Politechnika Wrocławska
Wydział Elektroniki Mikrosystemów i fotoniki

KIERUNEK: Mechatronika

PRACA DYPLOMOWA INŻYNIERSKA

Sterowanie stanowiskiem
do pomiarów efektu Halla
w materiałach $A^{III}B^{V}$

Control of the
Hall effect measurement system
for $A^{III}B^{V}$ materials

AUTOR:
Bartłomiej Polkowski

PROMOTOR:
dr inż. Damian Radziewicz

OCENA PRACY I PODPIS PROMOTORA:

Wrocław 2019

1. Wstęp

1.2. Streszczenie/Abstract

Przedmiotem pracy było zaprojektowanie, wykonanie oraz testowania urządzenia pomiarowego metodą Halla w materiałach AIIIBV.

W pierwszej kolejności został zbudowany system kontroli położenia magnesów stałych nad próbką pomiarową. W tym celu został opracowany system pozycjonowania czujników w obudowie urządzenia.

Następnie została zaprojektowana oraz wykonana płytka obwodów drukowanych. Ma ona na celu ograniczyć liczbę przewodów wyprowadzonych z mikrokontrolera, ułatwić pomiar próbki oraz sterować położeniem magnesu nad próbką.

Oprócz wykonanej płytki został napisany kod sterujący w języku C mający na celu ustawiać parametry pomiaru. Są to wyprowadzenia na płytce pomiarowej multimetrów oraz położenie magnesu nad próbką.

Pond to całość komunikuje się z komputerem w którym znajduje się aplikacja ułatwiająca sterowanie oraz analizę otrzymanych wyników

1.3. Wprowadzenie

Projekt urządzenia jest rozwinięciem pracy inżynierskiej Gabriela Ceballos pt. „Stanowisko do pomiarów efektu Halla w materiałach AIIIBV" realizowaną w zakładzie Mikroelektroniki i Nanotechnologii Wydziału Elektroniki Mikrosystemów i Fotoniki w 2015 roku. Przedmiotem jest projekt urządzenia do badania właściwości elektrycznych materiałów półprzewodnikowych.

Edwin Hall, doktorant Uniwersytetu imienia Johnsa Hopkinsa w Baltimore, starał się w 1879 roku potwierdzić teorię szerzoną przez Jamesa Clerka Maxwella. Głosiła ona, że siła przesuwająca przewodnik w polu magnetycznym nie ma wpływu na uporządkowany przepływ elektronów. Oddziałuje natomiast na przenoszący go prąd. W swoich badaniach chciał udowodnić że pod wpływem przyłożenia zewnętrznego pola magnetycznego zadziała siła Lorentza odchylająca przepływ ładunków w przewodniku. Eksperyment wcześniej wykonywał promotor amerykańskiego uczonego, Henry Augustus Rowland, jednak zakończył się niepowodzeniem, ze względu na zastosowanie zbyt grubych próbek. Hall użył zatem cienkiej folii ze złota. Przepuszczając przez nią prąd, przyłożywszy prostopadle do powierzchni folii zewnętrzne pole magnetyczne, spodziewał się zakrzywienia toru nośników. Co za tym idzie, oczekiwał wystąpienia różnicy potencjałów na brzegach folii [1]. W rezultacie zaowocowało to rzeczywistym zwiększeniem się gęstości elektronów po jednej ze stron przewodnika. Uzyskane napięcie zostało nazwane nazwiskiem naukowca, a jego wartość jest określana za pomocą równania (1.1).

$$U_H = \frac{I \cdot B}{d} \cdot R_H$$

gdzi$: U_H - napięcie\ Halla, I - natężenie\ prądu, B - indukcja\ magnetyczna, d - grubość\ warstwy, R_H - stała\ H$

$$R_H = \frac{E_y}{j_x \cdot B} = \frac{1}{n \cdot q}$$

gdzi$: E_y - natężenie\ pola\ elektrycznego\ w\ osi\ Y, j_x - gęstość\ prądu\ w\ osi\ X, n - koncentracja\ nośników, q - ład$

$$\mu = \frac{1}{\rho \cdot q \cdot n}$$

gdzi$: \mu - ruchliwość\ nośników, \rho - rezystywność\ warstwy$

Efektem wyprowadzonych równań jest stała Halla. Jest ona wypadkową natężenia pola elektrycznego E_y w osi Y w kierunku prostopadłym do kierunku działania indukcji magnetycznej oraz gęstości prądu j_x w osi X. Może być również wyznaczona z odwrotności ilorazu ładunku elementarnego oraz koncentracji nośników. Traktując ją jako stałą materiałową może ona posłużyć do wyznaczenia koncentracji nośników.

Metoda pomiarowa van der Pauwa zakłada pomiar właściwości elektrycznych na specjalnie dobranym podłożu o określonych wymiarach geometrycznych. Próbka posiada cztery symetrycznie ułożone wyprowadzenia. Pozwala to na pomiar w dwóch konfiguracjach, co pozwala na dokładniejsze określenie właściwości badanego materiału. Skupia się ona na pomiarze wartości rezystancji powierzchniowej badanej próbki.

1.4. Stan projektu

Szkielet obudowy urządzenia został zbudowany tak, aby wewnątrz mogły znajdować się wszystkie niezbędny elementy sytemu pomiarowego (rysunek 1). Do tego celu użyto piętnastu aluminiowych profili. Jest on zbudowany w formie prostopadłościanu, co pozwala na łatwe ekranowanie próbki od zewnętrznego oświetlenia podczas pomiaru. W centrum urządzenia znajduje się układ dwóch magnesów stałych o przeciwnie ułożonej biegunowości. Zamontowany jest on na wale osadzonym w dwóch łożyskach kulkowych, przymocowanych do dwóch dodatkowych belek pośrodku konstrukcji. Ze względu na konieczność dokładnego pozycjonowania ustawienia magnesu nad próbką został wybrany silnik krokowy. Pozwala on na precyzyjny obrót magnesów. Silnik krokowy posiada również odpowiednią moc do przemieszczania magnesów. Wewnątrz zostały umieszczone specjalne prowadnice w których umieszczony jest mikrokontroler oraz dodatkowa płytka obwodów drukowanych. Dzięki temu można w szybki sposób zdemontować mikrokontroler z urządzenia i wykorzystać do innych prac.

Rysunek 1. Aluminiowy szkielet konstrukcji.

1.5. Założenia projektowe

Założeniem projektu jest zaprojektowanie, zbudowanie oraz uruchomienie konstrukcji automatycznego stanowiska do pomiaru parametrów elektrycznych materiałów AIIIBV. Aparatura ma za

zadanie mierzyć podstawowe parametry elektryczne takie jak: ruchliwość nośników, koncentracje swobodnych nośników oraz wartości napięcia oraz prądu. Do wyznaczenia tych wartości w systemie będzie wykorzystane zjawisko Halla. Aby zjawisko mogło zaistnieć, konieczny jest wymuszony przepływ prądu elektrycznego w badanej warstwie oraz poddanie działaniu pola magnetycznego, którego wektor jest skierowany prostopadle do kierunku wektora przepływu prądu. Stanowisko to będzie wykorzystywane w Zakładzie Mikroelektroniki i Nanotechnologii Politechniki Wrocławskiej, gdzie materiały te są wytwarzane. W przyszłości może również posłużyć jako stanowisko dydaktyczne dla studentów.

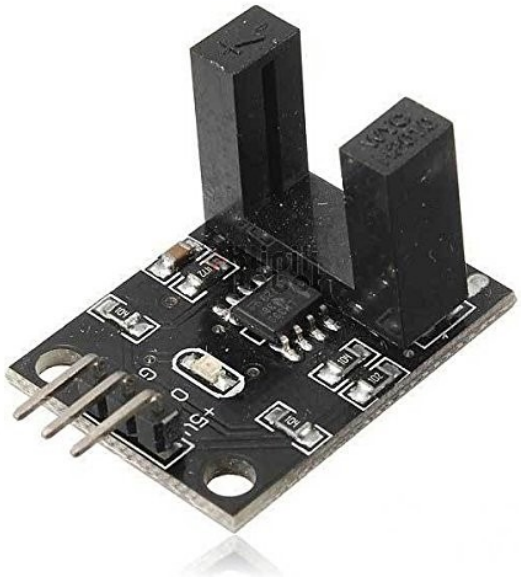
Założeniem wykonania niniejszej pracy była kompleksowa integracja elementów z trzech głównych dziedzin, o jakich traktuje mechatronika, czyli mechaniki, elektroniki oraz informatyki. Częścią mechaniczną jest układ pozycjonowania magnesu stałego. Jest ona oparta o konstrukcje mechaniczną Część elektroniczną stanowić będzie układ sterowania magnesem, oparty na zestawie czujników położenia, oraz układ przełączający kontakty do próbki. W projekcie uwzględnione zostało ponadto wykonanie płytki obwodów drukowanych. Najobszerniejszą częścią pracy było zaprojektowanie i napisanie oprogramowania sterującego systemem pomiarowym przy pomocy mikrokontrolera.

Praca została podzielona na pięć etapów. Pierwszy z nich dotyczy pozycjonowania magnesu stałego względem próbki, aby uzyskiwać zmianę kierunku wektora natężenia pola magnetycznego. Proces ten musiał odbywać się tak, aby magnesy zawsze znajdowały się precyzyjnie nad próbką. Celem dokładnego określenia położenia magnesu zastosowano system czujników, dzięki którym można określić, czy magnesy znajdują się w najbardziej optymalnej pozycji. Pozwala to na stworzenie zamkniętego układu sterowania ze sprzężeniem zwrotnym. Drugim etapem pracy było zaprojektowanie i zbudowanie układu przełączającego, pozwalającego na pomiar wybranych parametrów próbki przy różnych kierunkach przepływu prądu. W tym celu należało określić konfiguracje oraz sposób przeprowadzania pomiarów. Trzeci etap dotyczył oświetlenia próbki. Fragment zaprojektowanego układu elektronicznego, sterującego pomiarem, został przygotowany do późniejszego podłączenia oświetlenia, np. ze źródeł laserowych. Aby spełnić to założenie, aparatura musi ponadto zostać odizolowana od zewnętrznych i wewnętrznych źródeł światła. Kolejnym etapem czyli zadaniami oprogramowania oraz mikrokontrolera było sterowanie warunkami, w jakich będzie przeprowadzony pomiar. Składać się na to musiały algorytmy: przemieszczania magnesu nad próbkę oraz ustawiania odpowiedniego układu połączeń między woltomierzem i zasilaczem, a kontaktami mierzonej próbki. Polecenia do wykonywania powyższych funkcji są wysyłane przez aplikację komputerową do mikrokontrolera przez specjalnie opracowany protokół komunikacyjny.

2. Rozbudowa części mechanicznej

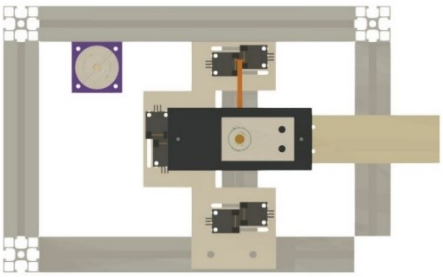
2.2. Projekt rozmieszczenia czujników

Układ ma za zadanie pozycjonować magnes w jednym z trzech miejsc: gdzie działa na próbkę pole magnetyczne o biegunowości północ-południe, odwrotnej oraz takim, w którym pole magnetyczne nie oddziałuje. Każde ze zdefiniowanych położeń jest wyposażone w układ dwóch czujników optycznych. Są one zbudowane z diody podczerwonej oraz odbiornika fal z zakresu nadajnika. Całość urządzenia jest wyposażona w sześć czujników WYC H2010 (rysunek 2).



Rysunek 2. Czujnik optyczny WYC H2010

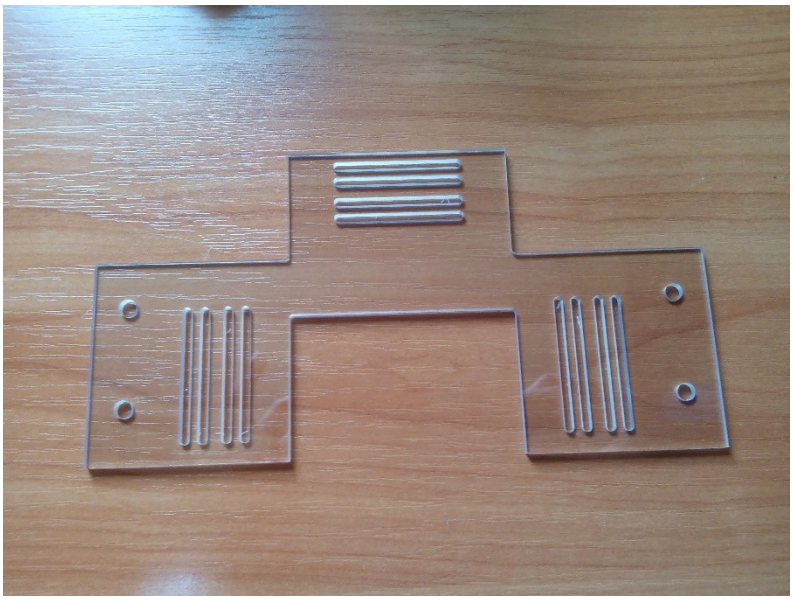
Powodem zdublowania czujników w każdym z trzech położeń jest dokładny odczyt aktualnego położenia magnesu nad próbką. Rozmieszczenie ich jest widoczne na rysunku 3. Ustalają one przez to zakres położenia, w którym ma znaleźć się magnes.



Rysunek 3. Projekt rozmieszczenia czujników w trzech położeniach – widok urządzenia z góry.

2.3. Sposób montażu czujników optycznych

Czujniki są umieszczone w specjalnie wykonanym podłożu. Zbudowane jest ono z poliwęglanu. W podłożu zostały wyfrezowane dwie prowadnice dla każdego z czujników widocznych na rysunku 4. Sensor przytwierdzony jest do podstawy przez śrubę przechodzącą przez otwór w module czujnika i prowadnice, przykręconą do blaszki z nagwintowanymi otworami.



Rysunek 4. Poliwęglanowa podstawa czujników

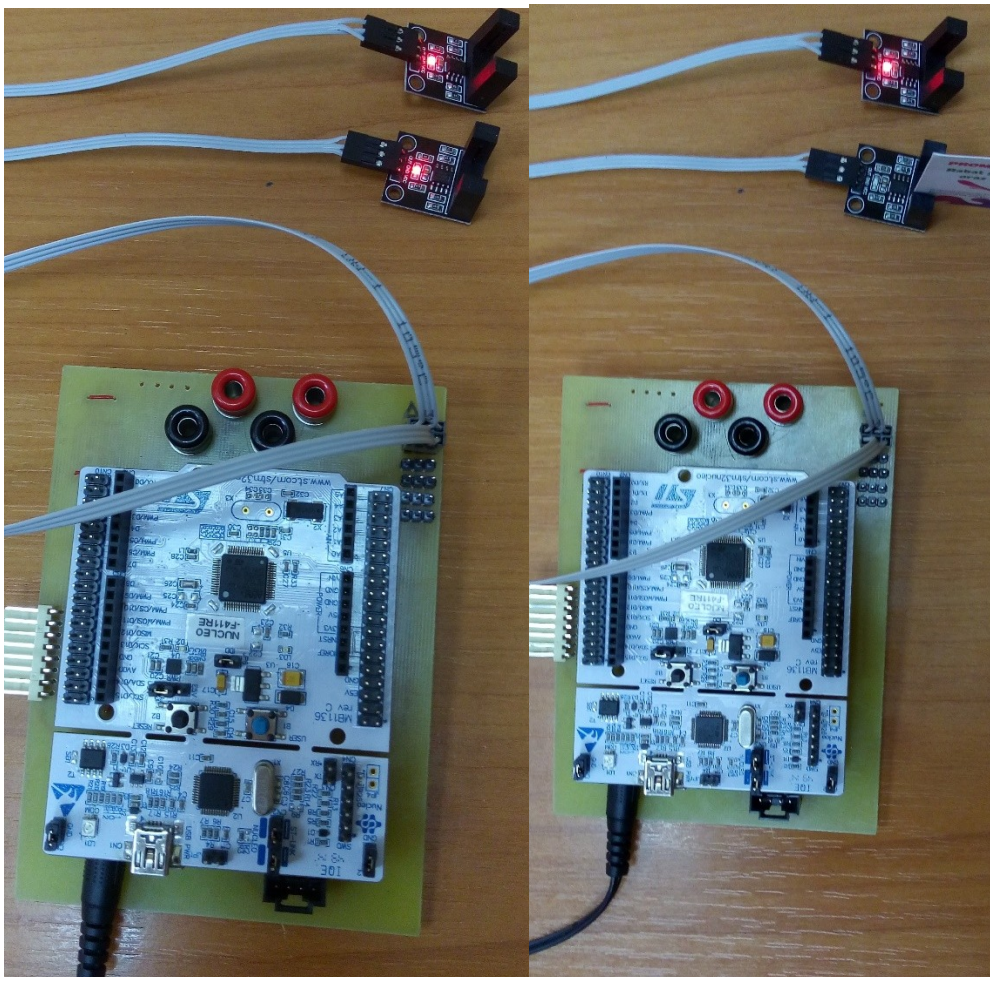
Przy pomocy tego układu można mechanicznie kalibrować położenie czujników, polepszając dokładność i docelowe położenie magnesu nad próbką. Do elektrycznych wyprowadzeń czujników zostały wykorzystane przewody miedziane złączone ze sobą w postaci taśmy z trzema wyprowadzeniami żeńskimi. Przewodem trójżyłowym prowadzone jest zasilanie (5V i masa) oraz napięcie odpowiadające jednemu z dwóch stanów czujnika(5 lub 0V).

3. Realizacja elektronicznej części projektu

3.2. Obsługa i działanie układu czujników położenia

Sensory są skomunikowane z mikrokontrolerem wykorzystując poziomy logiczne. Gdy przestrzeń między nadajnikiem a odbiornikiem zostanie przesłonięta, czujnik zmieni stan z wysokiego na niski, wysyłając informacje poprzez wyprowadzenie do mikrokontrolera o zmianie stanu.. Sonda przymocowana do magnesów, przesłaniając pierwszy z czujników zmienia stan czujnika na niski, informując tym samym mikrokontroler o konieczności dostosowania kroku silnika. Na obudowie czujnika znajduję się dioda informująca o aktualnie wysyłanym stanie logicznym, co można

zaobserwować na rysunku 5. Szerokość markera jest dostosowana tak, by w pożądanym położeniu przesłonić dwa czujniki. Odebranie przez mikrokontroler dwóch stanów niskich, informuje o tym, że magnes znajduje się dokładnie w zadanym położeniu, w którym działa maksymalne pole magnetyczne. Poprzez zastosowanie dwóch czujników możemy kontrolować dokładną pozycję i w razie konieczności skorygować ją.



Rysunek 5. Widok czujników optycznych podłączonych do układu. Po prawej – jeden czujnik przesłonięty.

3.3. Zasada działania układu przekaźników

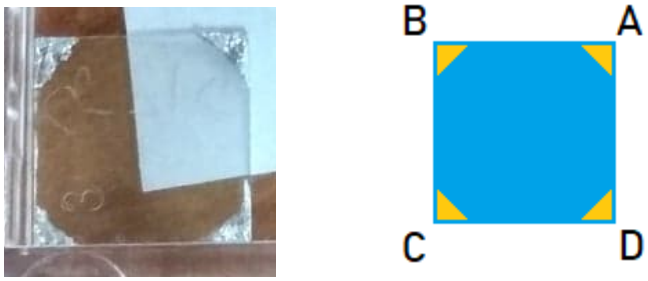
Układ przekaźników został wykonany tak, by można było prowadzić pomiary za pomocą multimetrów mierzących jedną z szesnastu konfiguracji przedstawionych w tabeli 1. Pola kontaktowe na próbce są podłączone do wyprowadzeń na próbce oznaczonych literami A, B, C oraz D, widocznych na rysunku 6. Najistotniejsze w pomiarach są złącza zasilacza na kontaktach. Możliwości ich podłączenia są opisane w ostatniej kolumnie tabeli 1. Konfiguracje nr od 10 do 13 posiadają taki sam układ zasilacza jak już poprzednio zdefiniowane konfiguracje nr od 2 do 5, jednak ze zmienioną

polaryzacją pomiaru napięcia. Z tego powodu zdecydowano się na pominięcie obsługi tych konfiguracji.

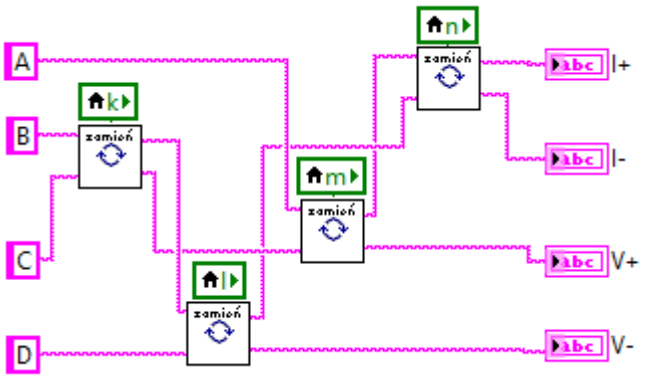
Tabela 1. Konfiguracja pomiarów

l.p	n	m	l	k	A	B	C	D	Komenda	złącza zasilacza na kontaktach
0	0	0	0	0	I+	I-	V+	V-	p0	AB
1	0	0	0	1	I-	I+	V+	V-	p1	BA
2	0	0	1	0	I+	V-	V+	I-	p2	AD
3	0	0	1	1	I-	V-	V+	I+	p3	DA
4	0	1	0	0	V+	I-	I+	V-	p4	CB
5	0	1	0	1	V+	I+	I-	V-	p5	BC
6	0	1	1	0	V+	V-	I+	I-	p6	CD
7	0	1	1	1	V+	V-	I-	I+	p7	DC
8	1	0	0	0	I+	V+	I-	V-	p8	AC
9	1	0	0	1	I-	V+	I+	V-	p9	CA
10	1	0	1	0	I+	V+	V-	I-	-	AD
11	1	0	1	1	I-	V+	V-	I+	-	DA
12	1	1	0	0	V+	I+	I-	V-	-	BC
13	1	1	0	1	V+	I-	I+	V-	-	CB
14	1	1	1	0	V+	I+	V-	I-	pE	BD
15	1	1	1	1	V+	I-	V-	I+	pF	DB

Przez układ przekaźników można prowadzić pomiar bez konieczności ręcznego zamieniania przewodów pomiarowych wyprowadzonych z próbki. Dzięki wykorzystaniu czterech przekaźników połączonych jak na rysunku 6, możemy prowadzić pomiar w dwunastu konfiguracjach.

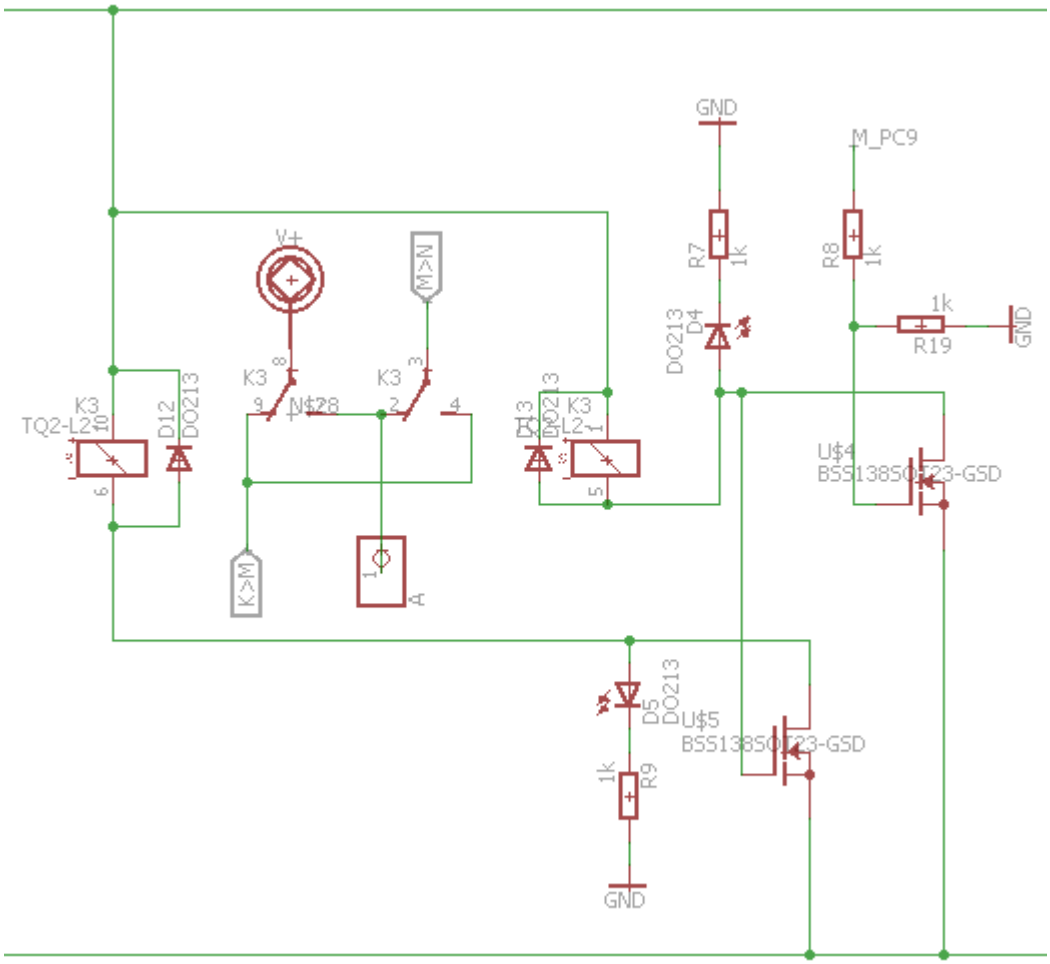


Rysunek 6. Położenie pól pomiarowych na przykładowej próbce.



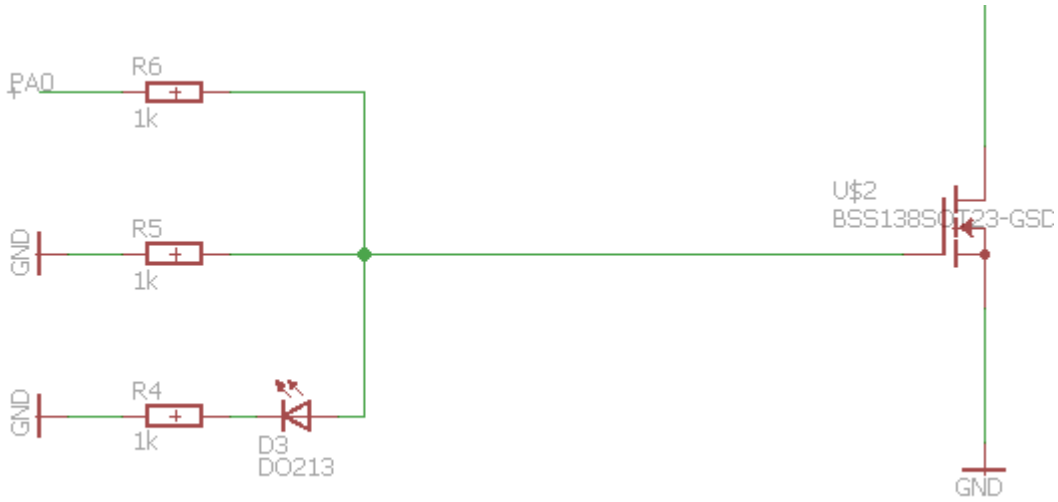
Rysunek 7. Schemat ideowy połączenia przekaźników.

Układ przekaźników składa się z czterech zestawów. Jeden zestaw składa się z dwóch tranzystorów, przekaźnika, dwóch diod elektroluminescencyjnych oraz rezystorów w konfiguracji przedstawionej na rysunku 8.



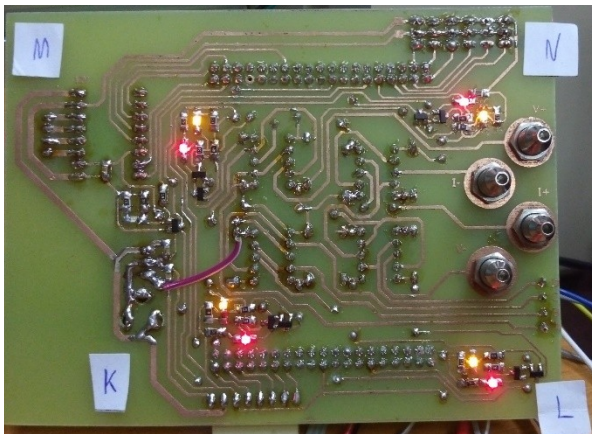
Rysunek 8. Układ jednego z czterech przekaźników

Całość jest podłączona do tranzystora sterującego wszystkimi przekaźnikami. By zmienić stan na każdym z przekaźników należy ustalić za pomocą mikrokontrolera stan wysoki na bramce tranzystora. Dzięki zielonej diodzie dodanej do układu tranzystora sterującego widzimy kiedy tranzystor jest włączony lub wyłączony. Schemat podłączenia jest umieszczony na rysunku 9.



Rysunek 9. Schemat podłączenia tranzystora sterującego

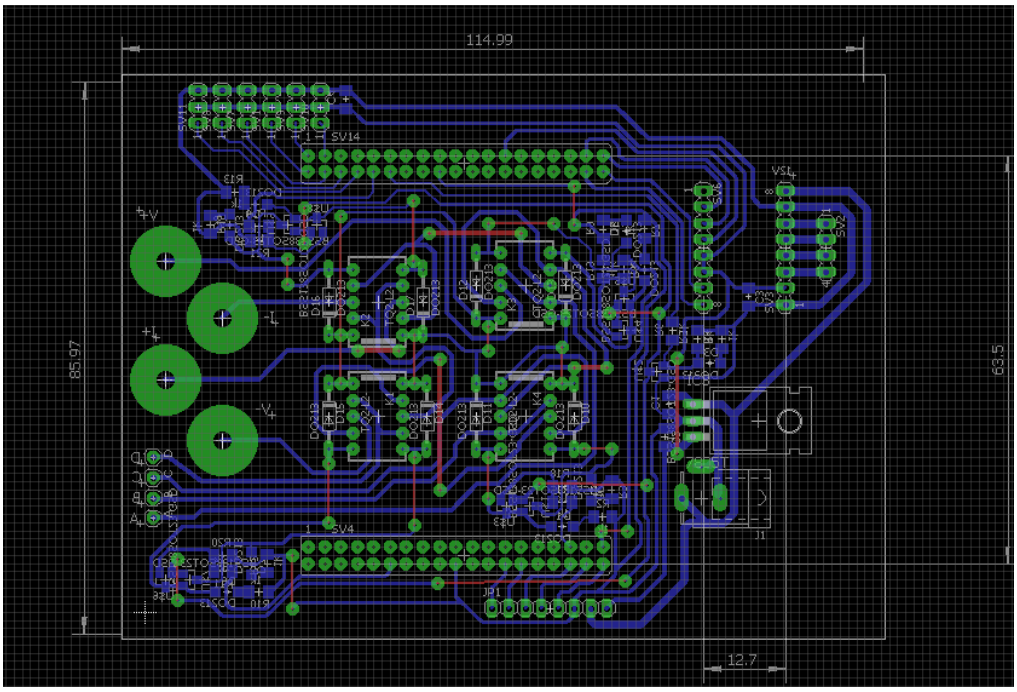
Każdy z układów przekaźników posiada dwie diody: żółtą oraz czerwoną. Każda z nich jest podłączona do tranzystorów, które mają za zadanie ustawić każdy z przekaźników w stan SET lub RESET. Użyty model AZ850 zawiera dwie cewki, co oznacza, że przekaźniki tego typu są bistabilne. Potrzebują one tylko krótkiego impulsu do zmiany stanu, nie jest tym samym konieczne ciągłe generowanie pola magnetycznego, aby utrzymać zadany stan. Dzięki zastosowanemu układowi tranzystorów, wystarczy jeden sygnał cyfrowy do sterowania obiema cewkami. Gdy mikrokontroler wystawi napięcie odpowiadające stanowi logicznemu ‘1’, po zadziałaniu tranzystora sterującego prąd płynie przez cewkę „SET” (świeci przy tym tylko dioda żółta). Gdy na wyprowadzeniu panuje stan logiczny ‘0’, to po włączeniu tranzystora sterującego prąd płynie przez cewkę „RESET” i zapalona zostaje dioda czerwona. W momencie gdy tranzystor sterujący nie jest włączony, są włączone obie diody (rysunek 10).



Rysunek 10. Domyślne działanie przekaźników

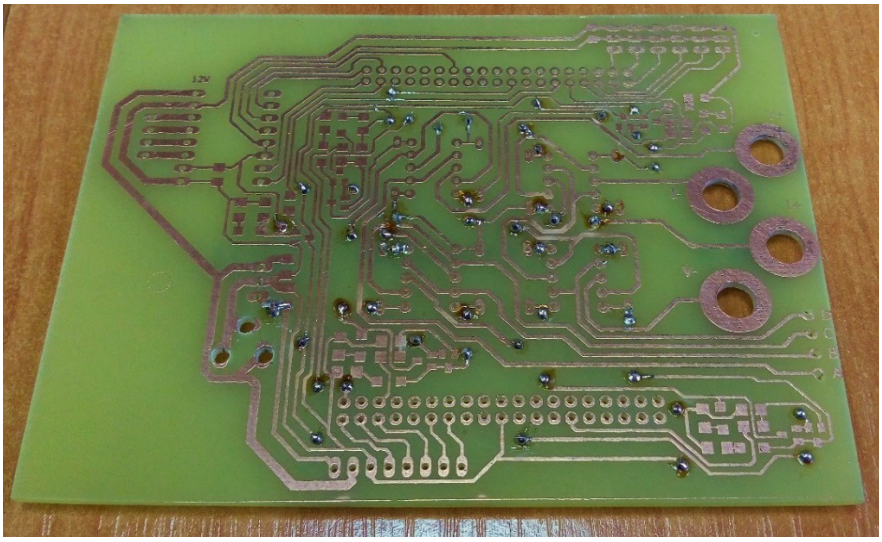
Zestaw diod pozwala na sprawdzenie poprawności załączeń przekaźników. Celem umożliwienia wykonywania pomiaru przy określonym oświetleniu próbki (np. przy pomocy lasera), należy usunąć je z płytki obwodów drukowanych. Mogą również zostać odpowiednio zasłonięte, tak by nie interferowały z docelowym oświetleniem próbki.Projekt oraz wykonanie płytki obwodów drukowanych.

Dedykowana płytka obwodów drukowanych została zaprojektowana i wykonana aby ograniczyć liczbę przewodów, zapewnić niezawodność połączeń oraz integrację wszystkich urządzeń na jednym podłożu (rysunek 11). Ze względu na ograniczenia technologiczne płytka została zaprojektowana i wykonana jednostronnie Jednak z powodu dużego skomplikowania wykorzystane zostały dodatkowe połączenia na górnej części płytki w postaci prostych miedzianych przewodów (rysunek 13).

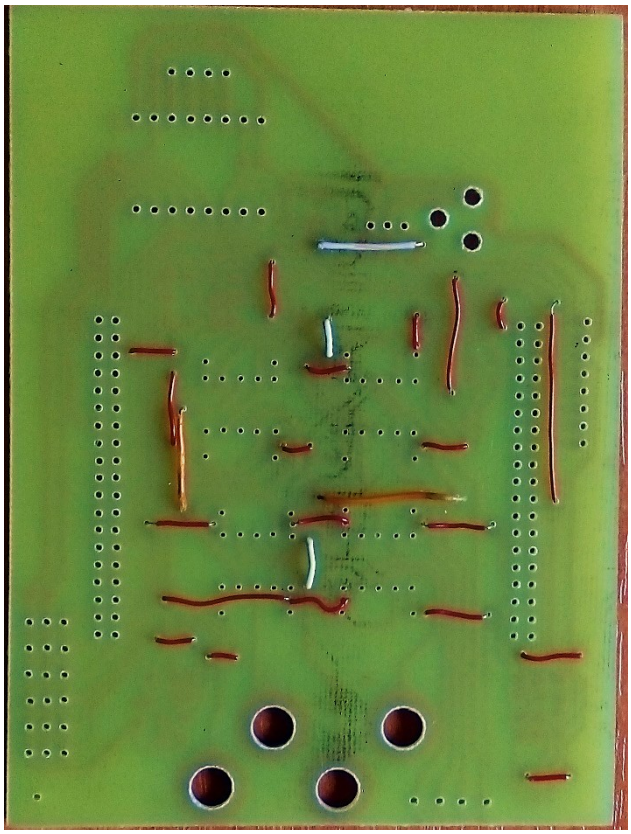


Rysunek 11. Projekt płytki obwodów drukowanych w programie Eagle

Na płytce znajdują się elementy przystosowane zarówno do montażu przewlekanego jak i SMD, widoczne na rysunku 12. Montaż przewlekany został wykonany za pomocą lutowania z użyciem lutownicy kolbowej, natomiast elementy SMD zostały przylutowane za pomocą pasty lutowniczej oraz gorącego powietrza (*hot air*) [2].

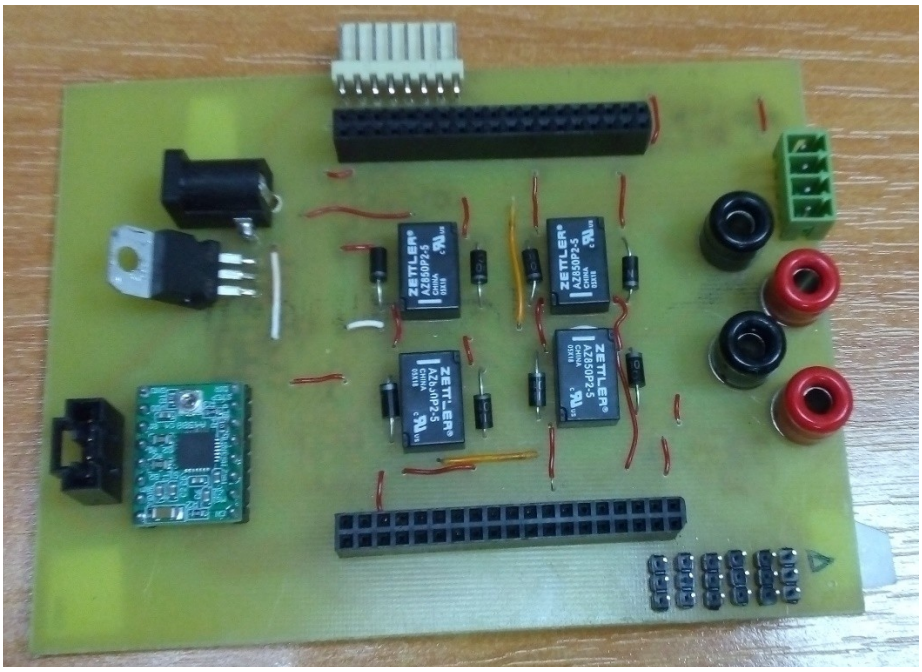


Rysunek 12. Płytka obwodów drukowanych po trawieniu



Rysunek 13. Górna część płytki po przylutowaniu miedzianych przewodów.

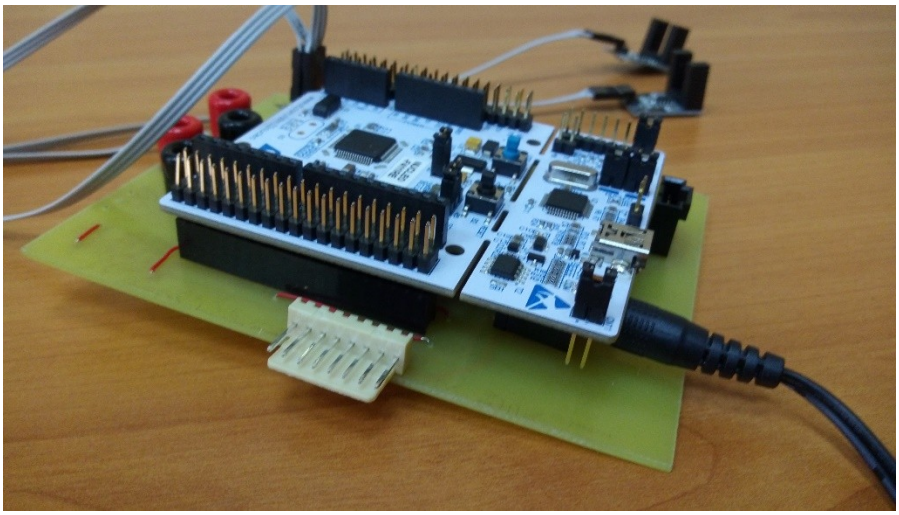
Płytka została wykonana w technologii board-to-board polegającej na dostosowaniu płytki obwodów drukowanych do mikrokontrolera, przez zastosowanie złącz wciskowych widocznych na rysunku 14. Do płytki jest doprowadzone zasilanie 12 V, oraz 5 V stabilizowane przy pomocy stabilizatora napięcia 7805. Silnik krokowy jest zasilany napięciem 12 V, natomiast układ czujników, sterownik silnika oraz przekaźniki są zasilane napięciem 5 V. Został użyty zewnętrzny zasilacz 12 V, ponieważ zasilanie dostępne w mikrokontrolerze nie posiada odpowiedniego natężenia prądu do zasilenia urządzeń znajdujących się na płytce. Na płytce znajdują się ponadto cztery wyprowadzenia. Jedna para wyprowadzeń jest do zasilacza, a druga - do woltomierza. Posiada również cztery wyprowadzenia do próbki na każdą z krawędzi. Cztery przekaźniki oraz sterujące nimi tranzystory znajdują się w centralnej części płytki. Podczas złożenia zostają one zasłonięte przez mikrokontroler.. Wyprowadzenia do zasilania oraz odbioru sygnałów z czujników oraz sterowania oświetleniem, zostały umieszczone na zewnętrznej krawędzi płytki poza obrysem mikrokontrolera. Sterownik silnika wraz z jego wyprowadzeniami również został umieszczony na jednej z krawędzi płytki w celu ułatwienia podłączenia silnika. Płytka została ona wykonana w warsztacie OpenLab Wydziału Elektroniki Mikrosystemów i Fotoniki.



Rysunek 14. Płytka obwodów drukowanych

3.4. Projekt wyprowadzeń do układu oświetlenia

Aby zapewnić możliwość rozwinięcia urządzenia w przyszłości, na płytce umieszczone zostały wyprowadzenia do układu oświetlenia. Jest to pięć wyprowadzeń sterujących podłączonych do portów wejścia-wyjścia mikrokontrolera. Można je dowolnie zaprogramować w tryb odbierania lub wysyłania. Wyprowadzenia mogą być podłączone bezpośrednio do zastosowanych urządzeń oświetlających próbkę lub do kolejnej płytki obwodów drukowanych. Ponadto zostały przygotowane wyjścia napięcia 12 oraz 5 V dla zastosowanych w przyszłości urządzeń. Ostatnim wyprowadzeniem jest wspólne złącze masowe dla mikrokontrolera, płytki obwodów drukowanych oraz peryferii. Wszystkie wyprowadzenia zostały umieszczone na skraju płytki (rys. 15).

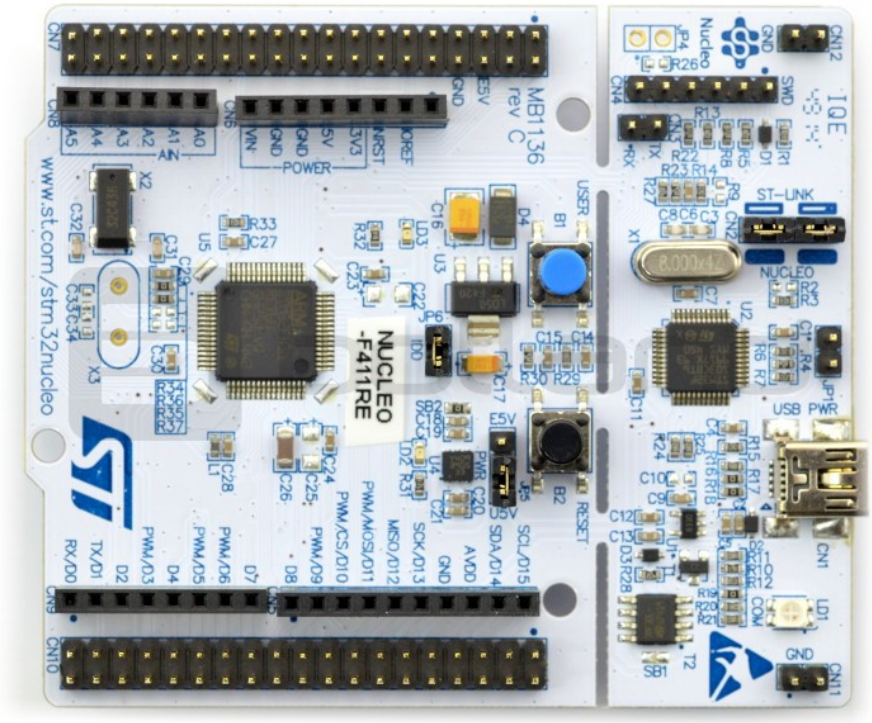


Rysunek 15. Wyprowadzenia kątowe do oświetlenia

Realizacja programowego algorytmu sterowania

3.5. Opis mikrokontrolera oraz środowiska programowania

Mikrokontroler wybrany do projektu to STM 32 Nucleo F411RE. Jest to mikrokontroler z rodziny AVR (Advanced RISC Machines) W module możemy wykorzystać napięcie 5 V oraz 3,3 V, pięć portów analogowych, przetworniki analogowo-cyfrowe, porty cyfrowe w magistralach A, B, C, D, H po trzynaście w każdej, oraz komunikację poprzez UART, I2C lub SPI. Porty te podłączone są na dwóch złączach męskich CN7 i CN10, widoczne na rysunku 16. Dzięki temu można połączyć je ze złączami żeńskimi zaprojektowanej płytki obwodów drukowanych. Przez dużą ilość wyprowadzeń pozwala na obsługę wszystkich założonych funkcjonalności. Dodatkowo jest na tyle rozbudowany, że pozostałe wyprowadzenia mogą zostać użyte w przyszłych rozbudowach urządzenia.



Rysunek 16. Mikrokontroler STM32 Nucleo F411RE

Moduł bazowy zawiera w sobie trzydzieści dwa wyprowadzenia, mikrokontroler, programator SWD oraz moduł do komunikacji UART- USB. Urządzenie bazowe posiada mikrokontroler z architekturą w wersji trzynastej nazwanej Cortex. Architektura ta jest wykorzystywana w jednej z trzech konfiguracji: Cortex-A, Cortex-M i Cortex-R. Układ ten wykorzystuje Cortex-M, czyli układ ze zintegrowaną pamięcią oraz układami peryferyjnymi (Cotex-M4). Zawiera instrukcje DSP oraz koprocesor operacji zmiennoprzecinkowych. Dodatkowo wersja F4 pozwala na taktowanie zegara do 180 Mhz.

Środowiskiem programistycznym, wybranym do pracy nad kodem, był System Workbench for STM32, dostępny w ramach projektu OpenSTM32. Ta aplikacja bazuje na popularnym środowisku Eclipse. Jest to stabilne i powszechnie wykorzystywane środowisko programowania. W projekcie zostały wykorzystane biblioteki innych środowisk, takich jak Cube oraz StdPeriph. System Workbench for STM32 pozwala na tworzenie projektów w językach C oraz C++. Do naszego projektu został wykorzystany język C. W środowisko wbudowany został system kontroli wersji bazujący na platformie GitHub. Podczas pracy nad kodem zostało założone repozytorium umożliwiające aktualizację oraz bezpieczne przechowywanie kopii zapasowych oprogramowania.

3.6. Schemat UML oraz procedury

Do programu został napisany diagram UML. Dzięki niemu zostały przedstawione główne zadania programu oraz sposób ich wykonania. Do opisania projektu został wykorzystany diagram aktywności. Jest on diagramem interakcji, który służy do modelowania dynamicznych aspektów programu. Jego głównymi zadaniami jest przedstawienie sekwencji wykonywanych kolejnych kroków programu [3].

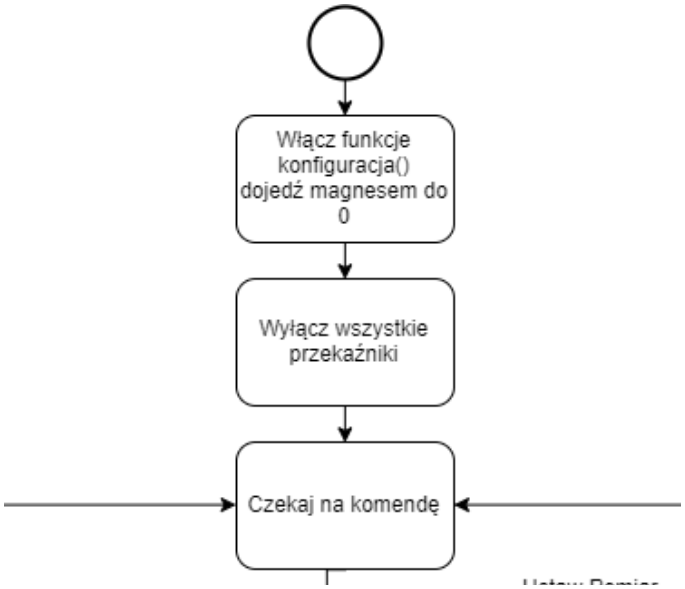
Cały program składa się z trzech plików. W pliku *main.c* znajdują się pliki nagłówkowe pozostałych dwóch plików oraz główna pętla działania programu. Drugi plik *init.c* zawiera deklaracje portów wyjścia i wejścia oraz rozpoczęcie zliczania zegara obsługującego przerwania i komunikację z portem szeregowym. Na rysunku 17 została przedstawiona przykładowa deklaracja wyprowadzenia czternastego w magistrali B jako wyjście w trybie „Push and Pull” [4].

```
77      //deklaracja wyjścia sterującego do przełącznika K1 L
78      gpio.GPIO_Pin = GPIO_Pin_14;
79      gpio.GPIO_Mode =  GPIO_Mode_OUT;
80      gpio.GPIO_OType =  GPIO_OType_PP;
81      GPIO_Init(GPIOB, &gpio);
82
```

Rysunek 17. Przykładowy kod pliku init.c

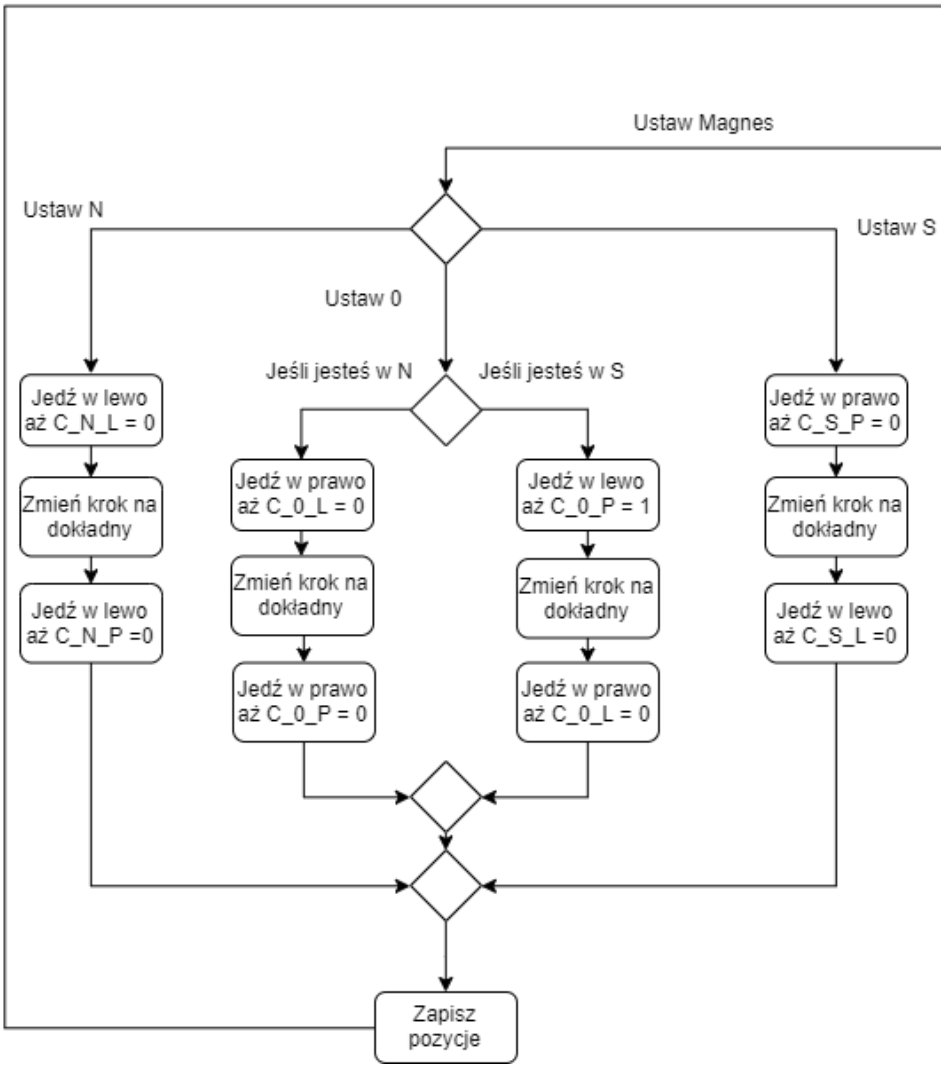
Trzeci plik *parser.c* zawiera wszystkie funkcje wykorzystywane do obsługi przekaźników, silnika i komunikacji z mikrokontrolerem. Na początku pętli głównej wykonuje się funkcja *init()*, która przygotowuje program do działania. W tej funkcji konfigurowane są wszystkie porty wejścia i wyjścia oraz zegar. Po niej wykonywana jest funkcja *konfiguracja()*, która wyłącza wszystkie przekaźniki oraz ustawia magnes w pozycji zerowej, w której nie ma on wpływu na badaną próbkę. Na koniec program wchodzi w nieskończoną pętlę, w której oczekuje na komendy od

użytkownika. Fragment tego kodu przedstawiony jest w górnej części schematu UML na rysunku 18.



Rysunek 18. Fragment diagramu UML opisujący początkową fazę działania programu.

W zależności od wysłanej komendy program zaczyna wykonywać odpowiednią sekwencję. W przypadku wybrania komendy dotyczącej obsługi magnesu rozpoczną się procedury z lewej części diagramu widocznej na rysunku 19. Procedury te wykonują obrót magnesu do odpowiedniego położenia, w kierunku który zależy od aktualnej pozycji. Ponieważ ustawianie skrajnych pozycji (N oraz S) jest wykonywane zawsze w tym samym kierunku, przypadki te zostały opisane w postaci jednej kolumny zadaniowej. Pozwala to na uproszczenie a zarazem większą czytelność diagramu. W rzeczywistości zmieniany jest tylko położenie startowe magnesu, jednak finalne sprawdzenie pozycji odbywa się tak samo dla każdego z przypadków. Przy ustawianiu pozycji zerowej magnes może obracać się zarówno w prawo jak i w lewo. Dlatego należało rozróżnić, który z czujników zostanie zasłonięty pierwszy. Stąd ten przypadek został rozbity na dwa, w zależności od pozycji startowej. Gdy magnes wystartuje z pozycji S, marker pojawi się w pierwszej kolejności w polu czujnika prawego, natomiast w przypadku startu z pozycji N w pierwszej kolejności przesłonięty zostanie czujnik lewy.



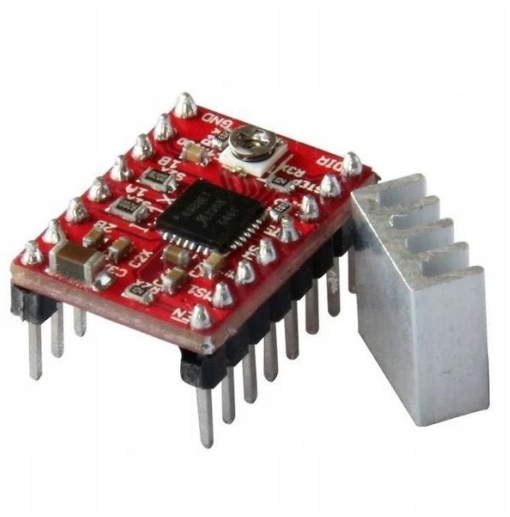
Rysunek 19. Diagram UML przedstawiający sekwencje wykonywanych kroków obsługi silnika

Po wykonaniu każdej z instrukcji zapisywana jest aktualna pozycja silnika oraz program wraca do pętli oczekującej na kolejne komendy.

Obsługa konfiguracji pomiarowych jest znacznie prostsza i polega na wybraniu odpowiedniej funkcji na podstawie przychodzących znaków (rys. 20). Po wykonaniu powraca do pętli głównej programu oczekującej na dalsze komendy.

3.7. Sterowanie położeniem magnesu

Sterowanie magnesem odbywa się poprzez odpowiednią komunikację ze sterownikiem silnika krokowego A4988 (rys. 20). Został on przylutowany do płytki obwodów drukowanych.



Rysunek 20. Sterownik silnika krokowego A4988

Jest do niego doprowadzone napięcie 12 V zasilające silnik oraz 5 V do sterowania samym sterownikiem. Ponadto jedno wyprowadzenie jest używane do określania kierunku obrotu silnika. Trzy wyprowadzenia MS1, MS2 i MS3 decydują o długości pojedynczego kroku. Długość kroku zostanie zmieniona w zależności od wystawianych stanów logicznych na poszczególnych wyprowadzeniach przedstawionych w tabeli 2.

Tabela 2. Konfiguracja kroków silnika krokowego

MS1	MS2	MS3	Wielkość kroku
0	0	0	Pełen krok
1	0	0	1/2 kroku
0	1	0	1/4 kroku
1	1	0	1/8 kroku
1	1	1	1/16 kroku

Kolejne z wyprowadzeń jest użyte do wyzwolenia kroku silnika. W programie jest wyzwalane załączeniem stanu wysokiego oraz niskiego w odstępie 50 ms. Pozostałe cztery wprowadzenia są to przewody sterujące cewkami silnika.

Na początku wykonywania programu, przed wejściem w pętlę główną, wywoływana jest funkcja *konfiguracja()*. Dzięki niej jest realizowany przejazd w prawą stronę do momentu napotkania czujnika odpowiadającego pozycji zerowej magnesu. Jeśli przy inicjalizacji magnes będzie znajdował się w tej pozycji, zostanie on obrócony do pozycji S a następnie wykona dalszy obrót, aż do ponownego napotkania pozycji 0. Tym samym program zapewnia prawidłowe ustawienie magnesu po uruchomieniu urządzenia, niezależnie od stanu w jakim ono się znajduje.

Założenia projektu przewidywały zaprogramowanie sześciu przypadków ustawiania magnesu:

- Z pozycji 0 do S
- Z pozycji 0 do N
- Z pozycji N do S
- Z pozycji N do 0
- Z pozycji S do N
- Z pozycji S do 0

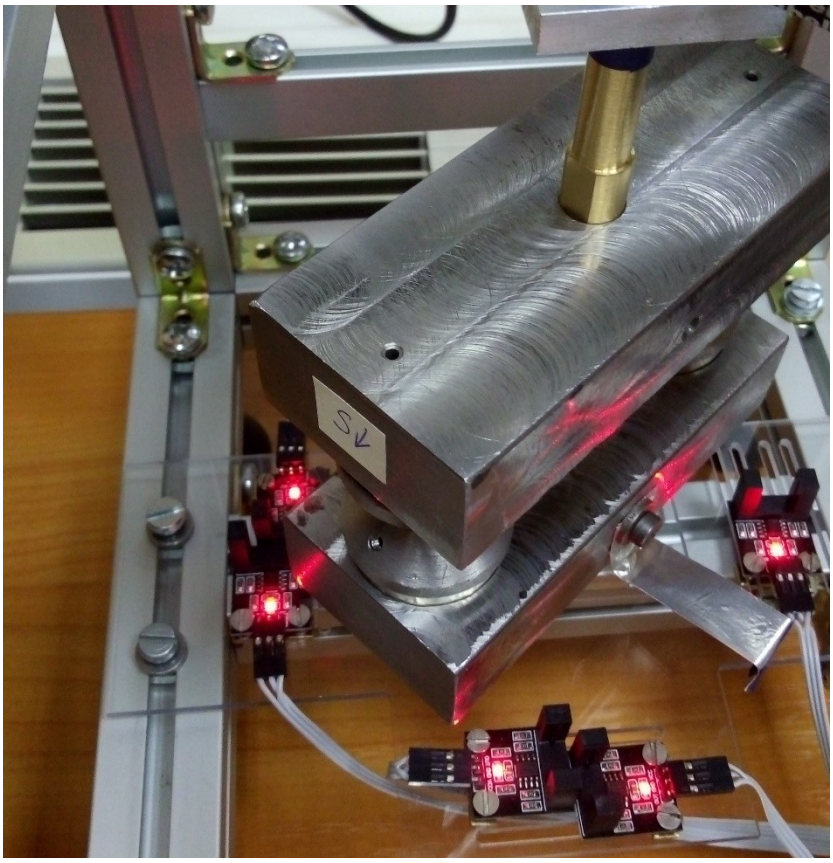
Zostały one zaimplementowane w postaci sześciu odrębnych funkcji. Pozwala to na zachowanie przejrzystości kodu oraz łatwej interpretacji do ewentualnego serwisowania lub rozbudowy urządzenia. Przykładowy kod obsługi obrotu magnesu został umieszczony na rysunku 21.

```
156 void jedz_z_0_do_S()
157 {
158     GPIO_ResetBits(GPIOD, GPIO_Pin_2); // ustaw silnik w prawo
159         while(GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_4) == 0)
160         {
161             GPIO_SetBits(GPIOC, GPIO_Pin_11);
162             delay(1);
163             GPIO_ResetBits(GPIOC, GPIO_Pin_11);
164             delay(50);
165         }
166     GPIO_SetBits(GPIOC, GPIO_Pin_10);
167         while(GPIO_ReadInputDataBit(GPIOC, GPIO_Pin_3) == 0)
168         {
169             GPIO_SetBits(GPIOC, GPIO_Pin_11);
170             delay(1);
171             GPIO_ResetBits(GPIOC, GPIO_Pin_11);
172             delay(200);
173         }
174     GPIO_ResetBits(GPIOC, GPIO_Pin_10);
175     pozycja = 'S';
176     send_string("Magnes w pozycji S\r\n");
```

Rysunek 21. Kod programu przejazdu magnesu z pozycji zerowej do południowej

W pierwszej kolejności zostaje ustawiony kierunek przejazdu, w tym przypadku - w prawo. Aby ten warunek został wykonany na wyprowadzeniu drugim w magistrali D zostaje zmieniony stan na niski. Funkcja zostaje wprowadzona w pętlę '*while()*' która będzie się wykonywać aż zostanie spełniony warunek zakończenia pętli. W tym przypadku jest to zasłonięcie czujnika prawego w pozycji S zadeklarowanego na wyprowadzeniu czwartym magistrali A. Jego stan logiczny w tym momencie zmieni się na niski. W pętli generowany jest impuls powodujący wykonanie kroku silnika, w odstępie czasowym 50 ms. Silnik przemieszcza się do zadanej pozycji (rys. 22). Gdy program wyjdzie z pętli, krok silnika zostaje zmniejszony

o połowę poprzez włączenie stanu wysokiego w wyprowadzeniu dziesiątym w magistrali C, który jest połączony z wyprowadzeniem MS1 sterownika silnika. Następnie obsługa silnika wchodzi w kolejną pętlę, która działa tak samo jak poprzednia, jednak warunkiem jej zakończenia jest przesłonięcia lewego czujnika w pozycji S. Ma to na celu dokładne skorygowanie pozycji. Po zakończeniu pętli, krok silnika ponownie ustawiany jest na maksymalny oraz zmiennej '*pozycja*' przypisywana jest litera 'S'. Jest to zmienna znakowa typu 'char', która przechowuje informację o aktualnej pozycji magnesu. Na końcu funkcji wysyłany jest do komputera ciąg znaków informujący użytkownika o ustawionej pozycji.



Rysunek 22. Przemieszczanie magnesu w zadaną pozycję

W podobny sposób napisano pozostałe pięć funkcji. Różnią się zadeklarowanymi czujnikami kończącymi pętle. Są to czujniki odpowiadające docelowym położeniom magnesu.

Poprzez łagodny start silnika unikamy ryzyka związanego z wpadnięciem urządzenia w drgania. Natomiast poprzez łagodne hamowanie unikamy przemieszczenia magnesów poza zadaną pozycje w wyniku działania sił bezwładności, spowodowanych dużą masą własną dwóch magnesów.

3.8. Sterowanie konfiguracjami pomiaru

Konfiguracje pomiaru wymagają kontroli nad pięcioma wyprowadzeniami mikrokontrolera. Cztery sterują stanami przekaźników, a piąte wyzwala przełączenie przy pomocy tranzystora sterującego.

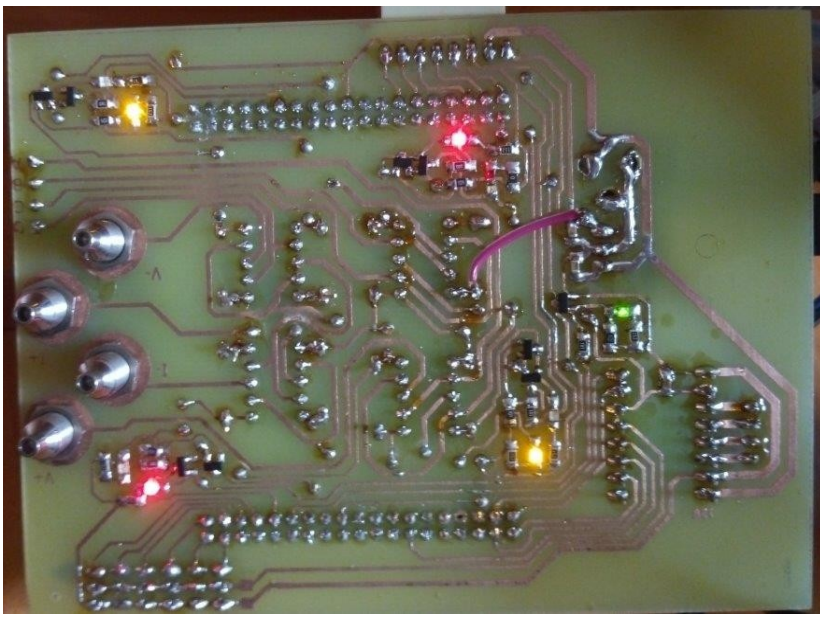
```
296 void P6()
297 {
298     GPIO_ResetBits(GPIOB, GPIO_Pin_9); //k
299     GPIO_SetBits(GPIOB, GPIO_Pin_14); //l
300     GPIO_SetBits(GPIOC, GPIO_Pin_9); //m
301     GPIO_ResetBits(GPIOA, GPIO_Pin_1); //n
302
303     GPIO_SetBits(GPIOA, GPIO_Pin_0); //wyłącz przekaźnik P
304     delay(1000);
305     GPIO_ResetBits(GPIOA, GPIO_Pin_0);
306     send_string("CD\r\n");
307 }
```

Rysunek 23. Fragment kodu sterującego przekaźnikami

Na rysunku 23 przedstawiony jest przedstawiona funkcja ‘P6’ odpowiadająca za włączenie czwartej konfiguracji z tabeli 1. Cztery wyprowadzenia odpowiadają za załączanie kolejno przekaźników:

- Wyprowadzenie dziewiąte w magistrali B za przekaźnik K
- Wyprowadzenie czternaste w magistrali B za przekaźnik L
- Wyprowadzenie dziewiąte w magistrali C za przekaźnik M
- Wyprowadzenie pierwsze w magistrali A za przekaźnik N

W tym przypadku włączone są dwa przekaźniki L oraz M. Na końcu programu przez sekundę włączony jest tranzystor sterujący oraz w tym momencie zmieniany jest stan na wszystkich przekaźnikach. W ostatniej instrukcji wysyłana jest informacja na których kontaktach próbki umieszczone są przewody amperomierza. W tym przypadku są to kontakty C oraz D. Efekt działania programu możemy zaobserwować na rysunku 24.

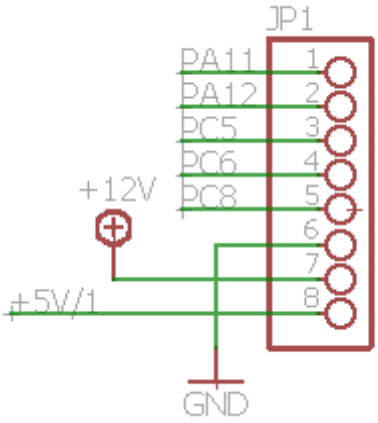


Rysunek 24. Efekt wykonania funkcji 'P6'

3.9. Sterowanie oświetleniem

Sterowanie oświetleniem, które może zostać zrealizowane w przyszłości zostało przygotowane do realizacji na poziomie sprzętowym. Pięć wyprowadzeń mikrokontrolera zostało podłączonych do wyjść kątowych płytki widocznych na rysunku 15. Dedykowanymi wyprowadzeniami do sterowania są (rys. 25):

- Port jedenasty w magistrali A
- Port dwunasty w magistrali A
- Port piąty w magistrali C
- Port szósty w magistrali C
- Port ósmy w magistrali C



Rysunek 25. Schematyczne podłączenie wyprowadzeń

3.10. Dobór protokołu komunikacyjnego z komputerem

Spośród wielu możliwych do wykorzystania protokołów komunikacyjnych w mikrokontrolerze STM32 Nucleo F411RE został wybrany najpopularniejszy RS232. Jest to protokół dostępny powszechnie w wielu urządzeniach w tym również współpracującymi w projekcie. Aktualnie jest wypierany przez złącze USB (rys. 26).

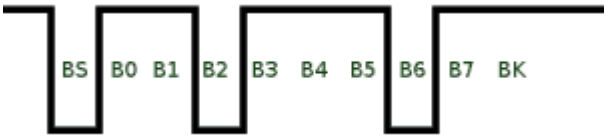
Rysunek 26. Porównanie RS232 z USB

.

Ponadto jest to protokół łatwy w obsłudze. Ze względu na nietypowe napięcia ±11 V konieczne było stosowanie konwerterów takich jak MAX232. Protokół ten jest nadal bardzo popularny w mikrokontrolerach w postaci UART (ang. *Universal Asynchronous Receiver and Transmitter*). Wykorzystywane są napięcia od 0 V do 3,3 V.

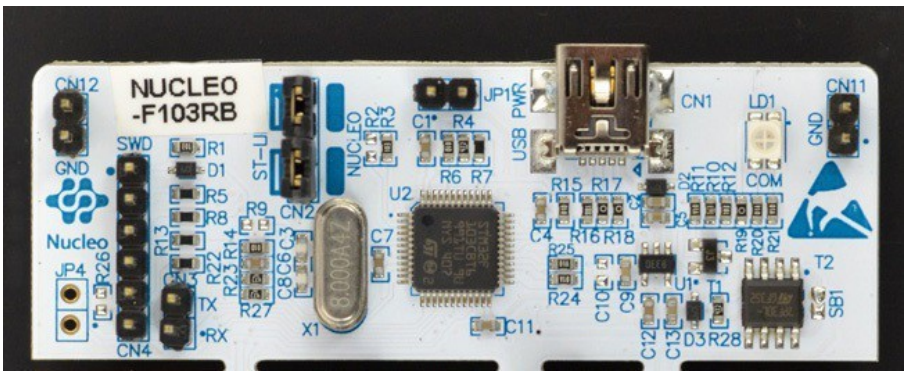
Początkiem komunikacji jest bit startu, zaznaczony na rysunku 27 jako BS. Jest to zawsze logiczne zero. Następnie w zależności od konfiguracji następuje siedem, osiem lub dziewięć

bitów, które przekazują informacje. Na końcu transmisji znajduje się bit końca zaznaczony na rysunku jako BK. Jest on zawsze logiczną jedynką.



Rysunek 27. Przykładowa ramka RS232

Mikrokontroler wykorzystywany w projekcie nie posiada portu szeregowego RS232. Natomiast jest wyposażony w konwerter UART-USB dzięki czemu nasze urządzenie jest widoczne jako port szeregowy COM. Konwerter wraz z programatorem znajduje się na części widocznej na rysunku 28. Jest to część którą można wyłamać i użyć do innych projektów. Komunikacja może być prowadzona synchronicznie lub asynchronicznie w zależności od potrzeb danego projektu. W wypadku tego projektu została zastosowana komunikacja asynchroniczna.



Rysunek 28. Programator STM 32 Nucleo

Aby poprawnie uruchomić komunikację, w pliku *init.c* zostało wykonane zainicjowanie zegara oraz możliwość alternatywnego wykorzystania wyprowadzeń (rys. 29.). Na płytce znajduje się kilka modułów USART. W projekcie został wybrany drugi, ponieważ znajduje się w części z programatorem i można nim się posługiwać przez port USB.

```
25      RCC_APB2PeriphClockCmd(RCC_APB2Periph_SYSCFG, ENABLE);
26      RCC_APB1PeriphClockCmd(RCC_APB1Periph_USART2, ENABLE);
```

Rysunek 29. Konfiguracja zegara.

Kolejnym krokiem była konfiguracja portów wejścia i wyjścia (rys. 30). Linia wyjściowa (TX) jest obecna na porcie drugim magistrali A. Została ona zadeklarowana jako wejście z alternatywną funkcją *push-pull*. Natomiast linia wejściowa (RX) znajdująca się na porcie trzecim magistrali A została ustawiona w tryb *floating*, to znaczy pływający - bez rezystora podciągającego.

```
33      GPIO_StructInit(&gpio);
34      gpio.GPIO_Pin = GPIO_Pin_2;
35      gpio.GPIO_Mode = GPIO_Mode_AF;
36      gpio.GPIO_OType =  GPIO_OType_PP;
37      gpio.GPIO_PuPd = GPIO_PuPd_UP;
38      GPIO_Init(GPIOA, &gpio);
39
40      GPIO_StructInit(&gpio);
41      gpio.GPIO_Pin = GPIO_Pin_3;
42      gpio.GPIO_Mode = GPIO_Mode_AF;
43      gpio.GPIO_OType =  GPIO_OType_PP;
44      gpio.GPIO_PuPd = GPIO_PuPd_UP;
45      GPIO Init(GPIOA, &gpio);
```

Rysunek 30. Konfiguracja komunikacji

Sama konfiguracja portu USART polega w głównej mierze na wybraniu prędkości transmisji - w tym przypadku jest to 9600 (rys. 31).

```
47      USART_StructInit(&uart);
48      uart.USART_BaudRate = 9600;
49      USART_Init(USART2, &uart);
50      USART_Cmd(USART2, ENABLE);
```

Rysunek 31. Konfiguracja prędkości transmisji.

Aby wysłać dane należy wykorzystać funkcję *USART_SendData()*. Przyjmuje ona dwa argumenty: wykorzystywany interfejs oraz wartość, czyli bajt danych. Wysyłany napis jest dzielony na bajty i wysyłany jeden bajt po drugim. Jednak prędkość wysyłania jest mała i szybko zapełnia bufor nadawczy przez co kolejne dane mogą zostać zgubione. W tym celu została wykorzystana funkcja *USART_GetFlagStatus()*, która sprawdza czy bufor nadawczy jest pełny czy pusty. Aby całość działała poprawnie, została napisana funkcja *send_char()*, której procedura przedstawiona jest na rysunku 32. Pętla *while* sprawdza, czy bufor nadawczy jest pusty, a następnie wysyła dane.

```
538 void send_char(char c)
539 {
540     while (USART_GetFlagStatus(USART2, USART_FLAG_TXE) == RESET);
541     USART_SendData(USART2, c);
542 }
```

Rysunek 32. Kod funkcji send_char

Aby wysłać cały napis, została napisana funkcja *send_string()*. Jej kod widoczny na rysunku 33 przyjmuje jako argument ciąg znaków. Wskaźnik ustawiony jest na jego początek, a następnie wysyłany jest znak po znaku.

```
544 void send_string(const char* s)
545 {
546     while (*s)
547     send_char(*s++);
```

Rysunek 33. Kod funkcji send_string

Po zdefiniowaniu wysyłania oraz odbierania znaków, została napisana funkcja *parser()*. Ma ona za zadanie obsługiwać wszystkie komendy z tabeli 3. Bazuje ona na instrukcji *switch-case* [5]. Jako argument jest przekazywana dwuznakowa komenda. Argument jest tablicą znaków, dzięki temu można wskazać na pierwszy oraz drugi znak argumentu. Pierwszymi znakami są:

- M – sterowanie płożeniem magnesu
- S – sterowanie oświetleniem
- P – sterowanie pomiarem
- C – sprawdzenie poprawności działania całego systemu

Dzielą one komunikację na bloki zadaniowe odpowiadające za docelową obsługę jednego modułu.

Drugim znakiem w konfiguracji M są docelowe położenia magnesu:

- 0 – do położenia ‘0’
- S – do ustawienia bieguna S nad próbką
- N – do ustawienia bieguna N nad próbką

Przy późniejszym rozwoju projektu można uzupełnić ten moduł obsługi oświetlenia, wykorzystując tę samą logikę, np. gdzie litera jest pierwszą literą włączanego koloru np.:

- R – czerwony
- G – zielony
- B – Niebieski

Sterowanie pomiarem wykorzystuje numeracje szesnastkową od 0 do F, gdzie numery oznaczają kolejno numery konfiguracji z tabeli 1.

Tabela 3. Tabela komend

Komenda	Wykonanie
c	Sprawdzenie poprawności działania urządzenia
m0	ustawienie pozycji zerowej magnesu
mN	ustawienie pozycji N

mS	ustawienie pozycji S
p0	Wszystkie przekaźniki wyłączone
p1	Włączony przekaźnik K
p2	Włączony przekaźnik L
p3	Włączony przekaźnik K i L
p4	Włączony przekaźnik M
p5	Włączony przekaźnik K i M
p6	Włączony przekaźnik M i L
p7	Włączony przekaźnik K, L i M
p8	Włączony przekaźnik N
p9	Włączony przekaźnik N i K
pE	Włączony przekaźnik N, M i L
pF	Wszystkie przekaźniki włączone

W każdej instrukcji wyboru jest zagnieżdżona funkcja obsługująca. Przykładowy fragment kodu przedstawiono na rysunku 34. Jeśli pierwszym znakiem będzie litera ‘m’, instrukcja zacznie wybierać jedną z możliwości. W przypadku gdy drugim znakiem jest ‘0’, funkcja sprawdzi aktualne położenie magnesu i na tej podstawie wykona odpowiednią funkcję obsługi silnika. Jeśli zmienna *pozycja* będzie miała wartość taką samą jak zadana, urządzenie przemieści magnes do najbliższego znanego położenia, a następnie ustawi położenie docelowe, aby je dokładnie spozycjonować.

```
case 'm':
    switch(slowo[1]){
        case '0':
            if(pozycja=='S')
            {
                jedz_z_S_do_0();
            }
            else if(pozycja=='N')
            {
                jedz_z_N_do_0();
            }
            else
            {
                jedz_z_N_do_S();
                jedz_z_S_do_0();
            }
            break;
```

Rysunek 34. Obsługa przejazdu silnika

Obsługa przekaźników jest prostsza, ponieważ w dowolnej chwili mogą być zmienione niezależnie od stanu poprzedniego lub aktualnego. Przykładowy kod przedstawiony na rysunku 35 zawiera wywołanie dedykowanej funkcji dla komendy, której zawartość jest widoczna na rysunku 36.

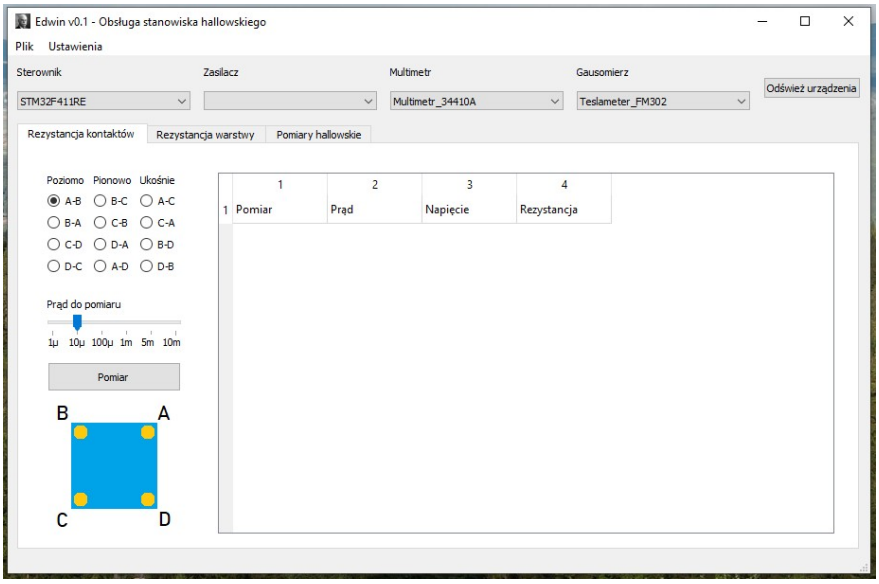
```
case 'p':
    switch(slowo[1]){
        case '0':
            P0();
            break;
```

Rysunek 35. Obsługa komendy ‘p0’.

```
224 void P0()
225 {
226     GPIO_ResetBits(GPIOB, GPIO_Pin_9); //k
227     GPIO_ResetBits(GPIOB, GPIO_Pin_14); //l
228     GPIO_ResetBits(GPIOC, GPIO_Pin_9); //m
229     GPIO_ResetBits(GPIOA, GPIO_Pin_1); //n
230
231     GPIO_SetBits(GPIOA, GPIO_Pin_0); //wyłącz przekaźnik P
232     delay(1000);
233     GPIO_ResetBits(GPIOA, GPIO_Pin_0);
234     send_string("AB\r\n");
235 }
```

Rysunek 36. Wykonanie funkcji p0().

Schemat komend został podzielony tak, by w prosty sposób komunikować mikrokontroler z aplikacją sterującą z komputera oraz by ułatwić rozwijanie kodu w przyszłości. Aplikacja, sterująca z poziomu komputera, dla której została napisana komunikacja, ułatwi w znaczący sposób obsługę urządzenia. Czytelnie opisane przyciski aplikacji wysyłają ciągi znaków, które są potrzebne tylko do komunikacji między mikrokontrolerem a aplikacją, co eliminuje konieczność tworzenia instrukcji obsługi i opisywania każdej komendy użytkownikowi. Na rysunku 37 został umieszczony ekran główny prototypowej aplikacji.



Rysunek 37. Ekran główny aplikacji sterującej

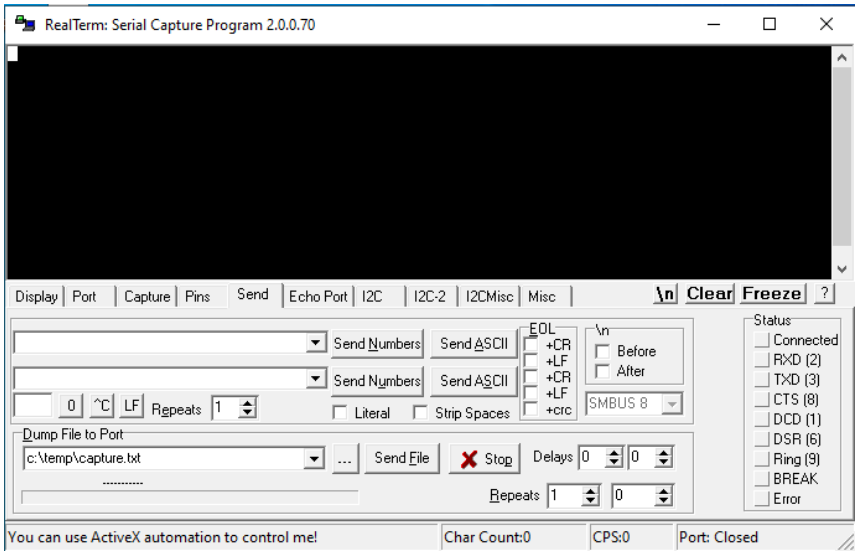
4. Podsumowanie

4.2. Uruchomienie oraz testowanie

Elementy urządzenia były testowane wraz z postępem prac nad nim. Został on podzielony na części, które mogły stwarzać problemy. Lokalizowanie błędów podczas takich działań jest znacznie łatwiejsze niż w przypadku złożenia projektu w całości i następnie przejścia do testowania.

W pierwszej kolejności został sprawdzony kod sterujący działaniem silnika. Zostały wykorzystane trzy z sześciu czujników położenia. Układ przycisków zbudowany na płytce stykowej zastępował komunikację z komputerem. Następnie zostały dodane kolejne trzy czujniki pozycjonujące położenie.

Aby sprawdzić poprawność działania komunikacji do każdej z komend został dopisany ciąg znaków informujący użytkownika, że mikrokontroler poprawnie wykonał zadanie. Wysyłanie oraz odbieranie ciągów było możliwe dzięki wykorzystaniu aplikacji *RealTerm*. Jej ekran główny z polami do wysyłania znaków oraz ich odbierania został przedstawiony na rysunku 38.

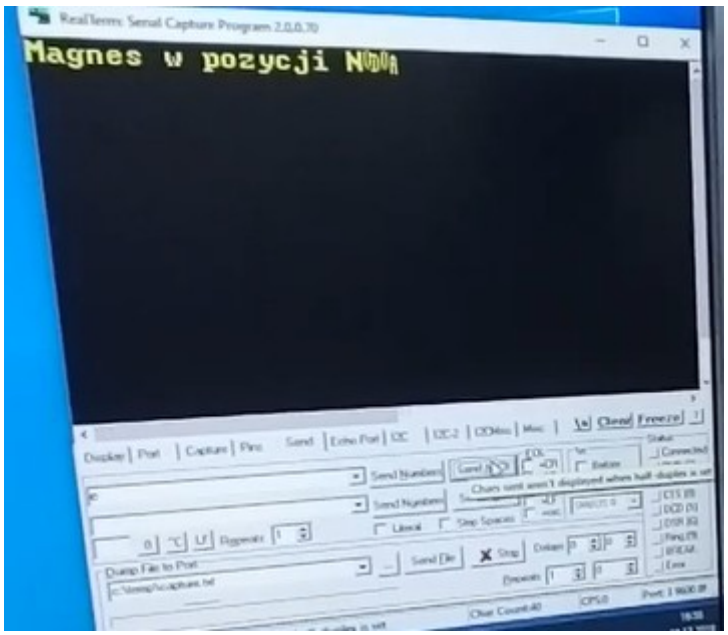


Rysunek 38. Ekran główny aplikacji RealTerm.

Do wysyłania została wykorzystana funkcja *send_string()*. Użycie tej funkcji zostało zademonstrowane na rysunku 39. Po wykonaniu całej procedury mikrokontroler wysyła odpowiedź z linijki 106, którą widać na rysunku 40.

```
109 void jedz_z_S_do_N()
110 {
111     GPIO_SetBits(GPIOD, GPIO_Pin_2); // ustaw kierunek w lewo
112         while(GPIO_ReadInputDataBit(GPIOC, GPIO_Pin_1) == 0) // sprawdz czujnik lewy N
113         {
114             GPIO_SetBits(GPIOC, GPIO_Pin_11);
115             delay(1);
116             GPIO_ResetBits(GPIOC, GPIO_Pin_11);
117             delay(50);
118         }
119     GPIO_SetBits(GPIOC, GPIO_Pin_10);
120         while(GPIO_ReadInputDataBit(GPIOC, GPIO_Pin_0) == 0)
121         {
122             GPIO_SetBits(GPIOC, GPIO_Pin_11);
123             delay(1);
124             GPIO_ResetBits(GPIOC, GPIO_Pin_11);
125             delay(200);
126         }
127     GPIO_ResetBits(GPIOC, GPIO_Pin_10);
128     pozycja = 'N';
129     send_string("Magnes w pozycji N\r\n");
130 }
```

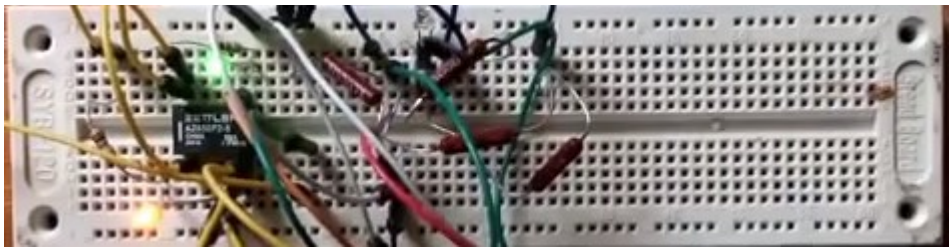
Rysunek 39. Procedura przejazdu magnesu z komunikacją



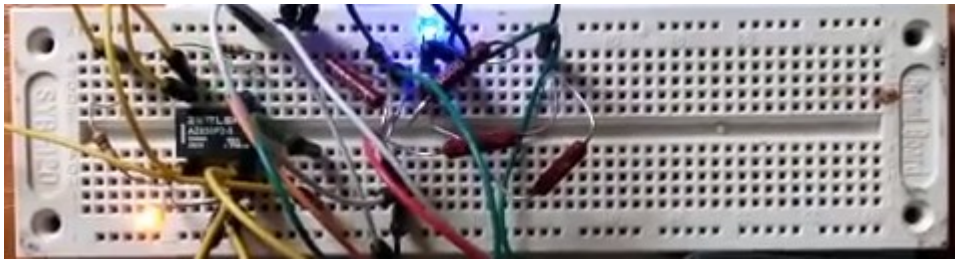
Rysunek 40. Odpowiedź zwrotna po wykonaniu funkcji

Kolejnym krokiem była zamiana przycisków płytki stykowej na porty cyfrowe, które będą kontrolować silnik oraz ustawianie magnesu względem próbki.

Aby sprawdzić poprawność działania modułu przekaźników, został odtworzony układ jednego z nich na płytce stykowej. Poprawność działania przekaźnika sygnalizowały diody. Na rysunku 41 dioda zielona sygnalizuje przekaźnik w stanie SET, natomiast na rysunku 42 dioda niebieska - przekaźnik w stanie RESET.

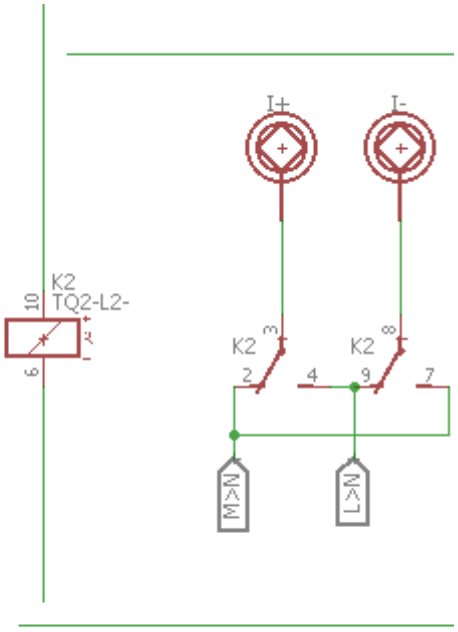


Rysunek 41. Sygnalizacja przekaźnika w stanie SET



Rysunek 42. Sygnalizacja przekaźnika w stanie RESET

Kolejnym krokiem było wytworzenie płytki obwodów drukowanych. Aby układ przekaźników działał w sposób poprawny została wykorzystana symulacja w programie *LTSpice*. Dzięki tej symulacji uniknięto pierwotnego błędu spowodowanego złym doborem rezystancji rezystorów w modułach sterujących przekaźnikami. Następnie projekt został przeniesiony do programu do projektowania płytek obwodów drukowanych. Po przerysowaniu schematu oraz poprowadzeniu ścieżek została użyta wbudowana funkcja sprawdzenia poprawności połączeń o nazwie „ERC" (*Electrical Rule Check)*. Na rysunku 43 został przedstawiony przykładowy błąd połączenia w programie. W tym miejscu dzięki podpowiedziom programu uniknięto przypadkowych błędów na schemacie, oraz później w przypadku mozaiki połączeń - zwarć i odpowiednio dobrano szerokość ścieżek oraz odstępu między nimi.

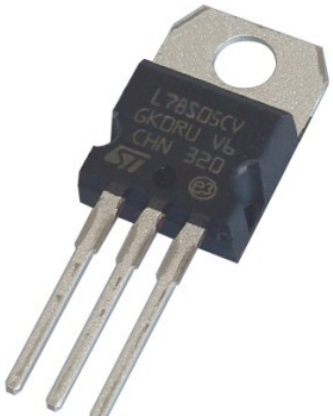


Rysunek 43. Przykładowy błąd braku połączenia ścieżek w schemacie

Po wydrukowaniu oraz naniesieniu ścieżek metodą termotransferu, specjalnym flamastrem zostały poprawione przerwy w ścieżkach oraz wielkości pól lutowniczych. Po wytrawieniu płytki, pod mikroskopem została sprawdzona jakość wykonania ścieżek przewodzących.

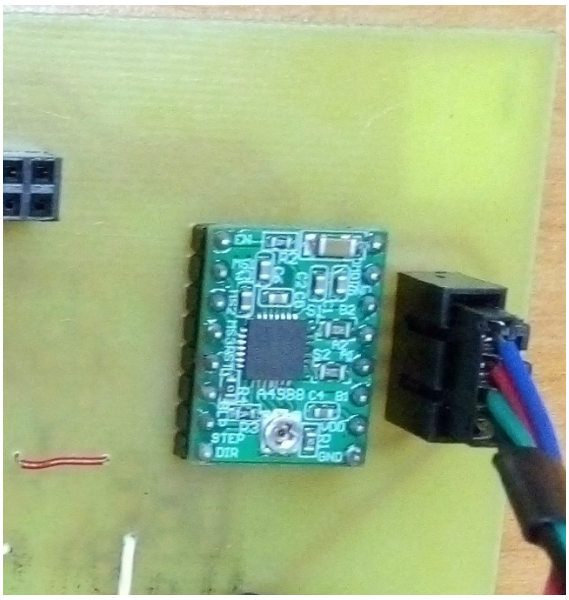
Kolejny etap to etapowe przyłączanie elementów elektronicznych do podłoża płytki obwodów drukowanych czyli drugi poziom montażu elektronicznego. W pierwszej kolejności zostały przylutowane elementy zasilania, czyli złącze sieciowe, zestaw kondensatorów oraz stabilizatora napięcia 7805, widoczny na rysunku 44. Aby sprawdzić poprawność wykonania

dodano również wszystkie złącza wejściowe oraz wyjściowe. Aby sprawdzić czy zasilanie działa poprawnie, zmierzono napięcia 5 V i 12 V na płytce za pomocą multimetru.



Rysunek 44. Stabilizator napięcia 7805

Następnie dołączony został sterownik silnika oraz gniazdo wyprowadzeń do cewek silnika. Aby uniknąć błędnego podłączenia wtyczki, zastosowano gniazdo z kluczem widocznym na rysunku 45. Dzięki temu nie zostaną one wpięte w odwrotnej biegunowości.



Rysunek 45. Sterownik silnika wraz z wyprowadzeniami cewek silnika w formie złącza z kluczem

By sprawdzić poprawność połączeń, podłączono silnik, oraz wyprowadzenia z mikrokontrolera, bezpośrednio do sterownika silnika. Zostały do tego wykorzystane przewody o złączach żeńsko-męskich. Sprawdzono poprawne wykonywanie procedury wcześniej napisanego programu obsługi silnika.

Kolejnym etapem było przyłączenie i sprawdzenie poprawności działania zestawu przekaźników. Każdy z czterech zestawów został przylutowywany osobno i testowany zaraz po montażu. Po jego zakończeniu do wyprowadzeń sterujących było doprowadzane napięcie zewnętrzne 3,3 V i sprawdzany kolor diody sygnalizującej. Żółta dioda oznacza przekaźnik w stanie SET, a czerwona – w stanie RESET. Po zamontowaniu wszystkich przekaźników, na podstawie funkcji *check()*, której kod znajduje się na rysunku 46, sprawdzono poprawność wykonania wszystkich połączeń modułu pomiarowego z przekaźnikami. Do sprawdzenia tego posłużyła osobna płytka stykowa oraz zewnętrzny mikrokontroler, który w sposób zsynchronizowany z przełączaniem przekaźników wystawiał na czterech wyprowadzeniach stan wysoki 5 V. Wyprowadzenia te były podłączone do czterech diod: zielonej, niebieskiej, białej i żółtej. Każdy cykl wiązał się ze zmianą kolejności włączania się diod, co potwierdziło poprawność wykonania całego modułu pomiarowego.

```
void check()
{
    //konfiguracja();
    jedz_z_0_do_N();
    jedz_z_N_do_0();
    jedz_z_0_do_S();
    jedz_z_S_do_0();
    jedz_z_0_do_N();
    jedz_z_N_do_S();
    jedz_z_S_do_N();
    konfiguracja();
    P0();
    delay(2000);
    P1();
    delay(2000);
    P2();
    delay(2000);
    P3();
    delay(2000);
    P4();
    delay(2000);
    P5();
    delay(2000);
    P6();
    delay(2000);
    P7();
    delay(2000);
    P8();
    delay(2000);
    P9();
    delay(2000);
    PE();
    delay(2000);
    PF();
    delay(2000);
    P0();
    delay(2000);
}
```

Rysunek 46. Kod programu funkcji check()

4.3. Ocena wykonanego projektu i wnioski

Założenia projektowe zostały wykonane w pełni. Urządzenie w szybki i automatyczny sposób przemieszcza magnes nad próbkę w każdej z możliwych konfiguracji. Dzięki zestawowi

czujników optycznych, nabiegunniki w każdym z możliwych położeń znajdują się w miejscu docelowym.

Układ elektroniczny pozwala na rozpoczęcie wykonywania pomiarów. Zestaw przekaźników działa poprawnie, co potwierdziły testy. W przyszłości, do wykonania kolejnej płytki obwodów drukowanych dla tego projektu, lepszym wyborem byłoby wytworzenie jej w technologii fotolitografii.

Napisane oprogramowanie spełnia każde z przedstawionych założeń. Jest napisane i opisane w taki sposób, by rozbudowanie go nie stanowiło problemu. Dodatkowym atutem oprogramowania jest możliwość jego rozwoju w oparciu o system kontroli wersji *GitHub*.

4.4. Możliwości dalszego rozwoju

Urządzenie może być dodatkowo rozbudowane o kolejne funkcjonalności. Pierwszą z nich jest układ oświetlenia próbki. Do tego celu należy przestrzeń wewnątrz obudowy urządzenia odizolować od zewnętrznych niepożądanych źródeł światła. Dodatkowo urządzenia oświetlające powinny mieć odpowiednią moc oraz oświetlać próbkę równomiernie. Najlepszym rozwiązaniem tego zadania byłoby zastosowanie laserów półprzewodnikowych.

Kolejną możliwością rozwoju projektu jest działanie na próbkę poprzez regulację temperatury. Jednak, jak w poprzednim przykładzie, konieczne jest odizolowanie niektórych elementów aparatury pomiarowej. Zmienne cykle temperaturowe lub wielkość jej amplitudy mogą trwale uszkodzić urządzenie.

4.5. Podziękowania

Praca została wykonana pod opieką i wsparciem Pana Doktora inżyniera Damiana Radziewicza oraz Pana Gabriela Ceballosa. Pragnę podziękować za wsparcie na każdym etapie realizacji pracy dyplomowej oraz wykonywania związanego z nią projektu.

Kieruję podziękowania do Pana Magistra inżyniera Grzegorza Klubińskiego za pomoc przy realizacji płytki obwodów drukowanych oraz Panu Januszowi Szydłowskiemu za pomoc przy jej projektowaniu

Przedstawiona praca dyplomowa była realizowana dzięki wsparciu ze strony projektu „Międzyuczelniane Centrum Dydaktyczno-Technologiczne ‘TECHNOPOLIS’ we Wrocławiu” współfinansowanego przez Unię Europejską ze środków Europejskiego Funduszu Rozwoju Regionalnego w ramach Programu Infrastruktura i Środowisko nr UDA-POIS.13.01-021/09-00.

5. Bibliografia

[1] Podstawy fizyki. David Halliday, Robert Resnick, Jearl Walker Wydawnictwo naukowe PWN

[2] Jan Felba. Montaż w Elektronice. Oficyna Wydawnicza Politechniki Wrocławskiej

[3] https://www.michalwolski.pl/ Diagramy UML

[4] https://forbot.pl/ Kurs STM 32

[5] Język C Szkoła programowania Wydanie VI Stephen Prata