Instytut Elektroniki Politechniki Łódzkiej
©Piotr M. Szczypiński – pms@ck-sg.p.lodz.pl
Wykłady: **Systemy Mikroprocesorowe**

1. Literatura

[1] http://www.eletel.p.lodz.pl/~pms/dyda_pl.html
[2] H. Feichtinger, *Mikrokomputery – poradnik*, WKŁ,
[3] P. Misiurewicz, *Podstawy techniki mikroprocesorowej*, WNT,
[4] J. Bielecki, *TurboAssembler*, PLJ,
[5] S. Kruk, *Turbo Debugger*, PLJ,
[6] Z. Mroziński, *Mikroprocessor 8086 – podręczny katalog elektronika*, WNT
[7] E. Wróbel, *Assembler 8086*
[8] *Elektronika dla wszystkich* nr 8, 9, 10, 11 / 1997 (opis sterownika 8051),
[9] Philips, *Catalog and datasheets*, (http://www-us6.semiconductors.com/handbook/handbook_66.html),
[10] Intel, *MSC 51/151/251 Microcontrollers*, (http://developer.intel.com/design/MCS51/docs_mcs51.htm)
[11] Altera, *Data Sheet*, (http://www.altera.com/html/literature),
[12] T. Łuba, K. Jasiński, B. Zbierzchowski, *Specjalizowane układy cyfrowe w strukturach PLD i FPGA*, Wydawnictwa Komunikacji i Łączności, Warszawa 1997.
[13] [3] Lattice, *Low Density PLDs*, (http://www.latticesemi.com/products/devices/lowplds.html),
[14] [4] Xilinx, *Data Book*, (http://www.xilinx.com/partinfo/databook.htm),

2. Zakres materiału

a) rozwój technik obliczeniowych,
b) bramki logiczne (zasady działania, technologia wykonania, zastosowania),
c) proste układy logiczne, układy synchroniczne,
d) pamięci cyfrowe: (rodzaje, sposób działania, zastosowania),
e) rodzaje danych i metody ich zapisywania w pamięciach cyfrowych,
f) mikroprocesor, definicja, schemat blokowy, sposób działania,
g) układy wejścia / wyjścia i pamięć,
h) układy przerwań i układy DMA,
i) system mikroprocesorowy na przykładzie komputera PC (podstawy):
   - konstrukcja mikroprocesora Intel 8086 (rozkazy procesora, rejestry, adresowanie pamięci),
   - przestrzeń adresowa pamięci i układów wejścia / wyjścia,
   - procedury obsługi sprzętu (BIOS),
   - system operacyjny (na przykładzie MS-DOS),
j) rozwój systemów opartych na mikroprocesorze Intel 80x86 i Pentium
k) inne rodzaje procesorów w przykładach (ogólne informacje):
   - procesory sygnałowe,
   - sterowniki (mikrokomputery jednoukładowe),
l) układy programowalne (ogólne informacje),

m) asembler 8086 na przykładzie programów firmy Borland:
   - co to jest najniższy poziom abstrakcji w programowaniu komputerowym,
   - język asemblera a kod maszynowy,
   - kompilacja (TASM) i konsolidacja (TLINK) programu asemblerowego,

– składnia języka asemblera (słowa kluczowe, stałe, zmienne, mnemoniki, etykiety, procedury, makrodefinicje),
– odpluskwiacz czyli debbuger (TD)

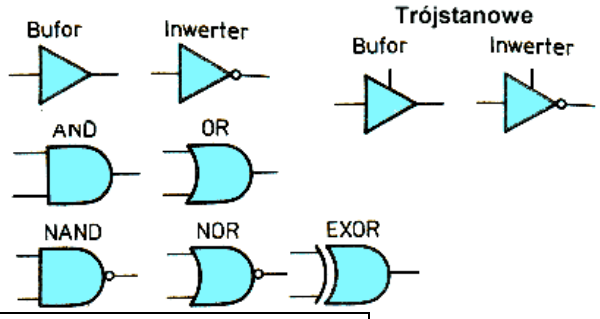
3. Historia

a) Liczydło powstało 5000 lat temu w Babilonie.
b) Mechaniczny arytmometr wynaleziony został w 1623 przez Wilhema Schnickarda i niezależnie w 1652 przez Blaise Pascala.
c) Tabulator, urządzenie umożliwiające zapis danych na kartach perforowanych, wynalazł Herman Hollerith w 1880 roku. Był on przydatny podczas spisu ludności w USA.
d) Komputer lampowy Eniac opracowano w 1946 na potrzeby armii Stanów Zjednoczonych. Komputer zajmował 150m^3, pobierał 50kW energii i wykonywał ok. 10000 operacji na sekundę. Zapoczątkował I generację komputerów.
e) W 1956 w MIT opracowano komputer tranzystorowy (II generacja).
f) Powstały cyfrowe układy scalone zawierające tzw. bramki logiczne. Komputery konstruowane z takich układów scalonych zalicza się do III generacji.
g) Zwiększa się stopień „upakowania" tranzystorów na powierzchni układów scalonych. Powstają układy VLSI. Komputery budowane z takich układów zalicza się do IV generacji.
h) **mikroprocesory** (definicja: *mikroprocesor, mikrokomputer*):
   - w 1970 powstają mikroprocesory (µP) 4 bitowe: TMS1000 firmy Texas Instruments oraz procesor firmy Intel,
   - powstają µP 8 bitowe: Intel 8008 w 1972, Intel 8080 w 1974, Motorola 6800, i inne,
i) **komputer osobisty**: (komputer, na który stać pojedynczego zjadacza chleba, cena<1000$)
   - Mark 8 (do samodzielnego montażu) 1974,
   - Altair (w częściach), 400$, Intel 8080, 1975,
   - Apple, 900$, procesor 6502, 1976,
   - IBM, komputer osobisty IBM PC, 1981,
   - Apple, MacIntosh, 1983,

4. Bramki logiczne i podstawowe układy logiczne

a) Przykłady bramek logicznych.

►Narysować tablicę przejść inwertera i bramki NOR. Zaprojektować układ realizujący funkcje bramki NAND wyłącznie z bramek typu NOR.

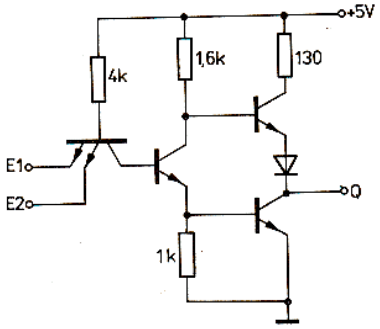


NAND		
Wejście 1	Wejście 2	Wyjście
0	0	1
0	1	1
1	0	1
1	1	0

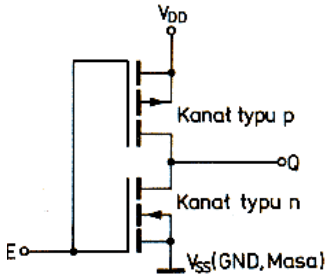
EXOR		
Wejście 1	Wejście 2	Wyjście
0	0	0
0	1	1
1	0	1
1	1	0

b) Przykłady konstrukcji bramek w technologii TTL (Transistor-Transistor-Logic) i CMOS.

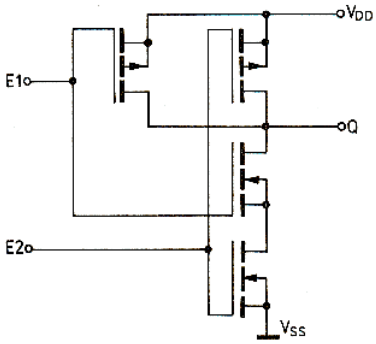
►Narysuj schemat bramki NOR w technologii CMOS.



TTL-NAND



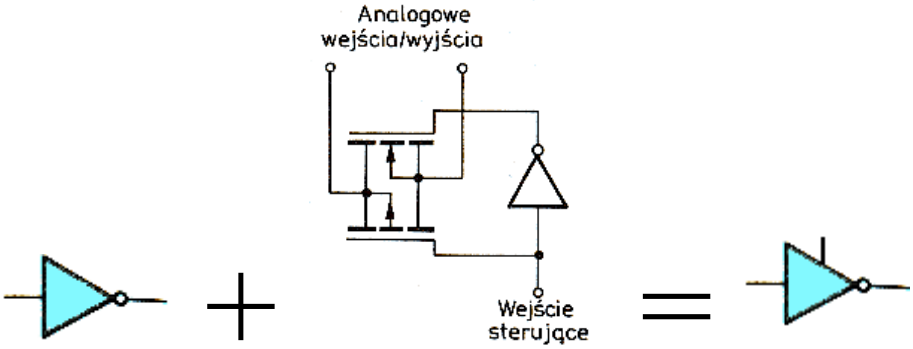
CMOS-Inwerter



CMOS-NAND

c) Zasady łączenia wejść i wyjść bramek logicznych.

d) Bramki trójstanowe umożliwiają połączenie ze sobą wyjść układów logicznych.

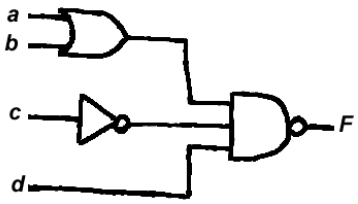


e) Zapis funkcji logicznych: znak + , | lub **OR** oznacza sumę
*, & lub **AND** oznacza iloczyn logiczny, nadkreślenie lub znak ! oznacza negację.

Przykład:

$$F = !\big((a \mid b) \,\&\, !c \,\&\, d\big)$$

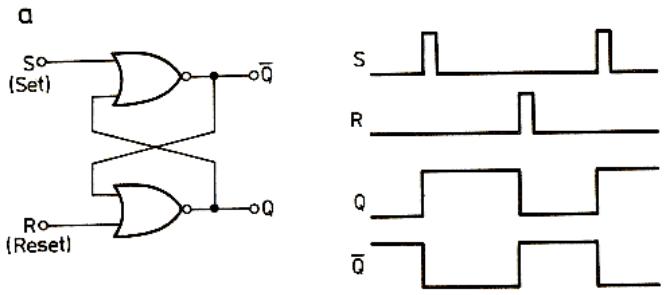
$$F = \overline{(a+b)\cdot \bar{c} \cdot d}$$

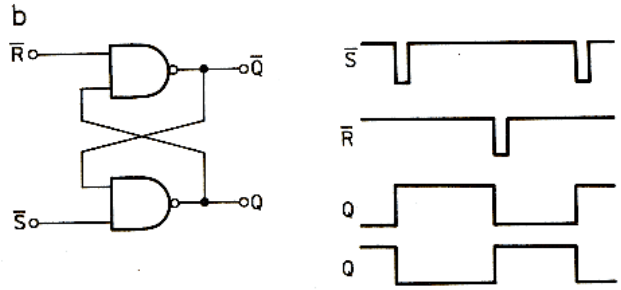


►Przekształcić funkcję z przykładu tak, by w zapisie nie występowały działania sumowania logicznego. Narysuj układ o identycznym działaniu jak ten z przykładu, składający się wyłącznie z bramek typu NAND.

f) Przerzutniki – układy pozwalające zapamiętać wartość 0 lub 1.

Przerzutniki typu RS





►Jaka jest zasada działania przerzutników typu D i JK?
Na czym polega synchroniczność układu logicznego (przerzutników)?
Co to jest wyzwalanie zboczem, czym są przerzutniki typu Master/Slave?

5. Typy danych

a) liczby całkowite dodatnie, liczba bitów,

b) liczby całkowite ze znakiem, kody U1 i U2,
U1 – zmiana znaku poprzez zanegowanie najstarszego bitu,
U2 – zmiana znaku poprzez zanegowanie wszystkich bitów i dodanie 1 (przeniesienie nie jest brane pod uwagę), **►Uzupełnij puste pola tabeli.**

Zapis binarny	Interpretacja dziesiętna liczby bez znaku	Interpretacja dziesiętna w kodzie U2
01111111	127	127
01111110	126	126
...	...	...
00100111		
00100110		
00100101		
...	...	...
00000011	3	3
00000010	2	2
00000001	1	1
00000000	0	0
11111111	255	-1
11111110	254	-2
...	...	...
11101000		
11100111		
...	...	...
10000001	129	-127
10000000	128	-128

c) kod BCD, kolejne czwórki bitów określają cyfrę dziesiętną,
np.: 0010 0110 1001 0011 oznacza dziesiętnie 2693,
►Zapisz w kodzie BCD liczby 6352, 17 i 100.

d) liczby stałoprzecinkowe, fragment ciągu bitów traktowany jest jako część całkowita liczby a fragment jako część po przecinku,
np.: w ciągu bitów 0010 1011 pierwsze cztery bity traktujemy jako część całkowitą, następne cztery jako część po przecinku co daje dziesiętnie liczbę $3\ ^{11}/_{16}$ czyli 3,6875.

e) liczby zmiennoprzecinkowe,

S	**E**	**M**
znak	cecha	mantysa

dla liczb 32 bitowych S zajmuje jeden bit, E zajmuje 8 a M zajmuje 23 bity.

$$\text{Wartość} = (-1)^{\mathbf{S}}\ 2^{\mathbf{E}-127}\ (1,\mathbf{M})$$

ponadto **S**=0, **E**=0 i **M**=0 oznacza wartość 0 a **E**=255 i M=0 oznacza nieskończoność.
►Co to za liczba zmiennoprzecinkowa?
`0 10000000 10010010000111111011010`

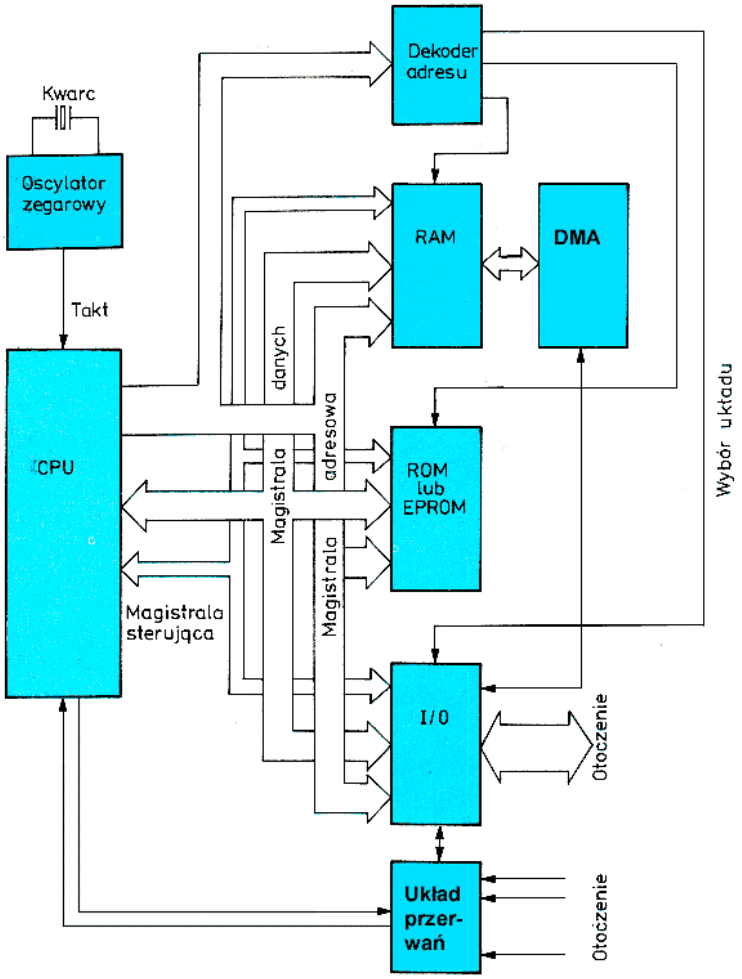
f) znaki alfanumeryczne (i sterujące), ISO-7-bit-code, ASCII,

(Uwaga: dane zapisane w pamięci mają zawsze postać zer i jedynek, o tym w jaki sposób mają być rozumiane decyduje program komputerowy.)

g) zapis binarny, dziesiętny, szesnastkowy, ASCII w asemblerze,
np.: binarnie 01001001**b**, dziesiętnie 288171, szesnastkowa 1a8f**h**, (uwaga zamiast zapisu fa12**h** należy zastosować zapis 0fa12**h**, pierwsza podana jest zawsze cyfra nie litera), znak ASCII **'**d**'**.
►Zapisz liczby dziesiętne 1, 4, 7, 16, 57, 255 w postaci binarnej i szesnastkowej. Jakie znaki ASCII odpowiadają tym wartościom?

6. Schemat blokowy komputera

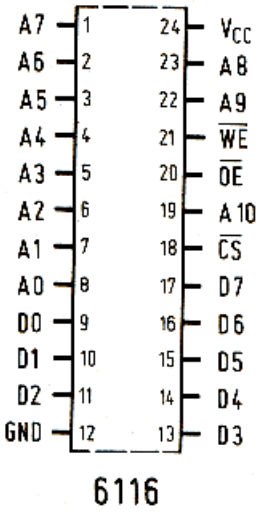
Pamięć, µP, układy wejścia/wyjścia, układy DMA, przerwania.



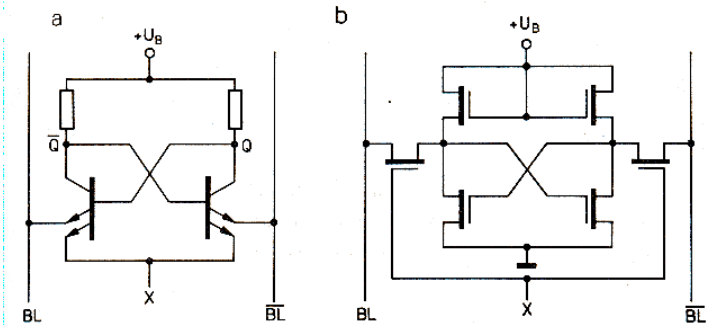
7. Pamięć

a) pamięć RAM (Random Access Memory):

- SRAM (pamięć statyczna RAM),

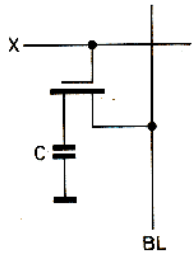


VCC i GND – zasilanie,
A0 do A10 – wejścia adresujące,
D0 do D7 – wejścia / wyjścia danych,
/CS – wejście wyboru układu,
/WE i /OE – wejścia sterujące zapisem i odczytem danych.



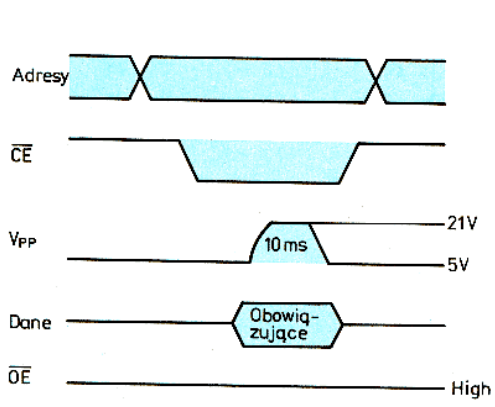
komórka pamięci statycznej zbudowana w technologii
a) TTL oraz
b) MOS

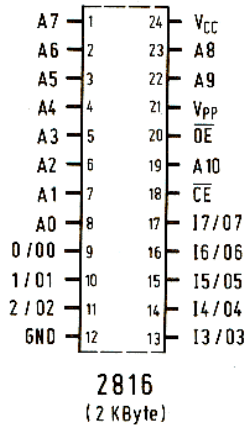
- DRAM – pamięć dynamiczna RAM



uproszczony schemat komórki pamięci dynamicznej,

b) ROM (Read Only Memory) pamięć tylko do odczytu, nieulotna (pamięci PROM, EPROM, EEPROM),

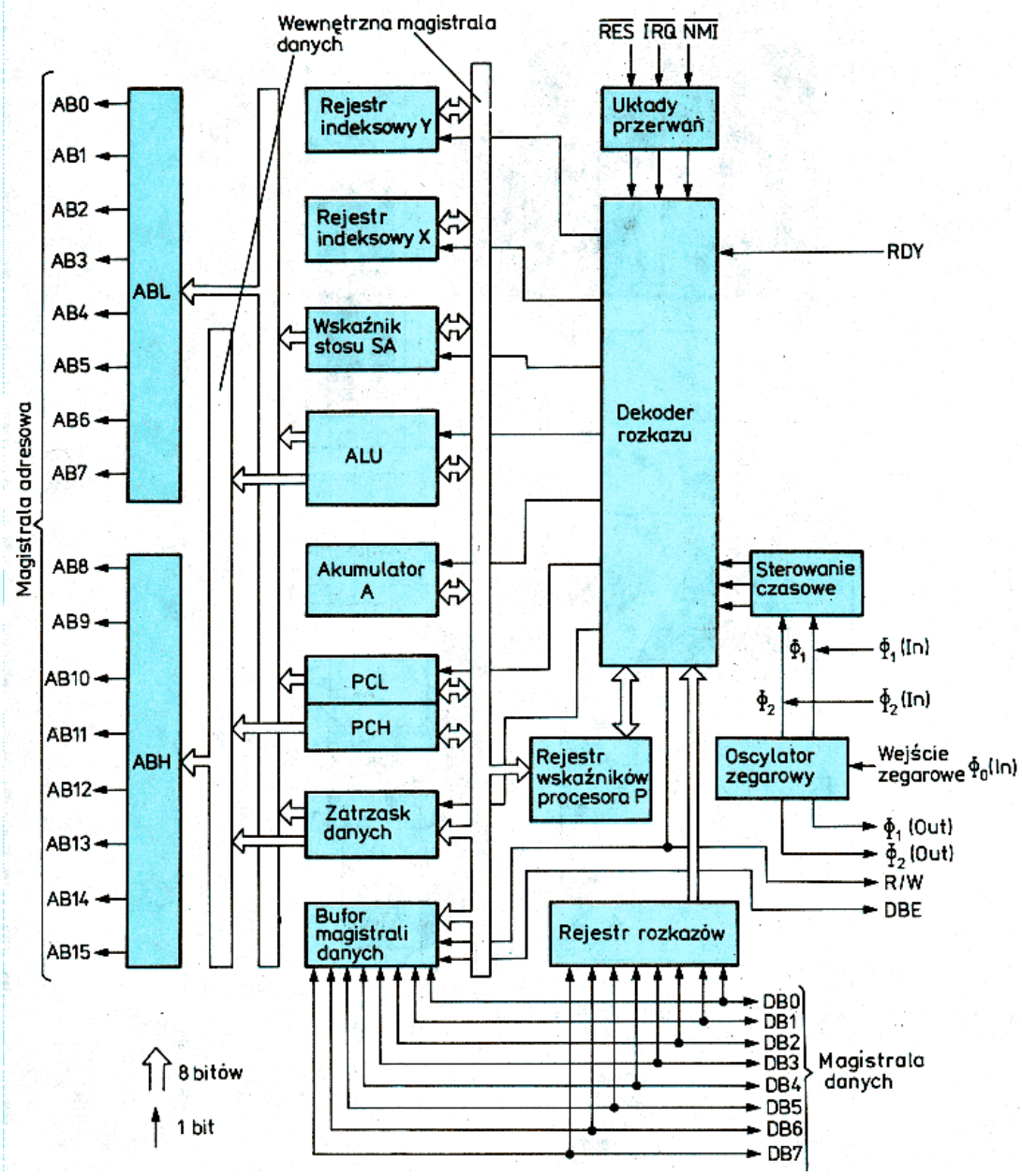




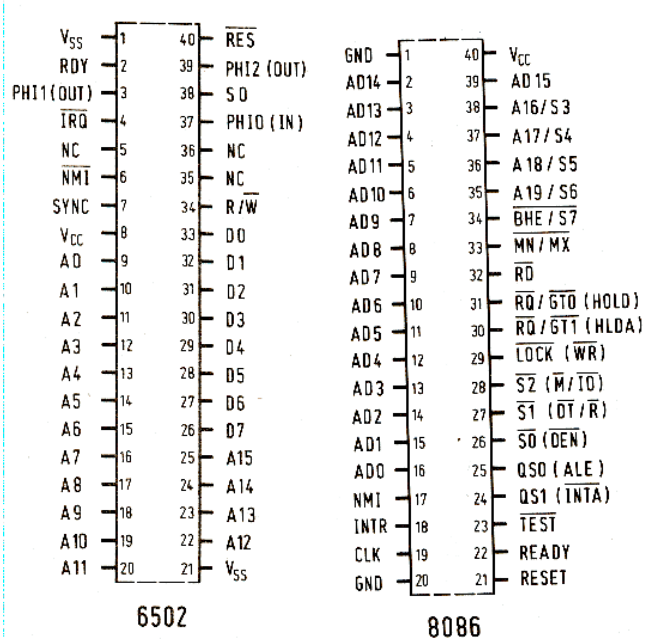
programowanie przykładowego układu typu EEPROM

8. Mikroprocesor

a) działanie µP, schemat blokowy:



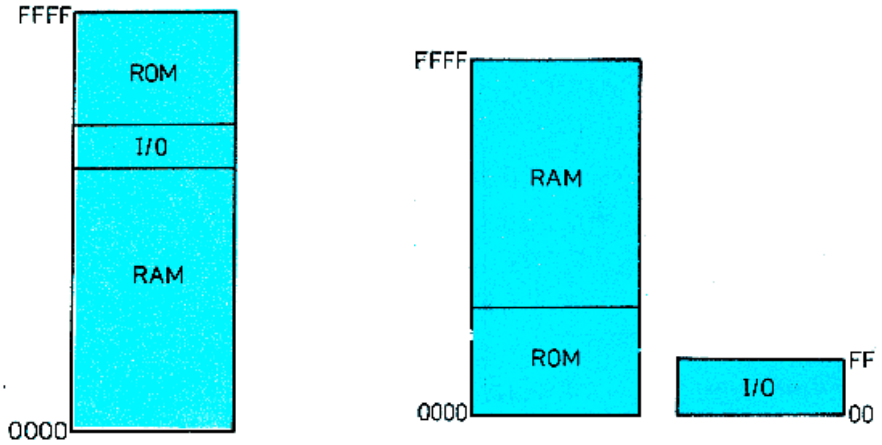
- rejestry,
- jednostka arytmetyczno logiczna,
- dekoder rozkazu,
- układy przerwań,
- instrukcja (rozkaz) procesora,



Wyprowadzenia przykładowych mikroprocesorów firm Motorola i Intel.

b) działanie µP, program,
   - rodzaje instrukcji (rozkazów),
   - sposoby adresowania pamięci
     (zasada adresowania bezpośredniego, pośredniego i natychmiastowego),

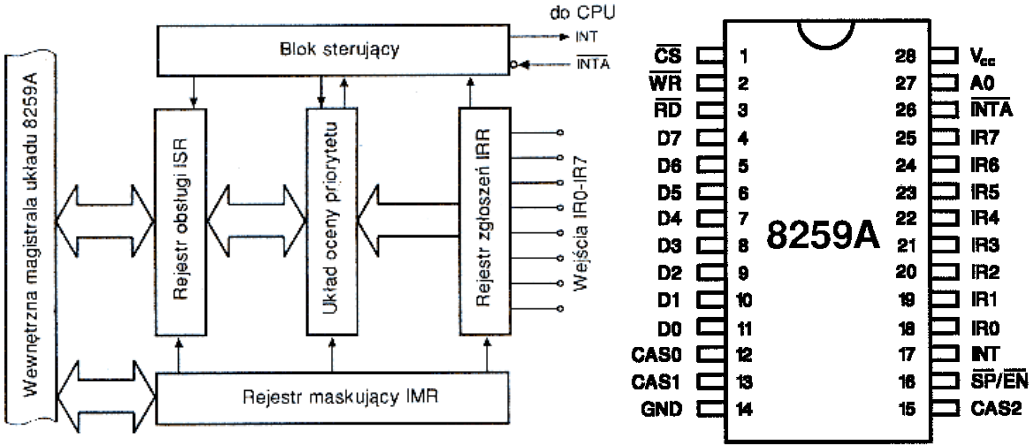
9. Układy wejścia i wyjścia (porty)



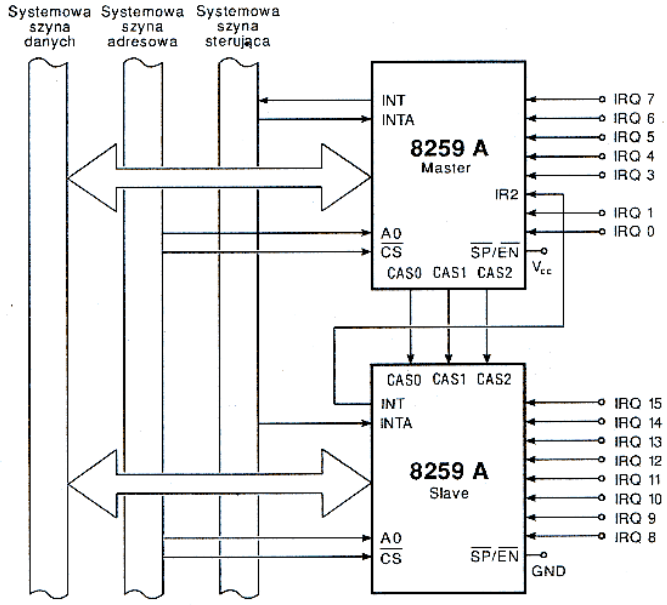
Różne metody projektowania przestrzeni adresowej pamięci i portów.

►Co to są mapy portów i mapy pamięci?

10. Układy przerwań (przykład układu stosowanego w komputerach PC AT)



Schemat blokowy układu kontroli przerwań i przykład układu.



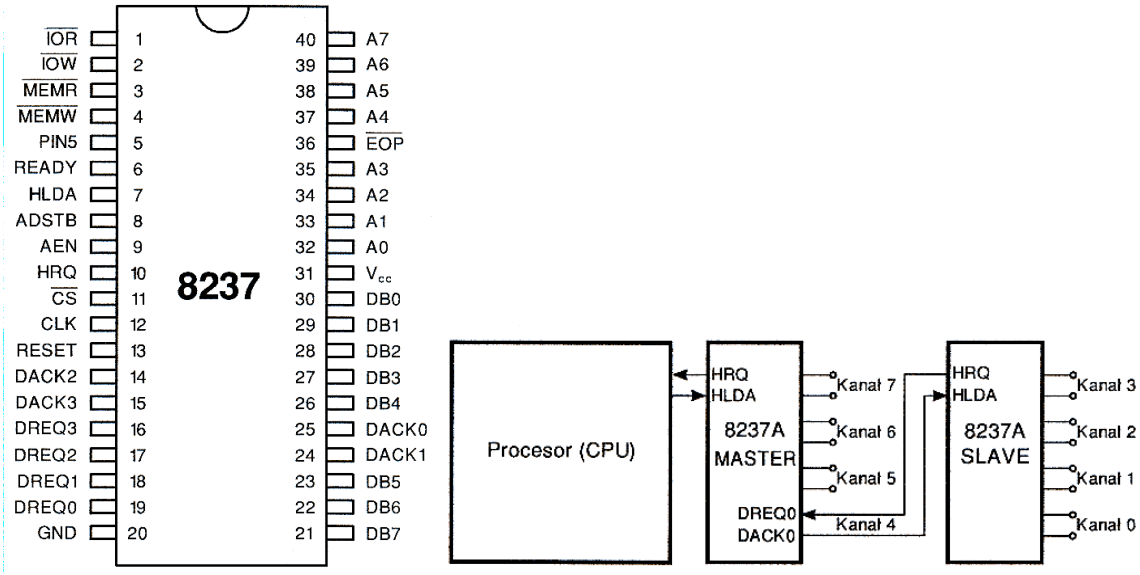
Kaskadowe połączenie dwu układów przerwań

Urządzenia obsługiwane za pomocą przerwań (dla komputera PC AT)					
Układ nadrzędny (Master)			Układ podrzędny (Slave)		
Nr przerwania	Wektor	Urządzenie	Nr przerwania	Wektor	Urządzenie
0	08h	zegar systemowy	8	70h	zegar czasu rzeczyw.
1	09h	klawiatura	9	71h	przerwanie IRQ2
2	0Ah	do układu Slave	10	72h	
3	0Bh	COM 1	11	73h	
4	0Ch	COM 2	12	74h	
5	0Dh	LPT 2	13	75h	koprocesor
6	0Eh	napęd dyskietek	14	76h	dysk twardy
7	0Fh	LPT 1	15	77h	

Wektor oznacza numer indeksu wskazującego adres procedury obsługi danego przerwania, umieszczony w tzw. tablicy wektorów przerwań. Tablica ta znajduje się w przestrzeni adresowej w obszarze 0000h-03FFh i zawiera czterobajtowe pozycje (segm:offs) reprezentujące adresy początków procedur.

Programowanie działania układu 8259A w komputerze PC AT realizowane jest poprzez porty 20h, 21h (Master) oraz A0h i A1h (Slave).

11. Układy DMA



Przykład układu sterowania układami DMA i połączenie kaskadowe dwu takich układów.

Programowanie działania układu 8237 w komputerze PC AT realizowane jest poprzez porty 00h-0Fh, 81h-87h (Slave) oraz 89h-8Dh, C0h-C7h, D0h-DEh (Master).

12. Obsługa urządzeń zewnętrznych na poziomie sprzętowym (przykład)

Klawiatura (urządzenie wejściowe),

Naciśnięcie (lub zwolnienie) klawisza klawiatury powoduje przesłanie tzw. kodu naciśnięcia. Po otrzymaniu kodu układy wewnętrzne komputera obsługujące klawiaturę udostępniają kod znaku poprzez port o adresie 60h i wywołują przerwanie procesora IRQ1.

```
mov dx, 60h        ;załadowanie do rej. dx adresu portu klawiatury
in al, dx          ;pobranie kodu naciśnięcia i umieszczenie do w rej. al
```

Karta graficzna VGA (urządzenie wyjściowe),

Sterowanie kartą graficzną VGA obywa się poprzez zapisywanie odpowiednich wartości do portów o adresach 3B4h do 3DAh. Poniższy kod asemblera powoduje zmianę koloru wyświetlania brzegu ekranu.

```
mov al, 11h        ;załadowanie indeksu „koloru krawędzi” do rej. al
mov dx, 3C0h       ;załadowanie do rej. dx adresu portu indeksującego
out dx, al         ;wysłanie do portu 3C0h wartości 11h
mov al, kolor      ;załadowanie do al kodu koloru (0 ≤ kolor ≤ FFh)
mov dx, 3C1h       ;załadowanie do rej. dx adresu portu
out dx, al         ;zapisanie kodu koloru w układach wewn. karty graficznej
```

13. System operacyjny komputera

a) Struktura warstwowa systemów operacyjnych.
b) System plików – gromadzenie danych.
c) Obsługa urządzeń zewnętrznych.
d) Zarządzanie procesami i wieloprogramowość.

e) System MS-DOS:
- funkcje BIOS,
- funkcje systemu DOS,
- programy usługowe.

Klawiatura (urządzenie wejściowe),
Odczytanie kodu wciśniętego klawisza. Poniższy kod asemblera powoduje załadowanie do rejestru ah kodu klawisza a do rejestru al odpowiadający mu kod ASCII znaku

```
mov ah, 0       ;załadowanie do rej. ah kodu funkcji
int 16h         ; przerwanie programowe - wywołanie procedury bios klawiatury
```

Grafika,
Sterowanie kartą graficzną za pomocą funkcji (przerwania BIOS).
Poniższy kod asemblera powoduje zmianę trybu wyświetlania na tekstowy 40x25 znaków

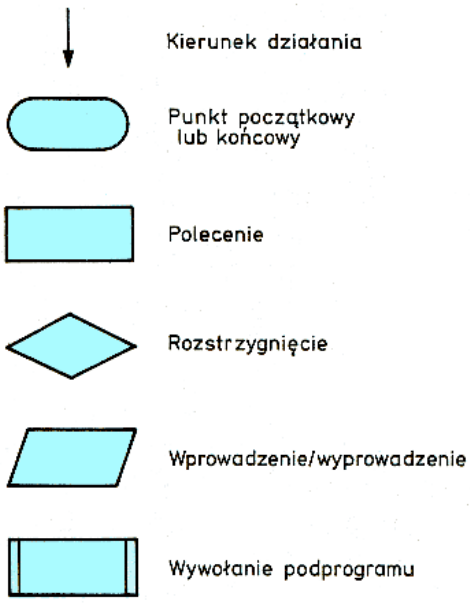
```
mov al, 1       ;nr trybu pracy karty graficznej
mov ah, 0       ;wymagane 0 w rejestrze ah
int 10h         ;przerwanie programowe - wywołanie procedury bios karty graficznej
```

Ten fragment powoduje wyświetlenie na ekranie tekstu znajdującego się w pamięci pod adresem określonym przez zawartość rejestrów ds:dx.

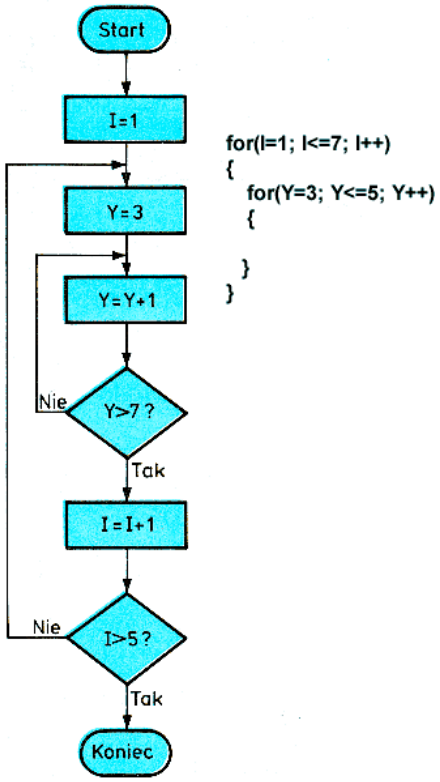
```
mov ah, 9       ;wymagane 0 w rejestrze ah
int 21h         ;przerwanie programowe - wywołanie procedury dos wypisującej tekst
```

14. Układanie algorytmu (przykład)

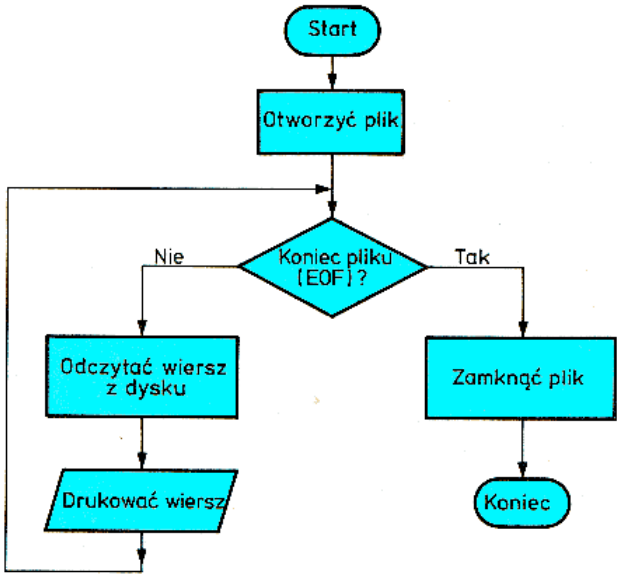
h) bloki działań,
i) skoki warunkowe,
j) pętle,
k) hierarchiczność,



Bloki działań



Przykład algorytmu wykorzystującego pętle



Przykład algorytmu programu wyświetlającego linia po linii zawartość pliku.

►Przedstawić graficznie algorytm mnożenia dwu liczb binarnych za pomocą operacji dodawania i przesunięcia zawartości rejestru.

►Przedstawić graficznie algorytm zamiany liczby binarnej 16-to bitowej w kodzie U2 na ciąg tekstowy (znaki ASCII).

15. Asembler

a) język asemblera,

```
;-------------------------------------------------------------
;           Program przykładowy w asemblerze
;           Piotr M. Szczypinski
;           2000-10-07
;-------------------------------------------------------------

; Definicje stałych
CR                EQU   0DH   ;Znak POWROT KARETKI
LF                EQU   0AH   ;Znak NOWA LINIA
KONIEC_PRACY      EQU  4CH    ;Kod powrotu do DOS'u, wyjścia z programu
WYPISZ_TEKST      EQU  9      ;Kod funkcji DOS'u wypisującej tekst

; Tu zaczyna sie segment danych
DANE  SEGMENT
      TEKST DB    'Program przykladowy na zajecia z ', CR, LF
            DB    'Systemow Mikroprocesorowych.', CR, LF, '$'
DANE  ENDS

; Tu zaczyna sie segment stosu
STOSS       SEGMENT     STACK
      DB    100h DUP(?)
STOSS       ENDS

; Tu zaczyna się segment kodu programu
KOD   SEGMENT PUBLIC 'CODE'
      ASSUME      CS:KOD,DS:DANE

START:
      MOV   AX,STOSS          ;AX=ARGUMENT ZRODLOWY
      MOV   SS,AX             ;ARGUMENT DOCELOWY=AX
      MOV   SP,100h           ;WSKAZNIK STOSU

      MOV   AX,DANE           ;AX=ARGUMENT ZRODLOWY
      MOV   DS,AX             ;ARGUMENT DOCELOWY=AX
      LEA   DX, TEKST         ;DS:DX - ADRES TEKSTU DO WYPISANIA
      MOV   AH,WYPISZ_TEKST   ;KOD FUNKCJI SYSTEMOWEJ
      INT   21H               ;WYWOLANIE DOS-U

      MOV   AH,KONIEC_PRACY   ;KOD FUNKCJI SYSTEMOWEJ
      INT   21H               ;WYWOLANIE DOS-U

KOD   ENDS
      END START
```

b) proces kompilacji,

Programy (produkty firmy Borland):

TASM – asembler, zamienia kod w języku asemblera na kod maszynowy,

TLINK – konsolidator, łączy pliki w kodzie maszynowym i tworzy plik wykonywalny programu,

TD – debuger, pozwala obserwować wykonywanie programu i wyszukiwać błędy.

16. Pisanie programów w asemblerze dla komputerów PC z procesorem Intel 8086 i systemem MS-DOS

a) Procresor Intel 8086:
   - ✓ rejestry procesora,
   - ✓ zbiór rozkazów,
   - ✓ adresowanie pamięci danych i urządzeń we/wy.,
   - ✓ tryby adresowania argumentów rozkazów.

b) Konfiguracja sprzętowa:
   - ✓ organizacja przestrzeni adresowej pamięci,
   - ✓ organizacja przestrzeni adresowej układów wejścia / wyjścia,
   - ✓ przypisanie przerwań sprzętowych urządzeniom,
   - ✓ organizacja bezpośredniego dostępu do pamięci.

c) Funkcje systemowe BIOS i MSDOS:
   - ✓ zbiór funkcji systemowych,
   - ✓ system plików,
   - ✓ sposób odwoływania się do funkcji systemu.

d) Program asemblera.

Rejestry procesora

Symbol	Liczba bitów	Opis
AX (AH, AL)	16 (8, 8)	akumulator - operacje arytmetyczne i logiczne
BX (BH, BL)	16 (8, 8)	operacje arytmetyczne i logiczne, rejestr bazowy
CX (CH, CL)	16 (8, 8)	operacje arytmetyczne i logiczne, rejestr licznikowy
DX (DH, DL)	16 (8, 8)	operacje arytmetyczne mnożenia i dzielenia, adresowania portów,
SI	16	rejestry indeksujące
DI	16	
BP	16	rejestr bazowy
CS	16	rejestr segmentowy kodu programu
DS	16	rejestr segmentowy danych
ES	16	rejestr segmentowy danych (dodatkowy)
SP	16	rejestr segmentowy stosu
IP	16	rejestr licznikowy kodu programu
FL	16 (9)	rejestr określa stan wykonywania programu
SP	16	adresuje dane odkładane na stosie, wskazuje wierzchołek stosu

Bity rejestru flagowego **FL**		
nr	nazwa	funkcja
0	**C**	Carry (przeniesienie)
2	**P**	Parity (parzystość)
4	**A**	Auxiliary (przeniesienie tetrady)
6	**Z**	Zero
7	**S**	Sign (znak)
8	**T**	Trap (pułapka)
9	**I**	Interrupt (przerwanie)
10	**O**	Overflow (nadmiar)
11	**D**	Direction (kierunek)

Zbiór rozkazów

Grupa rozkazów	Mnemoniki rozkazów	Przykład	
operacje arytmetyczne	ADC, ADD, CBW, CMP, CWD, DEC, DIV, IDIV, IMUL, INC, MUL, NEG, SBB, SUB,	INC A	zwiększ zawartość rejestru A o jeden
operacje arytmetyczne w kodzie BCD	AAA, AAD, AAM, AAS, DAA, DAS,		
operacje logiczne	AND, NOT, OR, TEST, XOR,	AND AH, 15	iloczyn logiczny bitów rejestru A i liczby 15, wynik umieszcza w rej. A
przesunięcia bitów	RLC, RCR, ROL, ROR, SAL, SAR, SHL, SHR,	SHL AX, 3	przesunięcie bitów rej. AX o 3 pozycje w lewo
przesłania danych	IN, INSB, INSW, LAHF, LDS, LES, LEA, MOV, OUT, POP, POPA, POPF, PUSH, PUSHA, PUSHF, XCHG, XLAT,	MOV AX, 15 PUSH A	umieszcza w rej. A liczbę 15 zapisuje zawartość A na stosie
upraszczające operacje na tablicach i buforach danych	CMPSB, CMPSW, LODSB, LODSW, MOVSB, MOVSW, OUTSB, OUTSW, REP, REP__ (REPNE, REPZ), SCASB, SCASW, STOSB, STOSW,		
operacje na bitach rejestru flagowego	CLC, CLD, CLI, CMC, STC, STD, STI,	STC	ustawienie znacznika C rej. flagowego na 1
rozkazy sterujące wykonywaniem programu (skoki)	BOUND, CALL, ENTER, ESC, HLT, INT, INTO, IRET, J__ (JNZ, JZ, itp), JMP, LEAVE, LOCK, LOOP, RET, RETF, RETN, WAIT,	CALL proc RET JZ rel	skok do procedury „proc” powrót z procedury skok do miejsca oznaczonego przez „rel” gdy wynik zerowy
rozkaz nic	NOP	NOP	nic nie robi

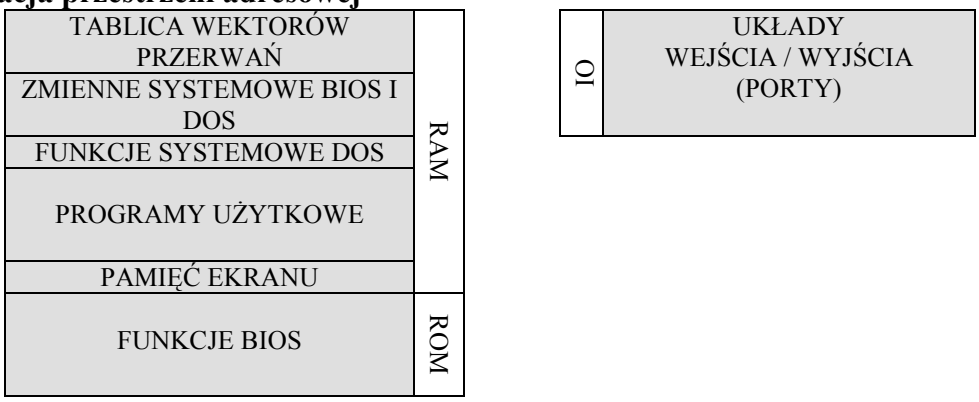
Adresowanie pamięci i przestrzeni urządzeń we/wy (portów)

Zapis SEGMENT:OFFSET (ADRES = 16 SEGMENT+OFFSET),
przestrzeń adresowa portów.

Tryby adresowania argumentów rozkazów

Adresowanie	Opis	Przykład
rejestrowe	argument zawarty w rejestrze procesora	INC AX
natychmiastowe	argument zawarty jest w rozkazie	MOV AL, 15
bezpośrednie pamięci	argumentem jest zmienna w pamięci, której adres podany jest w rozkazie	MOV AL, [2453] MOV AL, nazwa
bezpośrednie rozkazów	argumentem jest adres, do którego ma nastąpić skok	CALL [2453] CALL procedura
pośrednie pamięci i portów	argumentem jest zmienna w pamięci o adresie podanym w rejestrze procesora	MOV AL, [bx] IN AL
indeksowe pośrednie pamięci	argumentem jest zmienna w pamięci o adresie podanym jako suma rejestru bazowego i indeksowego	MOV AL, [bx+si] MOV AL, [24+bx+si]

Organizacja przestrzeni adresowej



Przerwania sprzętowe i programowe (rozkaz INT)

Przykłady:

0h – dzielenie przez zero,
1h – przerwanie pracy krokowej,
2h – przerwanie niemaskowalne NMI,
...
8h
...
15h } przerwania sprzętowe,
...
10h – przerwanie programowe funkcji karty graficznej,
...
13h – funkcje obsługa stacji dyskietek,
16h – funkcje obsługi obsługa klawiatury,
...
21h – funkcje udostępniane przez system MSDOS,
...
70h
...
77h } przerwania sprzętowe,

Jaka jest różnica między przerwaniem obsługi klawiatury sprzętowym (wektor 9) i programowym (wektor 16h)? Co to jest program rezydentny?

Zbiór funkcji systemowych

Obsługa funkcji systemowych w systemie MSDOS wymaga ustawienia parametrów wywoływanej funkcji w odpowiednich rejestrach oraz wywołania przerwania programowego rozkazem **INT**

Przykłady:

```
DX := adres tekstu do wypisania,
AH := 9 (kod funkcji wypisującej
      tekst)
Wywołać przerwanie nr 21H

AL := kod wyjścia z programu,
AH := 4Ch (kod funkcji zakończenia
      programu)
Wywołać przerwanie nr 21H
```

```
AH := 0
Wywołać przerwanie nr 16H
W rejestrach AL i AH pojawią się
odpowiednio: kod ASCII znaku i kod
klawisza

AH := 0
AL := nr trybu graficznego
Wywołać przerwanie nr 10H
```