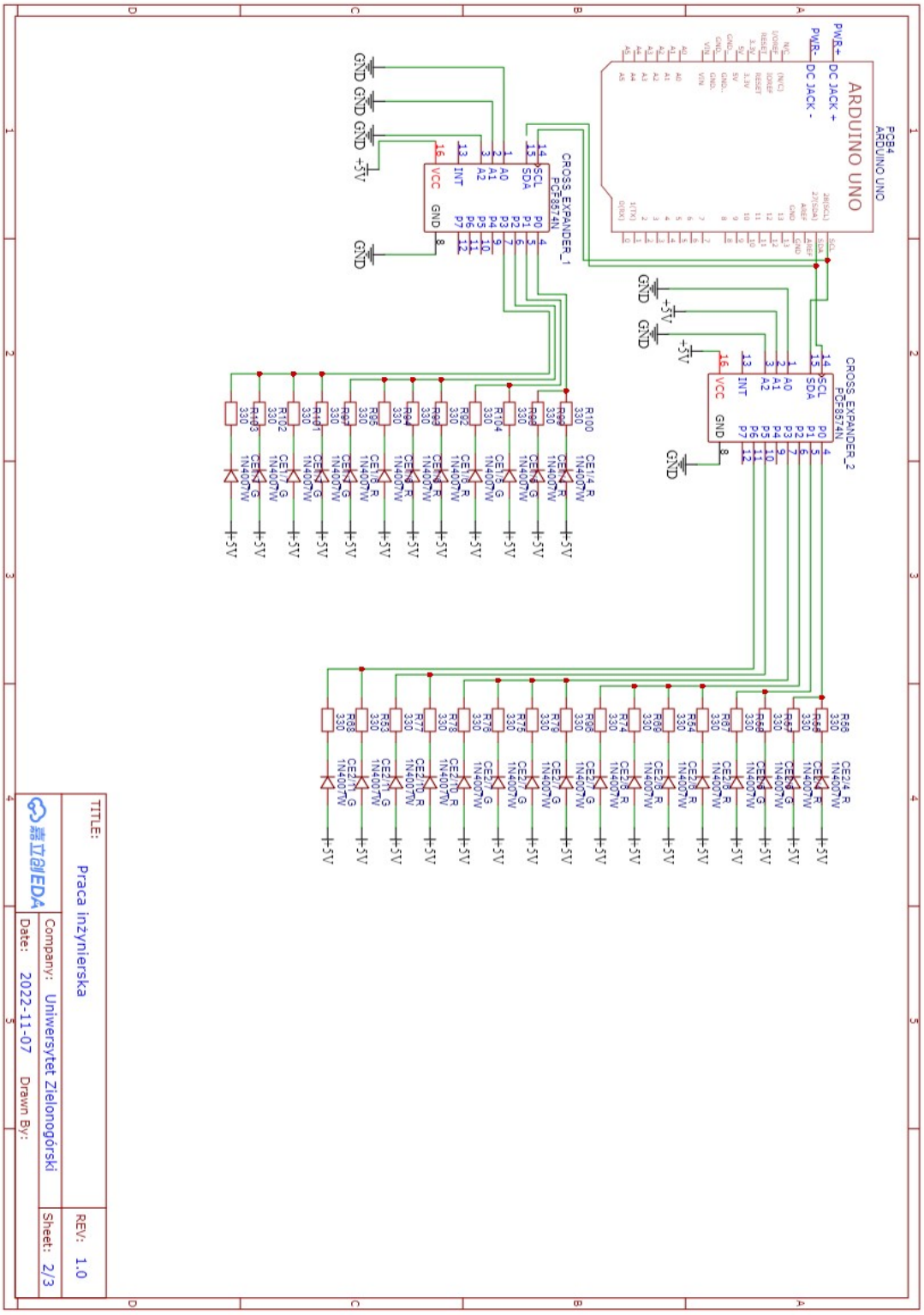
Schemat połączenia projektu został podzielony na trzy części. Na pierwszej (Rysunek 3.8) przedstawione zostało połączenie sygnalizacji świetlnej na drodze głównej oraz drodze podporządkowanej.



Rysunek 3.8 Połączenie sygnalizacji świetlnej

Schemat ten przedstawia dwa ekspandery PCF8574N, które to będą odpowiadać za sterowanie diodami, poprzez podawanie sygnału wysokiego lub niskiego a co za tym idzie będą zapalać lub gasić diody.

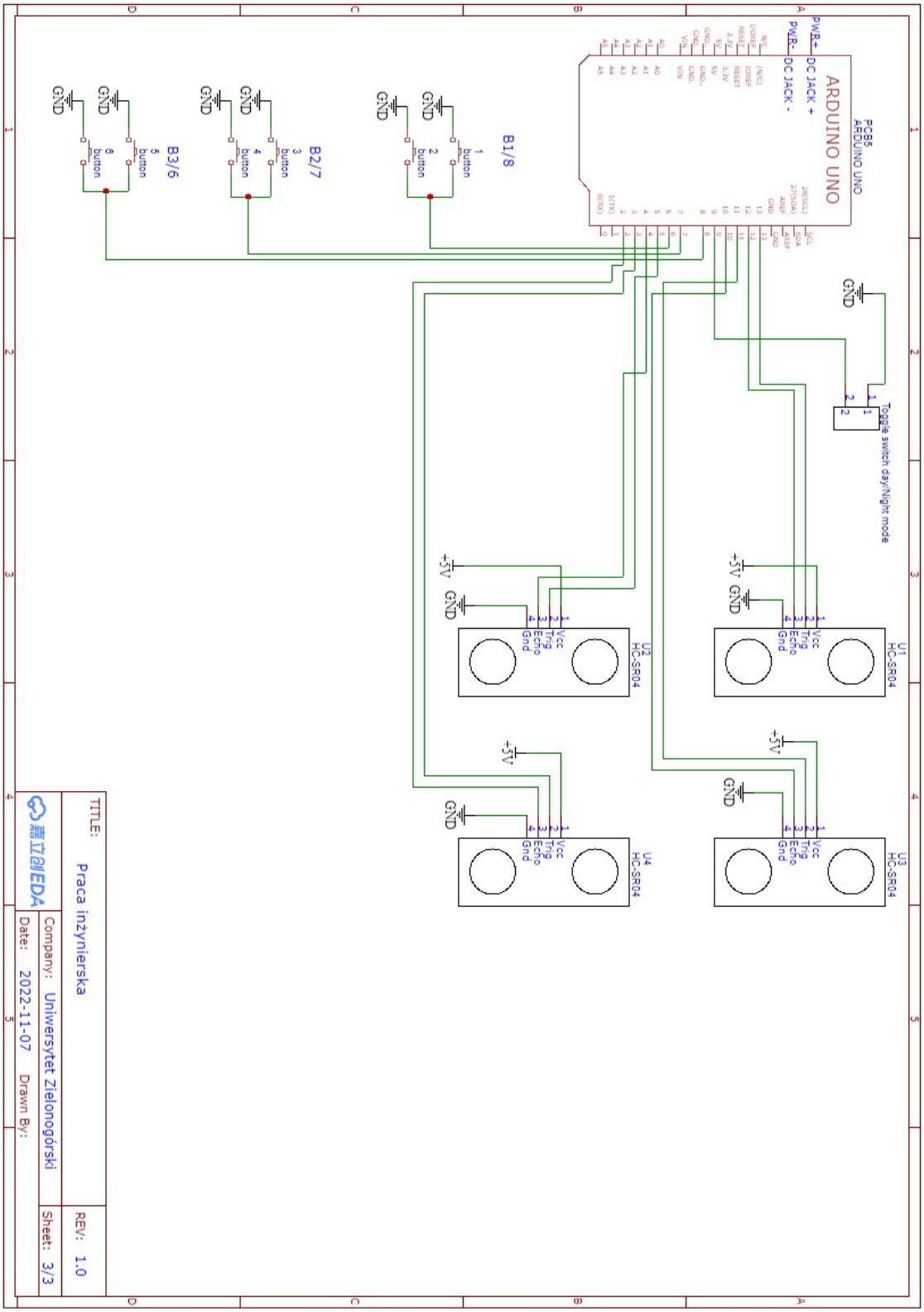
Kolejna część schematu (Rysunek 3.9) przedstawia połączenie świateł odpowiedzialnych za przejście dla pieszych. Również jak w części pierwszej (Rysunek 3.8) schemat został podzielony na dwa ekspandery PCF8574N, oraz odpowiednią ilość diód.



Rysunek 3.9 Połączenie przejść dla pieszych

Ostatnią częścią (Rysunek 3.10) jest schemat połączenia czuników uktradzwiękowych. Czujniki HC-SR04 będą bezpośrednio podłączone pod piny wychodzące z mikrokontrolera Arduino. Podobna sytuacja dotyczy przycisków, które będą odpowiedzialne za zmianę świateł przejść dla pieszych. One natomiast są połączone ze soba w pary, taki zabieg pozwoli na takie samo sterowanie jednym przejściem w tym samym czasie.

Jest także przełącznik sterujący makietą poprzez zmianę trybu dzień/noc.

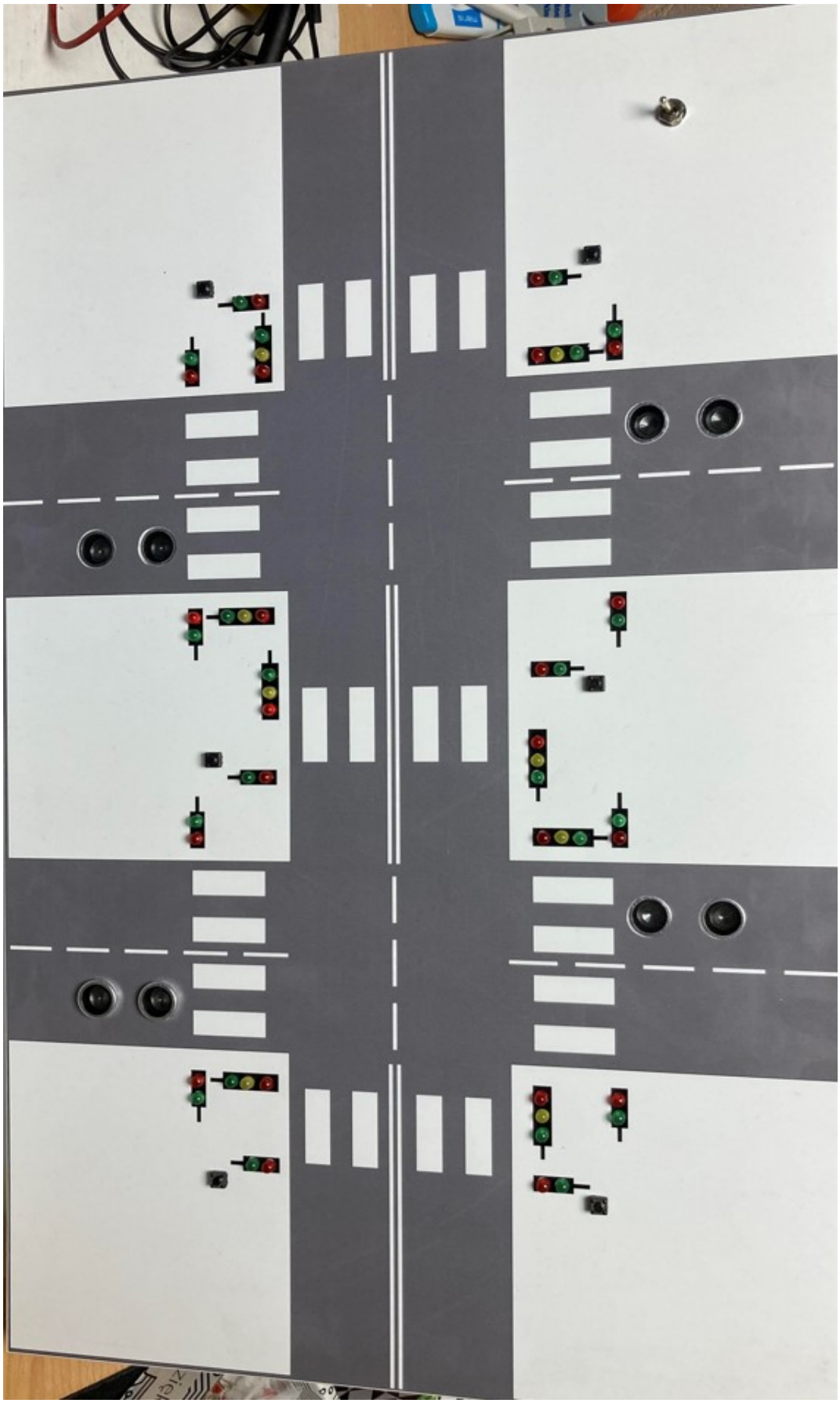


Rysunek 3.10 Podłączenie czujników

Zauważyć można na schematach (Rysunek 3.8, Rysunek 3.9), że dioda nie jest podłączona w standardowy sposób. Wynika to z tego, że wydajność prądowa wychodząca z ekspandera jest znacznie większa podczas sterowania stanem niskim, niż podczas sterowania stanem wysokim. Według specyfikacji producenta maksymalna wartość mogąca popłynąć podczas sterowaniem stanem niskim to 25 mA, natomiast podczas sterowaniem wyoskim wartość ta drastycznie maleje i wynosi tylko 300µA, więc dzięki takiemu zabiegowi, diody znacznie wydajniej świecą.

3.4. Wykonanie makiety

Makieta została wykonana z materiału plexi (Rysunek 3.11), na którym nadrukowany został projekt skrzyżowania.



Rysunek 3.11 – Makieta skrzyżowania

Pierwszym krokiem był montaż poszczególnych elementów, zaczynając od nawierceniu otworów pod diody, a także czujniki HC-SR04.

Następnie zostały zamontowane przyciski odpowiedzialne za przejścia dla pieszych oraz przełącznik sterujący trybem dzień oraz noc.

Później nastąpił montaż czujników i diód w odpowiednie miejsca.

Końcowym etapem był etap lutowania poszczególnych peryferii elekronicznych zgodnie ze schematem połączeniowym opisanym wyżej.

3.5. Opis działania makiety

Makieta działa w dwóch trybach dziennym i nocnym sterowanym poprzez przełącznik (Rysunek 3.11).

W trybie dziennym skrzyżowanie jest sterowane czasowo. Po włączeniu rozpoczyna się ustawianie parametrów początkowych, czyli na drodze głównej włączone jest światło zielone umożliwiające przejazd pojazdom. Natomiast na drodze podporządkowanej przejazd pojazdów jest uniemożliwiony poprzez zapalenia diód koloru czerownego.

Dodatkowo przejścia dla pieszych na drodze głównej są wyłączone z użytku, czyli włączone zostaje światło czerwone. Na drodze podporządkowanej przejścia dla pieszych, są włączone co symbolizuje dioda koloru zielonego.

Po określonym czasie następuje zmiana. Światła na drodze głównej oraz podporządkowanej zostają zmienione na przeciwne, a także światła sygnalizujące przejście dla pieszych. Program działa w nieskończonej pętli, co za tym idzie po określonym czasie znowu zmieniają się światła. Tym razem zmiana następuje poprzez zieloną falę. Co umożliwi płynne i szybkie pokonanie skrzyżowania przez pojazdy drogowe.

Tryb nocny charakteryzuje to, że parametrami początkowymi jest ustawienie diód tak aby ciągle możliwy był przejazd pojazdów poprzez drogę główną, a także uniemożliwienie przejścia pieszym. Droga główna natomiast jest oznaczona światłem czerownym, tak aby uniemożliwić przejazd, ale również włączone jest światło zielone dla pieszych.

W tym trybie widoczne przyciski na makiecie (Rysunek 3.11) odpowiadają za umożliwienie przejścia pieszym, chcącym przejść przez drogę główną. Po naciśnieciu następuje włączenie świateł czerwonych dla pojazdów, tym samym uniemożliwiając im przejazd. Następnie zostaje zapalone światło czerwone dla przejść dla pieszych.

Wykorzystane zostały również czujniki odległości HC-SR04 (Rysunek 3.11), które po wykryciu pojazdu znajdującego się nad tym czujnikiem da sygnał do skrzyżowania aby to zmieniło swoje światła na drodze głównej oraz podporządkowanej tak aby umożliwić przejazd temu pojazdu.

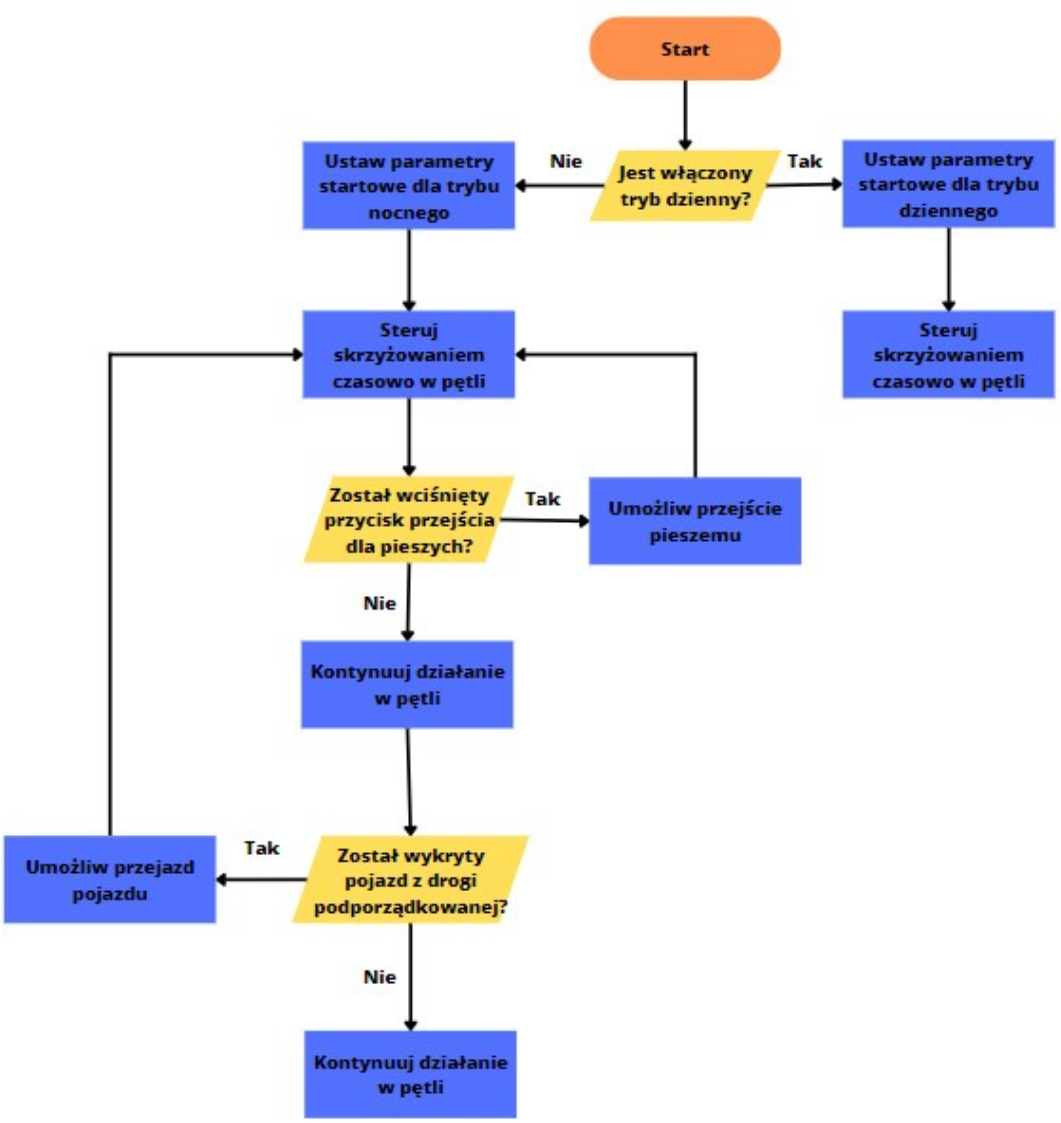
4. Implementacja oprogramowania

4.1. Założenia

Przed rozpoczęciem realizacji oprogramowania przyjęto założenia:

- Do programowania jako środowisko programistyczne zostanie wykorzysany Visual Studio Code, wraz z odpowiednimi bibliotekami i wtyczkami.
- Program będzie napisany w sposób obiektowy z podziałem na odpowiednie klasy.
- Zachowane zostaną standardy czystego kodu opisane w książce Clean Code.

4.2. Schemat blokowy algorytmu



Rysunek 4.1 Rysunek blokowy alogrytmu sterowania makietą ruchu drogowego

4.3. Kod programowy realizujący algorytm

Założeniem kodu było napisanie go w sposób jasny i przejżysty, dodatkowo zgodnie z paradygmantami programowania obiekowego.

Na potrzeby oprogramowania postał osobny plik z klasą o nazwie **TrafficLight**, która to definiuje metody użytkowe na zewnątrz. Oznacza to, że wymienione metody z parametrem *public* są dostępne z poziomu obiektu tejże klasy.

```
class TrafficLight
{
   public:
      void setUpTrafficLight(PCF8574 CE1, PCF8574 LE2, PCF8574 CE2, PCF8574 LE1);
      void dayMode();
      void setDayModeStartParameters();
      void nightMode();
      void setNightModeStartParameters();
};
```

Listing 4.1 – Klasa odpowiedzialna za sterowanie skrzyżowaniem.

Omawiając wyżej wymienione funkcje to *setUpTrafficLight* odpowiada konstrukcji konstruktora z parametrami. W niej po podaniu obiektów klasy PCF8674 czyli ekspanderów. Zostaje ustawiona klasa wraz ekspanderami oraz ich właściwościami takimi jak odpowiednie przypisanie diód do pinów.

Funkcje *setDayModeStartParameters* oraz *setNightModeStartParameters* odpowiada za ustawienie parametrów domyślnych dla poszczególnych trybów działania makiety.

W funkcjach *dayMode* oraz *nightMode* wykonywany jest algorytm sterujący poszczególnymi trybami. Korzystają one z funkcji prywatnych, w których to następuje poszczególna logika sterująca diodami tak jak w założeniach działania makiety.

Zadbane zostało o to, aby funkcje wykonywujące logikę w której dochodzi do zmian świateł nie były możliwe do modyfikacji poza tą klasą. Dlatego, żeby wywoływać te funkcje zostały stworzone funkcje publiczne widoczne na Listingu 4.1.

5. Testy

W tym rozdziale zostaną opisane testy manualne makiety skrzyżowania drogowego sterowanego poprzez mikrokontroler Arduino Uno. Testy zostały przeprowadzone osobno w trybie dziennym oraz nocnym poprzez wpisanie wartośći w kodzie programu, a także w połączeniu obu tych trybów poprzez ich zmianę przełącznikiem

Rozpoczynając od testowania trybu dziennego w kodzie programu została wprowadzona wartość informująca mikrokontroler, że teraz działa w trybie dziennym. Co za tym idzie zgodnie z działaniem makiety i parametrów startowych dla tego trybu ustawił odpowiednie diody oraz przeszedł do sterowania ruchem czasowo, zgodnie z zadeklarowaną wartościa. Najpierw został umożliwiony przejazd przez drogę podporządkowaną, następnie wrócił do parametrów startowych i poprzez zieloną fale, został umożliwiony przejazd wszystkim pojazdom. Zachowanie w tym trybie było zgodne z oczekiwaniem opisanym w rozdziale 3 dotyczącym działania makiety.

Następnie testy dotyczyły trybu nocnego w którym to po uruchomieniu makieta przyjęła wartośći startowego tego trybu i w nim pozostała do momentu naciśnięcia przycisku. W tym momencie światło na drodze głównej zmieniło swoją wartość na kolor czerwony, a przejścia dla pieszych kolor zielony. Umożliwiło to bezpieczne pokonanie ulicy przez pieszych. Następnie makieta wróciła do parametrów startowych.

Kolejnym ważnym elementem było sprawdzenie poprawności działania czujników odegłości HC-SR04. W momencie wykrycia pojazdu stojącego na czujniku, doszło do zmiany koloru świateł z czerwonego na zielony oraz zmiany kolorów na drodze głównej z zielonego na czerwony. Umożliwiło to przejazd pojazdom z drogi podporządkowanej, następnie skrzyżowanie wróciło do stanu początkowego.

Ostatnim wykonanym testem jest test przełącznika, który zmienia tryby makiety. Przy pierwszym włączeniu, kiedy przełącznik jest w trybie dziennym makieta ustawia swoje parametry startowe w sposób zgodny z trybem dziennym. Następnie po zmianie trybu, przełączeniu przełącznika dochodzi do zmiany

trybu na nocny, tym samym przybierając parametry startowe odpowiednie dla trybu nocnego.

Wniosek płynący z testów manualnych jest taki, że zgodnie z opisem działania makiety zamieszczonym w rozdziale 3. Skrzyżowanie drogowe sterowane poprzez mikrokontroler, spełnia wszystkie wymagania i działa zgodnie z założeniami.

6. Podsumowanie

W realizacji pracy zaprojektowano wygląd makiety wraz z niezbędnymi otworami na peryferia elektroniczne takie jak diody czy czujniki odległości HC-SR04. Został zaprojektowany również obwód elektroniczny pozwalający na sterowanie odpowiednimi cześciami z poziomiu programu komuterowego.

Przedstawiono możliwe rozwiązania jednostki sterującej oraz uzasadniono wybór, który padł na Arduino Uno.

Zanim przystąpiono do realizacji oprogramowania, które pozwoli sterować skrzyżowaniem, został zaprojektowany algorytm sterujący oraz założenia jakie mikrokontroler musi realizować. Przy implementacji oprogramowania, niezbędnym czynnikiem było zapoznanie się z obsługą elementów w kodzie programu.

Zostały wykonane testy manualne po zakończeniu montażu części elektronicznych oraz wgraniu oprogramowania. Testy dotyczyły każdego elementu, działającego w trybie dziennym jak i nocnym.

Zdaniem autora cel został osiągnięty. Została zbudowana makieta symulująca sterowanie skrzyżowaniem, które jest sterowane poprzez mikronoktroler Arduino Uno.

7. Bibliografia

[1] S. Monk Programming, Arduino: Getting Started with Sketches Second Edition, McGraw Hill TAB, June 9, 2016.

[2] J. Blum, Exploring Arduino: Tools and Techniques for Engineering Wizardry, Wiley November 19 2019.

[3] J. Purdum Beginning C for Arduino, Second Edition: Learn C Programming for the Arduino, Apress July 1 2015.

[4] W. Oskay, E. Schlaepfer, Open Circuits: The Inner Beauty of Electronic Components, No Starch Press, 1 Novemver 2022.

[5] J. Lienig, J. Scheible, Fundamentals of Layout Design for Electronic Circuits, Springer, 20 March 2021.

[6] T. Denton, Automobile Electrical and Electronic Systems, 12 September 2017.

[7] N. Ford, M. Richards, Fundamentals of Software Architecture: An Engineering Approach. A Comprehensive Guide to Patterns, Characteristics and Best Practices, O'Reilly Media, 6 March 2020.

[8] M. Robert, Clean Code: A Handbook of Agile Software Craftsmanship, Financial times Prentice Hall, 14 September 2008.

[10] J. Culkin, E. Hagan, Learn Electronics with Arduino: An Illustrated Beginner's Guide to Physical Computing, O'Reilly Media, Inc. USA, 12 September 2017.

[11] P. Denise, D. Jenny, Electrical Engineering, Career Paths. Student's Book, Express Publishing, 1 October 2020.

[12] J. Bird, Electrical Circuit Theory and Technology, 22 March 2017.

[13] Texas Instruments, PFC8574 Remote 8-Bit I/O Expander for I^2C Bus March 2015.

[14] N. David, A. Chima, A. Ugochukwu, E. Obinna, Design of a Home Automation System Using Arduino, 6 June 2015.

[15] Arduino, [Online]. Available https://store.arduino.cc/products/arduino-uno-rev3/ [Data uzyskania dostępu: 19 Listopad 2022].