Sprawozdanie z laboratorium
Systemy i sieci przemysłowe

Maciej Misiewicz	Oskar Zieliński	Dariusz Witek vel Witkowski	Szymon Panek
215305	215373	215364	215319

4 listopada 2020

Spis treści

1 Ćwiczenie 1: Komunikacja CAN **3**
1.1 Cel ćwiczenia . . . 3
1.2 Realizacja ćwiczenia . . . 3
1.2.1 Konfiguracja połączenia z robotem . . . 3
1.2.2 Ustawienie cyklu pracy układu lokalnego 30ms . . . 4
1.2.3 Ustawienie cyklicznego odczytu danych z akcelerometru . . . 5
1.2.4 Ustawienie cyklicznego odczytu danych z czujników odległości . . . 5
1.2.5 Zmiana położenie serwa 1 i serwa 2 . . . 5
1.2.6 Zmiana cyklu pracy układu lokalnego na 50ms . . . 6
1.3 Wnioski i spostrzeżenia . . . 6

2 Ćwiczenie 2: Sterowanie manipulatorem za pomocą sterownika CompactRIO (NI) **7**
2.1 Cel ćwiczenia . . . 7
2.2 Realizacja ćwiczenia . . . 7
2.2.1 Uzupełnienie brakującej części kodu . . . 7
2.2.2 Dobranie nastaw regulatorów PID dla ramienia oraz nadgarstka robota . . . 9
2.3 Wnioski i spostrzeżenia . . . 10

3 Ćwiczenie 3: Sterowanie robota za pomocą panelu operatorskiego sprzężonego przez sieć Ethernet ze sterownikiem CompactRIO. **10**
3.1 Cel ćwiczenia . . . 10
3.2 Realizacja ćwiczenia . . . 11
3.2.1 Kod programu . . . 11
3.2.2 Gotowa aplikacja . . . 12
3.3 Wnioski i spostrzeżenia . . . 13

1 Ćwiczenie 1: Komunikacja CAN

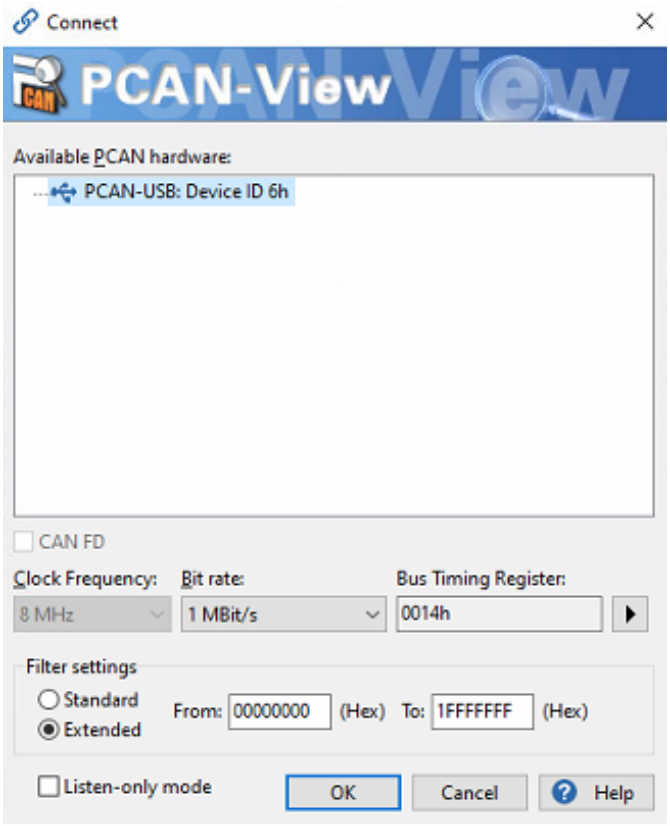
1.1 Cel ćwiczenia

Celem ćwiczenia było zapoznanie się z komunikacją za pomocą protokołu CAN na przykładzie połączenia z częścią składową robota hipermobilnego Wheeeler. Ćwiczenie obejmowało wysyłanie rozkazów oraz odbieranie informacji od lokalnego sterownika za pomocą ramek danych.

1.2 Realizacja ćwiczenia

1.2.1 Konfiguracja połączenia z robotem

Konfiguracja połączenia z robotem ograniczała się do określenia dwóch parametrów - prędkości transmisji jako 1Mbit/s oraz formatu identyfikatora jako format extended.

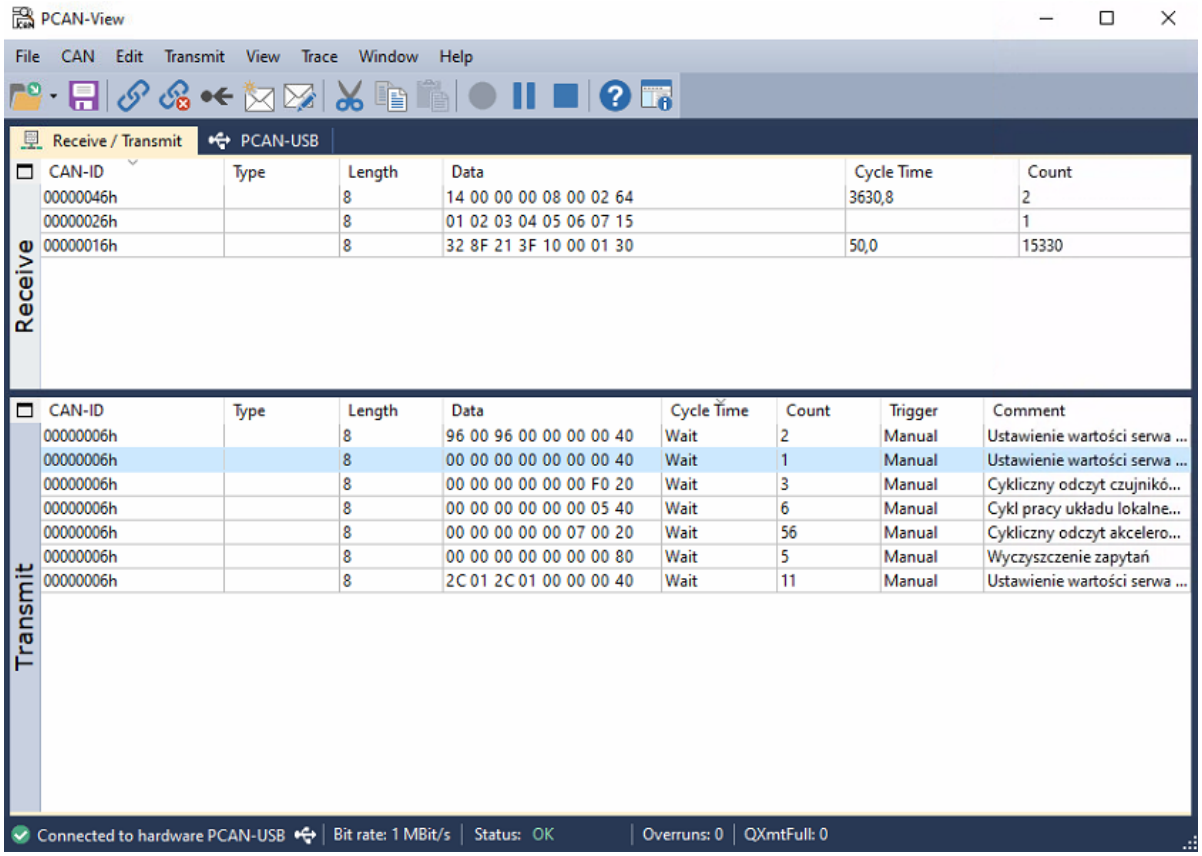


Rysunek 1: Okno nawiązywania połączenia

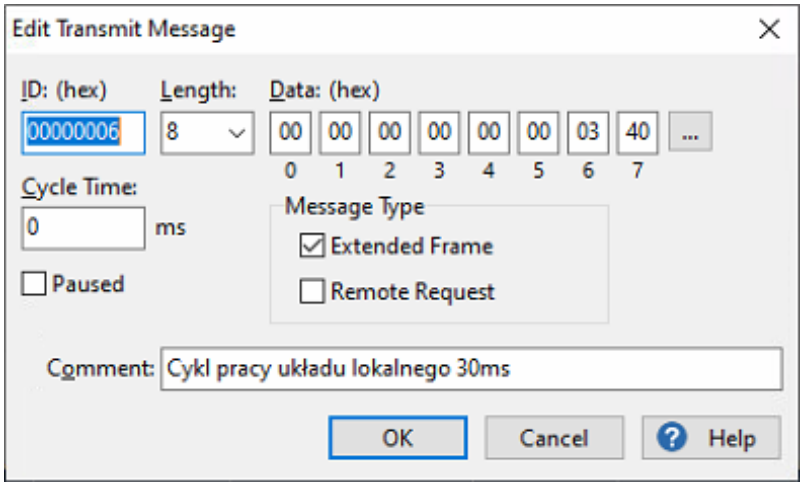
Po udanym połączeniu zostały odebrane trzy ramki danych.

CAN-ID	Type	Length	Data
00000046h		8	14 00 00 00 08 00 02 64
00000026h		8	01 02 03 04 05 06 07 15
00000016h		8	32 8F 21 3F 10 00 01 30

Rysunek 2: Odebrane ramki danych



Rysunek 3: Widok interfejsu w programie PCANView



Rysunek 4: Kreator rozkazów

1.2.2 Ustawienie cyklu pracy układu lokalnego 30ms

Ustawienie cyklu pracy układu lokalnego odbywa się za pomocą 6 bajtu, podana w nim wartość w zakresie 2-20 (dec) zwiększających cykl co 10ms. Bajt 7 odpowiada za ustawienie zadanych wartości w instrukcji. Ramka realizujaca zadanie wygląda następująco:

00000006h 00 00 00 00 00 00 03 40

CAN-ID	Type	Length	Data	Cycle Time
00000046h		8	14 00 00 00 08 00 02 64	3630,8
00000026h		8	01 02 03 04 05 06 07 15	
00000016h		8	31 C0 20 60 10 07 01 47	30,0

1.2.3 Ustawienie cyklicznego odczytu danych z akcelerometru

Ustawienie odczytu danych z akcelerometru odbywa się za pomocą 4 bajtu. Ustawiona wartość 07 jest składową odczytu osi X, Y i Z. Bajt 7 odpowiada za cykliczne odczytywanie pomiarów. Ramka realizujaca zadanie wygląda następująco:

00000006h 00 00 00 00 00 07 00 20

1.2.4 Ustawienie cyklicznego odczytu danych z czujników odległości

Ustawienie odczytu danych z czujników odległości odbywa się za pomocą 6 bajtu. Ustawiona wartość F0 jest składową adresów 4 czujników odległości. Bajt 7 odpowiada za cykliczne odczytywanie pomiarów.

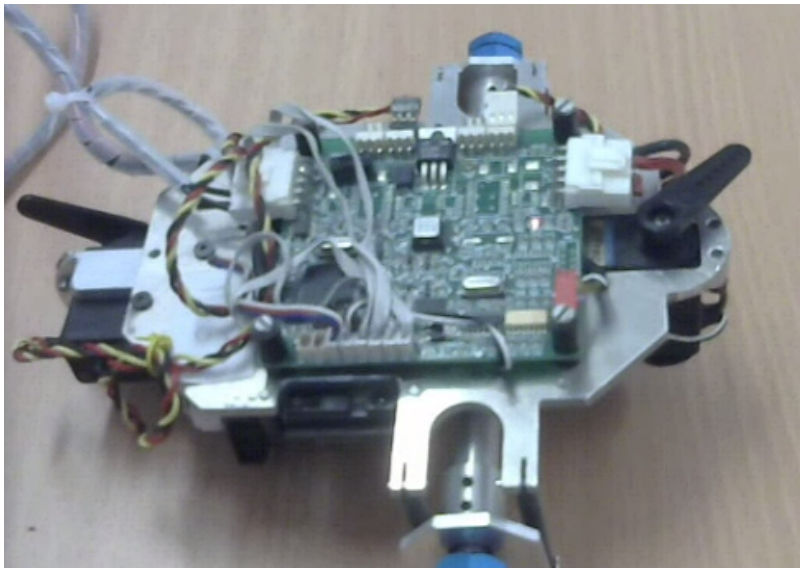
00000006h 00 00 00 00 00 00 F0 20

1.2.5 Zmiana położenie serwa 1 i serwa 2

Zakres roboczy obu serw mieści się między 0 a 300 (dec), naszym zadaniem było ustawienie 3 zadanych pozycji: minimalnej, środkowej oraz maksymalnej. Wysterowanie pojedynczego serwa odbywa się przy pomocy dwóch bajtów. Dla serwa 1 jest to bajt0 oraz bajt1, natomiast dla serwa 2 bajt2 oraz bajt3.

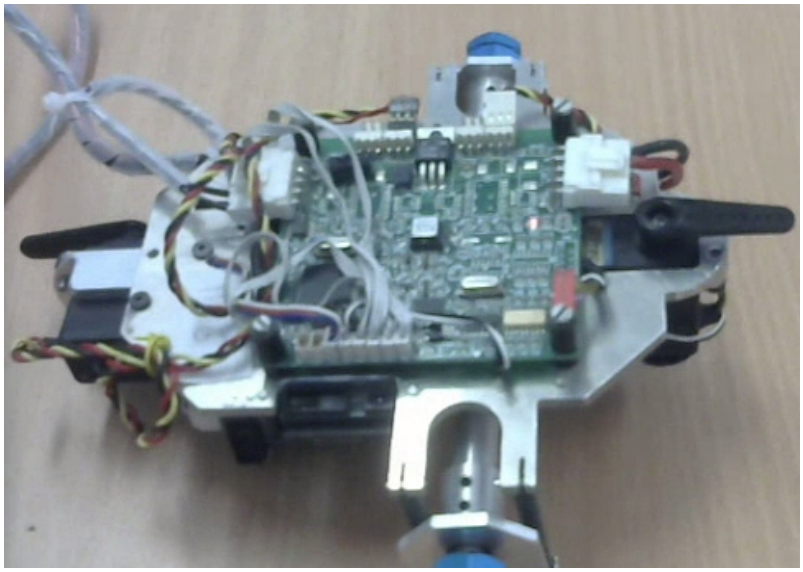
- Maksymalna pozycja serw:

  00000006h 2C 01 2C 01 00 00 00 40

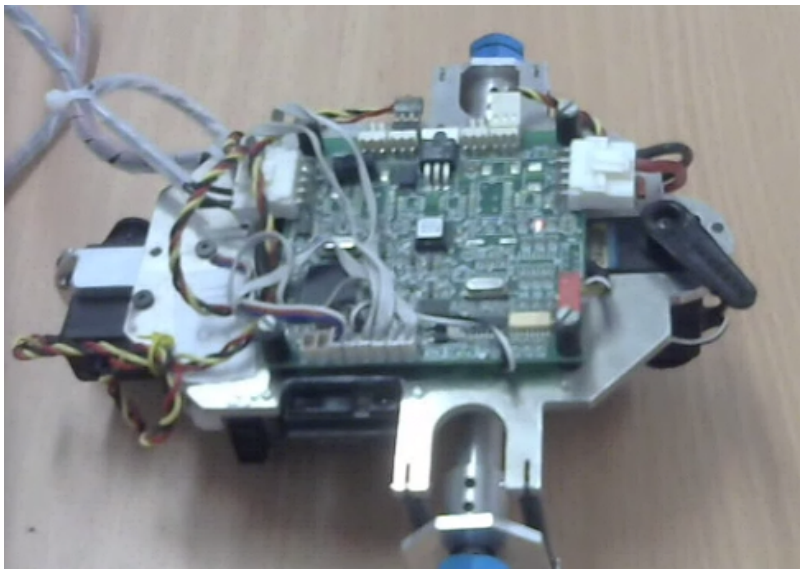


- Środkowa pozycja serw:

  00000006h 96 00 96 00 00 00 00 40



- Minimalna pozycja serw:
  00000006h 00 00 00 00 00 00 00 40



1.2.6 Zmiana cyklu pracy układu lokalnego na 50ms

Analogicznie do punktu 1.2.2, zmianie uległa jedynie zadana długość cyklu.
00000006h 00 00 00 00 00 00 05 40

CAN-ID	Type	Length	Data	Cycle Time
00000046h		8	14 00 00 00 08 00 02 64	3630,8
00000026h		8	01 02 03 04 05 06 07 15	
00000016h		8	32 8F 21 3F 10 00 01 30	50,0

1.3 Wnioski i spostrzeżenia

Oprogramowanie PCANView jest łatwym, czytelnym i intuicyjnym narzędziem. Bezpośrednio pokazuje komunikację za pomocą ramek, dzieląc je na przychodzące i wysyłane. Jest dobrym programem do testowania oraz podglądu transmisji protokołem CAN. Zaletą jest także możliwość wyboru sposobu wysyłania rozkazów - możemy wysłać

ramkę pojedynczo manualnie lub wysyłać cyklicznie w zadanym interwale. Program sygnalizuje status połączenia w protokole.

2 Ćwiczenie 2: Sterowanie manipulatorem za pomocą sterownika CompactRIO (NI)

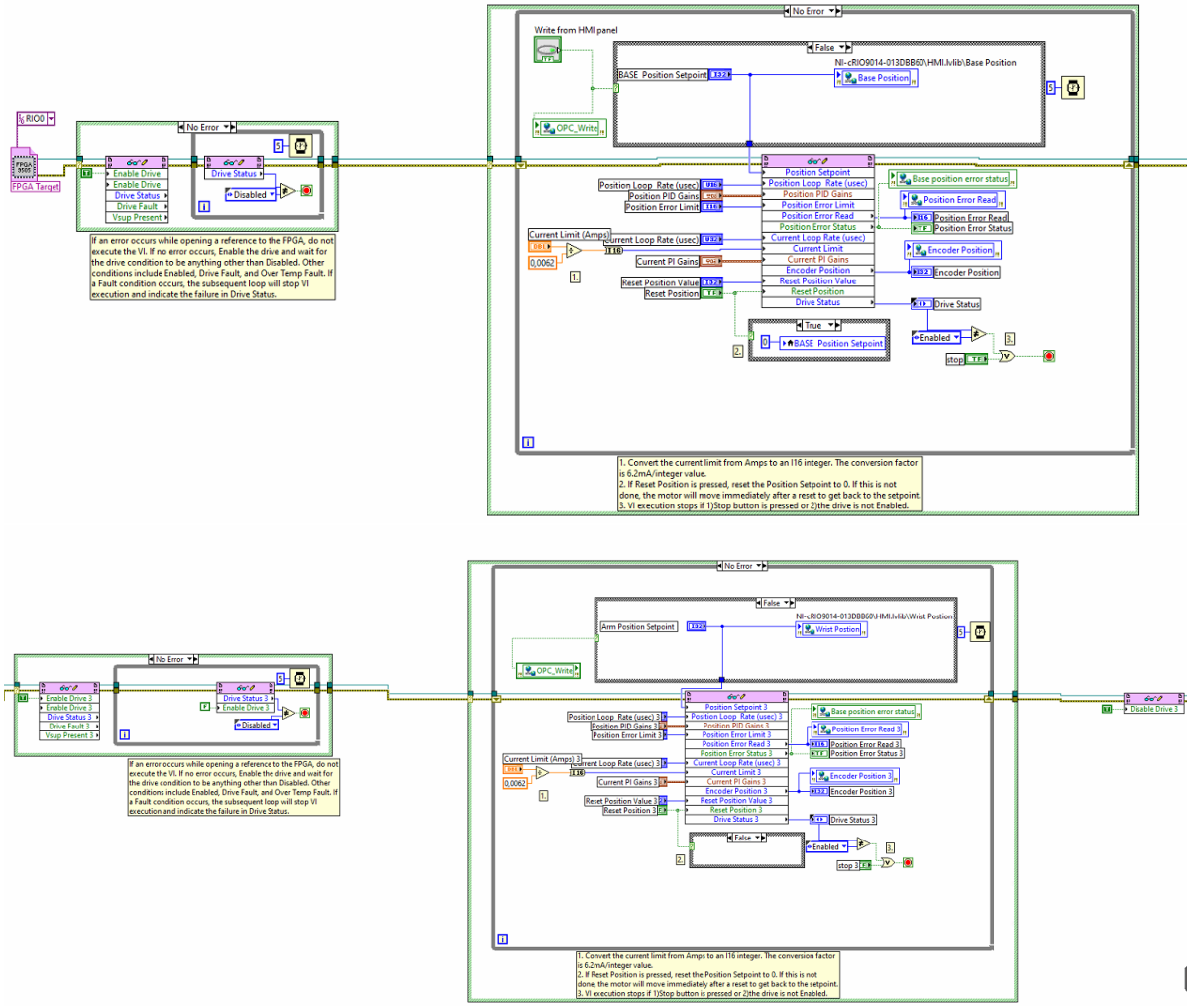
2.1 Cel ćwiczenia

Zapoznanie się z ze strukturą FPGA i Real-Time sterownika CompactRIO jego współpracą z środowiskiem programowym LabVIEW na przykładzie sterowania manipulatorem o trzech stopniach swobody.

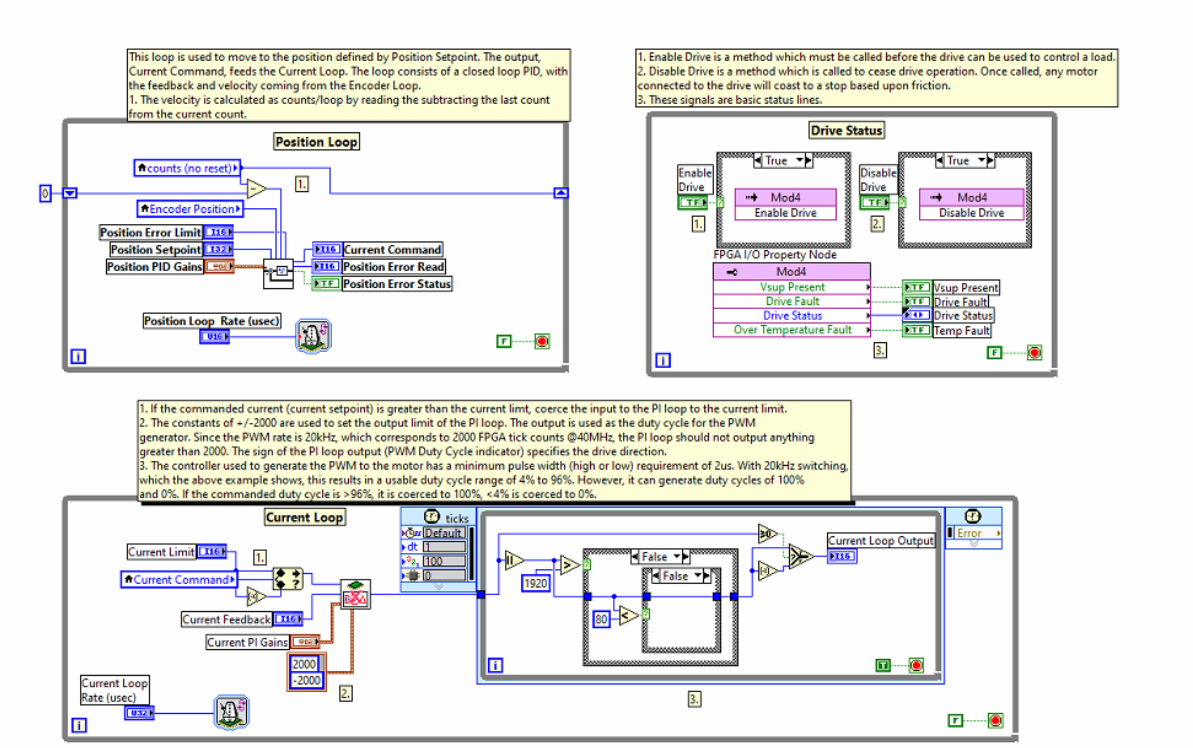
2.2 Realizacja ćwiczenia

2.2.1 Uzupełnienie brakującej części kodu

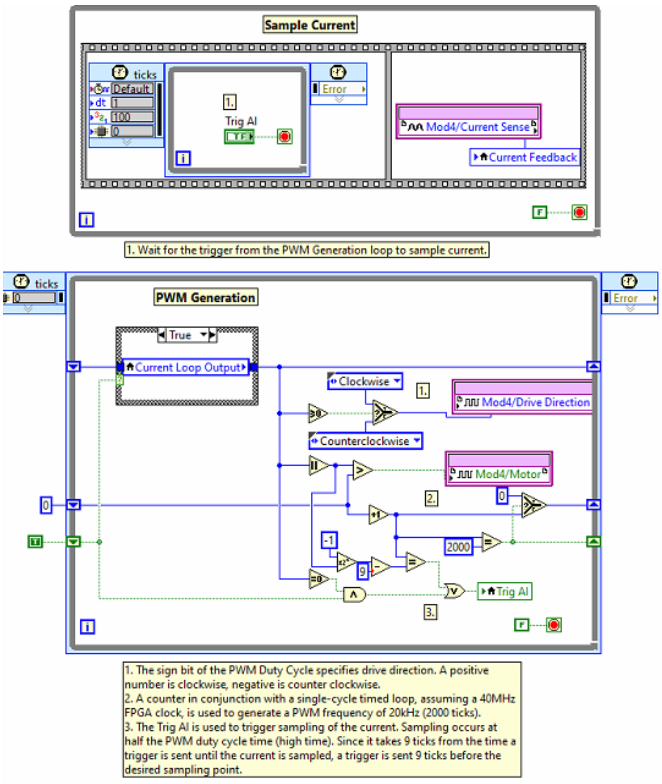
Uzupełnienie brakującej części programu polegało na dodaniu suwaków zadających pozycję poszczególnych złączy manipulatora, których skala została odpowiednio dobrana do zakresu pracy złącz. Dane odbierane z dodanych elementów interfejsu zostają przesłane do bloku FPGA oraz udostępniane jako zmienne globalne środowiska RIO. W celu warunkowego umożliwienia kontroli z poziomu panelu HMI, kod odpowiedzialny za uaktualnianie wartości położenia został objęty blockiem case-if.



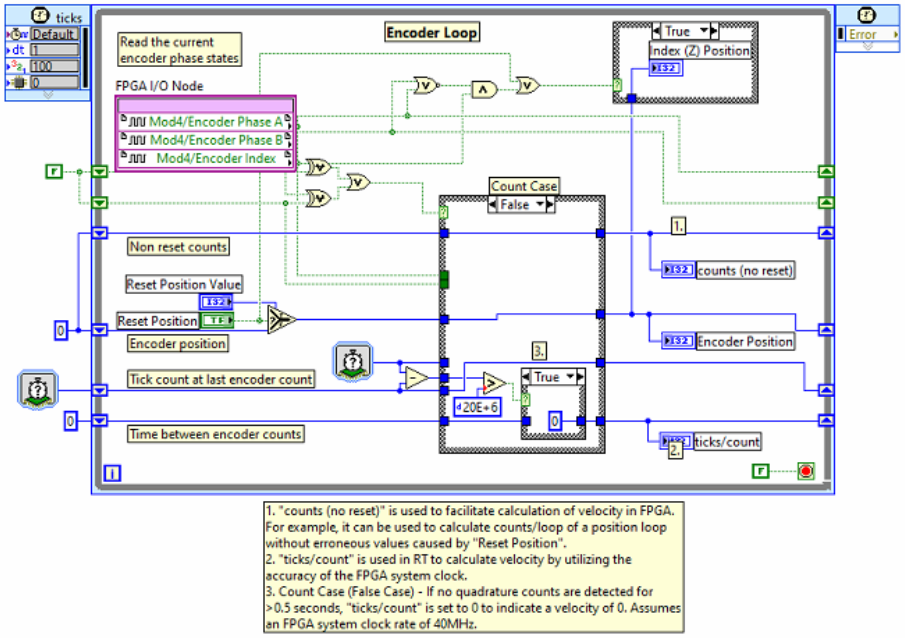
Rysunek 5: Realizacja połączenia panelu kontrolnego użytkownika z blokiem FPGA.



Rysunek 6: Struktura programu FPGA odpowiedzialnego za realizację regulacji pozycji oraz prądu.



Rysunek 7: Struktura programu FPGA odpowiedzialnego za realizację pomiaru prądu oraz generację sygnału sterującego PWM.



Rysunek 8: Struktura programu FPGA odpowiedzialnego za odczyt oraz interpretację wyjść kanałów A i B enkodera.

2.2.2 Dobranie nastaw regulatorów PID dla ramienia oraz nadgarstka robota

Po umożliwieniu zadawania pozycji pozostałym złączom, naszym zadaniem było dobranie dla nich nastaw regulatorów. Ograniczyliśmy się do strojenia regulatorów pozycji. Nasza metodyka ich doboru opierała się na początkowym zmniejszeniu członu proporcjonalnego Kp tak, aby uzyskać pracę złącz pozbawioną oscylacji. Następnie dopiero dobieraliśmy Kp i Kd tak, aby przyśpieszyć proces regulacji, bez ponownego wprowadzenia złącz w stan oscylacji. Człon całkujący okazał się niepotrzebny w regulacji pozycji, układ był w stanie osiągnąć wartość zadaną bez znaczącego błędu statycznego.

Rysunek 9: Widok panelu kontrolnego sterowników poszczególnych złącz wraz z dobranymi nastawami regulatorów

2.3 Wnioski i spostrzeżenia

Nastawy regulatorów zostały dobrane w taki sposób, aby ruch poszczególnych złącz manipulatora do pozycji zadanych odbywał się w jak najkrótszym czasie, jednocześnie nie wywołując oscylacji.

Za szybka zmiana zadanego położenia powodowała gwałtowny skok błędu odczytu enkodera, który był sygnalizowany przez kontrolkę w panelu użytkownika. Błąd ten skutkował zatrzymaniem pracy złącza, czego efektem było niezrealizowanie zadanego położenia.

Podczas realizacji ćwiczenia zauważyliśmy, że zmienne pozycyjne nadgarstka i ramienia są ze sobą zamienione. Może to być spowodowane błędem definicji zmiennych globalnych lub niewłaściwym podłączeniem silników do sterownika.

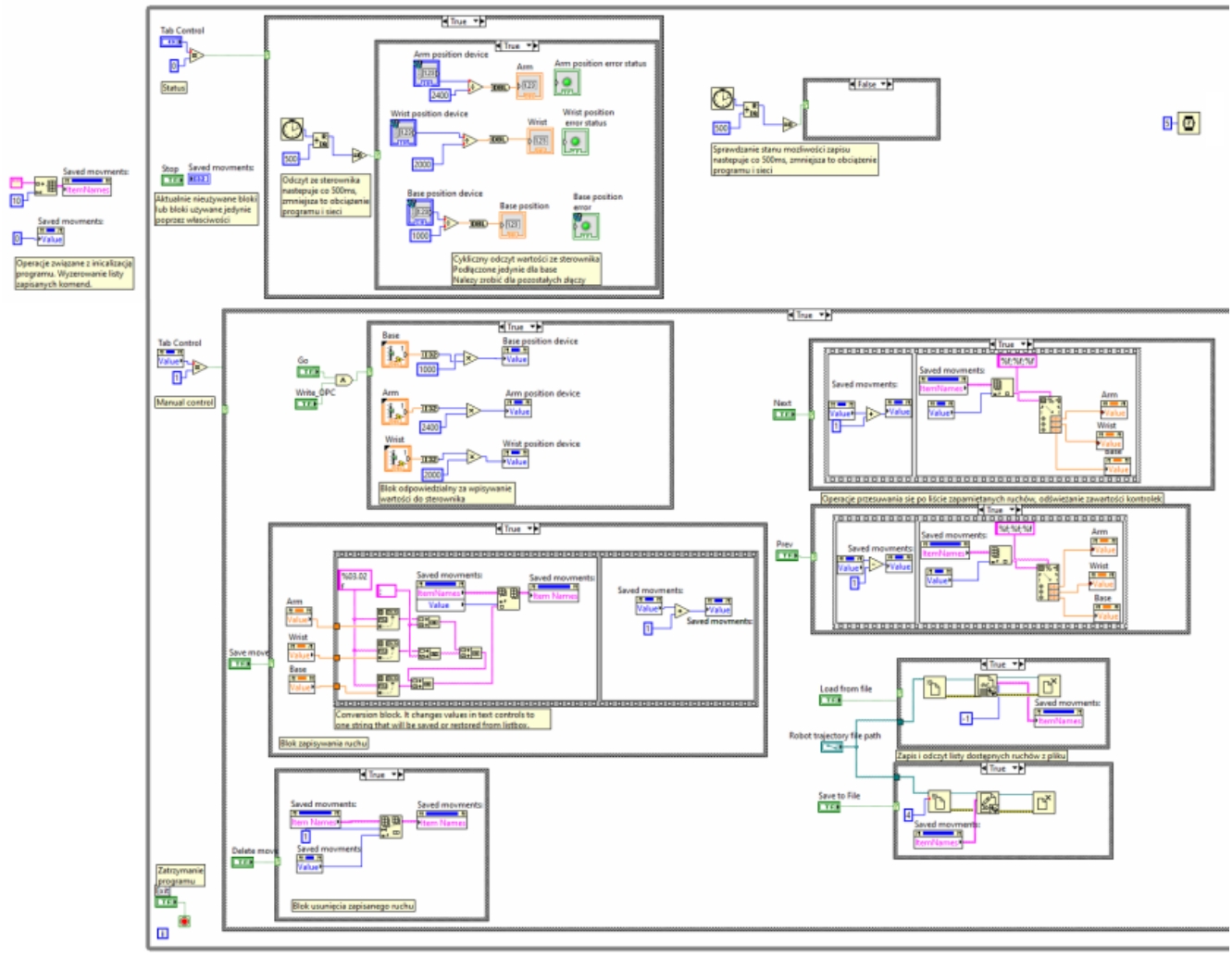
3 Ćwiczenie 3: Sterowanie robota za pomocą panelu operatorskiego sprzężonego przez sieć Ethernet ze sterownikiem CompactRIO.

3.1 Cel ćwiczenia

Celem ćwiczenia było zapoznanie się z sterowaniem systemem (w tym wypadku robotem) przy wykorzystaniu dodatkowego urządzenia wyjściowego komunikującego się z sterownikiem za pomocą zmiennych współdzielonych. W trakcie ćwiczenia uzupełnialiśmy kod LabView realizujący funkcjonalność panelu HMI.

3.2 Realizacja ćwiczenia

3.2.1 Kod programu



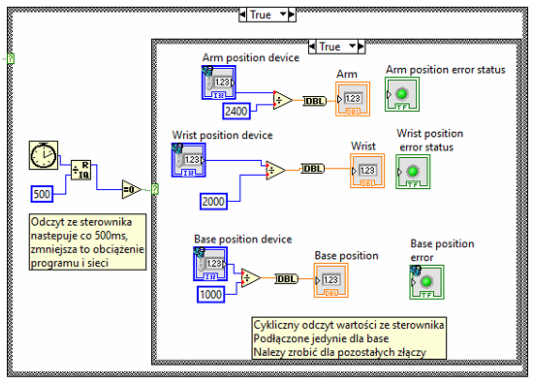
Rysunek 10: Diagram pogramu LabView

Powyższy diagram implementuje funkcjonalność panelu HMI. Panel operatorski posiada trzy zakładki:

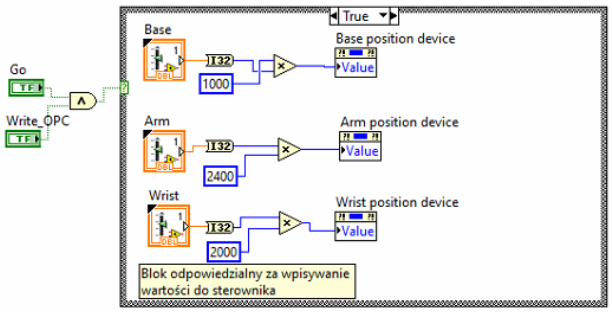
- Status - węwnątrz tej zakładki wyświetlane były obecne pozycje oraz statusy złącz manipulatora.
- Manual control - zakładka pozwalała na zadawanie pozycji złącz manipulatora oraz zapisywanie/odczytywanie historii wykonywanych operacji do pliku.
- Settings - zakładka pozwalała na wprowadzenie ścieżki pliku zarówno do zapisu jak i odczytu trajektorii.

Warto zaznaczyć, że sterowanie ręczne z poziomu 'panelu HMI' było możliwe dopiero po przejściu sterownika w tryb działania manualnego.

W trakcie laboratorium zadaniem naszej grupy było dokończenie dwóch bloków funkcjonalnych diagramu, odpowiedzialnych za zapis oraz odczyt pozycji poszczególnych złącz z sterownika.

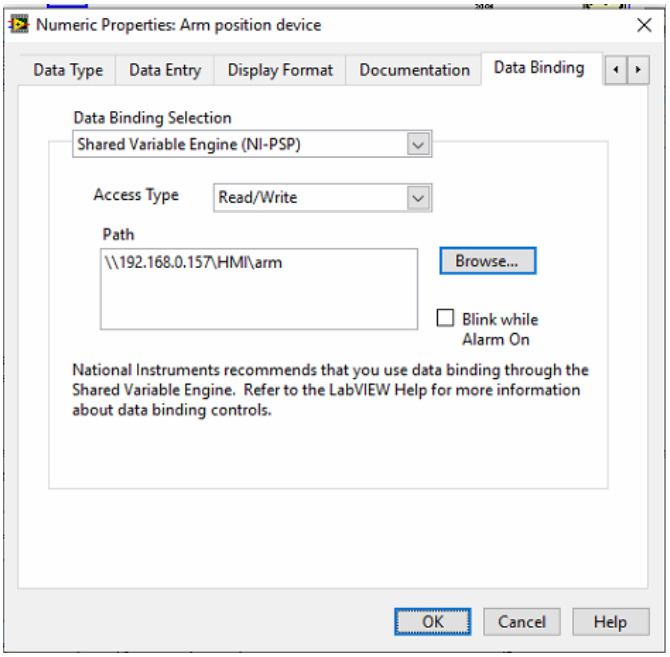


Rysunek 11: Kod odpowiedzialny za odczyt pozycji złącz manipulatora



Rysunek 12: Kod odpowiedzialny za zapis pozycji złącz manipulatora

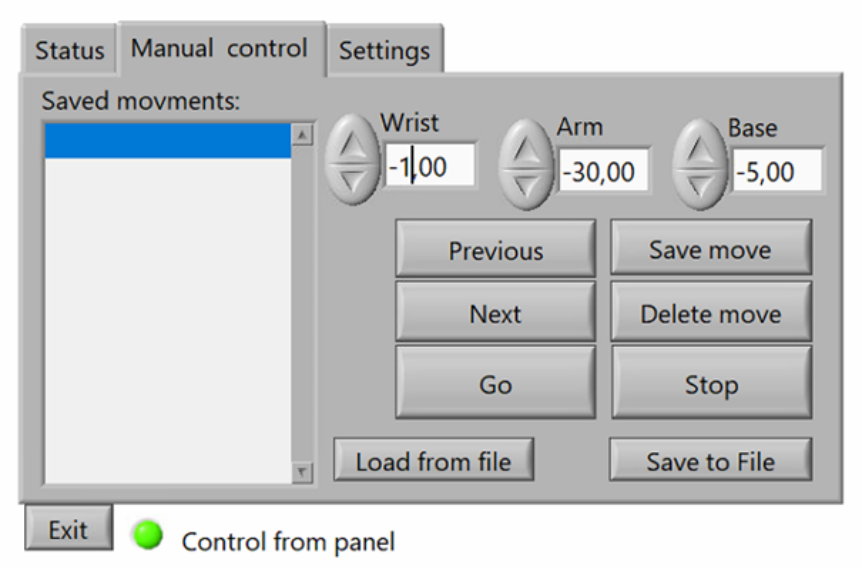
W ramach realizacji kodu, musieliśmy dodać dwie brakujące zmienne złączowe Arm oraz Wrist, a także powiązać je z odpowiednimi zmiennymi współdzielonymi. Musieliśmy również dokonać odpowiedniego przeskalowania wartości zmiennych zarówno wysyłanych do jak i odbieranych od sterownika. Odpowiednie wartości skalujące zostały wyznaczone na podstawie zakresów zmiennych złączowych podanych w instrukcji laboratorium.



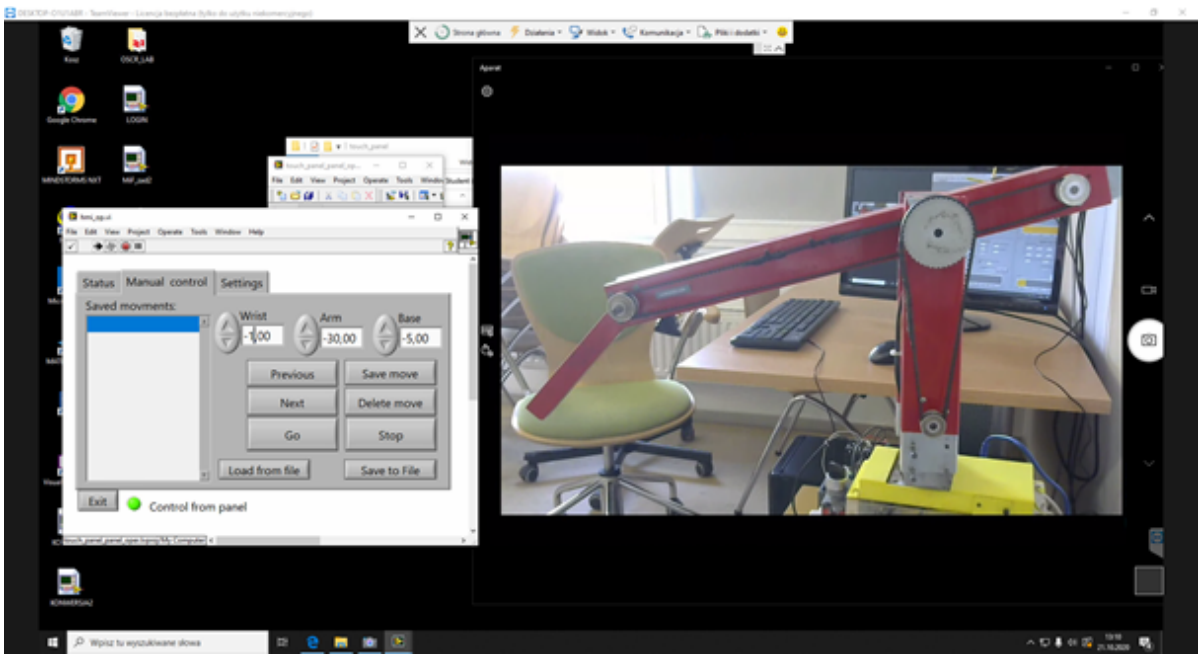
Rysunek 13: Powiązywanie zmiennych Arm oraz Wrist z odpowiednimi zmiennymi współdzielonymi

3.2.2 Gotowa aplikacja

Gotowa aplikacja panelu HMI już pozwalała nam na zadawanie pozycji poszczególnym złączom manipulatora. Sprawne były również pozostałe wcześniej opisane funkcjonalności panelu operatorskiego m.in. zapis historii operacji.



Rysunek 14: Widok panelu HMI



Rysunek 15: Widok panelu oraz sterowanego robota

3.3 Wnioski i spostrzeżenia

Komunikacja ze sterownikiem z poziomu panelu operatorskiego funkcjonowała prawidłowo. Nie mieliśmy jednak niestety wpływu na implementacje programu sterownika - w trakcie testów zaobserwowaliśmy, że zmienne złączowe Wrist i Arm były ze sobą zamienione. Przy źle dobranych nastawach regulatorów pozycji poszczególnych złącz, dało się zaobserwować liczne oscylacje przy dochodzeniu do pozycji zadanych. Wnioskujemy również, że w trakcie pracy nad relatywnie dużym projektem programistycznym, działającym czasem na wielu różnych urządzeniach, przydatne jest ustalenie jednolitej konwencji nazewnictwa zmiennych występujących w projekcie. Zauważyliśmy również, że korzystanie ze zmiennych współdzielonych jest prostą w implementacji metodą komunikacji między urządzeniami.