

1. Spis treści

1.	SPIS TREŚCI	1
2.	WSTEP	3
3.	OPIS STANOWISKA LABORATORYJNEGO	4
3.1.	Budowa stanowiska laboratoryjnego	4
3.2.	Falownikowy układ napędu	7
3.2.1.	Budowa układu	7
3.2.2.	Charakterystyka falownika Micromaster	7
3.2.3.	Panel operatorski OPM2	10
3.2.4.	Filtr typu RFI	11
3.2.5.	Blok stycznika	11
3.2.6.	Układ sterowania lokalnego	13
4.	UKŁAD POMIAROWY	14
4.1.	Właściwości układu	14
4.2.	Budowa i opis układu	14
4.3.	Zasilacz stabilizowany napięcia stałego ± 15 V	18
4.4.	Moduł pomiarowy	18
5.	UKŁAD MONITORINGU I STEROWANIA ZDALNEGO	21
5.1.	Budowa i opis układu	21
5.2.	Moduł wejść / wyjść cyfrowych	22
5.2.1.	Opis i właściwości modułu	24
5.2.2.	Zasada działania modułu	24
5.3.	Karta zbierania danych	25
5.3.1.	Opis, budowa i właściwości karty	25
5.3.2.	Ustawienia, konfiguracja i instalacja karty	30
5.4.	Komputer nadzorujący i sterujący pracą falownikowego układu napędu	32
6.	MONITORING I STEROWANIE Z WYKORZYSTANIEM KOMPUTERA	33
6.1.	Wstęp	33
6.2.	Komunikacja komputera z falownikiem	33
6.2.1.	Parametry transmisji	34
6.2.2.	Budowa telegramów	34
6.2.3.	Programowy interfejs komunikacyjny	39
6.3.	Sterowanie kartą zbierania danych	41
6.4.	Opis programu „Micro”	44
6.4.1.	Wstęp	44
6.4.2.	Opis poszczególnych okien programu	45
6.4.3.	Algorytm działania programu	56
6.4.4.	Przykłady działania programu	60
7.	MONITORING I STEROWANIE Z WYKORZYSTANIEM SIECI KOMPUTEROWEJ INTRANET / INTERNET	65
7.1.	Wstęp do komunikacji sieciowej	65
7.2.	Aplikacje „Klient” ↔ „Serwer”	67

7.2.1. Wstęp	67
7.2.2. Opcja Server programu „Micro”	68
7.2.3. Opis i algorytm działania Server’a	71
7.2.4. Aplikacja „Client TCP/IP”	74
7.2.5. Opis i algorytm działania programu „Client TCP/IP”	78
7.2.6. Przykłady działania programu Client’a	80
7.3. „Klient WWW”	83
7.3.1. Zadania stawiane aplikacji	83
7.3.2. Dlaczego Java?	83
7.3.3. Opis programu „Klient WWW”	84
7.3.4. Opis programu do rysowania przebiegów – ChartApplet	100
8. WNIOSKI I UWAGI KOŃCOWE	105
9. BIBLIOGRAFIA	107
10. ANEKS	109
10.1. Zestawienie elementów	109
11. DODATKI	111

2. Wstęp

W obecnej chwili na świecie pracuje ogromna ilość autonomicznych urządzeń sterujących układami automatyki oraz urządzeniami kontrolno pomiarowymi. Urządzenia i systemy nadzorujące procesy przemysłowe stają się coraz bardziej rozproszone. Tym trudniejsza staje się kontrola nad nimi. Gdy kontrolowane obiekty zajmują względnie niewielką przestrzeń (teren zakładu, budynku, itp.) do wymiany informacji między obiektami można wykorzystać protokoły takie jak Ethernet czy Profibus oraz aplikacje dostarczane przez producentów urządzeń. Jednak w przypadku systemów rozproszonych niezbędnym staje się wykorzystanie protokołów takich jak TCP/IP czy UDP, mogących wymieniać dane pomiędzy urządzeniami pracującymi w różnych topologiach sieciowych oraz aplikacjami sterująco-monitorującymi z dowolnego miejsca i pracującymi na dowolnym komputerze. W takiej sytuacji, niezbędna jest rozległa sieć łącząca kontrolowany obiekt z miejscem, z którego dokonywane jest sterowanie i nadzór. Idealnym rozwiązaniem jest publicznie dostępna sieć Internet oraz protokół TCP/IP, wraz z dedykowaną aplikacją potrafiącą wymieniać dane pomiędzy punktem sterowania a punktem sterowanym. Aplikacja pozwala operatorowi na sterowanie i monitoring z jednego dowolnego miejsca grupą bądź też pojedynczym układem lub procesem technologicznym. Dzięki temu, znacznie wzrasta szybkość reakcji na zmiany zachodzące w kontrolowanych obiektach oraz zwiększa się łatwość porównywania i zestawiania informacji pochodzących od wielu obiektów znacznie od siebie oddalonych.

Celem niniejszej pracy było opracowanie monitoringu i sterowania układu automatyki z wykorzystaniem nowych możliwości jakie oferuje, dynamicznie rozwijająca się sieć komputerowa Intranet/Internet. Dla potrzeb dydaktycznych zostało zaprojektowane i zbudowane stanowisko laboratoryjne, w skład którego wchodzi następujące elementy:

- falownikowy układ napędu,
- układ pomiarowy,
- układ monitoringu i sterowania zdalnego.

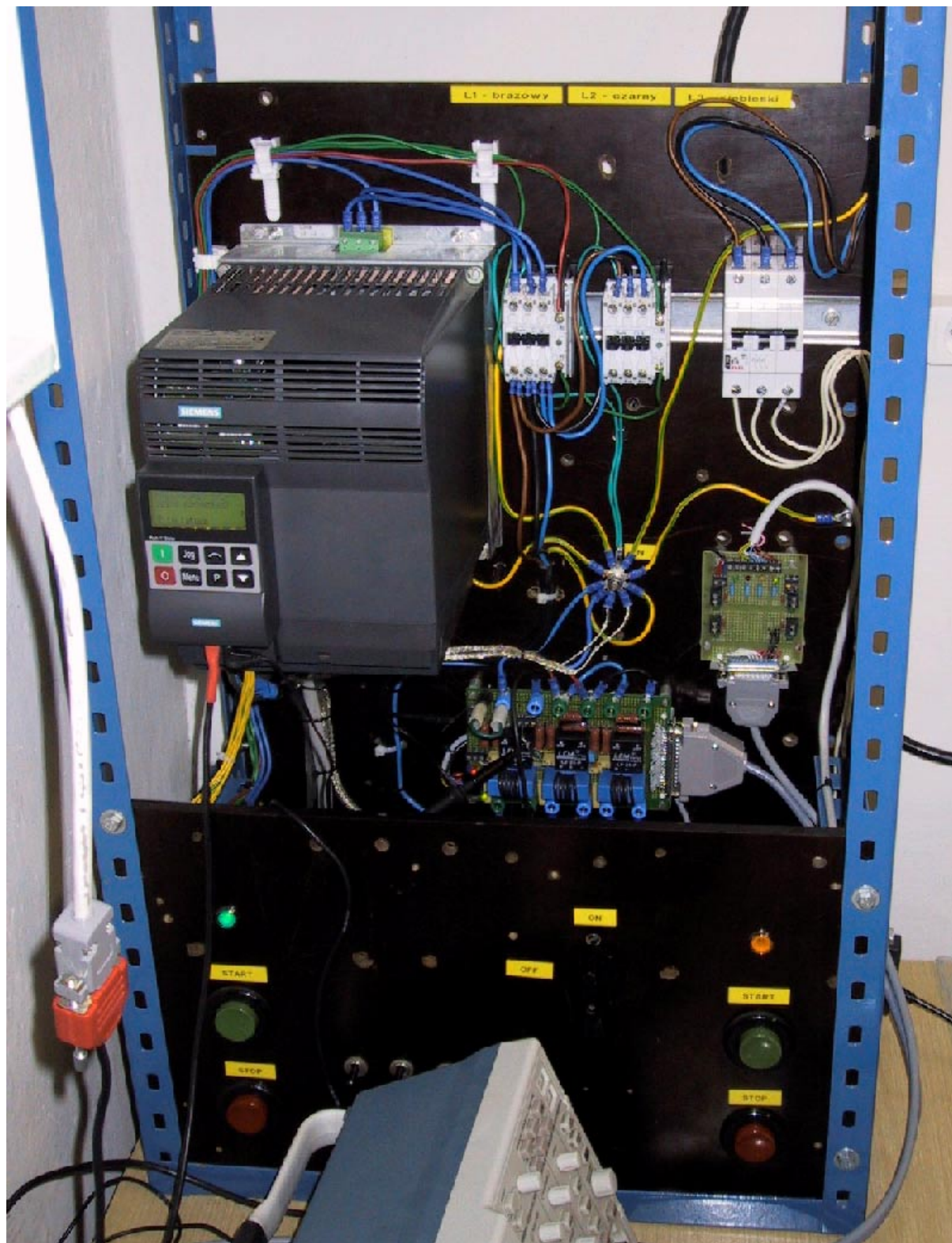
Realizacja komunikacji przez sieć komputerową wymagała również wykonania dedykowanego oprogramowania umożliwiającego:

- komunikację sygnałową między falownikiem a komputerem
- zdalny monitoring pracy i sterowania falownika przez komputer
- monitoring i sterowanie układu napędowego poprzez sieć lokalną („zakładową”) Intranet i sieć globalną Internet .

Budowa stanowiska i oprogramowania ma na celu udostępnienie nowych możliwości w poznawaniu działania układów automatyki napędu i praktyczne sprawdzenie ich właściwości. Dostęp do sieci Internet jest powszechny. Sterowanie i monitoring z dowolnego miejsca na świecie, w dowolnej chwili, daje duże możliwości nie tylko w stanowiskach dydaktycznych, ale także w zastosowaniach przemysłowych.

3. Opis stanowiska laboratoryjnego

3.1. Budowa stanowiska laboratoryjnego



Zdjęcie 1. Widok stelaża stanowiska laboratoryjnego



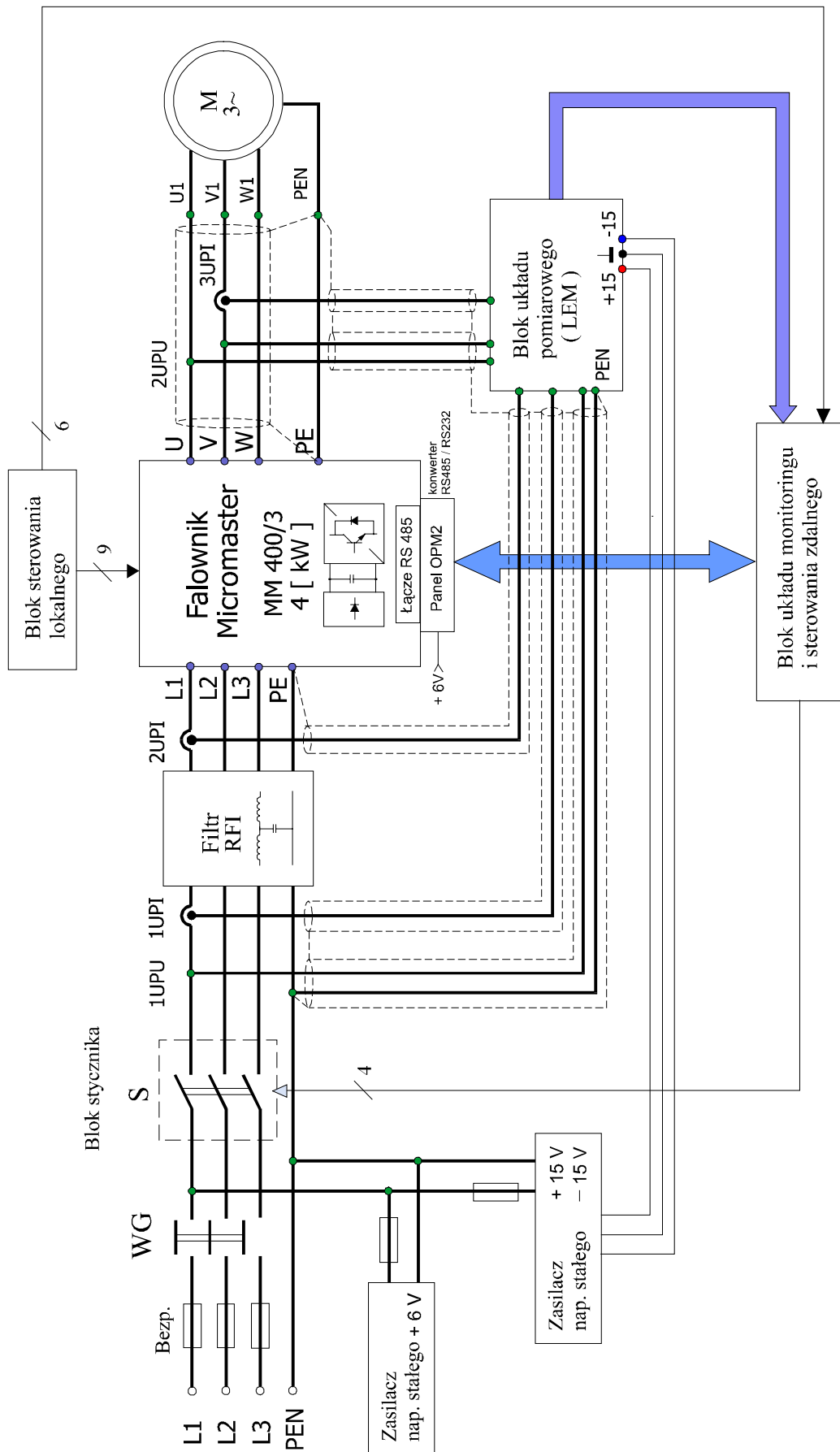
Zdjęcie 2. Widok całego stanowiska laboratoryjnego

Stanowisko laboratoryjne jest zbudowane ze stelaża zawierającego:

- falownikowy układ napędowy sterujący pracą silnika,
- układ pomiarowy,
- układy monitorujące i sterujące

oraz komputera stanowiącego główną część układu monitoringu i sterowania zdalnego, na którym działające oprogramowanie umożliwia:

- przetwarzanie i wizualizację danych pomiarowych,
- zdalne sterowanie i nadzór pracy całego układu,
- monitoring i sterowanie przez sieć komputerową Intranet/Internet.



Rys. 1. Schemat ideowo - blokowy stanowiska laboratoryjnego

3.2. Falownikowy układ napędu

3.2.1. Budowa układu

Falownikowy układ napędu został zbudowany z wykorzystaniem następujących elementów:

- falownika typu „Micromaster” firmy SIEMENS zasilanego z sieci trójfazowej 3 x 380 V,
- silnika indukcyjnego,
- filtru wejściowego typu RFI,
- układu stycznika umożliwiającego załączanie i odłączenie układu napędowego, do sieci zasilającej,
- układu sterowania lokalnego,
- panelu operatorskiego OPM2,
- zasilacza stabilizowanego napięcia stałego +6 V.

W celu zmniejszenia wpływu zakłóceń elektromagnetycznych falownika, rozchodzących się w przewodach zasilających jak i emitowanych na inne bloki układów stanowiska laboratoryjnego, zastosowano dwustronne ekranowanie tych przewodów (Rys. 1).

Parametry trójfazowego silnika indukcyjnego M 3~ :

- silnik klatkowy firmy WIEFAMEL,
- typ STKe 90 sk,
- znamionowa moc silnika: $P_N = 1,5 \text{ kW}$,
- znamionowe napięcie: $U_N = 380 \text{ V}$,
- znamionowa prędkość: $n_N = 1420 \text{ obr/min}$,
- znamionowy prąd: $I_N = 3,4 \text{ A}$,
- współczynnik mocy: $\cos\varphi = 0,8$,
- klasa izolacji: B,
- rodzaj pracy: S1.

3.2.2. Charakterystyka falownika Micromaster



Zdjęcie 3. Wygląd falownika

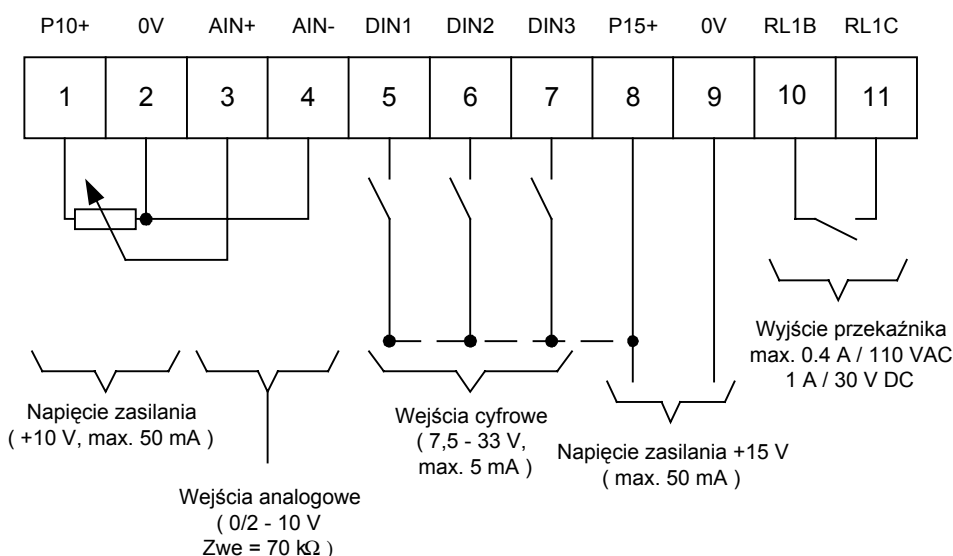
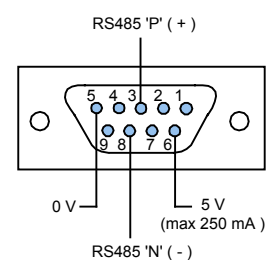
„Micromaster” jest rodziną falowników, przeznaczoną do regulacji prędkości obrotowej trójfazowych silników indukcyjnych prądu przemiennego. Jest urządzeniem sterowanym mikroprocesorowo, zbudowanym w oparciu o zaawansowaną technologię tranzystorów IGBT. Zastosowana w przemiennikach częstotliwości serii „Micromaster” metoda sterowania, modulacji szerokości impulsów (PWM – *Pulse Width Modulation*) z możliwością wyboru wielkości częstotliwości nośnej umożliwia, bardzo cichą pracę silnika. W procesie sterowania metodą PWM zmiana wartości napięcia zasilania jest realizowana wg funkcji $U/f = \text{const}$. Możliwości programowe falownika zapewniają funkcje zabezpieczenia silnika oraz falownika.

Właściwości falownika :

- łatwa instalacja, programowanie i eksploatacja,
- możliwość zastosowania pętli sprzężenia zwrotnego wykorzystującej wbudowany programowo algorytm regulatora liniowego PI,
- wysoki startowy moment obrotowy silnika,
- możliwość zdalnego sterowania do 31 falowników jednocześnie za pośrednictwem łącza szeregowego typu RS 485 przy użyciu protokołu komunikacji USS,
- możliwość przystosowania falownika do większości wymagań poszczególnych aplikacji, dzięki zastosowaniu obszernego zakresu nastaw programowych,
- łatwość w obsłudze panelu operatorskiego z wyświetlaczem cyfrowym i z membranowymi przyciskami umożliwiającymi wielokrotną zmianę nastaw w warunkach przemysłowych,
- wbudowana, baterijnie podtrzymywana i odporna na zakłócenia pamięć, zachowująca wybrane przez użytkownika wartości nastaw,
- możliwość regulacji częstotliwości napięcia wyjściowego falownika (a tym samym prędkości obrotowej silnika) poprzez użycie:
 - klawiatury na pulpicie operatorskim falownika,
 - analogowego wejścia napięciowego,
 - zewnętrznego potencjometru precyzyjnego,
 - wejść dwustanowych do zadawania skokowego,
 - łącza szeregowego.
- wbudowany mechanizm hamowania silnika prądem stałym,
- możliwość programowania dynamiki narastania i zmniejszania częstotliwości,
- wbudowany programowalny przekaźnik wyjściowy – możliwość ustawienia jego reakcji na działanie falownika,
- możliwość zastosowania dodatkowego wyświetlacza tekstowego (OPM2) oraz zewnętrznego modułu komunikacji sieci przemysłowej (PROFIBUS),
- integralny wentylator chłodzący sterowany mikroprocesorowo,
- szybkie ograniczenie prądowe pozwalające na niezawodną ochronę silnika,
- kompaktowe wykonanie, pozwalające na montaż kilku falowników blisko siebie na bardzo małej przestrzeni.

Parametry zastosowanego falownika :

Micromaster	typ MM400/3
Napięcie wejściowe	3 AC 380 - 500 ±10% V
Moc silnika	4,0 kW
Moc pozorna	7,0 kVA
Znamionowy prąd wyjściowy (380/500)V	9,2 / 8,1 A
Maksymalny wyjściowy prąd ciągły	10,4 A
Znamionowy prąd wejściowy	13,6 A
Częstotliwość zasilania	47 – 63 Hz
Częstotliwość wyjściowa	0 – 400 Hz
Współczynnik mocy	≥ 0,7
Przebieżalność prądowa	1,5 I _n przez 60 s.
Rozdzielczość cyfrowa nastaw	0,01 Hz
Czas rozruchu i hamowania	0 – 650 s
Ochrona przeciw:	- przegrzaniu silnika, - przegrzaniu falownika, - przetężeniom i przepięciom, - zwarciom międzyfazowym i doziemnym,
Ochrona dodatkowa przeciwko:	- pracy bez obciążenia w układzie otwartym,
Kontrola przegrzania silnika	obliczana całka I ² t
Zakres regulacji	cztery kwadranty
Regulacja	sygnałem prądowym lub napięciowym
Zakres sygnałów wejść analogowych	(0 –10) V lub (2 –10) V
Rozdzielczość wejść analogowych	10 bitowa
Interfejs cyfrowy	szeregowy typu RS 485

Listwa sterująca falownika**Panel przedni
złącze RS 485 typu D**

Rys. 2. Złącza falownika

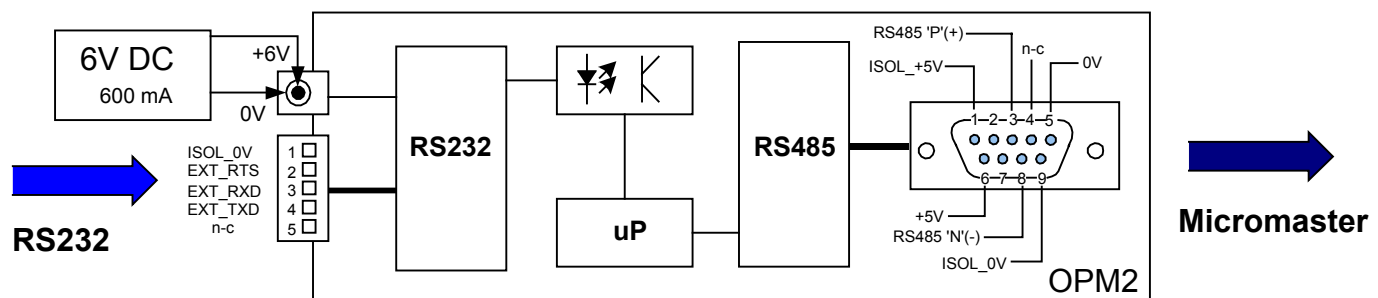
Rysunek 2, przedstawia opis listwy sterującej przekształtnika oraz jego złącza RS485.

3.2.3. Panel operatorski OPM2

OPM2 zbudowany jest w postaci wyświetlacza ciekłokrystalicznego i klawiatury sterującej z przyciskami membranowymi. Może być montowany bezpośrednio na obudowie przekształtnika lub też za pośrednictwem odpowiedniego przewodu na obudowie szafy sterowniczej. Pozwala on na sprawniejsze programowanie pojedynczego lub do 10 falowników połączonych w sieć (za pośrednictwem łącza szeregowego RS485). Wyświetlacz ten posiada możliwość wyświetlania komunikatów o błędach i opisów poszczególnych nastaw, a także wyświetlania do kilku wartości naraz jakie dostarcza falownik (np. napięcia zasilającego silnik, częstotliwości wyjściowej, wyjściowego prądu przewodowego, prędkości itp). Zastosowana w nim pamięć pozwala na trwałe przechowywanie do 31 pełnych zestawów wartości nastaw, z możliwością ich automatycznego odczytania lub załadowania z/do pamięci wybranego przekształtnika.

Dzięki wbudowanemu konwerterowi RS485/RS232 (Rys.3), po podłączeniu dodatkowego 6V źródła zasilania, możliwe jest podłączenie sieci przekształtników z komputerem nadzorującym klasy IBM PC poprzez port szeregowy COM.

W falownikowym układzie napędu panel operatorski pełni rolę konwertera, umożliwiając monitoring i sterowanie pracą falownika za pomocą komputera i dedykowanego oprogramowania.



Rys. 3. Schemat blokowy konwertera RS232/RS485

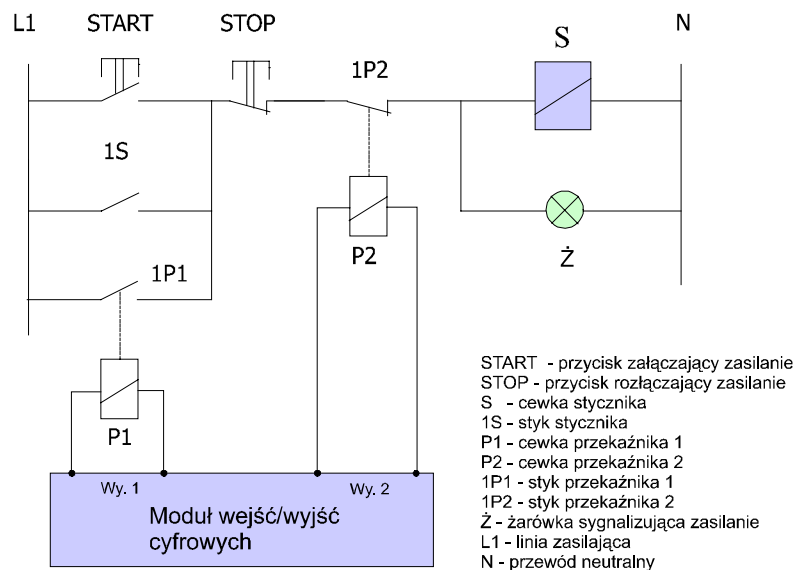
Parametry zastosowanego zasilacza stabilizowanego napięcia stałego +6 V:

- zasilacz firmy TATAREK,
- typ ZS 6V
- napięcie wejściowe: 220 V ~50Hz
- moc pobierana: 12 VA
- napięcie wyjściowe: 6 V napięcia stałego,
- obciążalność prądowa: 600 mA,
- klasa izolacji: B

3.2.4. Filtr typu RFI

Filtr typu RFI zastosowany w układzie jest zalecany przez firmę SIMENS do tego typu falowników w sytuacjach gdzie układ napędowy pracuje w środowisku wrażliwym na zakłócenia elektromagnetyczne rozchodzące się zarówno drogą polową jak i obwodową. Takie zakłócenia emitowane przez przekształtnik mogą niekorzystnie wpływać na poprawność pracy innych urządzeń sąsiadujących lub też fragmentów układu napędowego. Zastosowany filtr wg specyfikacji [1] zapewniał kompatybilność falownika w klasie 1 i 2 w wymaganiach ogólnych dla przemysłu oraz w wymaganiach dla przemysłu ciężkiego.

3.2.5. Blok stycznika



Rys. 4. Schemat blokowy układu styczników

Blok stycznika przedstawiony na rysunku 4, służy do załączania głównego obwodu zasilającego falownika. Może odbywać się to poprzez lokalny panel sterujący stanowiska laboratoryjnego jak też z wykorzystaniem komputera i sygnałów wyjściowych z modułu wejść/wyjść cyfrowych.

Załączenie z lokalnego panelu sterowania odbywa się przyciskiem „START” załączającym cewkę stycznika 3-fazowego S. Powoduje to załączenie głównego obwodu zasilania przedstawionego na rysunku 1.

Styk 1S jest stykiem podtrzymującym zasilanie cewki stycznika. Przycisk „STOP” odłącza cewkę stycznika S, wyłączając tym samym główny obwód zasilania. Użyta w bloku stycznika żarówka sygnalizuje stan załączenia głównego obwodu zasilania.

Do wyjść cyfrowych modułu (WY.1,WY.2) zostały podłączone cewki dwóch przekaźników: P1 i P2. Styk 1P1 (zwierny) podłączono równolegle z przyciskiem „START”, natomiast styk 1P2 (rozwierny) włączono w szereg z przyciskiem „STOP” i cewką S stycznika.

Takie podłączenie przekaźników umożliwiło zachowanie tego samego systemu sterowania stycznikiem dla obu metod. Cewki użytych przekaźników przystosowane są do zasilania napięciem stałym. W momencie kiedy na wyjściu pierwszym modułu (WY.1) pojawia się napięcie (+15V), styk 1P1 przekaźnika pierwszego zwiera się i tym samym załącza napięcie na cewkę stycznika 3-fazowego S, co powoduje załączenie głównego obwodu zasilania. Po załączeniu styku 1P1, napięcie z cewki przekaźnika P1, może zostać zdjęte i nie spowoduje to rozłączenia obwodu zasilania. Kiedy na wyjściu drugim modułu (WY.2) pojawi się napięcie (+15V), styk 1P2 zostaje rozarty i przerywa dopływ napięcia do cewki stycznika S. Powoduje to rozłączenie głównego obwodu zasilania falownika. Ponowne załączenie obwodu możliwe będzie po zdjęciu napięcia z cewki przekaźnika P2. Kontrolę nad sterowaniem przekaźników sprawuje komputer i działające na nim oprogramowanie, które nadzoruje pracę wyjść cyfrowych karty zbierania danych.

Parametry zastosowanych przekaźników firmy **Relpol**:



- P1 – typ zwierny
- P2 – typ rozwierny

Maksymalne napięcie zestyków AC/DC.....	400 V/ 250 V
Znamionowy prąd obciążenia w kategorii AC.....	8A / 250 V AC
Znamionowy prąd obciążenia w kategori DC.....	8A / 24 V DC
Znamionowe napięcie cewki	12 V
Roboczy zakres napięcia cewki.....	od 8,4 do 29,5 V
Minimalny prąd zestyków.....	5 mA
Obciążalność prądowa trwała.....	8A
Rezystancja zestyków.....	≤ 100 mΩ przy 1 A, 24 V
Maksymalna częstość łączeń przy obciążeniu znamionowym w kategorii AC1.....	600 cykle/h
Bez obciążenia.....	72 000 cykle/h
Czas zadziałania.....	10 ms
Czas powrotu.....	5 ms
Znamionowe napięcie izolacji.....	400 V AC

Parametry użytego stycznika firmy **Danfoss**:

- Typ CI12

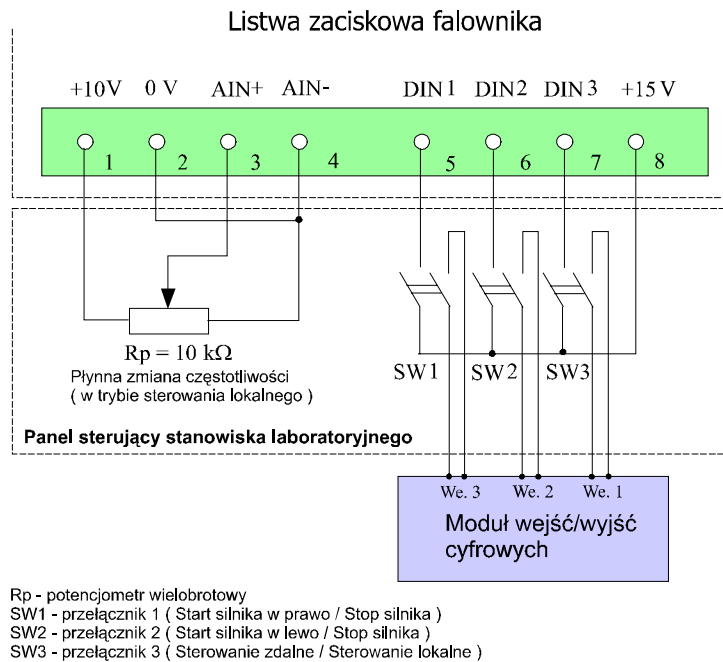
Obciążenie AC3 przy napięciu pracy $U_e = 220 - 240$	3 kW
Obciążenie AC3 przy napięciu pracy $U_e = 380 - 690$	5,5 kW
Prąd obciążenia I_e	12 A
Ilość styków głównych	3
Napięcie cewki	220-230 V AC 50 Hz



- Styk pomocniczy typ CB-NO

Funkcja styku	zwarcie
Obciążenie napięciowe U_e	500 V
Obciążenie prądowe I_e	6 A

3.2.6. Układ sterowania lokalnego



Rys. 5. Schemat blokowy układu sterowania lokalnego

Do sterowania lokalnego falownikiem wykorzystano jego listwę sterującą. Schemat połączeń i elementów układu sterowania lokalnego przedstawia rysunek 5.

Na panelu sterowniczym stanowiska umieszczono trzy przełączniki dwustanowe (SW1 ÷ SW3), które podłączono do zacisków wejść cyfrowych DIN1 ÷ DIN3 listwy zaciskowej falownika. Funkcje tych wejść zostały określone za pomocą ustawień odpowiednich parametrów falownika [1][2]. Wykorzystując napięcie +15 V z listwy, możliwe było podanie stanu wysokiego lub niskiego na wejścia. Przełącznik SW3 pozwala na wybór trybu sterowania falownikiem, między sterowaniem lokalnym a sterowaniem zdalnym. Kiedy przełącznik jest otwarty (stan niski wejścia), możliwe jest sterowanie lokalne. Może ono odbywać się za pomocą przycisków panelu operatorskiego OPM2 lub też poprzez zaciski listwy sterującej. W trybie tym, nie ma możliwości sterowania łączem RS485. Wysyłane rozkazy przez łącze szeregowe będą ignorowane. Jeżeli zostanie wybrany tryb sterowania zdalnego (stan wysoki wejścia), sygnały podawane na zaciski listwy falownika będą ignorowane. W trybie tym, falownik jest ustawiony do komunikacji i sterowania poprzez łącze szeregowe RS485.

Stany przełączników podawane są także do modułu wejść/wyjść cyfrowych, w celu ich monitorowania. Przy pomocy potencjometru wieloobrotowego podłączonego do wejść analogowych listwy, jest możliwe zadawanie płynne częstotliwości falownika.

Funkcje pozostałych dwóch przełączników SW1 i SW2 opisane zostały na rysunku 5.

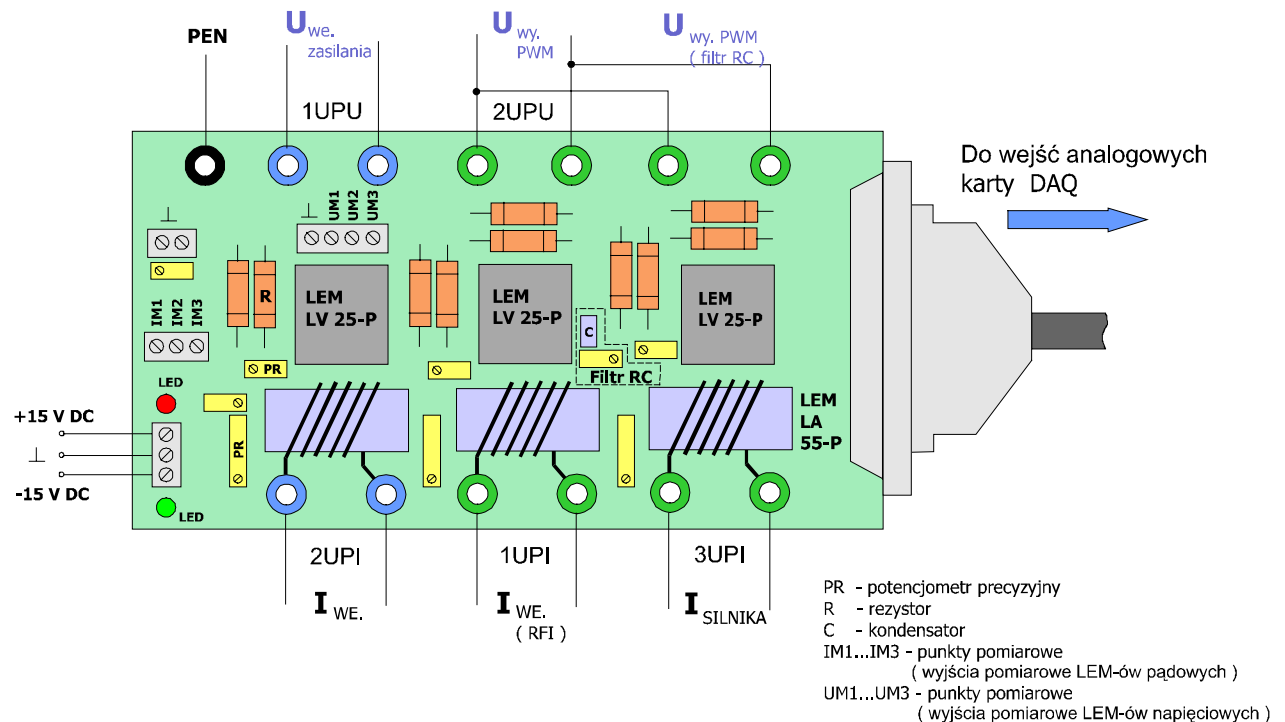
4. Układ pomiarowy

4.1. Właściwości układu

Układ pomiarowy pozwala na realizację następujących funkcji:

- pomiaru rzeczywistych przebiegów wartości chwilowych:
 - napięcia wejściowego zasilającego falownik (napięcie fazowe sieci zasilającej),
 - prądu zasilania falownika,
 - prądu zasilania falownika z filtrem typu RFI,
 - napięcia wyjściowego przewodowego falownika,
 - napięcia wyjściowego przewodowego falownika po zastosowaniu filtra typu RC,
 - prądu wyjściowego falownika (prądu stojana silnika klatkowego),
- monitoringu pracy zasilacza napięcia stabilizowanego ± 15 V, poprzez pomiar jego napięć wyjściowych,
- przystosowania sygnałów pomiarowych do wymagań wejść analogowych karty zbierania danych,
- separacji galwanicznej sygnałów mierzonych od sygnałów pomiarowych.

4.2. Budowa i opis układu



Rys. 6. Schemat poglądowy modułu pomiarowego

W skład układu pomiarowego wchodzi:

- zasilacz stabilizowany napięcia stałego ± 15 V,
- moduł pomiarowy.

Moduł pomiarowy został zbudowany z wykorzystaniem przetworników elektrycznych typu LEM. Zapewniają one galwaniczne odseparowanie sygnałów mierzonych o wartościach kilkuset voltów i kilkunastu amperów od sygnałów pomiarowych niskonapięciowych.

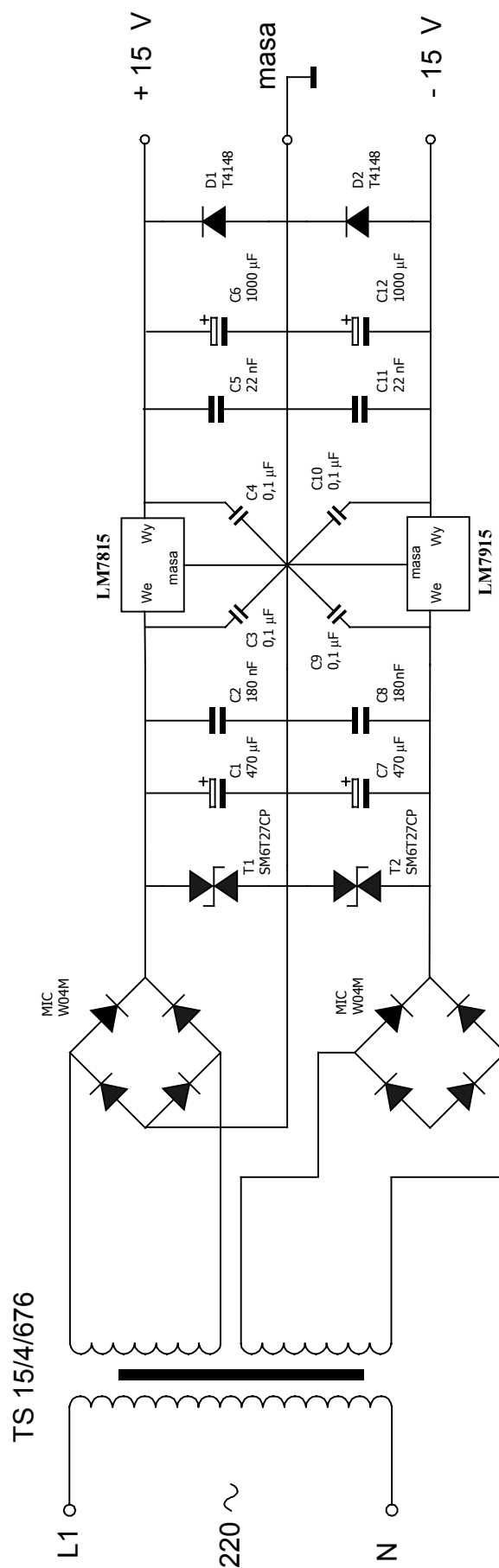
Zastosowane w układzie LEM-y napięciowe i prądowe charakteryzują się następującymi parametrami [24]:

LEM napięciowy typ **LV 25-P**

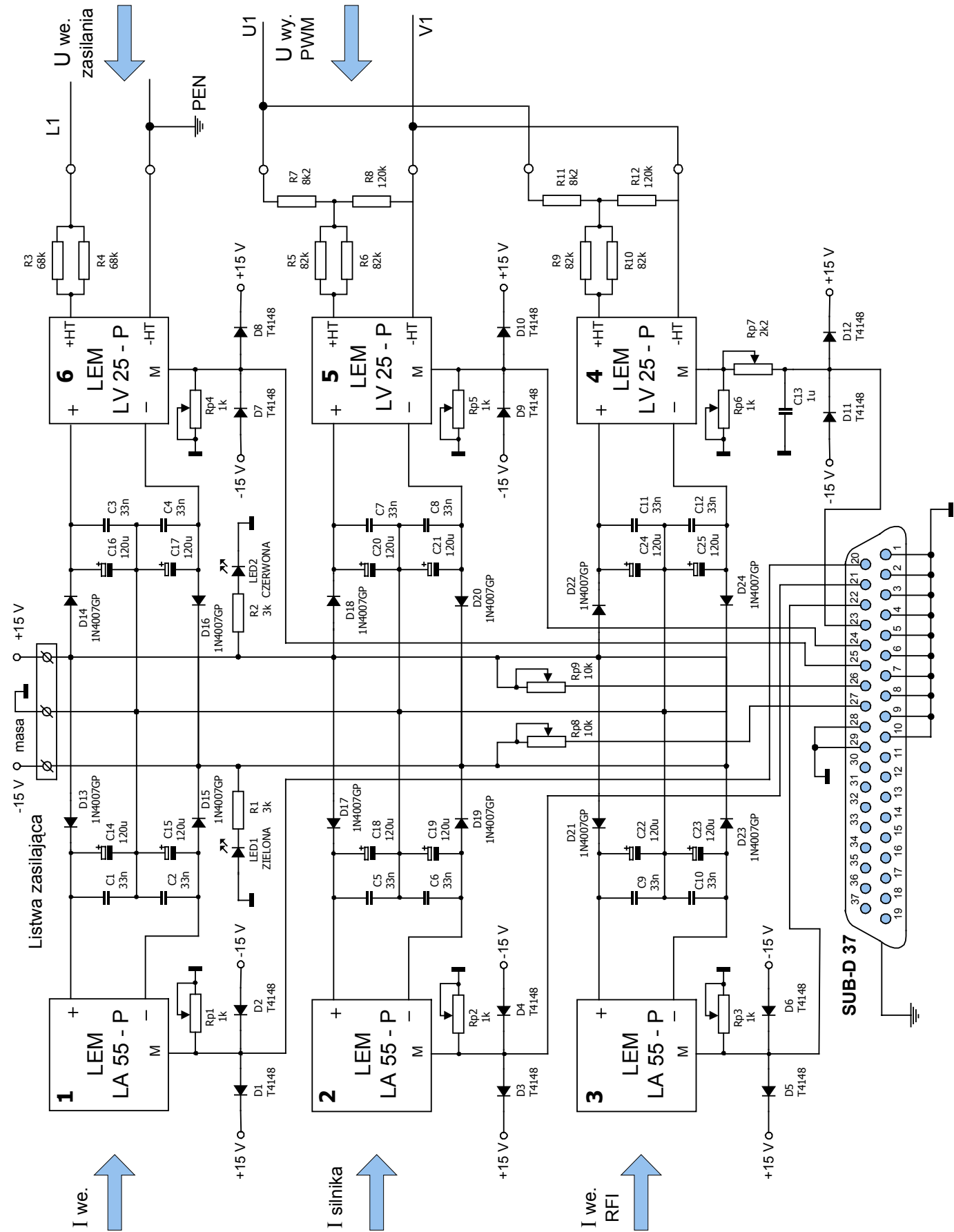
- zakres mierzonego napięcia	10 V – 500 V
- znamionowy prąd mierzony	$I_p = 10$ mA
- zakres pomiarowy prądu	0 ± 14 mA
- znamionowy prąd wtórny (pomiarowy)	$I_w = 25$ mA
- napięcie izolacji	2,5 kV _{sk}
- czas reakcji	40 μ s
- dokładność pomiaru I_w	$\pm 0,8$ %
- liniowość	$\pm 0,2$ %

LEM prądowy typ **LA 55-P**

- zakres mierzonego prądu	0 ± 100 A
- znamionowy prąd pierwotny (mierzony)	$I_n = 50$ A
- rezystancja wewnętrzna	$R_w = 145$ Ω
- znamionowy prąd wtórny (pomiarowy)	$I_w = 25$ mA
- napięcie izolacji	2,5 kV _{sk}
- czas reakcji	40 μ s
- dokładność pomiaru I_n	$\pm 0,65$ %
- liniowość	$\pm 0,15$ %
- pasmo przenoszenia	0 – 200 kHz



Rys. 7. Schemat ideowy zasilacza napięcia stabilizowanego ± 15 V



Rys. 8. Schemat ideowy modułu pomiarowego

4.3. Zasilacz stabilizowany napięcia stałego ± 15 V

Schemat ideowy zasilacza przedstawia rysunek 7.

Układ zasilacza stabilizowanego napięcia stałego, został zbudowany między innymi z użyciem transformatora o dwóch oddzielnych uzwojeniach wtórnych i stabilizatorów napięć z serii LM78 i LM79 z ochroną przeciwzwarciową dla dodatniego i ujemnego napięcia wyjściowego. Zastosowano także diody dwukierunkowe do ochrony przepięciowej, tzw. Transil'e (są to połączone przeciwsobnie dwie diody Zenera charakteryzujące się bardzo krótkim czasem zadziałania i ostrą charakterystyką ograniczenia napięcia stabilną w długim okresie czasu). Oprócz tego zastosowano szereg kondensatorów odsprężających od wyższych częstotliwości i poprawiających stabilizację napięcia wyjściowego.

Parametry zasilacza napięcia stałego:

- napięcie wejściowe: 220 V ~50Hz
- moc pobierana: 15 VA
- napięcia wyjściowe: +15 V, - 15 V napięcia stałego
- obciążalność prądowa: 1000 mA

4.4. Moduł pomiarowy

Schemat ideowy modułu przedstawia rysunek 8.

Moduł pomiarowy zawiera pięć układów pomiaru napięć i trzy układy pomiaru prądów.

Mierzone napięcia to:

- wejściowe fazowe napięcie zasilające falownik (1UPU),
- wyjściowe międzyprzewodowe napięcie falownika (2UPU, 3UPU),
- wyjściowe napięcia zasilacza stabilizowanego ± 15 V.

Układ pomiaru napięcia (1UPU) został włączony w obwód za blokiem styczników między fazę L1 i punkt zerowy sieci zasilającej. Układy 2UPU, 3UPU zostały włączone między zaciski U1-V1 listwy łączącej silnik z falownikiem. Miejsce połączeń przedstawia rysunek 1 w rozdziale: 3. Opis stanowiska laboratoryjnego.

Mierzone napięcia zasilacza, zostały pobrane z zacisków zasilających moduł pomiarowy.

Poprzez ustalenie zakresów napięć mierzonych (układy 1UPU, 2UPU, 3UPU), zostały dobrane odpowiednie wartości rezystorów na poszczególnych wejściach LEM-ów.

Za pomocą potencjometrów precyzyjnych ich wyjścia pomiarowe zostały skalibrowane do odpowiednich poziomów napięć.

Przykład ustawień :

Przy pomiarze napięcia zasilającego - 1UPU, zakres pomiarowy został ustalony na 250 V wartości skutecznej (353,5 V wart. max). Wartość rezystorów: R4 i R5 (Rys. 8), została tak dobrana, aby prąd po stronie pierwotnej LEM'a - I_p , osiągał wartość znamionową równą 10mA, przy napięciu mierzonym równym 250 V. Dzięki temu, po stronie wtórnej LEM'a osiągnięty został znamionowy prąd wtórny $I_w = 25$ mA. Ustawiając odpowiednią wartość potencjometru precyzyjnego Rp4, dostosowano poziom napięcia pomiarowego do wymagań wejść karty zbierania danych (wejścia analogowe bipolarne ± 10 V).

Przy pomiarze napięcia wyjściowego falownika (U_{WY} . PWM), potrzebne było zastosowanie dodatkowego dzielnika napięcia na wejściu LEM'a. Wartość maksymalna amplitudy tego napięcia wahała się od 580 do 590 V i wykaczała poza jego zakres pomiarowy (10 ÷ 500V). Z uwagi na kształt przewodowego napięcia wyjściowego falownika - PWM (*Pulse Width Modulation* - Modulacja Szerokości Impulsu), aby zaobserwować przebieg pierwszej harmonicznej na którą reaguje silnik, zastosowano na wyjściu LEM'a (układ 3UPU) filtr dolnoprzepustowy typu RC [22].

Częstotliwość graniczna została przyjęta dla $f = 500$ Hz, aby przesunięcie fazowe przebiegu za filtrem było jak najmniejsze (dla częst. 50 Hz wynosi $\approx 6^\circ$). Wartości R i C filtru zostały wyliczone z następującego wzoru :

$$f_g = \frac{1}{2\pi \cdot RC} \quad \text{dla } f = 0,1 \cdot f_g \rightarrow \text{faza} \approx 6^\circ \quad (1)$$

gdzie:

- f_g - częstotliwość graniczna
- R - rezystancja filtru
- C - pojemność filtru

obliczenia:

- wartości przyjęte: $f_g = 500$ Hz, $C = 1\mu\text{F}$

- wartość wyliczona:

$$R = \frac{1}{2\pi \cdot C \cdot f_g} = \frac{1}{2 \cdot 3,14 \cdot 0,000001 \cdot 500} = 318,47\Omega \quad (2)$$

Parametrom R i C filtru odpowiada w układzie (Rys. 8) potencjometr Rp7 ustawiony na wyliczoną wartość oraz kondensator C13.

Wartości napięć zasilających moduł pomiarowy (+15 V i -15 V), zostały przekazane do dwóch pozostałych wejść analogowych karty zbierania danych, z dzielników napięcia (potencjometr Rp8 i Rp9), w stosunku 1/3. Umożliwiło to monitoring napięć zasilacza stabilizowanego.

Mierzone prądy w układzie to :

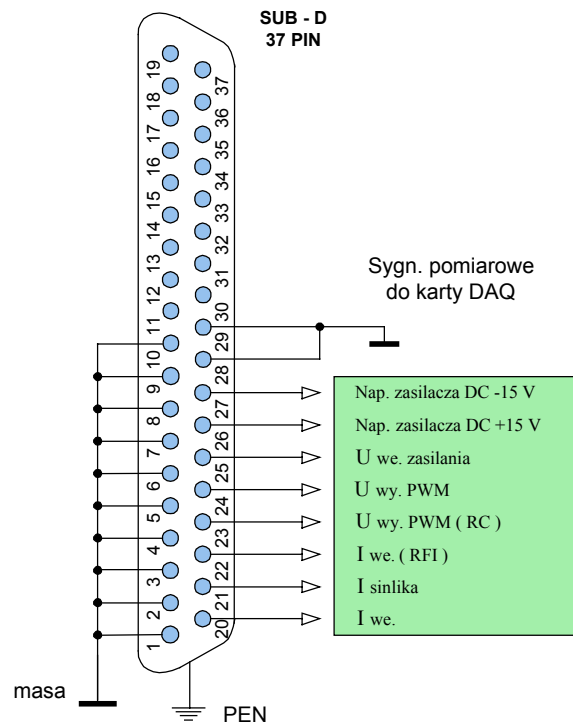
- wejściowy prąd zasilania falownika z filtrem wejściowym RFI (1UPI),
- wejściowy prąd zasilania falownika (2UPI),
- wyjściowy prąd przewodowy zasilania silnika (3UPI).

Układ pomiaru prądu (1UPI) został włączony w fazę L1 między wyjściem z bloku styczników a wejściem filtru RFI. Natomiast układ 2UPI został włączony między zaciskiem wejściowym zasilającym falownik a wyjściem filtru. Przy pomiarze prądu wyjściowego falownika układ 3UPI został włączony pomiędzy zacisk V falownika a zacisk V1 listwy, do której podłączony jest silnik. Miejsce połączeń przedstawia rysunek 1 w rozdziale: 3.Opis stanowiska laboratoryjnego.

Zakres pomiarowy prądu, układu LEM-ów prądowych, jest ustalony przez dobór uzwojeń wtórnych układu. Zakres napięcia wyjściowego układu pomiaru prądu dobrano potencjometrem precyzyjnym w układzie. Wzmocnienie układu pomiaru prądu wynosi 1V/1A. Wszystkie sygnały mierzonych napięć i prądów, doprowadzone do płytki modułu pomiarowego, zostały zrealizowane z doprowadzeniem sygnałów przewodami dwustronnie ekranowanymi (Rys.1), w celu zmniejszenia wpływu oddziaływania pola elektromagnetycznego na ich kształt i wartość.

Każdy z torów zasilania ± 15 V, układów LEM, został odseparowany od wysokich częstotliwości poprzez kondensatory odsprężające, a parametry stabilizacji ich napięć zostały poprawione (Rys.8).

Na wyjściach z LEM-ów, amplitudy napięć sygnałów pomiarowych zostały ograniczone (poprzez szybkie diody impulsowe), w celu dodatkowego zabezpieczenia wejść analogowych karty zbierania danych oraz zmniejszenia wartości chwilowych zakłóceń w sygnałach pomiarowych.



Rys. 9. Opis wyjść złącza SUB-D37 modułu pomiarowego

Sygnaly pomiarowe zostały doprowadzone do karty zbierania danych przewodem 37- pinowym, typu „skrętka”, ekranowanym dwustronnie (Rys. 9).

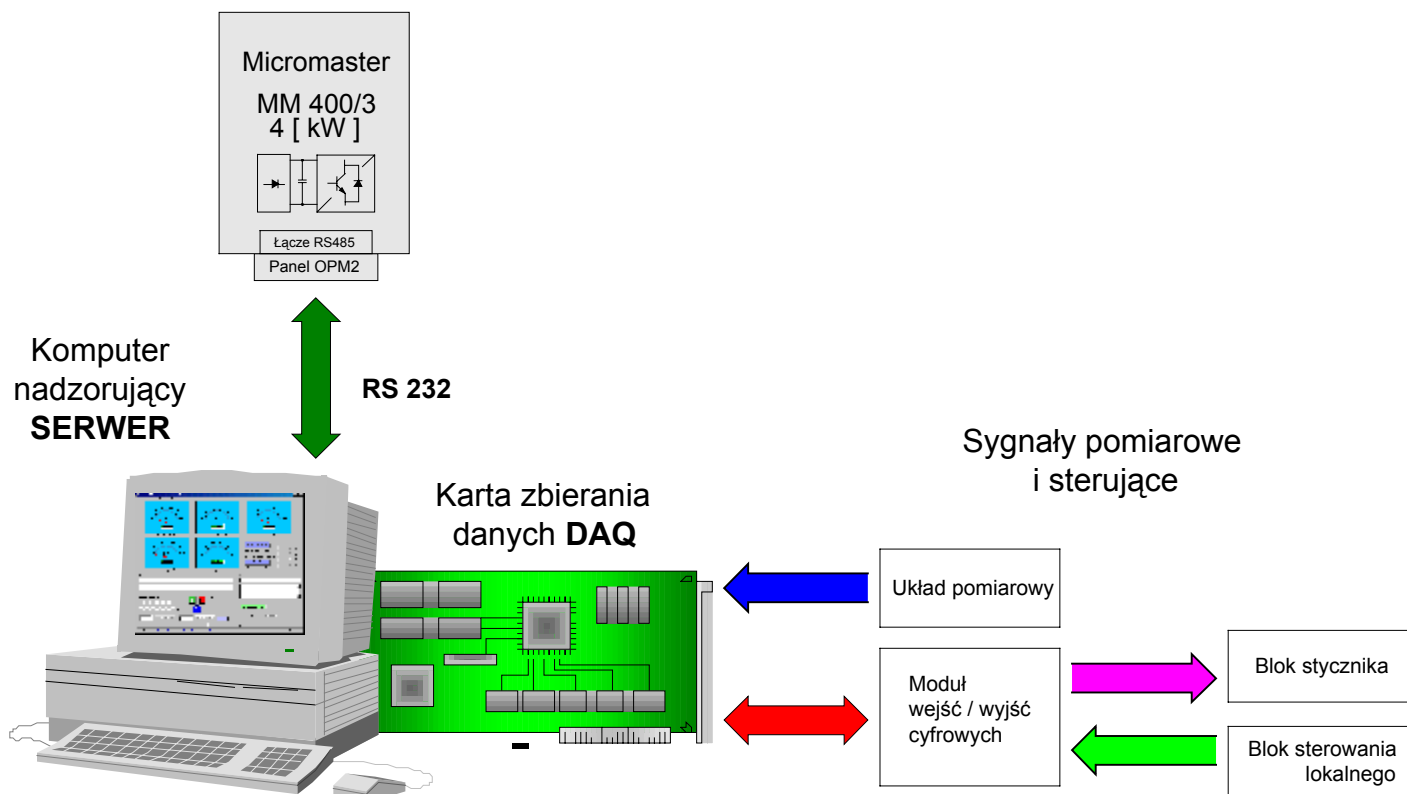
Szczególnie ważną sprawą było podłączenie do tego samego uziomu, współpracujących ze sobą karty zbierania danych znajdującej się w komputerze i modułu pomiarowego. Dlatego też złącze SUB-D modułu pomiarowego i ekran przewodu zostały połączone z przewodem PEN sieci zasilającej co zapewniało niską impedancję dla sygnałów o wysokich częstotliwościach.

5. Układ monitoringu i sterowania zdalnego

5.1. Budowa i opis układu

Układ monitoringu i sterowania zdalnego został zbudowany z następujących podzespołów :

- modułu wejść/wyjść cyfrowych dla karty zbierania danych,
- karty zbierania danych DAQ,
- komputera i oprogramowania umożliwiającego:
 - komunikację z falownikiem,
 - monitoring i sterowanie jego pracą,
 - monitoring i sterowanie elementami falownikowego układu napędu,
 - akwizycję i obserwację wartości chwilowych przebiegów napięć i prądów,
 - pełnienie roli serwera dla monitoringu i sterowania poprzez sieć komputerową Internet/Intranet.



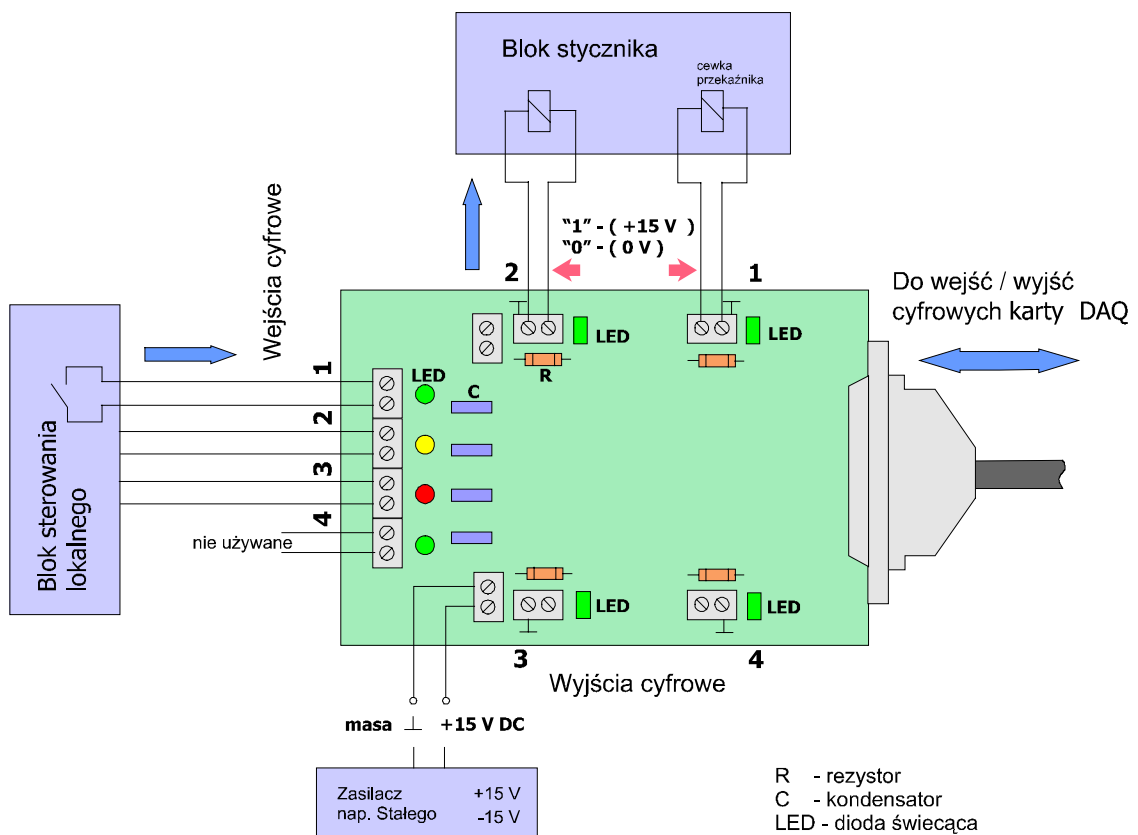
Rys. 10 Schemat blokowy układu monitoringu i sterowania zdalnego

Wykorzystując złącze szeregowo RS485 falownika oraz konwerter RS485/RS232 zawarty w panelu operatorskim OPM2, podłączono do falownika komputer klasy IBM PC (poprzez port szeregowy - COM2). Do obsługi komunikacji między komputerem a falownikiem stworzono własne oprogramowanie, które umożliwiło monitoring i sterowanie pracą układu napędu z przemiennikiem częstotliwości „Micromaster”.

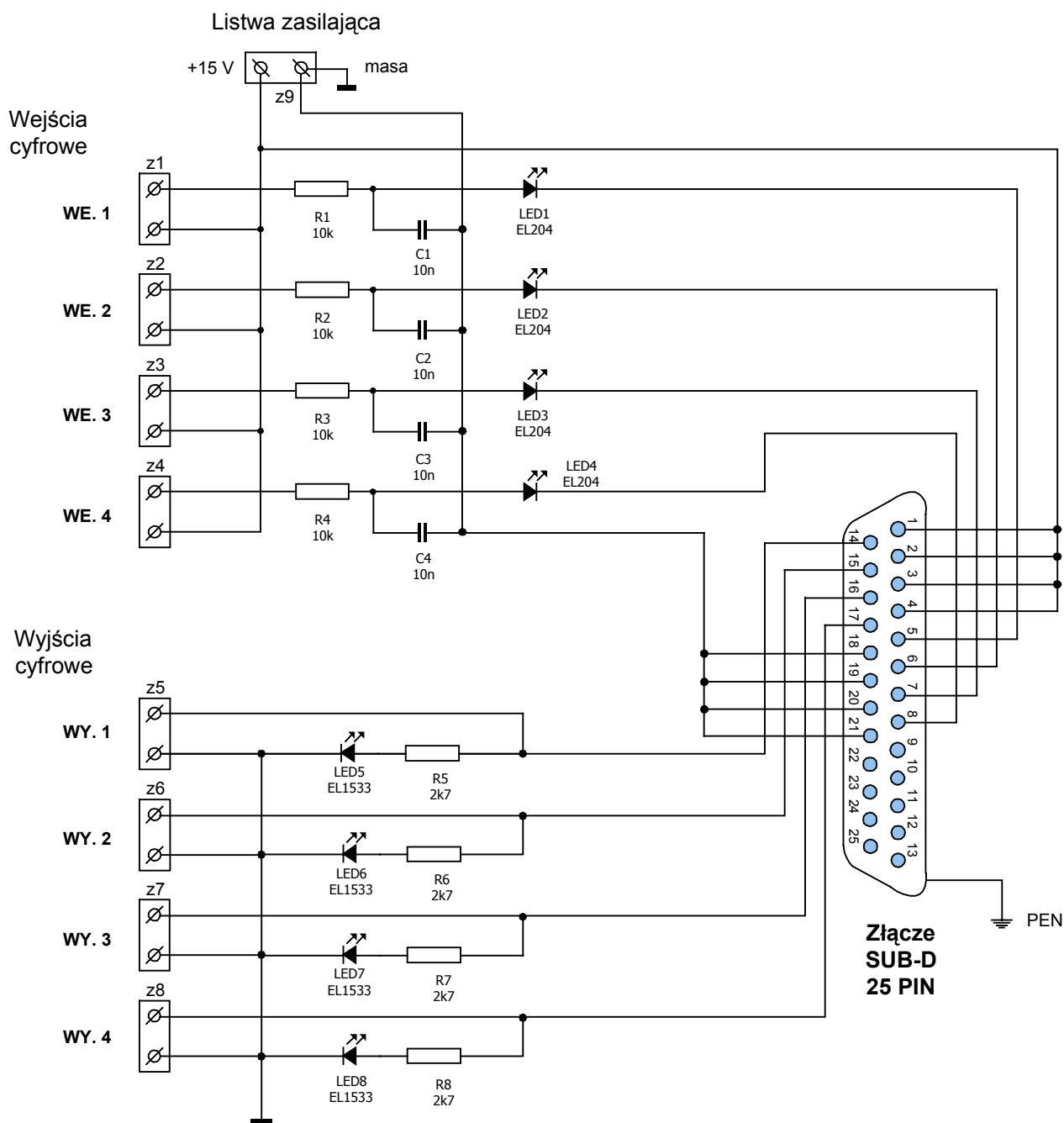
Oprogramowanie karty zbierania danych - DAQ, pozwoliło na obserwacje przebiegów wartości chwilowych napięć i prądów dostarczanych z układu pomiarowego.

Zbudowanie modułu wejść/wyjść cyfrowych umożliwiło sterowanie pracą bloku stycznika (załączanie i wyłączanie głównego obwodu zasilania), a także monitorowanie pracy bloku sterowania lokalnego (stan wejść dwustanowych falownika).

5.2. Moduł wejść/wyjść cyfrowych



Rys. 11. Schemat blokowo - montażowy modułu wejść/wyjść cyfrowych



Rys. 12. Schemat ideowy modułu wejść/wyjść cyfrowych

5.2.1. Opis i właściwości modułu

Moduł umożliwia współpracę karty zbierania danych z innymi blokami falownikowego układu napędu. Zasilany jest napięciem stałym +15 V podanym na listwę zasilającą modułu. Zawiera cztery wejścia i cztery wyjścia cyfrowe. Każde z nich posiada dwa stany logiczne: wysoki - „1” logiczna (+15 V) oraz niski - „0” logiczne (0 V).

Układ pozwala na sterowanie przekaźnikami załączającymi i wyłączającymi stycznik główny obwodu zasilania falownikowego układu napędu oraz monitorowanie stanów wejść dwustanowych listwy falownika. Umożliwia także wizualną kontrolę stanu wejść i wyjść cyfrowych poprzez ich sygnalizację za pomocą diod świecących. Poprzez złącza dwuzaciskowe (z1÷z9) możliwe jest łatwe podłączenie przewodów łączących moduł z wybranymi blokami. Połączenie z kartą zbierania danych zrealizowane jest za pomocą złącza SUB-D 25-pinowego i przewodu ekranowanego typu „skrętka”. Schemat połączeń z modułem przedstawia rysunek 11.

5.2.2. Zasada działania modułu

Schemat ideowy modułu wejść/wyjść cyfrowych przedstawia rysunek 12.

Zasada działania omówiona zostanie na przykładzie jednego zespołu wejścia i wyjścia, gdyż budowa innych par wejść i wyjść jest taka sama.

Wejście pierwsze (WE.1), jest podłączone do przełącznika SW3 w układzie sterowania lokalnego (Rys. 5). Posiada on dwa stany i przełącza sterowanie falownikowym układem napędu ze sterowania lokalnego na sterowanie zdalne. W chwili kiedy nastąpi zwarcie łącznika, napięcie +15V z zacisku wejściowego modułu zostanie podane poprzez rezystor i diodę świecącą na wejście cyfrowe karty DAQ. Spowoduje to ustawienie jej wejścia w stan wysoki. Zdarzenie to zostanie zasygnalizowane zaświeceniem diody LED1.

Rozwarcie łącznika ustawi wejście karty w stan niski, wygaszając diodę świecącą. Kondensator C1 tłumy stany przejściowe występujące w chwili zwierania i rozwierania łącznika. Rezystor R1 ogranicza prąd diody świecącej.

Wyjście pierwsze (WY.1) modułu, podłączone jest do cewki przekaźnika P1 bloku stycznika (Rys. 4). Przekaznik załącza napięcie cewki stycznika głównego obwodu zasilania. Do wyjścia cyfrowego karty DAQ (pin 1) odpowiadającemu wyjściu pierwszemu modułu jest podawane przez cały czas napięcie +15V. W chwili kiedy wyjście karty jest ustawione w stan wysoki (pin 1 i pin 14 zwarte ze sobą), napięcie to jest dostarczane do wyjścia modułu i zasilą cewkę przekaźnika. Stan ten jest sygnalizowany przez diodę świecącą LED5. Kiedy nastąpi ustawienie wyjścia karty w stan niski, napięcie zostaje odcięte z wyjścia modułu (pin 1 i pin 14 rozwarty, dioda gaśnie). Dioda LED5 i rezystor R5 są połączone równolegle do wyjścia modułu (równolegle z cewką przekaźnika), aby nie powodować dodatkowego podziału napięcia, jaki miałby miejsce w połączeniu szeregowym z jednym z zacisków wyjścia modułu.

Budowa i działanie wejść i wyjść cyfrowych karty wykorzystanych w tym module jest opisana w następnym rozdziale.

5.3. Karta zbierania danych

5.3.1. Opis, budowa i właściwości karty



Zdjęcie 4. Wygląd karty

Zastosowana karta zbierania danych niemieckiej firmy ADDI-DATA powiększyła możliwości monitoringu i sterowania falownikowym układem napędu. Umożliwiła ona realizację następujących funkcji:

- zbierania chwilowych przebiegów sygnałów pomiarowych odzwierciedlających pracę i właściwości układu (wejścia analogowe karty),
- zamianę wartości analogowych na postać cyfrową w celu ich dalszej obróbki przez komputer i możliwości prezentacji i analizy danych w napisanym oprogramowaniu,
- sterowania poszczególnymi elementami układu (wyjścia cyfrowe karty),
- zobrazowania stanów pracy wybranych podzespołów układu (wejścia cyfrowe karty),

Karta pełni funkcję interfejsu między układami zawartymi w stanowisku laboratoryjnym a komputerem i oprogramowaniem na nim działającym.

Korzystanie z funkcji karty i sterowania jej pracą było możliwe dzięki napisaniu oprogramowania wykorzystującego dostarczone wraz z nią sterowniki.

Parametry karty DAQ¹:

Typ karty:	PA 3000
Typ złącza:	ISA – 16 bitowe
Wejścia analogowe:	
Ilość wejść analogowych:	8
Rozdzielczość wejścia analogowego:	12 – bit, 0 ÷ 4096
Częstotliwość próbkowania (na 1 wejście) :	142,867 kHz
Transmisja danych:	Dane do komputera (16-bit) 1) przez komendy wejść/wyjść 2) przerwanie po EOC ² 3) transfer DMA ³ po EOC
Rozpoczęcie zbierania danych:	1) przez wyzwolenie w programie 2) TIMER 0 (82C54) 3) TIMER 0 i 1 (82C54)
Zakres wejść analogowych:	Napięciowy : - Unipolarne we. 0 – 10 V - Bipolarne ± 10 V programowany software'owo. Prądowy: - Unipolarne we. 0-20 mA
Zabezpieczenie przeciwprzepięciowe:	70 V _{pp} ² przy włączonym zasilaniu karty
Pasma przenoszenia:	ograniczone filtrem dolnoprzepustowym o częst. granicznej $f_g = 159$ kHz (-3 dB)
Impedancja wejściowa:	10 ¹² Ω // 20 nF do masy
Zakres czułości:	programowany na każdy kanał : 1, 2, 5, 10, 20, 50, 100, 200, 500
Błąd przesunięcia:	po kalibracji : - Bipolarne we. ±1/2 LSB ⁴ - Unipolarne ±1/2 LSB
	Dryft (0°C do 60°C) - Bipolarne ± 7ppm/°C - Unipolarne ± 7ppm/°C
Dokładność:	±1 LSB (± 2,44 mV)
Izolacja optyczna od komputera:	500 V napięcia stałego
Kanały DMA:	kanał nr 5, 6 i 7 (jeden lub dwa kanały naraz)
Przerwania:	IRQ ⁵ 9,10,11,12,14,15 dla AT
Liczniki:	3 x 16 –bit-owy licznik (dwa na sekwencję programu)

¹ DAQ – *Digital Acquisition* – cyfrowa akwizycja (zbieranie) danych

² EOC – *End of Conversion* – koniec zbierania danych

³ DMA – *Direct Memory Access* – bezpośredni dostęp do pamięci

⁴ LSB – *Less Significant Bit* – bit na najmniejszej pozycji w słowie cyfrowym

⁵ IRQ – *Interrupt request* – żądanie przerwania

Wejścia cyfrowe:

Ilość wejść cyfrowych:	4
Prąd wejściowy przy 24 V:	3 mA
Zakres napięcia wejściowego:	0 – 30 V
Izolacja optyczna od komputera:	1000 V nap. przemiennego
Poziom napięć dla logicznego „0”:	0 – 5 V nap. stałego
Poziom napięć dla logicznej „1”:	10 – 30 V nap. stałego

Wyjścia cyfrowe:

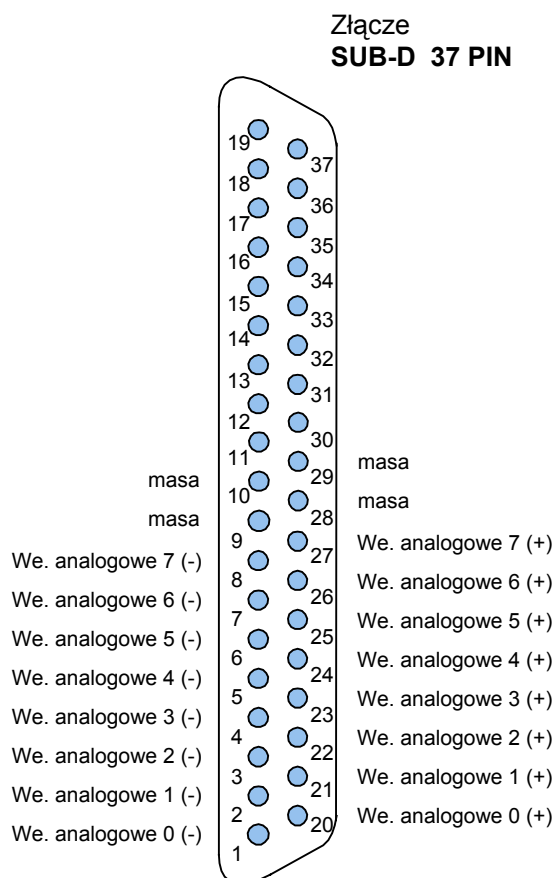
Ilość wyjść cyfrowych:	4
Maksymalny prąd przełączeń:	5 mA
Zakres napięciowy:	5 – 30 V
Izolacja optyczna od komputera:	1000 V nap. przemiennego

Wymagania dla komputera :

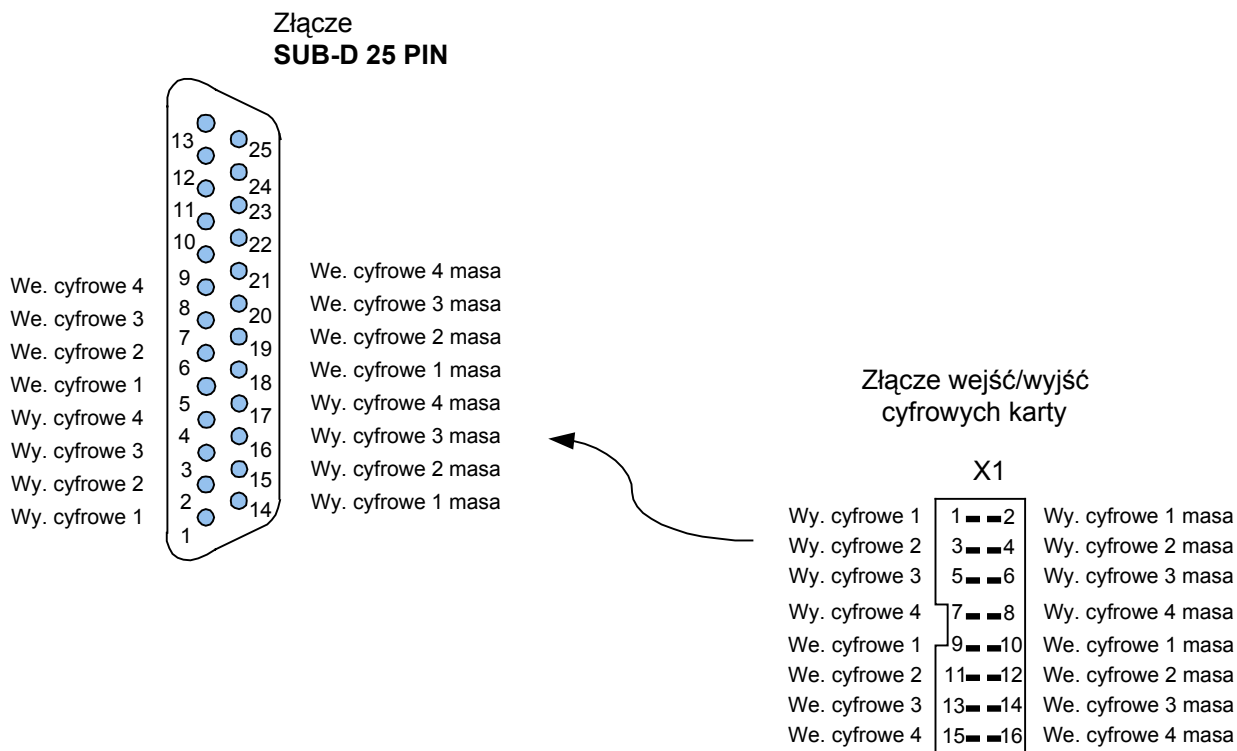
System operacyjny:	MS DOS 3.3 lub > Windows 3.1
Prędkość szyny danych :	8 Mhz

Wymagania dla zasilania karty:

Napięcie zasilające z komputera :	5 V \pm 5%
Typowy pobór prądu bez obciążenia:	580 mA



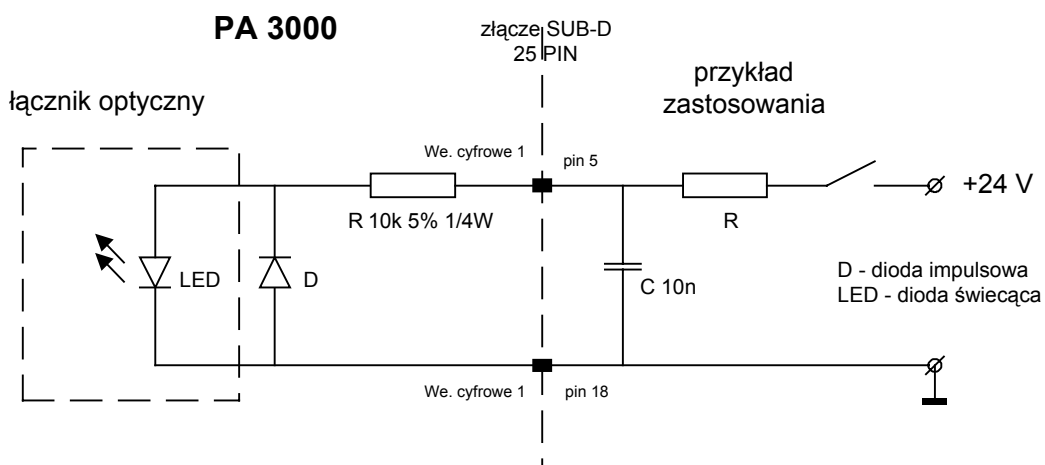
Rys. 13. Złącze wejść analogowych karty z opisem



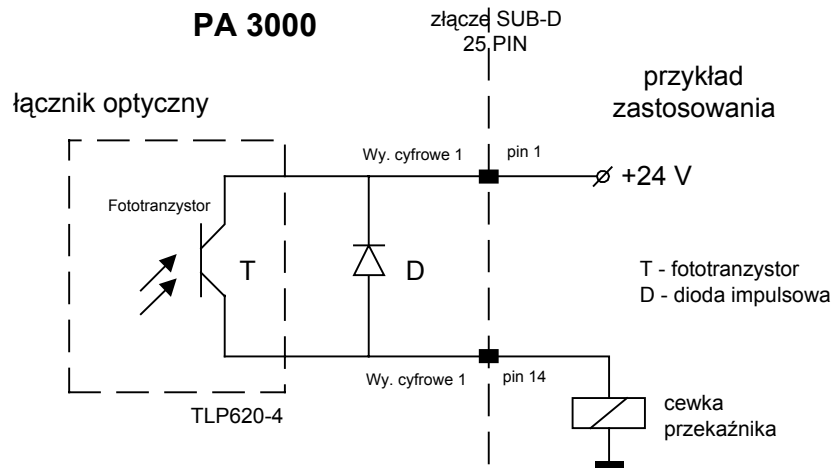
Rys. 14. Złącza wejść/wyjść cyfrowych z opisem

Złącze SUB-D 37, umożliwia połączenie bezpośrednio karty z modulem pomiarowym. Aby podłączyć moduł wejść/wyjść cyfrowych zastosowano dodatkowe przejście ze złącza X1, znajdującego się na karcie na złącze SUB-D 25 (Rys. 14). Dodatkowe złącze zostało umieszczone na listwie mocującej w tylnej części komputera.

Budowę wejść i wyjść cyfrowych karty przedstawiają rysunki: 15 i 16.



Rys. 15. Schemat budowy wejścia cyfrowego

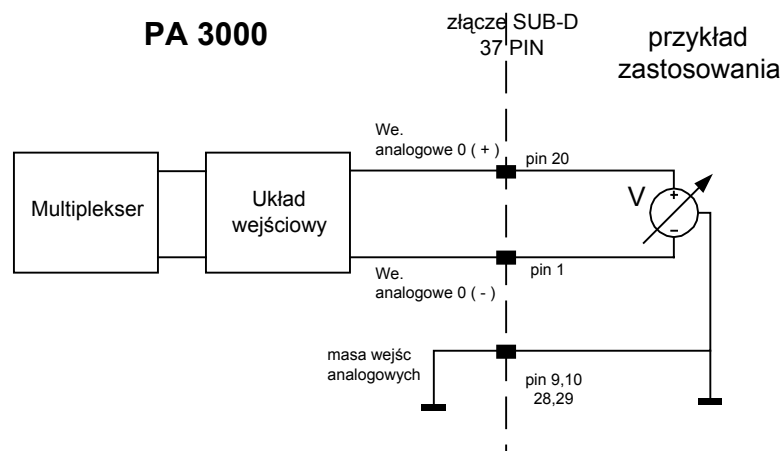


Rys. 16. Schemat budowy wyjścia cyfrowego

Wejścia i wyjścia cyfrowe karty wymagają zastosowania zewnętrznego źródła zasilania. W tym przypadku został wykorzystany tor zasilania +15 V zasilacza napięcia stałego.

Wejście cyfrowe karty jest zbudowane na transoptorowym łączniku (Rys. 15). Podając napięcie na to wejście powodujemy wygenerowanie przez diodę świecąca promienia światła. Następuje ustawienie wejścia w stan wysoki – „1” logiczna. Kiedy napięcie zdejmujemy z wejścia występuje stan niski czyli „0” logiczne. Zastosowany rezystor $R=10k\Omega$ ogranicza prąd diody. Wykorzystanie w budowie wejść/wyjść cyfrowych karty transoptorów powoduje odizolowanie galwanicznie od układów zewnętrznych.

Wyjście cyfrowe karty również wykorzystuje transoptor (Rys. 16). W chwili kiedy karta otrzymuje sygnał ustawienia swojego wyjścia w stan wysoki – „1” logiczna, nadajnik czyli dioda świecąca wytwarza promień świetlny, który padając na bazę fototranzystora powoduje jego wysterowanie. Wyjścia karty zostają zwarte i następuje zamknięcie zewnętrznego obwodu podłączonego do karty. W chwili kiedy wystąpi stan niski „0” – logiczne, dioda gaśnie i następuje zablokowanie fototranzystora. Wyjścia karty zostają rozwarne a obwód zewnętrzny przerywany. Dioda D, równolegle przyłączona do każdego z wyjść, zabezpiecza fototranzystor przed odwrotnym spolaryzowaniem napięcia.



Rys. 17 Schemat budowy wejścia analogowego karty

Budowę ogólną wejść analogowych przedstawia rysunek 17.

Wejścia te mogą pracować w rzeczywistym trybie różnicowym tak jak pokazuje to przykład na rysunku lub w trybie kiedy jest tworzona wirtualna masa. Tryb ten jest osiągnięty poprzez połączenie wejść analogowych oznaczonych symbolem „(-)” przez rezystor $1M\Omega$ do masy. W trybie tym poziom zera np. dla wejścia ustawionego na bipolarne odpowiada dokładnie połowie zakresu mierzzonego. Wejścia analogowe zawierają w układzie wejściowym filtr dolnoprzepustowy w celu ograniczenia wyższych częstotliwości. Układ wejściowy spełnia rolę dopasowania poziomów sygnałów do dalszych obwodów karty. Wejścia analogowe są multipleksowane i sygnały w odpowiedniej kolejności są podawane na przetwornik A/D (analogowo-cyfrowy) w celu dalszego ich przetworzenia na postać cyfrową.

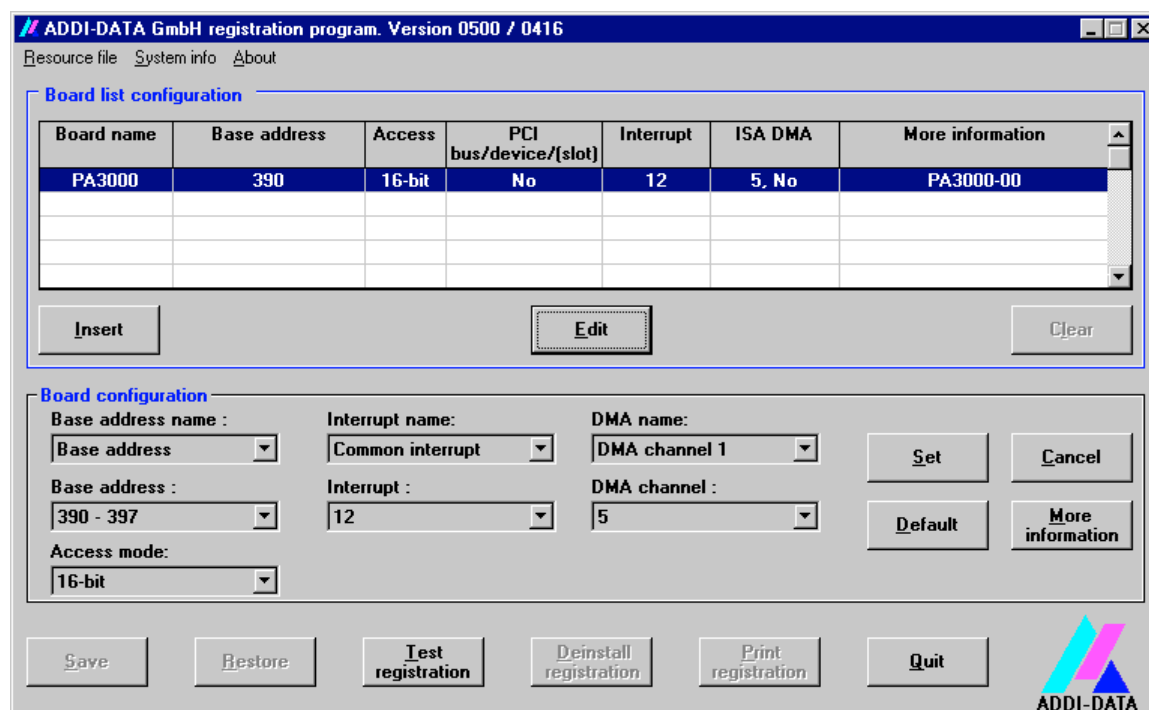
5.3.2. Ustawienia, konfiguracja i instalacja karty

W karcie wykorzystano wszystkie osiem wejść analogowych. Za pomocą odpowiednich zwerek na karcie, został ustawiony tryb z wirtualną masą oraz odpowiedni zakres czułości dla sygnału wejściowego. Aby współpraca karty z komputerem była możliwa, na karcie za pomocą switch'y (przełączników), ustawiany jest jej adres. Dokładny opis ich ustawień jest zawarty w specyfikacji technicznej karty [3].

Współpraca karty z oprogramowaniem w środowisku Windows 9x/NT wymaga zarejestrowania jej w systemie za pomocą programu ADDIREG dodanego do karty.

W programie tym ustawia się wszystkie potrzebne parametry do prawidłowego sterowania kartą i wykorzystywanie jej funkcji.

Rysunek 18 przedstawia okno ustawień tego programu.



Rys. 18. Wygląd okna programu „ADDIREG”

Okno programu podzielone jest na dwie części:

1) **Board list configuration** – lista ustawień karty, w której wyświetlane są aktualne ustawione jej parametry, takie jak:

- **Board name** – nazwa karty
- **Base address** – jej adres podstawowy
- **Access** – rodzaj dostępu (16-bitowy lub 8-bitowy)
- **PCI bus/device/(slot)** – określający czy karta posiada złącze PCI (rodzaj złącza komputerowego)
- **Interrupt** – numer przerwania pod którym jest obsługiwana karta
- **ISA DMA** – numer kanałów DMA dla karty o złączu ISA
- **More information** – dodatkowe informacje karty

Board name	Base address	Access	PCI bus/device/(slot)	Interrupt	ISA DMA	More information
PA3000	390	16-bit	No	12	5, No	PA3000-00

2) **Board configuration** – ustawienia karty, w tej części dokonuje się wszystkich potrzebnych ustawień karty. Oprócz wspomnianych powyżej, zawiera dodatkowo następujące ustawienia:

- **Base address name** – podstawowa nazwa adresu, w przypadku kiedy karta używa kilku adresów (jeden dla portu 1, inny dla portu 2), możliwy jest wybór różnych nazw adresów. Dla zastosowanej karty można ustawić tylko jeden adres.
- **Interrupt name** – określa rodzaj przerwania jaki obsługuje karta, może występować:
 - Common** – wspólne jedno przerwanie
 - Single** – pojedyncze przerwania dla różnych portów karty

Zastosowana karta posiada możliwość ustawienia tylko jednego wspólnego przerwania.

- **DMA name** – określa czy karta posiada dwa kanały DMA czy tylko jeden.
- **DMA channel** – umożliwia wybór numerów poszczególnych kanałów DMA

Base address name :	Interrupt name:	DMA name:		
Base address	Common interrupt	DMA channel 1	Set	Cancel
Base address :	Interrupt :	DMA channel :		
390 - 397	12	5	Default	More information
Access mode:				
16-bit				

Dla zastosowanej karty - PA3000, zostały wybrane następujące ustawienia:

- **Base address** – standardowy adres karty równy 390 Hex,
- **Acces mode** – 16-bitowy,
- **Interrupt** – 12,
- **DMA name** – jeden kanał DMA,
- **DMA channel** – 5.

Przy pracy karty w systemach Windows 9x, nie jest możliwe wykorzystanie kanałów DMA. Dlatego zastosowanym systemem, pod którym karta została użyta, był Windows NT umożliwiający w pełni wykorzystanie parametrów karty.

5.4. Komputer nadzorujący i sterujący pracą falownikowego układu napędu

Zastosowany komputer jest głównym elementem w układzie monitoringu i sterowania zdalnego. Sam komputer, w sensie sprzętowym, nie stanowi samodzielnego urządzenia pełniącego rolę jaką chcemy mu powierzyć. Dlatego, aby jego zadanie kontroli i sterowania pracą falownikowego układu napędu było realizowane, niezbędne jest zainstalowanie na nim systemu operacyjnego, umożliwiającego stabilną pracę. Do tych celów użyty został system Windows NT Server firmy Microsoft. Cechuje go stabilność i kontrola nad pracującymi pod nim programami i procesami. Umożliwia wielozadaniowość i wielowątkowość co jest wymagane przy stworzeniu sterowania i monitoringu takiego układu.

Użyty komputer powinien umożliwiać:

- komunikację z falownikiem,
- obsługę i sterowanie zastosowaną kartą zbierania danych,
- szybkie przetwarzanie danych i wykonywanie obliczeń matematycznych,
- szybką komunikację z innymi komputerami w sieci.

W zbudowanym układzie wykorzystany został komputer o następujących parametrach:

Procesor :	Pentium II Celeron 366 Mhz
Pamięć :	128 Mb
Dysk twardy :	20 Gb, firmy IBM, 7200 rpm
Karta grafiki :	Riva ZX , 4 Mb
Karta sieciowa :	10/100 Mb, firmy 3COM

Komputer z systemem operacyjnym nie stanowi jeszcze w pełni działającego układu monitoringu i sterowania zdalnego. Aby było to możliwe, potrzebne jest napisanie oprogramowania które będzie pełnił rolę opisaną na początku rozdziału.

Działanie i opis oprogramowania jest zawarte w następnym rozdziale.

6. Monitoring i sterowanie z wykorzystaniem komputera

6.1. Wstęp

Budując system monitoringu i sterowania z wykorzystaniem komputera trzeba wziąć pod uwagę fakt, iż do tego celu będzie potrzebne oprogramowanie. Gotowe oprogramowania przeznaczone są do użytku wyłącznie z całym systemem podzespołów danego producenta lub też są projektowane i wykonywane na zamówienie. W naszym przypadku ze względu na własny projekt systemu monitoringu i sterowania falownikowym układem napędu, budowę poszczególnych układów oraz wykorzystywanie nowych technologii i możliwości jakie niesie ze sobą sieć komputerowa Intranet/Internet, zostało napisane własne oprogramowanie. Współczesne języki programowania coraz bardziej ułatwiają proces tworzenia aplikacji. Znaczna większość programów jest pisana pod system Windows. System „okienek”, jest najbardziej rozpowszechnionym systemem na świecie, a jego interfejs graficzny umożliwia tworzenie wielu ciekawych i funkcjonalnych aplikacji. Obecne metody tworzenia oprogramowania coraz częściej opierają się na zautomatyzowaniu pewnych czynności i wykorzystaniu gotowych modułów. Przyspiesza to proces pisania aplikacji, jak również ułatwia jej testowanie.

Do napisania programu wybrany został język Visual Basic. W przeciwieństwie do innych języków programowania takich jak na przykład C++, gdzie programista musi zaprogramować nawet najdrobniejszy fragment kodu, Visual Basic część pracy wykonuje samodzielnie. Pisany w nim program tworzy się z komponentów, które nanosi się na formularz okienka. Następnie wykorzystując zdarzenia, metody, właściwości, dokonuje się ich oprogramowania. Oczywiście umożliwia on pisanie także złożonych i rozbudowanych aplikacji, co miało miejsce w naszym przypadku.

Tworzone oprogramowanie musiało realizować następujące zadania:

- komunikacji komputera z falownikiem,
- sterowania kartą zbierania danych i jej komunikacji z programem,
- zaimplementowania wyżej wymienionych zadań do potrzeb monitoringu i sterowania w tworzonej aplikacji.

1.2. Komunikacja komputera z falownikiem

Jednym z głównych celów w układzie monitoringu i sterowania, było stworzenie komunikacji między komputerem a falownikiem. Wymagało to zapoznania się z protokołem transmisji USS (*Universal Serial Interface Protocol* – uniwersalny szeregowy protokół interfejsu) [5][6]. Pozwala on na połączenie i komunikację poprzez łącze szeregowe do 30 falowników typu „Micromaster”. Umożliwia także uzyskanie następujących funkcji:

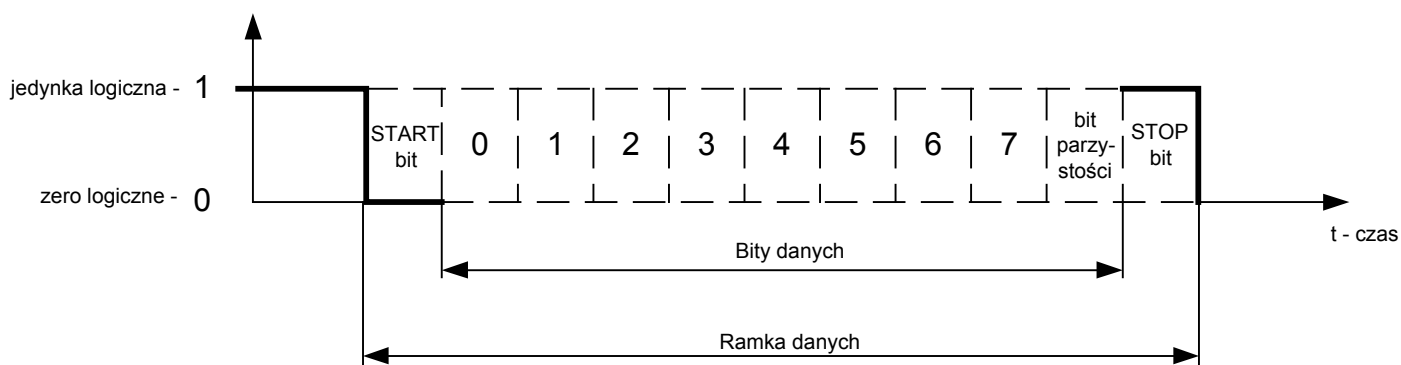
- pełnej kontroli i sterowania falownikiem np. Start, Stop, zmiana częstotliwości itp.
- odczyt i zmiana jego parametrów,
- przekazywania z falownika takich wartości jak:
 - prądu i napięcia przewodowego wyjściowego,
 - napięcia obwodu pośredniczącego prądu stałego,
 - częstotliwości bieżącej napięcia wyjściowego,
 - prędkości obrotowej wału silnika.

Protokół USS działa w systemie **Master – Slave**. Polega to na tym, że komputer (*Master* – pan) wysyła wiadomość do przekształtnika (*Slave* – sługa), a ten ją odbiera i odsyła razem ze swoją odpowiedzią. **Slave** wyśle odpowiedź do **Master’a** tylko wtedy kiedy otrzymana wiadomość nie zawierała błędów i był w niej zawarty jego adres. Każdy z przekształtników podłączonych do łącza szeregowego musi posiadać swój adres. W zastosowanym układzie podłączony jest jeden przekształtnik. Protokół ten umożliwia także wysyłanie wiadomości jednocześnie do wszystkich podłączonych falowników.

6.2.1. Parametry transmisji

Zastosowana, standardowa prędkość komunikacji wynosi 9600 Baud⁶. Możliwe jest zwiększenie tej prędkości do 19.2 kBaud przy czym nie wszystkie konwertery RS232/RS485 pozwalają na jej osiągnięcie. W zastosowanym połączeniu obydwie prędkości przeszły pomyślne testy i można było je stosować.

Wszystkie wiadomości przesyłane pomiędzy komputerem a falownikiem, zawierają po 14 bajtów. Każdy bajt danych przesyłany jest w formacie zawierającym 1 bit startu, 8 bitów danych, 1 bit parzystości i jeden bit stopu, łącznie 11 bitów. Wygląd takiej ramki danych przedstawia rysunek 19.



Rys. 19. Ramka danych

Transmisja odbywa się w trybie *half duplex* (pół dupleks), oznacza to że **Master** nie może jednocześnie wysyłać i odbierać wiadomości w tym samym czasie.

6.2.2. Budowa telegramów

Wiadomości wysyłane i odbierane nazwano telegramami ze względu na swoją krótką długość. Ich konstrukcja jest następująca:

Telegram wysyłany od Master’a do Slave’a

STX	LGE	ADR	PKE	IND	VAL	STW	HSW	BCC
-----	-----	-----	-----	-----	-----	-----	-----	-----

⁶ Baud – bodów (ilość bitów na sekundę)

Telegram wysyłany od Slave'a do Master'a

STX	LGE	ADR	PKE	IND	VAL	ZSW	HIW	BCC
-----	-----	-----	-----	-----	-----	-----	-----	-----

Każdy telegram rozpoczyna się pojedynczym bajtem **STX** (*Start of Text* – początek wiadomości). Bajt ten posiada stałą wartość 02 Hex i użyty jest do zasygnalizowania początku wiadomości.

Telegramy różnią się w swojej budowie dwoma przedostatnimi polami. Pozostałe pola mają tą samą funkcję chociaż mogą przyjmować różne wartości.

Opis pozostałych bajtów :

LGE – (*Telegram length* – długość telegramu) – 1 bajt określający ilość pozostałych bajtów występujących w telegramie. Dla Micromaster'a przyjmuje on wartość numeryczną 12.

ADR – (*Address byte* – bajt adresu) – 1 bajt określający adres przekształtnika (0 ÷ 30)

PKE – (*Parametr ID*) – 2 bajty, używane do kontroli parametryzacji falownika, w których podaje się numer parametru i rodzaj zadania do wykonania (np. odczyt parametru, zapis)

IND – (*Indexed parameters* – indeksowanie parametrów) - 2 bajty. Dla tego typu falownika nieużywane. Powinny być zawsze wypełnione zerami.

VAL – (*Parameter Value* – wartość parametru) – 2 bajty, wskazują wartość dla danego parametru określonego w polu PKE. W telegramie **Master'a**, kiedy żąda on wartości danego parametru, pole to jest wyzerowane. Kiedy natomiast następuje zmiana wartości parametru, to pole to zawiera nową wartość. Dla Slave'a, pole to jest wykorzystywane przy czytaniu parametru.

BCC – (*Block check charakter*) - 1 bajt zawierający sumę kontrolną określającą poprawność wysłanego telegramu. Wyznaczana jest poprzez wykonanie funkcji logicznej XOR na wszystkich poprzednich bajtach wiadomości.

STW – (*Control word* – słowo kontrolne) – słowo **Master'a**, zawiera 2 bajty używane do sterowania pracą falownika. Poszczególne ustawienie bitów tego słowa powoduje np.: start falownika, stop, zmianę kierunku wirowania silnika itp.

HSW – (*Main setpoint*) – słowo **Master'a**, 2 bajty, zawiera wartość żądanej częstotliwości wysyłanej do falownika.

ZSW – (*Status word* – słowo statusu) – słowo **Slave'a**, zawiera 2 bajty określające bieżący status falownika, np.: czy jest gotowy do pracy, czy nie ma żadnych błędów itp.

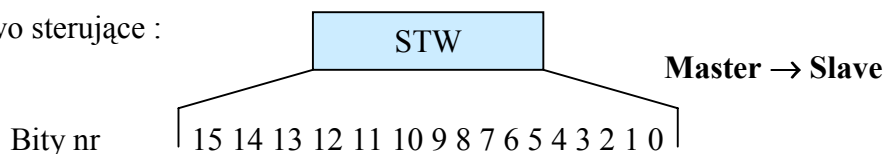
HIW – (*Main actual value*) – słowo **Slave'a**, 2 bajty reprezentujące aktualną częstotliwość wyjściową falownika.

Dla potrzeb tworzonego oprogramowania wymagane było poznanie dokładnego znaczenia poszczególnych bitów w obydwu słowach telegramu:

W słowie sterującym – **STW** oraz w słowie statusu – **ZSW**.

Zawierają one po 16 bitów, ich znaczenie jest następujące :

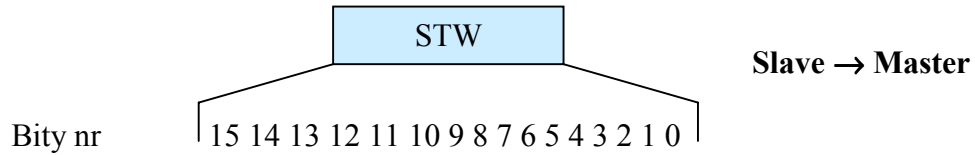
A) Słowo sterujące :



Nr bitu	wartość	znaczenie	komentarz
0	1	<i>ON</i>	Załączenie napięcia wyjściowego falownika na silnik.
	0	<i>OFF1</i>	Wyłączenie – częstotliwość zmniejsza się od wartości ustalonej do zera. Napięcie wyjściowe zdjęte przy częst < częst. min.
1	1	<i>Operating condition</i>	Komenda OFF2 wyłączona.
	0	<i>OFF2</i>	Natychmiastowe zdjęcie napięcia wyjściowego falownika.
2	1	<i>Operating condition</i>	Komenda OFF3 wyłączona.
	0	<i>OFF3</i>	Szybkie zatrzymanie, silnik zostaje zatrzymany tak szybko jak jest to możliwe. Podane jest napięcie DC z obwodu prądu stałego.
3	1	<i>Enable operation</i>	Zezwala na kontrolowanie i załączenie napięcia wyjściowego falownika.
	0	<i>Inhibit operation</i>	Wstrzymuje kontrolowanie i odłącza natychmiast napięcie wyjściowe.
4	1	<i>Enable ramp generator</i>	Umożliwia wytwarzanie napięcia wyjściowego falownika.
	0	<i>Inhibit ramp generator</i>	Wstrzymuje wytwarzanie napięcia wyjściowego.
5	1	<i>Start ramp generator</i>	Zezwala na zmianę wytwarzanego napięcia wyjściowego.
	0	<i>Stop ramp generator</i>	Wstrzymuje wytwarzanie zmiany napięcia wyjściowego. Zmiana częstotliwości jest niemożliwa. Falownik pracuje z obecną wartością częstotliwości.

6	1	<i>Enable setpoint</i>	Umożliwia pracę falownika z wybraną częstotliwością.
	0	<i>Inhibit setpoint</i>	Wartość częstotliwości jest ustawiana na 0.00 Hz.
7	1	<i>Aknowledge</i>	Zezwala na ponowną pracę po wystąpieniu błędu i wstrzymaniu pracy falownika.
	0	<i>No significance</i>	Bez znaczenia.
8	1	<i>Jog right</i>	Pełzanie (bardzo mała prędkość obrotowa) silnika w prawo .
	0	<i>No jog</i>	
9	1	<i>Jog left</i>	Pełzanie w lewo.
	0	<i>No jog</i>	
10	1	<i>Request control</i>	Kontrolowanie zdalne jest możliwe. Dane sterujące od Master'a są przyjmowane przez Slave'a.
	0	<i>No control</i>	Nie jest możliwe zdalne sterowanie. Wysyłanie słowa sterującego nie powoduje żadnej reakcji w falowniku.
11	1	<i>Request direction right</i>	Załączenie kierunku obrotów w prawo.
	0	<i>No action</i>	
12	1	<i>Request direction left</i>	Załączenie kierunków obrotów w lewo.
	0	<i>No action</i>	
13,14, 15	1	<i>Free</i>	Nie używane, ustawienie dowolne.
	0		

B) Słowo statusu :



Nr bitu	wartość	znaczenie	komentarz
0	1	<i>Ready - to switch ON</i>	Napięcie zasilania falownika włączone, elektronika zainicjalizowana, brak napięcia na wyjściu.
	0	<i>Not ready – to switch ON</i>	Trwa inicjalizacja falownika.
1	1	<i>Ready</i>	Falownik jest załączony, pracuje poprawnie, nie ma żadnych błędów. Może przyjmować komendy sterujące.
	0	<i>Not ready</i>	Przypadki: falownik nie pracuje, wystąpił błąd, wystąpiły komendy OFF2 lub OFF3, napięcie wyjściowe wstrzymane.
2	1	<i>Operation enabled</i>	Napięcie wyjściowe falownika załączone.
	0	<i>Operation inhibited</i>	Nastąpiło odłączenie napięcie wyjściowego.
3	1	<i>Fault</i>	Wystąpił błąd. Praca falownika wstrzymana. Numer błędu zawiera odpowiedni parametr.
	0	<i>Fault -free</i>	Nie ma błędu.
4	1	<i>No OFF2</i>	Komenda wyłącz2 nie występuje.
	0	<i>OFF2 present</i>	Wystąpiła komenda Wyłącz2.
5	1	<i>No OFF3</i>	Komenda wyłącz3 nie występuje.
	0	<i>OFF3 present</i>	Wystąpiła komenda Wyłącz2.
6	1	<i>Switch – on inhibit</i>	Falownik po komendach OFF2 lub OFF3 przechodzi w stan wstrzymania, nie można załączyć napięcia wyjściowego. Stan ten można usunąć wysyłając komendę OFF1.
	0	<i>No switch – on inhibit</i>	Stan wstrzymania pracy falownika zdjęty.

7	1	<i>Alarm</i>	Wystąpił stan ostrzegawczy, np. został przekroczony prąd wyjściowy. Falownik dalej pracuje.
	0	<i>No alarm</i>	Nie ma ostrzeżeń
8	1	<i>Not used</i>	Bit nie używany. Zawsze ustawiony na jedynekę logiczną.
9	1	<i>Remote control</i>	Falownik jest w trybie pracy zdalnej. Możliwe sterowanie przez łącze szeregowo.
	0	<i>Local control</i>	Tryb pracy lokalnej. Sterowanie może odbywać się z panelu operatorskiego lub listwy sterowania zewnętrznego.
10	1	<i>F reached</i>	Obecna częstotliwość wyjściowa falownika osiągnęła wartość zadaną.
	0	<i>F fallen-below</i>	Częstotliwość wyjściowa jest poniżej wartości zadanej.
11	1	<i>Clockwise</i>	Kierunek obrotów w prawo.
	0	<i>Not clockwise</i>	Inny kierunek obrotów.
12	1	<i>Counter - clockwise</i>	Kierunek obrotów w lewo.
	0	<i>Not counter clockwise</i>	Inny kierunek obrotów.
13,14, 15	1	<i>Not used</i>	Nie używane.
	0		

6.2.3. Programowy interfejs komunikacyjny

Znając budowę i właściwości protokołu USS, stworzony został programowy interfejs komunikacyjny. Do tego celu wykorzystana została standardowa kontrolka ActiveX systemu Windows. Służy ona do obsługi i sterowania portami szeregowymi komputera. Kontrolki ActiveX są gotowymi obiektami programowymi które można wykorzystywać nie tylko w języku programowania Visual Basic ale również w C++ czy Delphi.

Zawierają w sobie następujące cechy:

- właściwości, w których ustawiamy parametry kontrolki,
- metody, poprzez które sterujemy ich pracą,
- zdarzenia, które przekazują stan ich pracy oraz umożliwiają odpowiednią reakcję w programie.

Zastosowana kontrolka – *Microsoft Comm Control 6.0*, zawarta jest w katalogu systemowym Windows, w postaci pliku „mscomm32.ocx” i jest przeznaczona do aplikacji 32-bitowych.

W budowanym programie wykorzystywane są następujące ustawienia i komendy tej kontrolki :

a) właściwości:

- `MSComm1.CommPort` – określa numer portu,
- `MSComm1.Settings` – ustawia parametry transmisji,
- `MSComm1.RTSEnable` – załącza sygnał RTS (*Request To Send line* – żądanie wysłania) dla transmisji szeregowej,
- `MSComm1.InputLen` – wartość określająca ilość znaków poprawnie odczytanych z bufora wejściowego.

b) metody:

- `MSComm1.Output` – wysyła ciąg znaków na wyjście portu szeregowego,
- `MSComm1.PortOpen` – otwiera lub zamyka port,
- `MSComm1.Input` – czyta przychodzące znaki z portu,
- `MSComm1.InBufferCount` – zwraca ilość znaków zawartych w buforze wyjściowym portu.

Dzięki zastosowaniu tej kontrolki, możliwe było zbudowanie ciągłej transmisji między komputerem a falownikiem. Polegało to na stworzeniu w programie pętli, w której następuje wysyłanie i odbieranie danych przez port szeregowy. Pętla ta działa cały czas po ustanowieniu komunikacji z falownikiem.

Fragment kodu odpowiedzialny za transmisję szeregową:
(tekst po apostrofie nie jest częścią programu ale komentarzem)

‘Formatka Form1 (plik main.frm) – główne okno programu „Micro” – **Micro-Main control panel**, wirtualnego panelu sterowania.

```
Private Sub Form_Load()           ‘ procedura ładowaniu okna
```

```
...
```

```
MSComm1.CommPort = 2             ‘ ustawiony port COM2
```

```
MSComm1.RTSEnable = True        ‘ załączony sygnał RTS
```

```
...
```

```
MSComm1.Settings = "9600,e,8,1" ‘ prędkość 9600 Baud, bit parzystości, 8 bitów danych, 1 bit stopu
```

```
...
```

```
End Sub                          ‘ koniec procedury
```

```

'Formatka Form1 (plik main.frm) - główne okno programu.
Private Sub cmdConect_Click() ' procedura załączenia transmisji z
                              falownikiem, zdarzenie wciśnięcia przycisku
                              Connect
...
MSComm1.PortOpen = True      ' otwarcie portu
MSComm1.InputLen = 0         ' wyzerowanie bufora wejściowego
...
Do While Ok = 0              ' główna pętla odpowiedzialna za ciągłą
                              komunikację z falownikiem, wykonuje się
                              dopóki jest spełniony warunek "ok = 1".
                              Kiedy kończymy transmisję przyciskiem
                              Disconnect, wskaźnik ok = 1
...
    MSComm1.Output = Tablica(Ktory) ' wysłanie telegramu zawartego
                                    w tablicy do falownika
...
Do                             ' pętla odpowiedzialna za odczytanie
                              telegramu od falownika, wykonywana
                              dopóki w buforze wejściowym nie znajdzie
                              się 28 znaków
...
Loop Until MSComm1.InBufferCount = 28

acon$ = MSComm1.Input         ' przekazanie do zmiennej tekstowej całej
                              wiadomości odczytanej z portu, czyszczenie
                              bufora wejściowego
...
Loop
...
MSComm1.PortOpen = False     ' po wyjściu z głównej pętli – transmisja
                              wyłączona, następuje zamknięcie portu
...
End Sub

```

6.3. Sterowanie kartą zbierania danych

Mając zainstalowaną kartę zbierania danych w komputerze i zarejestrowaną w systemie za pomocą programu „ADDIREG”, konieczne było jej oprogramowanie. Możliwość obsługi i sterowania kartą daje w środowisku programowym, dostarczony z nią sterownik producenta - plik PA3000.dll. DLL (*Dynamic Link Library* - biblioteka dołączana dynamicznie) to skompilowany kod zawierający procedury i funkcje do których może odwoływać się dowolny program. Aby takie odwołanie było możliwe potrzebny jest API (*Application Programming Interface* - interfejs programowy aplikacji). W tym przypadku rolę takiego interfejsu w Visual Basic, spełnia moduł PA3000.bas dołączony razem z kartą. Zawiera on deklaracje wszystkich funkcji istniejących w pliku sterownika (DLL).

Fragment kodu modułu PA300.bas :

```
...
Declare Function i_PA3000_InitCompiler Lib "PA3000.DLL"
    (ByVal i_CompilerInit As Integer) As Integer

Declare Function i_PA3000_SetBoardInformationWin32 Lib
    PA3000.DLL" (ByVal s_Identifier As String, _
        ByVal i_AnalogInputChannelNbr As Integer, _
        pi_BoardHandle As Integer) As Integer
...
```

We fragmencie tym widać dwie przykładowe zadeklarowane funkcje karty.

Słowa *Declare Function* określają deklaracje funkcji, po nich występuje nazwa funkcji i słowo *LIB* (*Library* – biblioteka), które odnosi się do pliku biblioteki, w której znajduje się funkcja. Wartości w nawiasach określają parametry przekazywane do funkcji. Natomiast za nawiasami określone jest jakiego typu wartość ma być przez nią zwrócona. Funkcje w odróżnieniu od procedur zwracają po ich użyciu pewną wartość.

Mając do dyspozycji zadeklarowane funkcje, możemy w łatwy sposób się do nich odwoływać i w pełni sterować kartą zbierania danych. Poniżej zostanie przedstawiony i opisany fragment kodu, w którym wyjaśnione zostanie użycie kilku z nich. Fragment ten przedstawiać będzie część napisanego programu, w którym czytamy i zbieramy wartości chwilowe z jednego z wejść analogowych karty.

‘Formatka Form5 (plik wykres.frm) - okno **Oscilloscope**.

```
Private Sub C_ReadAnalogInput_Click() ‘procedura czytania jednego
    wejścia analogowego, zdarzenie wciśnięcia przycisku
Read one channel
```

...

```
i_ChannelArray.i_Value(0) = WYBOR ‘określenie numeru kanału do czytania
i_GainArray.i_Value(0) = PA3000_1_GAIN ‘ustawienie czułości na 1
i_PolarityArray.i_Value(0) = PA3000_BIPOLAR ‘ustawienie trybu pracy
    wejścia analogowego na bipolarny
```

...

```
PROBKI = CLng(Text2.Text) ‘określenie ilości próbek do zbierania
CZASPROB = CLng(Text1.Text) ‘czas próbkowania
```

...

‘w tym miejscu następuje użycie funkcji przeznaczonej do odczytania z wejścia analogowego jednej próbki. Funkcja ta jest wykonywana w pętli w celu określenia miejsca, od którego będziemy zbierać cały przebieg (tzw. wyzwalanie).

```
i_returnvalue = i_PA3000_Read1AnalogInput(i_BoardHandle,
    Kan_synch, PA3000_1_GAIN,
    PA3000_BIPOLAR, 7,
    PA3000_DISABLE, i_ReadValue)
```

‘funkcja wywoływana jest z takimi parametrami jak: uchwyt karty, kanał który czytamy, czułość itp. Przy czym w wartości `i_ReadValue` funkcja zapisuje wartość przeczytanej próbki. W zmiennej `i_returnvalue` funkcja zwraca wartość która określa poprawność jej wykonania (np. 0 – oznacza prawidłowe wykonanie bez błędów).

```
...
If (i_ReadValue <= 2060 And i_ReadValue >= 2020 And
i_ReadValue > pop) Or start1 = 1 Then
‘warunek jeśli znajdziemy punkt zerowy przebiegu (wartość zawarta w odpowiednim
przedziale), to od tego momentu nastąpi zbieranie próbek z przebiegu i zatrzymanie pętli.
```

```
...
i_returnvalue =
i_PA3000_InitAnalogInputAcquisition(i_BoardHandle, _
                                     1, _
                                     i_ChannelArray, _
                                     i_GainArray, _
                                     i_PolarityArray, _
                                     PA3000_SIMPLE_MODUS, _
                                     PA3000_DISABLE, _
                                     CZASPROB, _
                                     0, _
                                     PROBKI, _
                                     PA3000_DMA_USED, _
                                     PA3000_SINGLE)
```

‘funkcja `i_PA3000_InitAnalogInputAcquisition` ustawia odpowiednio parametry akwizycji i przygotowuje kartę na zbieranie próbek

```
...
i_returnvalue = i_PA3000_StartAnalogInputAcquisition
                 (i_BoardHandle) ‘funkcja ta rozpoczyna cykliczne
                               zbieranie ustalonej wcześniej ilości
                               próbek przebiegu
```

```
...
End If ‘koniec warunku
```

```
...
```

```
End Sub ‘koniec procedury
```

Dokładny opis poszczególnych funkcji, ich działania, parametrów i wartości jakie zostają przez nie zwracane, zawarta jest w specyfikacji technicznej karty [4].

W tworzonym oprogramowaniu do monitorowania i sterowania falownikowym układem napędu, zostały dodatkowo użyte następujące funkcje karty:

- `InitCompiler` – określa język programowania w którym karta jest używana,
- `SetBoardInformationWin32` – sprawdza wszystkie informacje sprzętowe o karcie z ustawionymi przy pomocy programu `ADDIREG`, sprawdza czy karta jest obecna,
- `SetBoardIntRoutineWin32` – ustawia parametry dla podprogramu przerw, używanego przy generowaniu przerwania występującego na zakończenie zbierania próbek wybranego przebiegu,

- `StopAnalogInputAcquisition` – zatrzymuje cykliczne zbieranie danych
- `ClearAnalogInputAcquisition` – deinstaluje kanał DMA (*Direct Memory Access* – bezpośredni dostęp do pamięci komputera), ustawiony przy użyciu funkcji `InitAnalogInputAcquisition`,
- `ResetBoardIntRoutine` – deinstaluje i zatrzymuje działanie podprogramu obsługi przerwań,
- `SetOutputMemoryOn` – aktywuje pamięć dla wyjść cyfrowych, wyjścia ustawione po zakończeniu działania programu dalej utrzymują ten stan,
- `SetOutputMemoryOff` – deaktywuje pamięć dla wyjść cyfrowych,
- `CloseBoardHandle` – zamyka dostęp do karty,
- `Read4DigitalInput` – odczytuje stan wszystkich czterech wejść cyfrowych,
- `Set1DigitalOutputOn` – ustawia pojedyncze wyjście cyfrowe w stan wysoki (jedynek logiczna),
- `Set1DigitalOutputOff` – kasuje pojedyncze wyjście i ustawia je w stan niski (zero logiczne).

Użyte funkcje umożliwiły w programie realizację następujących opcji:

- zbierania przebiegów wartości chwilowych napięć i prądów, z jednego lub z wielu wejść analogowych karty jednocześnie,
- wykorzystania kanału DMA, dzięki czemu możliwe było wykorzystanie maksymalnej częstotliwości próbkowania karty ($f = 142 \text{ kHz}$),
- sterowania wyjściami cyfrowymi,
- zbierania informacji o stanach wejść cyfrowych.

6.4. Opis programu „Micro”

6.4.1. Wstęp

Program do zdalnego monitorowania i sterowania falownikowego układu napędu, został napisany w języku Visual Basic 6.0. Ze względu na stabilność pracy, pełne wykorzystanie możliwości karty zbierania danych i pełnienie dodatkowej roli serwera dla komunikacji poprzez sieć Intranet/Internet, program ten przystosowany został do pracy pod systemem Windows NT. Rozwiązania tego typu układów mogą być realizowane tylko w tym lub nowszym (Windows 2000) systemie komputerowym. System ten przystosowany jest do zastosowań przemysłowych i rozbudowanych systemów sieciowych. Praca programu pod systemami Windows 95/98 jest możliwa ale będzie ograniczona do funkcji monitoringu i sterowania przez łącze szeregowo oraz wykorzystywania niektórych funkcji serwera obsługującego komunikację w sieci komputerowej. Opcje programu związane z wykorzystywaniem wejść analogowych karty zbierania danych, pod systemami Windows 9x nie będą mogły być realizowane.

Minimalne wymagania programu:

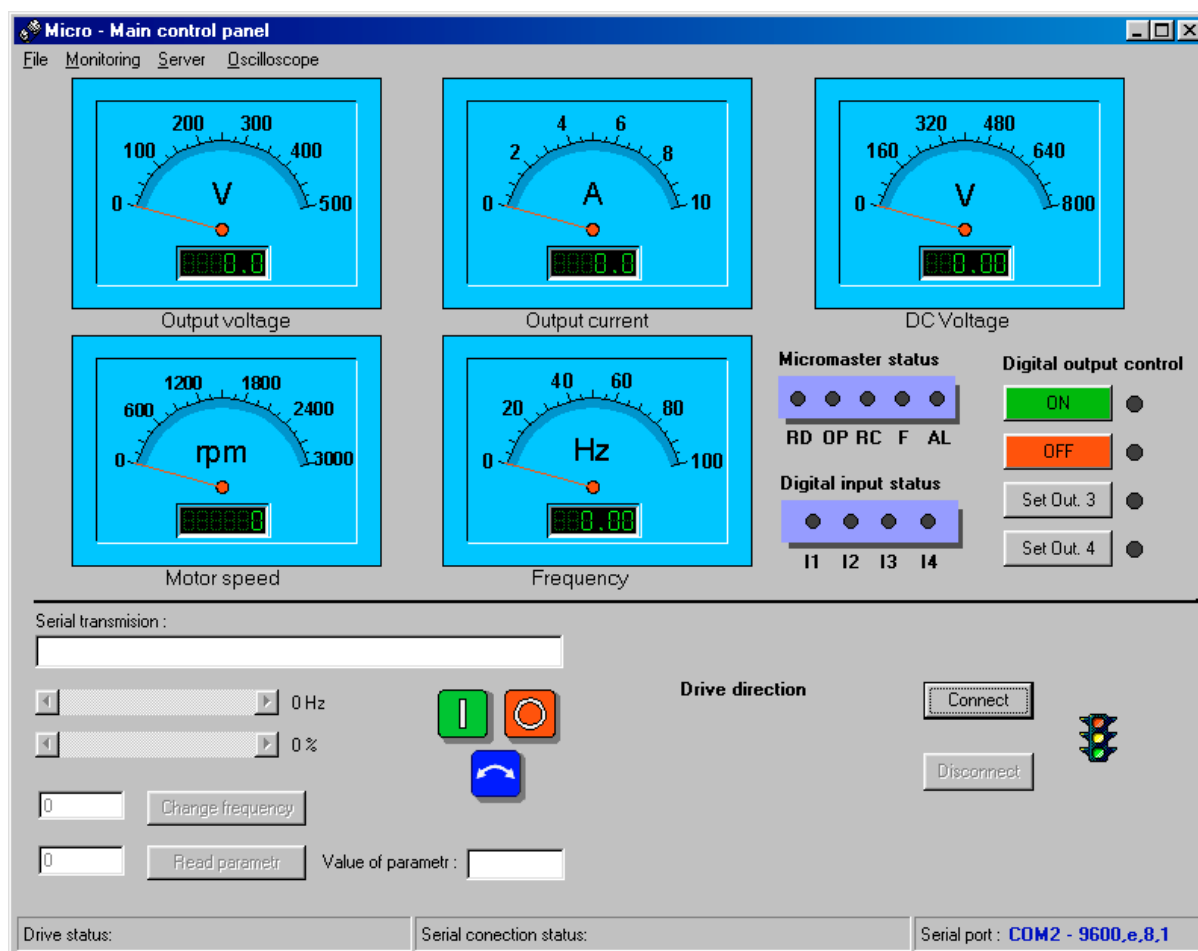
- komputer klasy PC z procesorem Pentium 200 lub szybszym,
- 8 MB pamięci,
- karta grafiki SVGA pracująca przy rozdzielczości min. 800x600, z ustawieniami palety kolorów na *High color* – 16 bitowe. Zalecana rozdzielczość 1024x768,
- karta sieciowa o transferze 10 Mb/s,
- 10 MB wolnej przestrzeni dyskowej,
- do korzystania z opcji **Oscilloscope**, wymagana jest karta zbierania danych - PA3000 firmy ADDIDATA.

Program został tak napisany aby nie nastęczał żadnych kłopotów w obsłudze.

Większość oznaczeń skrótowych, przycisków, okienek przeznaczonych do wprowadzania i wyświetlania danych, zawiera podpowiedź tekstową pojawiającą się po naprowadzeniu myszki na dany obiekt. Cały program został zbudowany w postaci okienek i posiada ogólną budowę taką jaką występuje w standardowych aplikacjach systemu Windows. Obsługa poszczególnych opcji programu odbywa się za pomocą myszki i/lub klawiatury.

6.4.2. Opis poszczególnych okien programu

Opcje programu wybierane są z menu początkowego okna **Micro – Main control panel**. Przedstawia ono wirtualny panel, sterujący pracą falownikowego układu napędu.



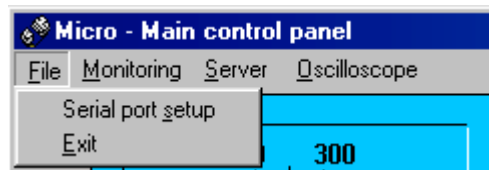
Rys. 20. Widok wirtualnego panelu sterującego

Opis tych opcji jest następujący:

File – menu plik zawierające:

Serial port setup – ustawienia dla portu szeregowego,

Exit – wyjście z programu.



Monitoring – menu monitoringu zawierające:

Micromaster words – podgląd słów: statusu i sterowania protokołu USS oraz transmisji szeregowej między komputerem a falownikiem

Parameters – przegląd wszystkich parametrów falownika.



Server – serwer obsługujący przychodzące połączenia w sieci komputerowej i żądania, od aplikacji „Client TCP/IP” i “Klient WWW”. Opis i działanie tego okna zostanie omówione w rozdziale: 7.2.3. Opis i algorytm działania serwera,

Oscilloscope – wirtualny dwukanałowy oscyloskop.

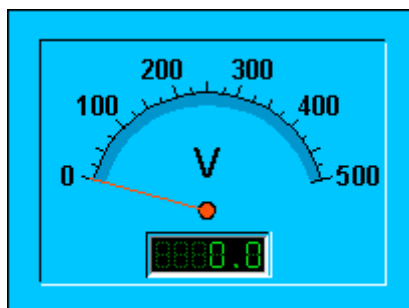


Okno głównego panelu sterowania - Micro – Main control panel

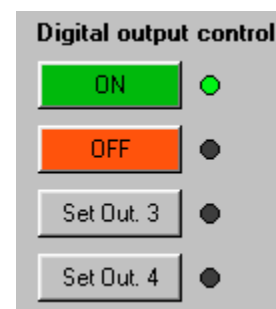
Jego widok przedstawia rysunek 20.

Wirtualny panel sterujący zawiera następujące komponenty:

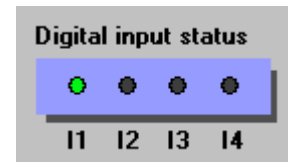
- Sześć mierników analogowo-cyfrowych, wizualizujących wartości otrzymywane z falownika poprzez łącze szeregowo,



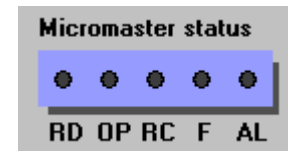
- obsługę wyjść cyfrowych – **Digital output control**, z których dwa z nich sterują załączeniem i wyłączeniem stycznika głównego obwodu zasilania falownika. Pozostałe dwa nie mają przypisanej funkcji w układzie ale są przystosowane programowo i sprzętowo do ich użycia,



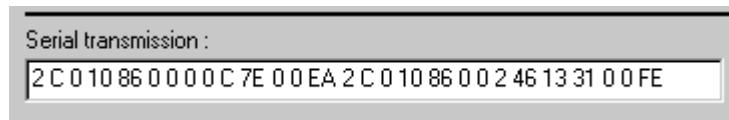
- stan wejść cyfrowych – **Digital input status**, określający stan przełączników z lokalnego panelu sterującego stanowiska laboratoryjnego,



- status falownika – **Micromaster status**, zawierający podstawowe informacje o aktualnym stanie jego pracy, np. czy jest gotowy, czy nie ma żadnych błędów itp.



- Podgląd transmisji szeregowej – **Serial Transmission**, przedstawia telegramy wysyłane i odbierane między komputerem i falownikiem,

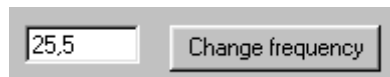


- Zespół przycisków umożliwiających:

- sterowanie pracą falownika (START, STOP, NAWRÓT),



- zmianę wart. częstotliwości jego napięcia wyjściowego - **Change frequency**, wprowadzanej w pole tekstowe,



- czytanie pojedynczego parametru falownika - **Read parameter**



- załączenie – **Connect** oraz rozłączenie – **Disconnect**, transmisji szeregowej między komputerem a falownikiem



- Dwa paski przewijania – umożliwiające zadawanie wartości częstotliwości zarówno w Hz jak i w % (częstotliwości $f=50$ Hz odpowiada 100%, zakres częstotliwości jest ograniczony programowo do 100 Hz)



- Wskaźnik kierunku obrotów silnika – **Drive direction**:

- obroty w lewo



- obroty w prawo



- Wskaźnik stanu komunikacji szeregowej:



- połączenie nieaktywne



- połączenie aktywne, poprawna transmisja



- połączenie przerwane, błąd transmisji

- Pasek statusu znajdujący się na dole okna:



- status falownika – **Drive status**, wskazujący:
 - gotowość do pracy (**Ready**),
 - brak gotowości do pracy (**Not Ready**)
 - wystąpienie błędu (**Fault**),
 - odłączenie od komputera (**Disconnected from computer**),
 - ostatni stan pracy przed utratą komunikacji z komputerem (**Last status - Ready**),
 - wystąpienie ostrzeżenia (**Alarm !**)
- status połączenia szeregowego – **Serial connection status**, wskazujący:
 - zamknięcie połączenia (**Closed**),
 - próba nawiązania komunikacji komputera z falownikiem (**Connecting.....**),
 - błąd w transmisji, błędna transmisja (**Transmission is fault**),
 - transmisja prawidłowa (**Transmission...OK**),
 - utrata komunikacji z falownikiem (**Lost comunication with inverter**),
- aktualne ustawienia portu szeregowego – **Serial port**:
(ustawienie standardowe: "**COM2 - 9600,e,8,1**")

Przy budowie mierników analogowo-cyfrowych wykorzystano filtr cyfrowy [23].

Jego zadaniem było urzeczywistnienie zachowań wskazówek mierników.

Przy odczycie wartości cyfrowych, które otrzymujemy przez łącze szeregowe, można zauważyć, że zmieniają się one skokowo. Spowodowane jest to tym, że falownik próbuje wartości mierzone z małą częstotliwością. Mając nawet do dyspozycji szybko zbierane próbki, ograniczeniem była by prędkość przesyłu ich przez łącze szeregowe.

Odpowiedź filtru wyznaczana jest na podstawie poprzednich wartości zadanych i nowych wyliczonych. W efekcie wskazówka zachowuje się tak, że płynnie dochodzi do ustalonej wartości. Odnosi się wrażenie, że miernik zachowuje się jak „prawdziwy” miernik analogowy.

Zależność pozwalająca na wyznaczenie aktualnej wartości wskazania miernika:

$$y_{(k)} = -(A_m(2) \cdot y_{(k-1)} + A_m(3) \cdot y_{(k-2)}) + B_m(1) \cdot ref_{(k-1)} + B_m(2) \cdot ref_{(k-2)} \quad (2)$$

oznaczenia:

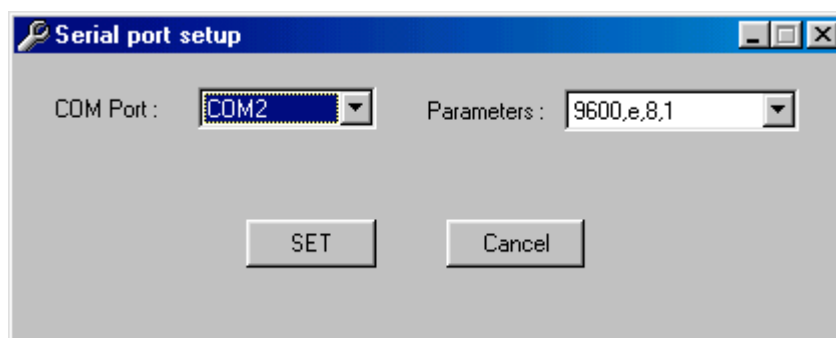
- $y_{(k)}$ - wartość wyjściowa wyliczona w chwili bieżącej
- $y_{(k-1)}$ - wartość wyjściowa poprzednia, wyliczona w chwili k-1
- $y_{(k-2)}$ - wartość wyjściowa poprzednia, wyliczona w chwili k-2
- $ref_{(k-1)}$ - wartość zadana w chwili k-1
- $ref_{(k-2)}$ - wartość zadana w chwili k-2
- A_m, B_m - parametry filtra
- $A_m = [1 \ -1.4 \ 0.49]$
- $B_m = [0.06 \ 0.03]$

W pisany programie zastosowano filtr w ten sposób, że co pewien krótki przedział czasu, dokonuje się kilkudziesięciu obliczeń na podstawie wzoru (2).

Wartość wyliczona bieżąca $y_{(k)}$ jest podawana do miernika. Wartością zadaną ref w chwili bieżącej jest wartość cyfrowa otrzymana przez łącze szeregowo. W kolejnych chwilach wartość ref staje się $ref_{(k-1)}$, a $ref_{(k-1)}$ staje się $ref_{(k-2)}$. Wartości parametrów filtra są stałe i nie podlegają zmianie.

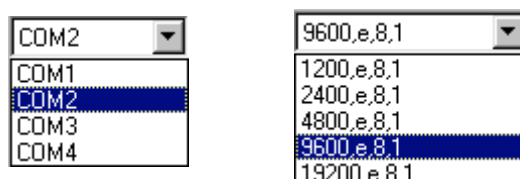
Okno ustawień portu szeregowego - Serial port setup

Umożliwia ustawienie parametrów transmisji szeregowo. Wartości tych ustawień są zapamiętywane przez program. Standardowe ustawienia przedstawia poniższy rysunek.



Rys. 21. Widok okna do ustawień parametrów transmisji szeregowo

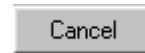
Wybór portu szeregowego COM i parametrów transmisji odbywa się za pomocą rozwijanych list.



Przycisk **SET** – służy do zaakceptowania i ustawienia nowych wartości.



Przycisk **Cancel** – powoduje anulowanie wybranych wartości zamknięcie okna.



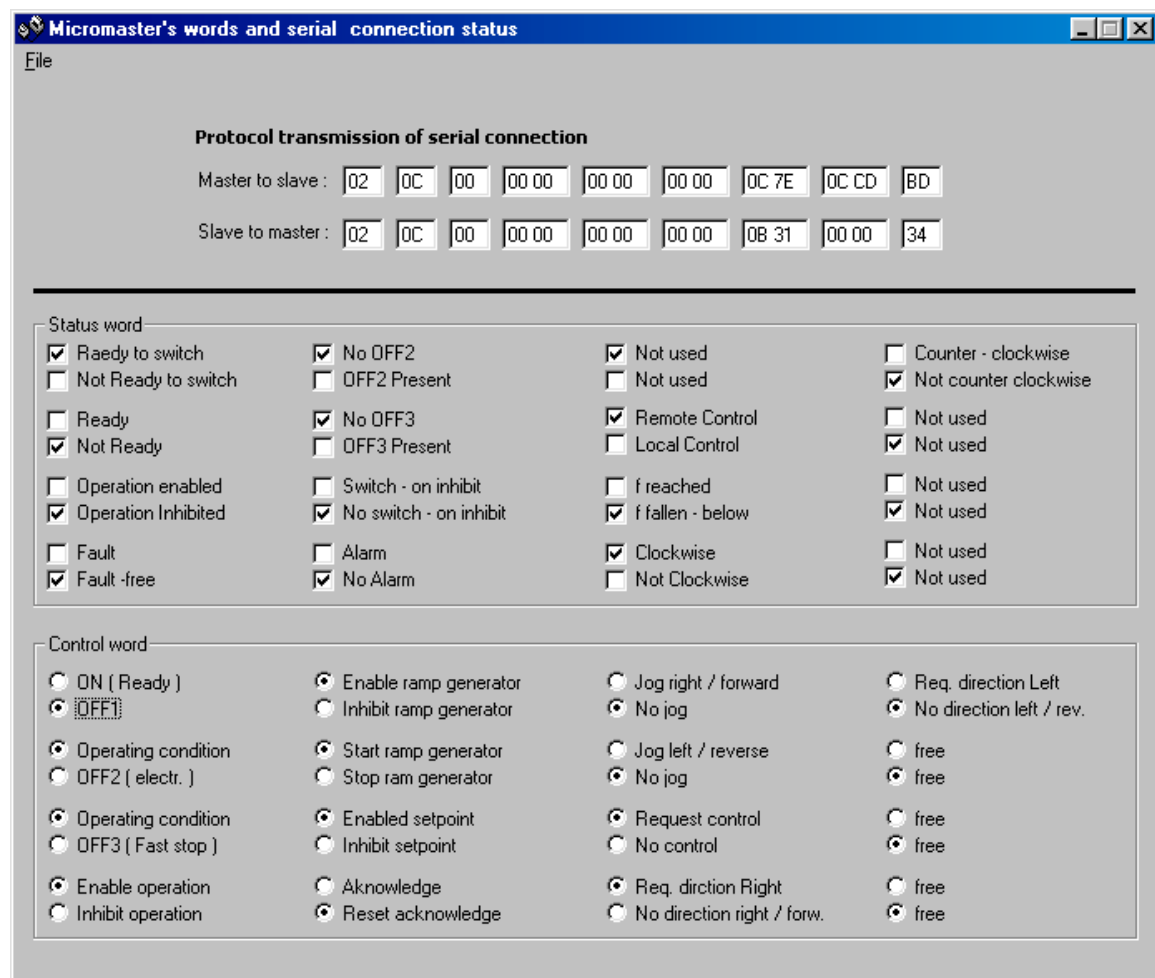
Opis parametrów transmisji szeregowej zawarty jest w rozdziale 6.2.1.

Okno podglądu połączenia szeregowego i słów protokołu USS falownika - Micromaster's words and serial connection status

Wygląd okna przedstawia rysunek 22.

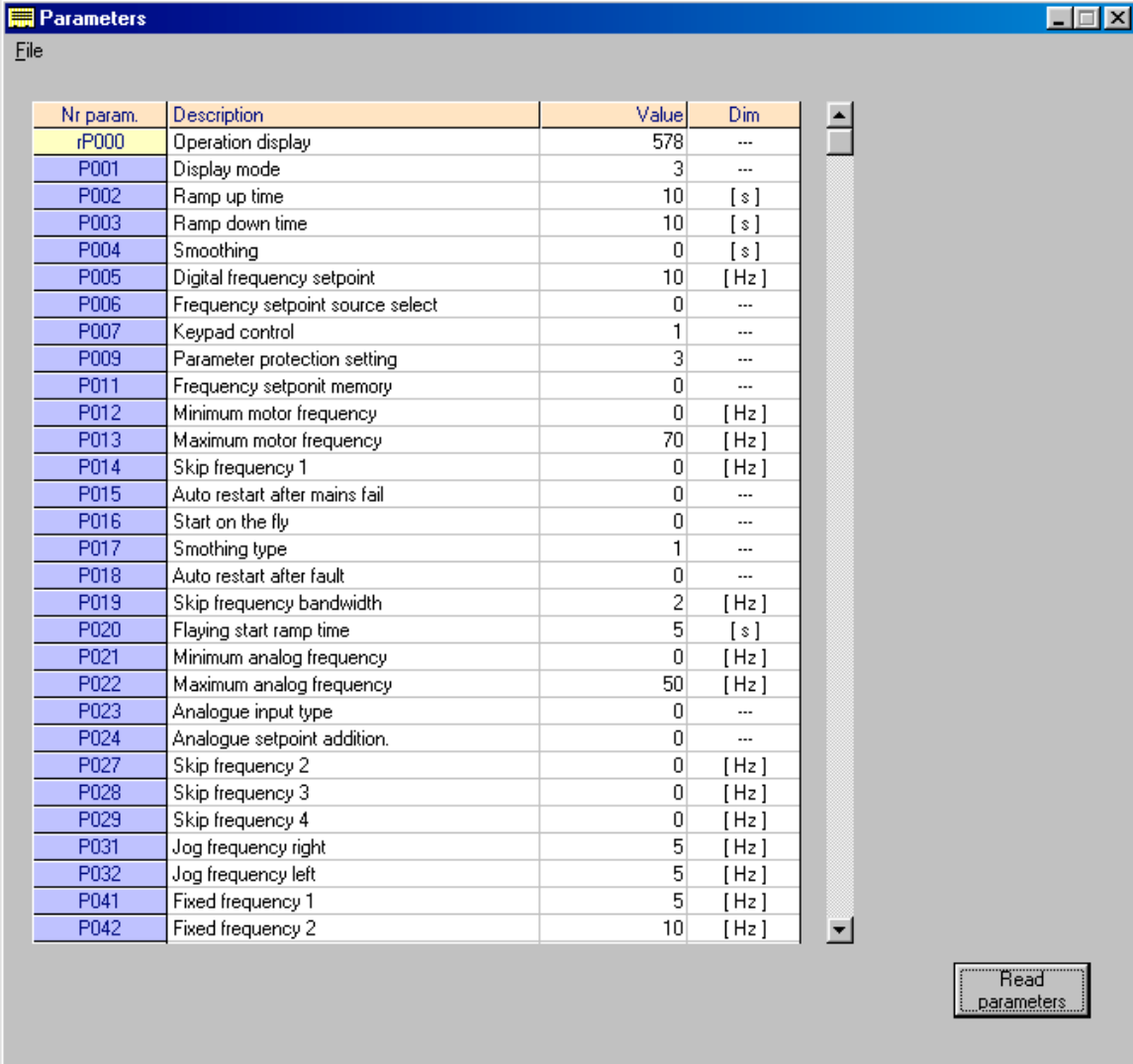
Pokazane w nim zostały poszczególne bity słowa sterującego - **Control word**, oraz słowa statusu - **Status word**. Przedstawiony został monitoring transmisji szeregowej z opisem poszczególnych bajtów protokołu USS. Wybierając opcje przycisku **optionbutton** (przycisk wyboru), w słowie sterującym ustawiane są poszczególne stany bitów. Umożliwia to sterowanie pracą falownika.

Na podstawie słowa statusu, rozłożonego na poszczególne bity, których wartości przedstawiane są w postaci pól **checkbox'ów**, możliwy jest podgląd stanu pracy falownika. Opis poszczególnych bitów i funkcji jakie spełniają, zawarta jest w rozdziale 6.2.2.



Rys. 22 Wygląd okna monitoringu połączenia szeregowego i telegramów falownika.

Okno parametrów falownika - Parameters



Nr param.	Description	Value	Dim
rP000	Operation display	578	...
P001	Display mode	3	...
P002	Ramp up time	10	[s]
P003	Ramp down time	10	[s]
P004	Smoothing	0	[s]
P005	Digital frequency setpoint	10	[Hz]
P006	Frequency setpoint source select	0	...
P007	Keypad control	1	...
P009	Parameter protection setting	3	...
P011	Frequency setpoint memory	0	...
P012	Minimum motor frequency	0	[Hz]
P013	Maximum motor frequency	70	[Hz]
P014	Skip frequency 1	0	[Hz]
P015	Auto restart after mains fail	0	...
P016	Start on the fly	0	...
P017	Smoothing type	1	...
P018	Auto restart after fault	0	...
P019	Skip frequency bandwidth	2	[Hz]
P020	Flaying start ramp time	5	[s]
P021	Minimum analog frequency	0	[Hz]
P022	Maximum analog frequency	50	[Hz]
P023	Analogue input type	0	...
P024	Analogue setpoint addition.	0	...
P027	Skip frequency 2	0	[Hz]
P028	Skip frequency 3	0	[Hz]
P029	Skip frequency 4	0	[Hz]
P031	Jog frequency right	5	[Hz]
P032	Jog frequency left	5	[Hz]
P041	Fixed frequency 1	5	[Hz]
P042	Fixed frequency 2	10	[Hz]

Rys. 23. Wygląd okna parametrów.

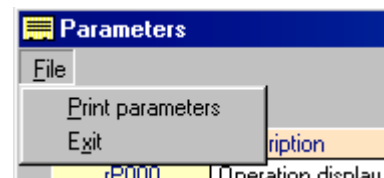
Przedstawione na rysunku 23. okno umożliwia: czytanie, wyświetlanie i zmianę parametrów falownika, a także możliwość ich wydrukowania.

Posiada następujące menu:

File – menu plik zawierające:

Print parameters – wydruk parametrów

Close – zamknięcie okna



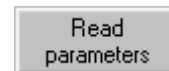
Spis wszystkich parametrów wyświetlany jest w postaci tabeli.

Nr param.	Description	Value	Dim
rP000	Operation display	575	---
P001	Display mode	3	---
P002	Ramp up time	10	[s]

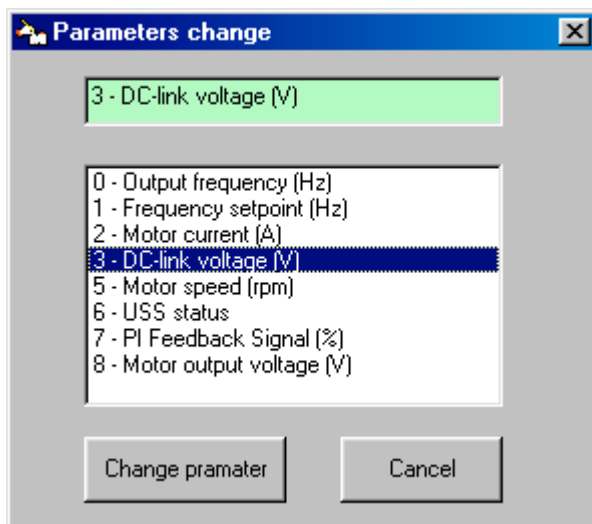
Posiada ona nagłówki o następujących oznaczeniach :

- **Nr param.** – numer parametru. Pola w tej kolumnie zawierające literę „r” z przodu np. „rP000”, określają parametry tylko do odczytu. Wartości pozostałych parametrów mogą być zmieniane,
- **Description** – zawiera krótki opis informujący o funkcji jaką spełnia dany parametr,
- **Value** – wartość parametru,
- **Dim** – jednostka parametru (symbol „---”, oznacza, że dany parametr nie posiada jednostki).

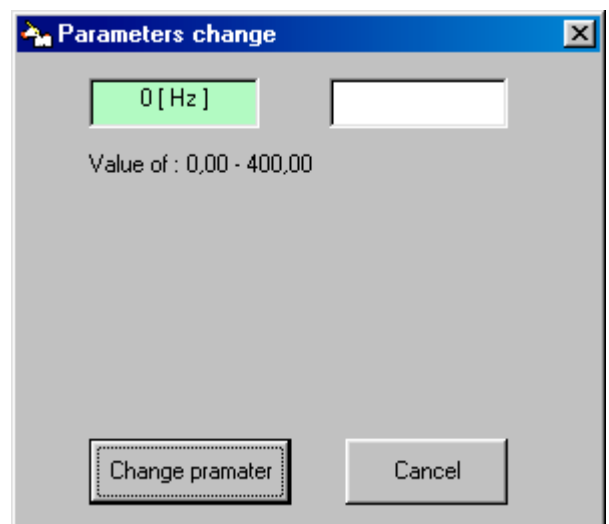
Przycisk **Read parameters** - służy do przeczytania wszystkich parametrów falownika.



Zmiana wartości parametru następuje przez dwukrotne kliknięcie myszką na wybranym wierszu parametru. W zależności czy chcemy zmienić parametr w którym wybieramy pewną opcję, czy też parametr określający wartość np. częstotliwości, pojawia się odpowiednie do tego przeznaczone okienko. Wygląd tych okienek przedstawiają rysunki: 24. i 25.

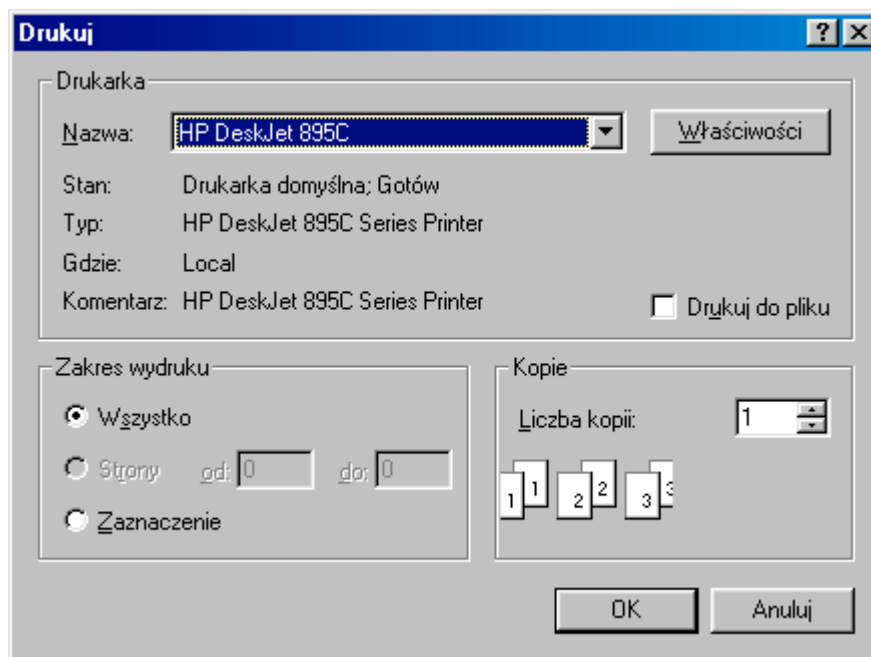


Rys. 24. Okno zmiany parametru – wybór opcji



Rys. 25. Okno zmiany parametru – zmiana wartości

Opcje wydruku parametrów - **Print parameters** - obsługuje standardowe okienko sterujące wydrukami w systemie Windows:



Rys. 26. Wygląd okna wydruku

Okno oscyloskopu - Oscilloscope

Służy do obserwowania i odczytywania przebiegów wartości chwilowych napięć i prądów jakie dostarcza karta DAQ z układu pomiarowego.

Posiada następujące opcje menu:

File – menu plik:

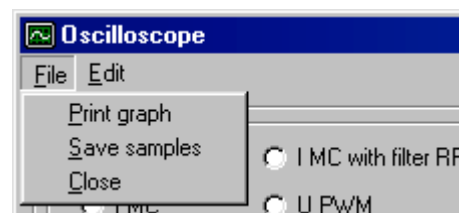
Print graph – umożliwia wydruk

przebiegów z okna oscyloskopu,

Save samples – pozwala zapisać wszystkie
zebrane próbki przebiegu do pliku.

Plik po zapisaniu zawiera w kolumnie wartości kolejnych próbek, w postaci liczb z zakresu od 0 do 4096 (zakres zbierania wejść analogowych karty),

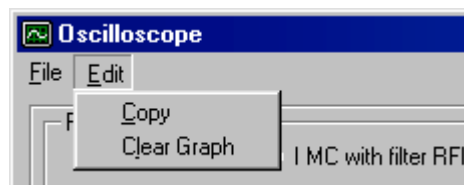
Close – zamknięcie okna oscyloskopu.

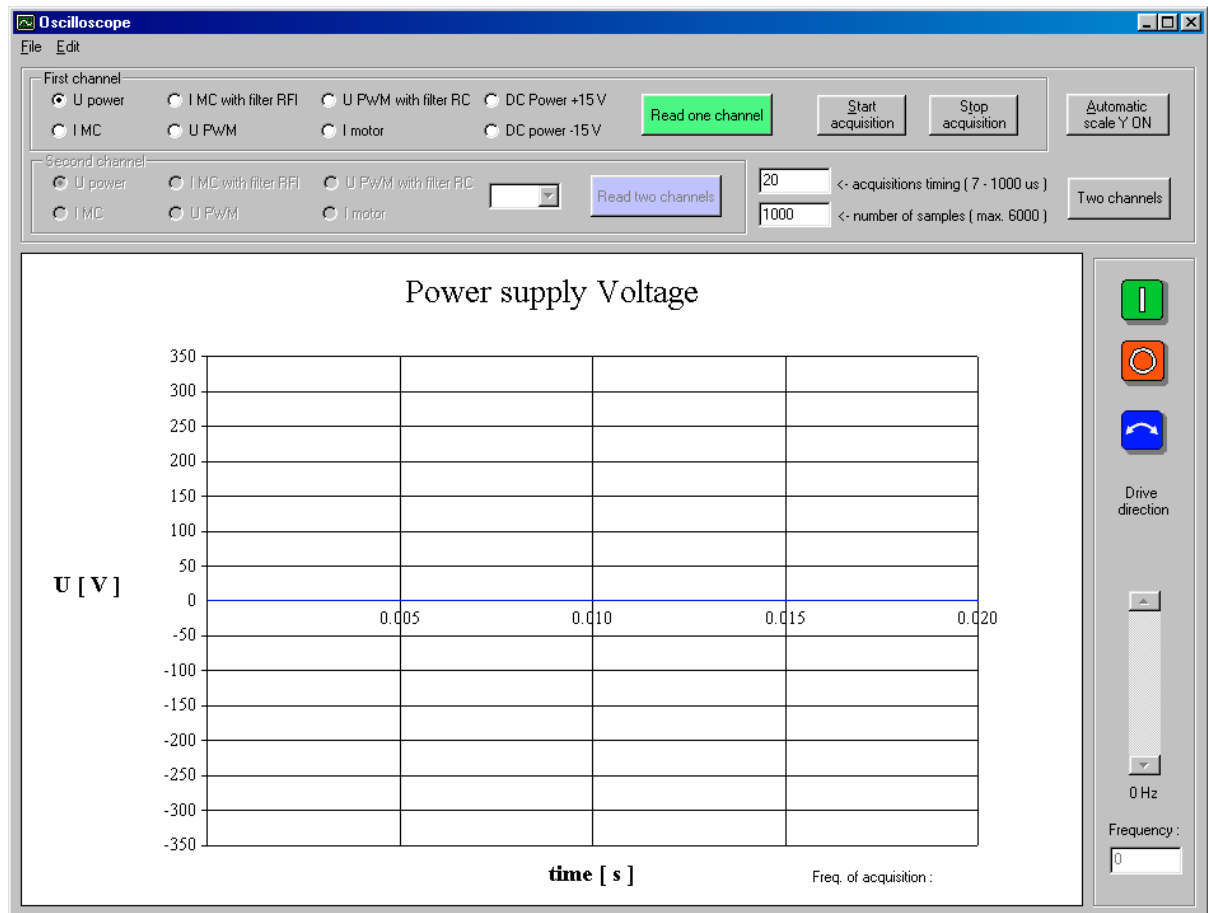


Edit – menu edycji:

Copy – opcja ta umożliwia skopiowanie okna przebiegów do schowka systemu Windows i wklejenia go do innego programu, np. Microsoft Word.

Clear graph – czyści zawartość okienka oscyloskopu.





Rys. 27. Okno oscyloskopu

Umożliwia obserwowanie i zbieranie przebiegów następujących wartości:

- **U power** – napięcie fazowe zasilania falownika,
- **I_{MC}** – prąd wejściowy falownika,
- **I_{MC with filter RFI}** – prąd wejściowy falownika z filtrem RFI,
- **U_{PWM}** – napięcie wyjściowe falownika,
- **U_{PWM with filter RC}** – napięcie wyjściowe falownika z filtrem typu RC,
- **I motor** – prąd wyjściowy przewodowy falownika,
- **DC power +15V** – napięcie +15 V zasilacza stabilizowanego,
- **DC power -15V** – napięcie -15V zasilacza stabilizowanego.

Opis przycisków:

Read one channel - czytaj jeden kanał, umożliwia zbieranie próbek z jednego wybranego kanału.



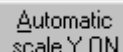
Read two channels – czytaj dwa kanały, próbki zbierane są z dwóch wybranych kanałów jednocześnie.



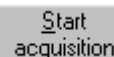
Two channels/One channels – dwa kanały, przycisk wyboru między opcjami czytania dla jednego lub dwóch kanałów.



Automatic scale Y ON / Automatic scale OFF – załącza lub wyłącza automatyczne skalowanie wartości osi Y czytanego przebiegu.

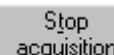


Start acquisition – rozpoczyna cykliczne zbieranie danych wybranego kanału (opcja jest dostępna tylko przy czytaniu z jednego kanału)



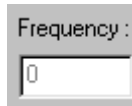
z

Stop acquisition – zatrzymuje cykliczne zbieranie danych z kanału

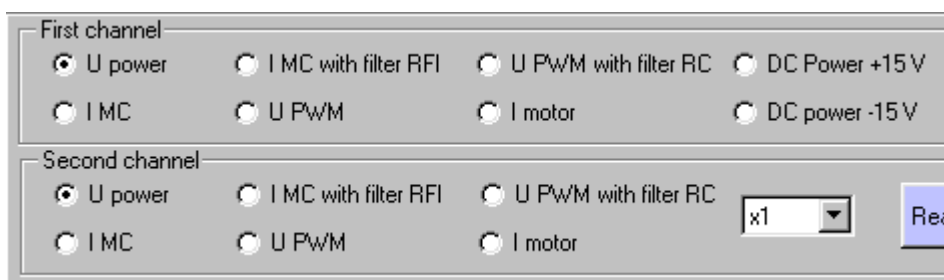


Dodatkowo w oknie tym znajdują się przyciski sterowania silnikiem (START, STOP, NAWRÓT) oraz pasek przewijania służący do zadawania częstotliwości napięcia wyjściowego falownika. Służy to wygodzie używania programu i umożliwia sterowanie pracą falownika bez przełączania się między tym oknem a oknem głównego panelu sterowania. Komponenty te są aktywne, gdy jest ustanowione połączenie z falownikiem.

Pole **Frequency** (częstotliwość), wyświetla aktualną częstotliwość falownika.



Wybór kanału który chcemy odczytać odbywa się za pomocą przycisków wyboru (**optionbutton**).

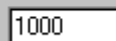


The screenshot shows two sections for channel selection: 'First channel' and 'Second channel'. Each section contains radio buttons for various measurement options: U power, I MC, I MC with filter RFI, U PWM, I motor, U PWM with filter RC, DC Power +15 V, and DC power -15 V. The 'Second channel' section also includes a dropdown menu set to 'x1' and a 'Real' button.

Parametry zbierania próbek wpisywane są w następujące pola tekstowe:

Acquisition timing – czas próbkowania, występujący między dwoma zebranymi próbkami. Zakres wprowadzanej wartości wynosi od 7 do 1000 μ s


 <- acquisitions timing (7 - 1000 us)

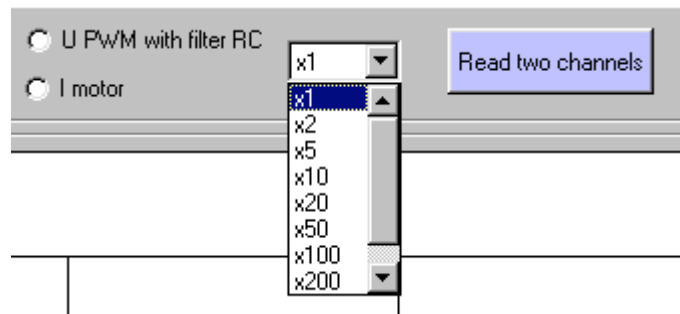

 <- number of samples (max. 6000)

Number of samples – liczba zbieranych próbek. Ilość wprowadzanych próbek zawarta jest w zakresie od 2 do 6000.

Pole **Freq. of acquisition** (częstotliwość próbkowania), zawarte w oknie wyświetlanych przebiegów, określa aktualną częstotliwość z jaką zostały zebrane próbki danego kanału.

Ustawiając odpowiednią ilość próbek przy stałym czasie próbkowania, uzyskujemy powiększenie (rozciągnięcie) przebiegu w osi X. Zmiana czasu próbkowania ma wpływ na dokładność oglądanego przebiegu. Minimalny czas jaki można uzyskać to 7 μ s, odpowiada to maksymalnej częstotliwości próbkowania równej 142,857 kHz .

Przy wyborze opcji czytania dwóch kanałów jednocześnie, możliwe jest powiększenie wartości drugiego kanału względem pierwszego. Do tego celu służy mnożnik wybierany z rozwijalnej listy znajdującej się w opcjach drugiego kanału.



Ustawienie tego mnożnika, umożliwia przedstawienie np. prądu falownika (I_{MC}) o amplitudzie kilku amperów, na tle napięcia zasilania (**U power**) o amplitudzie kilkuset voltów. Opcja ta była wymagana ze względu na to, że wartości osi Y przy wyświetlaniu przebiegów są wspólne dla obu kanałów.

6.4.3. Algorytm działania programu

Podczas uruchomienia programu dokonuje się wiele ustawień i poleceń których użytkownik nie widzi. Pozwalają one na przygotowanie i sprawdzenie komponentów programu zanim będzie można z niego korzystać oraz na ustawienie początkowych wartości programu. Napisany program, przede wszystkim opiera się na obsłudze pewnych zdarzeń jakie zostają wygenerowane przez użytkownika, który dokonuje obsługi programu, np. przez wciśnięcie przycisku lub wybranie danej opcji. Oczywiście pewne rzeczy w programie wykonują się cyklicznie i mogą trwać do momentu wyjścia z programu jak i też powtarzać się z pewnymi odstępami w czasie. Wiele czynności może dokonywać się jednocześnie w tym samym czasie. Poniżej zostanie opisany ogólny algorytm działania programu i poszczególnych jego opcji.

Początek działania programu

W czasie gdy program zostaje uruchomiany występuje ładowanie głównego okna (Rys. 20), **Micro – Main control panel** (zdarzenie `Form_Load` formatki `Form1`). Wykonywane są następujące czynności:

- ❑ ustawienie interfejsu graficznego użytkownika (pozycja okna , wygląd),
- ❑ ustawienie początkowych parametrów dla transmisji szeregowej,
- ❑ inicjalizacja karty DAQ (wywołanie określonych funkcji stwierdzających jej obecność oraz ustawienie jej parametrów),
- ❑ ustawienie pewnych zmiennych programowych,
- ❑ uruchomienie procedury `Timer2` (Timer - jest to kontrolka, która zawiera w sobie część kodu programu, która wykonuje się co pewien ustalony w jej właściwościach czas. W tym przypadku co pół sekundy).

Jeśli wystąpią jakieś błędy podczas wykonywania tych czynności, użytkownik dostanie informacje o rodzaju błędu i jego przyczynie.

Procedura `Timer2`, od tego momentu działa przez cały czas do zakończenia programu. Obsługuje następujące funkcje:

- jeśli jest ustanowione połączenie z falownikiem to na podstawie telegramu transmisji (słowa statusu) ustala:
 - kierunek obrotów silnika,
 - stan pracy falownika (**Drive status**),
 - opcje statusu falownika (**Micromaster status**)
- czyta stan wejść cyfrowych i wyświetla je w polu **Digital input status**,
- zbiera stan wejść i wyjść cyfrowych dla opcji **Server**'a (serwer umożliwia przesyłanie tych informacji poprzez sieć komputerową do programów: „**Client TCP/IP**”).

Po uruchomieniu czekamy na reakcję użytkownika. Ma on możliwość wyboru poszczególnych opcji z menu programu. Przyciski sterujące pracą falownika nie są aktywne dopóki nie nastąpi nawiązanie z nim komunikacji. Możliwe jest tylko sterowanie wyjściami cyfrowymi **Digital output control** i załączenie transmisji szeregowo z falownikiem.

W momencie początkowym programu, wyjścia cyfrowe odpowiedzialne za załączenie napięcia na poszczególne przełączniki w bloku stycznika, są w stanie niskim („0” logiczne – brak napięcia na cewkach przełącznika). Kiedy zostanie załączone wyjście pierwsze (poprzez przycisk **ON** programu w **Digital output control**), pojawia się na nim stan wysoki i załącza odpowiedni przełącznik. W momencie wciśnięcia przycisku **OFF**, wyjście drugie cyfrowe jest ustawiane w stan wysoki, natomiast pierwsze ustawiane jest z powrotem w stan niski, aby nie było sytuacji w której oba przełączniki są załączone jednocześnie. Gdy znowu zostanie wciśnięty przycisk **ON**, sytuacja jest odwrotna – wyjście pierwsze ustawia się w stan wysoki a drugie w stan niski.

Kiedy nastąpi wyjście z programu, przerwanie jego pracy lub wyłączenie komputera, oba wyjścia zostają ustawione w stan początkowy (stan niski, brak napięcia na cewkach przełączników).

Gdy zostanie wciśnięty przycisk **Connect** (zdarzenie `cmdConnect_Click` tej formatki), uruchamia się główna procedura obsługi transmisji szeregowo (komunikacji z falownikiem). Port szeregowo zostaje otwarty i podejmowana jest próba nawiązania komunikacji. Jeśli to nastąpi, uruchamiane są poszczególne czynności:

- załączenie panelu sterowania
- włączone zostają wirtualne mierniki i obsługa procedury `Timer1` (wyliczenia filtru cyfrowego i wyznaczanie pozycji wskazówek mierników). Procedura ta trwa do momentu zakończenia lub przerwania komunikacji z falownikiem. Jej wyłączenie powoduje także wybranie opcji **Server**'a, kiedy to aplikacja przygotowana jest do obsługi komunikacji sieciowej i zwiększone jest obciążenie procesora
- uruchomiona zostaje pętla do obsługi transmisji szeregowo i wymiany telegramów między komputerem a falownikiem. Pętla ta działa tak szybko jak pozwala na to system i dostarcza aktualnych wartości dla mierników. Pozwala ona także na czytanie parametrów falownika, ich zapis oraz jego monitoring i sterowanie. Jest to pętla główna, która działa w tle i dostarcza danych do poszczególnych bloków programu. Dzięki niej możliwe jest przekazywanie informacji z falownika do **Server**'a, który przesyła je dalej poprzez sieć komputerową.

Gdy zostanie wciśnięty przycisk **Disconnect** (zdarzenie `cmdDisconnect_Click`), główna pętla przerywa swoje działanie. Mierniki zostają wyłączone, przerwana jest praca procedury `Timer1` oraz następuje wysyłanie odpowiedniego słowa sterującego do falownika aby zakończył swoje działanie. Na koniec działania tej procedury zostaje zamknięty port szeregowy. To samo słowo sterujące wysyłane jest także przy zakończeniu działania całego programu.

Opcja **Micromaster's words**

Po jej wybraniu ładowane jest okno **Micromaster's words and serial connection status** (Rys. 22) – formatka *Form2*. Następnie uruchamiana jest procedura `Timer1` tej formatki. Obsługuje ona następujące czynności :

- wyświetla poszczególne bajty słów telegramów transmisji szeregowej w polach tekstowych,
- wyświetla stany poszczególnych bitów słowa statusu za pomocą zaznaczenia lub odznaczenia odpowiednich pól na oknie.

Jej działanie trwa dopóki nie zostanie zamknięte okno.

Poszczególne bajty telegramów są pobierane z tablicy `Telegram()`, która jest tworzona i cały czas odświeżana w pętli głównej procedury obsługi transmisji szeregowej. Użytkownik zmieniając stany poszczególnych bitów słowa sterującego (**Control word**) za pomocą przycisków wyboru, generuje zdarzenia ich wciśnięcia (np. `Option1_Click`). W procedurach tych zmienione słowo sterujące przekazywane jest do pętli transmisji szeregowej a stamtąd przesłane zostaje do falownika.

Opcja **Parameters**

Opcja ta powoduje załadowane okna **Parameters** (Rys. 23 – formatka *Form3*).

W procedurze zdarzenia `Form_load` tej formatki wykonywane są następujące czynności:

- ustawienie wyglądu poszczególnych wierszy i kolumn tabeli parametrów,
- pobranie z pliku *lista.txt* poszczególnych opisów parametrów i umieszczenie ich w odpowiednich wierszach tabeli.

W momencie kiedy użytkownik wciśnie przycisk **Read parameters**, zostaje wykonana procedura `Command1_Click()`. Ustawiane są w niej odpowiednie wskaźniki dla pętli transmisji szeregowej, pozwalające na wykonanie pewnych fragmentów jej kodu i odczytanie wszystkich wartości parametrów z falownika. Dodatkowo zostaje uruchomiona procedura `Timer1`, która cyklicznie sprawdza czy wszystkie wartości zostały przeczytane. Jeśli tak się zdarzy następuje zatrzymanie procedury `Timer1`, przetworzenie wartości parametrów do postaci zrozumiałej dla użytkownika i wyświetlenie ich w tabeli. Przeczytane parametry dostarczane są z pętli transmisji szeregowej do procedury `Timer1` przez zmienną tekstową `S_param`.

Gdy użytkownik będzie dokonywał zmian parametrów, wygeneruje zdarzenie `cmdChangeP_Click()` w oknie **Parameters change** (Rys. 24 – formatka *Form6*). Spowoduje to ustawienie odpowiednich wskaźników dla pętli transmisji szeregowej, mówiących o zapisie parametru i przekazanie do niej wartości nowego parametru poprzez zmienne tekstowe `par_zap_s` i `par_zap_m` (część starsza i młodsza wartości

parametru). Nastąpi także uruchomienie procedury `Timer1` tej formatki, która będzie się wykonywać do momentu otrzymania potwierdzenia zapisu parametru od falownika.

W momencie otrzymania tego potwierdzenia (wskaźnik `par_status` w pętli transmisji szeregowej będzie zawierał pewne wartości), wyświetlany jest odpowiedni komunikat informujący o zapisie. Procedura `Timer1` kończy swoje działanie.

W oknie tym wykorzystywana jest kontrolka Visual Basic'a: *Microsoft FlexGrid control 6.0* zawarta w pliku `VSFLEX32.OCX`, pozwala ona na prezentację danych w postaci arkusza zawierającego wiersze i kolumny.

Opcja Oscilloscope

Po jej wybraniu następuje ładowanie okna **Oscilloscope** (Rys. 27 – formatka *Form5*). Wykonywane są następujące funkcje w zdarzeniu `Form_load` tej formatki:

- ustalane są początkowe wartości parametrów kanałów,
- ustawiany jest wygląd graficzny okna oscyloskopu,
- następuje inicjalizacja podprogramu obsługi przerwania karty zbierania danych (funkcja `SetBoardIntRoutineWin32`)
- oraz załączenie przycisków i paska przewijania opcji sterowania falownikiem.

Głównymi procedurami tego okna są: `C_ReadAnalogInput_Click()`, `c_Dwakanaly_Click()`, oraz `Timer1` i `Timer2`.

Procedura `C_ReadAnalogInput_Click()` obsługuje zdarzenie przycisku **Read one channel**. Służy do czytania wartości próbek z jednego kanału karty. Kiedy użytkownik dokona wciśnięcia tego przycisku, wykonują się zawarte w niej polecenia oraz uruchamia się procedura `Timer1` tego okna, oczekująca na wystąpienie przerwania od karty. Opis procedury `C_ReadAnalogInput_Click()` zawarty jest w rozdziale: 6.3. Sterowanie kartą zbierania danych Kiedy karta zbierze próbki przebiegu, wykonywany jest podprogram obsługi przerwania (`v_InterruptRoutine` – moduł *wykres* programu), w którym są one zapamiętywane w tablicy `l_SaveArray(l_Cpt)` oraz ustawiany jest wskaźnik przerwania. W momencie, kiedy procedura `Timer1` stwierdzi zmianę wartości wskaźnika przerwania, dokonuje następujących operacji:

- obróbki przeczytanych próbek,
- wyświetlenia ich w okienku oscyloskopu,
- wykonania pewnych ustawień karty,
- zakończenia swojego działania.

Działanie procedury `c_Dwakanaly_Click()` jest identyczne, przy czym próbki są zbierane z dwóch kanałów karty jednocześnie. Obsługuje ona zdarzenie przycisku **Read two channels**.

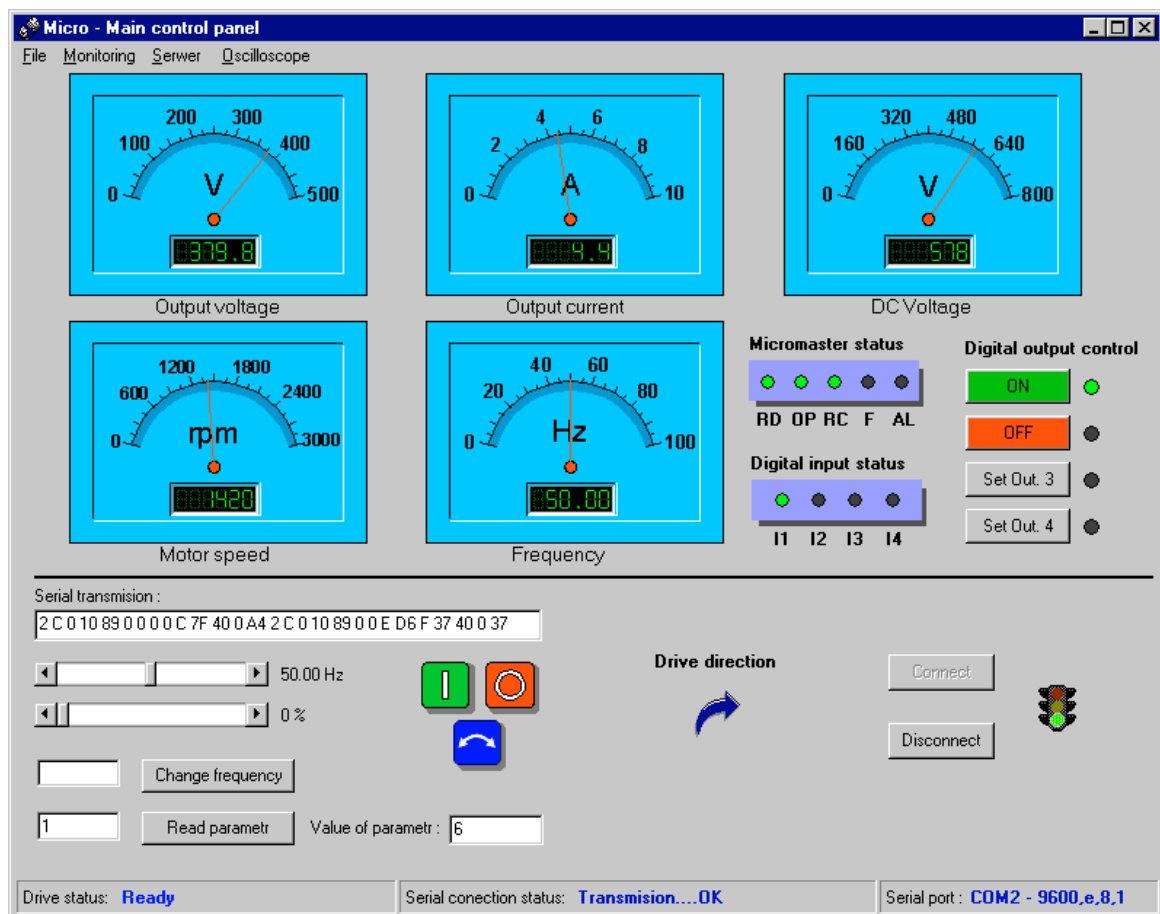
Procedura `Timer2` umożliwia cykliczne zbieranie danych z jednego kanału, w odstępach półsekundowych. Jej działanie opiera się na podobnym cyklu jaki ma miejsce w procedurach `C_ReadAnalogInput_Click()` i `c_Dwakanaly_Click()`.

Jej uruchomienie odbywa się przez procedurę zdarzenia `C_Start_Click()` przycisku **Start acquisition**, a jej zatrzymanie dokonywane jest przez procedurę `C_Stop_Click()` przycisku **Stop acquisition**. Do swojego działania wykorzystuje także procedurę `Timer1`.

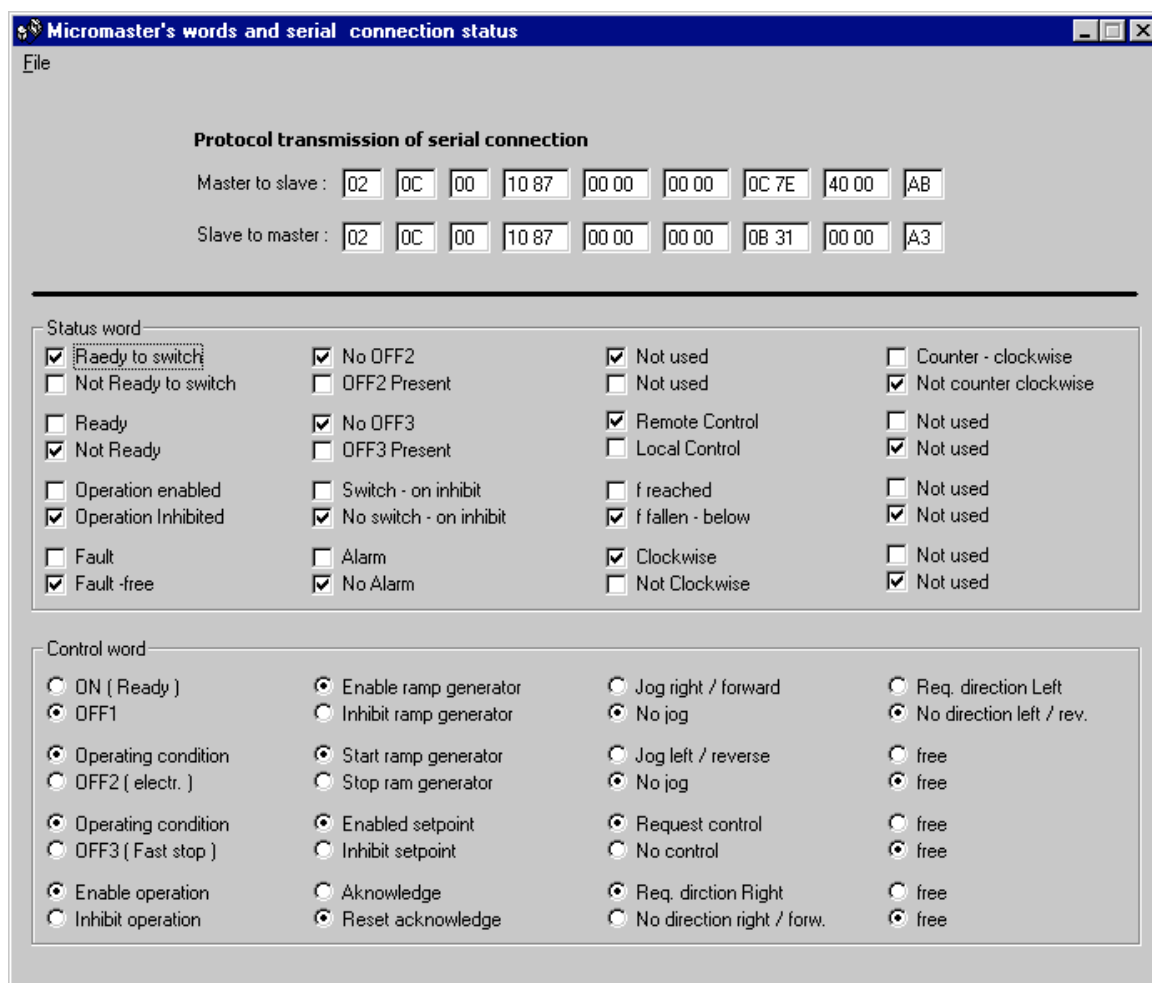
Do wyświetlania zebranych próbek w oknie oscyloskopu wykorzystywana jest kontrolka: *Pinacle-BPS Graph control*, zawarta w pliku GRAPH32.OCX. Umożliwia wyświetlanie danych poprzez przekazywanie do niej pojedynczych wartości próbek, podając jednocześnie jej numer i pozycję na osi X. Pozwala na wyskalowanie osi Y w zakresie określonym przez programistę lub też wykorzystanie automatycznego dopasowania do przekazanych wartości, co zostało wykorzystane w procedurze przycisku **Automatic scale Y ON**.

6.4.4. Przykłady działania programu

Na rysunkach od 28 do 32, przedstawiony został wygląd poszczególnych okien programu, podczas jego pracy.



Rys. 28. Wygląd głównego okna programu podczas pracy falownika i z ustanowioną komunikacją z komputerem (wartość zadana częstotliwości $f = 50$ Hz, obroty silnika w prawo, załączony główny obwód zasilania układu – przycisk ON, możliwe zdalne sterowanie pracą falownika).



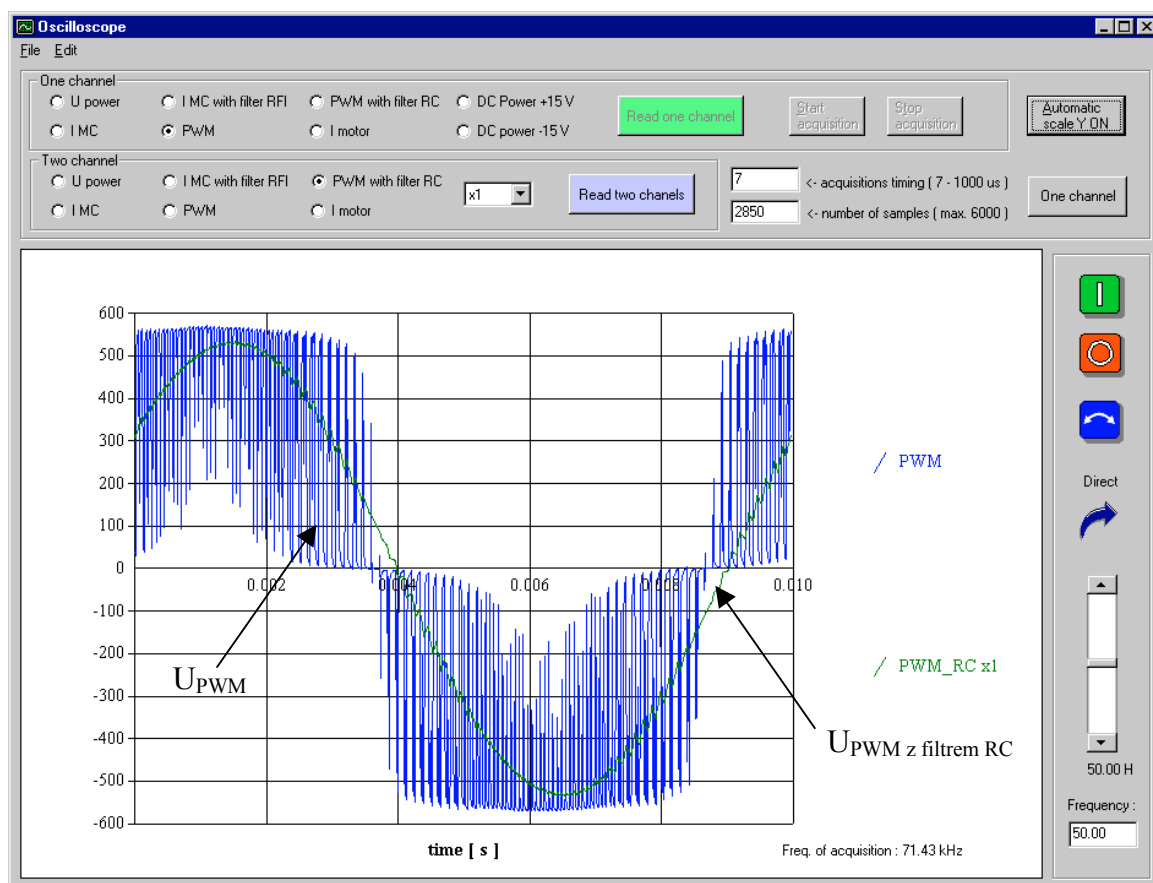
Rys. 29. Wygląd okna podglądu aktualnej transmisji szeregowej między komputerem a falownikiem oraz stanu poszczególnych bitów słowa sterującego i statusu

Nr param.	Description	Value	Dim
rP000	Operation display	0	---
P001	Display mode	6	---
P002	Ramp up time	10	[s]
P003	Ramp down time	10	[s]
P004	Smoothing	0	[s]
P005	Digital frequency setpoint	10	[Hz]
P006	Frequency setpoint source select	0	---
P007	Keypad control	1	---
P009	Parameter protection setting	3	---
P011	Frequency setpoint memory	0	---
P012	Minimum motor frequency	0	[Hz]
P013	Maximum motor frequency	70	[Hz]
P014	Skip frequency 1	0	[Hz]
P015	Auto restart after mains fail	0	---
P016	Start on the fly	0	---
P017	Smoothing type	1	---
P018	Auto restart after fault	0	---
P019	Skip frequency bandwidth	2	[Hz]
P020	Flaying start ramp time	5	[s]
P021	Minimum analog frequency	0	[Hz]
P022	Maximum analog frequency	50	[Hz]
P023	Analogue input type	0	---
P024	Analogue setpoint addition.	0	---
P027	Skip frequency 2	0	[Hz]
P028	Skip frequency 3	0	[Hz]
P029	Skip frequency 4	0	[Hz]
P031	Jog frequency right	5	[Hz]
P032	Jog frequency left	5	[Hz]
P041	Fixed frequency 1	5	[Hz]
P042	Fixed frequency 2	10	[Hz]

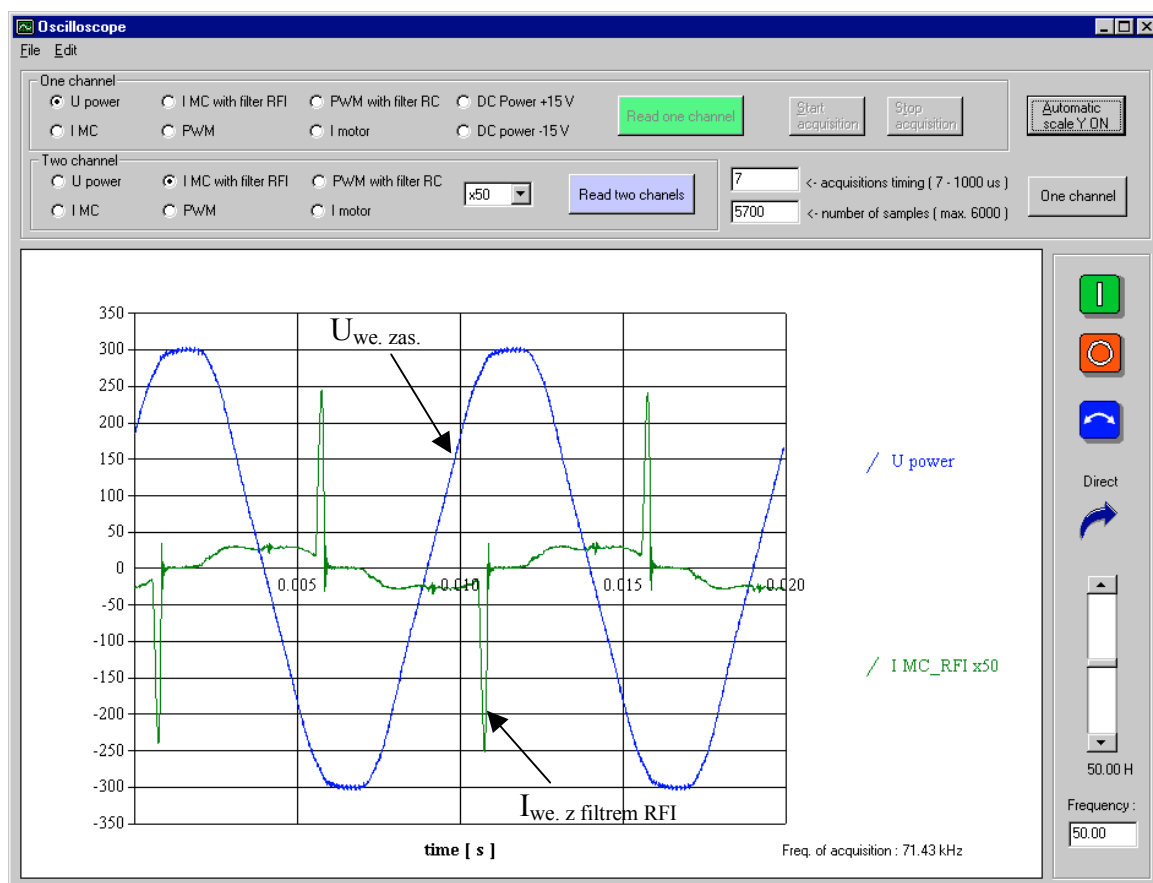
Please Wait reading parameters.....

Read parameters

Rys. 30. Wygląd okna parametrów falownika w czasie ponownego ich odczytu.



Rys. 31. Wygląd okna oscyloskopu podczas jednoczesnego odczytu z dwóch kanałów następujących przebiegów: napięcia międzyprzewodowego wyjściowego falownika (U_{PWM}) i tego samego napięcia z użyciem filtra RC (U_{PWM} z filtrem RC).

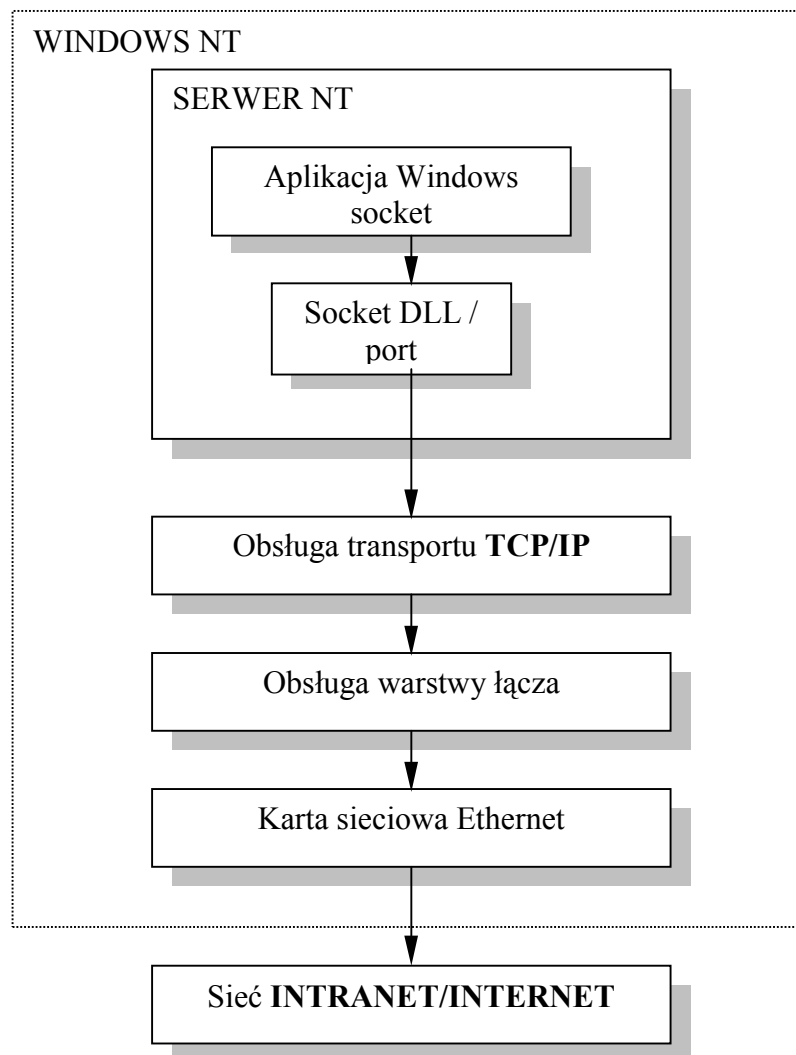


Rys. 32. Wygląd okna oscyloskopu podczas jednoczesnego odczytu z dwóch kanałów następujących przebiegów: napięcia fazowego zasilania falownika ($U_{we. zas.}$) i prądu wejściowego falownika z filtrem RFI ($I_{we. z filtrem RFI}$).

7. Monitoring i sterowanie z wykorzystaniem sieci komputerowej Intranet/Internet

7.1. Wstęp do komunikacji sieciowej

Komunikacja sieciowa odbywa się poprzez połączenie pomiędzy komputerami-hostami: hostem odbiorcą i hostem nadawcą. Aby doszło do połączenia, czy też do przesłania danych, niezbędna jest znajomość adresu identyfikującego hosta odbiorcę. Każdy host (komputer zdalny) mogący odbierać dane musi być przystosowany do świadczenia określonych usług. Każda usługa jest związana z procesem działającym na hoście. Host udostępnia swoje usługi kojarząc proces z numerami portów, przez które odbywa się komunikacja z użytkownikami. Dlatego każda usługa, taka jak WWW, Telnet, Ftp, E-Mail, itp., jest obsługiwana przez inny numer portu, a co za tym idzie inną aplikację dla niej dedykowaną. Umożliwia to jednoczesny dostęp wielu użytkowników do hosta. Również w ramach jednej usługi (jednego portu) możliwa jest obsługa wielu użytkowników. Dzieje się tak dzięki technologii gniazd (socket) dostarczanej przez system operacyjny Windows NT [12]. Architektura gniazd na platformie Windows NT przedstawia rysunek 33.



Rys. 33. Architektura gniazd na platformie Windows NT

Gdy użytkownik komunikuje się z wybranym portem, prowadząca stały nasłuch aplikacja skojarzona z wybraną usługą otwiera nowe gniazdo. Przejmuje ono połączenie zwalniając tym samym gniazdo nasłuchujące dla kolejnych użytkowników. Technologia ta pozwala na dostęp do jednej usługi wielu użytkownikom jednocześnie.

Odnosząc się do sterowania układem automatyki, gniazda umożliwiają sterowanie i monitoring jednocześnie wielu użytkownikom (klientom: "Klient TCP/IP" oraz „Klient WWW”), obsługiwanych przez wspólną aplikację "Micro".

Należy zaznaczyć, iż przy wyborze numeru portu trzeba pamiętać o standardowo zastrzeżonych numerach portów od 0 do 1023 [12]. Są to numery przeznaczone dla usług sieciowych (WWW, Telnet, Ftp, E-Mail).

Aby wymiana informacji pomiędzy hostem-odbiorcą a użytkownikiem była możliwa, przede wszystkim musi być ustanowione połączenie oraz dane muszą być przesyłane w odpowiedni sposób. Opracowany model sieci Internet przewiduje przenoszenie informacji pod kontrolą protokołów na bazie datagramów lub kanałów wirtualnych. Oba sposoby mają inne zalety i inną „cenę”. Datagram UDP oferuje większą prostotę przy ograniczonej niezawodności. Kanał wirtualny oferuje wysoką niezawodność.

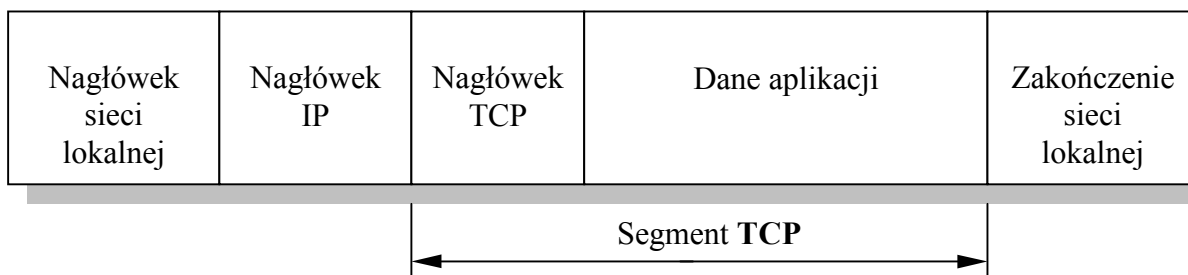
Wybór rozwiązania zależy od wymaganej niezawodności dla przesyłanych danych. Na przykład przy przesyłaniu poczty elektronicznej w przypadku wystąpienia błędu w transmisji można powtórzyć operację przesyłu – wystarczająca jest metoda datagramów. Jeżeli jednak od otrzymywanych danych wymagana jest wysoka niezawodność, powinno się wykorzystać kanał wirtualny.

Gdy mówimy o sterowaniu układem automatyki mamy na myśli przesłanie odpowiedniej sekwencji danych (odpowiedniej postaci słowa sterującego), na które bezpośrednio reaguje urządzenie sterujące - falownik. Niedopuszczalnym jest zatem możliwość odebrania przez falownik zafalszowanej informacji, gdyż to mogłoby doprowadzić do niezamierzonych czynności wykonanych przez falownik bądź utraty nad nim kontroli. Dodatkowym wymogiem jest również utrzymywanie ciągłej transmisji.

Naprzeciw stawianym wymogom wychodzi protokół TCP. W nomenklaturze, ze względu na szerokie zastosowanie w sieci INTERNET nazywany TCP/IP (*Transfer Communication Protocol / Internet Protocol*) [11]. TCP/IP spełnia sześć istotnych funkcji :

- Podstawowy transfer danych – przepływ danych w dwóch kierunkach.
- Niezawodność – czas połączenia i kontrola pakietów.
- Kontrola przepływu – rozmiar i kolejność danych.
- Multipleksowanie – obsługa wielu usług jednocześnie.
- Połączenia – ustanowienie i podtrzymanie.
- Nadrzędność i bezpieczeństwo – przydzielanie atrybutów.

Ponieważ dane przesyłane w sieci - ze względu na jej przepustowość - przesyła się partiami, stworzono model definiujący postać przesyłanych danych tzw. ramkę transmisji internetowej zilustrowaną na rysunku 34.



Rys. 34. Ramka transmisji internetowej

Klient komunikując się z hostem wysyła dane w postaci ramki. Ramka zawiera informacje niezbędne do połączenia z hostem czyli adres hosta, numer portu i dane. Na podstawie adresu IP i numeru portu kojarzone jest nowe gniazdo z wybranym portem. Teraz kontrolą połączenia i przesyłu danych zajmuje się protokół TCP/IP wykorzystując swoje funkcje. Ponadto ustanawia połączenie między klientem a serwerem oraz pilnuje jego poprawności. Jednak największą zaletą protokołu determinującą jego wykorzystanie w sterowaniu jest kontrola przepływu danych.

Dane, podzielone na oktety, po przesłaniu do serwera są weryfikowane i dopiero po potwierdzeniu ich zgodności przez nadawcę, przyjmowane przez serwer. Nie występuje zatem sytuacja aby do serwera dotarły inne dane niż te, które wysłał klient. Ponieważ TCP/IP umożliwia jednoczesny przepływ danych w obu kierunkach, również dane przesyłane od serwera do klienta podlegają weryfikacji i to w znacznie krótszym czasie niż dla transmisji jednokierunkowej, jak to ma miejsce w metodzie datagramów. W przypadku wystąpienia błędu w danych, jedna ze stron zgłaszająca błąd żąda ponownego przesłania porcji danych.

Przychodzące pakiety są również kolejgowane co umożliwia czytanie ich w odpowiednim porządku. Ważną cechą protokołu TCP/IP jest wysyłanie takiej partii danych jaką możliwą ilość przyjęcia zadeklaruje, czy też jest w stanie przyjąć, jedna ze stron połączenia. Eliminuje to „zapychanie” portu, zarówno po stronie serwera jak i klienta. TCP/IP umożliwia również ustawianie poziomu bezpieczeństwa przesyłanych danych.

Z uwagi na niezawodność oraz obsługę przesyłu danych, do procesu sterowania falownikiem został wykorzystany protokół TCP/IP.

7.2. Aplikacje „Klient” ↔ „Serwer”

7.2.1. Wstęp

W istniejących sieciach komputerowych wymiana informacji między komputerami odbywa się w systemie „klient ↔ serwer”. Jest to system w którym jeden z komputerów pełni rolę serwera, a drugi, lub wiele innych, jest klientem. Zadaniem serwera jest obsługiwanie żądań klienta i wysyłanie do niego odpowiedzi. Lecz jeśli klient żąda informacji od serwera, to serwer może, ale nie musi na nie odpowiedzieć.

Klient jest więc w pełni uzależniony od poczynań serwera. W systemie tym mogą się komunikować ze sobą komputery zawierające różne systemy operacyjne (np. komputer pracujący pod systemem Windows może komunikować się z maszyną UNIX’ową). Jest to możliwe dzięki temu, że używają one do komunikacji, tych samych protokołów transmisji. Ze względu na rodzaj protokołu: TCP/IP czy UDP (protokoły te zostały opisane we wstępie rozdziału), rozróżnia się serwery połączeniowe i bezpołączeniowe.

Serwer bezpołączeniowy (wykorzystujący protokół UDP) to taki, który otrzymując wiadomość od klienta wykonuje odpowiednie zadania i odpowiada klientowi. Umożliwia on bardzo szybką wymianę informacji, ale kosztem obniżenia kontroli poprawności wysyłanych wiadomości.

Serwer połączeniowy (wykorzystujący protokół TCP/IP) to taki, który otrzymując żądanie od klienta, musi je zaakceptować i ustanowić z nim oddzielne połączenie w celu wymiany informacji. Następnie po wykonaniu poszczególnych zadań wysyła odpowiedź i jeśli klient nie ma żadnych kolejnych żądań może zamknąć połączenie. Połączenie może być także zamknięte od strony klienta. Wykorzystywana jest tu rozbudowana kontrola poprawności przesyłu informacji.

W lokalnych sieciach komputerowych (sieć Intranet) „klientów” może być od kilku do kilkuset, natomiast serwerów od kilku do kilkudziesięciu. W sieciach rozbudowanych (Internet), klientów i serwerów może być bardzo dużo, gdyż jest to sieć globalna (światowa) zawierająca połączone między sobą sieci lokalne. Wykorzystanie możliwości sieci komputerowej Intranet/Internet w powstałym systemie monitoringu i sterowania zdalnego, wymagało rozbudowania istniejącego oprogramowania i stworzenia nowego. Zadania jakie powinny być realizowane przez oprogramowanie są następujące:

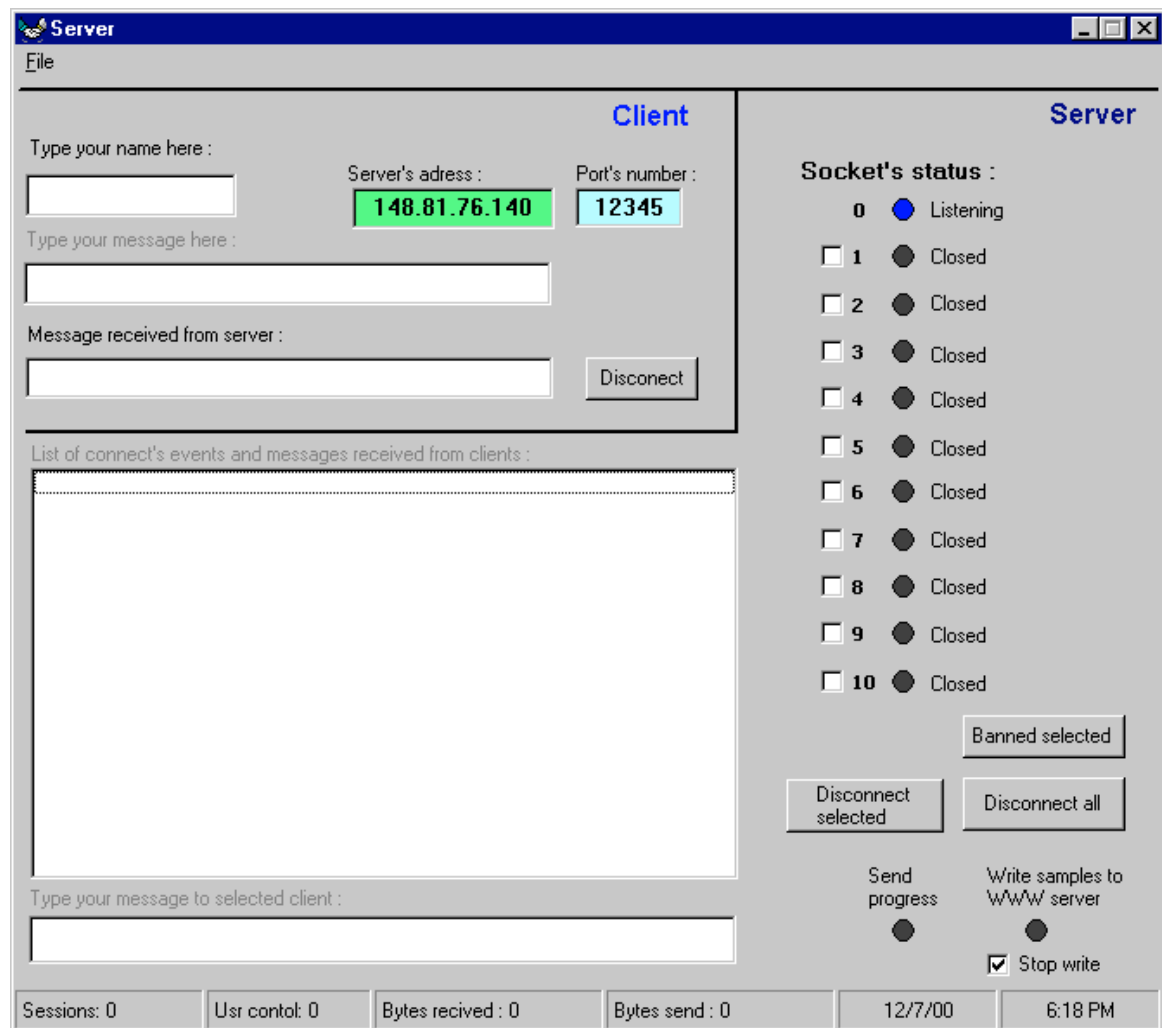
- obsługa protokołu transmisji TCP/IP używanego w połączeniach sieci komputerowej,
- wymiany informacji między programami,
- możliwość sterowania i monitorowania pracy falownikowego układu napędu z dowolnego miejsca w sieci komputerowej.

W tym celu rozbudowano aplikację „Micro” o dodatkową opcję „Server”, oraz napisano nowe oprogramowanie o nazwie „Client TCP/IP”.

7.2.2. Opcja Server programu „Micro”

Opcja ta jest wybierana z menu programu „Micro” (rozdział 6.4.2.) i jest jego integralną częścią.

Okno Server'a



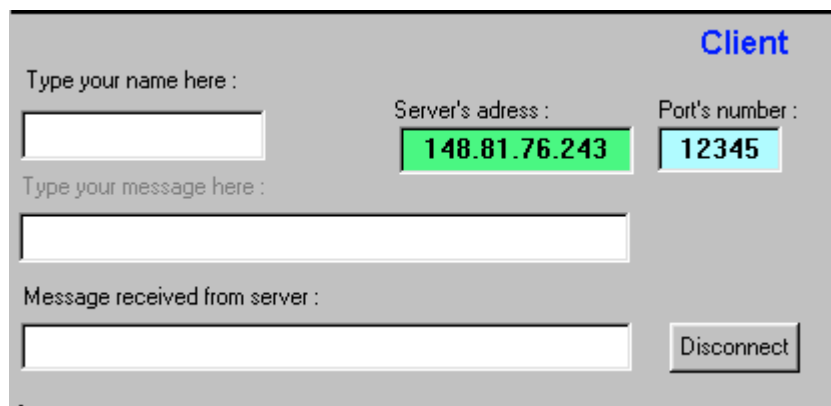
Rys. 35. Wygląd okna serwera

Opcja **Server** jest głównym pomostem między komputerem lokalnym, nadzorującym pracę falownikowego układu napędu a komputerami klientów, na których uruchomione jest odpowiednie oprogramowanie. Umożliwia on obsługę do dziesięciu klientów jednocześnie. Wygląd serwera przedstawia rysunek 35.

Okno podzielone jest na dwie części zawierające blok **Client**'a i blok **Server**'a.

Blok **Client**'a (Rys. 36) jest przykładem programu klienta, który umożliwia sprawdzenie następujących funkcji:

- ❑ komunikacji sieciowej opartej na protokole TCP/IP, w danym komputerze na którym jest uruchomiony serwer,
- ❑ możliwości połączenia z serwerem,
- ❑ poprawności jego działania.



Rys. 36 Wygląd części klienta

W polu tekstowym **Server's adress**, wyświetlany jest adres IP komputera na którym został uruchomiony serwer. Pole te umożliwia także wpisanie innego adresu IP komputera z którym chcemy się połączyć.

Pole **Port's number**, służy do wpisywania numeru portu na którym chcemy uzyskać połączenie klienta z danym komputerem. Przy uruchomieniu okna, w polu tym wyświetlany jest numer portu na którym działa serwer.

Aby dokonać połączenia za pomocą tego klienta trzeba wpisać swoje imię w polu **Type your name here** i zatwierdzić klawiszem „Enter”.

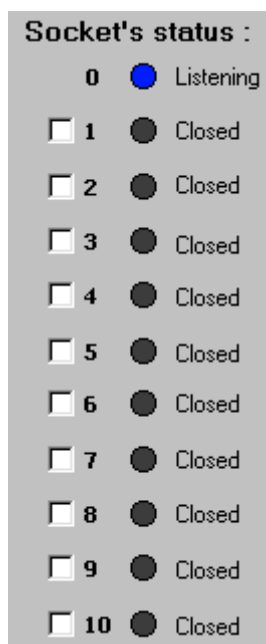
W polu **Type your message here**, jest możliwe wpisywanie wiadomości tekstowych przekazywanych do serwera.

Pole **Message received from server** bloku klienta, pokazuje aktualne wiadomości otrzymane od serwera.

Użycie przycisku **Disconnect** powoduje odłączenie klienta od serwera.

Blok Server'a

W części **Socket's status** (Rys. 37.), przy pomocy kontrolki (lampek zmieniających swój kolor) i umieszczonej obok nich etykiety, przedstawiony jest stan poszczególnych gniazd. Wskazują one ilość użytkowników, podłączonych w danej chwili do serwera. Pola wyboru od jednego do dziesięciu umożliwiają zaznaczenie dowolnego klienta, w celu np. jego odłączenia.



Rys. 37. Podgląd stanu gniazd

Blok serwera zawiera następujące przyciski:

- **Banned selected** – powoduje odłączenie wybranego klienta i zablokowanie jego numeru IP. Spowoduje to uniemożliwienie ponownego połączenia z serwerem.
- **Disconnect selected** – jego użycie spowoduje odłączenie wybranego klienta od serwera.
- **Disconnect all** – odłącza od serwera wszystkich połączonych w chwili klientów od serwera.

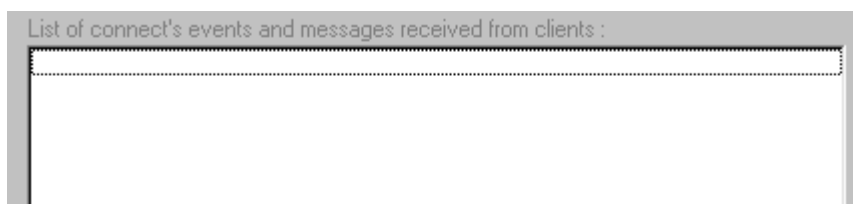
Banned selected

Disconnect selected

Disconnect all

danej

Wszystkie zdarzenia połączeń, żądań oraz wiadomości klientów, wyświetlane są w oknie: **List of connect's events and messages received from clients.**(Rys. 38).



Rys. 38. Wygląd okienka zdarzeń serwera

Pole **Type your message to selected client**, umożliwia przesyłanie wiadomości tekstowych do poszczególnych klientów, wybieranych przez zaznaczenie pól w części **Socket's status** serwera.



Kontrolka **Send progress** umieszczona na dole okna serwera sygnalizuje proces wysyłania wiadomości. W momencie połączenia przynajmniej jednego klienta, rozpoczyna ona swoją pracę.



Znajdująca się obok niej kontrolka **Write samples to WWW server**, sygnalizuje zapis próbek na dysk twardy. Ze wszystkich wejść analogowych karty następuje zbieranie przebiegów wartości chwilowych. Próbkę te są następnie wykorzystywane przez aplikację „Client WWW”. Ich zapis jest prowadzony w 5-cio sekundowych odstępach czasu. Opcja wyboru **Stop write** tej kontrolki pozwala na zatrzymanie lub zezwolenie zapisu próbek. Przy uruchamianiu okna „Server”, opcja ta jest zaznaczona i proces zapisu jest wstrzymany. Aby ponowny zapis był możliwy, opcja musi zostać odznaczona.



Pasek statusu serwera zawiera podgląd następujących informacji:

Sessions: 0	Usr control: 0	Bytes received : 0	Bytes send : 0	00-12-08	13:11
-------------	----------------	--------------------	----------------	----------	-------

- **Session** – pokazuje liczbę otwartych sesji (aktualna liczba połączonych klientów),
- **Usr control** – liczba połączonych użytkowników mających dostęp do opcji sterowania,
- **Bytes recived** – ilość bajtów otrzymanych od klienta,
- **Bytes send** – ilość bajtów wysyłanych przez serwer do klienta.

Dodatkowo na pasku statusu, wyświetlany jest aktualny czas i data.

7.2.3. Opis i algorytm działania Server'a

Po wybraniu opcji „Server” w programie „Micro”, następuje załadowanie jego okna (Rys. 35, formatka *Form4* programu) i uruchomienie procedury *Form_load*, w której wykonywane są następujące czynności:

- ustawiany jest numer portu na którym ma pracować serwer (nr 12345) i rodzaj obsługiwane protokołu transmisji (TCP/IP),
- ładowane jest gniazdo (nr 0) i zostaje ustawione w tryb nasłuchu (czeka na żądanie połączenie od klienta),
- ładowane jest pozostałych 10 gniazd,
- uruchamiana jest procedura *tmrTimer_Timer*,
- ustawiany jest podprogram obsługi przerwań karty zbierania danych.

Procedura `tmrTimer_Timer` programu, wykonywana jest w odstępach jednosekundowych i powoduje wykonanie procedury `UpdateStatus`. Procedura ta odpowiedzialna jest za sprawdzenie i wyświetlanie statusu gniazd w części **Socket's status** okna oraz wyświetlania na pasku statusu okna, liczbę aktualnie podłączonych użytkowników (ilość sesji – **Session**) i liczby użytkowników posiadających prawa do sterowania układem (**Usr control**).

Po załadowaniu okna, serwer oczekuje na połączenie.

W momencie kiedy na gniazdo nasłuchujące przychodzi żądanie połączenia, uruchamiana jest procedura `wsk_ConnectionRequest`. Procedura ta najpierw sprawdza na którym gnieździe zostało ustanowione połączenie, następnie jeśli adres IP użytkownika nie jest zablokowany akceptuje je. Jeśli było to pierwsze połączenie na serwerze, to uruchamiana jest procedura `tmrSend_Timer`, która wysyła co pół sekundy dane do klienta, lub do innych klientów połączonych w późniejszym czasie. Procedura `tmrSend_Timer`, działa do momentu, kiedy nie będzie już żadnego aktywnego połączenia.

Dane wysyłane do klientów, pobierane są z różnych części programu „Micro”. Występują w postaci jednego słowa i zawierają następujące informacje:

- wartości napięć, prądu, prędkości i częstotliwości pobieranych z pętli transmisji szeregowej,
- ilość aktywnych sesji,
- ilość użytkowników sterujących,
- słowa telegramów protokołu USS,
- stany wejść/wyjść cyfrowych karty zbierania danych.

Poszczególne wymienione części tego słowa oddzielone są znakiem „#”, ułatwiającym późniejszą ich obróbkę w programach klientów.

W momencie kiedy serwer odbiera wiadomość wysłaną od klienta, uruchamiana jest procedura `wsk_DataArrival`. Procedura ta zawiera kod obsługi poszczególnych usług serwera. Jeśli wiadomość odebrana ma odpowiednią składnię, odpowiadającą danej usłudze, serwer wykonuje przyporządkowany do niej program. Wszystkie wiadomości odebrane przez serwer, oprócz słów sterujących pracą falownika (telegramy protokołu USS), wyświetlane są w odpowiednim oknie. Pozwala to na przegląd zdarzeń jakie miały miejsce w układzie. Po wykonaniu danej usługi, w zależności od jej rodzaju, serwer przesyła klientowi odpowiednie dane.

Usługi serwera:

- sterowanie pracą falownika,
- czytanie próbek przebiegów wartości chwilowych z karty DAQ,
- odczyt i zmiana wartości parametrów falownika,
- załączenie i rozłączenie obwodu zasilania falownika,
- załączenie i rozłączenie transmisji szeregowej między komputerem a falownikiem.

Przychodzące do serwera słowa sterujące pracą falownika są sprawdzane i analizowane pod kątem ich poprawności. Do pewnych usług oferowanych przez serwer ma dostęp tylko klient posiadający uprawnienia administratora.

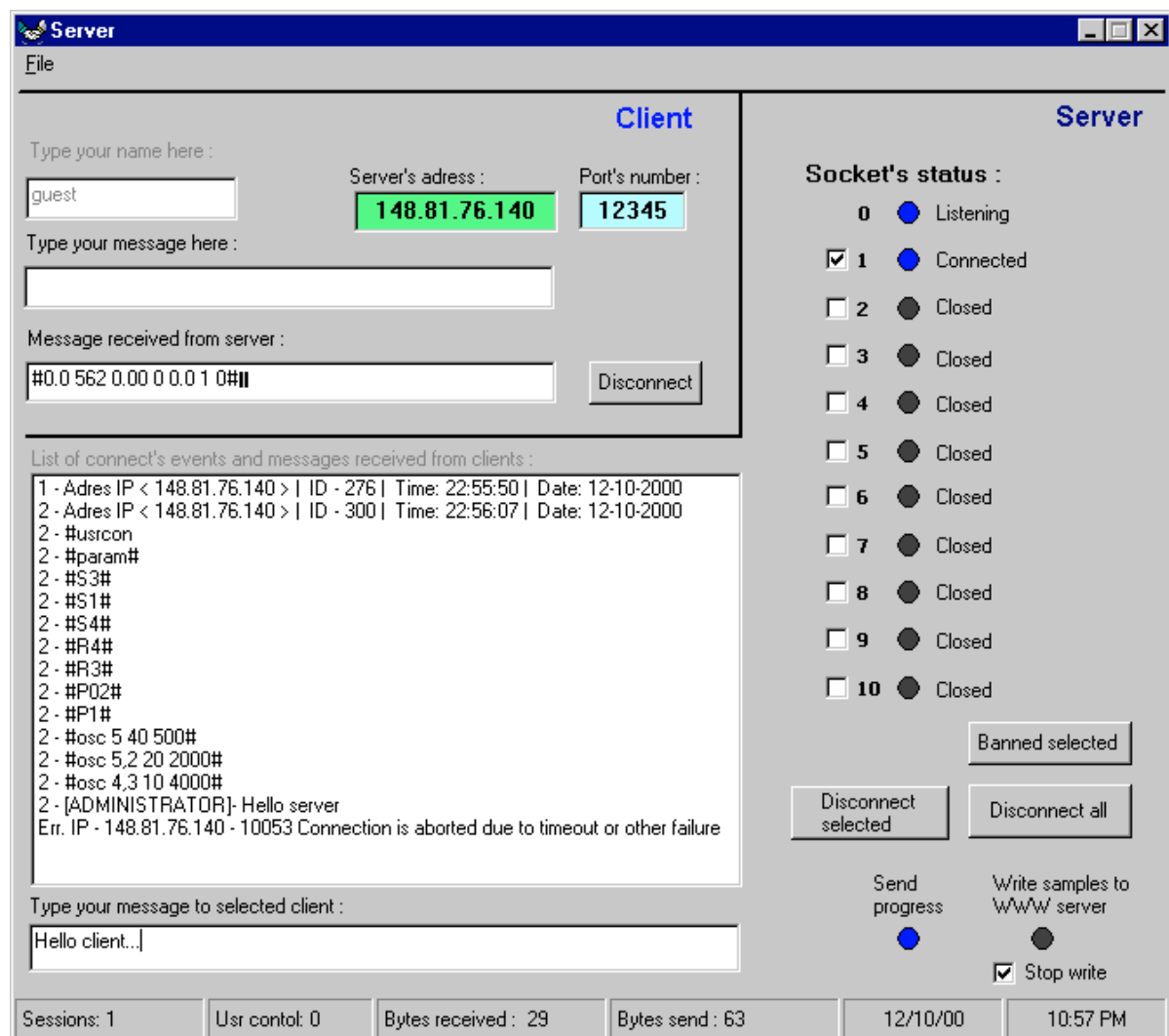
W sytuacjach kiedy serwer otrzymuje kilka zleceń naraz, zostają one zapamiętane w tablicy TabFIFO, w kolejności ich wystąpienia. Po wykonaniu określonej usługi wykonywana jest procedura KolejkaFIFO, w której sprawdzana jest ta tablica. Jeśli występują w niej zlecenia, to zostają one wykonane w odpowiedniej kolejności.

Kiedy użytkownik odłącza się od serwera wygenerowane jest zdarzenie `wsk_close`. Procedura ta powoduje zamknięcie gniazda obsługującego dane połączenie. Jeśli rozłączenia dokonał użytkownik, mający uprawnienia pozwalające na sterowanie falownikiem, następuje zatrzymanie jego pracy poprzez wysłanie odpowiedniego słowa sterującego. W momencie kiedy nastąpi błąd lub przerwanie w komunikacji sieciowej, sytuacja jest podobna.

W czasie działania serwera wyłączona jest opcja **Oscilloscope** programu „Micro”. Spowodowane jest to tym, że serwer przejmuje obsługę nad kartą zbierania danych.

Podczas działania serwera nadal możliwe jest sterowanie za pomocą głównego panelu sterującego (**Micro - Main control panel**).

Rysunek 39 przedstawia wygląd okna działającego serwera.



Rys. 39. Wygląd okna serwera podczas obsługi połączenia dwóch klientów

7.2.4. Aplikacja „Client TCP/IP”

Aplikacja „Client TCP/IP” została napisana jako program działający pod systemem Windows 9x, NT i Windows 2000. Do jej napisania wykorzystano tak jak w programie „Micro”, język Visual Basic.

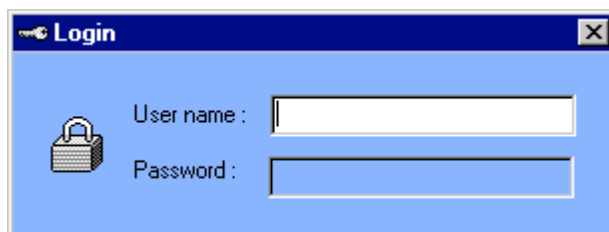
Zadaniem jakie powinien realizować program klienta było przeniesienie i wykorzystanie we własnej budowie funkcji i właściwości jakie posiadał wcześniej napisany program „Micro”. Monitoring i sterowanie, miało odbywać się z wykorzystaniem sieci komputerowej Intranet/Internet. Dodatkowo program klienta musiał mieć zapewniony podstawowy system zabezpieczeń, aby osoby niepowołane nie mogły korzystać z opcji sterowania układem.

Wygląd okien „Client’a TCP/IP” jest bardzo podobny do programu „Micro”. Poniżej zostaną przedstawione i opisane różnice w wyglądzie programu klienta. Opis pozostałych okien ze względu na ich taką samą budowę nie został umieszczony.

Okno logowania – Login

Okienko to pojawia się na początku ładowania programu. Jego wygląd przedstawia rysunek 40. Za pomocą tego okna ograniczono dostęp do programu osobom niepowołanym. W tym celu dokonano podziału na następujących użytkowników:

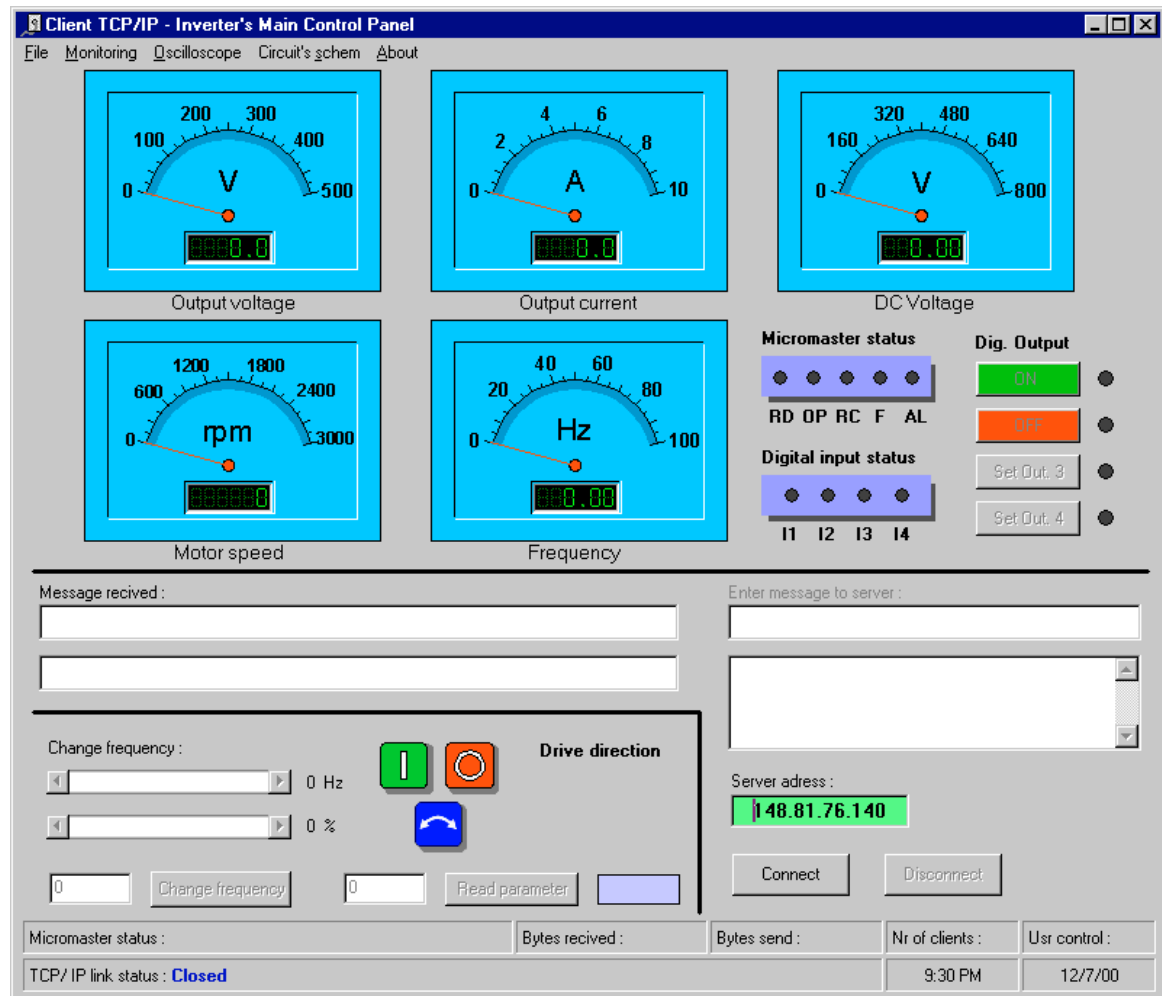
- administrator – użytkownik posiadający uprawnienia do używania wszystkich opcji programu,
- guest – (gość), jest to użytkownik posiadający następujące uprawnienia:
 - możliwość obserwowania pracy układu,
 - ograniczonego korzystania z opcji oscyloskopu (tylko podgląd napięcia zasilania falownika),
 - wysyłania i odbierania krótkich wiadomości tekstowych do serwera
- użytkownik utworzony przez administratora:
 - użytkownik posiadający status użytkownika sterującego układem. Posiada on możliwość korzystania ze wszystkich opcji programu oprócz opcji sterowania wyjściami cyfrowymi (załączanie i odłączanie obwodu zasilania falownika – uprawnienia administratora)
 - użytkownik nie posiadający takiego statusu, tzn. nie mogącego sterować pracą falownika. Użytkownik ten może natomiast korzystać z takich opcji jak: podgląd pracy falownika, czytania jego parametrów, korzystania z opcji oscyloskopu.



Rys. 40. Wygląd okna logowania

Okno główne programu, panel sterujący – Inverter's Main Control Panel

Wygląd okna przedstawia Rys. 41.



Rys. 41. Wygląd głównego okna programu – panelu sterującego

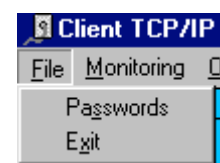
W programie klienta doszły dwie opcje menu:

- **Circuit's schem** – schemat blokowy falownikowego układu napędu. Zawiera także budowę poszczególnych bloków układu.
- **About** – okienko zawierające informacje o programie.



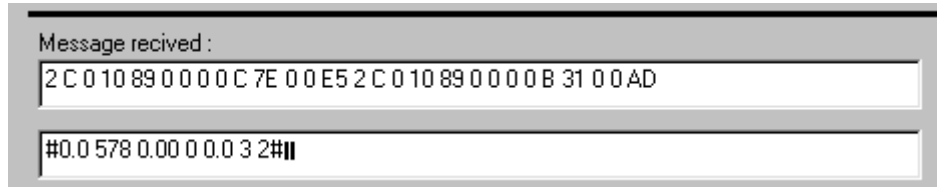
Oprócz tego w menu **File** nie ma opcji "Serial setup", gdyż komunikacja odbywa się z wykorzystaniem sieci komputerowej a nie łącza szeregowego. Zamiast tego występuje nowa opcja **Passwords**. Umożliwia ona administratorowi wykonywanie następujących opcji:

- dodawania nowych użytkowników,
- nadawania uprawnień do sterowania układem,
- usuwania i edycji istniejących użytkowników.

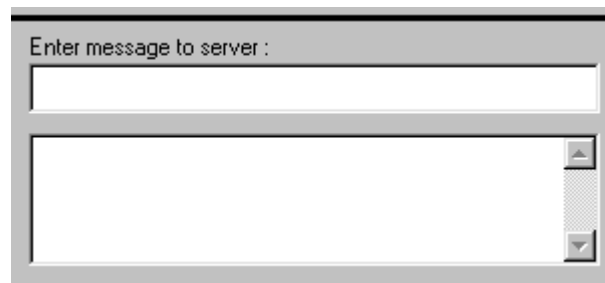


Okno panelu sterującego rozbudowane jest dodatkowo o pola:

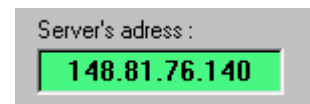
- **Message recived** – zawiera dwa pola tekstowe w których wyświetlane są wiadomości przesyłane od serwera. Górne pole służy do podglądu telegramów protokołu USS komunikacji między komputerem a falownikiem, zaś dolne przedstawia dane wysyłane przez serwer.



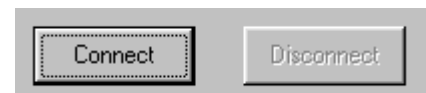
- **Enter message to server** – pole to umożliwia pisanie i wysyłanie krótkich wiadomości tekstowych do serwera. Pole znajdujące się poniżej pola: Enter message to server, służy do odbierania wiadomości tekstowych wysyłanych przez serwer.



- **Server's adress** – pole to służy do wpisywania numeru IP serwera z którym chcemy się połączyć. Po uruchomieniu programu pole zawiera standardowy adres IP serwera, określonego w kodzie programu.



Przyciski **Connect** i **Disconnect**, służą do połączenia i rozłączenia



Pasek statusu umiejscowiony na dole okna zawiera podgląd następujących informacji:

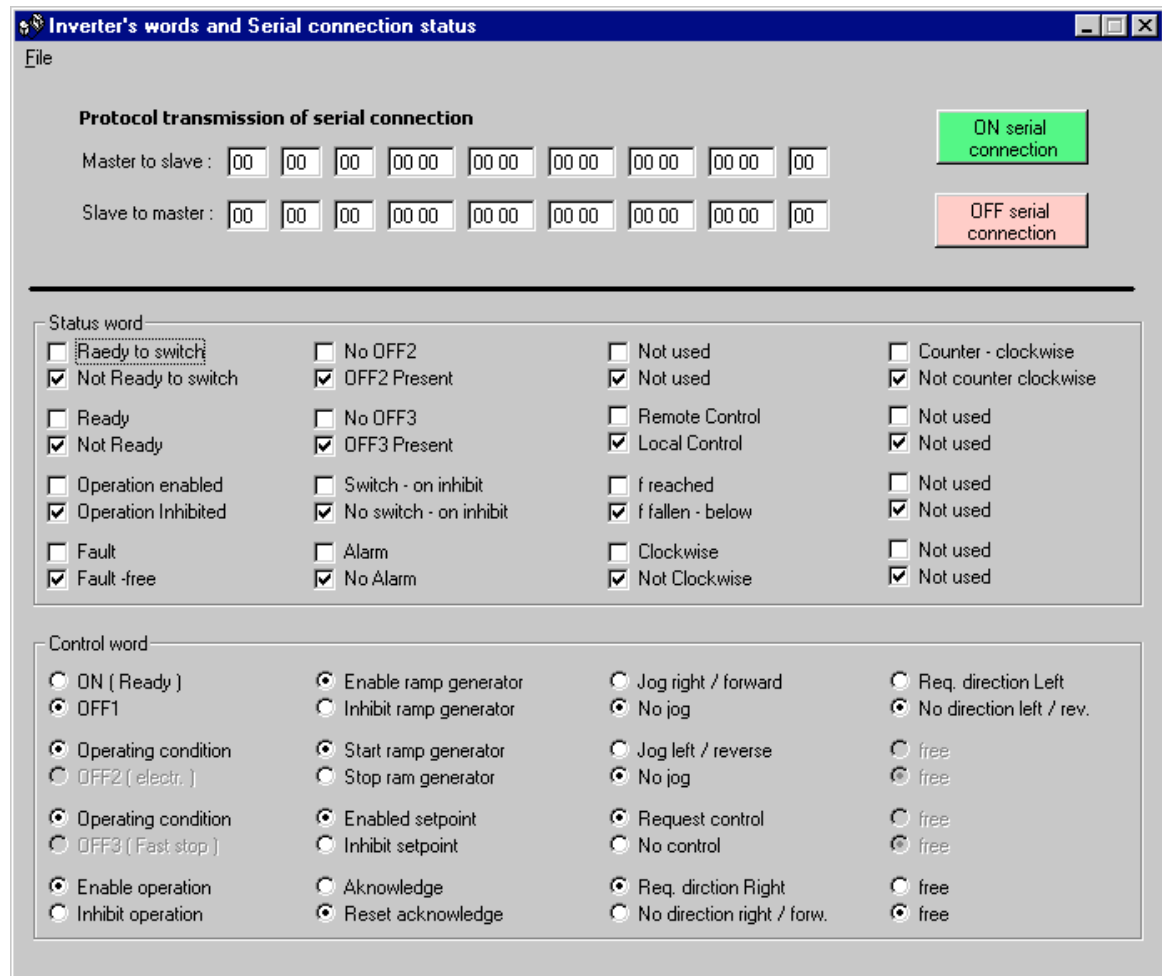
Inverter link status :	Bytes received :	Bytes send :	Nr of clients :	Usr control :
TCP/IP link status : Closed			13.48	00-12-08

- **Inverter link status** – status falownika wskazujący na:
 - połączony z komputerem sterującym (**Inverter connected to control computer**),
 - odłączony od komputera sterującego (**Disconnect from control computer**),
- **TCP/IP link status** – status połączenia sieciowego sygnalizujący następujące stany:
 - połączenie nawiązane (**Connected**),
 - połączenie zamknięte (**Closed**),
 - następuje zamknięcie połączenia przez serwer (**Closing...by Server**),
 - próba nawiązania połączenia (**Connecting...**),
 - wystąpienie błędu (**Error**), dodatkowo jest wyświetlany opis tego błędu
- **Bytes send** - ilość bajtów wysłanych przez klienta,
- **Bytes recived** – ilość bajtów otrzymanych od serwera,

- **Nr of clients** – wyświetla liczbę wszystkich klientów podłączonych do serwera,
- **Usr control** - wyświetla liczbę podłączonych użytkowników mających uprawnienia do sterowania układem.

Okno podglądu połączenia szeregowego i słów protokołu USS falownika – Inverter's words and serial connection status

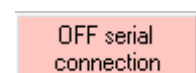
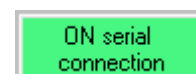
Wygląd okna przedstawia rysunek 42.



Rys. 42. Wygląd okna monitoringu połączenia szeregowego i telegramów falownika.

W oknie tym dodano dwa przyciski o następującym znaczeniu:

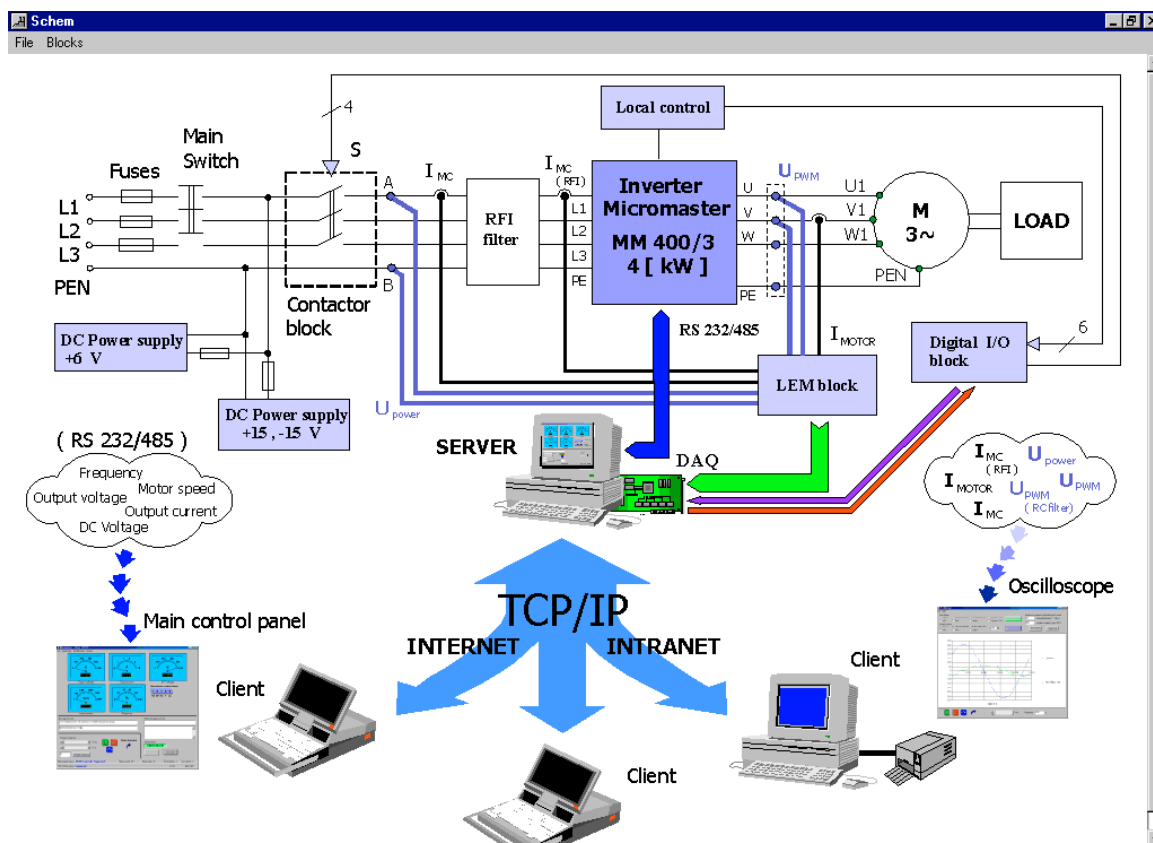
- **ON serial connection** – powoduje zdalne załączenie transmisji szeregowej między komputerem a falownikiem.
- **OFF serial connection** – powoduje zdalne wyłączenie transmisji szeregowej między komputerem a falownikiem.



Reszta opcji pozostaje bez zmian w stosunku do programu „Micro”.

Okno zawierające schemat układu – opcja Circuit's schem

Okno to przedstawia schemat blokowy falownikowego układu napędu, z uwzględnieniem wykorzystania sieci komputerowej Intranet/Internet oraz umożliwia użytkownikowi przegląd całej topologii układu. Wygląd tego okna przedstawia rysunek 43.



Rys. 43. Wygląd okna schematu blokowego falownikowego układu napędu i jego systemu monitoringu i sterowania.

Okno to zawiera także opcję **Blocks**, w której są zawarte schematy podstawowych bloków układu.

7.2.5. Opis i algorytm działania programu „Client TCP/IP”

Ze względu na podobną do programu „Micro”, strukturę działania poszczególnych okien programu klienta, omówiony zostanie algorytm działania jego głównej części odpowiedzialnej za komunikację w sieci komputerowej. Część ta została zawarta w oknie **Inverter's Main Control Panel**.

Po uruchomieniu programu ładowane jest okienko logowania – **Login** (Rys. 40), w którym program prosi o podanie nazwy użytkownika i jego hasła. Ładowane jest także główne okno programu **Inverter's Main Control Panel** (Rys. 41). W zdarzeniu `Form_load` okna głównego (formatka *main*), wykonywane jest ustawienie protokołu komunikacji sieciowej na TCP/IP oraz ustawienie portu na którym klient będzie się łączył z serwerem (nr 12345). Po udanym zalogowaniu, nazwa użytkownika jest zapamiętywana i następuje zamknięcie okna logowania oraz uruchomienie procedury `Timer1_Timer` okna głównego. Procedura ta działa cały czas gdy połączenie sieciowe jest aktywne i odpowiedzialna jest za następujące funkcje programu:

- wyświetlanie statusu połączenia sieciowego (**TCP/IP link status**)
- wyświetlania statusu falownika (**Inverter link status**), określającego stan jego połączenie z lokalnym komputerem sterującym
- jeśli występuje połączenie falownika z lokalnym komputerem sterującym, uaktywnia ona poszczególne opcje:
 - sterowania układu,
 - jego stanu pracy (**Micromaster status**),
 - kierunku obrotów silnika.
- jeśli następuje zakończenie komunikacji między falownikiem a komputerem lokalnym, procedura powoduje wyłączenie opcji sterowania

W momencie kiedy użytkownik użyje przycisku **Connect**, wykonywana jest procedura `cmdConnect_Click`. Następuje połączenie z serwerem o numerze IP zawartym w polu **Sever's adress** i oczekiwanie na odebranie pierwszych danych.

Kiedy to nastąpi, w zależności jaki użytkownik załogował się w programie klienta, załączane są odpowiednie opcje sterowania. Jeśli falownik jest połączony z lokalnym komputerem i trwa transmisja szeregową, załączane są mierniki na panelu sterowania klienta. Dane przyjmowane od serwera są przetwarzane i wyświetlane na poszczególnych miernikach.

W momencie kiedy użytkownik użyje przycisku **Disconnect**, następuje uruchomienie procedury `cmdDisconnect_Click`, która zamyka połączenie i wyłącza poszczególne opcje panelu operatorskiego.

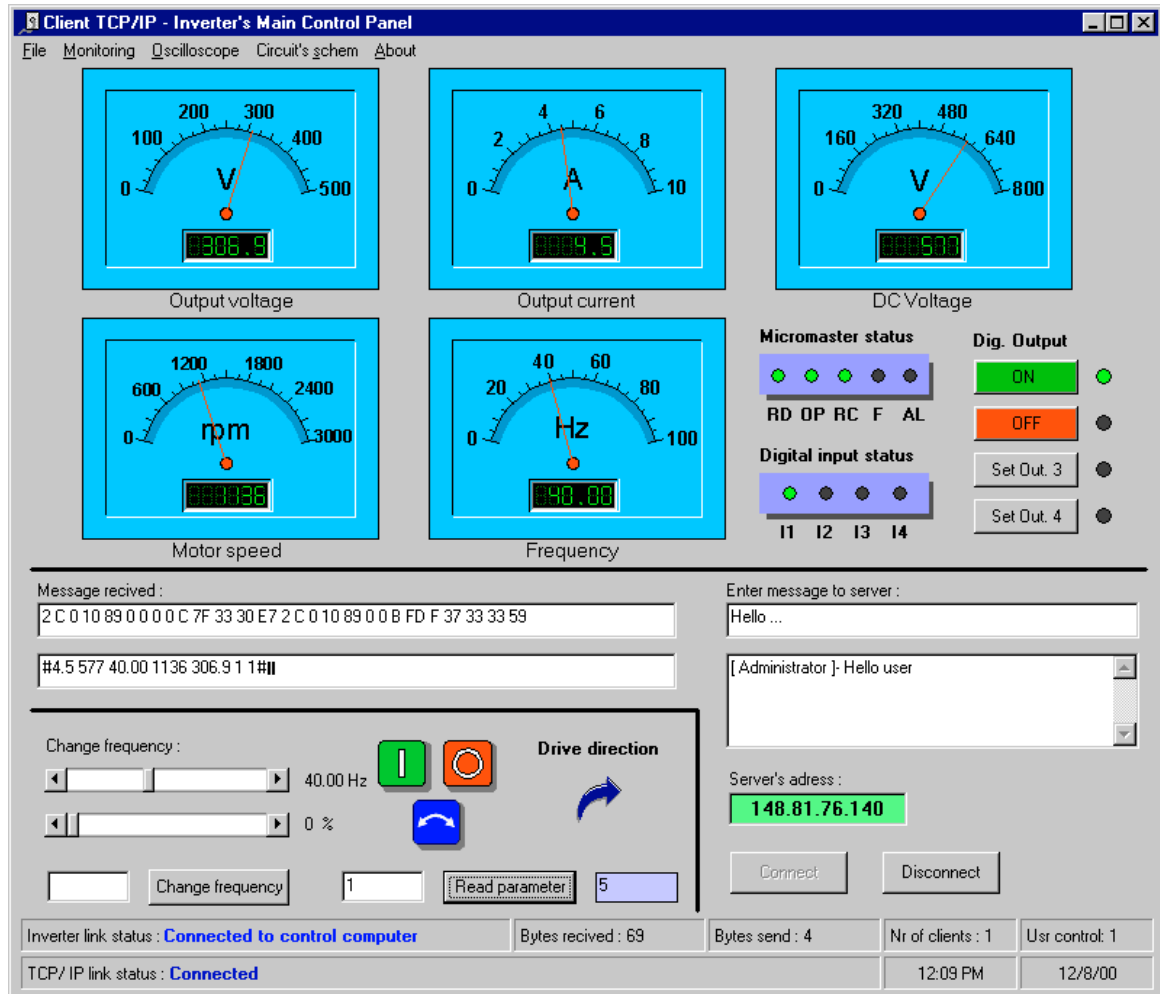
W opcjach programu klienta, wykonanie polecenia związanego z takimi czynnościami jak: odczyt przebiegów w oknie **Oscilloscope**, czytanie parametrów falownika (okno **Parameters**), sterowanie pracą falownika z głównego okna programu, powoduje wysłanie do serwera odpowiedniego słowa danych. Serwer wykonuje poszczególne operacje związane z żądaniem klienta i wysyła odpowiednie dane z powrotem do programu klienta.

Procedurą programu realizującą odbiór i przetwarzanie otrzymanych danych od serwera jest procedura `wsk2_DataArrival`. Odpowiada ona za wykonywanie następujących funkcji programu:

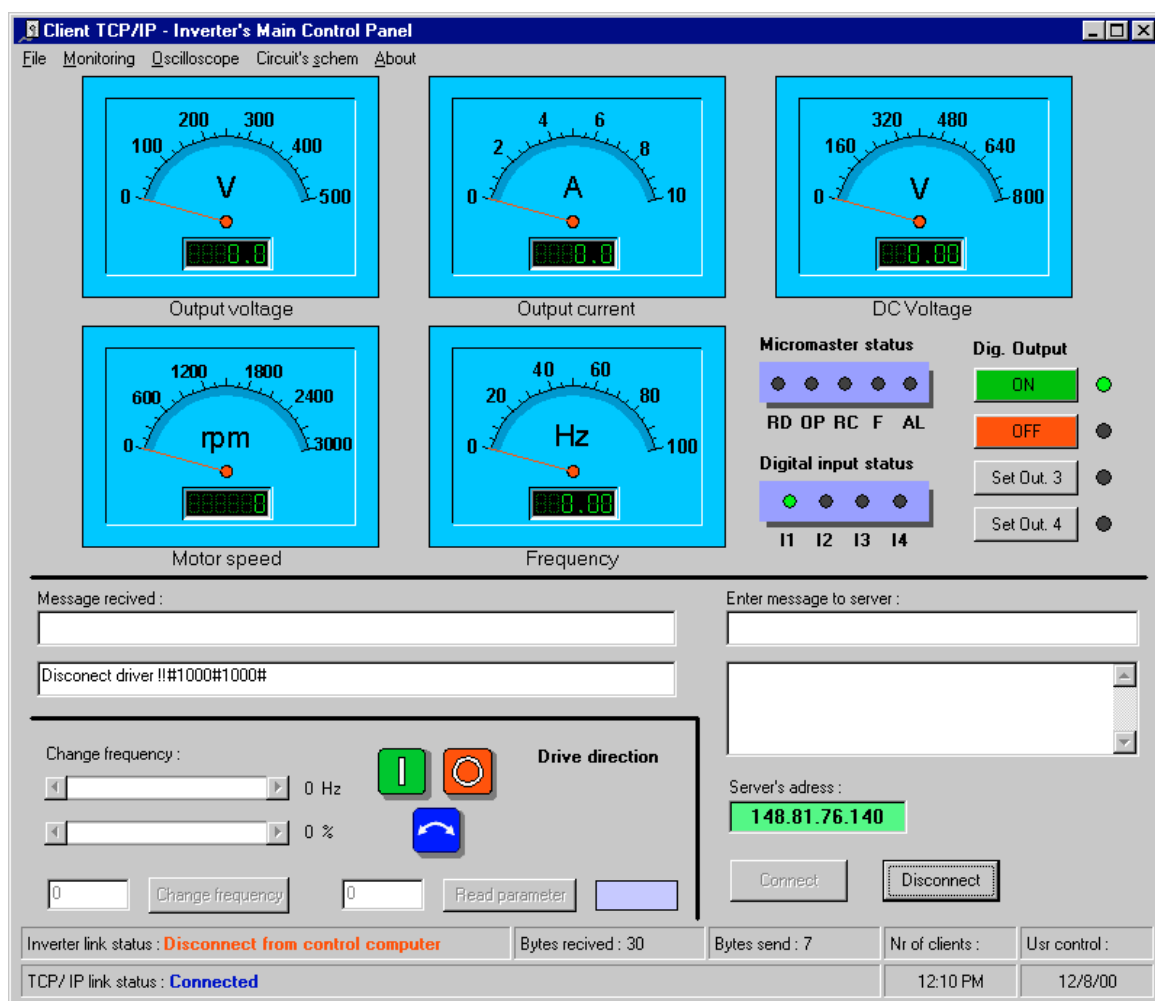
- przetwarzania i wyświetlania poszczególnych wartości mierników,
- wyświetlania poszczególnych stanów wejść/wyjść cyfrowych układu (części **Digital Input status** i **Dig. output** programu),
- przekazywania informacji o stanie połączenia falownika z lokalnym komputerem sterującym do procedury `Timer1_Timer`,
- wyświetlania otrzymanych danych w polach **Message received**,
- wyświetlania odebranych wiadomości tekstowych,

- przetwarzania otrzymanego słowa danych, zawierającego próbki wartości zbieranych przebiegów i przekazywania ich do odpowiedniej tablicy w części programu okna **Oscilloscope**
- odbierania zebranych wartości parametrów falownika i przekazywania ich do części programu okna **Parameters**

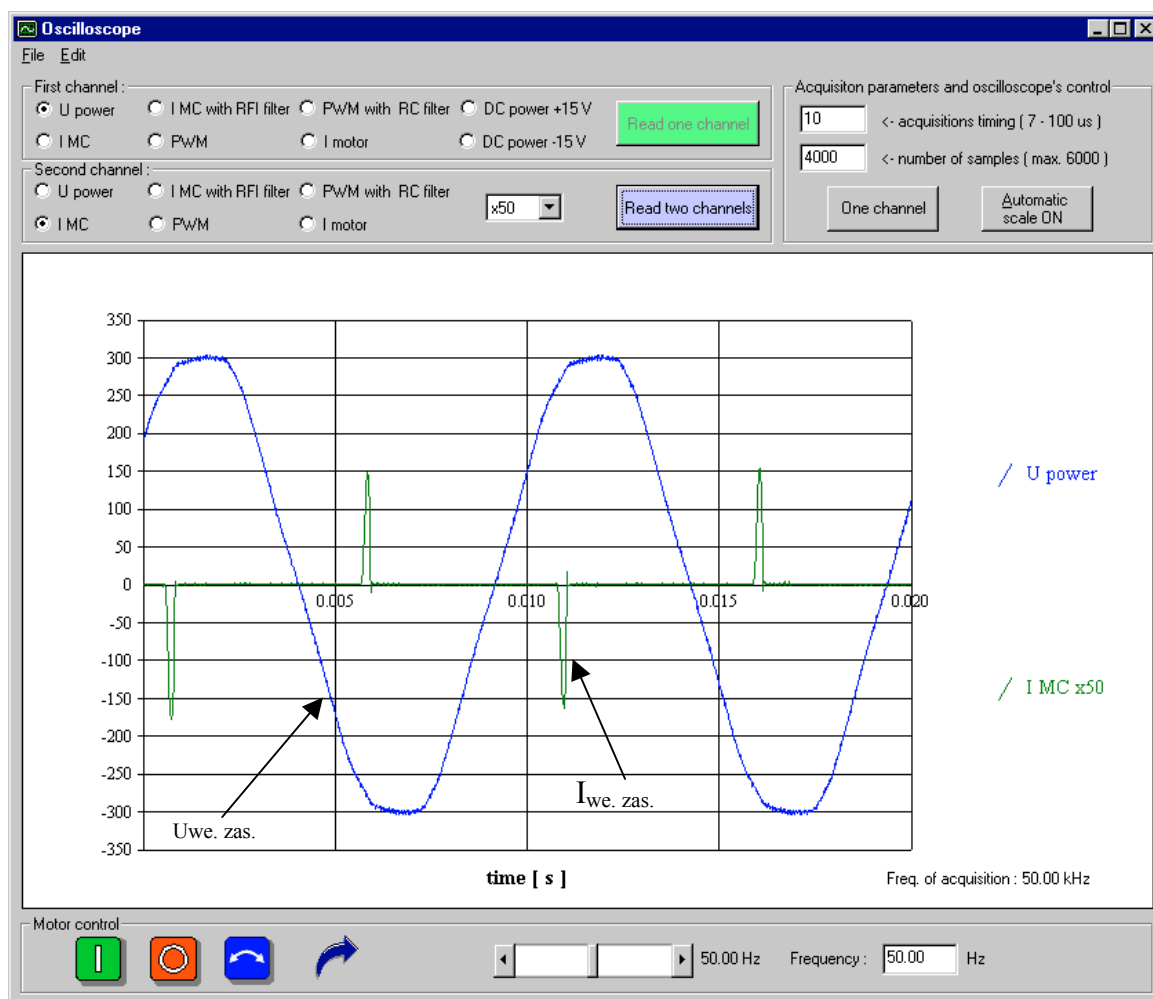
7.2.6. Przykłady działania programu Client'a



Rys. 44 Wygląd głównego okna programu – panelu sterowania, podczas połączenia z serwerem i sterowania układem.



Rys. 45 Wygląd głównego okna programu – panelu sterowania, gdy nie ma połączenia między falownikiem a lokalnym komputerem sterującym.



Rys. 46. Wygląd okna oscyloskopu podczas jednoczesnego odczytu z dwóch kanałów, następujących przebiegów: napięcia fazowego zasilającego falownik ($U_{WE. zas.}$) i prądu wejściowego falownika ($I_{WE. zas.}$).

7.3. „Klient WWW”

7.3.1. Zadania stawiane aplikacji

Aby wybrać odpowiedni język programowania należy odpowiedzieć sobie na pytanie, jakie zadania ma spełniać aplikacja sieciowa?

Przede wszystkim powinna być łatwo dostępna dla użytkowników. Powinna dostarczać narzędzia do współpracy z siecią, takiej jak wymiana informacji. Ponieważ aplikacja ma być dostępna dla szerokiej rzeszy osób pracujących na różnych systemach operacyjnych (Windows, UNIX), musi być programem, który uruchamia się bez względu na to ograniczenie na komputerach wszystkich użytkowników. Z uwagi na nieograniczony dostęp do sieci Internet wielu użytkowników, aplikacja musi spełniać wymogi bezpieczeństwa. Dodatkowo powinna jak najmniej obciążać system operacyjny.

7.3.2. Dlaczego Java?

Prawie każdy użytkownik sieci Internet korzysta, do przeglądania zasobów sieciowych, z niezwykle przydatnych programów jakimi są przeglądarki internetowe. Umieszczone na stronie HTML dane są łatwo dostępne. Jednak strony HTML są mało interakcyjne z użytkownikiem i uniemożliwiają nieustanną wymianę informacji, a także z uwagi na dostarczane narzędzia mają ubogie możliwości. Dlatego producenci przeglądarek wyposażają je w obsługę języków skryptowych takich jak JavaScript. Języki te wprawdzie dodają nowe funkcje do elementów sterujących HTML, takich jak przycisków czy list, jednak nie umożliwiają pełnego zarządzania wyglądem interfejsu użytkownika [15]. Nie można ich również użyć do połączenia komputera-klienta z siecią. Elementy sterujące ActiveX umożliwiają komunikację, ale również pozwalają na niemal nieograniczony dostęp do komputera-klienta, przez co ten staje się otwarty na atak z zewnątrz.

Najbardziej elastycznym rozwiązaniem jest zastosowanie apletów Javy. Aplety Javy umieszcza się na stronie HTML. Ponieważ Java działa na wszystkich popularnych platformach systemowych, aplety są wyświetlane i działają w pożądanym sposobie zawsze wtedy, gdy użytkownik odwiedzający daną stronę Internetu korzysta z jednej z kompatybilnych z Javą przeglądarek.

Java jest językiem najnowszej generacji z ukierunkowaniem na zastosowania sieciowe. Jest to język zorientowany obiektowo, współbieżny ułatwiający programowanie aplikacji klient-serwer. Język „obiektoowy” oznacza programowanie z użyciem klas tworzących szkielet aplikacji.

Java została tak skonstruowana, że połączenie ze stroną WWW lub aplikacją internetową i odczyt oraz zapis danych prawie niczym nie odbiega od wykonania tych samych operacji na komputerze lokalnym. Współbieżność języka znacznie polepsza jego funkcjonalność w zastosowaniu sieciowym. Pozwala na tworzenie wątków, tzw. „lekkich procesów” i ich równoczesną pracę. Wątki umożliwiają apletom wykonywanie wielu zadań jednocześnie, na przykład przesyłania, odbioru danych czy też animacji, nie zakłócając się nawzajem i nie obciążając procesora.

Warto zwrócić uwagę na fakt, iż aplety uruchamiane są na komputerze użytkownika, a więc absorbują serwer jedynie w momencie pierwszego inicjującego połączenia, gdy serwer udostępnia aplet użytkownikowi. Znacznie zmniejsza to konieczność rozbudowy sprzętowej serwera. Po uruchomieniu, aplet wykorzystuje zasoby komputera klienta.

Podnosząc kwestię bezpieczeństwa należy pamiętać, że applety są uruchamiane w pewnym zamkniętym środowisku przeglądarki. Java implementuje mechanizm zabezpieczający zwany „piaskownicą” (*sandbox*), przez co applety działające na komputerze użytkownika końcowego mogą uzyskać dostęp do ograniczonego zbioru jego funkcji systemowych, tak by nie mogły uczynić żadnych szkód.

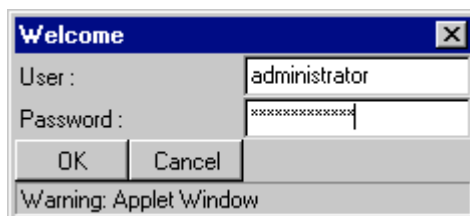
Do uruchomienia appletu należy używać przeglądarki HTML Internet Explorer firmy Microsoft, z domyślnie włączoną obsługą Javy. Alternatywnym rozwiązaniem jest również uruchomienie klienta przy pomocy programu *appletviewer* dostępnego wraz ze środowiskiem JDK (*Java Development Kit*, Zestaw programowania w Javie) firmy Sun Microsystems, wersja 1.1. Jest ono udostępnione na zasadzie bezpłatnej licencji na stronie internetowej firmy Sun Microsystems (<http://www.sun.com>). Na komputerze systemowym niezbędne jest uruchomienie aplikacji serwera Micro, a także serwera obsługującego serwis WWW. Jako serwer WWW w projekcie został zastosowany serwer Apache w wersji 1.3.6 dla Microsoft Windows [20]. Z uwagi na elastyczność oraz wysoką skalowalność, Apache jest najpopularniejszym serwerem WWW do zastosowań w małych i średnich sieciach.

7.3.3. Opis programu „Klient WWW”.

Po uruchomieniu klienta do programu zostaną przekazane parametry :

- adres hosta (HOST) – czyli adres serwera, na którym znajduje się aplikacja serwera,
- wysokość i szerokość (*height*, *width*) jaką będzie zajmował applet w przeglądarce.

Następnie pojawi się okno logowania monitorujące o podanie nazwy użytkownika oraz hasła dostępu (Rys. 47).



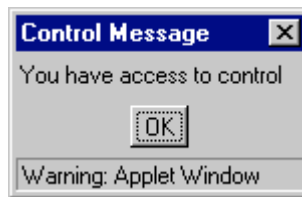
Rys. 47. Okno logowania

Aby ograniczyć dostęp do pewnych funkcji falownika oraz układu automatyki zostały wprowadzone trzy kategorie użytkowników:

- **administrator** – posiada dostęp do wszystkich funkcji udostępnianych użytkownikowi o profilu *guest* oraz czytanie i zmianę parametrów falownika, załączenie i wyłączenie układu zasilania falownika, rozłączenie bądź wznowienie transmisji szeregowej z falownikiem.
- **guest** (gość)– użytkownik mający dostęp do sterowania i monitorowania układu: START, STOP, zmiana prędkości silnika i kierunku obrotów, czytanie parametrów, dostęp do graficznego okna rejestrowanych przebiegów prądów i napięć,
- **anonymous** – użytkownik anonimowy o najniższym poziomie dostępu. Może jedynie monitorować pracę układu bez możliwości jego sterowania. Dodatkowo ma dostęp do rejestrowanych przebiegów.

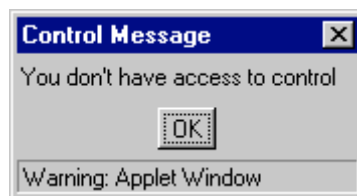
Profile administratora oraz użytkownika „gościa” wymagają podania poprawnego hasła dostępu. Profil użytkownika anonimowego uzyskuje się rezygnując z opcji logowania.

Następnie użytkownik zostanie poinformowany komunikatem w oknie wiadomości **Control Message**, o uzyskanym dostępie do funkcji panelu sterowania. Komunikat uzyskania dostępu („*You have access to control*”) przedstawia rysunek 48.



Rys. 48 Okno komunikatu przyjęcia dostępu

W przypadku odmowy dostępu do sterowania okno wyświetla komunikat o braku dostępu do sterowania („*You don't have access to control*”) przedstawiony na rysunku 49.



Rys. 49 Okno komunikatu odmowy dostępu

Po przejściu etapu logowania przeglądarka wyświetli ekran graficzny panelu sterowania pokazany na rysunku 50, na którym są umieszczone przyciski umożliwiające sterowanie, wirtualne mierniki analogowo-cyfrowe obrazujące mierzone wielkości oraz pola kontrolne transmisji i przesyłanych danych.

Opis panelu sterowania

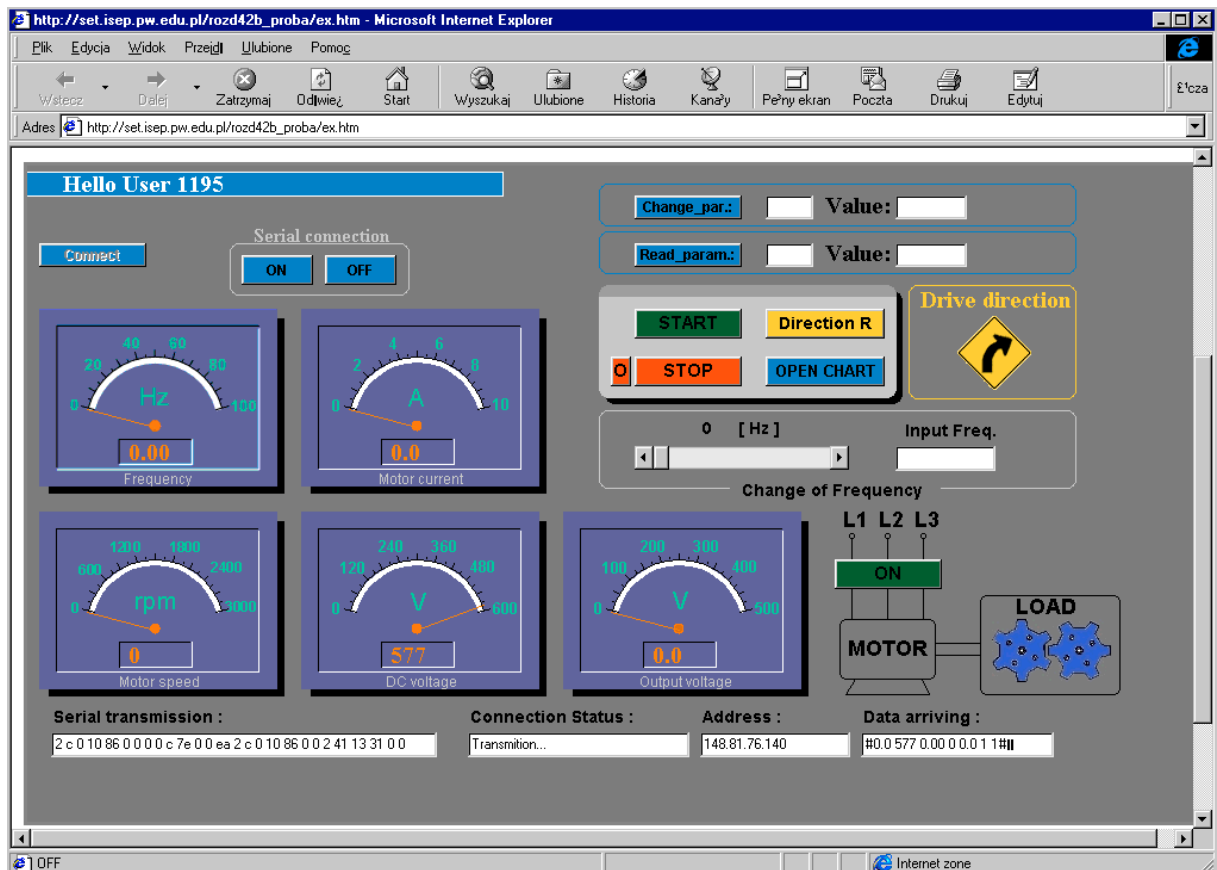
Panel sterowania zawiera następujące komponenty:

- Numer portu komputera użytkownika nadany przez system operacyjny, umożliwiający identyfikację użytkownika:



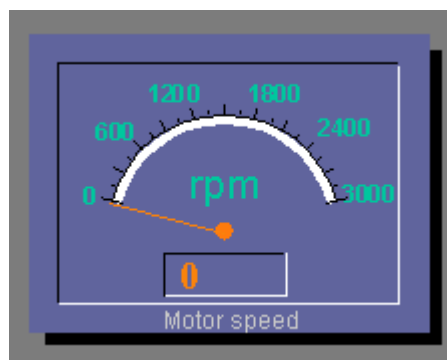
- Przyciski startu i stopu silnika (**START**, **STOP**) ze wskaźnikami ich wciśnięcia „I” oraz „O”





Rys. 50. Panel sterowania

- Sześć wirtualnych mierników analogowo-cyfrowych wskazujących wielkości odebrane z falownika (poniżej miernik prędkości obrotowej silnika **rpm**):



- Przyciski do zmiany kierunku obrotów silnika (**Directon R**, **Direction L**):



- Wskaźnik kierunku obrotów silnika w prawo i w lewo:



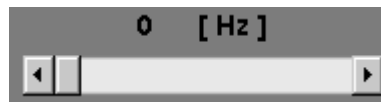
- Przycisk okna przebiegów wielkości mierzonych przez kartę zbierania danych DAQ (**OPEN CHART**):



- Przyciski **ON**, **OFF** sekcji *Serial connection*, służą do rozłączenia lub nawiązania ponownej transmisji z falownikiem poprzez łącze szeregowe:



- Pasek przewijania płynnej zmiany częstotliwości w Hz (częstotliwość można regulować w zakresie od $f=0$ Hz do $f=100$ Hz; ograniczenie zostało wprowadzone programowo):



- Pole skokowej zmiany częstotliwości w Hz (**Input freq.**), zakres regulacji jak dla paska przewijania częstotliwości:



- Przycisk odczytu wartości parametru falownika (**Read param.**) wraz z polem do wpisywania numeru żądanego parametru i polem wyświetlania wartości parametru (**Value**):



- Przycisk zapisu wartości parametru falownika (**Change param.**) wraz z polem do wpisywania numeru żądanego parametru i polem nowej wartości parametru (**Value**):



- Przycisk połączenia z serwerem (**Connect**) – przy braku połączenia z serwerem przycisk jest aktywny:



- Pola kontrolne
 - Transmisji szeregowej słów sterujących między komputerem a falownikiem (**Serial transmission**), dane odbierane i wysyłane między falownikiem a komputerem:

Serial transmission :
2 c 0 10 89 0 0 0 0 c 7 e 0 0 e 5 2 c 0 10 89 0 0 0 0 13 31 0 0

- Połączenia i transmisji danych z komputerem i falownikiem (**Connection Status**):

Connection Status :
Transmission...

Dostępne komunikaty:

- „**Transmission...**” - prawidłowa transmisja danych pomiędzy serwerem a falownikiem oraz pomiędzy serwerem a **Klientem** WWW,
 - „**No connection with inverter**” - zamknięcie połączenia pomiędzy serwerem a falownikiem,
 - „**Server Closed**” – oznacza zamknięcie programu serwera lub niemożliwość nawiązania z nim połączenia.
- Adresu internetowego komputera połączonego z falownikiem łączem szeregowym w postaci IP(**Address**):

Address :
148.81.76.140

- Podglądu danych przychodzący z serwera(**Data arriving**):

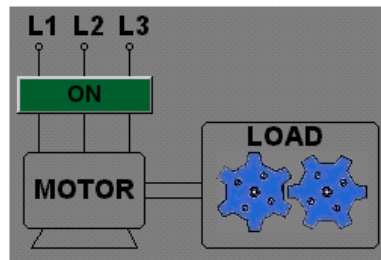
Data arriving :
#0.0 577 0.00 0 0.0 1 1#||

Dostępne informacje:

- Przykładowe dane wartości wielkości mierzonych odbierane z serwera: „#0 0.0 0.0 0 587 0 1 1#” - prawidłowa transmisja danych pomiędzy serwerem a falownikiem,
 - „**Disconnect driver !!#1000#0000#**” – dane wysyłane przez serwer w przypadku braku komunikacji z falownikiem informujące o stanach wejść i wyjść cyfrowych karty DAQ,
 - „disconnected” - oznacza zamknięcie programu serwera.
- Rysunek przedstawiający umowny schemat zasilania silnika oraz obciążenia, z przemiennym przyciskiem załączającym i wyłączającym obwód zasilania falownika (**ON, OFF**):



Obciążenie jest przedstawione jako obracające się koła zębate:



Obsługa panelu sterowania

Po wciśnięciu przycisku **START** prędkość silnika może być zadawana płynnie lub skokowo poprzez zmianę częstotliwości. Do płynnej zmiany prędkości służy pasek przewijania obsługiwany myszką. Wciskając strzałki, w prawo lub w lewo, częstotliwość silnika zwiększa się lub zmniejsza o 1 Hz. Można również przesuwając blok paska przewijania ustawiając go w dowolnym położeniu lub wciskając na jego tło zmieniać wartość częstotliwości o 10 Hz. Gdy wartość częstotliwości zostanie ustawiona na zero silnik zatrzyma się. Drugim elementem do zmiany prędkości silnika jest pole z etykietą **Input Freq** (częstotliwość wejściowa). Umożliwia rozruch silnika i rozpędzenie do zadanej prędkości, a także zmianę prędkości podczas jego pracy. Zadana częstotliwość wpisuje się z klawiatury i potwierdza klawiszem `<Enter>`. Ustawienie częstotliwości na zero zatrzymuje silnik. Aby zatrzymać silnik w dowolnej chwili należy wcisnąć przycisk **STOP**. Podczas pracy silnika wyświetlana jest animacja informująca o jego obrotach.

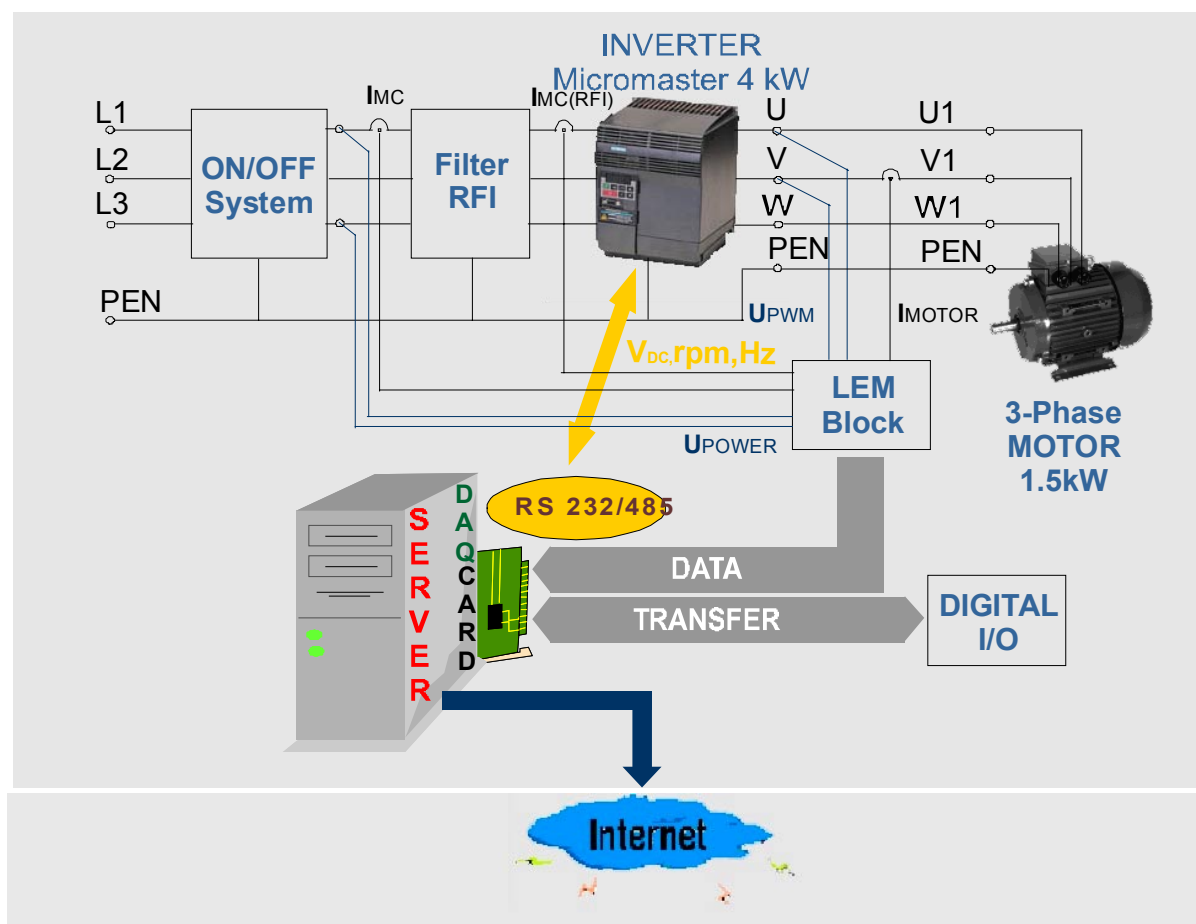
Panel sterownia wyposażony jest w zmianę kierunków obrotu silnika – przyciski prawo, lewo: Direction P (Kierunek P) i Direction L (Kierunek L). Kierunek obrotów może być zadawany przed wystartowaniem silnika lub podczas jego pracy, wówczas realizowany jest nawrót silnika. Potwierdzeniem kierunku obrotów jest wyświetlany znak graficzny kierunku umieszczony w polu z opisem Drive direction (kierunek napędu). Powyżej znajduje się przycisk Read_param (czytaj parametr). Wpisując do pola umieszczonego obok niego, wartości z zakresu od 0 do 971 i wciskając Read_param, zostanie odczytana wartość podanego parametru falownika i wyświetlona w polu z etykietą Value (wartość). Przycisk Read_param umożliwia zmianę wartości wybranego parametru falownika. Wirtualne mierniki analogowo-cyfrowe wskazują kolejno częstotliwość pracy silnika, prąd silnika, prędkość, napięcie obwodu pośredniczącego falownika oraz napięcie zasilania silnika. Wartości te są przekazywane przez falownik.

Poniżej mierników znajdują się pola kontrolne. Pola **Serial Transmission** i **Data arriving** (dane przychodzące) wyświetlają odpowiednio telegram wymieniany między komputerem a falownikiem (opisany w rozdziale 6.2) oraz dane zawierające wartości wielkości mierzonych i ilość użytkowników podłączonych do serwera. Kolejne pola **Connection status** (status połączenia) i **Address** (adres) zawierają odpowiednio status połączenia z falownikiem i serwerem oraz adres serwera, na którym jest uruchomiona aplikacja Micro. Gdy dojdzie do zerwania połączenia, użytkownik może je przywrócić wciskając przycisk **Connect** (połączenie). W przypadku zerwania transmisji szeregowej z falownikiem można je przywrócić przyciskiem **ON** w sekcji *Serial connection*. Z panelu operatorskiego można również załączać i wyłączać zasilanie obwodu głównego przyciskiem **ON** umieszczonego na umownym schemacie zasilania silnika i obciążenia.

W panelu sterowania znajduje się jeszcze jeden przycisk **OPEN_CHART** (otwórz wykres). Umożliwia użytkownikowi podgląd rejestrowanych przebiegów: U_{POWER} , U_{PWM} , $U_{PWM(RC\ filter)}$, I_{MC} , $I_{MC(RFI)}$ oraz I_{MOTOR} . Po wciśnięciu otwiera się nowe okno przeglądarki z menu wyboru zarejestrowanych przebiegów.

Obsługą wyświetlania zajmuje się oddzielny program **ChartApplet** opisany w rozdziale 7.3.4.

Powyżej apletu „Klient WWW” jest umieszczony rysunek 51, przedstawiający schemat blokowy obrazujący układ automatyki.



Rys. 51. Schemat blokowy

Umożliwia to użytkownikowi poznanie układu oraz informuje o sterowanym obiekcie i mierzonych wielkościach.

Kiedy „Klient WWW” uzyska połączenie pojawi się monit o podanie hasła dostępu. Znajomość hasła umożliwia sterowanie, w innym przypadku użytkownik może jedynie monitorować pracę układu automatyki, odczytywać parametry falownika oraz oglądać przebiegi mierzonych wielkości. Po uzyskaniu dostępu do sterowania użytkownik steruje pracą silnika zadając prędkość oraz kierunek obrotów. Dane te są wysyłane do falownika poprzez aplikację serwera działającego na komputerze systemowym. W odpowiedzi na połączenie, klient otrzymuje od aplikacji serwera dane związane z pracą silnika i falownika, i wyświetla je w polach kontrolnych oraz jako wielkości mierzone na wirtualnych miernikach analogowo-cyfrowych: częstotliwości (Frequency), prądu zasilania silnik

(Current motor), prędkości obrotowej silnika (Motor speed), napięcia stałego obwodu pośredniczącego falownika (DC voltage) oraz napięcia zasilania silnika, czyli napięcia wyjściowego falownika (Output voltage). Dane od serwera są przesyłane nieustannie. Klient wysyła dane na życzenie użytkownika, dzięki temu jest prowadzony stały nadzór pracy układu. Użytkownik może odczytywać ustawienia parametrów falownika oraz oglądać przebiegi zarejestrowanych prądów i napięć. Aby zakończyć działanie programu należy zamknąć przeglądarkę lub przejść na inną stronę.

Poniżej zostaną przedstawione fragmenty kodu programu „Klient WWW” oraz rysunek przedstawiający algorytm działania programu dla zobrazowania opisu działania klienta. Kompletny kod programu wraz z plikami binarnymi jest zawarty na CD-ROM’ie dołączonym do niniejszej pracy.

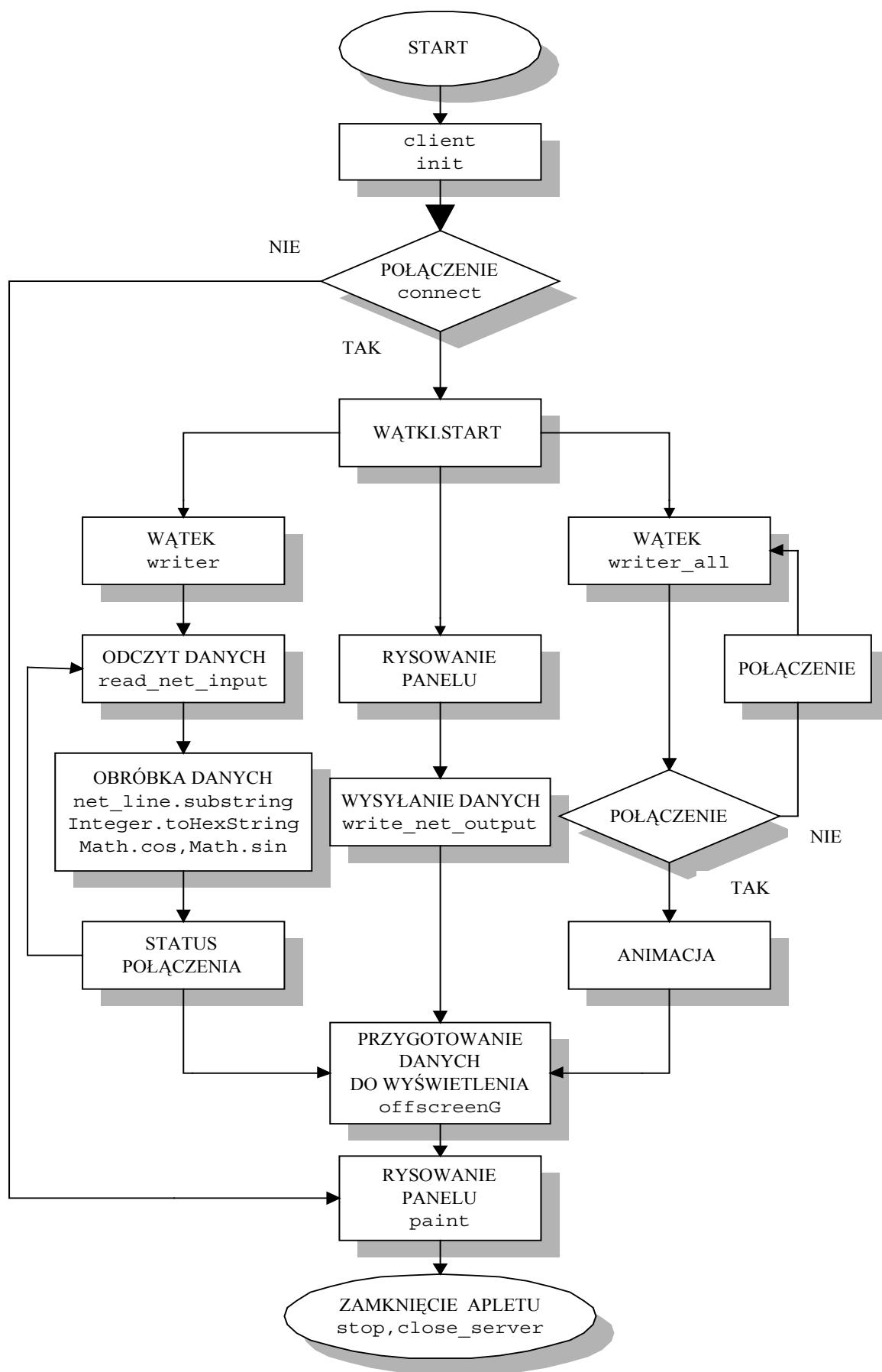
Plik HTML

Aplety Javy umieszcza się na stronie HTML w znaczniku <APPLET>. Plik ex.html uruchamiający aplet „Klient WWW” (client.class) wygląda, jak poniżej:

```
(..)  
<APPLET code="client.class" width="980" height="550" Align=center>  
<param name="HOST" value="148.81.76.140">  
</APPLET >  
(..)
```

Program „Klient WWW”

Algorytm działania programu przedstawia rysunek 52. Główna klasa client definiuje zmienne używane w programie, a także dostarcza narzędzi niezbędnych do połączenia apletu z serwerem. Uruchomione zostają dwa niezależne wątki (writer, writer_all) odbierający i obrabiający dane oraz kontrolujący stan połączenia i funkcjonowania programu. Zauktualizowane dane są bezpośrednio wyświetlane na ekranie.



Rys. 52 Algorytm działania programu

Inicjalizacja Apletu

Funkcja `init` dostarcza nazwę komputera serwerowego z pliku HTML przechowywaną w zmiennej `HOST`, a następnie łączy kolejno rysunki mierników wirtualnych (`img_Hz`, `img_v5`, `img_v6`, `img_rpm`, `img_A`) kierunku obrotów (`curv_l`, `curv_r`) oraz dźwięku (`beep`) informującego o wciśnięciu jednego z przycisków sterujących. Po uzyskaniu adresu następuje próba połączenia z komputerem funkcją `connect()`. Jeżeli zakończona jest sukcesem, tworzony jest nowy obiekt `writer` i rozpoczyna się połączenie. W przeciwnym razie funkcja zwraca błąd.

```
public void init()
{
    (...)
    host = getParameter("HOST");
    (...)
    // załadowanie rysunków
    img_Hz = getImage(getCodeBase(), "Hz.gif");
    img_v5 = getImage(getCodeBase(), "V500.gif");
    img_v6 = getImage(getCodeBase(), "V600.gif");
    img_rpm = getImage(getCodeBase(), "rpm.gif");
    img_A = getImage(getCodeBase(), "A.gif");
    curv_l = getImage(getCodeBase(), "curv_left_yellow.gif");
    curv_r = getImage(getCodeBase(), "curv_right_yellow.gif");

    //dzwieki
    beep = getAudioClip (getCodeBase(), "Drip.au");
    (...)
    if (connect())
    {
    (...)
        connected = true;
        w = new writer(this);
        w.start();
        wa = new writer_all(this);
        wa.start();
    }
    else
    {
        connected = false;
        username = "NOT CONNECTED";
    }
    (...)
}
```

Połączenie z serwerem

Klasa `client` używa funkcji `connect()` do połączenia z serwerem. W tym celu utworzony jest nowy obiekt `Socket`. Z funkcji `connect()` jest przekazywana nazwa komputera serwerowego oraz numer portu (12345) do obiektu `Socket`. Należy pamiętać o wychwyceniu wyjątku, czyli błędu podczas wykonywania działania programu, w przypadku nieudanego połączenia z serwerem, w przeciwnym razie nastąpi zawieszenie programu (blok `try` i `catch`). Funkcja `connect` tworzy unikalną nazwę użytkownika (`username`) korzystającego z lokalnego numeru portu. Umożliwia to powrót do portu lokalnego, do którego jest podłączone gniazdo (`socket`). Aby otrzymywać i wysyłać dane

przez gniazdo trzeba również otrzymać strumienie wejścia i wyjścia używane przez gniazdo. Do tego celu są wykorzystywane funkcje `getInputStream` (strumień wejściowy) oraz `getOutputStream` (strumień wyjściowy). I tu również jest konieczne „wyłapanie” wyjątku. Na koniec funkcja wysyła nazwę użytkownika (`write_net_output`) i zwraca wartość `true` jeżeli połączenie zakończyło się pomyślnie:

```
boolean connect ()
{
    try
    {
        server = new Socket (host, 12345);
    }
    catch (IOException e)
    {
        return false;
    }

    username = "" + server.getLocalPort ();

    try
    {
        net_input = server.getInputStream ();
        net_output = server.getOutputStream ();
    }
    catch (IOException e)
    {
        return false;
    }

    write_net_output (username + "\n");
    return true;
}
```

Czytanie i pisanie w sieci komputerowej

Strumień wejściowy czytający dane z sieci obsługuje funkcja `read`. Do niej z kolei jest przekazywane odwołanie do tablicy bajtów `byte [1024]`, a także liczba znaków jaka jest w niej przechowywana. Gdy funkcja `read` odczyta znak to funkcja `read_net_input` zwróci go, po uprzednim przekształceniu w łańcuch. Jeżeli nie będzie bajtów do przekształcenia, czyli serwer nie wysyła danych zwracana jest wartość `null` oznaczająca przerwanie połączenia lub brak danych:

```
String read_net_input ()
{
    byte bytes [];
    int number_of_bytes;

    try
    {
        bytes = new byte [1024];
        number_of_bytes = net_input.read (bytes);
        if (number_of_bytes > 0 )
            return (new String (bytes, 0, 0, number_of_bytes));
        else return null;
    }
    catch (IOException e)
```

```
    {  
        return null;  
    }  
}
```

Do wysyłania danych przez użytkownika służy funkcja `write_net_output`. Funkcja ta przesyła dane w sieć na żądanie użytkownika, np. gdy użyje jednego z przycisków na panelu operatorskim. Podobnie jak w funkcji czytającej dane z sieci tak i tu przed wysłaniem łańcucha znaków należy go przekształcić w tablicę bajtową `byte_array[]`:

```
void write_net_output(String string)  
{  
    byte byte_array[];  
  
    int length = string.length();  
    byte_array = new byte[length];  
    string.getBytes(0, length, byte_array, 0);  
    try  
    {  
        net_output.write(byte_array);  
    }  
    catch (IOException e)  
    {}  
}
```

Zamykanie połączenia z serwerem

Do zamknięcia połączenia z serwerem służy funkcja `close_server`. Jest ona wywołana przez funkcję `stop` gdy okno przeglądarki zostanie zamknięte lub użytkownik opuści stronę z apletem:

```
void close_server()  
{  
    try  
    {  
        server.close();  
    }  
    catch (IOException e)  
    {}  
}
```

Zatrzymanie działania apletu

Gdy użytkownik chce zamknąć aplet, zamyka przeglądarkę lub przechodzi na inną stronę HTML. Powoduje to zamknięcie ustanowionego połączenia z serwerem. Wywoływana jest wówczas funkcja `stop` aktualizująca działanie apletu. W funkcji `stop` następuje zamknięcie połączenia z serwerem (`close_server`) oraz zamknięcie wszystkich wątków apletu (`w`, `wa`). Przeglądarka zwalnia również zasoby systemowe zarezerwowane dla apletu:

```
public void stop()  
{  
    close_server();  
    if(w!=null) w.stop();  
}
```

```
        if (beep!=null) beep.stop();
        if (wa!=null) wa.stop();
    }
```

Funkcja paint

Funkcja `paint` ma szczególne znaczenie w działaniu apletu. Jej zadaniem jest natychmiastowe odświeżanie okna apletu, gdy tylko zaistnieje taka potrzeba. Jest ona wywoływana po każdej zmianie danych przychodzących od serwera oraz po każdej zmianie będącej odpowiedzią na reakcję użytkownika. Zatem służy do odświeżania wskazań wirtualnych mierników (`offscreenG.drawImage`, `offscreenG.drawString`), zmiany kierunku obrotu silnika (`curv_l`, `curv_r`), wyświetlania okien tekstowych m.in. statusu połączenia (`inputField7.setText`), przychodzących danych (`inputField6.setText`), czy też elementów grafiki (`offscreenG.fillRect`, `offscreenG.setColor`, `offscreenG.fillRoundRect`). Funkcja `paint` pełni jeszcze jedną istotną rolę, dba o to aby aplet był ponownie widoczny gdy przesłonimy okno przeglądarki inną aplikacją lub gdy zostanie ono zminimalizowane:

```
public void paint(Graphics g)
{
    (..)
    //tło pod przyciski
    offscreenG.fillRect(0,0,980,550);
    offscreenG.setColor(Color.black);
    offscreenG.fillRoundRect(485,105,250,95,15,15);
    offscreenG.setColor(new Color(153,153,153));
    offscreenG.fillRoundRect(480,100,250,40,15,15);
    offscreenG.setColor(Color.lightGray);
    offscreenG.fillRect(480,110,250,20);
    offscreenG.fillRoundRect(480,115,250,80,15,15);
    (..)
    //wirtualny miernik czestotliwosci Hz
    if (img_Hz !=null)
    {
        offscreenG.setColor(Color.black);
        offscreenG.fillRect(xHz+8,yHz+8,198,147);
        offscreenG.drawImage(img_Hz, xHz, yHz, this);
        offscreenG.setColor(new Color(231, 118, 5));
        offscreenG.drawString(w.display_third, xHz+75, yHz+127);
    }
    (..)
    inputField6.setText(w.net_line);
    inputField7.setText(""+w.b);
    g.drawImage(offscreenI, 0, 0, this);
}
```

Ponieważ podczas działania apletu występuje efekt migotania obrazu, aby tego uniknąć została zastosowana technika podwójnego buforowania [14]. Migotanie obrazu jest związane ze zmienną częstotliwością odświeżania wielu powierzchni rysowanego ekranu graficznego. W aplecie wyświetlane są elementy, które zmieniają się w różnym czasie oraz z różną szybkością. I tak, tło apletu zmienia się szybko, podczas gdy wyświetlanie danych w polach kontrolnych zmienia się wolniej. Technika podwójnego buforowania znakomicie sprawdza się w zastosowaniu do apletów. Polega ona na rysowaniu wszystkich elementów apletu w pamięci komputera i po zakończeniu tej operacji przeniesieniu całego obrazu na ekran

monitora. Pamięć jest zwalniana i operacja powtarza się. Stąd nazwa podwójne buforowanie, gdyż zawsze są tworzone dwa obrazy. Jeden, niewidoczny dla użytkownika przechowywany w buforze pamięci i drugi narysowany na ekranie monitora. W funkcji `paint` obrazy są przenoszone do bufora `offscreenG`, a ich narysowaniem zajmuje się funkcja `drawImage` wyrysowując na ekranie utworzony kontekst graficzny obrazu `offscreenI`.

Klasa `writer`

Klasa `writer` tworzy wątek (*thread*). Jej zadaniem jest odczytywanie danych przychodzących z sieci i ich obróbka. Tutaj następuje odczyt poszczególnych wartości wielkości mierzonych takich jak: prąd, napięcie, częstotliwość czy prędkość (`net_line.substring`). Po uzyskaniu mierzonych wartości następuje obliczanie punktów współrzędnych niezbędnych do wyświetlenia położenia wskazówek wirtualnych mierników (`Math.cos`, `Math.sin`). Klasa `writer` przekształca również słowa sterujące, wysyłane od falownika do aplikacji serwera oraz od serwera do falownika, w ciąg znaków heksadecymalnych (`Integer.toHexString`), tak aby słowo sterujące miało postać zdefiniowaną zgodnie z protokołem USS opisanym w rozdziale 6.2. Ponieważ klasa `writer` jest wątkiem, działa jednocześnie z innymi funkcjami nie wstrzymując ich działania:

```
class writer extends Thread
{
    (...)

    public writer(client c)
    {
        this.c = c;
    }

    public void run()
    {
        (...)

        //odczyt prądu i wyliczenie współrzędnych
        display_first=net_line.substring(index_next,i);
        alfa_A=(15*A)+15;
        alfa_rA=(alfa_A*3.14)/180;
        xdwaA=Math.cos(alfa_rA)*L;
        ydwaA=Math.sin(alfa_rA)*L;
        xxdwaA=(int)xdwaA;
        yydwaA=(int)ydwaA;

        (...)

        znak_16_st = znak_16_st+Integer.toHexString(znakASCII_st)+" ";

        (...)

    }
}
```

Klasa `writer_all`

Klasa `writer_all` jest drugim wątkiem apletu zajmującym się wyświetlaniem animacji oraz monitorowaniem połączenia z serwerem. Gdy silnik się obraca następuje ładowanie kolejnych obrazów animacji (zmienna `currentimg`) i widoczny jest, na panelu sterowania, efekt obracania się kół zębatych symbolizujących obciążenie.

```
if (a.w.s!=0)
{
(..)

    a.currentimg=a.nekopics[ii];
    ii++;
    if(ii==4) ii=0;

(..)
}
```

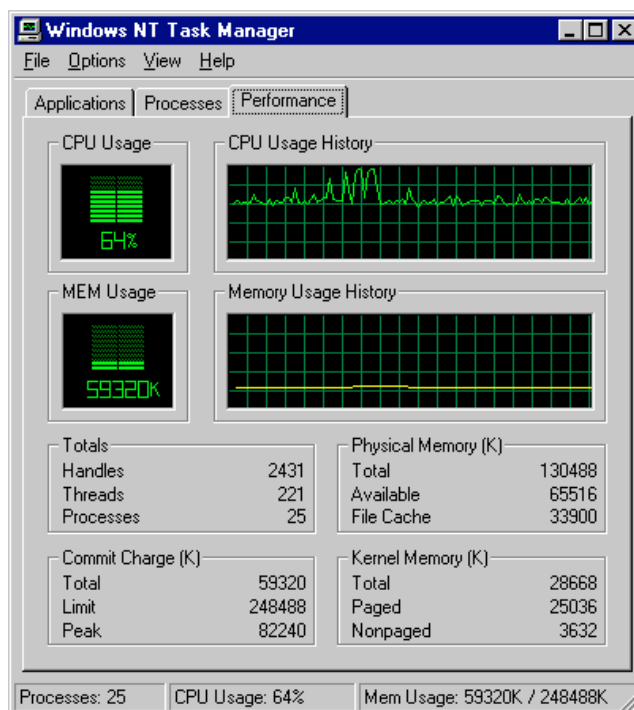
Drugą funkcję jaką spełnia wątek jest kontrola połączenia. Przerwanie połączenia z serwerem zatrzyma wykonywanie wątku `writer`, wówczas wątek `writer_all` uaktywni przycisk `Connect` na panelu sterowania, aby umożliwić użytkownikowi ponowne połączenie.

```
if (a.w==null)
{
(..)

    a.connect_b.setEnabled(true); //aktywacja przycisku Connect

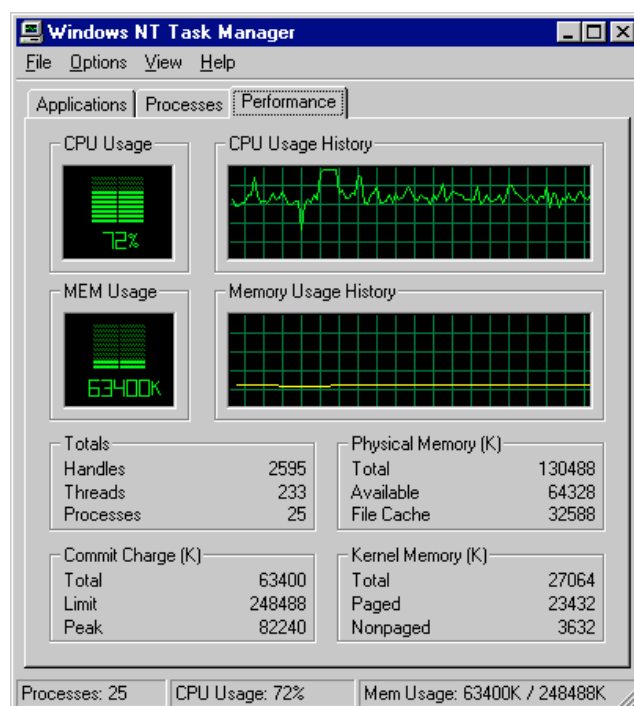
(..)
}
```

Program nieznacznie obciąża system operacyjny wykorzystując jedynie ok. 4MB pamięci operacyjnej. Przeprowadzono testy wydajności obciążenia systemu dla zobrazowania wpływu programu. W tym celu został wykorzystany Menadżer zadań (Windows NT Task Manager) systemu Windows NT. Rysunek 53 przedstawia obciążenie systemu podczas pracy programu serwera Micro i przeglądarki Internet Explorer. Wykorzystanie pamięci operacyjnej wynosi 59.3 MB, wykorzystanie procesora to ok. 64%.



Rys. 53. Menedżer zadań systemu Windows NT

Rysunek 54 przedstawia sytuację po uruchomieniu programu "Klienta WWW". Wykorzystanie pamięci wzrosło do 63.4 MB, wykorzystanie procesora do ok.72%.



Rys. 54. Menadżer zadań systemu Windows NT

Konkludując, program wykorzystuje jedynie 4MB pamięci operacyjnej przy równie niewielkim 8% wzroście wykorzystania procesora.

7.3.4. Opis programu do rysowania przebiegów – ChartApplet

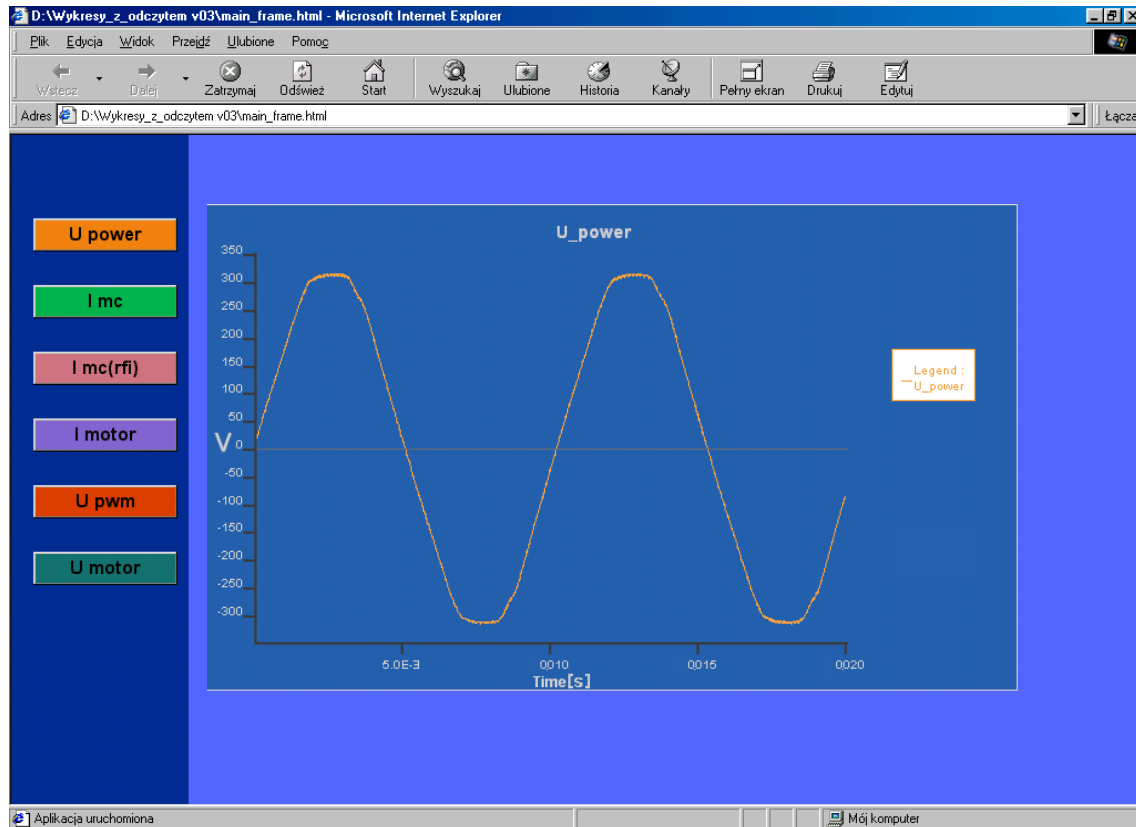
Do przedstawienia przebiegów U_{POWER} , U_{PWM} , $U_{PWM(RC\ filter)}$, I_{MC} , $I_{MC(RFI)}$ oraz I_{MOTOR} zastosowano aplet "ChartApplet". Program pobiera dane z sześciu plików tekstowych umieszczonych na serwerze WindowsNT kolejno nazwanych od 1.txt do 6.txt, zawierających dane dla kolejnych przebiegów. Następnie dokonywana jest konwersja i formatowanie danych niezbędne do wyświetlenia wykresów.

Program jest wywoływany przez aplet „Klient WWW”. Po uruchomieniu zostaje wyświetlone okno przeglądarki przedstawione na rysunku 55, które zawiera menu wyboru poszczególnych przebiegów.



Rys. 55 Menu wyboru

Po wyborze żądanego przebiegu jest on wyświetlany w prawej części okna przeglądarki. Okno z przebiegiem przedstawia rysunek 56 na przykładzie przebiegu napięcia fazowego zasilania falownika U_{POWER} .

Rys. 56 Menu oraz wykres napięcia zasilania - U_{POWER}

Plik HTML

Z chwilą wyboru z menu żadanego przebiegu otwierana jest strona WWW i wyświetlana w ramce prawej. Strona ta uruchamia aplet **ChartApplet** i przekazuje do niego parametry związane z nazwą przebiegu (TITLE), plikiem z próbkami (PLIK), oznaczeniem osi współrzędnych (XLABEL, YLABEL), podziałką (BIG_TIC_INTERVALX, BIG_TIC_INTERVALY), ilością próbek do odczytania (XSCALE_MAX) oraz formą wyświetlanych opisów i kolor wykresu (SERIE_STYLE_1). Fragment pliku main_frame.html jest przedstawiony poniżej:

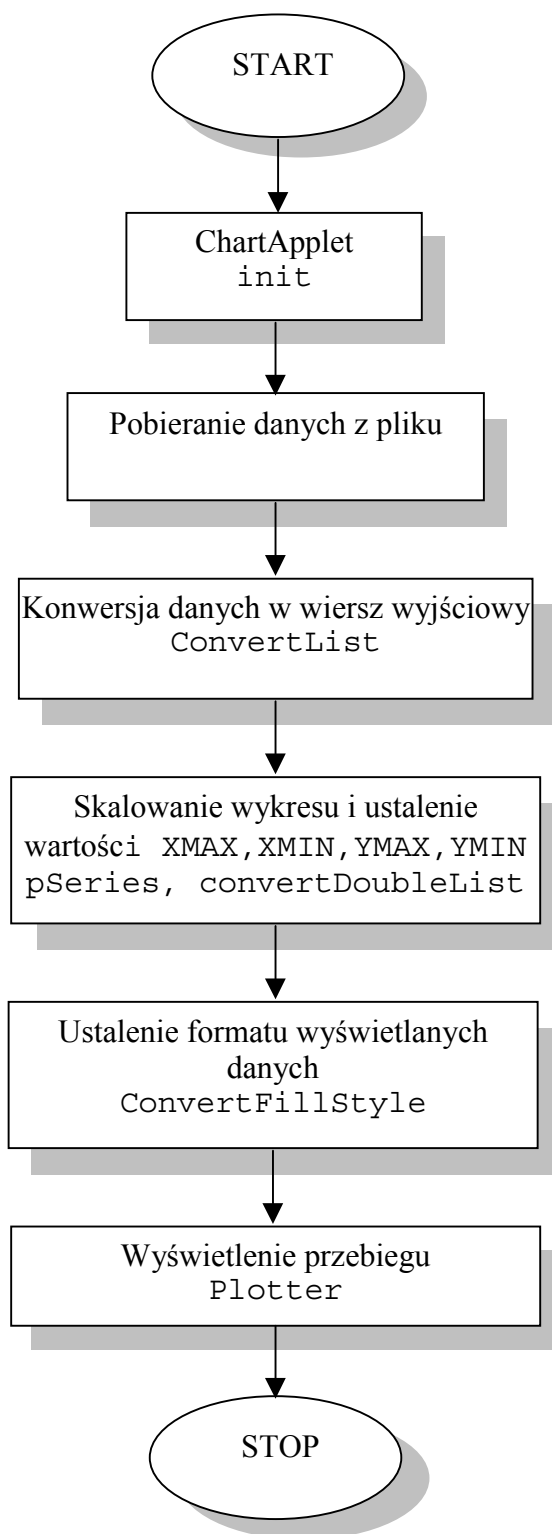
```
<APPLET CODEBASE = "." CODE = "ChartApplet.class" WIDTH =700 HEIGHT=400 >
```

```
(..)
```

```
<PARAM NAME = "PLIK" VALUE = "6.txt">
<PARAM NAME = "TITLE" VALUE = "Power_supply">
<PARAM NAME = "XLABEL" VALUE = "Time">
<PARAM NAME = "YLABEL" VALUE = "V">
<PARAM NAME = "XSCALE_MAX" VALUE = "2001">
<PARAM NAME = "SERIE_STYLE_1" VALUE = "0.2,#E77605,LINE">
<PARAM NAME = "BIG_TIC_INTERVALX" VALUE = "500">
<PARAM NAME = "BIG_TIC_INTERVALY" VALUE = "50">
</APPLET>
```

Program ChartApplet

Algorytm opisujący działanie programu ilustruje rysunek 57.



Rys. 57. Algorytm działania programu ChartApplet

Funkcja `init` dostarcza do apletu parametry zawarte w pliku HTML. Gdy zostanie odczytana nazwa pliku z próbkami następuje pobieranie kolejnych danych – próbka po próbce. Następnie funkcja `convertLineStyle` konwertuje dane:

```
public LineStyle convertLineStyle(String f) {  
    String[] items=convertList(f);  
  
    if (items==null) return null;  
    (...)  
  
}
```

a funkcja (`convertList`) przekształca je w wiersz danych:

```
private String[] convertList(String items){  
  
    String[] itema=new String[2500];  
    int itemCount=0;  
  
    // count number of items  
    int p=items.indexOf(",");  
    while (p>=0) {  
        itema[itemCount++]=items.substring(0,p);  
        items=items.substring(p+1,items.length());  
        p=items.indexOf(",");  
    }  
  
    (...)  
  
    return result;  
  
}
```

oraz dekoduje zadeklarowany w pliku HTML kolor przebiegu, z postaci szesnastkowej do postaci dziesiętnej (`ConvertColor`):

```
private java.awt.Color convertColor(String c) {  
  
    (...)  
  
    try {  
        return java.awt.Color.decode(c);  
    } catch (Exception e) {return java.awt.Color.black;}  
}
```

Kolejne próbki są ładowane do tablicy `pSeries[]` i wyszukiwana jest wartość maksymalna i minimalna próbki oraz ilość próbek (`ConvertDoubleList`):

```
private double[] convertDoubleList(String s) {  
  
    (...)  
  
    double[] d=new double[items.length];  
    for (int i=0; i<items.length;i++) {  
        try {  
            d[i]=new Double(items[i]).doubleValue();  
        }  
    }  
}
```

```
} catch (Exception e) {d[i]=0;}  
}  
return d;  
}
```

Funkcja skaluje wykres, tak aby umieścić cały przebieg w oknie apletu. Dalej odbywa się formatowanie zadeklarowanej, w pliku HTML, czcionki opisów przebiegu oraz opisów osi. Gdy dane są już przygotowane i sformatowane funkcja `Plotter` rysuje przebieg w oknie przeglądarki:

```
public class Plotter extends ChartComponent  
{  
  
    (...)  
  
    public void plot(Graphics g)  
    {  
        for(int i = 0; i < Series.size() ; i++)  
        {  
            DataSerie s = (DataSerie)Series.elementAt(i);  
            plotSerie(g, s, i);  
        }  
    }  
  
    (...)  
  
}
```

8. Wnioski i uwagi końcowe

Opracowanie i budowa stanowiska laboratoryjnego pozwoliły na realizację systemu zdalnego monitoringu i sterowania falownikowego układu napędu. Interfejs komunikacyjny falownika oraz wykonane bloki automatyki umożliwiły dwukierunkowe przenoszenie informacji pomiędzy układem a oprogramowaniem.

Napisana aplikacja „Micro” zrealizowała postawione jej we wstępie zadania.

Zaimplementowanie protokołu transmisji szeregowej USS między komputerem a falownikiem, umożliwiło jego monitoring i zdalne sterowanie.

Wszystkie opcje sterowania, stan pracy falownika, dostęp i zmiana parametrów, mogą być realizowane z poziomu programu. Wykorzystano także wewnętrzne zdolności pomiarowe przemiennika częstotliwości, polegające na przekazywaniu odpowiednich wartości: napięć, prądu, prędkości oraz częstotliwości do komputera. Wartości te są wyświetlane na wirtualnych miernikach analogowo-cyfrowych.

Zastosowanie i oprogramowanie karty zbierania danych oraz wykorzystanie jej funkcji w programie, umożliwiło zbudowanie wirtualnego oscyloskopu dwukanałowego, a także sterowanie pracą poszczególnych podzespołów stanowiska laboratoryjnego i ich nadzór.

Budując główny panel sterujący, na którym umieszczone zostały wirtualne komponenty, uzyskano łatwy i komfortowy dostęp za pomocą myszki i/lub klawiatury komputera do wielu podzespołów falownikowego układu napędu, umożliwiając ich monitoring i sterowanie.

Szybki dostęp do nastaw parametrów falownika i możliwość ich wydruku, pozwolił na dużą wygodę obsługi w stosunku do ręcznego przeglądania pojedynczych parametrów z jego panelu operatorskiego. Możliwość zapisu do pliku próbek zbieranych przebiegów pozwala na dalszą ich obróbkę w takich programach jak np. „Matlab”.

Stosowanie takiego rodzaju monitoringu i sterowania układu automatyki jest jak najbardziej wskazane i coraz częściej stosowane w rozbudowanych układach przemysłowych.

Dla potrzeb naszego projektu stanowiska laboratoryjnego, napisane oprogramowanie monitorujące i sterujące pracą falownikowego układu napędu z wykorzystaniem komputera, było podstawowym krokiem w dalszym etapie rozwoju całego systemu, jego rozbudowy i wykorzystania sieci komputerowej Intranet/Internet.

Zastosowany system komunikacji (Klient ↔ Serwer), umożliwił wymianę informacji między programem „Micro” i programami klientów. Protokół TCP/IP, zapewniając szybką i poprawną transmisję danych w sieci umożliwia sterowanie i monitorowanie układu z komputera podłączonego w dowolnym miejscu sieci.

Komunikacja w sieci Intranet/Internet została zrealizowana dwiema metodami. Jedna z nich polegała na napisaniu aplikacji pod dany system operacyjny komputerowy, druga na napisaniu aplikacji działającej niezależnie od platformy systemowej.

Programem napisanym pod dany system była aplikacja „Client TCP/IP”. Zastosowano w niej większość opcji i funkcji programu „Micro”.

Aplet „Klient WWW” umożliwia sterowanie i monitorowanie zdalnymi układami automatyki przyłączonymi do sieci Internet. Wraz z apletem ChartApplet, ilustrującym przebiegi prądów i napięć, stanowi w pełni wystarczającą aplikację służącą do komunikacji z serwerem oraz zdalnej obsługi układów automatyki.

Program klienta umieszczony na stronie WWW jest łatwo dostępny i jednocześnie bezpieczny – wymiana informacji jest chroniona przez protokół komunikacyjny TCP/IP, a dostęp do programu przez hasło. Dzięki zastosowaniu języka Java program „Klient WWW” pracuje również pod systemem UNIX. Pozwala to na szerokie zastosowanie tego

typu programów niezależnie od systemu operacyjnego, na którym odbywa się sterowanie i monitoring.

Aplikację można rozbudować do postaci pozwalającej operatorowi na sterowanie i monitoring z jednego miejsca grupy układów bądź procesów technologicznych.

Umożliwi to znaczny wzrost szybkości reakcji na zmiany zachodzące w kontrolowanych obiektach oraz zwiększy łatwość porównywania i zestawiania informacji pochodzących od wielu obiektów.

9. Bibliografia

- [1] **Micomaster – instrukcja obsługi.** T-System Projekt Sp. z o. o. Siemens partner. Luty 1998.
- [2] **Micomaster – Operating instructions.** Siemens PLC 1997
- [3] **Technicaldescription ADDIALOG PA 3000 – Analog inputs.** ADDI-DATA. 10th edition. 07/1999
- [4] **Technical description ADDIALOG PA3000 – Standard software.** ADDI-DATA 10th edition 07/1999
- [5] **USS Protocol User Guide.** SIEMENS PLC. 16/10/98
- [6] Marc Murphy.: **Debugging Serial Communication with the MICROMASTER, MIDIMASTER, Combi and 6SE21 Inverters.** 08/08/97
- [8] Bob Roselman, Richard Peasley, Wayne Pruchniak.: **Poznaj Visual Basic 6.** Wydawnictwo MIKOM. Warszawa, maj 1999.
- [9] Steve Brown.: **Visual Basic 6. Tom1, Visual Basic 6. Tom2.** Wydawnictwo EXIT. Warszawa 1998.
- [10] Douglas E. Comer, David L. Stevens.: **Sieci komputerowe TCP/IP. Programowanie w trybie klient-serwer. Wersja BSD.** Wydawnictwo Naukowo Techniczne. Warszawa 1997.
- [11] Miller, Mark A. **Sieci TCP/IP : wykrywanie i usuwanie problemów.** Wydawnictwo RM. Warszawa 1999.
- [12] Karanjit S. Siyan.: **Windows NT TCP/IP.** Wydawnictwo ROBOMATIC. Wrocław 1998.
- [13] Laura Lemay, Charles L. Perkins.: **Java 1.1.** Wydawnictwo Helion. Gliwice 1998.
- [14] Mike Morgan.: **Poznaj język JAVA 1.2.** Wydawnictwo MIKOM. Warszawa 1999.
- [15] Romowicz, Wojciech.: **HTML i JavaScript,** Wydawnictwo Helion. Gliwice1998.
- [16] Bishop, Judith M; Bishop, Nigel T.: **Java gently for engineers and scientists.** Wydawnictwo Pearson Education. Harlow 2000.
- [17] Sun Microsystems, Inc.: **Java™ Standard Edition Platform Documentation.** Strona internetowa <http://java.sun.com/docs> Sun Microsystems, Inc.1995.
- [18] Sun Microsystems, Inc.: **The Java Tutorial – A practical guide for programmers.** Strona internetowa <http://java.sun.com/docs/books/tutorial/index.html> Sun Microsystems, Inc. 2000.

- [19] Jacek Rumiński.: **Język Java – wykład.** Strona internetowa
<http://biont.eti.pg.gda.pl/java.htm>. Gdańsk 1999.
- [20] The Apache Group.: **Apache HTTP Server version 1.3 - User's Guide.**
Dokumentacja The Apache Group. Illinois 1995.
- [21] Grunwald, Zdzisław.: **Napęd elektryczny : praca zbiorowa.** Wydaw. Nauk. -
Techn., Warszawa 1987.
- [22] Paul Horovitz, Winfield Hill.: **Sztuka Elektroniki / Cz.1,2.** Wydaw. Komunik. i
Łączn. Warszawa 1995
- [23] Beliczyński Bartłomiej, Koziński Wojciech.: **Wprowadzenie do regulacji cyfrowej**
Wydaw. Politechniki Warszawskiej. Warszawa,1983.
- [24] **Podzespoły dla energoelektroniki.** Katalog firmy Darcpol Sp. z o.o. 1999
- [25] **ELFA nr 47. 1 kwietnia 1999 – 31 marca 2000.** Katalog firmy Elfa Polska
Sp. z o.o. Warszawa 1999
- [26] **Visual Basic Tools - pomocnik programisty.** Strona internetowa.
<http://www.vbtools.prv.pl/>. Wrocław 5.12.2000
- [27] Daniel Paul.: **Visual Basic for all.** Strona internetowa.
<http://www.vb4all.net/>. 1998-2000

10. Aneks

10.1. Zestawienie elementów

Schemat ideowo - blokowy stanowiska laboratoryjnego (rysunek 1)		
Nazwa	Symbol	Typ
Wyłącznik krzywkowy 3 biegunowy	WG	LE2-16-1753
Zasilacz napięcia stałego +6V		ZS 6V
Bezpieczniki		S193
Panel operatorski	OPM2	

Złącza falownika (rysunek 2)		
Nazwa	Symbol	Typ
Złącze RS 485	Złącze RS 485	D

Schemat blokowy układu styczników (rysunek 4)		
Nazwa	Symbol	Typ
Wyłączniki	START	
	STOP	
Przełączniki	P1, P2,	CB-NO
Stycznik ze stykiem pomocniczym 1Z	S	CI12
	1S	AC-15
Oprawka tablicowa 5x20, z lampką sygnalizacyjną	Ż	FEL

Schemat ideowy zasilacza napięcia stabilizowanego ± 15 V (rysunek 7)		
Nazwa	Symbol	Typ
Mostek prostowniczy	MIC W04m.	MIC W04M
Transile	T1, T2	SM6T27CP
Diody	D1, D2	T4148
Kondensatory	C1, C7	470 μ F/63 V
	C2, C8	180 nF/63 V
	C3, C4, C9, C10	0.1 μ F/63 V
	C5, C12	22 nF/63 V
	C6, C13	1000 μ F/63 V
Stabilizatory napięcia	LM7815	LM 7815
	LM7915	LM 7915
Transformator	TS	TS 15/4/676

Schemat ideowy modułu pomiarowego (rysunek 8)		
Nazwa	Symbol	Typ
Rezystory (MŁT 5%/0.25)	R1, R2,	3 kΩ
	R4, R5	68 kΩ
	R6, R7, R10, R11	82 kΩ
	R8, R12	8.2 kΩ
	R9, R13	120 kΩ
Potencjometry	Rp1, Rp2, Rp3, Rp4, Rp5, Rp6	1 kΩ
	Rp7	2.2 kΩ
	Rp8, Rp9	10 kΩ
Diody	D1, D2, D3, D4, D5, D6, D7, D8, D9, D10, D11, D12	T4148
	D13, D14, D15, D16, D17, D18, D19, D20, D21, D22, D23, D24	1N4007GP
DIODY LED	LED1, LED2	
Kondensatory	C1, C2, C3, C4, C5, C6, C7, C8, C9, C10, C11, C12,	33 nF
	C13	1 μF
	C14, C15, C16, C17, C18, C19, C20, C21, C22, C23, C25	120 μF
Układ pomiaru prądu LEM	LEM	LA 55 – P (3 szt.)
Układ pomiaru napięcia LEM	LEM	LV 25 – P (3 szt.)
Listwa łączeniowa do druku	Listwa zasilająca	MPT 0.5/2-2.54
Złącze SUB-D	SUB-D	SUB-D 37-pin

Opis wyjść złącza SUB-D 37-pin (rysunek 9)

Nazwa	Symbol	Typ
Złącze 37-pin	Złącze SUB-D 37-pin	SUB-D 37-pin

Schemat ideowy modułu wejść/wyjść cyfrowych (rysunek 12)

Nazwa	Symbol	Typ
Rezystory (MŁT 5%/0.25)	R1, R2, R3, R4	10 kΩ
	R5, R6, R7, R8	2.7 KΩ
Kondensatory	C1, C2, C3, C4	10 nF/63 V
Diody	LED1, LED2, LED3, LED4	EL204
	LED5, LED6, LED7, LED8	EL1533
Złącze 25-pin	Złącze SUB-D 25-pin	SUB-D
Listwy łączeniowe do druku	Z1...Z9	MPT 0.5/2-2.54

Złącza wejść/wyjść cyfrowych z opisem (rysunek 14)		
Nazwa	Symbol	Typ
Złącze 25-pin	Złącze SUB-D 25-pin	SUB-D 25-pin

11. Dodatki

CD-ROM, zawierający kody źródłowe oraz pliki binarne programów:

- Programy: „Micro” i „Client TCP/IP” zawarte są w katalogu **KodVB**
- Programy: „Klient WWW” i „ChartApplet” zawarte są w katalogu **KodJava**.