Politechnika Warszawska

Wydział Elektryczny
Laboratorium Programowania urządzeń mikroprocesorowych
automatyki elektroenergetycznej

Skrypt do ćwiczenia M.32

Wizualizacja zasady działania zabezpieczenia mikroprocesorowego – implementacja w języku asembler.

1. Cel i zakres ćwiczenia

Celem ćwiczenia jest poznanie podstawowych wiadomości związanych z budową oraz zasadą działania zabezpieczeń mikroprocesorowych. Zabezpieczenia te coraz częściej stosowane są do zabezpieczania krajowych obiektów energetycznych.

Program ćwiczenia, w oparciu o istniejące stanowisko laboratoryjne, obejmuje:

- omówienie budowy oraz zasady działania poszczególnych bloków funkcjonalnych wchodzących w skład zabezpieczenia mikroprocesorowego
- zapoznanie ze sterownikiem mikroprocesorowym mogącym pełnić funkcję zabezpieczenia cyfrowego
- przedstawienie działania sterownika na podstawie przykładowych programów
- ćwiczenia w zakresie programowania sterownika mikroprocesorowego.

2. Wprowadzenie

Elektroenergetyczna Automatyka Zabezpieczeniowa jest dziedziną automatyki elektroenergetycznej zajmującą się zapobieganiem oraz likwidacją zakłóceń w SEE lub poszczególnych jego elementach.

W związku z ciągłym rozwojem techniki zmieniają się środki wykorzystywane przez urządzenia EAZ, co pociąga za sobą zmiany w kierunku rozszerzania zadań funkcjonalnych zabezpieczeń. Zmiany te powodują, że miejsce pojedynczych przekaźników zajmują zespoły zabezpieczeń określane mianem terminali.

Na początku lat dziewięćdziesiątych, w wyniku burzliwego rozwoju technik mikroprocesorowych oraz otwarcia Polski na świat, w dziedzinę EAZ wkroczyła technika mikroprocesorowa. Rozwój i coraz intensywniejsze wdrożenia zabezpieczeń cyfrowych następują od kilku lat, gdyż zabezpieczenia te wykazują wiele zalet, zwłaszcza w powiązaniu z nowoczesnymi systemami sterowania i nadzoru pracy sieci. Dodatkowym czynnikiem, który przyśpiesza ten proces jest stały rozwój telemechaniki.

Zaletami zabezpieczeń cyfrowych są przede wszystkim ich możliwości informatyczne, dotyczące rejestracji działania zabezpieczeń i wielkości zakłóceniowych, oraz inne właściwości takie jak:

- ciągła samokontrola, zwiększająca wydatnie niezawodność eksploatacyjną
- diagnostyka obwodów zewnętrznych
- zmniejszenie wymiarów aparatury i okablowania wtórnego
- pomiary parametrów energii elektrycznej
- możliwość pomiaru energii dostarczonej i niedostarczonej
- wizualizacja łączników pola
- możliwość zdalnej kontroli i zmiany nastaw
- szeroka gama funkcji zabezpieczeniowych.

Wymagania stawiane zespołom zabezpieczeniowym stają się coraz wyższe. Od zabezpieczeń wymaga się przede wszystkim selektywności i pewności w działaniu.

Technika mikroprocesorowa umożliwia zastosowanie odmiennych niż dotychczasowe algorytmów pomiarowych, które poprawiają selektywność jak również umożliwiają skrócenie czasu pomiaru, a w efekcie zmniejszenie czasu własnego działania zabezpieczeń. Prowadzi to do skrócenia czasów wyłączeń awarii.

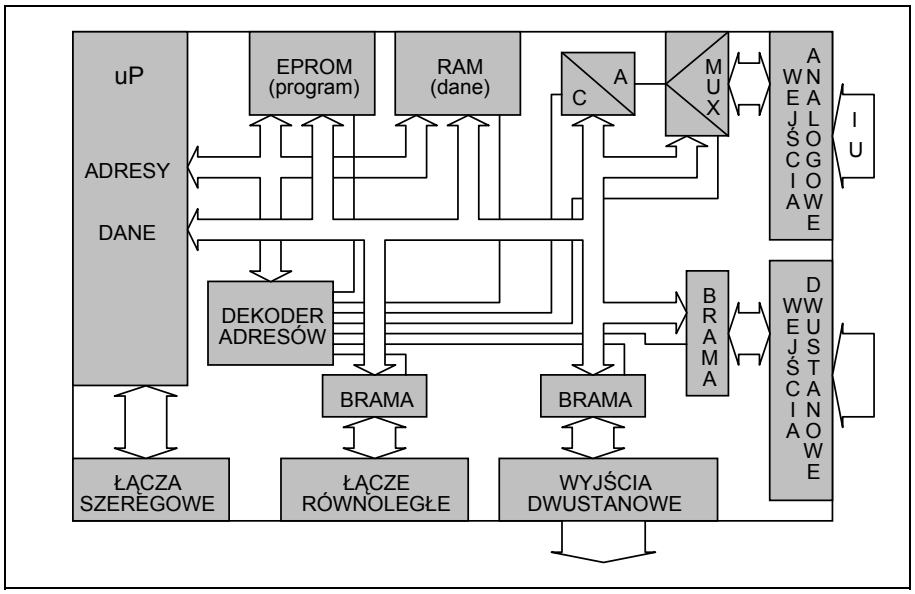
Zabezpieczenia cyfrowe są już trzecią generacją zabezpieczeń, poprzednie - to przekaźniki elektromechaniczne i elektroniczne.

Okres współdziałania zabezpieczeń różnej generacji będzie jeszcze długi i trudny na razie do określenia. Należy się jednak spodziewać, że przyszłość EAZ należeć będzie właśnie do zabezpieczeń cyfrowych.

3. Budowa i zasada działania zabezpieczenia mikroprocesorowego

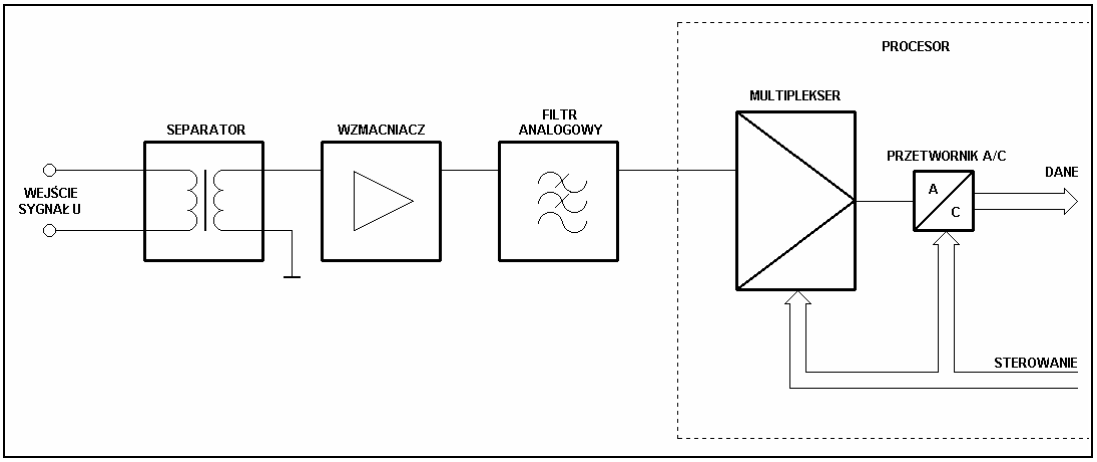
Każde zabezpieczenie mikroprocesorowe jest swego rodzaju sterownikiem mikroprocesorowym. Składa się ono z określonej liczby niezbędnych modułów, które pozwalają mu prawidłowo funkcjonować i spełniać swoje zadanie. Zabezpieczenie ma odbierać, zapamiętywać i obrabiać dane analogowe i cyfrowe doprowadzane do niego z zabezpieczanego obiektu. Powinno zatem posiadać następujące bloki funkcjonalne:

- wejścia analogowe przystosowane do współpracy z obwodami wtórnymi przekładników prądowych i napięciowych zabezpieczanego obiektu energetycznego
- wejścia dwustanowe współpracujące z obwodami wtórnymi zabezpieczanego obiektu energetycznego
- wyjścia dwustanowe współpracujące z łącznikami bądź innymi elementami obiektu energetycznego
- jednostkę centralną, wykorzystywaną do obróbki i przetwarzania przychodzących danych oraz do komunikacji z użytkownikiem
- zasilacz zasilający wszystkie obwody zabezpieczenia i gwarantujący ciągłość pracy urządzenia.



Rys. 3.1. Schemat blokowy zabezpieczenia

Wejścia analogowe przeznaczone są do współpracy z obwodami wtórnymi przekładników prądowych i napięciowych zabezpieczanego obiektu energetycznego. Zadaniem ich jest dopasowanie sygnałów analogowych do poziomów napięć przetwornika analogowo-cyfrowego używanego do przekształcania sygnału analogowego na cyfrowy.

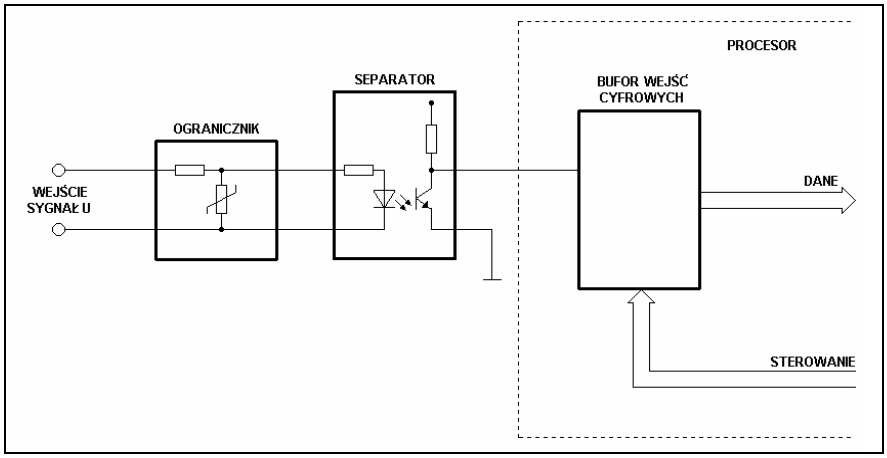


Rys. 3.2. Wejścia analogowe

Wejścia analogowe w zabezpieczeniu dzielimy na prądowe i napięciowe. Tory napięciowe przystosowane są do współpracy z przekładnikami napięciowymi zabezpieczanego obiektu energetycznego, a prądowe do współpracy z przekładnikami prądowymi. Wartości napięć i prądów doprowadzanych do wejść analogowych z obwodów wtórnych przekładników powinny być wartościami znormalizowanymi i wynosić np. Un=100V i In=1A (są to wartości skuteczne napięć i prądów). Dodatkowo wejścia napięciowe powinny dokładnie przenosić przetężenia o wartości do 2Un, natomiast prądowe (40 ### 100)In.

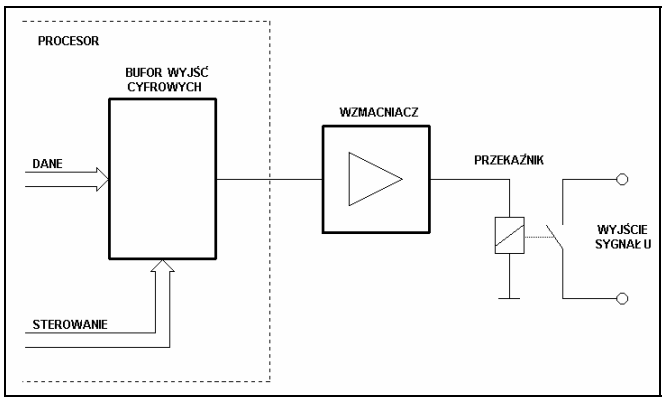
W kanałach analogowych sygnały są najpierw separowane galwanicznie (ze względów bezpieczeństwa) od obwodów wtórnych przekładników głównych zabezpieczanego obiektu energetycznego oraz od siebie. We wzmacniaczu są one dopasowywane do poziomów napięć przetwornika analogowo-cyfrowego. Ze wzmacniacza sygnał trafia na filtr analogowy dolnoprzepustowy. Jest on nieodzowną częścią wejść analogowych. Jego rolą jest wycięcie z przebiegu wejściowego częstotliwości większych od połowy częstotliwości próbkowania. Zapobiega on w ten sposób powstawaniu efektu zdudnienia częstotliwości objawiającego się tym, że sygnał o wysokiej częstotliwości może być odbierany jako sygnał o częstotliwości niskiej.

Wejścia dwustanowe przystosowują zabezpieczenie do współpracy z obwodami wtórnymi zabezpieczanego obiektu energetycznego. Ich przeznaczeniem jest dopasowanie napięć pomocniczych (220V DC, 110V DC) do poziomu sygnału cyfrowego (zwykle TTL). Kanały wejściowe dwustanowe podobnie jak kanały analogowe zapewniają separację galwaniczną urządzenia od napięcia pomocniczego obwodów wtórnych obiektu.



Rysunek 3. Wejścia dwustanowe

Wyjścia dwustanowe sterują łącznikami (np. wyłącznikami) bądź innymi elementami zabezpieczanego obiektu energetycznego. Przystosowane są do współpracy z napięciem pomocniczym stacji (220V DC lub 110V DC) oraz posiadają określoną zdolność łączeniową, np. 4A.



Rysunek 4. Wyjścia dwustanowe

Jednostkę centralną stanowi układ mikroprocesorowy posiadający przetwornik analogowo-cyfrowy, układy wejść i wyjść cyfrowych, pamięć, wyświetlacz i klawiaturę, łącza szeregowe do współpracy z modemem oraz komputerem nadrzędnym. Podstawowym zadaniem jednostki centralnej jest zbieranie danych z przetwornika analogowo-cyfrowego oraz wejść cyfrowych, potem zapisywanie ich do pamięci i zmiana stanu wyjść cyfrowych na podstawie realizowanego algorytmu. Jednostka centralna obsługuje łącze szeregowe w celu zapewnienia komunikacji zabezpieczenia z urządzeniem nadrzędnym którym jest przeważnie komputer klasy IBM PC. Klawiatura umożliwia obsłudze bezpośrednią komunikację z zabezpieczeniem.

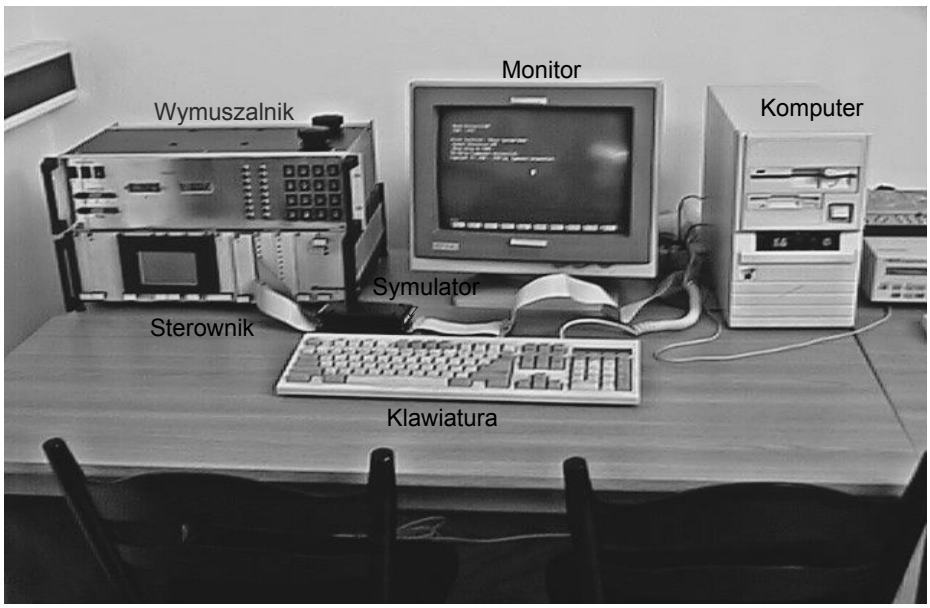
Zasilacz powinien dostarczać napięć zasilających potrzebnych do pracy wszystkich układów wchodzących w skład zabezpieczenia. Ważna jest zatem jego duża bezawaryjność. Zabezpieczenie

mikroprocesorowe powinno posiadać pewne, bezprzerwowe zasilanie. W większości przypadków dostarczane jest do zasilacza napięcie stałe (220V DC lub 110V DC) które jest gwarantowane na stacji.

4. Wyposażenie i opis stanowiska laboratoryjnego

Zbudowane stanowisko laboratoryjne składa się z kilku elementów funkcjonalnych. Każdy z nich pełni określoną funkcję. Zostały one dobrane w taki sposób, aby razem tworzyły w pełni dydaktyczną pomoc naukową.

Fotografia 1 przedstawia zdjęcie zbudowanego stanowiska.



Fotografia 1. Stanowisko laboratoryjne

W skład stanowiska wchodzą następujące elementy:

- wymuszalnik prądowy
- wymuszalnik napięciowy
- wymuszalnik sygnałów dwustanowych z modułem wizualizacji zadziałania sterownika
- sterownik mikroprocesorowy
- symulator pamięci EPROM / debuger sprzętowy
- komputer klasy IBM PC.

Wymuszalnik prądowy został podłączony do wejść prądowych sterownika. Dostarcza on prąd sinusoidalnie zmienny o częstotliwości 50Hz. Prąd ten wytwarzany jest przez autotransformator z włączonym w szereg z jego zaciskami wyjściowymi rezystorem o wartości 50Ω. Dzięki takiemu rozwiązaniu można uzyskać prąd o wartości regulowanej w zakresie od 0 do 4A.

Wymuszalnik napięciowy dostarcza napięcia sinusoidalnie zmiennego o częstotliwości 50Hz do wejść napięciowych wykorzystanego sterownika. Jego źródłem jest także autotransformator. Zapewnia on regulację napięcia w zakresie od 0 do 250V.

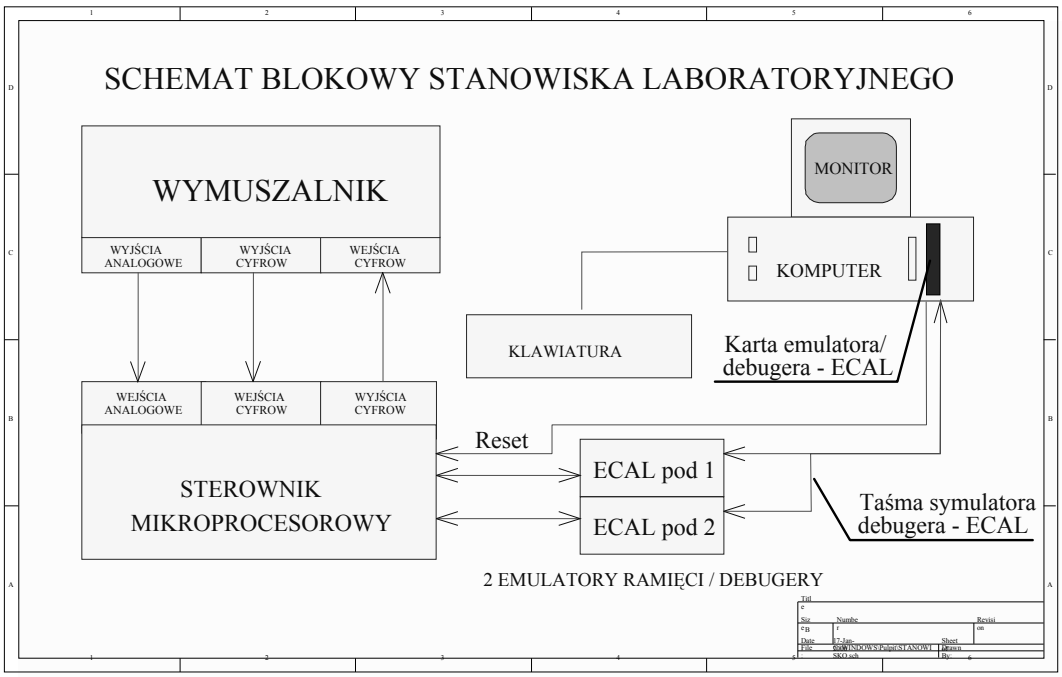
Do wejść cyfrowych sterownika mikroprocesorowego doprowadzono sygnały dwustanowe wytworzone przez układ wymuszalnika sygnałów dwustanowych. Sygnałami tymi są napięcia o wartości 220V DC załączane za pomocą przełączników. Wyjścia cyfrowe sterownika podłączono do modułu wizualizacji, który stanowią diody LED. Układ informuje o tym, że realizowany algorytm spowodował zadziałanie bądź nie układu mikroprocesorowego.

Podstawowym elementem wykonanego stanowiska jest sterownik mikroprocesorowy zbudowany w oparciu o procesor SAB 80C166 firmy SIEMENS. Urządzenie posiada wejścia analogowe, wejścia i wyjścia cyfrowe, jednostkę centralną, zasilacz. Sterownik wykonuje określony algorytm na podstawie zapisanego w jego pamięci programu i sygnałów doprowadzonych do jego wejść.

Symulator pamięci EPROM, jest urządzeniem, dzięki któremu sterownik mikroprocesorowy może prawidłowo funkcjonować bez fizycznej obecności pamięci EPROM (w pamięci tej zapisywany jest program sterujący pracą mikroprocesora).

Komputer klasy IBM PC jest urządzeniem nadrzędnym i służy do komunikacji ze sterownikiem. Posiada niezbędne oprogramowanie umożliwiające edycję, kompilację, symulację i przesyłanie gotowych programów za pośrednictwem symulatora pamięci EPROM do sterownika mikroprocesorowego oraz śledzenia programu wykonywanego przez sterownik.

Schemat połączeń pomiędzy poszczególnymi elementami stanowiska zamieszczono na rysunku 4.



Rysunek 5. Schemat połączeń między poszczególnymi elementami stanowiska

5. Opis wykorzystanego sterownika mikroprocesorowego

Podstawowym elementem wykonanego stanowiska laboratoryjnego jest sterownik mikroprocesorowy mogący pełnić funkcję zabezpieczenia cyfrowego.

Sterownik posiada:

- wejścia analogowe (prądowe i napięciowe)
- zabezpieczenie wejść analogowych procesora
- wejścia dwustanowe
- wyjścia dwustanowe
- jednostkę centralną
- zasilacz.

Sterownik posiada 4 wejścia prądowe i 4 wejścia napięciowe. Wejścia prądowe mogą współpracować z obwodami wtórnymi przekładników prądowych obiektu energetycznego o prądzie wynoszącym 1A. Posiadają one separację galwaniczną (2kV/1min.) zrealizowaną przy pomocy transformatorów pośredniczących.

Wejścia napięciowe mogą być przyłączone do przekładników napięciowych głównych obiektu energetycznego o napięciu znamionowym wynoszącym 100V. Posiadają one separację galwaniczną (2kV/1min.) zrealizowaną przy pomocy transformatorów pośredniczących. Wejścia te, podobnie jak wejścia prądowe, są zrealizowane na wzmacniaczach operacyjnych.

W wykorzystanym sterowniku nie zainstalowano filtru dolnoprzepustowego. Zostało to podyktowane faktem, iż sterownik ma służyć jedynie celom dydaktycznym w zakresie poznania budowy i zasady działania zabezpieczeń mikroprocesorowych a także ze względów oszczędnościowych.

Jeżeli chodzi o wejścia dwustanowe, to jest ich 7 (docelowo sterownik został zaprojektowany na 14 wejść). Zostały one dostosowane do napięcia pomocniczego 220V DC występującego na większości stacji elektroenergetycznych w Polsce. Separacja galwaniczna od obwodów wtórnych obiektu wynosi 2kV/1min. Została ona zrealizowana za pomocą transoptorów.

W wykorzystanym sterowniku mikroprocesorowym istnieje 7 wyjść dwustanowych (ich również docelowo może być 14). Mogą one pracować z napięciem pomocniczym wynoszącym 220V DC.

Separację galwaniczną układów wewnętrznych sterownika od napięcia stałego 220V uzyskano przez zastosowanie przekaźników.

Jednostkę centralną stanowi procesor SAB 80C166 wraz z pamięcią i układami komunikacji z komputerem nadrzędnym. W sterowniku zastosowano 10-cio bitowy przetwornik analogowo-cyfrowy. Potrafi on przenieść bez zniekształceń z dokładnością 1%, 10-cio krotne przekroczenie prądu znamionowego In ($10In_0$ 1000%, $2^{10} = 1024 > 1000$). Przetwornik ten jest integralną częścią procesora.

Zasilacz dostarcza napięć niezbędnych do zasilania wszystkich układów wchodzących w skład sterownika.

W związku z błędami, jakie powstają na skutek nieliniowości wzmacniaczy i przetwornika AC na każdym kanale pojawia się offset. Ze względu na dokładność pomiaru offset ten musi być skorygowany do wartości bliskiej zeru.

W opisanym sterowniku wartość offsetu została zniwelowana programowo za pomocą odpowiednio ustalonych współczynników, które są dodawane do wartości próbek odpowiednich kanałów. Wartości 0 na wejściu odpowiada wartość próbki 200H.

Wartości rzeczywiste próbek zostały przedstawione w poniższej tablicy:

kanał	wartość próbki
0	1F9H
1	1F8H
2	1F3H
3	200H
4	1F7H
5	1F5H
6	20BH
7	200H

6. Programowanie sterownika mikroprocesorowego

Każdy układ mikroprocesorowy musi być wyposażony w pamięć, która służy do przechowywania informacji. Pamięć, do której informacja może zostać wpisana a następnie odczytana, nosi nazwę pamięci RAM (Random Access Memory – pamięć o dostępie swobodnym). Wszystkie informacje wpisane do tej pamięci są bezpowrotnie tracone w momencie wyłączenia zasilania. W wykorzystanym sterowniku pamięć RAM służy do przechowywania argumentów i wyników wszelkiego rodzaju obliczeń.

Pamięć, w której informacje są wpisane na stałe, to pamięć ROM (Read Only Memory – pamięć stała). Do tej pamięci w czasie normalnej pracy układy mikroprocesorowego nie może być wpisana żadna nowa informacja. Z pamięci tego typu informacje mogą być jedynie odczytywane i nie zostają one utracone w chwili wyłączenia zasilania (są dostępne po ponownym włączeniu zasilania). Zawartość pamięci ROM jest ustalana w momencie jej produkcji i w żaden sposób nie może być już potem zmieniana.

Najczęściej funkcję pamięci ROM w układzie mikroprocesorowym spełnia pamięć EPROM (Erasable and Programmable Read Only Memory – kasowalna i programowalna pamięć stała). Pamięć EPROM w czasie normalnej pracy układu zachowuje się tak, jak pamięć ROM. Jednak po wyjęciu z układu może być ona skasowana poprzez wystawienie jej na działanie promieniowania ultrafioletowego. Po skasowaniu do pamięci EPROM mogą być wpisane nowe informacje. Służą do tego specjalne urządzenia zwane programatorami pamięci EPROM.

Właśnie do pamięci ROM lub EPROM jest wpisywany ciąg rozkazów sterujących pracą procesora. Ten ciąg rozkazów stanowi program wykonywany przez procesor. Bez takiego programu sterownik mikroprocesorowy nie będzie pracował.

W zastosowanym sterowniku nie zainstalowano stałej pamięci EPROM, lecz symulator pamięci EPROM, który również posiada możliwość odbierania z uruchamianego systemu mikroprocesorowego zwrotnych informacji, przez co umożliwia śledzenie wykonywanego tam programu oraz wartości rejestrów, pamięci itp. Układ symulatora/debugera składa się z karty ISA pracującej w komputerze PC

oraz połączonych z nim wielożyłową wstęgą modułów emulatorów, bezpośrednio podłączonych do podstawek układów pamięci programu jednostki centralnej sterownika mikroprocesorowego.

Na komputerze PC znajduje się program o nazwie ECAL, który spełnia rolę środowiska uruchomieniowego. Integruje w sobie funkcje: edytora, asemblera, symulatora pamięci oraz debugera.

7. Ćwiczenia

Mikroprocesor wykonuje program zapisany w pamięci programu. Zawartość pamięci to zestaw bitów. Przygotowanie przez człowieka programu bezpośrednio w takiej postaci jest praktycznie niemożliwe. Dlatego dla każdego procesora istnieje jego asembler. Jest to język programowania, w którym rozkazowi napisanemu przez programistę odpowiada rozkaz procesora. Rozkazy zapisuje się w postaci symbolicznej, znacznie czytelniejszej dla człowieka. Program napisany przez programistę w postaci symbolicznej nazywany jest kodem źródłowym programu. Asembler tłumaczy kod źródłowy programu na postać binarną zrozumiałą przez procesor.

W celu zapoznania się z pracą i zasadą działania sterownika mikroprocesorowego stworzono cztery kody źródłowe programów, które przedstawiają działanie poszczególnych bloków funkcjonalnych sterownika.

Trzy pierwsze programy wykorzysują jedynie funkcje asemblera i symulatora programu ECAL, czwarty, pokazuje także możliwości śledzenia wykonywanego programu.

Aby uruchomić przykładowe programy należy:

- dokonać asemblacji kodu źródłowego programu
- przesłać program wynikowy do symulatora pamięci EPROM
- umożliwić mikroprocesorowi sterownika wykonanie programu zapisanego w symulatorze pamięci EPROM.

W przypadku czwartego programu należy dodatkowo uruchomić opcję śledzenia.

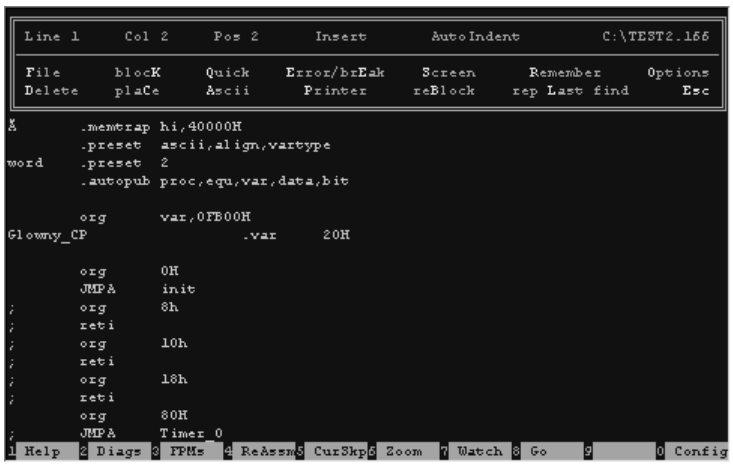
Wszystkie pliki potrzebne do przeprowadzenia ćwiczenia znajdują się w katalogu: C:\LAB_EAZC\PRZYKLAD

7.1. Edycja i asemblacja programu

Edycję oraz asemblację programu źródłowego wykonuje się z wykorzystaniem programu ECAL. Uruchomienie następuje bezpośrednio z linii poleceń systemu operacyjnego DOS. Po uruchomieniu pojawi się ekran pozwalający na wybór odpowiedniego kodu źródłowego programu (patrz rysunek nr 6).

Rysunek 6. Ekran wyboru kodu źródłowego programu

Pliki kodów źródłowych mają rozszerzenia **.166**. Za pomocą klawiszy kursorów należy wybrać odpowiedni program. Po podświetleniu i zatwierdzeniu wyboru (Enter), program wejdzie w funkcję edytora (patrz rysunek nr 7).

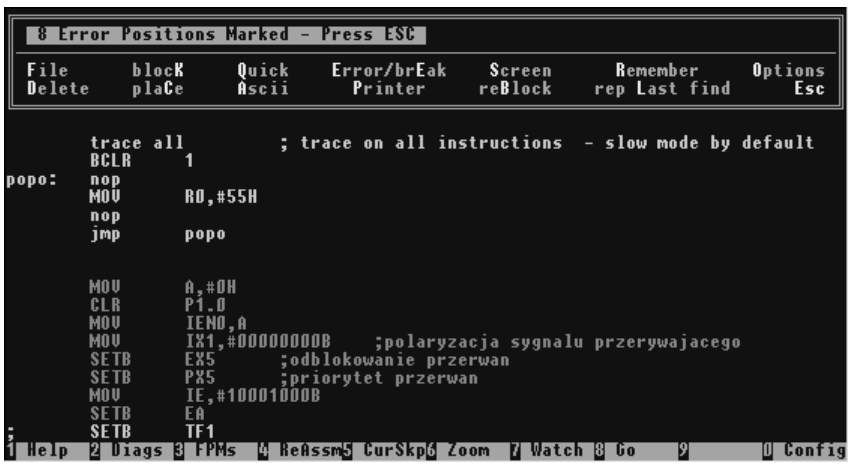


Rysunek 7. Ekran edycji pojawiający się po wybraniu programu

Podczas edycji dostępnych jest wiele opcji, przedstawionych w postaci górnego i dolnego paska narzędzi, z których często wykorzystywaną jest funkcja **DIAG** dostępna po naciśnięciu klawisza **F2**. Umożliwia ona (poprzez kolejne naciskanie klawisza **F2**)dostęp do: prostego kalkulatora, tablicy symboli pokazującej m.in. adresy wykorzystywanych zmiennych i procedur obliczone na podstawie asemblacji oraz kontekstową pomoc nt. dyrektyw wykorzystywanego asemblera, i mnemoników procesora. Wygląd ekranu z uruchomioną opcją **DIAG** przedstawia rysunek 8.

Rysunek 8. Ekran edycji po uruchomieniu opcji DIAG

Gdy otrzymamy ostateczną wersję programu, należy poddać go asemblacji. W tym celu będąc w oknie edycji należy nacisnąć klawisz **F4** lub klawisz **ESC.** Naciśnięcie klawisza **F4** spowoduje natychmiastową asemblecję, która w przypadku braku błędów nie będzie zgłaszała żadnych komunikatów. Jeżeli kod źródłowy będzie zawierał linie niezrozumiałe dla asemblera, zostaną one oznaczone na czerwono (patrz rysunek 9).



Rysunek 9. Ekran edycji po kompilacji i wystąpieniu błędów

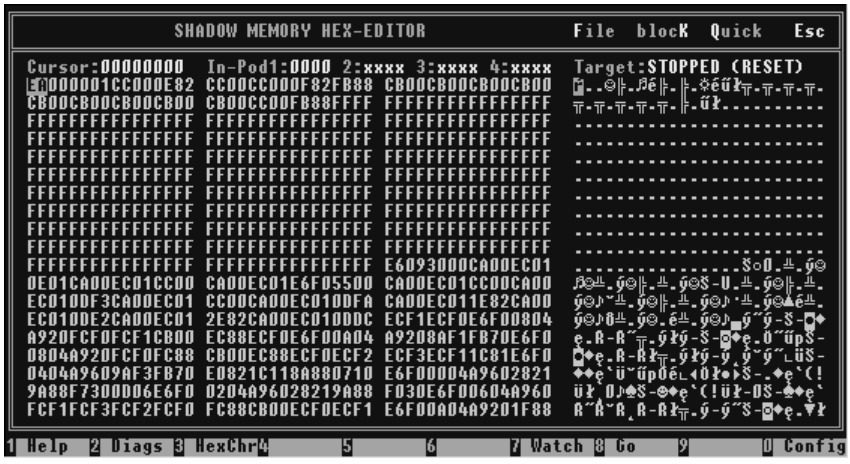
Asemblację można również przeprowadzić z poziomu ekran wyboru kodu źródłowego programu (partz rysunek 6), po wskazaniu zbioru do kompilacji oraz naciśnięciu klawisza **A** (pojawi się ekran przedstawiony na rysunku 10), a następnie **ENTER.**

Rysunek 10. Ekran asemblacji

W przypadku poprawnej asemblacji, u góry ekranu pojawi się znacznik pojawi się informacja o liczbie skompilowanych linii (patrz rysunke 11), wykorzystanych zmiennych oraz polecenie naciśnięcia klawisza **ESC**. W przypadku wystąpienia błędów program samoczynnie wejdzie do opcji edycji zbioru źródłowego i pokaże je oznaczone na czerwono (patrz rysunek 8).

Rysunek 11. Górna część ekranu programu ECAl po wykonaniu poprawnej asemblacji

W wyniku poprawnej asemblacji otrzymuje się zbiór z rozszerzeniem **.BIN**. Jest to plik zawierający kod wynikowy programu w formacie binarnym, zrozumiałym przez procesor sterownika mikroprocesorowego. Dodatkowo ten sam kod zostaje umieszczony w pamięci komputera PC i może zostać poprzez kartę emulatora/debugera ECAL przesłany taśmą do obu symulatorów pamięci "Ecal pod". Podgląd wartości heksadecymalnych skompilowanego kodu można zobaczyć z poziomu ekranu wyboru kodu źródłowego programu (patrz rysunek 6) op naciśnięciu klawisza **X**. Pojawi się wówczas okno pokazane na rysunku 12 zawierające wymienione informacje.



Rysunek 12. Ekran podglądu wartości heksadecymalnych skompilowanego kodu

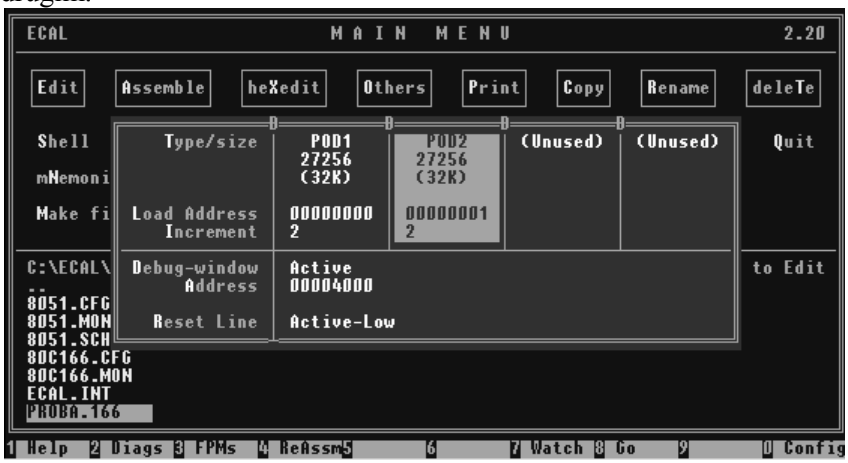
7.2. Przesłanie kodu do symulatorów pamięci EPROM oraz rozpoczęcie wykonywania programu

Mając po bezbłędnej asemblacji przygotowany kod wynikowy programu, można przystąpić do przesłania go do symulatorów pamięci EPROM sterownika. Następuje to z wykorzystaniem opcji symulatora pamięci EPROM, która uruchamiana jest w edycji lub oknie głównym programu ECAL przez naciśnięcie klawisza **F8** (na dolnym pasku zadań opcja ta jest oznaczona jako **8 GO**).

Funkcja symulatora pamięci EPROM wykorzystuje konfigurację sprzętową obu symulatorów sprzętowych o nazwach "ECAL pod 1" i "ECAL pod 2", którą można zobaczyć lub zmienić po naciśnięciu klawisza **F10** (patrz rysunek 13).

Funkcja wykorzystuje 2 urządzenia symulujące, ponieważ pracujący w jednostce centralnej sterownika mikroprocesor 80c166 jest szesnastobitowy (ma 16-to bitową szynę danych), a układ jednostki centralnej został przystosowany do wykorzystywania układów pamięci EPROM o orgamizacji 8 bitowej.

Przesyłanie danych oraz uruchamianie programu w sterowniku rozpoczyna się od przesłania kodu umieszczonego w pamięci komputera PC do karty ECAL, która jednocześnie wystawia sygnał "Reset", powodujący wstrzymanie cyklu rozkazowego procesor 80c166. Następnie kod programu przez łącze rónoległe w postaci wielożyłowej taśmy, przesyłany jest z karty do obu symulatorów. Na koniec karta zwalnia sygnał "Reset", powodując, że procesor zaczyna wykonywać program począwszy od adresu 0. Konfiguracja symulatorów pokazana na rysunku 13 pozwala na automatyczne rozdzielenie bajtów kodu, w taki sposób, że bajty o adresach pażystych są dostępne w jednym symulatorze, a o adresach nieparzystych w drugim.



Rysunek 13. Ekran podglądu wartości heksadecymalnych skompilowanego kodu

7.3. Śledzenie programu wykonywanego przez sterownik mikroprocesory

Po zwolnieniu sygnału Reset i uruchomieniu programu, zostaje również uruchomiony mechanizm śledzenia zwrotnych informacji trafiających z procesora 80c166 do symulatorów. Mechanizm ten pozwala na wizualizację stanu procesora (zawartość portów, rejestrów specjalnych, obszarów pamięci) oraz śledzenie i krokowe wykonywanie programu. Skonfigurowanie opcji śledzenia wykonywania programu i stanu procesora, polega na dodaniu w kodzie źródłowym programu:

- procedur przesyłających dane z/do symulatora, zawartych w pliku o nazwie "80c166.mon" (polecenie: **include 80C166.cfg**)
- definicji wyglądu okna śledzącego stan, zawartej w pliku o nazwie "80c166.mon " (polecenie **include 80C166.mon)**
- pułapek śledzących, przez naciśnięcie z poziomu edycji programu źródłowego, kombinacji klawiszy CTRL+E+I (info point) lub CTRL+E+B (break point) w wybranym miejscu programu. (Możliwe jest także wstawienie polecenia śledzenia wszystkich instrukcji programu przez wpisanie polecenia **trace all** powodującego, że wszystkie instrukcje położone poniżej tego polecenia zostaną automatycznie oznaczone jako - break point).

Rysunek 13. Wygląd ekranu okna śledzącego

Wyświetlenie okna śledzącego następuje po naciśnięciu klawisza **F7**. W oknie tym znajdują się cztery obszary:

- "Register watch" - obszar wizualizacji zawartości rejestrów specjalnych
- "Trace" - obszar pokazujący adres ostatnio wykonanej instrukcji
- "Breaks" - obszar pokazujący listę zdefiniowanych punktów break point (
- "Memory Watch" - obszar pozwalający na definicję adresów pamięci, których zawartość ma zostać pokazana.

W opcji śledzenia użytkownik ma poza tym możliwść:

- zmiany aktywnego obszaru okna, po naciśnięciu klawisza **F5**
- rozwinięcia aktywnego obszaru na cały ekran po naciśnięciu klawisza **F6**
- zmiany punktu śledzącego break point na info point - w oknie Breaks po naciśnięciu klawisza **F4**.

Na każdym poziomie pracy programu dostępna jest kontekstowa pomoc, po naciśnięciu klawisza **F1**.

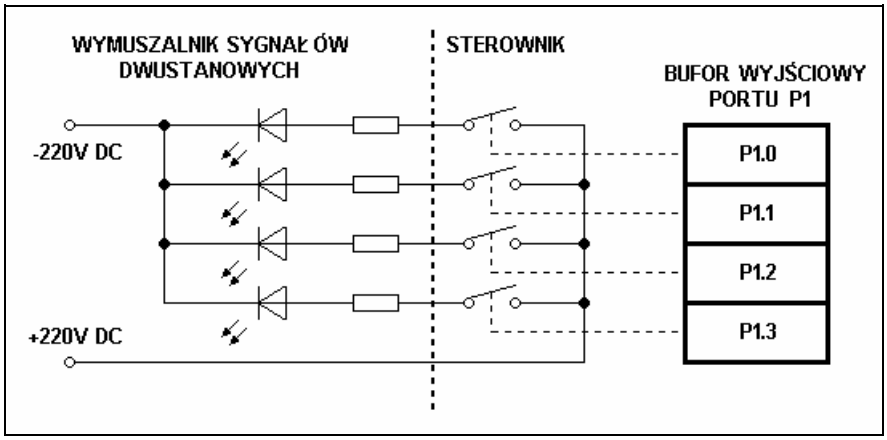
Na rysunku 13 pokazano wygląd ekranu okna śledzącego, a na rysunku 14 powiększaonego na cały ekran obszaru okna wizualizacji zawartości rejestrów specjalnych.

Rysunek 14. Powiększony na cały ekran obszar okna wizualizacji zawartości rejestrów specjalnych

7.4. Kody źródłowe przykładowych programów:

7.4.1. Program 1

Poniższy program służy do przedstawienia działania wyjść dwustanowych sterownika mikroprocesorowego. Działanie programu polega na tym, że kolejnym wyjściom dwustanowym sterownika przypisywany jest stan wysoki. Zmiana stanu wyjść jest sygnalizowana zapalaniem się kolejnych diod LED wejść dwustanowych wymuszalnika.



Rysunek 15. Schemat blokowy połączeń wykorzystywanych w programie 1

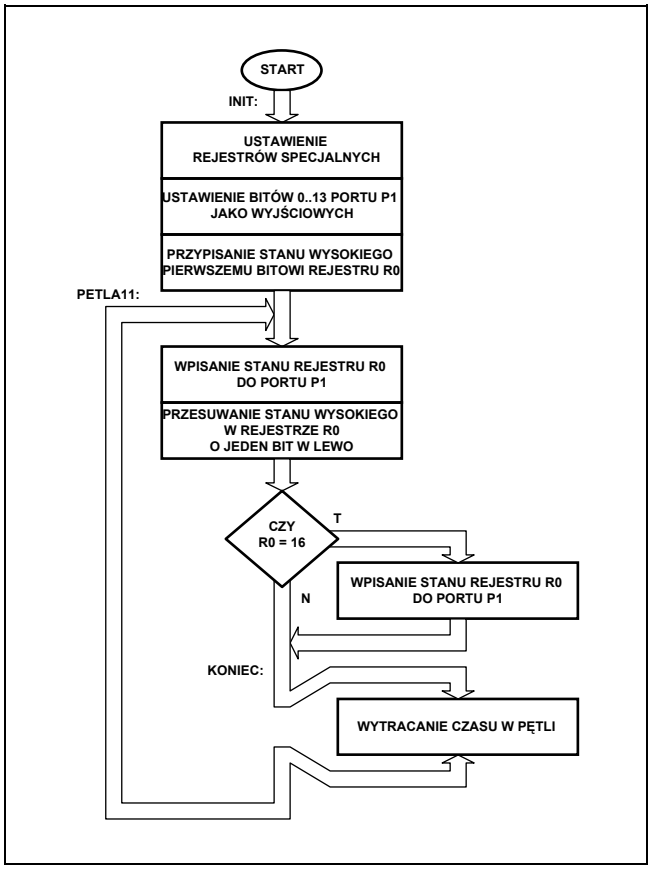
Na początku programu następuje deklaracja występujących zmiennych oraz rejestrów specjalnych, takich jak: SP – wskaźnik stosu, CP – wskaźnik kontekstu.

W dalszej części programu do rejestru R0 zapisywane są na przemian stan wysoki i stan niski. Następnie stany te są przepisywane do odpowiednich wyjść portu wyjściowego P1, co powoduje zadziałanie bądź nie wyjść dwustanowych sterownika (zależnie od tego, czy jest to stan wysoki czy niski).

Szybkość zmian stanu portu wyjściowego jest zależna od wprowadzonych opóźnień (patrz ramka). Dzięki zastosowanej pętli następuje cykliczne przypisywanie określonego stanu każdemu z wyjść dwustanowych sterownika mikroprocesorowego.

W ramach ćwiczenia należy:

- uruchomić program w wersji podstawowej i zaobserwować z jaką częstotliwością następują zmiany stanu wyjść dwustanowych sterownika
- dokonać zmian wartości opóźnień zastosowanych w programie poprzez zmianę wartości parametru “wait” i “opoz2” (patrz ramka) w celu zmiany częstotliwości zadziałania wyjść dwustanowych.



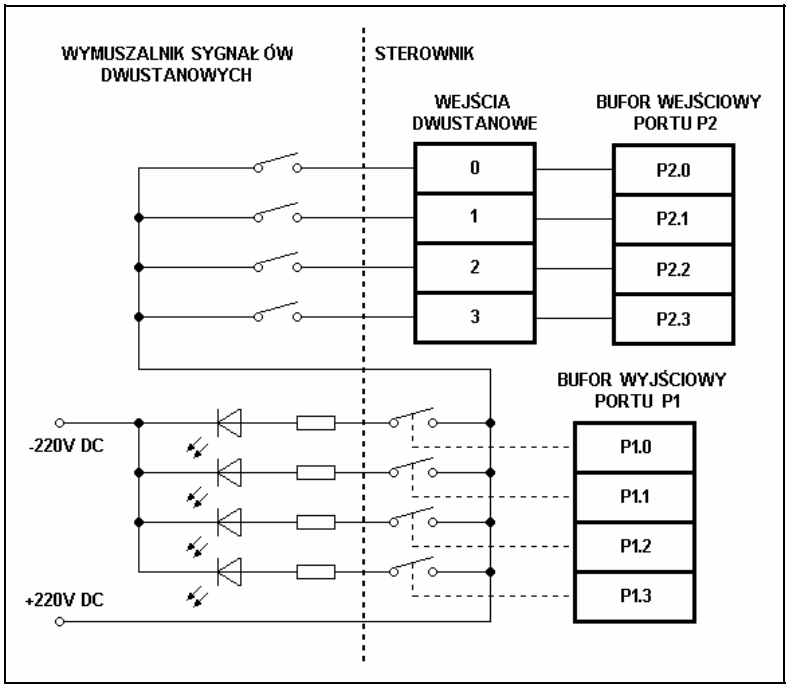
Rysunek 16. Algorytm działania programu 1

TEST1.166

```
            .preset         ascii,align,vartype
word        .preset         2
            .autopub    proc,equ,var,data,bit
Glowny_CP                   .var        20H

            org             0H
            JMPA            init

;-------------------------------------------------------------------;
;           Program główny
;-------------------------------------------------------------------;
            org             200H
init:       DISWDT                                      ;wyłączenie układu WATCHDOG
            EINIT                                       ;koniec inicjalizacji
            NOP
            OR              SYSCON,#0000000000101100B
;                                   5432109876543210
            MOV             cp,#Glowny_Cp
            MOV             sp,#Glowny_Cp
            MOV             stkun,#glowny_cp
            BSET            P3.12
            BSET            P3.13
            BCLR            P3.14
            BSET            DP3.12                      ;BHE/
            BSET            DP3.13                      ;WR/
            BSET            DP3.14                      ;EPROM
            AND             P1,   #1100000000000000B    ;P1.0 -P1.13 - wyjścia cyfrowe
            OR              DP1, #0011111111111111B     ;P1.14,P1.15 - wyjścia cyfrowe
;                                   5432109876543210
            nop
            mov             r0,#1                       ;przypisanie stanu wysokiego pierwszemu bitowi
                                                        ;rejestru R0
petla11:
            mov             p1,r0                       ;wpisanie do wyjścia P1 stanu rejestru R0
            rol             r0,#1                       ;
            cmp             r0,#16                      ;
            jmpr            cc_ne,koniec                ;
            mov             r0,#1                       ;
koniec:
            call            wait
            jmp             petla11                     ;skok do początku pętli


opoz1:      cmpd1           r2,#0                       ;pętla opóźniająca
            jmpr            cc_nz,opoz1
            cmpd1           r1,#0
            jmpr            cc_nz,opoz2
            ret
end.
```

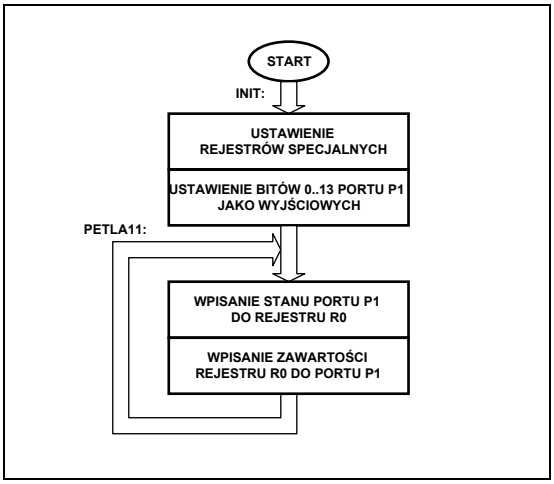
7.4.2. Program 2

Program ten służy do przedstawienia działania wejść dwustanowych sterownika mikroprocesorowego. Po uruchomieniu programu, sterownik mikroprocesorowy oczekuje na pojawienie się napięcia 220V DC na jego wejściach dwustanowych. Pojawienie się tego napięcia powoduje zmianę stanu wejść dwustanowych na wysoki a w następstwie zmianę stanu portu wejściowego jednostki centralnej sterownika. Algorytm programu powoduje przepisanie stanu wysokiego portu wejściowego do portu wyjściowego układu mikroprocesorowego. Następuje wówczas zadziałanie wyjść dwustanowych sterownika mikroprocesorowego, które sygnalizowane jest zapaleniem się diod LED na wejściach dwustanowych wymuszalnika. (Napięcie 220Vdc do polaryzacji wejść dwustanowych sterownika jest podawane z wymuszalnika sygnałów dwustanowych.)



Rysunek 17. Schemat blokowy połączeń wykorzystywanych w programie 2

Na początku programu następuje deklaracja występujących zmiennych oraz rejestrów specjalnych, takich jak: SP – wskaźnik stosu, CP – wskaźnik kontekstu. W dalszej części programu, stan wejść portu wejściowego P2 jest przepisywany do rejestru R0. Stan ten jest następnie przepisywany z rejestru R0 do odpowiednich wyjść binarnych portu wyjściowego P1, co powoduje zadziałanie bądź nie wyjść dwustanowych sterownika (zależnie od tego, czy jest to stan wysoki czy niski). Proces odczytu stanu wejść portu wejściowego oraz przepisywania tego stanu do wyjść dwustanowych wykonywany jest ciągle dzięki zastosowanej pętli.



Rysunek 18. Algorytm działania programu 2

W ramach ćwiczenia należy:

- uruchomić program
- sprawdzić, czy pobudzeniom poszczególnych wejść dwustanowych odpowiada zadziałanie odpowiednich wyjść dwustanowych sterownika mikroprocesorowego
- wprowadzić zmianę w programie powodującą inne przypisanie .wejść i wyjść sterownika mikroprocesorowego.

TEST2.166

```
        .preset     ascii,align,vartype
word    .preset     2
```

```
            .autopub    proc,equ,var,data,bit

Glowny_CP                   .var        20H

            org             0H
            JMPA            init

;-------------------------------------------------------------;
;           Program główny
;-------------------------------------------------------------;
            org             200H
init:       DISWDT                                      ;wyłączenie układu WATCHDOG
            EINIT                                       ;koniec inicjalizacji
            NOP
            OR              SYSCON,#0000000000101100B
;                                   5432109876543210
            MOV             cp,#Glowny_Cp
            MOV             sp,#Glowny_Cp
            MOV             stkun,#glowny_cp
            BSET            P3.12
            BSET            P3.13
            BCLR            P3.14
            BSET            DP3.12                      ;BHE/
            BSET            DP3.13                      ;WR/
            BSET            DP3.14                      ;EPROM
            AND             P1,  #110000000000000B      ;P1.0 -P1.13 - wyjścia cyfrowe
            OR              DP1,#0011111111111111B      ;P1.14,P1.15 - wyjścia cyfrowe
;                                   5432109876543210
            nop

petla11:
            mov             r0,p2                       ;wczytanie stanu 16 bitów wejścia P2 do
                                                        ;rejestru R0
            mov             p1,r0                       ;przepisanie 16-bitowej zawartości rejestru R0
                                                        ;do 16 bitów wyjścia P1
            jmp             petla11                     ;skok do początku pętli
end.
```

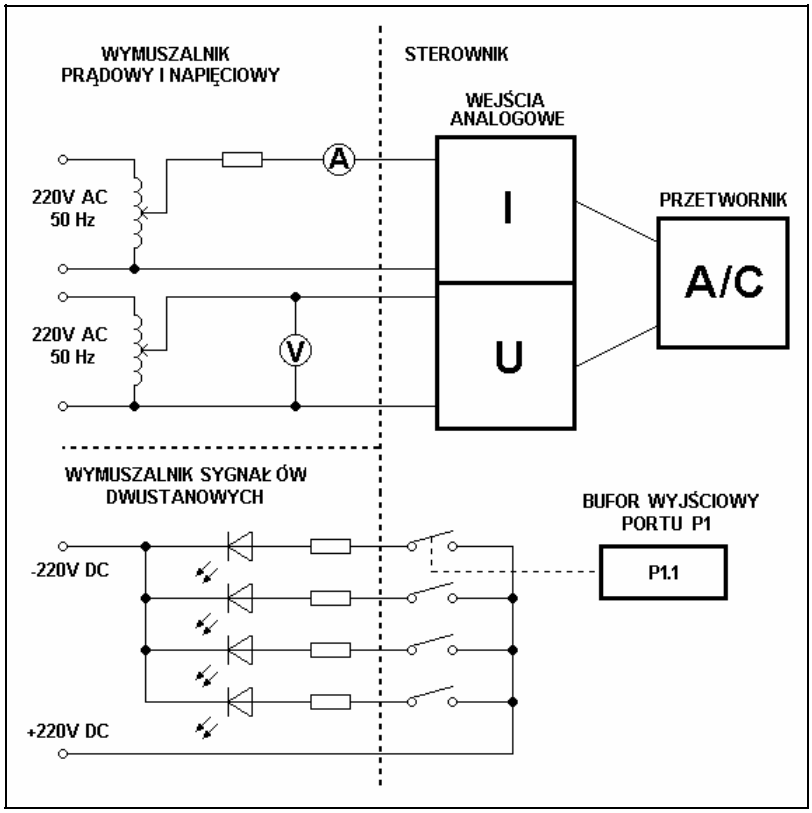
7.3.3. Program 3

Program ten służy do przedstawienia działania wejść analogowych sterownika mikroprocesorowego. Za pomocą tego programu można pokazać zachowanie się wejść prądowych i napięciowcyh sterownika. Wyboru rodzaju wejścia dokonuje się przez uaktywnienie określonej linii podprogramu **POM.166**. Linia poprzedzona znakiem " **;** " jest linią nieaktywną i jest traktowana jako komentarz (patrz ramka w **POM.166**).

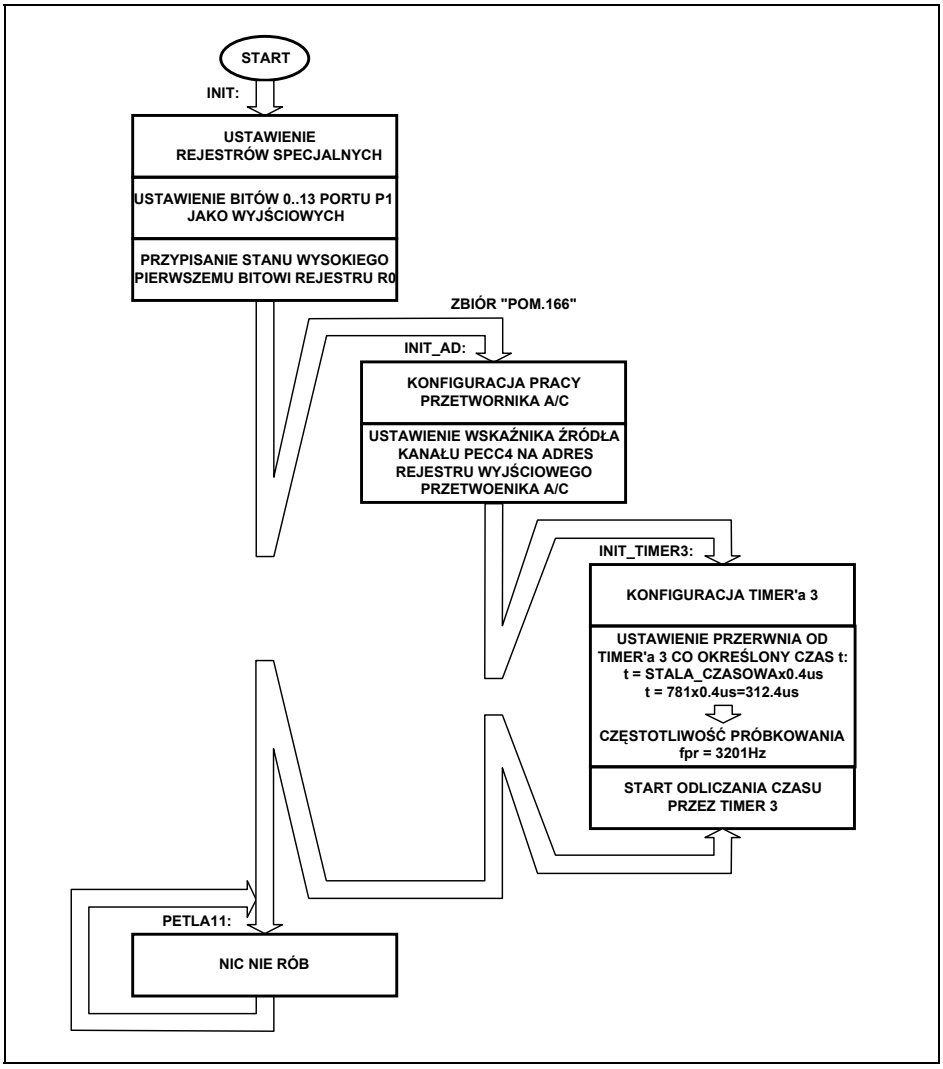
Opracowany program pomiarowo-zabezpieczeniowy powoduje zmianę stanu wyjść dwustanowych sterownika na skutek przekroczenia wartości progowej napięcia lub prądu ustawionych na ok. 20V dla wejść napięciowych i ok. 1A dla wejść prądowych. Sterownik wykonujący ten program pełni więc funkcję bardzo prostego zabezpieczenia nadnapięciowego bądź nadprądowego. Napięcia i prądy mogą być wymuszane za pomocą autotransformatorów zainstalowanych w wymuszalniku. Przekroczenie określonego progu powoduje zadziałanie wyjść dwustanowych sterownika, co jest sygnalizowane zapaleniem się diody LED na wejściu dwustanowym wymuszalnika.

Program składa się z dwóch części:

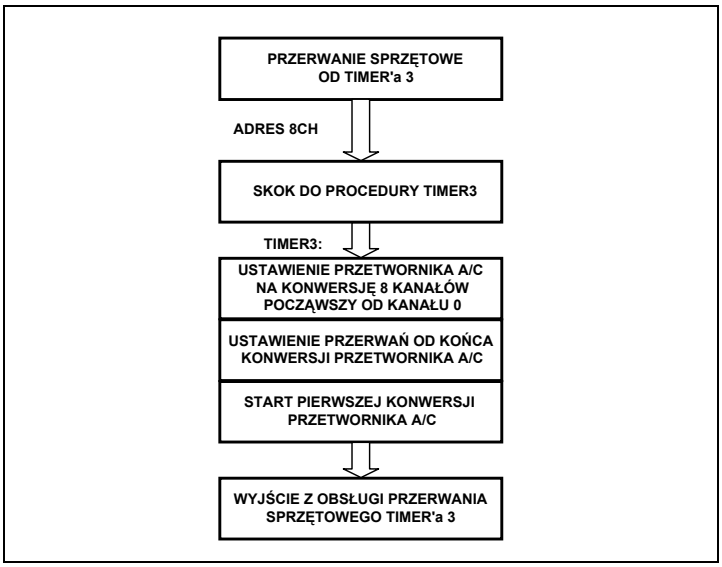
- zasadniczego programu (**TEST.166**), zawierającego deklaracje zmiennych, ustawienie wektorów przerwań, procedury inicjalizujące zbieranie pomiarów z przetwornika analogowo–cyfrowego oraz odblokowanie przerwań i
- podprogramu (**POM.166**) określającego:
    - sposób działania przetwornika analogowo–cyfrowego oraz jego przerwania
    - okres próbkowania - czas między kolejnymi seriami konwersji A/D sygnałów analogowych z kanałów pomiarów prądów i napięć
    - wartości progów zadziałania realizowanej funkcji nadprądowej bądź nadnapięciowej (patrz ramka)
    - ustawienia portów wejściowych i wyjściowych sterownika.



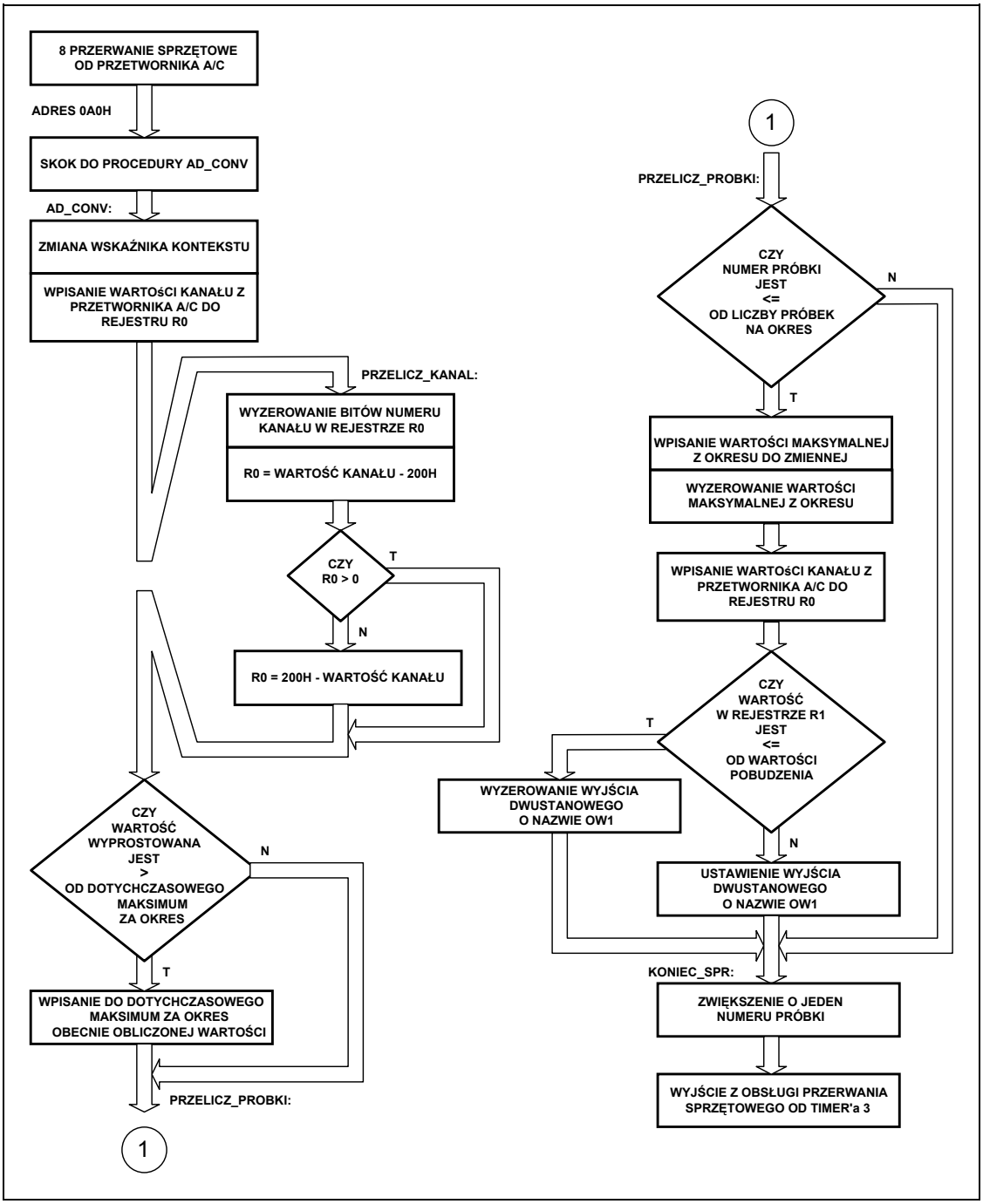
Rysunek 19.Schemat blokowy połączeń wykorzystywanych w programie 3



Rysunek 20. Algorytm działania programu głównego (**TEST.166**)



Rysunek 21. Procedura realizowana w obsłudze przerwania TIMER'a 3



Rysunek 22. Procedura realizowana w obsłudze 8 przerwania przetwornika A/C (część 1 i 2)

W ramach ćwiczenia należy:

- uruchomić program w wersji dla wejść prądowych i wejść napięciowych
- dokonać sprawdzenia czy sterownik działa za każdym razem przy tych samych progach napięcia ustawionych w programie (ok. 30V)
- dokonać zmiany kanału analogowego na prądowy (np. kanał I1_AN) oraz sprawdzić poziom pobudzania
- dokonać zmian wartości progowych zadziałania wejść analogowych sterownika poprzez zmianę wartości parametru "wart_pob" (patrz ramka w podprogramie **POM.166**) nie przekraczając jednak wartości maksymalnych, jakimi mogą zostać trwale obciążone wejścia analogowe tj. 4A dla wejść prądowych i 200Vdla wejść napięciowych. Do regulacji napięcia i prądu należy wykorzystać układy wymuszalnik.

TEST.166

```
            .memtrap  hi,40000H
            .preset             ascii,align,vartype
word        .preset             2
            .autopub  proc,equ,var,data,bit

            org                 var,0FB00H
Glowny_CP                       .var        20H

            org                 0H
            JMPA                init
            org                 80H
            org                 84H
            org                 8CH
            JMPA                Timer_3
            org                 90H
            org                 98H
            org                 0A0H
            JMPA                Ad_Conv
            org                 0A4H
            JMPA                Ad_Error
            org                 0A8H
            org                 0ACH
            org                 0B0H
;-------------------------------------------------------------------;
            .org                200H
init:       DISWDT                                      ;wyłączenie układu WATCHDOG
            EINIT                                       ;koniec inicjalizacji
            NOP
            OR                  SYSCON,#0000000000101100B
;                                       5432109876543210
            MOV                 cp,#glowny_cp
            MOV                 sp,#glowny_cp
            MOV                 stkun,#glowny_cp
            BSET                P3.12
            BSET                P3.13
            BCLR                P3.14
            BSET                DP3.12                  ;BHE/
            BSET                DP3.13                  ;WR/
            BSET                DP3.14                  ;EPROM
            AND                 P1,   #110000000000000B ;P1.0 -P1.13 - wyjścia cyfrowe
            OR                  DP1, #0011111111111111B ;P1.14,P1.15 - wyjścia cyfrowe
;                                       5432109876543210
            MOVB                RL0,#0
            CALL                INIT_AD
            BSET                IEN                     ;odblokowanie przerwań

pętla:      nop
            JMPR                pętla                   ;nieskończona pętla

            .include    POM.166

            end
```

POM.166

```
;
;definicje wejść cyfrowych
;
WEC0    .reg        P2.0        ;Pol_wyl
WEC1    .reg        P2.1        ;Pol_odl_szyn
WEC2    .reg        P2.2        ;Pol_odl_lin
WEC3    .reg        P2.3        ;Pol_uziem
WEC4    .reg        P2.4        ;Kontr_nazbr
WEC5    .reg        P2.5        ;Zal_recz
WEC6    .reg        P2.6        ;Wyl_recz
                                ;Zal_telest
                                ;Wyl_telest
                                ;Zabl_zab
                                ;P0_sygn
                                ;Blok_sygn
                                ;Blok_SPZ
```

```
                                                        ;SCO
;
;definicje wyjść cyfrowych
;
WYC0    .reg            P1.0                            ;ZW
WYC2    .reg            P1.1                            ;OW1
WYC3    .reg            P1.2                            ; OW2
WYC4    .reg            P1.3                            ; ZS_LRW
WYC5    .reg            P1.4                            ; Odstaw_zab
WYC6    .reg            P1.5                            ; Rozbr_Nap
                                                        ; AW
                                                        ; Uszk_zas
                                                        ; Dzial_P0_wyj
OW1     .reg            P1.1
;
;definicje wejść analogowych
;
        org             var,0FB20H
Dane_AC         .var    2                               ;Dane z przetwornika AC
kanal0                  .equ    Dane_AC                 ;kanał U1_AN
kanal1                  .var    2                       ;kanał U2_AN
kanal2                  .var    2                       ;kanał U3_AN
kanal3                  .var    2                       ;kanał U2_AN
kanal4                  .var    2                       ;kanał I1_AN
kanal5                  .var    2                       ;kanał I2_AN
kanal6                  .var    2                       ;kanał I1_AN
kanal7                  .var    2                       ;kanał I3_AN
;Ir      .reg            kanal0
;Is      .reg            kanal1
;It      .reg            kanal2
;I0      .reg            kanal3
;U0      .reg            kanal4
;Uf      .reg            kanal5
                        .var    2                       ;R9 - dla oscyloskopu
numer_probki            .var    2
kanal0_m                .var    2                       ;wartości maksymalne obliczane
kanal1_m                .var    2
kanal2_m                .var    2
kanal3_m                .var    2
kanal4_m                .var    2
kanal5_m                .var    2
kanal6_m                .var    2
kanal0_ok       .var    2                       ;wartości maksymalne aktualne
kanal1_ok       .var    2
kanal2_ok       .var    2
kanal3_ok       .var    2
kanal4_ok       .var    2
kanal5_ok       .var    2
kanal6_ok       .var    2
Wolny_obszar            .var    2

wart_pob        .equ    100

;-------------------------------------------------------------------------;
;
;       AD,Timer3 - zbieranie pomiarów analogowych i cyfrowych
;
INIT_AD:
        MOV             ADCIC,#01111100B                ;Konfiguracja przerwań od ADC.
;Nr bitu                         76543210
;                                    ***                number of PEC channel.
;                                 ***                   PEC service.
;                                *                      Intrrupt enable.
;                               *                       Interrupt request.
        MOV             ADEIC,#01111001B
        MOV             PECC1,#0
        MOV             ADCON,#0000000000100111B;Konfiguracja AD.
;Nr bitu                         5432109876543210
;                                        ****           Number of analog input chanell.
;                                    **                 Conversion mode selection.
;                                    00                 Single Channel Conversion.
;                                    01                 Single Channel Continous Conversion.
;                                    10                 Auto Scan Conversion.
;                                    11                 Auto Scan  Continous Convesion.
```

```
;                                       *           Don't care.
;                                      *            ADC Start Bit
;                                     *                     ADC Busy Flag.
;                                    0                      No conversion in progress.
;                                    1                      Conversion in progress.
;                                 *******                   Don't care.
        MOV             R0,#ADDAT
        MOV             SRCP4,R0                                    ;Source Pointer PEC4
        CALL            INIT_TIMER3                         ;Inicjalizacja timera czasu miedzy
                                                            ;pomiarami.
        RET
;-------------------------------------------------------------------;
INIT_TIMER3:
        MOV             T3CON,#0000000010000000B
;Nr bitu                         5432109876543210
;                                          ***                      T3I input selection 0.4us resolution.
;                                        **                 T3M timer 3 mode control.
;                                       *                   don't care.
;                                      *                    T3R run bit.
;                                    ***                    T3OE,T3UDE,T3UD count down.
;                                   *                               T3OTL timer output toggle latch.
        MOV             T2CON,#0000000000100111B
;Nr bitu                         5432109876543210
;                                          ***                      T4I.
;                                        ***                T4M.
;                                       *                   T4R run bit.
;                                      *                    T4UD up/down count bit.
        MOV             T2,STALA_CZASOWA_AD
        MOV             T3,STALA_CZASOWA_AD
        MOV             T3IC,#01111011B
;Nr bitu                       76543210
;                                 ***                       Number of PEC channel.
;                                ***                 PEC service.
;                               *                           Intrrupt enable.
;                              *                            Run bit.
        MOV             PECC3,#0
        BSET            T3R
        RET
;-------------------------------------------------------------------;
TIMER_3:
        PUSH            R0
        MOV             PECC4,#01000001000B                 ;Konfiguracja PEC Channel 4.
;Nr bitu                         09876543210
;                                      ********             PEC Transfer Counter.
;                                      11111111             Count value not decremented.
;                                       1H-FEH              Count value decremented after each
;                                                           transfer.
;                                      00000000             Interrupt generated.
;                                     *                     Transfer select bit.
;                                     0                     Word Transfer.
;                                   1                       Byte Transfer.
;                                  **                       Increment Control Field.
;                                  00                       Increment no pointer.
;                                  01                       Increment destination pointer.
;                                  10                       Increment source pointer.
;                                  11                       Reserved.
        MOV             R0,#DANE_AC
        MOV             DSTP4,R0                                    ;Destination Pointer PEC4.
        BSET            ADST
        POP             R0
        RETI
;-------------------------------------------------------------------;
;       obsługa przerwania ADC
;
Przelicz_Kanal:
        AND             R0,#07FFH
        MOV             R1,R0
        SUB             R1,#200H                    ;Prostowanie dwupołówkowe.
        JMPR            CC_NC,DOD_ZNAK_SYGN
        MOV             R1,#200H
        SUB             R1,R0

Dod_Znak_Sygn:
        RET
```

```
AD_CONV:
        SCXT    CP,#Wolny_obszar        ;wolny obszar dla rejestrów R0-R15
        nop
        MOV     R0,kanal0               ;kanał U1_AN
;       MOV     R0,kanal1               ;kanał U2_AN
;       MOV     R0,kanal2               ;kanał U3_AN
;       MOV     R0,kanal3               ;kanał U2_AN
;       MOV     R0,kanal4               ;kanał I1_AN
;       MOV     R0,kanal5               ;kanał I2_AN
;       MOV     R0,kanal6               ;kanał I1_AN
;       MOV     R0,kanal7               ;kanał I3_AN
        CALL    Przelicz_Kanal
        CMP     R1,kanal2_m
        JMPR    CC_ULT,Przelicz_Probki
        MOV     kanal2_m,R1

Przelicz_Probki:
        MOV     R0,Numer_Probki
        CMPI1   R0,Licz_Pr_Na_Okr       ;porównanie numeru próbki z liczbą próbek
                                        ;w okresie
        JMPR    CC_ULT,Zwieksz_Numer_Pr
        MOV     R1,kanal2_m
        MOV     kanal2_ok,R1            ;wpisanie wartości maksymalnej z okresu
                                        ;do zmiennej
        MOV     R0,#0
        MOV     kanal2_m,R0             ;zerowanie wartości maksimum z okresu
                                        ;dla następnego okresu

;Przek nad. bezzwłoczny
        CMP     R1,#wart_pob            ;sprawdzenie pobudzenia dla >
        JMPR    CC_ULT,Brak_Pob_I
        BSET    OW1
        JMPR    Koniec_Spr

Brak_Pob_I:
        BCLR    OW1

Koniec_Spr:

Zwieksz_Numer_Pr:
        MOV     Numer_Probki,R0         ;wpisanie do zmiennej nowego numeru
                                        ;próbki
Kon_AD:
        POP     CP
        NOP
        RETI
;
; przerwanie wykonane w przypadku błędu przetwornika A/D
;
Ad_Error:
        BCLR    ADEIR
        RETI
        .org    4000H

Licz_Pr_Na_Okr    .dw   64
Stala_czasowa_ad  .dw   781
```

7.3.4. Program 4

Program ten służy do zapoznania się z trybem śledzenia stanu procesora i wykonywania programu za pomocą systemu ECAL. Jego działanie i budowa jest funkcjonalnie identyczne z programem nr 2, przy czym w celu umożliwienia wizualizacji w programie tym dodano kilka linii wprowadzających podprogramy śledzenia do kodu programu (zbiór 80c166.mon), definicję wyglądu okna śledzącego (zbiór 80c166.cfg) oraz instrukcję śledzenia każdego polecenia - trace all.

W ramach ćwiczenia należy:

- uruchomić program
- sprawdzić, czy pobudzeniom poszczególnych wejść dwustanowych odpowiada zadziałanie odpowiednich wyjść dwustanowych sterownika mikroprocesorowego
- sprawdzić działanie opcji śledzenia w oknie śledzenia oraz oknie edycji
- dokonać prób zmiany zawartości rejestrów specjalnych oraz wpływu tej czynności na działanie programu

TEST41.166

```
            .preset             ascii,align,vartype
word        .preset             2
            .autopub    proc,equ,var,data,bit
            include 80C166.cfg

Glowny_CP                       .var        20H

            org                 0H
            JMPA                init

;---------------------------------------------------------------------;
;           Program główny
;---------------------------------------------------------------------;
            org                 200H
init:       DISWDT                                          ;wyłączenie układu WATCHDOG
            EINIT                                           ;koniec inicjalizacji
            NOP
            OR                  SYSCON,#0000000000101100B
;                                       5432109876543210
            MOV                 cp,#Glowny_Cp
            MOV                 sp,#Glowny_Cp
            MOV                 stkun,#glowny_cp
            BSET                P3.12
            BSET                P3.13
            BCLR                P3.14
            BSET                DP3.12                      ;BHE/
            BSET                DP3.13                      ;WR/
            BSET                DP3.14                      ;EPROM
            trace all
            AND                 P1,  #110000000000000B      ;P1.0 -P1.13 - wyjścia cyfrowe
            OR                  DP1,#0011111111111111B      ;P1.14,P1.15 - wyjścia cyfrowe
;                                       5432109876543210
            nop

petla11:
            mov                 r0,p2                       ;wczytanie stanu 16 bitów wejścia P2 do
                                                            ;rejestru R0
            mov                 p1,r0                       ;przepisanie 16-bitowej zawartości rejestru R0
                                                            ;do 16 bitów wyjścia P1
            jmp                 petla11                     ;skok do początku pętli

            include 80C166.mon

end.
```

7.3.5. Program 5

Program ten służy do zapoznania się z trybem śledzenia stanu procesora i wykonywania programu za pomocą systemu ECAL. Jego działanie i budowa jest funkcjonalnie identyczne z programem nr 1, przy czym w celu umożliwienia wizualizacji w programie tym dodano kilka linii wprowadzających podprogramy śledzenia do kodu programu (zbiór 80c166.mon), definicję wyglądu okna śledzącego (zbiór 80c166.cfg) oraz instrukcję śledzenia każdego polecenia - trace all.

W ramach ćwiczenia należy:

- uruchomić program
- sprawdzić, poprawne działanie programu - cykliczne zapalanie kolejnych wyjść dwustanowych
- odpowiedzieć na pytanie: czy w przypadku śledzenia potrzebna jest pętla opóźniająca o nazwie "wait" ?
- sprawdzić działanie opcji śledzenia w oknie śledzenia oraz oknie edycji
- dokonać prób zmiany zawartości rejestrów specjalnych oraz wpływu tej czynności na działanie programu

TEST42.166

```
            .preset         ascii,align,vartype
word        .preset         2
            .autopub    proc,equ,var,data,bit

            include 80C166.cfg

Glowny_CP                   .var        20H

            org             0H
            JMPA            init

;---------------------------------------------------------------;
;           Program główny
;---------------------------------------------------------------;
            org             200H
init:       DISWDT                                      ;wyłączenie układu WATCHDOG
            EINIT                                       ;koniec inicjalizacji
            NOP
            OR              SYSCON,#0000000000101100B
;                                   5432109876543210
            MOV             cp,#Glowny_Cp
            MOV             sp,#Glowny_Cp
            MOV             stkun,#glowny_cp
            BSET            P3.12
            BSET            P3.13
            BCLR            P3.14
            BSET            DP3.12                      ;BHE/
            BSET            DP3.13                      ;WR/
            BSET            DP3.14                      ;EPROM
            AND             P1,   #110000000000000B     ;P1.0 -P1.13 - wyjścia cyfrowe
            OR              DP1, #0011111111111111B     ;P1.14,P1.15 - wyjścia cyfrowe
;                                   5432109876543210
            nop
            mov             r0,#1                       ;przypisanie stanu wysokiego pierwszemu bitowi
                                                        ;rejestru R0
            trace all
petla11:
            mov             p1,r0                       ;wczytanie stanu rejestru R0 do wyjścia P1
            rol             r0,#1                       ;
            cmp             r0,#16                      ;
            jmpr            cc_ne,koniec                ;
            mov             r0,#1                       ;
koniec:
            call            wait
            jmp             petla11                     ;skok do początku pętli


opoz1:      cmpd1           r2,#0
            jmpr            cc_nz,opoz1
            cmpd1           r1,#0
            jmpr            cc_nz,opoz2
            ret

            include 80C166.mon

end.
```

7.3.6. Rejestrator zakłóceń.

Rejestracja zakłoceń jest niezbędna do właściwej analizy poprawności działania EAZ. Sygnałami pobudzającymi działanie rejestratora są prądy i napięcia których wartości zmieniają się na skutek zakłóceń w urządzeniach elektrycznych. Rejestratory zapisują kształty sygnałów analogowych wielkości elektrycznych oraz stany sygnałow dwustanowych reprezentujące decycje i polecenia. Ze względu na fakt rejestrowania wartości chwilowych wspomnianych sygnałów zapis nie jest porowadzony ciągle, a rozpoczyna się dopiero w przypadku wystąpienia warunku startu rejestracji. Co powinno być tożsame z zaistnieniem zakłócenia w monitorowanym urządzeniu elektrycznym.

Każda poprawna rejestracja przedstawia co najmniej trzy stany urządzenia oraz jego automatyki: stan bezpośrednio poprzedzający zakłócenie, stan zakłocenia oraz stan po zakłóceniu.

Program rejestratora zakłóceń jest przykładem rozbudowanego programu prezentującego w plikach wartości 8 przebiegów analogowych. Podobnie jak w poprzednich ćwiczeniach przebiegi analogowe wymuszane są przez wymuszalnik będący na wyposarzeniu stanowiska laboratyjnego.
Program rejestratora zakłóceń różni się znacznie od programów uruchamianych w poprzednich ćwiczeniach. Program rejestrator a zakłoceń składa się z dwóch części:

- pierwsza znajdująca się w podkatakogu \PC Jest uruchamiana na komputerze PC i służy do wymiany danych z rejestratorem oraz wizualizacji pomiarów. Program ten uruchamia się za pomocą pliku *w.exe* Opis obsługi samego programu znajduje sięw skrypcie „Laboratorium cyfrowej Elektroenergrtycznej Automatyki Zabezpieczeniowej" w ćwiczeniu:
  nr.18-Mikroprocesorowy rejestrtor zakłóceń firmy AMEPOL.
- druga uruchamiana jest w sterowniku. Program uruchamiany w sterowniku ze względu na swe rozmiary podzielony został na kilka oddzielnych plików, co miało ułatwić pisanie algorytmu i jego późniejsze poprawki. Program ten składa się z następujących plików:
  - ad.166
  - init.166
  - 0.166
  - inst.166
  - p9.166
  - ser.166
  - zap_od.166

W celu kompilacji programu złożonego z wielu plików niezbędne jest użycie opcji *make.file*, która musi wskazywać plik zawierający program główny (w naszy przypadku P9.166).

Aby możliwe było poprawne działanie rejestratora, konieczne było uruchomienie komunikacji pomiędzy sterownikiem a komputerem PC. Komunikacja ta zrealizowana została za pomocą łącza RS 232. Obsługa tego łącza musiała zostać zaimplementowana zarówno w programie umieszczonym w komputerze PC jak i w sterowniku.

Uruchomiony w czasie tego ćwiczenia rejestrator zakłóceń jest modyfikacją urządzenia będącego na wyposażeniu stanowiska z Cyfrowej EAZ. pt: „Mikroprocesorowy rejestrator zakłóceń firmy AMEPOL" (nr 18) i posiada identyczny program komputera nadrzędnego (umieszczonego w komputerze PC, natomiast modyfikacja samego sterownik polega na na tym, że rejestracja zakłóceń odbywa się tylko po przekroczeniu wielkości analogowych (prąd powyżej 2A i napięcia powyżej 30V). Rejestracje zapisywane są na dysku komputera nadrzędnego w postaci plików: *.rej . W plikach tych można wyodrębnić próbki z poszczególnych kanałów.

W ramach ćwiczenia należy:
- skompilować i uruchomić program sterownika
- uruchomić program koputera nadrzędnego
- wykonać rejestrację dla pobudzenia od kanałów analogowych prądowego i napięciowego
- zmienić progi pobudzenia na wartości podane przez prowadzącego i wykonać rejestrację
- obejżeć w edytorze *hexa* dowolny plik rejestracji *.rej i znaleźć w nim próbki poszczególnych kanałów.