MINISTERSTWO EDUKACJI
i NAUKI

Rafał Nowak

Analizowanie działania układów mikroprocesorowych 311[50].O1.06

Poradnik dla ucznia

Wydawca
Instytut Technologii Eksploatacji – Państwowy Instytut Badawczy
Radom 2005

„Projekt współfinansowany ze środków Europejskiego Funduszu Społecznego”

Recenzenci:
mgr inż. Henryk Krystkowiak
mgr inż. Bogdan Chmieliński

Opracowanie redakcyjne:
mgr inż. Katarzyna Maćkowska

Konsultacja:
dr inż. Janusz Figurski

Korekta:
mgr Joanna Iwanowska

Poradnik stanowi obudowę dydaktyczną programu jednostki modułowej 311[50].O1.06. Analizowanie działania układów mikroprocesorowych zawartego w modułowym programie nauczania dla zawodu technik mechatronik.

Wydawca
Instytut Technologii Eksploatacji – Państwowy Instytut Badawczy, Radom 2005

„Projekt współfinansowany ze środków Europejskiego Funduszu Społecznego"

SPIS TREŚCI

1. WPROWADZENIE

Poradnik będzie Ci pomocny w przyswajaniu wiedzy z zakresu budowy i zasady działania podstawowych mikroprocesorów, ich zastosowania, współpracy z powszechnie stosowanymi urządzeniami peryferyjnymi. Będzie on również przydatny w przyswajaniu wiadomości z zakresu podstaw programowania mikroprocesorów.

W poradniku zamieszczono:

- wymagania wstępne, czyli spis umiejętności, jakie musisz posiadać na wstępie kursu dotyczącego analizowania działania układów mikroprocesorowych,
- cele kształcenia będące spisem umiejętności jakie powinieneś nabyć w wyniku procesu kształcenia,
- materiał nauczania, zbiór wiadomości teoretycznych, będących niezbędnym minimum pozwalającym osiągnąć cele kształcenia,
- zestaw pytań – dający Ci możliwość sprawdzenia opanowanych wiadomości
- ćwiczenia pomocne będą w weryfikowaniu Twojej praktycznej oraz teoretycznej wiedzy,
- sprawdzian osiągnięć, przykładowy zestaw pytań testowych; zaliczenie testu może utwierdzić Cię w przekonaniu, iż zadowalająco opanowałeś materiał tej jednostki modułowej,
- literaturę uzupełniającą.

W celu prawidłowego wykorzystania przeznaczonego dla Ciebie poradnika powinieneś:

- zapoznać się z odpowiednią częścią materiału nauczania,
- przeanalizować zestaw ćwiczeń przeznaczonych do wykonania,
- opracować odpowiedzi do zamieszczonych pytań sprawdzających,
- przeanalizować zamieszczony zestaw pytań testowych.

2.WYMAGANIA WSTĘPNE

Przystępując do realizacji programu jednostki modułowej powinieneś umieć:
- opisywać budowę i działanie podstawowych elementów elektronicznych,
- czytać proste schematy elektryczne,
- stosować algebrę Boole'a,
- zapisywać liczby w różnych systemach,
- definiować pojęcia: stan wysoki, stan niski,
- omawiać działanie cyfrowych układów kombinacyjnych,
- omawiać działanie cyfrowych układów sekwencyjnych,
- rozróżniać technologie wykonywania układów cyfrowych,
- tworzyć proste algorytmy.

3. CELE KSZTAŁCENIA

W wyniku realizacji programu jednostki modułowej powinieneś umieć:

- omówić budowę typowych mikroprocesorów,
- rozpoznać poszczególne elementy wchodzące w skład systemu mikroprocesorowego,
- sklasyfikować mikroprocesory ze względu na ich budowę i zastosowanie,
- napisać proste programy w wybranym języku programowania, służące do przesyłu danych w systemie mikroprocesorowym,
- napisać proste programy w wybranym języku programowania, służące do wykonywania podstawowych operacji matematycznych,
- wykonać czynności związane z procesem asemblacji i ładowania programu do pamięci procesora.

4. MATERIAŁ NAUCZANIA

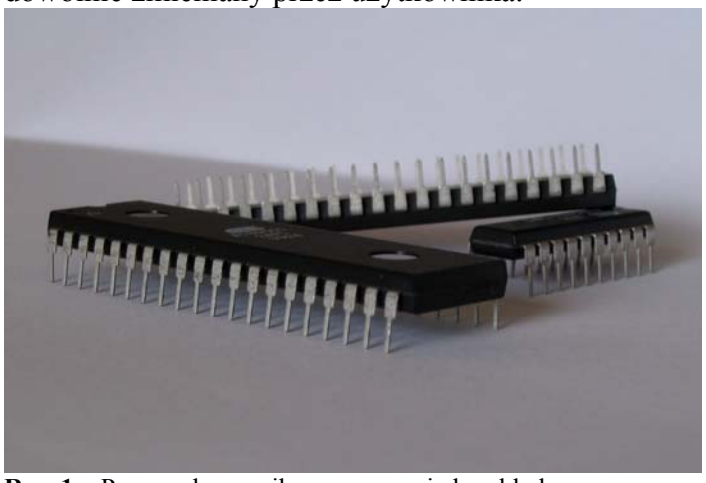
4.1. Mikroprocesory – struktura i charakterystyka

4.1.1. Materiał nauczania

Mikroprocesory – definicje podstawowe

Mikroprocesor (procesor), (rys.1) jest to układ scalony o wielkiej skali integracji przetwarzający sygnały cyfrowe, którego funkcjonowanie jest w pełni sterowane pobieranymi z pamięci wewnętrznej lub zewnętrznej rozkazami (poleceniami wykonywanymi przez mikroprocesor).

Ogół rozkazów, przetwarzanych przez mikroprocesor, napisany przez człowieka w odpowiedniej kolejności oraz w odpowiedni sposób nazywamy programem. Znajduje się on w pamięci programu i jest wykonywany przez procesor, a co najważniejsze, może być dowolnie zmieniany przez użytkownika.



Rys. 1. Przemysłowe mikroprocesory jednoukładowe

Strukturę wewnętrzną (architekturę) mikroprocesora przedstawiono na rys. 2.
Sercem systemu jest jednostka arytmetyczno-logiczna (ALU), wewnątrz której wykonywane są matematyczne operacje arytmetyczno-logiczne, będące żmudną pracą mikroprocesora. Widocznym dowodem pracy jednostki ALU może być np. kontrola procesu produkcyjnego, pomiar i wyświetlanie temperatury itd.

Wszelkie wyniki uprzednio przeprowadzonych obliczeń przechowywane są w specjalnie wydzielonej, do tego celu, części mikroprocesora (rejestrze)– zwanym akumulatorem (A).

Przy budowie mikroprocesora wykorzystano system magistralowego (szynowego) przekazywania informacji. Polega on na tym, że wszystkie bloki funkcjonalne wewnętrznego systemu mikroprocesorowego podłączone są do trzech magistral (szyn):
– szyny adresowej,
– szyny danych,
– szyny sterownia.

Wszelkie dane znajdujące się w magistrali danych są kierowane w konkretne miejsce (decyduje o tym uprzednio napisany program) – pod adres wskazany przez układ sterujący oraz przy pełnym wykorzystaniu szyny adresowej. To, czy informacja ma być przekazana do szyny danych, czy z szyny danych, generalnie decyduje o tym stan rejestru adresów (stan szyny adresowej).

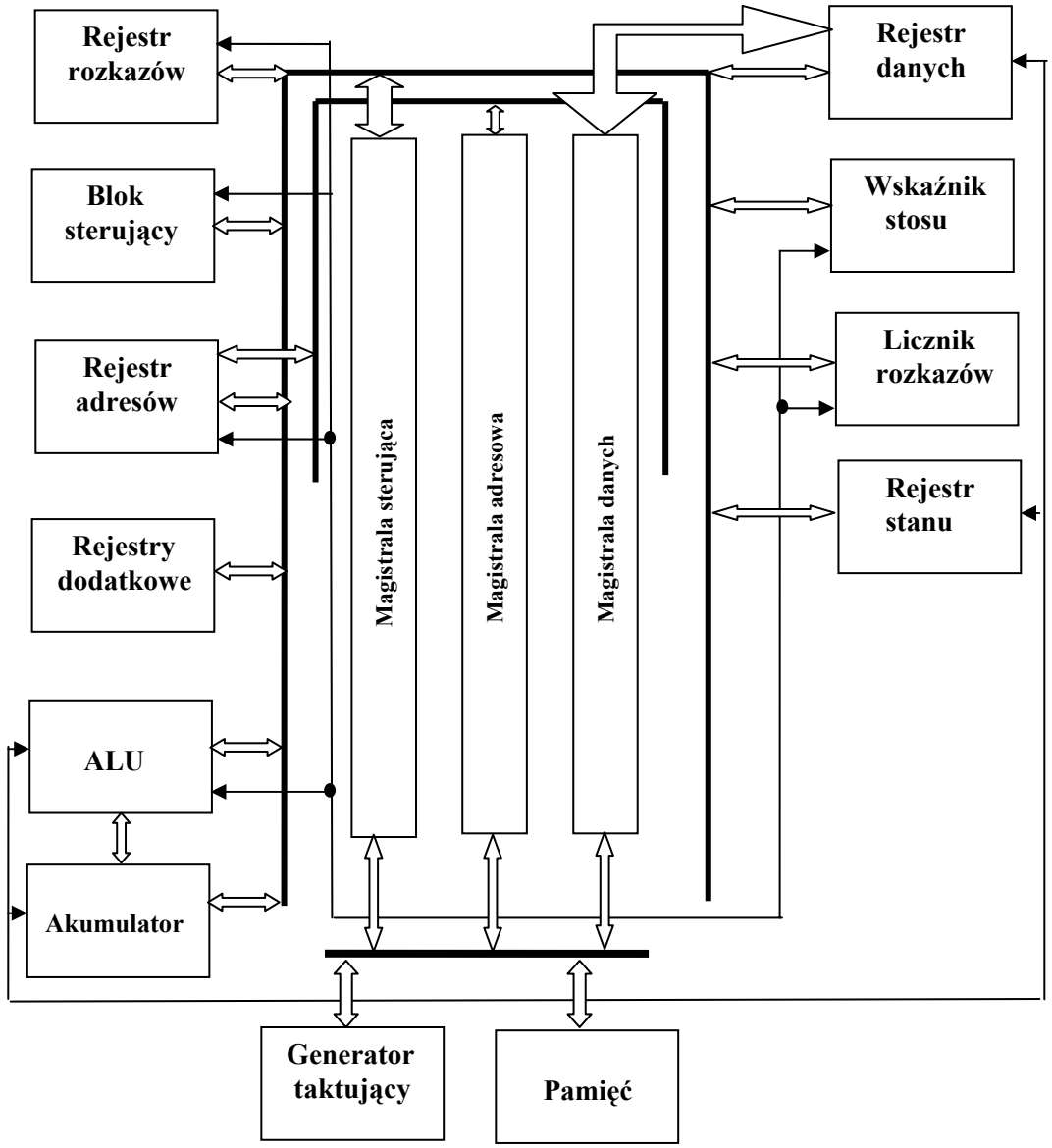
Wykonywanie programu przez mikroprocesor polega na cyklicznym pobieraniu rozkazów z pamięci w rytm odmierzany przez generator taktujący. Generator ten wytwarza przebieg o pewnej, z góry ustalonej częstotliwości wpływającej na wydajność całego systemu.

Do wykonania różnorodnych zadań mikroprocesor potrzebuje kilku impulsów pochodzących z generatora taktującego. Czas potrzebny na zrealizowanie jednego, elementarnego zadania w technice mikroprocesorowej nazywany jest cyklem maszynowym.

Niektóre rozkazy wykonywane są przez procesor w trakcie 1, 2, 3 lub 4 cykli maszynowych.

Każdy mikroprocesor posiada w swojej, wewnętrznej strukturze licznik rozkazów zawierający odpowiedni adres, spod którego ma zostać pobrany kolejny rozkaz. W chwili, gdy mikroprocesor wykona dany rozkaz stan licznika rozkazów jest zwiększany o określoną pozycję, pokazując jednocześnie adres następnego rozkazu przeznaczonego do wykonania.

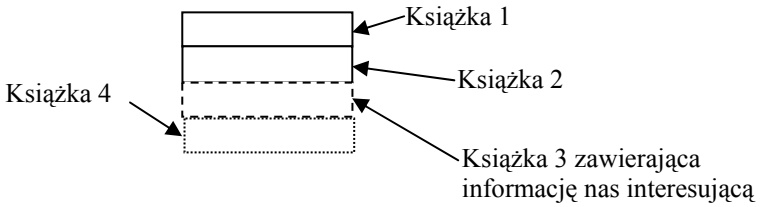
Istnieje możliwość wpisania innego (niż kolejny) adresu do licznika adresowego np. w wyniku wykonania polecenia skoku. Dzięki temu może być realizowany inny fragment programu, który znajduje się w dowolnym miejscu pamięci.



Rys.2. Wewnętrzna struktura mikroprocesora

Widoczny na rysunku rys. 2 blok dodatkowych rejestrów stanowi bank pamięci, który może być dowolnie zarządzany przez programistę lub wykorzystywany przez procesor do przechowywania ważnych informacji podczas wykonywania programu.
Sposób organizowania pracy bloku dodatkowych rejestrów mikroprocesora jest możliwy dzięki odpowiednim ustawieniom, drugiego pod względem ważności (po akumulatorze), rejestru stanu (PSW). Rejestr ten przechowuje również informacje dotyczące wykonywanych działań arytmetycznych (wskaźnik przeniesienia, przepełnienia oraz parzystości). Wskaźniki te (poszczególne bity rejestru PSW) mogą być ustawiane automatycznie lub za pośrednictwem odpowiednich rozkazów.

Równie ważnym elementem wchodzącym w skład architektury mikroprocesora jest stos (będący najczęściej fragmentem pamięci typu RAM).
Stosem nazywamy szczególny rodzaj pamięci – pamięci książkowej. Nazwa ta wynika ze sposobu dostępu do poszczególnych, uprzednio w niej zapisanych danych. Mianowicie jest on podobny do odkrywania treści podręczników ułożonych jeden na drugim (rys.3). Dokładniej oznacza to, że abyśmy mogli uzyskać informację zapisaną wcześniej, należy odczytać najpierw ostatnio zapisaną (zdjąć pierwszy podręcznik), potem kolejną (zdjąć następny podręcznik), aż w końcu dotrzemy do interesującej nas informacji (czyli do podręcznika, w którym zawarta jest informacja nas interesująca).



Rys. 3. Organizacja pamięci typu stos

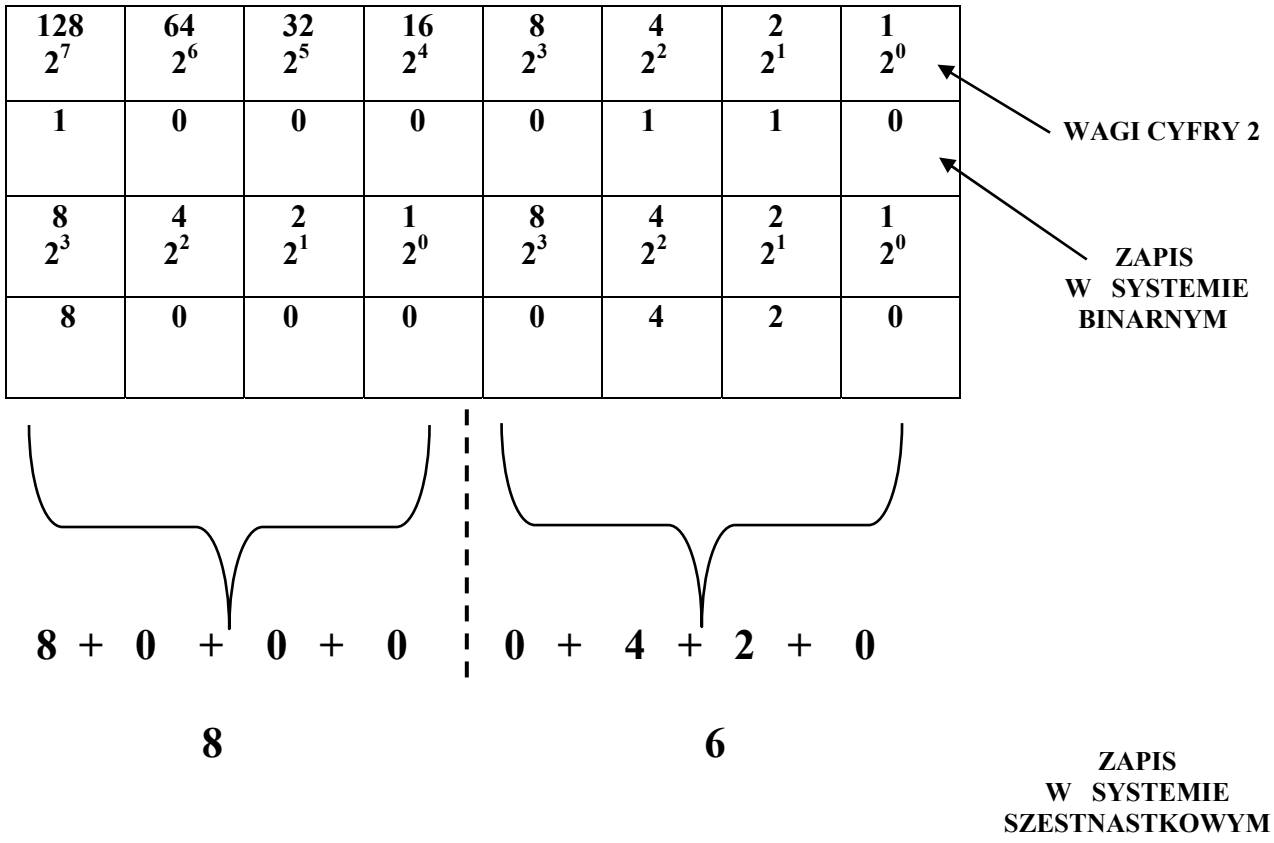
Nieodzownie z pojęciem mikroprocesor wiążą się zagadnienia dotyczące mikrokontrolerów. Mikrokontroler to również układ scalony o wielkiej skali integracji, zawierający w swojej wewnętrznej strukturze mikroprocesor oraz zespół dodatkowych elementów zewnętrznych, czyniący go podzespołem bardziej uniwersalnym. Mikrokontrolery umożliwiają budowę kompletnych sterowników, w których wszystkie, podstawowe funkcje kontrolne spełnia tylko jeden układ scalony.

Systemy i kody liczbowe stosowane w układach mikroprocesorowych

Mikroprocesory komunikują się ze sobą wykorzystując przy tym system binarny – oparty na dwóch stanach logicznych: niski (zero logiczne [0]) i wysoki (jedynka logiczna [1]). W praktyce poziomy logiczne oznaczają pewne, standaryzowane wartości napięć; i tak: zero logiczne to napięcie równe 0 V, zaś jeden logiczne (w zależności od standardu) to 5 V lub 12 V.

Ze względu na znaczną długość zapisu liczby w systemie binarnym częściej używany jest system heksadecymalny (szesnastkowy).

Sposób zamiany liczby zapisanej w systemie dziesiętnym, powszechnie używanym i opartym na dziesięciu elementach (cyfrach) – (0, 1, 2, 3, 4, 5, 6, 7, 8, 9) na liczbę w systemie binarnym a dalej w systemie szesnastkowym został pokazany na rys. 4.



Rys.4. Przykład zamiany liczby 134 zapisanej w systemie dziesiętnym na liczby zapisane w systemie binarnym i szesnastkowym

Z rys. 4 wynika, że aby daną liczbę dziesiętną (w przykładzie – 134) można było zamienić na liczbę w systemie szesnastkowym, należy ją najpierw zamienić na liczbę w kodzie binarnym. Otrzymaną liczbę w kodzie binarnym należy podzielić na grupy po cztery elementy każda. Następnie każdą grupę, z osobna, zamieniamy na odpowiedni symbol w kodzie szesnastkowym, który zawiera szesnaście znaków graficznych: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

Liczby zapisane w dowolnym systemie można zamienić na odpowiadające im liczby w innym systemie, a następnie wynik takiej zamiany zakodować. Kodowanie jest pewnego rodzaju sposobem odwzorowania zmiennej, którą chcemy zakodować, w ciąg słów (wyrazów) kodowych. W przypadku, gdy wszystkim wartościom, jakie przyjmuje dana zmienna, przypisane są słowa kodowe - to taki kod nazywamy zupełnym. Natomiast, gdy nie

jest spełniony ten warunek to taki kod nazywany jest kodem niezupełnym. Często zdarza się, że informacja poddawana kodowaniu jest reprezentowana wartością liczbową. Z tego też względu można ją zakodować przy wykorzystaniu kodu opisanego odpowiednim wzorem. Przykładem takiego kodu, najczęściej używanym w układach mikroprocesorowych, jest naturalny kod binarny zwany kodem wagowym. Jest to kod, który daje możliwość przypisania każdemu bitowi pewnej wagi (współczynnika) oraz pozwala na określenie zmiennej (poddawanej kodowaniu) dzięki zsumowaniu iloczynów poszczególnych wag i bitów. Innym, rzadziej stosowanym kodem jest kod Graya. Charakteryzuje się on tym, iż kolejne, jego wyrazy różną się między sobą na jednym bicie (poniżej, w podanym przykładzie wyodrębnione zostały po dwa wyrazy kodu Graya, różniące się między sobą na jednym bicie).

Przykład kodu Graya dla słowa dwubitowego:

0 **0** → pierwszy wyraz kodu
0 **1** →drugi wyraz kodu
1 1 → trzeci wyraz kodu
1 0 → czwarty wyraz kodu

Przykład kodu Graya dla słowa trzybitowego

0 0 **0** →pierwszy wyraz kodu
0 0 **1** →drugi wyraz kodu
0 1 1 → trzeci wyraz kodu
0 1 0 → czwarty wyraz kodu
1 1 0 → piąty wyraz kodu
1 1 1 → szósty wyraz kodu
1 0 1 → siódmy wyraz kodu
1 0 0 → ósmy wyraz kodu

Pamięci stosowane w układach mikroprocesorowych

Pamięć to układ elektroniczny służący do przechowywania informacji w postaci pojedynczych bitów lub całych bajtów. Pamięci dzielimy na: pamięci stałe – tylko do odczytu oraz pamięci zapisywalne – pamięci służące do odczytu i zapisu.

Do pamięci stałych możemy zaliczyć najpopularniejsze pamięci typu ROM. Składają się one z matrycy pamięciowej (maski) wytwarzanej podczas procesu produkcyjnego. Pamięci tego rodzaju wykonuje się na konkretne zamówienie, na potrzeby produkcji urządzeń elektronicznych wytwarzanych w bardzo dużych ilościach.

Pamięć EPROM jest pamięcią, którą użytkownik może dowolnie zaprogramować w sposób elektryczny z użyciem specjalnego programatora. Zawartość jej pozostaje niezmieniona aż do chwili doprowadzenia do specjalnie skonstruowanego okienka źródła promieniowania ultrafioletowego. Pewną odmianą pamięci EPROM jest pamięć typu OTP – pamięć jednorazowo programowalna elektrycznie bez możliwości jej skasowania i ponownego zaprogramowania.

Poprzednikiem pamięci EPROM (w chwili obecnej rzadko stosowana) była pamięć typu PROM**,** której programowanie jest równoznaczne z przepalaniem odpowiednich połączeń wewnętrznej struktury układu scalonego. Obwody tego rodzaju nie mogą być ponownie ani skasowane, ani zaprogramowane. Kolejnym rodzajem pamięci - często stosowanych w elektronice są tzw. pamięci typu EEPROM. Pamięci EEPROM są układami, które mogą być programowane oraz kasowane elektrycznie. Ich zawartość nie ulega „wymazaniu" po wyłączeniu zasilania. Zazwyczaj są to obwody wykonane w technologii MOS. Gdyby nie charakteryzowały się one ograniczoną liczbą cykli programowania (około 10000 razy) były to pamięci uniwersalne. Czyli ich zawartość nie „ginęłaby" w przypadku zaniku napięcia zasilającego a proces kasowania i programowania mógłby być przeprowadzany nieskończoną liczbę razy.

Obecnie w wielu mikrokontrolerach znajdują się bardzo praktyczne i dosyć tanie pamięci typu FLASH. Są to układy programowalne i kasowalne elektrycznie, tak jak EEPROM–y. Ważną różnicą między pamięciami EEPROM i FLASH jest znacznie mniejszy koszt pamięci typu FLASH. Nazwa FLASH związana jest z bardzo szybkim procesem kasowania i załadowywania informacji do wewnętrznej struktury pamięci.

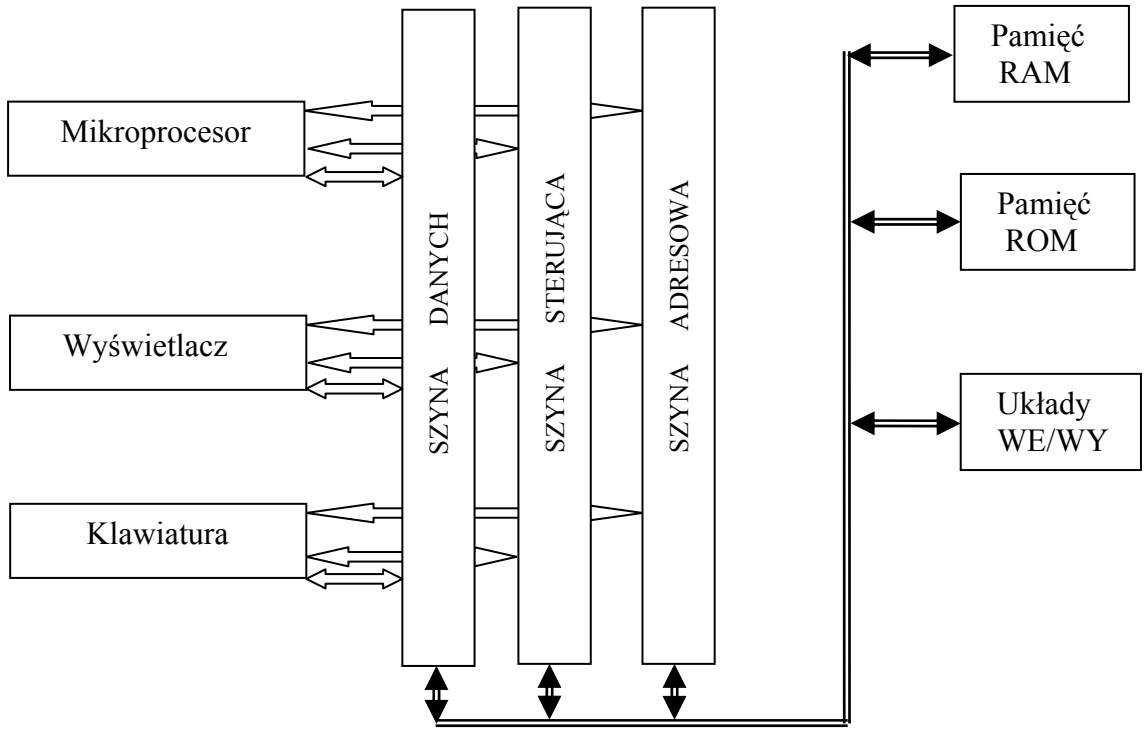
Pamięci typu RAM należą do grupy obwodów cyfrowych, które mogą być programowane i kasowane elektrycznie, z tą różnicą, że z chwilą wyłączenia napięcia zasilającego zawartość ich zostaje bezpowrotnie utracona. Wśród pamięci RAM występują pamięci statyczne SRAM oraz dynamiczne DRAM. Pamięć typu SRAM przechowuje informacje w postaci odpowiednich stanów przerzutników bistabilnych, natomiast DRAM przechowuje informacje pod postacią ładunków zgromadzonych w złączach tranzystorów MOS. Ze względu na powolny zanik tych ładunków (tracenie informacji) pamięci tego typu muszą być odświeżane co pewien okres czasu. Działania takie, będące widoczną wadą, nie dyskwalifikują ich, lecz wręcz przeciwnie, są często stosowane, ale przyczyną tego jest niski koszt ich produkcji.

Systemy mikroprocesorowe

System mikroprocesorowy to rozbudowany układ elektroniczny składający się z mikroprocesora oraz dodatkowych elementów zewnętrznych, z którymi on współpracuje.
Dodatkowe elementy wchodzące w skład systemu mikroprocesorowego to:

- wyświetlacze LED lub LCD,
- klawiatura,
- sygnalizatory LED,
- czujniki,
- przetworniki A/C i C/A,
- sterowniki układów wykonawczych,
- pamięci zewnętrzne,
- układy czasowe,
- sterowniki transmisji danych,
- porty WE/WY,
- układy watchdog,
- zegary czasu rzeczywistego,
- układy resetu.

Zasada działania systemu mikroprocesorowego jest podobna do funkcjonowania samego mikroprocesora. Podstawową różnicą jest zwiększenie możliwości komunikowania się systemu mikroprocesorowego z otoczeniem przy wykorzystaniu dodatkowych układów. Systemy mikroprocesorowe posiadają znacznie większe pamięci operacyjne, pozwalają na programową regulację i sterowanie dowolnymi procesami. Strukturę systemu mikroprocesorowego przedstawiono na rys. 5.



Rys. 5. Struktura systemu mikroprocesorowego

Sposoby przesyłu informacji stosowane w systemach mikroprocesorowych

We wszystkich systemach mikroprocesorowych istnieje konieczność przesyłania informacji wewnątrz ich samych, jak również do innych systemów. Najprostszym sposobem przesyłu danych jest transmisja równoległa. Polega ona na połączeniu systemów za pomocą odpowiedniej liczby linii głównych (w zależności od rodzaju współpracujących ze sobą procesorów, np. 8, 16, 32 lub 64 linii) oraz dwóch lub trzech linii sterujących. W przypadku popularnych procesorów 8–bitowych każdemu z bitów przyporządkowana jest konkretna linia. Łączna liczba użytych linii wynosi 8 plus dodatkowo 3 linie sterujące. Tak duże wykorzystanie wolnych linii mikroprocesora jest poważną wadą. Ponadto wyprowadzenie kilku (kilkunastu, kilkudziesięciu) przewodów z aktywnie pracującego systemu, z dużą częstotliwością, powoduje powstawanie zakłóceń przy przesyle informacji na duże odległości. Mimo tego transmisja równoległa jest dość często stosowana z tym, że przesył danych winien odbywać się na niewielkie odległości, oczywiście przy zachowaniu znacznych prędkości transferu.

W systemach mikroprocesorowych znacznie częściej używana jest szeregowa transmisja danych. Charakteryzuje się ona tym, że bity przesyłane są w sposób szeregowy, tzn. jeden za drugim. Możliwe jest to dzięki zastosowaniu jednej lub dwóch linii aktywnych, funkcjonujących w danym obwodzie.

Sposób przekazu danych z użyciem dwóch linii zwany jest szeregową transmisją synchroniczną, a z użyciem jednej – szeregową transmisją asynchroniczną. W układach mikroprocesorowych najczęściej wykorzystywana jest szeregowa transmisja asynchroniczna. Występująca w niej jedna linia, służąca do przekazu danych na duże odległości, to ogromna zaleta. Natomiast wadą tego sposobu transferu danych jest znacznie ograniczona szybkość przesyłu bitów (w odniesieniu do transmisji równoległej).

Przesył asynchroniczny zazwyczaj poprzedzony jest bitem startu, po którym przekazywane są dane. Każdy pakiet danych wyprowadzanych z systemu zakańczany jest 1 lub 2 bitami stopu.

Istnieje jeszcze wiele innych sposobów przesyłu informacji stosowanych w systemach mikroprocesorowych. Wśród nich na uwagę zasługuje system wykorzystujący szynę I^2C.

W skład szyny I^2C wchodzą dwie linie: SDA – linia danych oraz SCL – linia zegara taktującego. Wszystkie układy elektroniczne podłączone do szyny posiadają swoje adresy i mogą wysyłać lub odbierać dane. Niektóre z członów systemu I^2C mogą tylko odbierać sygnały bez możliwości ich wysyłania. Transmisję I^2C nadzoruje jeden lub kilka układów scalonych (przeważnie są to mikrokontrolery) – master(y). Każdy, inny obwód podłączony do szyny jest układem typu slave. Sposób przekazu informacji w systemie I^2C jest podobny do transmisji szeregowej (przy czym liczba przesyłanych bajtów danych nie jest ograniczana jak w transferze szeregowym).

W transmisji typu I^2C istnieje pewien problem w chwili, gdy do wspólnej szyny zostanie podłączonych kilka kontrolerów typu master i zaczynają one jednocześnie transmisję danych. Powstaje w tym momencie zerwanie przekazu lub co gorsze jego zafałszowanie. W celu uniknięcia takiej sytuacji zastosowano specjalną procedurę pozwalającą ustalić, który z procesorów jest urządzeniem typu master.

4.1.2. Pytania sprawdzające

Odpowiadając na pytania, sprawdzisz, czy jesteś przygotowany do wykonania ćwiczeń.

1. Co to jest mikroprocesor?
2. Jak wygląda wewnętrzna struktura mikroprocesora?
3. Do czego służą poszczególne elementy wewnętrznej struktury mikroprocesora?
4. Co to jest i do czego służy system mikroprocesorowy?
5. Jakie znasz pamięci stosowane w układach mikroprocesorowych?
6. Jakie znasz systemy liczbowe stosowane w układach mikroprocesorowych?
7. Jakie znane Ci są sposoby przesyłu informacji, stosowane w systemach mikroprocesorowych?

4.1.3. Ćwiczenia

Ćwiczenie 1

Liczbę 126, zapisaną w kodzie dziesiętnym, zamień na liczbę w kodzie binarnym oraz szesnastkowym.

Sposób wykonania ćwiczenia

Aby wykonać ćwiczenie powinieneś:

1) zanotować podaną liczbę dziesiętną,
2) znaleźć jej odpowiednik w systemie binarnym,
3) znaleźć jej odpowiednik w systemie szesnastkowym.

Wyposażenie stanowiska pracy:

- instrukcje zawierające treść ćwiczenia.

Ćwiczenie 2

Wymień wady i zalety metod przesyłu informacji stosowanych w systemach mikroprocesorowych.

Sposób wykonania ćwiczenia

Aby wykonać ćwiczenie powinieneś:

1) odnaleźć w dostarczonych materiałach metody przesyłu informacji stosowane w systemach mikroprocesorowych,
2) zanotować znalezione metody,
3) wypisać w formie tabeli wady i zalety odnalezionych metod przesyłu informacji.

Wyposażenie stanowiska pracy:

- literatura zawierająca informacje na temat metod przesyłu informacji stosowanych w systemach mikroprocesorowych.

4.1.4. Sprawdzian postępów

Czy potrafisz:	Tak	Nie
1) zdefiniować pojęcie mikroprocesor?	☐	☐
2) zdefiniować pojęcie system mikroprocesorowy?	☐	☐
3) scharakteryzować pamięci stosowane w systemach mikroprocesorowych?	☐	☐
1) zamieniać liczby z jednego systemu na inny system liczbowy?	☐	☐
2) określić różnice (elementy wspólne) między szeregowym i równoległym sposobem przesyłu informacji?	☐	☐

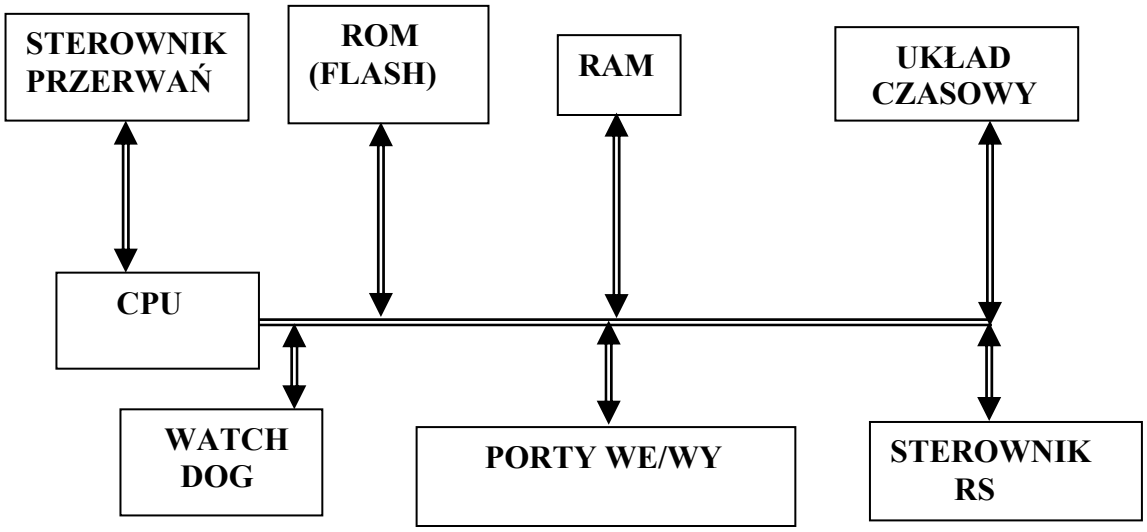
4.2. Podział ogólny i zastosowanie mikroprocesorów

4.2.1. Materiał nauczania

Mikroprocesory jednoukładowe

Mikroprocesory jednoukładowe, zwane również mikrokontrolerami, należą do największej rodziny procesorów stosowanych w przemyśle oraz w rozmaitych aplikacjach amatorskich i profesjonalnych. Prym wśród tej grupy wiedzie rodzina kontrolerów 8-bitowych serii MCS–51. Obecnie procesory te, o znacznie rozbudowanej architekturze, produkowane są przez wiele firm. Najbardziej znaczące wśród nich to: ATMEL (produkujący procesory typu 89C51) oraz INTEL (produkujący procesory typu 80C51).

Poglądowy schemat blokowy mikrokontrolera 8–bitowego typu MCS–51 przedstawiono na rys.6.



Rys. 6. Poglądowy schemat blokowy mikrokontrolera 8–bitowego typu 89S52

Na podstawie analizy rys. 6 można stwierdzić, że mikrokontroler 89S52 to nic innego, jak niewielki system mikroprocesorowy. Wynika z tego fakt, że jego zasada działania, wewnętrzny i zewnętrzny sposób przesyłu informacji są takie same jak w przypadku systemu mikroprocesorowego. Po to, by mikrokontroler był urządzeniem pracującym samodzielnie, oprócz jednostki centralnej CPU, z mikroprocesorem dodatkowo zamieszczono w jego wewnętrznej strukturze następujące układy:

- watchdog,
- sterownik przerwań,
- układ czasowy,
- sterownik RS,
- porty WE/WY.

Blok watchdog (czuwający pies) jest bardzo często stosowanym obwodem elektrycznym w mikroukładach, a przy tym wykorzystywanych głównie w sprzęcie profesjonalnym. Jego zadanie, polegające na nadzorowaniu poprawności wykonywania dowolnego programu,

w znaczący sposób poprawia niezawodność całego systemu. Urządzenie takie odmierza czas pomiędzy kolejnymi impulsami wysyłanymi do niego z systemu mikrokontrolera.

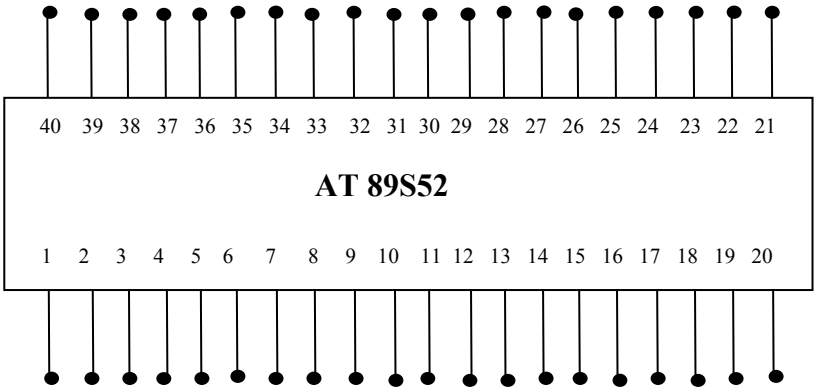
W przypadku, gdy od ostatnio otrzymanego impulsu upłynie zbyt długi okres czasu to procesor zostaje restartowany, a cały system powraca do stanu równowagi.

Sterownik przerwań jest to zespół obwodów generujących tzw. przerwanie programowe. Przerwanie jest zawieszeniem czynności związanych z wykonaniem głównego programu na rzecz wykonania innego, ważniejszego programu. Przerwanie może być zainicjowane sygnałami zewnętrznymi (przy wykorzystaniu specjalnych wejść) lub wewnętrznymi (pochodzącymi od układów czasowych, sterownika RS). W systemie przerwań, każde z nich, posiada pewien priorytet. Oznacza to, że istnieje kolejność wykonywania programów pochodzących od przerwań w chwili jednoczesnego ich zainicjowania.

Sterownik RS jest w pełni kompletnym modułem służącym do transmisji szeregowej danych (opisanej w rozdziale 4.1. na str. nr 13).

Porty WE/WY zwane również portami, to linie po 8 bitów każda, służące do równoległej transmisji danych do i z urządzeń zewnętrznych wchodzących w skład systemu mikroprocesorowego.

Na rys. 7 przedstawiono symbol graficzny typowego procesora jednoukładowego.



Rys. 7. Symbol graficzny wraz z oznaczonymi wyprowadzeniami procesora typu AT 89S52

Opis funkcji przyporządkowanych poszczególnym końcówkom układu scalonego AT 89S52 zestawiono w tabeli 1.

Tabela 1. Opis funkcji przyporządkowanych poszczególnym końcówkom układu scalonego AT 89S52

Pin	Realizowana funkcja
1	WY/WE portu P1 bit 0
2	WY/WE portu P1 bit 1
3	WY/WE portu P1 bit 2
4	WY/WE portu P1 bit 3
5	WY/WE portu P1 bit 4
6	WY/WE portu P1 bit 5
7	WY/WE portu P1 bit 6
8	WY/WE portu P1 bit 7
9	Reset układu
10	WY/WE portu P3 bit 0 (dodatkowo WE sterownika RS)
11	WY/WE portu P3 bit 1 (dodatkowo WY sterownika RS)
12	WY/WE portu P3 bit 2 (dodatkowo 1 WE przerwań – INT0)

13	WY/WE portu P3 bit 3 (dodatkowo 2 WE przerwań - INT1)
14	WY/WE portu P3 bit 4 (dodatkowo WE układu czasowego T0)
15	WY/WE portu P3 bit 5 (dodatkowo WE układu czasowego T1)
16	WY/WE portu P3 bit 6 (dodatkowo WY przy zapisie do pamięci zewnętrznej)
17	WY/WE portu P3 bit 7 (dodatkowo WY przy odczycie z zewnętrznej pamięci)
18 19	WE służące do podłączenia rezonatora kwarcowego (dla danej częstotliwości rezonatora fr czas trwania jednego cyklu maszynowego wynosi T=fr/12)
20	Masa
21	WY/WE portu P2 bit 0
22	WY/WE portu P2 bit 1
23	WY/WE portu P2 bit 2
24	WY/WE portu P2 bit 3
25	WY/WE portu P2 bit 4
26	WY/WE portu P2 bit 5
27	WY/WE portu P2 bit 6
28	WY/WE portu P2 bit 7
29	Wyjście sterujące odczytywaniem danych z zewnętrznej pamięci programu
30	Wyjście sygnału zegara $f=f_r/6$
31	Wejście wyzwalające pobieranie programu z zewnętrznej pamięci programu
32	WY/WE portu P0 bit 0
33	WY/WE portu P0 bit 1
34	WY/WE portu P0 bit 2
35	WY/WE portu P0 bit 3
36	WY/WE portu P0 bit 4
37	WY/WE portu P0 bit 5
38	WY/WE portu P0 bit 6
39	WY/WE portu P0 bit 7
40	+ Zasilanie

Układ elektroniczny służący do testowania programów (zwany układem uruchomieniowym) napisanych dla mikrokontrolerów typu MCS–51 został pokazany na rysunku 11 w rozdziale 4.3. „ Programowanie mikroprocesorów”.

Programowanie mikroprocesorów typu MCS–51 wymaga specjalnych urządzeń zwanych programatorami. Przed uaktywnieniem procesu programowania dany układ scalony należy włożyć do specjalnej podstawki (zazwyczaj typu ZIF) i dopiero wtedy rozpocząć programowanie. Wynika z tego fakt, że proces programowania nie może odbywać się, gdy procesor znajduje się w docelowym układzie roboczym. Dokładniej proces programowania mikrokontrolerów rodziny MCS–51, ze względu na ich dużą popularność, zostanie opisany w rozdziale 4.3. „Programowanie mikroprocesorów”.

Mikroprocesory typu RISC

Mikroprocesory typu RISC, których architektura została już zaprojektowana w latach 80, charakteryzują się bardzo ciekawymi parametrami szeregującymi je na najwyższej półce obecnie stosowanych mikroukładów.

Posiadają one znacznie skróconą liczbę rozkazów (w stosunku do mikroprocesorów jednoukładowych rodziny MCS–51) przez co upraszcza się proces ich programowania oraz czas wykonywania programu. Charakteryzują się one pewnymi ograniczeniami w komunikacji występującej pomiędzy mikroprocesorem a współpracującą z nim pamięcią. Przejawem takiego stanu rzeczy jest zastosowanie oddzielnych instrukcji służących do wprowadzania i wyprowadzania danych do i z pamięci. Pozostałe instrukcje mikroprocesorów typu RISC angażują tylko i wyłącznie rejestry, których ilość jest o wiele większa niż w procesorach jednoukładowych typu MCS–51.

Zastosowanie znaczącej liczby rejestrów (od 32 do 256) w widoczny sposób zwiększa wydajność systemu, odciążając jednocześnie szynę procesor–pamięć.
Najważniejszym elementem wpływającym na dużą wydajność i jakość pracy procesora typu RISC jest zastosowanie przetwarzania potokowego. Polega ono na tym, że kolejne rozkazy danego programu załadowanego w pamięci systemu wykonywane są w jednym cyklu maszynowym. W tym momencie należy wspomnieć, iż mikroprocesorom typu MCS–51 na wykonanie pojedynczego rozkazu – w zależności od typu, potrzebny był czas: jednego, dwóch, trzech lub czterech cykli maszynowych.

Wśród mikroprocesorów RISC możemy wyróżnić kilka, bardzo rozpowszechnionych grup (rodzin), mianowicie są to:
- procesory AMD
- procesory SPARC
- procesory ARM
- procesory Power PC
- procesory MIPS.

Zastosowanie procesorów o architekturze typu RISC jest ogromne – począwszy od mikroukładów stosowanych do celów przemysłowych i amatorskich, po rozbudowane systemy mikroprocesorowe oraz sprzęt PC. Wykorzystywane są również w: telefonii komórkowej oraz w urządzeniach służących do przesyłu informacji w sieciach internetowych.

Mikroprocesory typu AVR

Mikroprocesory typu AVR należą do grupy procesorów jednoukładowych powoli wypierających mikroukłady wchodzące w skład rodziny MCS–51.Układy te, opracowane przez firmę ATMEL, są w pełni kompatybilne z procesorami rodziny MCS–51. Ich kompatybilność wiąże się nie tylko z analogicznym rozkładem wyprowadzeń, ale również i z podobną listą wykonywanych rozkazów przez obydwa rodzaje procesorów.
Do podstawowych różnic między procesorami typu AVR i MCS–51 można zaliczyć:

- większą szybkość działania (wynika ona z zastosowania w procesorze AVR architektury typu RISC),
- szeroki zakres napięć zasilających (procesory AVR),
- niewielki pobór mocy (procesory AVR),
- funkcje akumulatora w procesorze typu AVR może przejąć dowolnie wybrany przez użytkownika rejestr (metoda taka w znaczny sposób poprawia wydajność wykonywanych algorytmów),
- zastosowanie szeregowego algorytmu programowania mikrokontrolera (algorytm ten umożliwia programowanie i kasowanie procesora bez potrzeby wyjmowania go z układu roboczego, tak jak to odbywa się w przypadku procesorów MCS–51).

Procesory typu AVR stosowane są w różnych układach sterujących, urządzeniach pomiarowych, rozbudowanych systemach alarmowych, blokach przetwarzających dane, mobilnych robotach przemysłowych oraz przemysłowych sterownikach swobodnie programowalnych.

Mikroprocesory sygnałowe

Mikroprocesory sygnałowe, zwane również cyfrowymi procesorami sygnałowymi (DSP), służą do przetwarzania szybkozmiennych sygnałów analogowych. Należą one do odrębnej, wyspecjalizowanej grupy procesorów. Cechą charakterystyczną tych procesorów jest ściśle wyodrębniona, w ich wewnętrznej strukturze, pamięć programu i pamięć danych.

Taka konstrukcja umożliwia zwiększenie szybkości wykonywania wszystkich operacji. Mikroprocesory te posiadają również możliwość równoczesnego odczytu danych i instrukcji. Praca procesorów DSP opiera się na potokowym przetwarzaniu instrukcji, dzięki czemu odznaczają się one dużą wydajnością i bardzo krótkim czasem wykonywania rozkazów.

Procesory sygnałowe budowane są zasadniczo jako: jednostki 16-, 24- lub 32-bitowe. Posiadają znacznie rozbudowaną jednostkę arytmetyczno-logiczną oraz szybki algorytm wykonywania operacji mnożenia danych. Nieodłącznym elementem ich wewnętrznej struktury są szybkodziałające przetworniki A/C i układy mnożące.

Powyższe własności procesorów DSP powodują, że znalazły one zastosowanie w urządzeniach służących do przetwarzania sygnałów analogowych, w szczególności dźwięku. Wykorzystuje się je do inteligentnej filtracji sygnałów, obróbki i zapisu mowy ludzkiej oraz do produkcji powszechnie używanych kart dźwiękowych, stosowanych w komputerach PC. Dodatkowo procesory DSP wykorzystywane są w licznych konstrukcjach urządzeń energoelektronicznych.

Najchętniej stosowaną rodziną mikroprocesorów DSP jest grupa procesorów typu TMS320, produkowanych przez firmę Texas Instruments.

4.2.2. Pytania sprawdzające

Odpowiadając na pytania, sprawdzisz, czy jesteś przygotowany do wykonania ćwiczeń.

1. Co to jest mikrokontroler?
2. Jakimi parametrami charakteryzuje się mikroprocesor typu RISC?
3. Do czego służą mikroprocesory typu AVR?
4. Co to jest mikroprocesor sygnałowy?

4.2.3. Ćwiczenia

Ćwiczenie 1

Opisz budowę oraz parametry mikroprocesora AT 89S52.

Sposób wykonania ćwiczenia

Aby wykonać ćwiczenie powinieneś:

1) odnaleźć w otrzymanych materiałach lub na wskazanej stronie internetowej dane katalogowe szukanego mikroprocesora,
2) narysować schemat obrazujący budowę mikroprocesora,
3) zapisać odpowiednie parametry szukanego mikroprocesora.

Wyposażenie stanowiska pracy:

- literatura dotycząca mikroprocesora AT 89S52,
- zestaw internetowy dający możliwość odnalezienia odpowiednich informacji na konkretnych stronach internetowych.

Ćwiczenie 2

Wskaż elementy wspólne oraz różnice między mikroprocesorami jednoukładowymi typu RISC i rodziny MCS –51.

Sposób wykonania ćwiczenia

Aby wykonać ćwiczenie powinieneś:

1) odnaleźć w dostarczonych materiałach charakterystyczne parametry danych mikroprocesorów,
2) odnaleźć w dostarczonych materiałach opis budowy wewnętrznej porównywanych ze sobą mikroprocesorów,
3) odnaleźć w dostarczonych materiałach zastosowania porównywanych ze sobą mikroprocesorów,
4) wypisać w formie tabeli różnice i podobieństwa występujące między wyszczególnionymi mikroprocesorami.

Wyposażenie stanowiska pracy:

- literatura zawierająca informacje na temat mikroprocesorów MCS–51 oraz RISC.

4.2.4. Sprawdzian postępów

Czy potrafisz:	**Tak**	**Nie**
1) zdefiniować pojęcie mikrokontroler?	☐	☐
2) zdefiniować pojęcie mikroprocesor typu RISC?	☐	☐
3) zdefiniować pojecie mikroprocesor sygnałowy?	☐	☐
4) zdefiniować pojęcie mikroprocesor AVR?	☐	☐

4.3. Programowanie mikroprocesorów

4.3.1. Materiał nauczania

Podstawy programowania mikroprocesorów

Mikroprocesor (mikrokontroler) wykonuje program zawarty w jego pamięci programu w postaci zestawu bitów. Taka postać jest mało czytelna i zbyt trudna w zrozumieniu dla przeciętnego człowieka, dlatego też dla każdego z typu mikroprocesora opracowano język programowania – asembler. Stanowi on zestaw instrukcji (rozkazów) napisanych w tzw. postaci symbolicznej i nazywany jest kodem źródłowym programu. Asembler umożliwia również zamianę (asemblację) kodu źródłowego programu na jego odpowiednik zapisany w postaci binarnej - zwany postacią wynikową.

Rozkazy interpretowane przez mikroprocesory między innymi obejmują:
- operacje arytmetyczne na danych,
- operacje logiczne na danych,
- operacje przesyłania danych,
- instrukcje służące do sterowania pracą programu – skoki,
- operacje obsługujące moduły WE/WY,
- operacje nadzorujące pracą czasomierzy,
- operacje obsługi przerwań.

Każdy program napisany w asemblerze składa się z pięciu, zasadniczych części:
1 – instrukcji wstępnie organizujących pamięć programu,
2 – deklaracji stałych (zmiennych) występujących w danym programie,
3 – bloku programów obsługujących przerwania,
4 – programu właściwego,
5 – symbolicznego zakończenia programu.

Ogólna (przykładowa) struktura programu w asemblerze będzie wyglądała w następujący sposób:

ORG 100H – Instrukcja ta informuje o organizacji pamięci, czyli program właściwy zostanie załadowany począwszy od komórki (w naszym przypadku) o adresie 100H.
- Komórki pamięci zaczynające się od adresu 0H do 100H przeznaczone są na podprogramy obsługujące przerwania. Literka H umieszczona po adresie 100 oznacza liczbę zapisaną w systemie szesnastkowym.
- Program, który nie będzie zawierał podprogramów obsługujących przerwania może zaczynać się od dowolnego adresu, np. 00H.

Parametr1 **EQU** 10101111B – deklaracja stałej Parametr1=10101111B, gdzie B oznacza liczbę zapisaną w systemie binarnym
Parametr2 **EQU** 20 – deklaracja stałej Parametr2=20 , 20 bez literki B lub H oznacza liczbę zapisaną w systemie dziesiętnym

ORG 03H
;tu może znajdować się podprogram obsługujący przerwanie pochodzące od pierwszego wejścia INT0
RETI; powrót do programu głównego.

ORG 0BH
;tu może znajdować się podprogram obsługujący przerwanie pochodzące od układu czasowego T0
RETI; powrót do programu głównego

ORG 13H
;tu może znajdować się podprogram obsługujący przerwanie pochodzące od drugiego wejścia INT1
RETI; powrót do programu głównego

ORG 1BH
;tu może znajdować się podprogram obsługujący przerwanie pochodzące od układu czasowego T1
RETI; powrót do programu głównego

ORG 23H
;tu może znajdować się podprogram obsługujący przerwanie pochodzące od portu szeregowego
RETI; powrót do programu głównego

ORG 2BH
;tu może znajdować się podprogram obsługujący przerwanie pochodzące od układu czasowego T2
RETI; powrót do programu głównego

;W TYM MIEJSCU ZNAJDUJE SIĘ PROGRAM WŁAŚCIWY (GŁÓWNY)

NOP; instrukcja typu – nic nie rób (stosowana dla stworzenia zwłoki czasowej równej 1 cykl maszynowy – w tym przypadku bezużyteczna)
END; instrukcja kończąca wszystkie programy napisane w asemblerze

Należy zauważyć, że w przypadku gdy piszemy program w asemblerze możemy korzystać ze specjalnych instrukcji zrozumiałych przez procesor oraz z poleceń ignorowanych przez niego, a pomocnych nam, czyli komentarzy pisanych po średniku (;).
Podsumowując można stwierdzić, że program to zestaw linii rozkazowych zawierających odpowiednie instrukcje lub instrukcje wraz z pewnymi parametrami oraz odpowiednio dobrane komentarze.

W dalszym etapie opisane zostaną najczęściej używane instrukcje pomocne przy tworzeniu programów w asemblerze.
1) Operacje arytmetyczne na danych, czyli: dodawanie, odejmowanie, mnożenie, dzielenie, zmniejszanie zawartości akumulatora o jeden (dekrementacja), zwiększanie zawartości akumulatora o jeden (inkrementacja).

DODAWANIE **ADD**
Operacja dodawania polega na algebraicznym dodaniu do zawartości akumulatora zawartości rejestru specjalnego (rejestry specjalne - są to banki pamięci wykorzystywane przez programistę do chwilowego przechowywania informacji oraz do przeprowadzania różnych operacji na danych. Występuje osiem podstawowych rejestrów, odpowiednio oznaczonych R0, R1, R2, R3, R4, R5, R6, R7) lub danej liczbowej i wpisaniu otrzymanego wyniku do akumulatora.

Przykład 1
ADD A,R1; operacja ta powoduje dodanie zawartości rejestru R1 do zawartości akumulatora A oraz umieszcza wynik w akumulatorze A.

Przykład 2
ADD A,#14; operacja ta powoduje dodanie wartości liczbowej równej 14 do zawartości akumulatora A oraz umieszcza wynik w akumulatorze A.

ODEJMOWANIE **SUBB**
W przypadku operacji odejmowania od zawartości akumulatora A jest odejmowany dany argument oraz zawartość specjalnego bitu pożyczki C
Przykład 1
SUBB A,#10; operacja ta powoduje zmniejszenie zawartości akumulatora o liczbę równą 10 oraz o stan bitu pożyczki C (w przypadku gdy nie był on wcześniej wyzerowany)

MNOŻENIE **MUL AB**
Wynikiem tej operacji jest pomnożenie ośmiobitowej liczby znajdującej się w akumulatorze A przez również ośmiobitową liczbę wpisaną uprzednio do rejestru specjalnego B. W wyniku mnożenia osiem bardziej znaczących bitów wpisywanych jest do rejestru B a osiem mniej znaczących do akumulatora A.

DZIELENIE **DIV AB**
Podobnie jak w przypadku mnożenia, liczba zawarta w akumulatorze A jest dzielona przez liczbę ośmiobitową zamieszczoną w rejestrze B. Wynik dzielenia, w postaci części całkowitej, wpisywany jest do akumulatora, zaś reszta do rejestru B.

DEKREMENTACJA **DEC**
Funkcja ta powoduje zmniejszenie wskazanego parametru (akumulatora, rejestru, zawartości pamięci) o jeden.

Przykład
DEC A; instrukcja ta powoduje zmniejszenie zawartości akumulatora o jeden

INKREMENTACJA **INC**
Funkcja ta powoduje zwiększenie wskazanego parametru (akumulatora, rejestru, zawartości pamięci) o jeden.

Przykład
INC A; instrukcja ta powoduje zwiększenie zawartości akumulatora o jeden

2) Operacje logiczne na danych,
   a) operacje na bajtach, czyli: iloczyn logiczny dwóch bajtów, suma logiczna dwóch bajtów, zerowanie (negowanie) zawartości akumulatora, zamiana półbajtów.

ILOCZYN LOGICZNY **ANL X_1, X_2**
Wykonanie tej instrukcji powoduje pomnożenie w sposób logiczny (bit po bicie) danych parametrów **X_1, X_2** oraz wpisanie wyniku do komórki, z której został pobrany parametr **X_1.**
Zmiennymi **X_1 i X_2** są: akumulator, rejestr, zawartości pamięci.

Przykład
ANL A,R1; w wyniku pomnożenia zawartości akumulatora oraz zawartości rejestru R1 uzyskany wynik zostanie wpisany do akumulatora

SUMA LOGICZNNA **ORL X_1, X_2**
Wykonanie tej instrukcji powoduje zsumowanie w sposób logiczny (bit po bicie) danych parametrów **X_1, X_2** oraz wpisanie wyniku do komórki, z której został pobrany parametr **X_1.**
Zmiennymi **X_1 i X_2** są: akumulator, rejestr, zawartości pamięci.

Przykład
ORL A,@R1; w wyniku zsumowania zawartości akumulatora oraz zawartości komórki pamięci zaadresowanej w sposób pośredni, przy pomocy rejestru R1, uzyskany wynik zostanie wpisany do akumulatora

ZEROWANIE ZAWARTOŚCI AKUMULATORA **CLR A**
Operacja ta powoduje wpisanie do akumulatora wartości binarnej równej: 00000000B

ZANEGOWANIE ZAWARTOŚCI AKUMULATORA **CPL A**
Operacja ta powoduje zanegowanie wszystkich bitów wpisanych uprzednio do akumulatora
Przykład
W akumulatorze znajduje się liczba: 00010101B. W wyniku operacji CPL A, zawartość akumulatora zmieni się w następujący sposób: 11101010B.

ZAMIANA PÓŁBAJTÓW **SWAP A**
Instrukcja SWAP wymienia półbajty (mniej i bardziej znaczący półbajt) w akumulatorze.

b) operacje logiczne na bitach, czyli: zerowanie, negowanie i ustawianie bitów oraz iloczyn i suma logiczna

Operacje logiczne na bitach stanowią pełną analogię do operacji na całych bajtach. Wśród nich możemy wyróżnić:
– ZEROWANIE BITU **CLR bit** (np. CLR ACC.3 – wyzeruj czwarty bit akumulatora),
– NEGOWANIE BITU **CPL bit** (np. CPL P1.2, zaneguj trzeci bit portu P1),
– USTAWIANIE BITU **SETB bit** (np. SETB ACC.3, ustaw czwarty bit akumulatora w stan –1 logiczny),
– ILOCZYN LOGICZNY **ANL C, bit**
– SUMA LOGICZNA **ORL C, bit** ; C bit przeniesienia rejestru specjalnego.

3) Operacje przesyłania danych, czyli: kopiowanie danych pomiędzy rejestrami oraz zapis i odczyt stosu

KOPIOWANIE DANYCH **MOV X1, X2**
Dane wskazane przez argument X2 kopiowane są do miejsca wskazanego przez argument X1
Wyróżniamy następujące rodzaje kopiowania danych:
– MOV A, R_x (kopiowanie zawartości rejestru R_x do akumulatora),
– MOV A, @R_x (pośrednie kopiowanie zawartości komórki pamięci zapisanej w rejestrze R_x do akumulatora),
– MOV A, #stała (wprowadzenie do akumulatora stałej liczbowej, np. MOV A, #10B).
Oprócz kopiowania danych do akumulatora istnieje również możliwość kopiowania danych do rejestrów oraz komórek pamięci z akumulatora.

ZAPIS DANYCH NA STOS **PUSH adres**
Instrukcja ta powoduje zapis zawartości z wewnętrznej pamięci danych o podanym adresie do komórki pamięci wewnętrznej specjalnego rejestru, o adresie zawartym w rejestrze zarezerwowanym do obsługi stosu.

ODCZYT DANYCH ZE STOSU **POP adres**
Instrukcja ta powoduje zapis zawartości z komórki pamięci wewnętrznej specjalnego rejestru, o adresie zawartym w rejestrze zarezerwowanym do obsługi stosu, do komórki wewnętrznej pamięci danych o podanym adresie.

4) Skoki – czyli skok bezwarunkowy, skok do podprogramu, skoki warunkowe zależne od bitu oraz skoki specjalne.

SKOK DO PODPROGRAMU **LCALL nazwa RET**
Po wykonaniu powyższej instrukcji następuje skok do podprogramu o podanej nazwie oraz powrót z niego po wykonaniu instrukcji RET

Przykład

```
LCALL zegar              ;skok do podprogramu o nazwie zegar
      (* )MOV A, R1      ;program główny (jest on interpretowany dopiero po
                         ;wykonaniu podprogramu oraz instrukcji typu RET)

zegar:
                ;jeżeli zachodzi taka potrzeba w tym miejscu mogą znajdować się
                ;instrukcje wchodzące w skład podprogramu o nazwie zegar
RET             ;powrót z podprogramu o nazwie zegar do miejsca oznaczonego (*)
```

SKOK BEZWARUNKOWY **LJMP nazwa**
Po wykonaniu powyższej instrukcji następuje skok do podprogramu o podanej nazwie.

Przykład

```
LJMP dioda2             ;skok do podprogramu dioda2
dioda1:
      CLR P2.3          ;program główny (w tym przypadku nie będzie on wykonywany)
dioda2:
                        ;tu mogą znajdować się instrukcje wchodzące w skład
                        ;podprogramu dioda2
END
```

SKOKI WARUNKOWE ZALEŻNE OD BITU
– **JB bit, nazwa** ;skok w odpowiednie miejsca programu (oznaczone nazwą), w przypadku gdy zmienna (bit) jest równa 1
– **JNB bit, nazwa** ;skok w odpowiednie miejsca programu (oznaczone nazwą), w przypadku gdy zmienna (bit) jest równa 0
– **JNZ nazwa** ;skok w odpowiednie miejsca programu (oznaczone nazwą), w przypadku gdy stan bitów akumulatora jest różny od 0
– **JZ nazwa** ;skok w odpowiednie miejsce programu (oznaczone nazwą), w przypadku gdy stan bitów akumulatora jest równy 0

SKOKI SPECJALNE
– CJNE arg1,arg2,nazwa ;funkcja ta dokonuje porównania dwóch argumentów, jeżeli są różne od siebie to następuje skok do miejsca w programie o podanej nazwie. W przypadku, gdy argumenty są równe program przechodzi do wykonywania następnej instrukcji po rozkazie CJNE.

– **DJNZ A, nazwa** ;instrukcja ta powoduje zmniejszenie zawartości akumulatora o jeden i jeżeli jego „zawartość jest różna od zera" to następuje skok do miejsca w programie o podanej nazwie. W przypadku, gdy „zawartość akumulatora jest równa zero" skok nie następuje i jest wykonywany rozkaz po instrukcji DJNZ.

5) Operacje obsługujące port szeregowy
Sterowanie transmisją szeregową w procesorach MCS–51 odbywa się za pomocą ustawień bitów rejestru specjalnego SCON.
Poszczególne bity tego rejestru mają następujące zadanie:
SCON.0 (RI) – bit odebrania znaku
SCON.1 (TI) – bit wysłania znaku
SCON.2 (RB8) – 9 bit transmisji odbieranego znaku
SCON.3 (TB8) – 9 bit transmisji wysyłanego znaku
SCON.4 (REN) – zezwolenie na odbiór danych
SCON.5 (SM2) – sterowanie komunikacją wieloprocesorową
SCON.6 (SM1) – ustawianie trybu pracy
SCON.7 (SM0) – ustawianie trybu pracy

5) Operacje nadzorujące pracą czasomierzy
W mikrokontrolerach pracą układów liczących sterują dwa rejestry specjalne: TMOD oraz TCON.
Zadania realizowane przez kolejne bity rejestru TMOD są następujące:
Licznik T0
TMOD.0 (M0) – tryb pracy
TMOD.1 (M1) – tryb pracy
TMOD.2 (C/T) – gdy bit C/T=0 to układ zachowuje się jak czasomierz, zliczający impulsy wewnętrzne
– gdy bit C/T=1 to układ zachowuje się jak licznik impulsów zewnętrznych
TMOD.3 (GATE) – kontrola nad pracą układu liczącego
– GATE =0 ,start i stop licznika odbywa się przez ustawienie bitu TRx(TR0=1) w słowie TCON
– GATE =1 ,start i stop licznika odbywa się przez ustawienie bitu TRx(TR0=1) w słowie TCON oraz dodatkowo o jego stanie pracy decyduje stan linii INTx (czyli, gdy TR0=1 i INT0=1 to następuje zliczanie impulsów)

Licznik T1
TMOD.4 (M0) – tryb pracy
TMOD.5 (M1) – tryb pracy
TMOD.6 (C/T) – gdy bit C/T=0 to układ zachowuje się jak czasomierz, zliczający impulsy wewnętrzne
– gdy bit C/T=1 to układ zachowuje się jak licznik impulsów zewnętrznych
TMOD.7 (GATE) – kontrola pracą układu liczącego
– GATE =0 ,start i stop licznika odbywa się przez ustawienie bitu TRx(TR1=1) w słowie TCON
– GATE =1 ,start i stop licznika odbywa się przez ustawienie bitu TRx(TR1=1) w słowie TCON oraz dodatkowo o jego stanie pracy decyduje stan linii INTx (czyli, gdy TR1=1 i INT1=1 to następuje zliczanie impulsów)

Tryby pracy układów liczących, uwzględniające stany znaczników M1 i M2 (poszczególnych bitów rejestru TMOD), zostały przedstawione w tabeli 4.3.1.

M0	M1	Rodzaj wykonywanej pracy
0	0	Tryb0; licznik 13 bitowy
0	1	Tryb 1; licznik 16 bitowy
1	0	Tryb 2; licznik 8 bitowy z automatycznym wpisywaniem wartości początkowej do rejestru THi
1	1	Tryb 3; licznik T1 nie pracuje, T0 pracuje jako dwa, 8 bitowe liczniki

Tabela 4.3.1. Tryby pracy układów liczących występujące w procesorze rodziny MCS-51

Zadnia realizowane przez kolejne bity rejestru TCON:
TCON.0 (IT0) – bit sterujący pracą przerwań
TCON.1 (IE0) – bit sterujący pracą przerwań
TCON.2 (IT1) – bit sterujący pracą przerwań
TCON.3 (IE1) – bit sterujący pracą przerwań
TCON.4 (TR0) – bit sterujący zliczaniem (licznik T0)
TCON.5 (TF0) – bit informujący o przepełnieniu licznika T0
TCON.6 (TR1) – bit sterujący zliczaniem (licznik T1)
TCON.7 (TF1) – bit informujący o przepełnieniu licznika T1

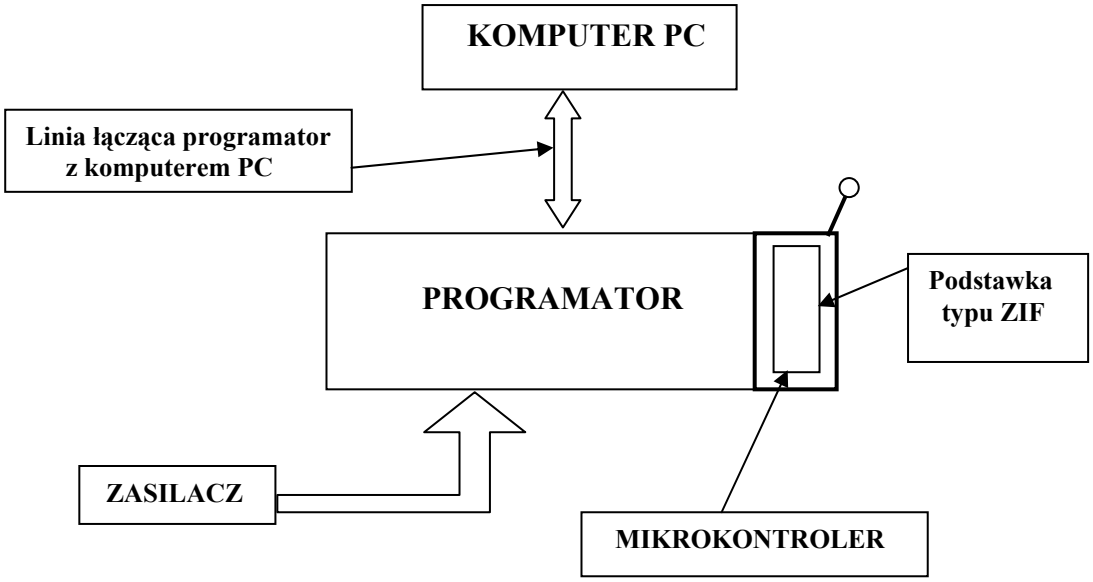
W przypadku, gdy bit TR1=0 i bit TR0=0 to następuje wstrzymanie pracy licznika, natomiast, gdy bit TR1=1oraz bit TR0=1 to następuje rozpoczęcie procesu zliczania.
Stany szesnastobitowych liczników wchodzących w skład układu czasowego mogą być modyfikowane programowo poprzez cztery rejestry specjalne:
TH0 – najbardziej znaczący bajt licznika T0
TL0 – mniej znaczący bajt licznika T0
TH1 – najbardziej znaczący bajt licznika T1
TL1 – mniej znaczący bajt licznika T1

6) Operacje obsługi przerwań

W mikrokontrolerach rodziny MCS - 51 istniej specjalny rejestr - IE nadzorujący pracą przerwań. Odpowiednie bity tego rejestru mają następujące znaczenie:
IE.0 (EX0) – zezwolenie na przerwanie z wejścia INT0
IE.1 (ET0) – zezwolenie na przerwanie z czasomierza T0
IE.2 (EX1) – zezwolenie na przerwanie z wejścia INT1
IE.3 (ET1) – zezwolenie na przerwanie z czasomierza T1
IE.4 (ES) – zezwolenie na przerwanie pochodzące ze sterownika transmisji szeregowej RS
IE.7 (EA) – ogólne zezwolenie na przerwanie

Programatory i proces programowania mikroprocesorów

Programatory to mniej lub bardziej skomplikowane urządzenia elektroniczne, których zadaniem jest wprowadzenie do wewnętrznej pamięci mikrokontrolera dowolnego programu. Złożoność budowy programatora jest związana z jego możliwościami, czyli liczbą obsługiwanych układów scalonych. Nieodzownym elementem (oprócz programatora) zestawu służącego do programowania jest komputer, zasilacz oraz linia łącząca te elementy. Schemat blokowy zestawu służącego do programowania mikrokontrolerów przedstawiono na rys.8.

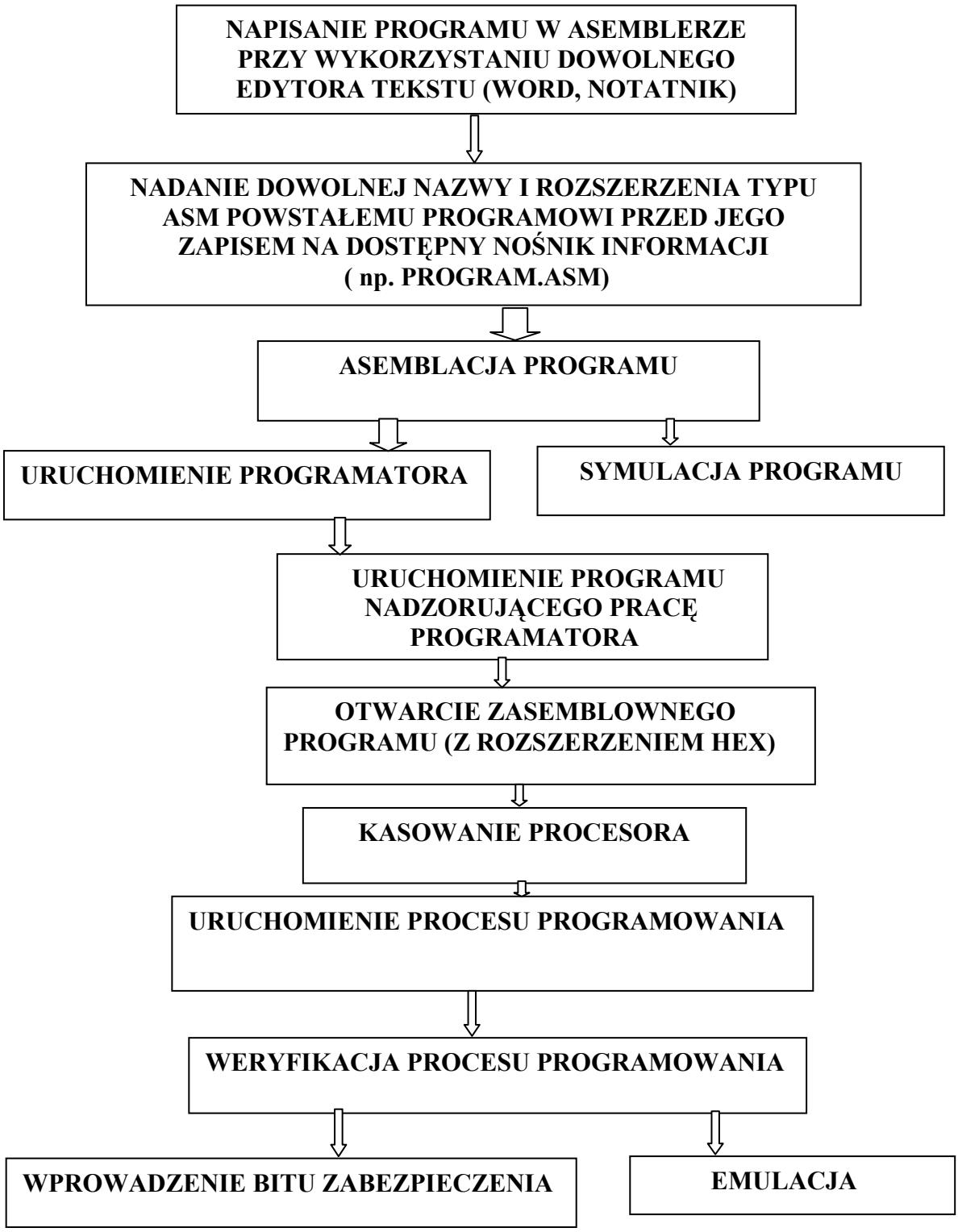


Rys.8. Schemat blokowy zestawu służącego do programowania mikrokontrolerów

W zestawach służących do programowania mikrokontrolerów stosowane są dwa sposoby łączenia ich z komputerem. Pierwszy sposób to zastosowanie łącza szeregowego, drugi to zastosowanie łącza typu równoległego.

Zestaw kolejnych czynności, jakie należy wykonać podczas programowania pokazany został na rysunku 9.

ZESTAW CZYNNOŚCI, JAKIE NALEŻY WYKONAĆ PODCZAS PROGRAMOWANIA MIKROKONTROLERÓW



Rys. 9. Algorytm programowania mikrokontrolerów

Czynności przedstawione na rysunku 9 najlepiej opisać na konkretnym przykładzie.

Przykład:

W oparciu o schemat zestawu uruchomieniowego z rys.11napisać program, którego wynikiem końcowym ma być zaświecenie się jednej diody – LED 1, dołączonej do czwartego wyjścia portu P1 mikrokontrolera typu 89S52.

Wydruk (listing) programu realizującego powyższe zadanie będzie wyglądał w następujący sposób:

LED.asm

```
ORG 0H
MOV P1,#00001111B; wyróżniony bit w danym zapisie jest odpowiedzialny za uaktywnienie
                  diody LED podłączonej do czwartego wyjścia portu P1

END
```

W wyniku asemblacji (użycia specjalnego programu) otrzymamy dwa, dodatkowe programy: pierwszy z heksadecymalnym kodem źródłowym – led.hex, ładowanym do pamięci procesora oraz drugi, pomocniczy typu: led.lst – informujący programistę o ewentualnie występujących w nim błędach. Listingi tych, dwóch programów przedstawiono poniżej.

led.hex

```
:0300000075900FE9
:00000001FF
```

led.lst

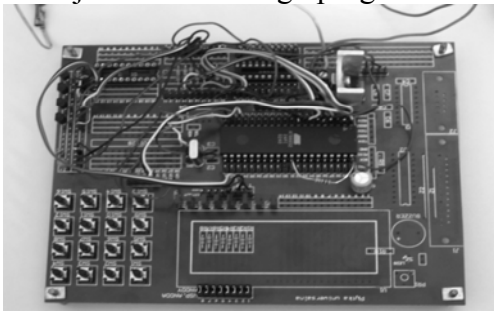
```
Line  I  Addr  Code        Source

   1:        N    0000    ORG 0H
   2:    0000  75 90 0F    MOV P1,#00001111B
   3:
   4:                      END
```

Kolejnym krokiem przy programowaniu (rys.9) jest symulacja komputerowa umożliwiająca sprawdzenie poprawności danego programu napisanego w asemblerze. W przypadku, gdy etap ten zakończy się sukcesem, można uruchomić programator i załadować program do umieszczonego w nim procesora. Należy pamiętać o tym, aby sprawdzić (zweryfikować) identyczność załadowanego programu z jego źródłową wersją.

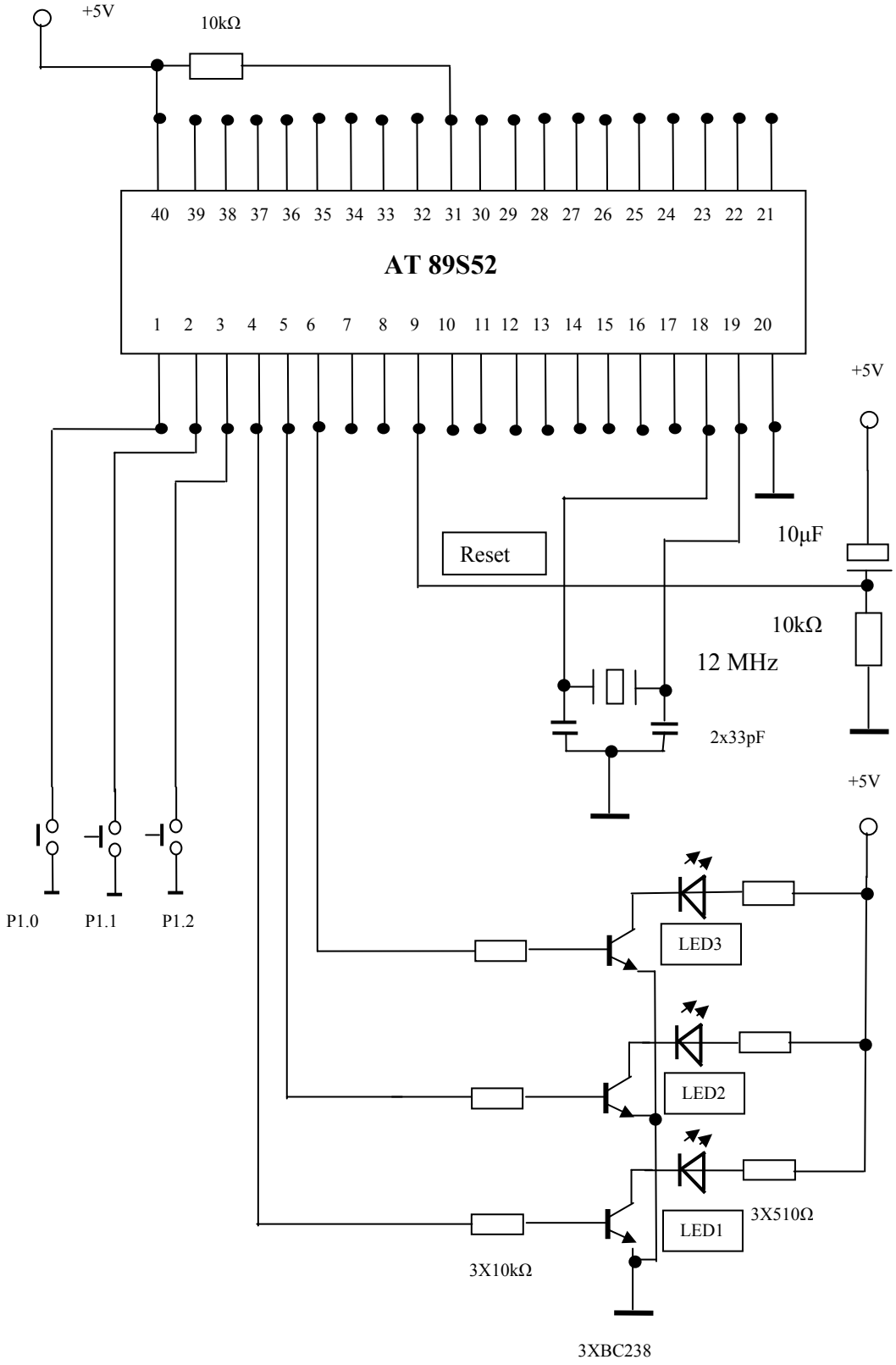
Procesor, z tak załadowanym programem, można umieścić dopiero w zestawie uruchomieniowym (rys. 10 i rys.11).

Zadaniem układów uruchomieniowych jest praktyczne sprawdzenie poprawności funkcjonowania danego programu w niemal rzeczywistych warunkach.



Rys 10. Zestaw uruchomieniowy

W celu zabezpieczenia własnej pracy można dodatkowo wprowadzić podczas procesu programowania do struktury wewnętrznej mikroprocesora bit zabezpieczenia. Uniemożliwia on odczytanie przez osoby postronne zawartości wewnętrznej pamięci mikroprocesora.



Rys. 11. Schemat „zestawu uruchomieniowego”

4.3.2. Pytania sprawdzające

Odpowiadając na pytania, sprawdzisz, czy jesteś przygotowany do wykonania ćwiczeń.

1. Co to jest asembler?
2. Jaka jest ogólna struktura programu napisanego w asemblerze?
3. Jak dzielimy instrukcje w asemblerze?
4. Jakie są elementy składowe zestawu służącego do programowania mikroprocesorów?
5. Jakie czynności należy wykonać, aby w poprawny sposób zaprogramować mikroprocesor?

4.3.3. Ćwiczenia

Ćwiczenie 1

Napisz program, którego wynikiem końcowym jest zaświecenie się dwóch diod LED (wykorzystaj schemat układu uruchomieniowego z rys.11).

Sposób wykonania ćwiczenia

Aby wykonać ćwiczenie powinieneś:

1) napisać program,
2) sprawdzić program przy wykorzystaniu dowolnego symulatora pracy mikroprocesora,
3) sprawdzić poprawność funkcjonowania mikroprocesora w układzie uruchomieniowym.

Wyposażenie stanowiska pracy:

- stanowisko do programowania mikroprocesorów,
- program symulacyjny,
- zestaw uruchomieniowy (rys.11).

Ćwiczenie 2

Napisz program, którego wynikiem końcowym jest zaświecenie się i przygasanie jednej diody LED (wykorzystaj schemat układu uruchomieniowego z rys.11).

Sposób wykonania ćwiczenia

Aby wykonać ćwiczenie powinieneś:

1) napisać program,
2) sprawdzić program przy wykorzystaniu dowolnego symulatora pracy mikroprocesora,
3) sprawdzić poprawność funkcjonowania mikroprocesora w układzie uruchomieniowym.

Wyposażenie stanowiska pracy:

- stanowisko do programowania mikroprocesorów,
- program symulacyjny,
- zestaw uruchomieniowy (rys.11).

Ćwiczenie 3

Napisz program, którego wynikiem końcowym jest zaświecenie się jednej diody LED w wyniku wciśnięcia jednego z przycisków (wykorzystaj schemat układu uruchomieniowego z rys.11).

Sposób wykonania ćwiczenia

Aby wykonać ćwiczenie powinieneś:
1) napisać program,
2) sprawdzić program przy wykorzystaniu dowolnego symulatora pracy mikroprocesora,
3) sprawdzić poprawność funkcjonowania mikroprocesora w układzie uruchomieniowym.

Wyposażenie stanowiska pracy:
- stanowisko do programowania mikroprocesorów,
- program symulacyjny,
- zestaw uruchomieniowy (rys.11).

Ćwiczenie 4

Napisz program, którego wynikiem końcowym jest:
a) zaświecenie diody LED1 po wciśnięciu przycisku P1.0,
b) zaświecenie diody LED2 po wciśnięciu przycisku P1.1 i upływie czasu około 1s,
c) zaświecenie diody LED3 po wciśnięciu przycisku P1.2 i upływie czasu około 2 s (wykorzystaj schemat układu uruchomieniowego z rys.11).

Sposób wykonania ćwiczenia

Aby wykonać ćwiczenie powinieneś:
1) napisać program,
2) sprawdzić program przy wykorzystaniu dowolnego symulatora pracy mikroprocesora,
3) sprawdzić poprawność funkcjonowania mikroprocesora w układzie uruchomieniowym.

Wyposażenie stanowiska pracy:
- stanowisko do programowania mikroprocesorów,
- program symulacyjny,
- zestaw uruchomieniowy (rys.11).

Ćwiczenie 5

Napisz program służący do zamiany liczby napisanej w kodzie dziesiętnym na liczbę zapisaną w kodzie szesnastkowym.

Sposób wykonania ćwiczenia

Aby wykonać ćwiczenie powinieneś:
1) napisać program,
2) sprawdzić program przy wykorzystaniu dowolnego symulatora pracy mikroprocesora.

Wyposażenie stanowiska pracy:
- zestaw komputerowy,
- program symulacyjny.

4.3.4. Sprawdzian postępów

Czy potrafisz:	**Tak**	**Nie**
1) zdefiniować pojęcie asembler?	☐	☐
2) wymienić podstawowe rozkazy asemblera?	☐	☐
3) zdefiniować pojęcie zestaw uruchomieniowy?	☐	☐
4) zdefiniować pojęcie programator?	☐	☐

5. SPRAWDZIAN OSIĄGNIĘĆ

INSTRUKCJA DLA UCZNIA

1. Sprawdź kompletność otrzymanych formularzy (powinieneś otrzymać: instrukcję dla ucznia, kartę odpowiedzi i zestaw pytań testowych).
2. Zapoznaj się z otrzymanymi dokumentami.
3. Otrzymany przez Ciebie test składa się z 9 pytań. Do każdego pytania dołączono po cztery odpowiedzi, z których tylko jedna jest poprawna.
   Za każdą poprawnie zakreśloną odpowiedź otrzymujesz po jednym punkcie.
   Zła odpowiedź lub jej brak jest równoznaczne z otrzymaniem przez Ciebie 0 punktów.
   Maksymalnie możesz uzyskać 9 punktów.
4. Przed przystąpieniem do udzielania odpowiedzi podpisz imieniem i nazwiskiem otrzymaną kartę odpowiedzi.
5. Odpowiedzi udzielaj na otrzymanej karcie odpowiedzi stawiając w odpowiedniej rubryce znak X.
6. W momencie gdy stwierdzisz, że zakreślona odpowiedź nie jest poprawna możesz dokonać korekty zaznaczając błędną odpowiedź kółkiem, a przy dobrej odpowiedzi ponownie postaw znak X.
7. W przypadku gdy nauczyciel stwierdzi niesamodzielność w rozwiązywaniu przez Ciebie zadań testowych, stracisz możliwość uzyskania oceny pozytywnej.
8. Na rozwiązanie zadań testowych przewidziano 30 minut.
9. Proponuje się następujące normy wymagań – uczeń otrzymuje następujące oceny szkolne:

Bardzo dobry	**9 punktów**
Dobry	**8–7 punktów**
Dostateczny	**6–5 punktów**
Dopuszczający	**4 punkty**
Niedostateczny	**1–3 punkty**

10. W przypadku jakichkolwiek wątpliwości proś o pomoc nauczyciela.

Powodzenia!

Zestaw pytań testowych

1. Co to jest mikroprocesor?
   a) układ scalony, którego funkcjonowanie jest sterowane pobieranymi z pamięci rozkazami,
   b) układ scalony o wielkiej skali integracji,
   c) układ scalony sterujący małymi procesami przemysłowymi,
   d) układ scalony, na którego funkcjonowanie nie może wpływać jego użytkownik.

2. System mikroprocesorowy jest to:
   a) układ scalony przetwarzający sygnały cyfrowe,
   b) mikroprocesor wraz z dodatkowymi elementami zewnętrznymi, z którymi on współpracuje,
   c) mikroukład sterowany pobieranymi z zewnątrz rozkazami,
   d) sieć przewodów łączących dwa mikroprocesory.

3. Pamięć ROM jest pamięciom:
   a) którą można programować i kasować elektrycznie,
   b) którą można kasować promieniowaniem ultrafioletowym,
   c) którą można tylko czytać,
   d) której zawartość jest bezpowrotnie tracona po wyłączeniu napięcia zasilającego.
4. Stosem nazywamy:
   a) zestaw baterii tworzących akumulator,
   b) pamięć typu FLASH,
   c) szczególny rodzaj pamięci – tzw. pamięci książkowej,
   d) dodatkową pamięć występującą w systemie mikroprocesorowym.
5. Zastosowanie szeregowej transmisji danych:
   a) powoduje zwiększenie szybkość transferu informacji
   b) powoduje zmniejszenie szybkość transferu informacji.
   c) powoduje zwiększenie podatność transferu informacji na zakłócenia zewnętrzne.
   d) pozwala na przekaz informacji na niewielkie odległości.
6. Akumulator jest to:
   a) ogniwo cynkowo-węglowe,
   b) bateria podtrzymująca napięcie w systemie mikroprocesorowym,
   c) pamięć typu EEPROM,
   d) główny rejestr wewnętrzny mikroprocesora przechowujący wszelkie wyniki przeprowadzanych operacji arytmetyczno-logicznych.
7. Przerwanie jest to:
   a) zawieszenie wykonywania aktualnego programu na rzecz wykonania programu o wyższym priorytecie,
   b) inaczej próbkowanie, czyli testowanie stanu wejść mikrokontrolera,
   c) zawieszenie wykonywania danego programu oraz wstrzymanie pracy mikroprocesora,
   d) reset procesora w chwili braku sygnałów pochodzących z zewnątrz.
8. Asembler jest to
   a) język programowania mikroprocesora
   b) pamięć wewnętrzna mikroprocesora.
   c) zestaw operacji arytmetyczno-geometrycznych wykonywanych przez procesor.
   d) program zapisany w systemie binarnym.
9. Programator jest to
   a) urządzenie służące do sprawdzania programów napisanych w asemblerze.
   b) element systemu mikroprocesorowego.
   c) urządzenie służące do programowania mikroprocesorów.
   d) układ nadzorujący sekwencyjne wykonywanie instrukcji wchodzących w skład danego programu.

KARTA ODPOWIEDZI

Imię i nazwisko.....

Analizowanie działania układów mikroprocesorowych

Zakreśl poprawną odpowiedź.

Nr zadania	***Odpowiedź***				**Punkty**
1	a	b	c	d	
2	a	b	c	d	
3	a	b	c	d	
4	a	b	c	d	
5	a	b	c	d	
6	a	b	c	d	
7	a	b	c	d	
8	a	b	c	d	
9	a	b	c	d	
				Razem:	

6. LITERATURA

1. Barlik R, Nowak M.: Układy sterowania i regulacji urządzeń energoelektronicznych. WSiP, Warszawa 1998
2. Dokumentacja techniczna firmy ATMEL 2003 (www.atmel.com)
3. Dokumentacja techniczna firmy Hitachi 1999 (www.hitachi.com)
4. Dokumentacja techniczna firmy INTEL 2005 (www.intel.com)
5. Gałka Piotr, Gałka Paweł.: Podstawy programowania mikrokontrolera 8051. Wyd. „MIKOM", Warszawa 2005
6. Jamniczek J., Stępień A.: Systemy mikroprocesorowe. Laboratorium systemów mikroprocesorowych cz. I i II. Wydawnictwo Elektronicznych Zakładów Naukowych, Wrocław 1995
7. Niederliński A.:Mikroprocesory, Mikrokomputery, Mikrosystemy Wyd II WSiP, Warszawa 1988
8. Rydzeski A., Sacha K.: Mikrokomputer. Elementy, budowa, działanie SIGMA, Warszawa 1987
9. Rydzewski A.: Mikrokomputery jednoukładowe rodziny MCS–51. WNT, Warszawa 1999