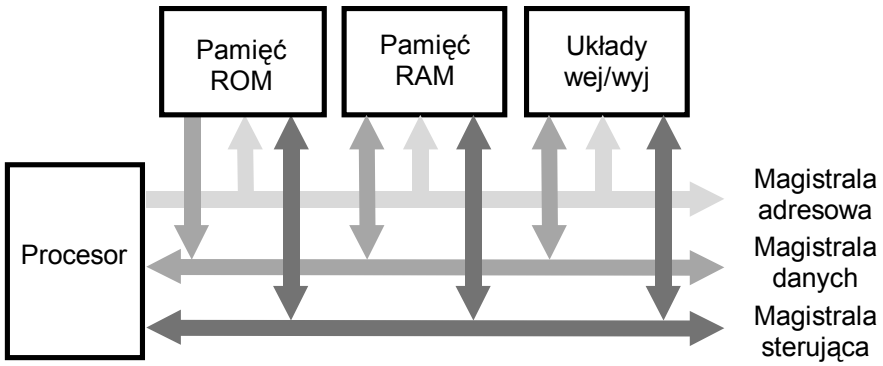
Podstawy Techniki Mikroprocesorowej - Laboratorium
Ćwiczenie A

Temat: Mikroprocesor Z80

Cel i przebieg ćwiczenia: Ćwiczenie wprowadzające do cyklu zajęć z Dydaktycznym Systemem Mikroprocesorowym (DSM). Zapoznanie się z architekturą kasety i wkładek DSM oraz z programem obsługi monitora szyny systemu. Praktyczna realizacja dekodera przestrzeni adresowej pamięci mikroprocesora Z80 - dekodowanie adresów pamięci RAM i ROM. Napisanie i uruchomienie prostego programu w asemblerze procesora Z80 wykonującego operacje zapisu/odczytu do komórek pamięci.

Wiadomości teoretyczne: Ogólną strukturę systemu mikroprocesorowego przedstawiono na Rys. 1 (architektura von Neumanna). Elementy składowe systemu - mikroprocesor, pamięć ROM i RAM oraz układy wejścia/wyjścia są połączone przy pomocy trzech magistral:

- adresowej
- danych
- sterującej



Rys. 1. Schemat blokowy systemu mikroprocesorowego

Na magistrali adresowej mikroprocesor wystawia adres komórki pamięci lub układu wejścia/wyjścia, z którego odczytuje informację względnie, do którego zapisuje informację. Istnieją mikroprocesory z rozdzielnym adresowaniem pamięci i układów wejścia/wyjścia (o tym czy adres dotyczy pamięci czy wejścia/wyjścia decyduje wyróżniony sygnał magistrali sterującej) lub z łącznym adresowaniem pamięci i układów wejścia/wyjścia.

Magistrala danych służy do przesyłania informacji między mikroprocesorem a pamięcią lub układami wejścia/wyjścia.

Magistrala sterująca pozwala mikroprocesorowi sterować bezpośrednio pracą pamięci i układów wejścia/wyjścia.

Przy porównywaniu i klasyfikacji różnych typów mikroprocesorów bierze się przede wszystkim pod uwagę długość słowa, na którym operują (jest to zwykle szerokość ich magistrali danych).

Jednym z najpopularniejszych mikroprocesorów 8-bitowych jest procesor Z80. Został on opracowany na przełomie lat 1976/1977 w firmie Zilog przez zespół konstruktorów, którzy wcześniej brali udział w stworzeniu mikroprocesora Intel 8080. Założeniem, które przyświecało twórcom Z80 była programowa zgodność nowego

mikroprocesora z 8080 - pozwoliło to na wykorzystanie istniejącego oprogramowania pracującego pod systemem CP/M.

Mikroprocesor Z80 jest 8-bitowym procesorem wyposażonym w duży zestaw rejestrów wewnętrznych. Pozwala to na przechowywanie większej ilości informacji wewnątrz mikroprocesora i minimalizuje konieczność odwołań do pamięci. Mikroprocesor Z80 adresuje 65536 komórek pamięci i tyle samo urządzeń wejścia/wyjścia (rozdzielone przestrzenie adresowe). Mikroprocesor wyposażony jest w rozbudowany system przerwań i możliwość generowania adresu odświeżania co ułatwia dołączenie do niego pamięci dynamicznych.

Z punktu widzenia użytkownika-konstruktora i programisty szczegóły budowy wewnętrznej mikroprocesora są drugorzędne. Istotne jest jakie rejestry dostępne programowo zawiera mikroprocesor, jego lista rozkazów i sposób dołączenia układów zewnętrznych.

Rejestry mikroprocesora Z80 można podzielić na trzy grupy:

- akumulator A i rejestr wskaźników F
- rejestry uniwersalne B, C, D, E, H, L
- rejestry specjalne IX, IY, SP, PC

Akumulator zawiera jeden z argumentów każdej operacji arytmetycznej lub logicznej (drugi argument pobierany jest z pamięci lub rejestru uniwersalnego). Do akumulatora zapisywany jest również wynik operacji.

Rejestr wskaźników F zawiera słowo stanu procesora - sześć jednobitowych znaczników (Rys. 2) przechowujących pomocniczą informację o wyniku ostatnio wykonanej operacji.

7	6	5	4	3	2	1	0
S	Z	-	H	-	P/V	N	C

Rys. 2. Rejestr wskaźników

Znaczenia poszczególnych bitów rejestru F:

C - bit przeniesienia - ustawiany gdy w wyniku ostatniej operacji nastąpiło przekroczenie zakresu liczb w NB

N - bit odejmowania - ustawiany gdy ostatnią operacją było odejmowanie (operacje na liczbach w kodzie BCD)

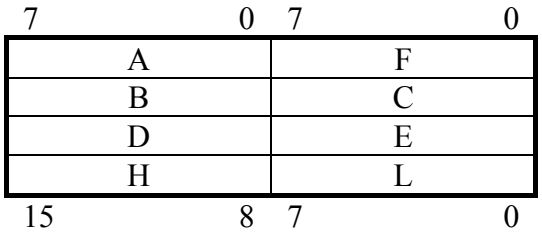
P/V - bit nadmiaru/parzystości - ustawiany gdy w wyniku ostatniej operacji wystąpił nadmiar (operacje arytmetyczne) lub otrzymany rezultat zawiera parzystą liczbę jedynek (operacje logiczne)

H - bit przeniesienia pomocniczego - w wyniku ostatniej operacji wystąpiło przeniesienie między bitami b3 i b4 (operacje na liczbach w kodzie BCD)

Z - bit zera - ustawiany gdy wynikiem ostatniej operacji jest zero

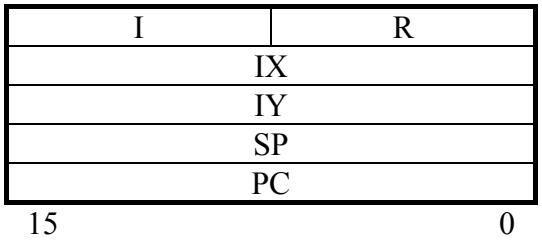
S - bit znaku - kopia najbardziej znaczącego bitu wyniku

Rejestry uniwersalne (Rys. 3) zawierają zwykle argumenty rozkazów wykonywanych przez mikroprocesor, przechowują pośrednie wyniki obliczeń. Mogą być także wykorzystane parami BC, DE, HL jako rejestry 16-bitowe. Mikroprocesor Z80 posiada dwa alternatywne zestawy rejestrów, z których każdy zawiera sześć rejestrów uniwersalnych, akumulator i rejestr wskaźników. Zawartość rejestrów zestawu głównego i pomocniczego może być wymieniana.



Rys. 3. Rejestry uniwersalne Z80.

W grupie rejestrów specjalnych (Rys. 4) najważniejszą rolę spełnia licznik rozkazów PC, który przechowuje adres komórki pamięci, z której będzie pobrany kod rozkazu do wykonania. Rejestr wskaźnika stosu SP przechowuje adres wierzchołka stosu (stos jest fragmentem pamięci gdzie mikroprocesor umieszcza adres powrotu przy wywołaniu podprogramu lub przy zgłoszeniu przerwania). Rejestry IX, IY zawierają adres bazowy przy adresowaniu indeksowym. Rejestr I wykorzystywany jest przez system przerwań mikroprocesora Z80, rejestr R może być użyty do odświeżania pamięci dynamicznej.



Rys. 4. Rejestry specjalne Z80

Rozkazy wykonywane przez mikroprocesor Z80 można podzielić na cztery grupy:

- rozkazy przesłań (przesłania danych między rejestrami mikroprocesora oraz między rejestrami i pamięcią)
- rozkazy arytmetyczne i logiczne (dodawanie i odejmowanie, suma i iloczyn logiczny, porównywanie, przesunięcia arytmetyczne i logiczne, ustawianie i kasowanie bitów)
- rozkazy sterujące wykonaniem programu (rozkazy skoków i wywołań podprogramów, rozkazy związane z przerwaniami)
- rozkazy wejścia/wyjścia (odczyt i zapis układów wejścia/wyjścia)

Ze względu na bardzo obszerną listę rozkazów mikroprocesora Z80 (i wiele dostępnych trybów adresowania) nie zostanie ona omówiona w instrukcji. Lista rozkazów Z80 jest dostępna w wielu pozycjach literaturowych - szczególnie można tu polecić [6]. Przykładowy program w języku asemblera Z80 przedstawiono w przykładzie 3.

Wszystkie rozkazy Z80 realizowane są w cyklu rozkazowym składającym się z następujących faz:

1. pobranie kodu rozkazu z komórki pamięci o adresie zawarty w PC
2. zdekodowanie rozkazu
3. wykonanie rozkazu (zmiana zawartości pamięci lub rejestrów procesora)
4. zwiększenie licznika rozkazów o liczbę zależną od długości rozkazu

Cykl rozkazowy składa się z cykli maszynowych - pojedynczych odwołań mikroprocesora do pamięci lub układów wejścia/wyjścia. Cykle maszynowe składają się z cykli zegarowych (taktów zegara mikroprocesora). Mikroprocesor Z80 może realizować osiem rodzajów cykli maszynowych:

1. cykl pobrania i dekodowania kodu rozkazu (cykl M1)
2. cykl odczytu lub zapisu pamięci

3. cykl odczytu lub zapisu układu wejścia/wyjścia
4. cykl przyjęcia żądania dostępu do magistrali
5. cykl przyjęcia przerwania maskowalnego
6. cykl przyjęcia przerwania niemaskowalnego
7. cykl zatrzymania (po wykonaniu rozkazu HALT)
8. cykl zerowania (sygnałem RESET)

Dane katalogowe mikroprocesora Z80 zawierają szczegółowe parametry czasowe poszczególnych cykli - sekwencje sygnałów na magistralach mikroprocesora. Do właściwego połączenia procesora z pamięcią lub układami wejścia/wyjścia najważniejsza jest jednak znajomość funkcji istotnych sygnałów magistrali sterującej.

Podstawowe sygnały magistrali sterującej Z80:

RESET - sygnał ustawienia mikroprocesora w stan początkowy (wyzerowanie licznika rozkazów)

HALT - mikroprocesor wykonał rozkaz HALT i znajduje się w stanie zatrzymania

M1 - pobranie kodu rozkazu

MREQ (Memory Request) - żądanie dostępu do pamięci - mikroprocesor będzie odczytywał informację z komórki pamięci (lub zapisywał informację do komórki pamięci) o adresie podanym na magistralę adresową

IORQ (I/O Request) - żądanie dostępu do układów wejścia/wyjścia - mikroprocesor będzie odczytywał informację z układu wejścia/wyjścia (lub zapisywał informację do układu wejścia/wyjścia) o adresie podanym na magistralę adresową

RD (Read) - odczyt z pamięci lub układu wejścia/wyjścia

WR (Write) - zapis do pamięci lub układu wejścia/wyjścia

Klasyfikacja pamięci półprzewodnikowych.

Pamięci półprzewodnikowe można ogólnie podzielić na dwie grupy - pamięci stałe (ROM - Read Only Memory) i pamięci zapisywalne (RAM - Random Access Memory). Podstawą tego rodzaju podziału jest zachowanie się układu pamięci po odłączeniu zasilania. Pamięci stałe przechowują zapisaną informację również po wyłączeniu zasilania, natomiast w pamięciach zapisywalnych informacja jest tracona. Z tego względu nie można konstruować systemu mikroprocesorowego wyposażonego wyłącznie w pamięć RAM - każdy zanik napięcia zasilającego prowadziłby do skasowania programu. Dlatego każdy system mikroprocesorowy musi zawierać pamięć ROM, w której zawarty jest program (w mikrokomputerach specjalizowanych) lub podstawowe procedury systemu operacyjnego (w mikrokomputerach uniwersalnych).

Do pamięci stałych zalicza się pamięci ROM (programowane przez producenta w procesie technologicznym - nie ma możliwości zmiany wpisanej zawartości), PROM (programowane przez użytkownika przy pomocy specjalnego programatora - brak możliwości modyfikacji wpisanej zawartości), EPROM (programowane przez użytkownika przy pomocy programatora - można wielokrotnie kasować zawartość promieniowaniem UV i ponownie programować), E^2PROM (proces kasowania oraz zapisu odbywa się na drodze elektrycznej i można go powtarzać).

Pamięci zapisywalne można podzielić na dwie podstawowe grupy: pamięci statyczne SRAM (elementem pamiętającym jest przerzutnik bistabilny) i pamięci dynamiczne DRAM (elementem pamiętającym jest pojemność rozproszona tranzystora polowego).

Naturalnym procesem występującym w pamięciach dynamicznych jest rozładowywanie się pojemności pamiętającej - dlatego pamięci DRAM wymagają odświeżania. Polega ono na odczytaniu, co pewien czas, wszystkich komórek pamięci. Mimo niedogodności związanych z odświeżaniem, pamięci dynamiczne są chętnie stosowane (z uwagi na mniejszą cenę za 1 bit w porównaniu z pamięciami statycznymi) w systemach zawierających pamięć RAM o pojemności rzędu MB.

Do zalet pamięci dynamicznych (w porównaniu ze statycznymi) należą także mniejszy pobór mocy i większa pojemność na tej samej powierzchni.

Istnieją również pamięci pseudostatyczne stanowiące połączenie pamięci dynamicznej z układem odświeżania w jednym układzie scalonym - z zewnątrz pamięć taką można traktować jak pamięć statyczną.

W przedstawionej klasyfikacji nie mieszczą się pamięci nieulotne NVRAM (NonVolatiIe RAM). Jest to połączenie pamięci RAM z pamięcią E^2PROM - włączenie zasilania powoduje przepisanie zawartości E^2PROM do RAM-u, a zanik zasilania - zaprogramowanie pamięci stałej zawartością RAM-u. Pamięci takie stosuje się w systemach o podwyższonej niezawodności.

Obecnie skrótem NVRAM określa się także pamięci E^2PROM przechowujące parametry konfiguracyjne programowalnego sprzętu elektronicznego.

Podstawowe parametry pamięci półprzewodnikowych:

- pojemność (liczba bitów jakie można zapisać w danym module pamięci – kb, Mb)
- organizacja (liczba bitów w jednym słowie pamięci - 1b, 4b, 8b)
- czas dostępu

O ile dwa pierwsze parametry są dosyć oczywiste i jednoznaczne, o tyle czas dostępu jest parametrem mającym kilka różnych definicji - ściślej definiuje się kilka różnych czasów dostępu. Najistotniejszy praktycznie jest czas dostępu rozumiany jako odstęp czasu od podania adresu na wejścia modułu pamięci do pojawieniem się stabilnych danych na magistrali danych. W katalogach bywa on oznaczany jako t_{AVQV} lub krótko t_a. Podstawowe parametry dwóch rodzajów pamięci półprzewodnikowych (używanych w ćwiczeniu) przedstawiono w Tab. 1.

Tab. 1. Parametry pamięci półprzewodnikowych

	6264-12	6264-10	2716	27A16
pojemność [b]	65536		16384	
długość słowa [b]	8		8	
liczba słów	8192		2048	
t_{AVQV} [ns]	120	100	450	350
t_{GLVQV} [ns]	60	50	120	120

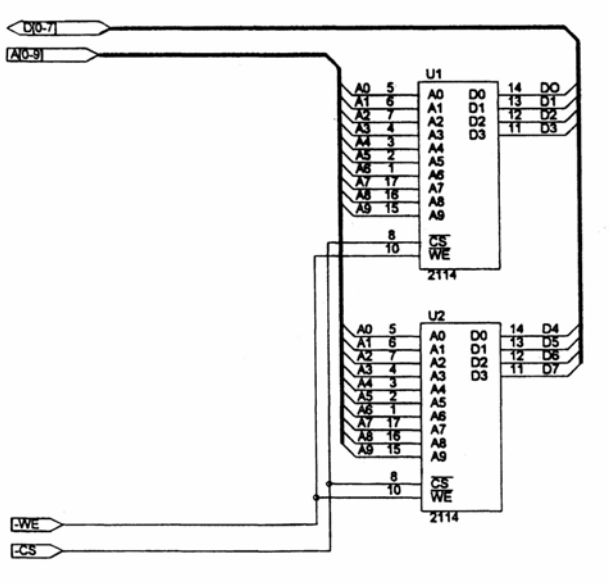
Przy konstruowaniu pamięci należy uwzględnić dwa wymagania:

- słowo pamięci musi mieć tyle bitów ile magistrala danych mikroprocesora
- każda komórka pamięci musi mieć inny adres - adresowanie komórek musi być jednoznaczne (czyli dwie komórki pamięci nie mogą być „widziane" pod jednym adresem)

Pierwsze wymaganie łatwo spełnić dobierając moduły pamięci o odpowiedniej długości słowa - dla mikroprocesorów 8-bitowych (jak Z80) należy zastosować pamięć o organizacji bajtowej. Możliwe jest również użycie modułów pamięci o słowie 1-bitowym lub 4-bitowym. Należy jednak wówczas dokonać równoległego połączenia ośmiu lub

dwóch modułów pamięci. Przykład takiego połączenia dla pamięci 2114 (1024x4b) przedstawiono na (Rys. 5).

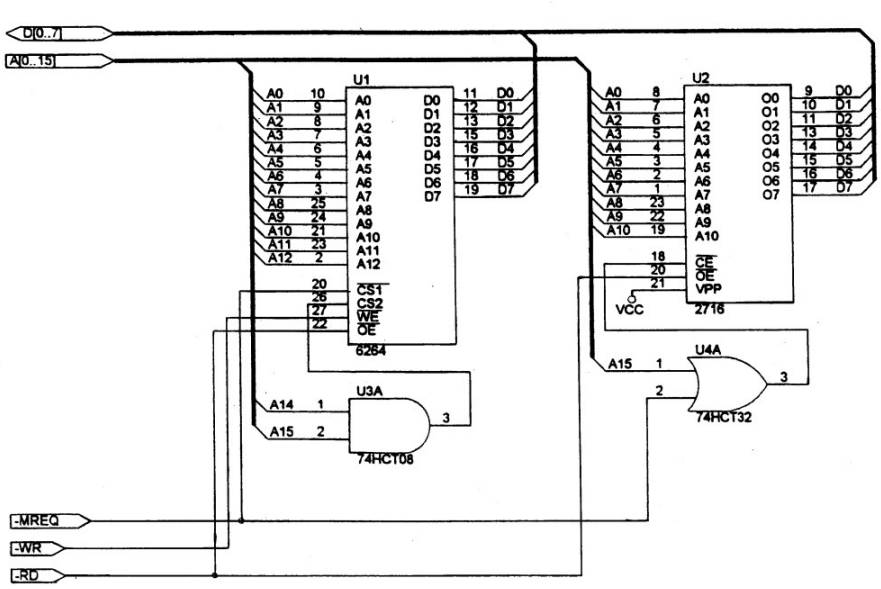
Do zapewnienia jednoznacznego adresowania pamięci stosuje się podział magistrali adresowej systemu mikroprocesorowego (Z80 ma 16-bitową szynę adresową). Młodsze bity magistrali wybierają komórkę pamięci wewnątrz modułu, starsze są doprowadzone do dekodera adresów, który wybiera poszczególne moduły pamięci. Najłatwiej dekoder adresów można zbudować przy pomocy scalonego dekodera - wówczas można zwykle zastosować pełne dekodowania adresów tzn. każda komórka pamięci ma jeden i tylko jeden adres. W prostych systemach mikroprocesorowych zwykle nie ma potrzeby wykorzystania całej przestrzeni adresowej. Stosuje się wówczas dekodery zbudowane z bramek logicznych, które nie dekodują wszystkich linii magistrali adresowej - jest to tzw. niepełne dekodowanie adresów. Oznacza to, że dana komórka pamięci ma więcej niż jeden adres - nie narusza to jednak warunku jednoznaczności.



Rys. 5. Łączenie modułów pamięci

Przykład 1. Zaprojektować dekoder adresów przestrzeni pamięci systemu mikroprocesorowego opartego na procesorze Z80. Układ ma zapewniać dekodowanie pamięci RAM 6264 i pamięci EPROM 2716. Opracować wersję dekodera z pełnym dekodowaniem adresów (użyć dekoderów 74HCT138 i 74HCT139) i z niepełnym dekodowaniem adresów (użyć elementów SSI).

Jako pierwsza przedstawiona zostanie wersja dekodera z niepełnym dekodowaniem adresów. Zakładając, że w pamięci EPROM umieszczony jest program realizowany przez system mikroprocesorowy konieczne jest zapewnienie dekodowania tej pamięci od adresu 0000h. Naturalny jest zatem wstępny podział przestrzeni adresowej pamięci procesora na dwa obszary: 0000h-7FFFh i 8000h-FFFFh. W pierwszym obszarze zostanie umieszczona pamięć EPROM wybierana sygnałami MREQ i A15 - będzie widoczna pod adresami: 0000h-07FFh, 0800h-0FFFh, 1000h-17FFh, 1800h-1FFFh, ..., 7800h-7FFFh. W analogiczny sposób można zapewnić dekodowanie pamięci RAM w obszarze 8000h-FFFFh. Warto jednak użyć dodatkowo sygnału A14 i pozostawić niewykorzystany obszar 16 kB pamięci (8000h-BFFFh) dla ewentualnej rozbudowy systemu. Tak więc pamięć RAM będzie widoczna w ostatnich 16 kB przestrzeni adresowej (w dwóch obszarach: C000h-DFFFh i E000h-FFFFh). Schemat dekodera przedstawiono na Rys. 6.

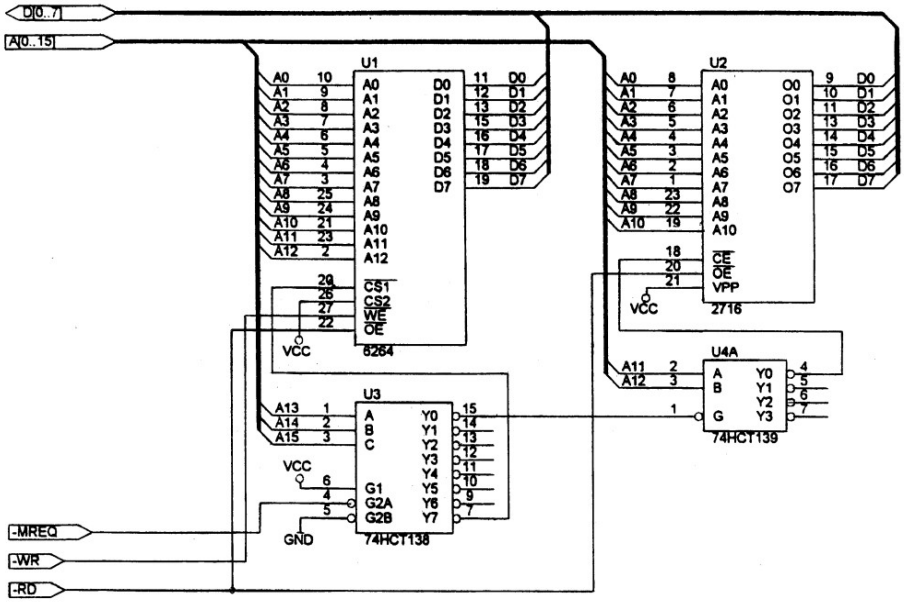


Rys. 6. Dekoder adresów - wersja 1 (z niepełnym dekodowaniem adresów)

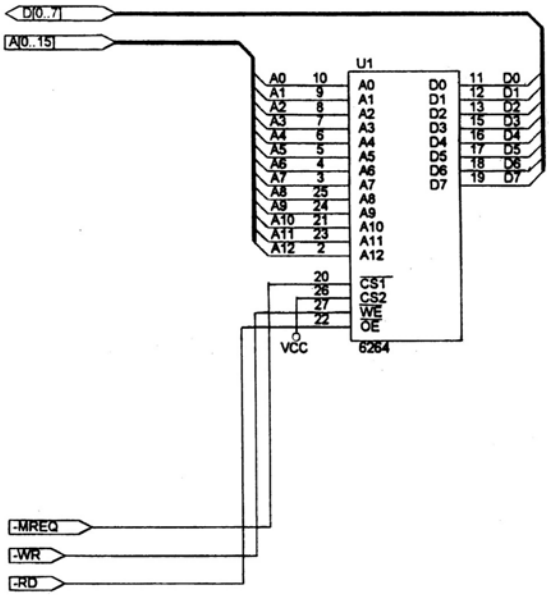
Wersja dekodera z pełnym dekodowanie adresów powinna oczywiście spełniać analogiczne założenie jak wersja z niepełnym dekodowanie adresów (dekodowanie pamięci EPROM od adresu 0000h). Mając do dyspozycji dekoder 3/8 (74HCT138) można wykorzystać go do podziału przestrzeni adresowej pamięci (o długości 64 kB) na bloki o długości 8 kB. Jeżeli przyjąć, że pamięć RAM ma być dekodowana w ostatnim takim bloku (E000h-FFFFh) to na wejście wybierające modułu 6264 należy podłączyć wyjście Y7 dekodera. Przy niepełnym dekodowaniu adresów do wybierania pamięci EPROM wystarczyłoby użyć wyjścia Y0 dekodera 74HCT138 - oznaczałoby to jednak, że pamięć 2716 jest dekodowana pod adresami 0000h-07FFh, 0800h-0FFFh, 1000h-17FFh, 1800h-1FFFh. Dla zrealizowania dekodera z pełnym dekodowaniem adresów należy użyć dodatkowo układu 74HCT139 (dekoder 2/4). Jego zadaniem będzie umieszczenie modułu pamięci 2716 w odpowiednim miejscu pierwszego bloku przestrzeni adresowej pamięci (o długości 8 kB). Do podziału tego bloku na dwa o długości 4 kB należy wykorzystać sygnał A12 (2^{12}=4096), a otrzymany blok o długości 4 kB można dalej podzielić na dwie połowy przy użyciu sygnału A11 (2^{11}=2048). Schemat dekodera przedstawiono na Rys. 7.

Przykład 2. Zaprojektować (maksymalnie prosty) sposób dekodowania pamięci RAM 6264 w przestrzeni adresowej mikroprocesora Z80. Pamięć ma być widoczna w obszarze 0000h-1FFFh (i ewentualnie w innych obszarach).

Umieszczenie pamięci RAM poczynając od adresu 0000h ma sens jedynie w systemie mikroprocesorowym, który ma odpowiednie mechanizmy zapisu tej pamięci (konieczne jest bowiem umieszczenie w pamięci programu realizowanego przez mikroprocesor). System DSM wykorzystywany w ćwiczeniu posiada takie mechanizmy - zapis do pamięci RAM możliwy jest dzięki monitorowi szyny. Dążąc do maksymalnej prostoty można „dekoder" zbudować beż użycia żadnych funktorów logicznych. Wystarczy na wejście wybierające pamięci podać sygnał MREQ i właściwie podłączyć sygnały RD i WR. Odpowiedni schemat przedstawiono na Rys. 8.
Taki sposób dekodowania wykorzystany będzie w kolejnych ćwiczeniach laboratoryjnych.



Rys. 7. Dekoder adresów - wersja 2 (z pełnym dekodowaniem adresów)



Rys. 8. Dekodowanie pamięci 6264

Przykład 3. Napisać program w asemblerze mikroprocesora Z80 do testowania pamięci RAM 6264 dekodowanej w obszarze 0000h-1FFFFh.

Program testujący pamięć RAM powinien po zapisaniu do komórki pamięci wybranego wzorca, odczytać tą komórkę i porównać odczytaną wartość ze wzorcem. W przypadku wystąpienia niezgodności powinien być sygnalizowany błąd. Jeżeli wartość odczytana jest identyczna ze wzorcem sprawdzanie można kontynuować dla kolejnych komórek testowanego bloku.

```
;Przykładowy programu dla Z80 do testowania pamięci RAM
PROG:                           ;nagłówek programu
        ORG 0000H               ;dyrektywa asemblera – program zostanie
                                ;umieszczony w pamięci od adresu 0000h
        LD SP,2000H             ;ustawienie początkowej wartości wskaźnika stosu
                                ;(po RESET SP zawiera wartość nieokreśloną)
        LD BC,1000H             ;liczba komórek pamięci do sprawdzenia
        LD HL,0100H             ;adres początku testowanego bloku pamięci
L0:
        LD A,0AAH               ;wartość testowa
        LD (HL),A               ;zapisanie wartości testowej do komórki pamięci
        LD A, (HL)              ;odczytanie zawartości komórki pamięci
        CP 0AAH                 ;sprawdzenie czy wartość odczytana jest identyczna z
                                ;zapisaną
        JP NZ,ERR               ;
        INC HL                  ;
        DEC BC                  ;
        CALL SUB                ;
        JP NZ,L0                ;
OK:     HALT                    ;pamięć pracuje poprawnie
ERR:    HALT                    ;uszkodzenie w komórce o adresie zawartym w HL

SUB:                            ;podprogram sprawdzania BC=0
                                ;(dekrementacja BC nie zmienia rejestru wskaźników)
        LD A,B                  ;
        CP 00H
        JP NZ,L1                ;
        LD A,C
        CP 00H
L1:     RET
        END                     ;dyrektywa asemblera – zaznaczenie końca programu
;Program kończymy END w polu instrukcji i "ENTER"
;Program musi mieć rozszerzenie .ASM np. ZEGAR.ASM
;Asemblacje uruchamiamy XASMZ80.COM ZEGAR.ASM
```

Modułowy system mikroprocesorowy DSM jest specyficznym środkiem uruchomieniowym układów mikroprocesorowych, w którym magistrale wyprowadzono w tylnej części kasety. Uniwersalność DSM polega na programowym (poprzez plik konfiguracyjny) określaniu tzw. definicji szyny, pozwalającej modułowi monitora BM01 współpracować z różnymi typami jednostek centralnych uruchamianych systemów. Typowe funkcje monitora szyny to:

- monitorowanie stanu magistral adresowej, danych i sterującej
- zapis/odczyt do pamięci lub układów we/wy uruchamianego systemu
- realizacja wykonania programu w trybie pracy krokowej, wolnobieżnej i z "pułapkami"

Monitor szyny nie umożliwia użytkownikowi na wgląd w rejestry wewnętrzne procesora.

Projekt wstępny: Zaprojektować dekoder adresów przestrzeni pamięci systemu mikroprocesorowego opartego na procesorze Z80. Układ ma w sposób pełny dekodować pamięć RAM (6264) od adresu 0000h oraz pamięć EPROM (2716) od arbitralnie wybranego adresu poczynając od 2000h (np. 3800h, A0000h lub D800h). Do budowy dekodera wykorzystać dwa układy: 74HCT138 i 74HCT139 znajdujące się na pakiecie MEM8. Przygotować schemat ideowy i montażowy (uwzględniający numery

wyprowadzeń elementów) zaprojektowanego układu. (Wskazówka: zadany układ można otrzymać przez modyfikację dekodera adresów zaprojektowanego w przykładzie 2). Napisać program w asemblerze procesora Z80 przepisujący zawartość pamięci EPROM do pamięci RAM.

Literatura:

1. Dokumentacja systemu DSM (pakiety Z80CPU i MEM8)
2. Instrukcja obsługi monitora szyny systemu DSM
3. Materiały pomocnicze do TMIK/L (mikroprocesor Z8400 SGS-Thompson)
4. K. Badźmirowski i in. „Układy i systemy mikroprocesorowe" WNT, Warszawa 1990, część 1: str. 119-132, część 2: str. 90-115, 138-147
5. K. Fedyna, M. Mizeracki „Układy mikroprocesorowe Z80" WKiŁ, Warszawa 1989, str. 13-54
6. J. Karczmarczuk „Mikroprocesor Z80" WNT, Warszawa 1987, str. 9-88
7. P. Misiurewicz „Podstawy techniki mikroprocesorowej" WNT, Warszawa 1991. str. 45-57, 169-198
8. pr. zb. „Modułowe systemy mikrokomputerowe" WNT, Warszawa 1990, str. 11-28, 63-70, 124-135
9. A. Rydzewski, K. Sacha „Mikrokomputer. Elementy, budowa, działanie" WCiKT NOT-SIGMA, Warszawa 1985, str. 75-125
10. K. Sacha, A. Rydzewski „Mikrokomputer w pytaniach i odpowiedziach" WNT. Warszawa 1985
11. Skorupski i in. „Zagadnienia techniki mikroprocesorowej" WPW, Warszawa 1991, str. 165-169

Dodatek A: Podstawowe funkcje programu monitora szyny
wywołanie: Z80 z katalogu DSM_Z80. Przed wywołaniem należy upewnić się czy jest włączona kaseta i opuścić program NC.

MONITOR		
	BUS STATE	okienko ze stanem magistrali dla sygnałów opisanych w pliku konfiguracyjnym
	OPEN MEMORY	edycja okienka pamięci
	LOAD MEMORY	załadowanie do pamięci kodu do wykonania (zbioru z rozszerzeniem HEX)
RUN		
	RESET SYSTEM	reset układu w kasecie
	FREE RUN	wykonanie programu z nominalną częstotliwością zegara procesora, praca bez monitorowania magistrali
	STEP OVER	praca krokowa (cykle maszynowe)

Posługiwanie się programem monitora szyny zostanie wyjaśnione na początku ćwiczenia przez prowadzącego.

Dodatek B: Podstawowe dane o asemblerze XASMZ80
wywołanie: xasmz80 [drv]name[.ext] [outdrv] flags

drv - nazwa dysku ze zbiorem źródłowym
name - nazwa zbioru źródłowego
ext - rozszerzenie zbioru, domyślnie .ASM

outdrv - nazwa dysku, na którym będą zapisywane zbiory wynikowe
flags - jedna lub kilka z następujących opcji:
L - tworzony tylko plik z listingiem [plik .PRN]
O - tworzony tylko object [plik .HEX]
X - do zbioru prn nie jest wpisywany listing źródłowy
Y - do zbioru prn nie jest wpisywana tablica symboli
C - wysłanie listingu na konsolę
P - wysłanie listingu na drukarkę
N - zaniechanie stronicowania zbioru prn
Q - zaniechanie wyświetlania przez program na ekranie plakietki z nazwą firmy i numerem wersji programu

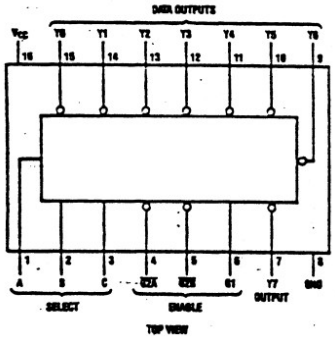
Standardowo assembler tworzy dwa pliki wynikowe:
plik typu object (.hex)
plik z listingiem (.prn)
Zaniechanie kierowania listingu źródłowego i tablicy symboli do pliku prn (opcje X i Y) spowoduje, że w pliku znajdzie się tylko listing linii, w których wystąpiły błędy.

Najczęściej używane dyrektywy assemblera:
symbol EQU expr - definicja stałej
ORG expr - ustawienie licznika pozycji (określa adres, pod którym w kodzie wynikowym zostaną zapisane kody kolejnych instrukcji) na wartość expr
END - wyznacza koniec programu źródłowego

Dodatek C: Wybrane dane katalogowe układów 74HCT138 i 74HCT139

Connection Diagram

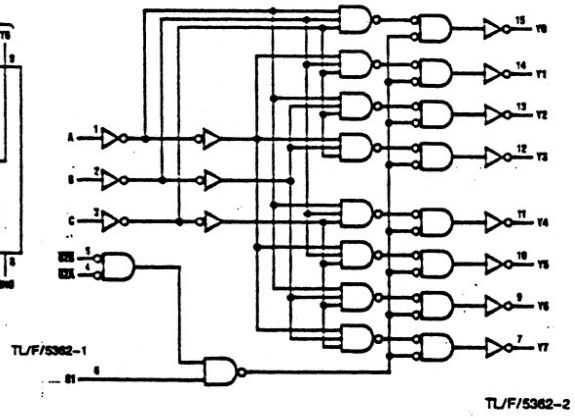
Dual-In-Line Package



Order Number MM54HCT138* or MM74HCT138*

*Please look into Section 8, Appendix D for availability of various package types.

Logic Diagram



TL/F/5362–2

Truth Table

Inputs					Outputs							
Enable		Select										
G1	$\overline{G2}$*	C	B	A	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
X	H	X	X	X	H	H	H	H	H	H	H	H
L	X	X	X	X	H	H	H	H	H	H	H	H
H	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	H	H	H	H	H	L	H	H	H	H
H	L	H	L	L	H	H	H	H	L	H	H	H
H	L	H	L	H	H	H	H	H	H	L	H	H
H	L	H	H	L	H	H	H	H	H	H	L	H
H	L	H	H	H	H	H	H	H	H	H	H	L

*$\overline{G2}$ = $\overline{G2A}$ + $\overline{G2B}$ H = high level L = low level X = don't care