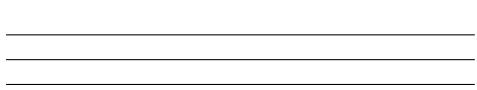


PROJEKT
WIZUALIZACJA DANYCH SENSORYCZNYCH

Wizualizacja danych z platformy czujników pojazdu

Michał Dołharz, 248943



Prowadzący:
dr inż. Bogdan Kreczmer
Katedra Cybernetyki i Robotyki
Wydziału Elektroniki
Politechniki Wrocławskiej

25 maja 2021

Spis treści

1 Opis projektu	1
1.1 Temat projektu	1
1.2 Etapy realizacji projektu	1
1.3 Oczekiwany rezultat	2
2 Harmonogram prac	3
2.1 Terminarz realizacji poszczególnych podcelów	3
2.2 Kamienie milowe	4
3 Dokumentacja i archiwizacja	4
4 Układ elektroniczny	5
4.1 Arduino Nano	5
4.2 Czujniki HC-SR04	5
4.3 Zasilanie	6
4.3.1 Tryb wizualizacji danych	6
4.3.2 Tryb czujników robota mobilnego	6
4.4 Zabezpieczenia zasilania	6
4.4.1 Odwrotna polaryzacja	6
4.4.2 Przepięcia	7
4.4.3 Filtracja zasilania	7
4.5 Schemat ideowy	7
4.6 Płytnica PCB	7
5 Aplikacja	11
5.1 Interfejs użytkownika	11
5.2 Komunikacja	12
5.2.1 Komunikacja z aplikacją	12
5.2.2 Protokół komunikacyjny OSOS	12
5.3 Testy	14
6 Podsumowanie	17
Spis rysunków	18
Bibliografia	18

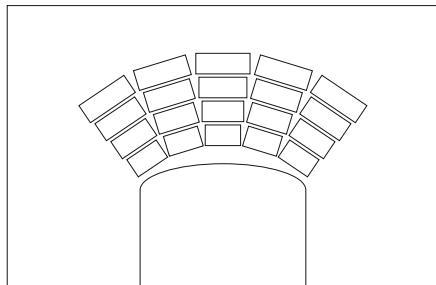
1 Opis projektu

1.1 Temat projektu

Celem projektu jest stworzenie aplikacji ostrzegającej o przeszkodach w otoczeniu robota mobilnego w oparciu o biblioteki Qt [5] języka C++, czujniki ultradźwiękowe HC-SR04 [2] oraz mikrokontroler Arduino Nano [1, 7, 8]. Wizualizacja ma przypominać wizualizację danych z czujników samochodu, wyświetlana na wewnętrznym, wbudowanym ekranie.

Robot mobilny o kryptonimie "Wędrowiec" będzie budowany w ramach projektu z kursu Roboty mobilne. W założeniu ma obsługiwać dwa tryby pracy: autonomiczny, gdzie po wykryciu przeszkody na drodze zmienia trajektorię ruchu oraz zdalnie sterowany z pilota budowanego również w ramach tego kursu. Integracja obu projektów przewidziana jest jako ostatni etap ich realizacji.

Projekt samej aplikacji będzie zależał od wyników testów czujników. W najbardziej optymistycznej wersji aplikacja będzie obsługiwała trzy widoki: czujniki przednie (pięć czujników jak na rysunku 1), czujniki tylne (analogicznie jak przednie, ale widok odwrócony w poziomie) oraz całe otoczenie pojazdu. Odpowiednie widoki będą wywoływanie automatycznie przy wykryciu przeszkody.



Rysunek 1: Rysunek koncepcyjny okna aplikacji

1.2 Etapy realizacji projektu

Każdy etap budowy aplikacji lub platformy czujników wymieniony poniżej będzie podlegał serii testów. Po ich analizie korygowane będą rzeczy takie jak ustawienie czujników, oprogramowanie czy kod aplikacji. Wyniki niektórych testów będą dodatkowo podstawą do dalszych konfiguracji i doboru optymalnych parametrów.

Podział na główne etapy prezentuje się następująco:

1. Zebranie literatury na tematy powiązane z tematem projektu.
2. Projekt układu elektronicznego (schemat ideowy). Projekt interfejsu graficznego.
3. Tworzenie aplikacji oraz testowanie czujników jeszcze bez platformy:
 - (a) Stworzenie wstępnej wersji aplikacji komunikującej się z mikrokontrolerem.
 - (b) Rozbudowa aplikacji o obsługę jednego czujnika. Wyświetlanie tekstowe danych.
 - (c) Rozbudowa aplikacji o obsługę pięciu czujników.
 - (d) Rozbudowa aplikacji o graficzną wizualizację danych.

- (e) Testowanie modułu czujnika VL53L1X [6] pod kątem użycia jako czujnika wspomagającego.
4. Budowa platformy czujników.
5. Rozbudowa aplikacji o obsługę przednich oraz tylnych czujników.
6. Integracja platformy z robotem mobilnym.
 - (a) Prawdopodobne poprawki oprogramowania.
 - (b) Możliwe poprawki ustawienia czujników.
 - (c) Testy gotowego urządzenia.

1.3 Oczekiwany rezultat

Finalna wersja aplikacja powinna charakteryzować się:

- poprawnym połączeniem z urządzeniem oraz poprawną komunikacją,
- poprawną interpretacją i graficzną wizualizacją danych pochodzących z czujników,
- obsługą dziesięciu niezależnych czujników HC-SR04,
- prostym, intuicyjnym i automatycznym interfejsem (ewentualnie z minimalną interakcją użytkownika) składającym się z trzech widoków: przedniego, tylnego, oraz całościowego.

Wykonana platforma czujników (bez integracji z robotem mobilnym) powinna charakteryzować się:

- zamocowaniem dziesięciu czujników HC-SR04,
- zasięgiem dobranym jako optymalny w trakcie testów czujników,
- możliwością przymocowania do robota mobilnego (a zatem dostosowanie do niego pod względem rozmieszczenia elementów),

Robot mobilny zintegrowany z platformą czujników powinien charakteryzować się:

- odmienną reakcją w zależności od trybu pracy i czujników wykrywających przeszkode,
- zredukowaną maksymalną prędkością w trakcie wizualizacji (kwestią bezpieczeństwa ze względu na komunikację przewodową).

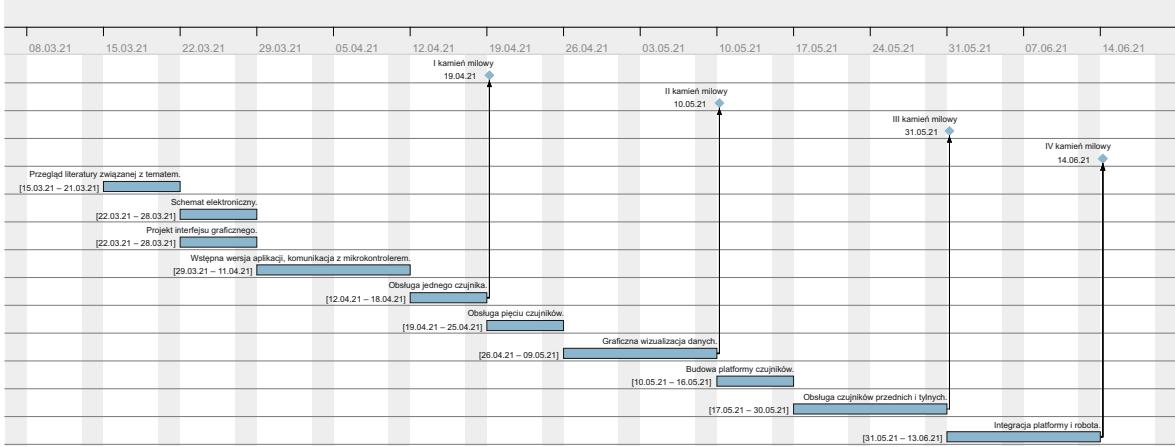
2 Harmonogram prac

2.1 Terminarz realizacji poszczególnych podcelów (z dokładnością do 1 tygodnia)

Każdy tydzień pracy w poniższym rozkładzie oprócz opisanych czynności zakłada testowanie wprowadzanych rozwiązań na bieżąco.

Harmonogram prac w rozkładzie tygodniowym prezentuje się następująco (podkreślone terminy oznaczają kolejne terminy oddania raportów z postępu prac nad projektem):

- 22 marca 2020 – zakończenie przeglądu materiałów związanych z tematem.
- 29 marca 2020 – zaprojektowanie schematu układu elektronicznego z uwzględnieniem wersji z jednym czujnikiem, pięcioma oraz dziesięcioma oraz zaprojektowanie interfejsu graficznego.
- 12 kwietnia 2020 – wstępna wersja aplikacji, komunikacja z mikrokontrolerem.
- 19 kwietnia 2020 – rozbudowanie aplikacji o obsługę jednego czujnika (tekstowe wyświetlanie danych).
- 26 kwietnia 2020 – rozbudowanie aplikacji o obsługę pięciu czujników (bez uwzględniania ich rozmieszczenia na platformie lub robocie).
- 4 maja 2020 – praca nad rozbudowaniem aplikacji o graficzną wizualizację danych (wersja uproszczona graficznie).
- 10 maja 2020 – stworzenie graficznej wizualizacji danych z uwzględnieniem przyszzej implementacji dodatkowych okien widoków oraz rozmieszczenia czujników na platformie.
- 17 maja 2020 – zbudowanie platformy czujników z uwzględnieniem przyszłego mocowania na robocie.
- 24 maja 2020 – praca nad rozbudowaniem aplikacji o obsługę czujników przednich i tylnych (łącznie dziesięć czujników).
- 31 maja 2020 – zaimplementowanie obsługi wszystkich dziesięciu czujników oraz trzech okien widoków i ich automatycznego wywoływania.
- 7 czerwca 2020 – praca nad integracją platformy i robota. Fizyczne mocowanie platformy do robota. Przeprojektowanie (połączenie) schematów ideowych.
- 14 czerwca 2020 – Przeprogramowanie mikrokontrolerów robota i platformy do komunikacji i współpracy. Testy gotowego urządzenia w obu trybach robota.



Rysunek 2: Diagram Gannta wykonywanych prac

2.2 Kamienie milowe

W harmonogramie prac nad projektem wyszczególnione są cztery kamienie milowe:

1. Pierwszy kamień milowy zakłada stworzenie aplikacji zdolnej do komunikacji z mikrokontrolerem oraz tekstowym wyświetlaniem danych pochodzących z jednego czujnika.
2. Drugi kamień milowy zakłada rozbudowanie aplikacji do obsługi pięciu czujników (nieprzymocowanych do platformy) i graficznego wyświetlania danych. Na tym etapie gotowe jest okno widoku przednich czujników.
3. Trzeci kamień milowy zakłada rozbudowę aplikacji o usługę czujników tylnych oraz budowę właściwej platformy czujników. Na tym etapie aplikacja jest gotowa.
4. Czwarty kamień milowy zakłada połączenie robota mobilnego i platformy czujników. Robot jest odpowiednio przeprogramowany, aby jego reakcja zależała od odpowiednich czujników.

3 Dokumentacja i archiwizacja

Projekt jest archiwizowany w zdalnym repozytorium. Bezpośrednio w głównym folderze znajduje się kod aplikacji Qt. W podfolderach można znaleźć również kod oprogramowania platformy czujników, dokumentację projektu stworzoną przy użyciu generatora Doxygen oraz kolejne raporty z realizacji projektu:

- Repozytorium:
<https://github.com/Repti993/Ambient-Sensors-Platform>
- Oprogramowaniem platformy:
https://github.com/Repti993/Ambient-Sensors-Platform/tree/main/platform_software
- Dokumentacja:
<https://github.com/Repti993/Ambient-Sensors-Platform/tree/main/doc/html>
- Dokumentacja online:
<http://panamint.ict.pwr.wroc.pl/~mdolharz/wds/>

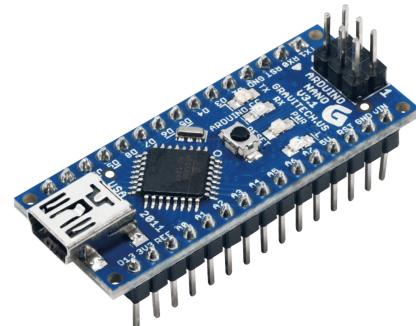
4 Układ elektroniczny

4.1 Arduino Nano

Sercem układu jest płytka Arduino Nano v3.0 (klon) z mikrokontrolerem ATmega328 (rys. 3). Odpowiada za przetwarzanie danych pochodzących z sensorów i komunikację z aplikacją.

Specyfikacja płytki:

- 14 portów I/O,
- 6 kanałów PWM,
- 8 przetworników ADC,
- 32 kB pamięci Flash,
- interfejsy komunikacyjne: SPI, UART i I2C,
- maksymalna częstotliwość zegara 16 MHz,
- zasilana napięciem 5 V lub od 7 V do 12 V,
- na płytce jest wgrany bootloader z Arduino Nano.



Rysunek 3: Arduino Nano v3.0

Płytki nie jest więc szczytem technologii, ale jej dużą zaletą jest niewielki rozmiar (ok. 43×18 mm). Dodatkowo nie posiada wielu pinów, co również czyni ją odpowiednią do niniejszego projektu.

4.2 Czujniki HC-SR04

Platforma składa się z dziesięciu czujników HC-SR04 (rys. 4). Pięć czujników o numerach od 0 do 4 znajdują się z przodu, a kolejne pięć, o numerach od 5 do 9, z tyłu platformy.

Specyfikacja czujnika:

- zasięg od 2 cm do 4 m,
- kąt pomiaru 15° ,
- częstotliwość pracy 40 Hz,
- długość impulsu Trig $10\ \mu s$,
- pobierany prąd 15 mm,
- zasilanie 5 V.



Rysunek 4: Sensor HC-SR04

Piny Trig każdego z sensorów są ze sobą połączone, natomiast piny Echo są podpięte pod osobne piny Arduino. W ten sposób sygnał wyzwalający aktywuje każdy czujnik, ale w danym czasie sprawdza sygnał zwrotny tylko na odpowiednim pinie Arduino. Dzięki takiemu rozwiązaniu układ jest "półmultipleksowany" i pozwala zaszczerdzić kilka pinów mikrokontrolera.

4.3 Zasilanie

4.3.1 Tryb wizualizacji danych

W trybie wizualizacji danych sensorycznych układ można zasilać na dwa sposoby.

Pierwszy sposób zakłada podłączenie platformy do komputera poprzez gniazdo micro USB. W takim wypadku zasilanie pobierane jest z komputera i żaden zewnętrzny zasilacz nie jest potrzebny.

Drugi sposób zakłada podłączenie platformy do komputera poprzez złącze ARK podłączone do pinów UART mikrokontrolera. Zasilanie wtedy nie jest pobierane z komputera. Do dyspozycji są dwa złącza do doprowadzenia zasilania: DC 2,1/5,5 oraz ARK. Złącze DC zostało przygotowane z myślą o zasilaniu stacjonarnym, a więc w trybie wizualizacji danych. Złącze ARK zostało przygotowane z myślą o pobieraniu zasilania z robota mobilnego. W praktyce po przygotowaniu odpowiednich przewodów wybór złącza nie ma większego znaczenia. Jedyna różnica polega na tym, że zastosowano złącze DC odcinające zasilanie "bateryjne" (w tym przypadku zasilanie ze złącza ARK) w przypadku podłączenia wtyczki DC.

Wymagane napięcie zasilania to 5 V. Projekt zakłada wykorzystanie zasilacza 5 V lub pobieranie stabilizowanego napięcia z robota.

4.3.2 Tryb czujników robota mobilnego

W trybie czujników robota mobilnego niezalecane jest przewodowe podłączenie platformy do komputera ze względów bezpieczeństwa, a więc zasilanie nie może być pobierane z komputera¹.

Platforma jest wyposażona w złącza DC 2,1/5,5 i ARK. Do dowolnego z nich można podłączyć stabilizowane do 5 V zasilanie z płytki robota (zasilanie platformy nie jest stabilizowane). Podłączenie zasilania do gniazda DC odcina zasilanie ze złącza ARK.

Niestety przy projektowaniu płytki robota zapomniano o aspekcie połączenia projektów, w związku z czym konieczne było dorobienie złącza ARK do powiązania łączności UART. Nie zostało również utworzone złącze do podłączenia platformy do zasilania. W tym przypadku rozwiązaniem jest wykorzystanie złącza robota do podłączenia czujnika HC-SR04 (w wersji bez platformy robot korzysta z jednego czujnika HC-SR04). Jeżeli robot korzysta z platformy, to czujnik robota jest odłączony i można wykorzystać piny zasilania. Oczywiście można także podłączyć inne, zewnętrzne zasilanie 5 V.

4.4 Zabezpieczenia zasilania

Niezależnie od sposobu podłączenia zasilania, podlega ono odpowiednim zabezpieczeniom i filtrowaniu, nie podlega natomiast stabilizacji, w związku z czym platforma musi być zasilana napięciem 5 V.

4.4.1 Odwrotna polaryzacja

Do zabezpieczenia układu przed odwrotną polaryzacją [3] jest użyty tranzystor unipolarny N-MOSFET IRF520 [4]. Tranzystor ten cechuje się prądem drenu do 9,7 A przy napięciu bramka-źródło 10 V.

¹Wyjątkiem może być tryb zdalnego sterowania robotem z ograniczeniem maksymalnej prędkości do minimum i zachowaniem szczególnej ostrożności (funkcję ograniczania prędkości będzie posiadał pilot).

Przy poprawnej polaryzacji zasilania tranzystor otworzy się i będzie przewodził prąd. Jeżeli zasilanie zostanie podłączone odwrotnie, tranzystor pozostanie zamknięty i nie będzie przewodził prądu. Został wpięty odwrotnie, aby jego wewnętrzna dioda nie przewodziła prądu przy odwrotnej polaryzacji.

4.4.2 Przepięcia

W celu ochrony układu przed zbyt dużym natężeniem prądu wykorzystany jest bezpiecznik topikowy 0,5 A.

4.4.3 Filtracja zasilania

Do filtrowania zasilania jest wykorzystywana popularna para kondensatorów ceramicznego 100 nF wraz z elektrolitycznym 10 μ F.

4.5 Schemat ideowy

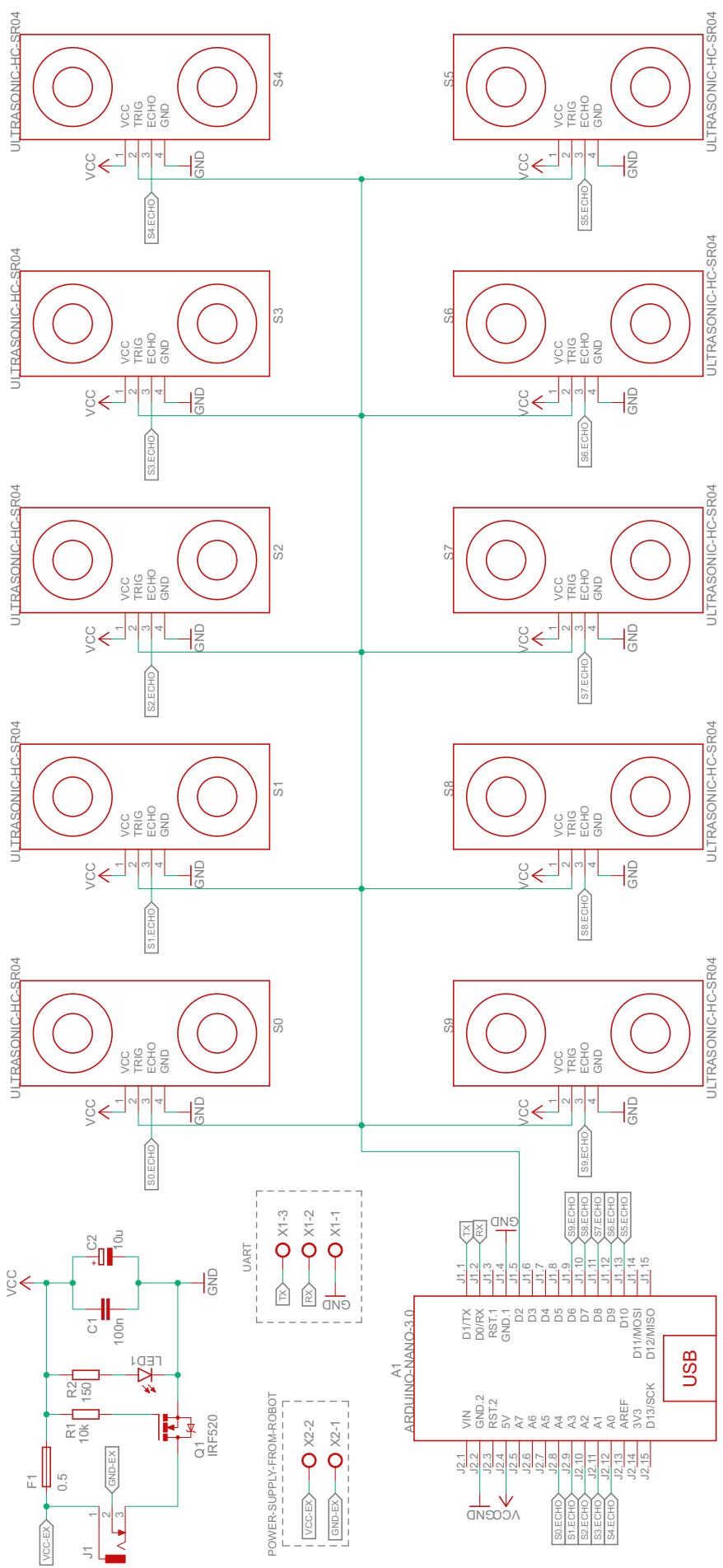
Schemat ideowy układu prezentuje się na rysunku 5. Zostało uwzględnione zasilanie, zabezpieczenia zasilania, komunikacja UART (poprzez piny) oraz, oczywiście, układ sensorów i mikrokontrolera.

4.6 Płytki PCB

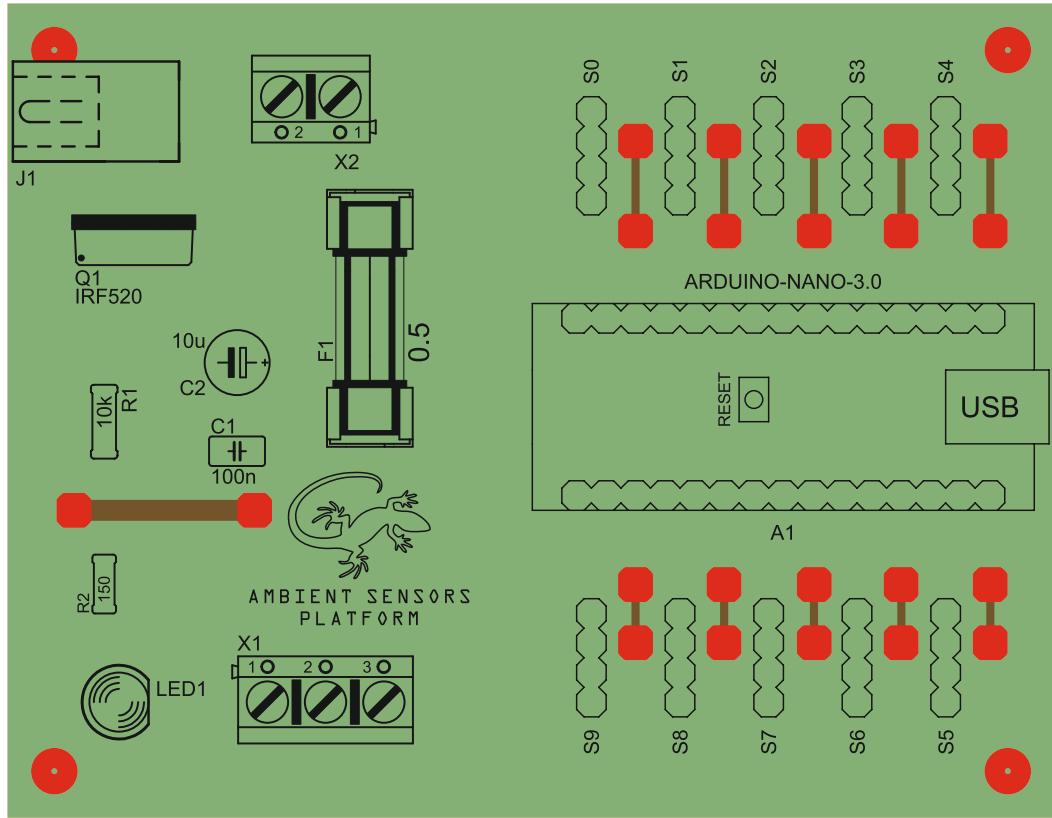
W ramach realizacji projektu zostanie wytrawiona dedykowana płytka PCB. Rysunek 6 prezentuje projekt górnej warstwy ścieżek i oznaczeń, a rysunek 7 ten sam widok wytrawionej już płytki. Analogicznie, rysunek 8 przedstawia projekt dolnej warstwy ścieżek, a rysunek 9 ten sam widok wytrawionej płytki.

Czerwone oznaczenia projektowe na górnjej warstwie to właśnie kilka ścieżek połączeń. Wyjaśnienia mogą wymagać pola lutownicze na obu końcach każdej z nich. Otóż płytka została wytrawiona własnoręcznie, w związku z czym wymagane było wywiercenie otworów na nieizolowane przewody łączące ścieżki po obu stronach płytki, a to z kolei wymagało pól lutowniczych. W ten sposób powstały przelotki.

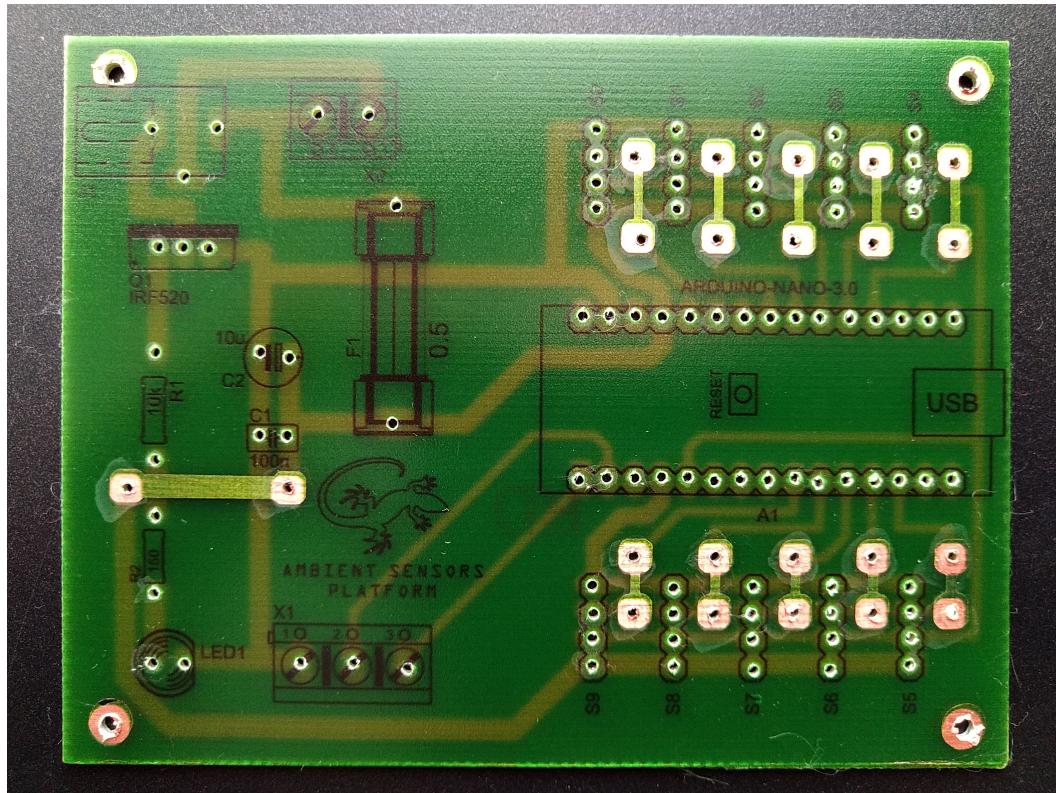
Czerwone okręgi w narożnikach płytka to jedynie znaczniki do wywiercenia otworów na śruby mocujące płytke. Pola lutownicze są nieco większe od standardowych ze względu na ręczne wiercenie otworów oraz rozmiar wiertła.



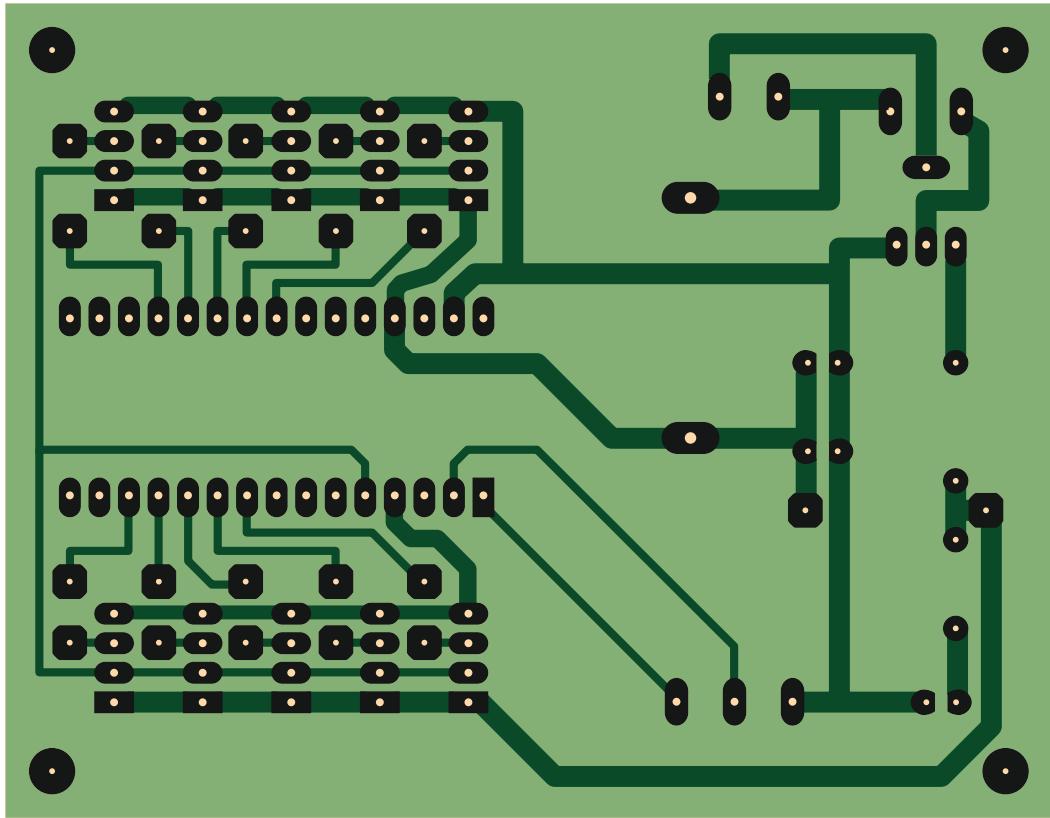
Rysunek 5: Schemat ideowy układu



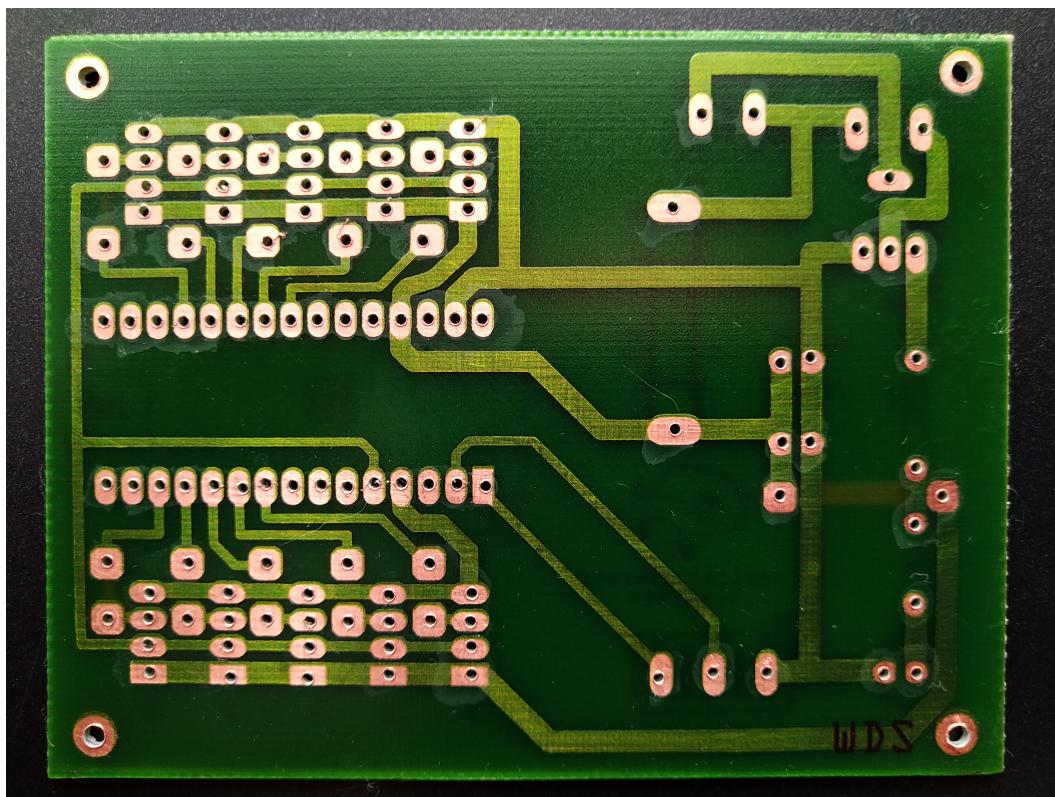
Rysunek 6: Projekt płytki PCB, widok oznaczeń i wierzchniej warstwy ścieżek



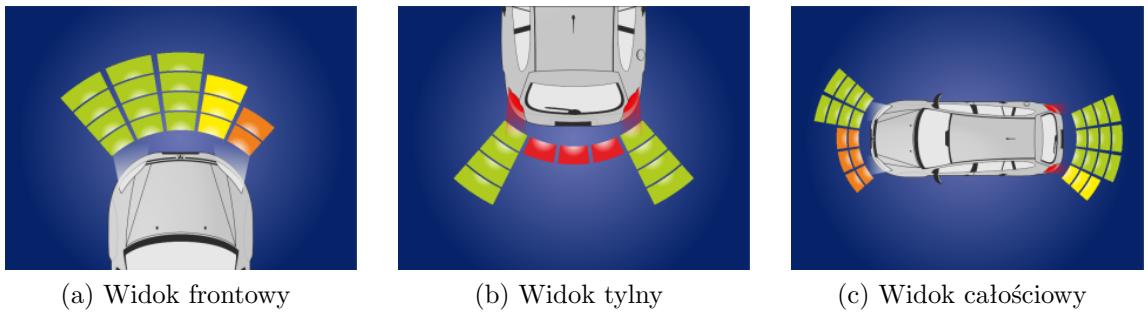
Rysunek 7: Płytki PCB, widok oznaczeń i wierzchniej warstwy ścieżek



Rysunek 8: Projekt płytki PCB, widok dolnej warstwy ścieżek



Rysunek 9: Płytnka PCB, widok dolnej warstwy ścieżek



Rysunek 10: Przykładowe widoki programu

5 Aplikacja

5.1 Interfejs użytkownika

Parkując samochód kierowca trzyma obie ręce na kierownicy i patrzy w lusterka lub rozgląda się przez okna samochodu badając otoczenie. Informacje wyświetlane na ekranie mają służyć pomocą, muszą więc być automatyczne i intuicyjne.

Interfejs składa się z trzech okien, nazywanych widokami. Jest w pełni automatyczny, co oznacza, że odpowiednie widoki są włączane w zależności od czujników, które wykryją przeszkodę. Jedynie przy starcie aplikacji użytkownik musi wybrać port COM, przez który będzie odbywała się komunikacja z mikrokontrolerem (okno komunikacji jest przedstawione w podrozdziale 5.2.1).

Czujniki są podzielone na przednie (rys. 10a) i tylne (rys. 10b). Wystarczy jeden czujnik z grupy, aby aktywować odpowiadający mu widok. Jeżeli przeszkody są wykrywane zarówno przez czujniki przednie, jak i czujniki tylne, wtedy włącza się widok całościowy (rys. 10c).

Zielony wskaźnik oznacza brak wykrytej przeszkody. Wskaźniki żółty, pomarańczowy, czerwony oraz brak wskaźnika oznaczają wykryte przeszkody w programowo ustalonych zakresach (więcej informacji o zakresach znajduje się w podrozdziale 5.3). Takie rozwiązanie zapewnia kierowcy możliwość natychmiastowego przyswojenia informacji wyświetlanych na ekranie.

W zamyśle aplikacja nie powinna wyświetlać żadnego okna, gdy żaden z czujników nie wykrywa przeszkody, umożliwiając działanie innych funkcjonalności ekranu samochodowego. Projekt ten nie zawiera jednak innych funkcjonalności, w związku z czym w trakcie bezczynności jest wyświetlane samo tło, bez żadnego widoku.

Zostało wykonane nagranie prezentujące działanie aplikacji wraz z platformą czujników. Niestety program do nagrywania nie uwzględniał rozwijanej listy z dostępnymi urządzeniami w czwartej sekundzie. W rzeczywistości lista rozwinęła się i zawierała urządzenia wypisane w okienku *Logs*. Adres strony z nagraniem znajduje się poniżej.

Prezentacja działania aplikacji i platformy czujników:
https://youtu.be/pFAddx3wh_E

5.2 Komunikacja

5.2.1 Komunikacja z aplikacją

Gdyby aplikacja była wgrana do komputera samochodowego lub innego urządzenia obsługującego wyświetlacz samochodowy, konfiguracja komunikacji mogłaby być zaprogramowana na stałe. W przypadku podłączenia urządzenia do komputera wielofunkcyjnego niezbędna jest możliwość wyboru portu, do którego podłączona jest platforma.

Aplikacja pozwala na wyszukiwanie dostępnych portów COM, a także do łączenia i rozłączania się z nimi. Służą do tego trzy przyciski:

- **Szukaj** – wyszukuje dostępne urządzenia na portach COM. Znalezione urządzenia są wyświetlane w rozwijanej zakładce obok.
- **Połącz** – powoduje próbę połączenia z urządzeniem wybranym w zakładce.
- **Rozłącz** – zamyka połączenie z urządzeniem.

Informacje wyświetlane są w dwóch oknach tekstowych:

- **Logi** – informacje dotyczące działania komunikacji, jak np. udane lub nieudane połączenie, udane parsowanie ramki, itd.
- **Komunikacja** – informacje użytkowe, czyli jaki jest nowy status i na którym czujniku.

Okno programu zawierające wymienione funkcjonalności prezentuje rysunek 12.

5.2.2 Protokół komunikacyjny OSOS

Platforma czujników komunikuje się z aplikacją za pomocą UART. Gdy dane pochodzące z czujnika wskazują na zmianę statusu, mikrokontroler wysyła odpowiedni komunikat. Dane wysyłane są w postaci ramki, której budowa znajduje się na rysunku 11. Komunikacja jest znakowa. Poszczególne dane są rozdzielane znakami spacji.

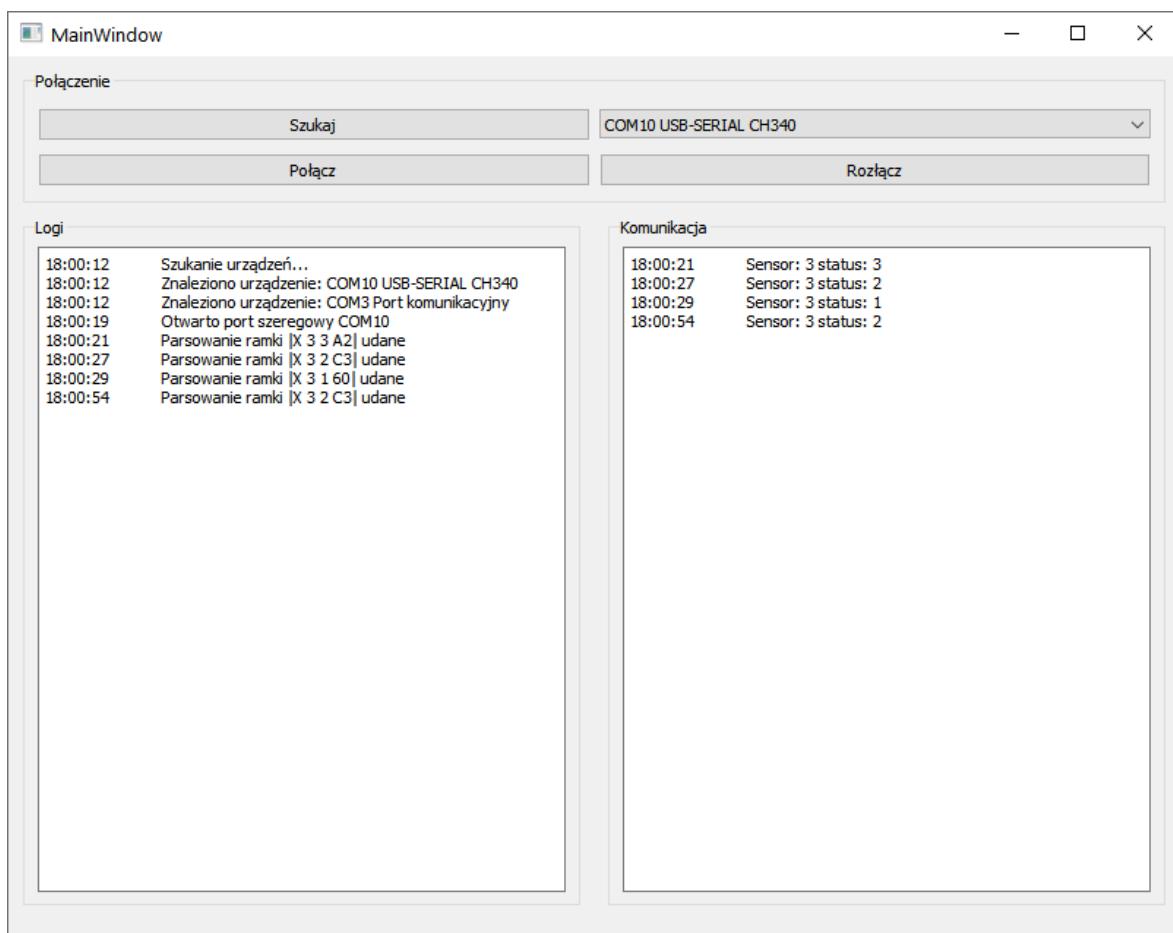


Rysunek 11: Ramka protokołu komunikacji OSOS

Budowa ramki wygląda następująco:

1. Ramka rozpoczyna się od znaku 'X'.
2. Numer sensora, którego komunikat dotyczy (Sn). Wartości od 0 do 9 w zapisie dziesiętnym.
3. Numer statusu (St). Wartości od 0 do 4 w zapisie dziesiętnym.
4. Suma kontrolna CRC8 w zapisie heksadecymalnym, zajmująca dwa znaki w ramce.

Ze względu na zawartość pojedynczej ramki danych, protokół otrzymał nazwę OSOS (*One Sensor One Status*).



Rysunek 12: Okno aplikacji.

Tablica 1: Parametry układu

Status	Strefa	Zakres od [cm]	Zakres do [cm]
0	zielona	20	–
1	żółta	15	20
2	pomarańczowa	10	15
3	czerwona	5	10
4	brak bloku	0	5

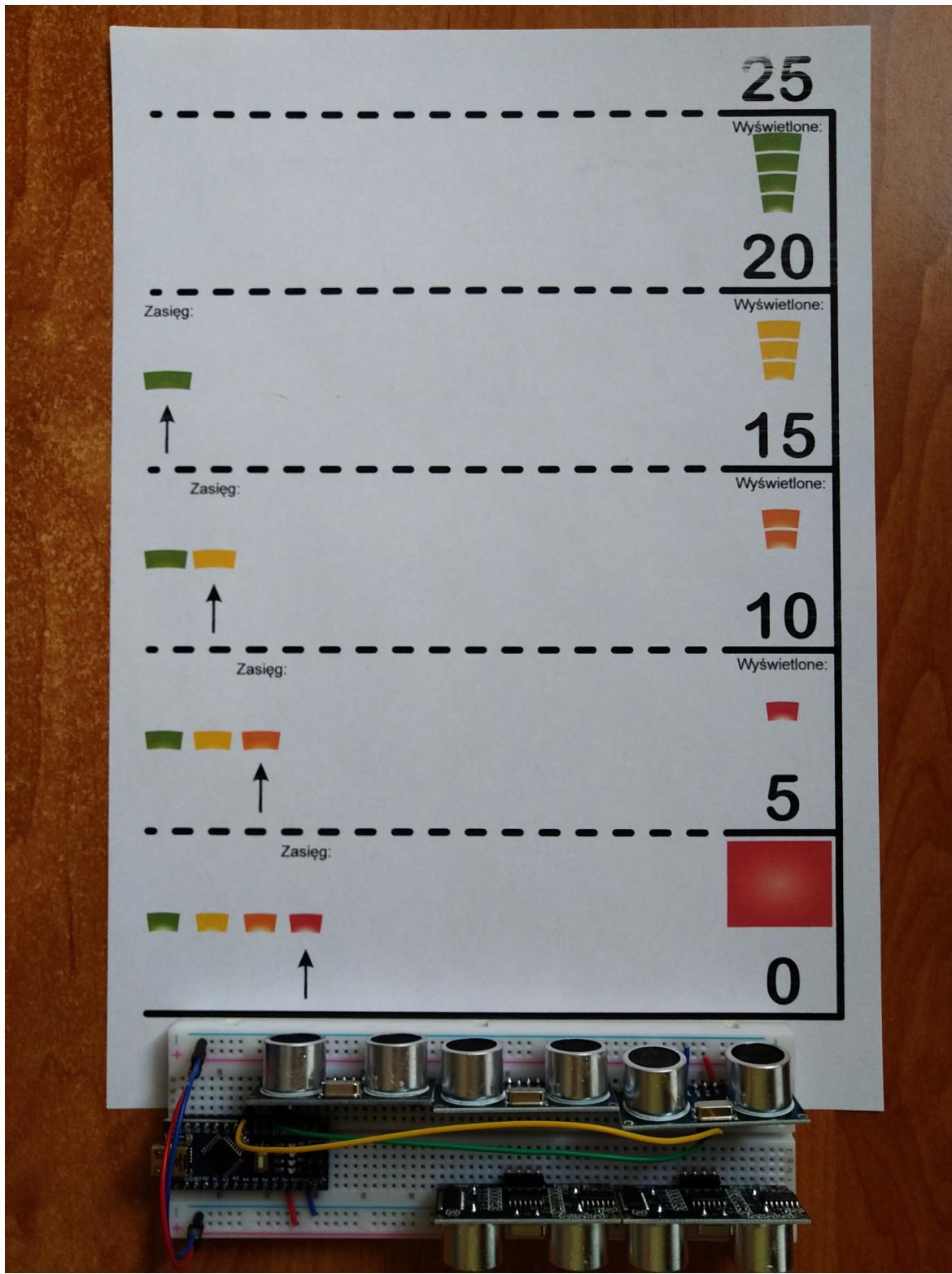
5.3 Testy

Do sprawdzania poprawności pomiarów czujnika używana jest plansza pomiarowa z zaznaczonymi odległościami (strefami) i odpowiadającymi im kolorami statusów. Parametry, przy których testowane są czujniki, tzn. konfiguracja poszczególnych statusów czujnika znajduje się w tabeli 1.

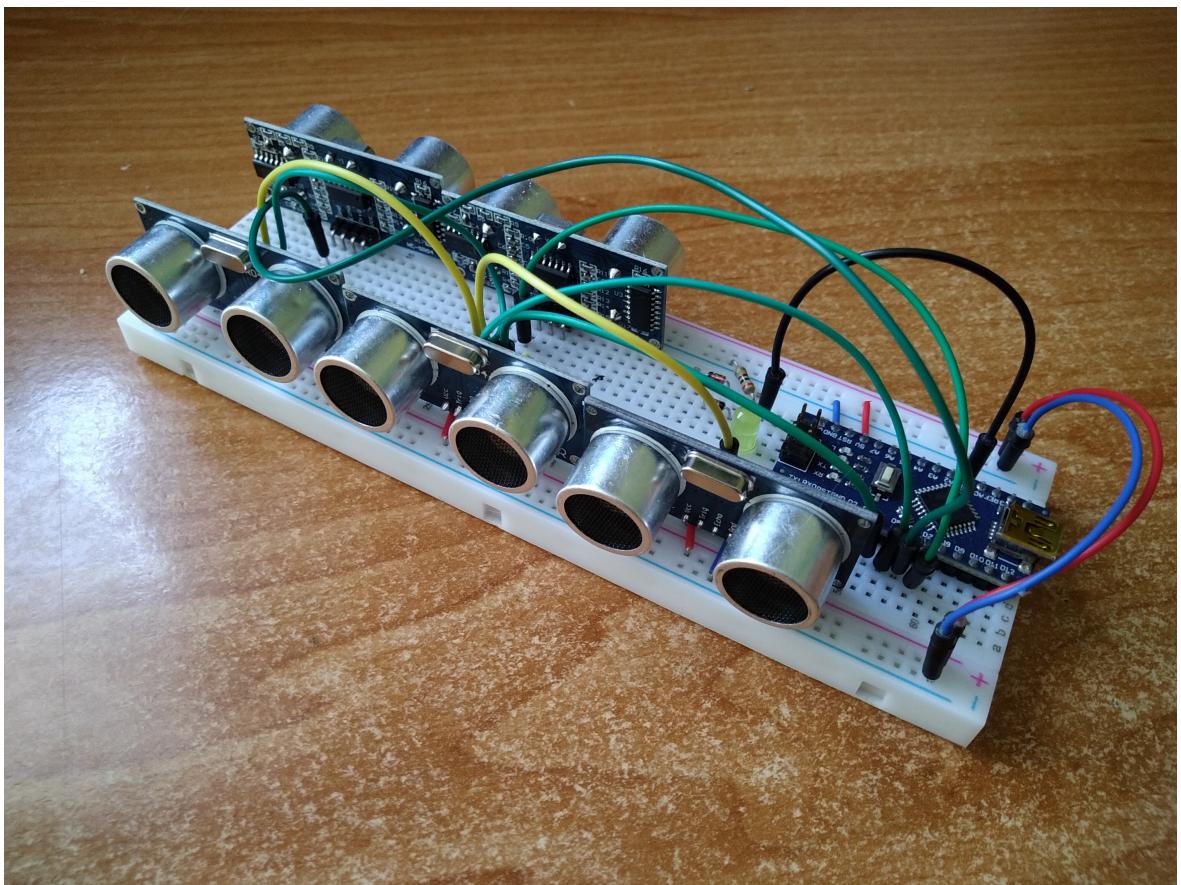
Plansza pomiarowa przedstawiona na rysunku 13 daje dwie możliwości interpretacji informacji o umieszczeniu przeszkody, chociaż tak naprawdę obie mówią o tym samym. Z prawej strony znajduje się informacja co zostanie wyświetcone w aplikacji, jeżeli w danej strefie znajdzie się przeszkoda. Z lewej strony znajduje się informacja o zasięgu, tzn. pierwsza strefa od góry, na której nie znajduje się przeszkoda, informuje strzałką o wyświetlanych blokach.

Wyświetlanie danych w aplikacji ma być intuicyjne, w związku z czym jest inspirowana działaniem światła. Oznacza to, że miejsce, w którym wykryto przeszkodę, jest już niedostępne (światło odbija się od przeszkody, nie wpada w nią). Innymi słowy, wyświetlany blok oznacza, że żadna przeszkoda się tam nie znajduje. Dopiero gdy zniknie, oznacza to, że w tym miejscu znalazła się przeszkoda. Przykład: aplikacja wyświetla dwa pomarańczowe bloki. Oznacza to, że przeszkoda znajduje się w strefie pomarańczowej (status czujnika 2), czyli w miejscu, gdzie znajdowałby się kolejny blok, a nie w miejscu, gdzie wyświetlany jest drugi pomarańczowy blok.

Testowany układ jest zmontowany na płytce stykowej, początkowo z jednym sensorem, następnie z pięcioma, aż w końcu z dziesięcioma. Testowany układ pięciu czujników prezentuje się na rysunku 14



Rysunek 13: Układ z planszą pomiarową



Rysunek 14: Jeden z układów czujników, na których odbywały się testy

6 Podsumowanie

Projekt okazał się świetną okazją do zapoznania się z zestawem bibliotek Qt. Początki nie były łatwe, ale po wstępny zapoznaniu się z zawartością i ustaleniu, które klasy będą odpowiednie do stworzenia graficznego interfejsu, praca z Qt okazała się wcale nie być tak trudną. Z pewnością Qt przyda się przy innych projektach w przyszłości.

Projekt pozwolił również poznać bliżej czujniki HC-SR04 i przebadać ich możliwości. Nie są rewelacyjne pod względem dokładności, lecz są wystarczająco dokładne do wykorzystania w niektórych projektach. Z pewnością projekt pomógł pokonać pewną "barierę", którą stanowiła wizualizacja działania czujników na komputerze.

W trakcie projektu powstawało i upadało wiele różnych pomysłów. Najważniejszy z nich to czujnik wspomagający VL53L1. Od początku zakładano jednak, że jest to koncepcja niepewna i wymaga testowania, na które nie stało się czasu. Drugim ważnym pomysłem była zmiana tła z niebieskiego na czerwony w wyniku pojawienia się piątego statusu (przeszkoda najbliżej pojazdu). Miało to podkreślać, że przeszkoda jest już bardzo blisko. Zrezygnowano z tego pomysłu, ponieważ zmiana tła nie wskazywała w żaden sposób, w którym miejscu znajduje się przeszkoda, a w prawdziwej sytuacji mogłyby wprowadzić niepotrzebne zamieszanie. Znacznie lepszym pomysłem byłoby zmodyfikowanie układu o możliwość dźwiękowego sygnalizowania przeszkody. Rozważano również takie pomysły jak bezprzewodowa komunikacja z wprowadzeniem dodatkowego mikrokontrolera.

Platfroma czujników zostanie teraz zintegrowana z robotem mobilnym według początkowych ustaleń. Wizualizacja danych z czujników zostanie odpuściżona na rzecz odmiennej reakcji robota na każdy z czujników. Być może w przyszłości układ zostanie rozbudowany o bezprzewodową komunikację, aby wizualizacja była nadal możliwa. Komunikacja nadal będzie się odbywać protokołem OSOS.

Podsumowując, projekt pozwolił na rozwinięcie umiejętności w szerokim zakresie, począwszy od programowania, poprzez tworzenie protokołu komunikacyjnego, oprogramowanie takie jak doxygen oraz git, projektowanie układu, aż do zaprojektowania, wytrawienia i zlutowania płytki PCB. Z pewnością był to jeden z bardziej wartościowych projektów i z równą pewnością był wart poświęconego mu czasu.

Spis rysunków

1	Rysunek koncepcyjny okna aplikacji	1
2	Diagram Gantta wykonywanych prac	4
3	Arduino Nano v3.0	5
4	Sensor HC-SR04	5
5	Schemat ideowy układu	8
6	Projekt płytki PCB, widok oznaczeń i wierzchniej warstwy ścieżek	9
7	Płytnka PCB, widok oznaczeń i wierzchniej warstwy ścieżek	9
8	Projekt płytki PCB, widok dolnej warstwy ścieżek	10
9	Płytnka PCB, widok dolnej warstwy ścieżek	10
10	Przykładowe widoki programu	11
11	Ramka protokołu komunikacji OSOS	12
12	Okno aplikacji.	13
13	Układ z planszą pomiarową	15
14	Jeden z układów czujników, na których odbywały się testy	16

Bibliografia

- [1] Arduino. [Arduino Nano \(V3.0\) User Manual](#).
- [2] Elecfreaks. [HC-SR04 Datasheet](#).
- [3] elektroda.pl. [Zabezpieczenie przed odwróceniem polaryzacji zasilania #24](#) edu elektroda.pl. URL: <https://youtu.be/Yy-VR0PhQaQ>.
- [4] International Rectifier. [IRF520N Datasheet](#).
- [5] Qt Development Frameworks. [Dokumentacja Qt](#). URL: <https://doc.qt.io>.
- [6] STMicroelectronics. [VL53L1X Datasheet](#).
- [7] Szymański D. [Kurs Arduino. Poziom I.](#) 2021. URL: www.forbot.pl/blog/kurs-arduino-podstawy-programowania-spis-tresci-kursu-id5290.
- [8] Szymański D. [Kurs Arduino. Poziom II.](#) 2021. URL: www.forbot.pl/blog/kurs-arduino-ii-wstep-spis-tresci-id15494.