Architektura komputerów

Wykład 2
Od 8086 do . . . ?

Wojciech Kordecki

Collegium Witelona
Wydział Nauk Technicznych i Ekonomicznych
Zakład Informatyki

Semestr letni 2023/24

Protoplaści i inni krewni – Intel 8080

Jeden z pierwszych mikroprocesorów, wyprodukowany przez Intela w kwietniu 1974. Jest 8-bitowym mikroprocesorem wykonanym w technologii *n*-MOS, pracującym z częstotliwością 2 MHz.
Jest uniwersalną jednostką centralną złożoną z jednostki arytmetyczno-logicznej, rejestrów roboczych i układu sterowania.
Dane i instrukcje są przesyłane do i z pamięci za pośrednictwem 8-bitowej szyny danych, pamięć jest adresowana 16-bitową szyną adresową.
Blok rejestrów z układem wybierającym:

- licznik rozkazów (PC) – 16-bitowy,
- wskaźnik stosu (SP) – 16-bitowy,
- sześć ośmiobitowych rejestrów uniwersalnych: B, C, D, E, H, L, które można łączyć w pary rejestr pomocniczy składający się z dwóch ośmiobitowych rejestrów W oraz Z.

Źródło: Wikipedia

Protoplaści i inni krewni – Z80

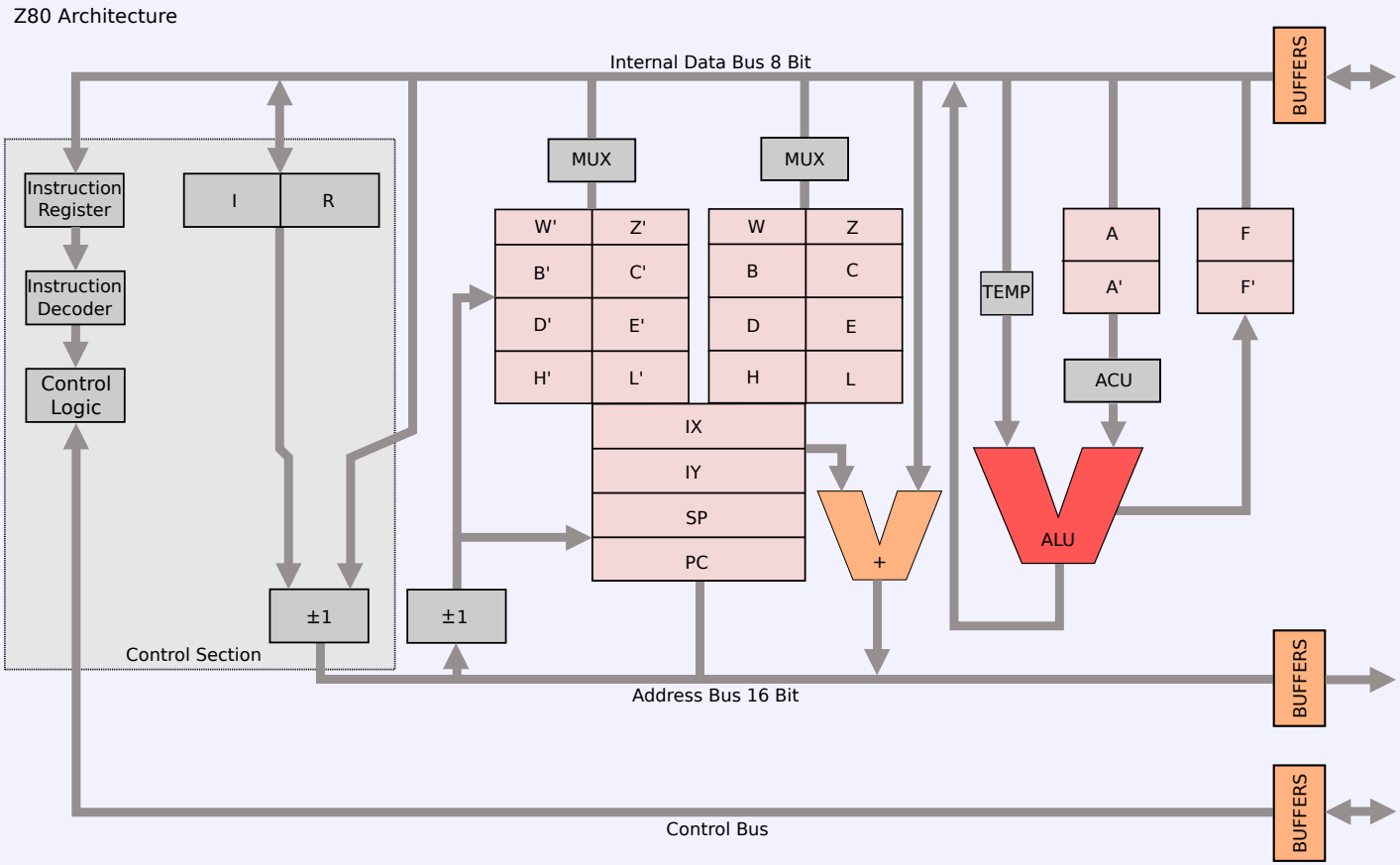
Mikroprocesor opracowany w firmie ZiLOG (przez uciekinierów z Intela).

- 158 rozkazów, w tym 78 zgodnych z Intel 8080 (zachowana pełna wsteczna kompatybilność z 8080),
- bardzo duży jak dla procesora 8-bitowego zestaw rejestrów wewnętrznych ogólnego przeznaczenia wraz z zestawem alternatywnych rejestrów (np. dla wygodnej obsługi przerwań) i rejestrami indeksowymi (np. dla wygodnej implementacji tablic) – A, F, A', F', B, C, D, E, H, L, B', C', D', E', H', L', IX, IY, SP, PC, WZ, IR,
- zestaw rozkazów operujących na 16-bitowych danych (rejestry można było „sklejać" parami).

Źródło: Wikipedia

Znany z zastosowania w mikrokomputerze ZX Spectrum.

Z80 – architektura



Źróło: Appaloosa - Praca własna, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=2933572

ZX Spectrum

ZX Spectrum – produkowany przez angielską firmę Sinclair Research od 1982 roku.

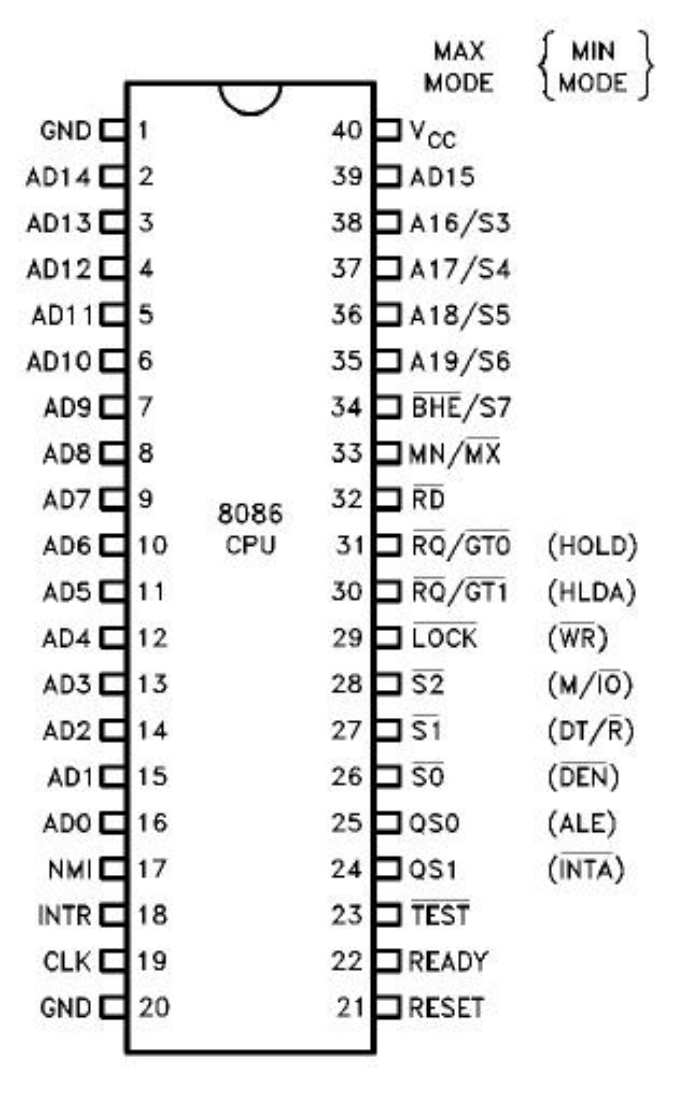
Źródło: Bill Bertram - Praca własna, CC BY-SA 2.5, https://commons.wikimedia.org/w/index.php?curid=170050

Intel 8080 a sprawa polska

MCY7880 polska wersja 8-bitowego mikroprocesora Intel 8080 produkowana w latach 70. i 80. XX w. przez istniejące wtedy Naukowo Produkcyjne Centrum Półprzewodników CEMI w Warszawie.
Był to jedyny mikroprocesor produkowany w latach 80. w Polsce.

Źródło: Wikipedia

Mikroprocesor Intel 8086



Źródło – Wikipedia

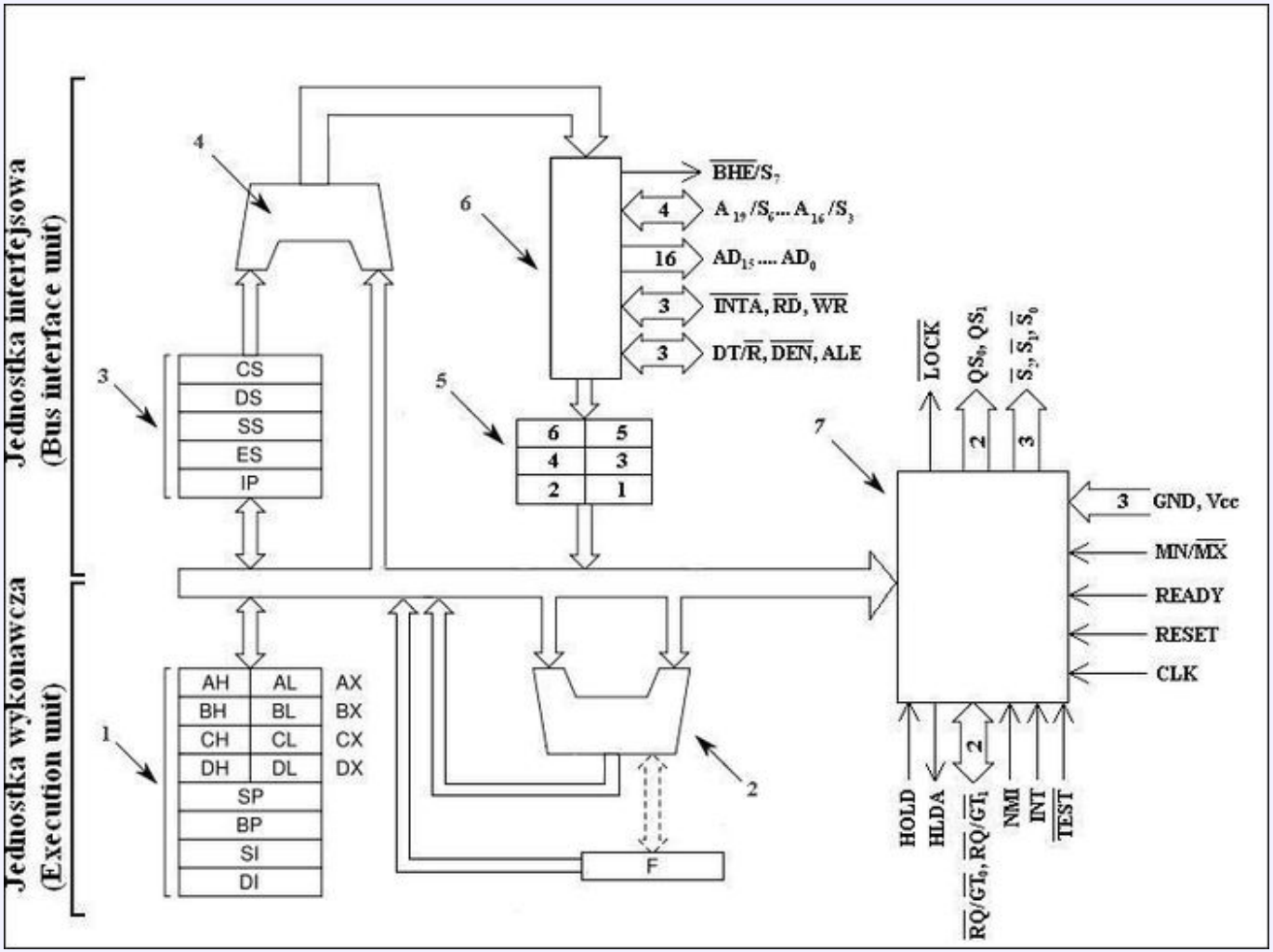
Wyprowadzenia mikroprocesora 8086,

Historia rodziny

Wprowadzony w roku 1987 procesor 8086 z zegarem 5 MHz miał 29 000 tranzystorów. Wprowadzony w roku 2013 sześciordzeniowy Core i7 EE 4960X pracuje z częstotliwością 4 GHz i ma 1.86 miliarda trazystorów ([Stallings(2022)] t. 1, str.26).

Do tej historii będziemy wracać w ciągu semestru.

Schemat blokowy procesora 8086



Źródło – Wikipedia

Rejestry procesora 8086

rejestr	poł. dolna	poł. górna	nazwa
AX	AL	AH	akumulator
BX	BL	BH	
CX	CL	CH	licznik
DX	DL	DH	
SP			wskaźnik stosu
BP			
SI			rejestr indeksowy źródła
DI			rejestr indeksowy przeznaczenia
CS			
DS			
SS			
ES			
IP			licznik rozkazów
			rejestr znaczników

Dla kontrastu – procesor ARM

Procesor RISC – uniwersalne rejestry, całkowicie odmienna struktura rozkazów.

Jeden z najpopularniejszych obecnie procesorów w urządzeniach mobilnych.

Bardziej szczegółowe omówienie później.

Rejestry procesora ARM

System and User	FIQ	Supervisor	Abort	IRQ	Undefined
r0	r0	r0	r0	r0	r0
r1	r1	r1	r1	r1	r1
r2	r2	r2	r2	r2	r2
r3	r3	r3	r3	r3	r3
r4	r4	r4	r4	r4	r4
r5	r5	r5	r5	r5	r5
r6	r6	r6	r6	r6	r6
r7	r7	r7	r7	r7	r7
r8	r8_fiq	r8	r8	r8	r8
r9	r9_fiq	r9	r9	r9	r9
r10	r10_fiq	r10	r10	r10	r10
r11	r11_fiq	r11	r11	r11	r11
r12	r12_fiq	r12	r12	r12	r12
r13	r13_fiq	r13_svc	r13_abt	r13_irq	r13_und
r14	r14_fiq	r14_svc	r14_abt	r14_irq	r14_und
r15 (PC)	r15 (PC)	r15 (PC)	r15 (PC)	r15 (PC)	r15 (PC)

Tryb ARM ▪ rejestry znaczników

CPSR	CPSR	CPSR	CPSR	CPSR	CPSR
	SPSR_fiq	SPSR_svc	SPSR_abt	SPSR_irq	SPSR_und

Asemblery

Dwa znaczenia słowa *asembler:*

- język, w którym programuje się procesor – zbiór instrukcji i reguł składni,
- program tłumaczący instrukcje na język wewnętrzny (binarny).

Asemblery

Dwa znaczenia słowa *asembler:*

- język, w którym programuje się procesor – zbiór instrukcji i reguł składni,
- program tłumaczący instrukcje na język wewnętrzny (binarny).

Wiele podręczników, m.in. [Wróbel(2011)], [Pirogov(2005)], [Farbaniec(2012)].

Asemblery

Dwa znaczenia słowa *asembler:*

- język, w którym programuje się procesor – zbiór instrukcji i reguł składni,
- program tłumaczący instrukcje na język wewnętrzny (binarny).

Wiele podręczników, m.in. [Wróbel(2011)], [Pirogov(2005)], [Farbaniec(2012)].

Różne darmowe (freeware) lub tanie (shareware) asemblery można znaleźć w internecie.

Microsoft Macro Assembler (MASM)
Dobry (i darmowy) asembler dla Windows to MASM32v11

Procesor i jego rejestry (1)

Procesor Intel 8086 posiada 16-bitowe rejestry.
Rejestry ogólnego przeznaczenia:

- AX – do operacji arytmetycznych i logicznych,
- BX – rejestr bazowy, adresowanie pamięci,
- CX – licznik,
- DX – rejestr danych, mnożenie i dzielenie, wysyłanie i odbieranie danych z portów.
- SI – rejestr indeksujący pamięć, wskazuje obszar skąd przesyła się dane,
- DI – rejestr indeksujący pamięć, wskazuje obszar dokąd przesyła się dane,
- SP – wskaźnik stosu,
- BP – rejestr do adresowania pamięci.

Procesor i jego rejestry (2)

Rejestry te można traktować jako dwa rejestry 8-bitowe:

AX = AH+AL
BX = BH+BL
CX = CH+CL
DX = DH+DL

Rejestr IP (wskaźnik rozkazu) zawiera numer w pamięci aktualnie wykonywanego rozkazu. Nie jest dostępny dla programisty.

Adresowanie pamięci

Adresowanie pamięci (wspólna pamięć programu i danych) wg wzoru:

$$\text{adres} = 16 \cdot \text{segment} + \text{offset}$$

Adres zapisywany jest w postaci segment:offset, np. 1000:7C00 i 1700:0C00 oznaczają ten sam adres 17C00. Oznacza to, że adresy są 20-bitowe, czyli adresować można do 1 MB.

Rejestry segmentowe

- CS – segment aktualnie wykonywanego rozkazu,
- DS – segment z danymi,
- ES – segment dodatkowy np. przy przesyłaniu łańcuchów
- SS –segment stosu.

Rejestry te są używane niejawnie, np. rozkaz skoku pod adres 0 jest skokiem pod adres CS:0.

Stos rośnie w kierunku malejących adresów (stos wiszący).

SS wskazuje segment stosu, SP – wierzchołek stosu.

Rejestr znaczników (flag)

– – – – O D I T S Z – A – P – C

- O – nadmiar,
- D – kierunek przesyłu danych,
- I – zezwolenie na przerwania,
- T – praca krokowa,
- S – znak wyniku operacji arytmetycznej,
- Z – znacznik zera j.w.,
- A – przeniesienie połówkowe,
- P – parzystość,
- C – przeniesienie

Dane i adresowanie

Stała, np. 1 np. MOV AX,1
Dane rejestrowe – zawartość rejestru
Dane pamięciowe (adres pośredni) – rejestr w nawiasie kwadratowym, np. [BX] – dana pod adresem wskazywanym przez rejestr (rejestry).
Zmienna – numer komórki pamięci dany przez etykietę, np.

```
jeden db 1
dwa   dw 1000
napis db 'napis'
```

Przesłania

MOV (prawie wszystkie kombinacje dwóch argumentów)
XCHG arg1,arg2 (zamiana między rejestrami lub rejestrem i adresem)
IN rejestr-a,adres (adres<100h)
IN rejestr-a,dx (rejestr-a, to AX lub AL)
OUT adres,rejestr-a
OUT adres,dx

Operacje arytmetyczne

ADD rejestr,adres i.in,
SUB rejestr,adres i.in, (bez pożyczki)
ADC rejestr,adres i.in,
SBB rejestr,adres i.in, (z pożyczką)
INC rejestr
DEC rejestr
MUL rejestr|adres (rejestr-a mnożony przez argument)
IMUL rejestr (j.w. ze znakiem)
DIV rejestr (rejestr-a mnożony przez argument)
IDIV rejestr (j.w. ze znakiem)
NEG rejestr|adres (zmiana znaku)

Przesunięcia i obroty

SHL adres|rejestr,CL|1 (logiczne)
SAL adres|rejestr,CL|1 (arytmetyczne)
SHR adres|rejestr,CL|1 (logiczne)
SAR adres|rejestr,CL|1 (arytmetyczne)

SHR zeruje najbardziej znaczący bit, SAR – powiela go.
Bit wysunięty poza bajt do C.

ROL, RCL, ROR, RCR – obroty w lewo i prawo bez przeniesienia lub z przeniesieniem C, argumenty jak poprzednio.

Operacje logiczne

NOT rejestr|adres (negacja bit po bicie)
AND rejestr|adres,rejestr|adres|wartość
OR rejestr|adres,rejestr|adres|wartość
XOR rejestr|adres,rejestr|adres|wartość
TEST rejestr|adres,rejestr|adres|wartość

TEST nie zmienia wartości pierwszego argumentu, tylko ustawia znaczniki.

Procedury, stos i przerwania (1)

CALL
JMP
wywołania procedur i skoki bezwarunkowe – zostaną omówione dalej.
LOOP etykieta
– zmniejsza CX o 1, skacze do etykiety, gdy CX>0
LOOPZ – jak LOOP, ale skok gdy CX>0 oraz Z=1
LOOPNZ – jak LOOP, ale skok gdy CX>0 oraz Z=0

Procedury, stos i przerwania (2)

INT nr
– wywołanie przerwania programowego o numerze nr
$(0 < \mathrm{nr} < 255)$
RET (powrót z procedury)
IRET (powrót z przerwania)
PUSH arg
POP arg

Argument arg jest adresem danej dwubajtowej albo jednym z rejestrów AX, BX, CX, DX, BP, SP, SI, DI, CS, DS, SS, ES. Nie jest możliwe ani położenie ani zdjęcie ze stosu danej jednobajtowej.

Przerwania

W segmencie 0 znajduje się tabela skoków, mająca 256 pozycji. Każda pozycja to jeden adres segment:offset, czyli 4 bajty. Razem 1024 bajty. Każdy adres to *wektor przerwań*. W czasie startu komputera BIOS i DOS wpisują tam adresy swoich procedur. Instrukcja INT wywołuje odpowiednie przerwanie. Jej argumentem jest pozycja w tabeli przerwań – numer przerwania. Wywołanie procedury może być też wywołane automatycznie tzn. sprzętowo. Procedury obsługi przerwania mogą mieć różne funkcje. Numer funkcji wraz z ewentualnymi argumentami musi być podany w rejestrze AX: w AH numer funkcji, w AL numer podfunkcji.

Program w konwencji MASM, wykorzystuje funkcję numer 9 – wyprowadzenie na standardowe wyjście łańcucha znaków. Łańcuch kończy sie znakiem '$'.

Przerwanie INT 21 – przykład (1)

```
mstos SEGMENT STACK
      DB 100h DUP (?)     ;pamięć na stos
mstos ENDS
dane SEGMENT
kom1 db 'Witamy na programowaniu niskopoziomowym',13,10,'$
dane ENDS
prog SEGMENT          ;program
      assume CS:prog,DS:dane
pocz PROC
```

Przerwanie INT 21 – przykład (2)

```
start:
mov ax,dane
mov ds,ax
lea DX,kom1
mov ah,9     ;argument przerwania - numer funkcji
             ;wyświetlenie ciągu znaków
int 21h      ;przerwanie
mov ah,4Ch   ;argument przerwania - koniec programu
int 21h      ;przerwanie
pocz ENDP
prog ENDS
END start
```

Przerwanie INT 21 – przykład (3)

W konwencji TASM. Czytanie znaku ze standardowego wejścia z echem na standardowym wyjściu. W tym przypadku wejście=klawiatura, wyjście=ekran.

```
.MODEL tiny ;model
.STACK 100h ;pamięć na stos
.CODE       ;program
start:
mov ah,1    ;argument przerwania - numer funkcji
            ;czytanie znaku z echem
mov cx,10   ;licznik, 10 znaków
l: int 21h  ;przerwanie, czeka na znak z echem
loop l      ;pętla 10 razy, aż CX=0
mov ah,4Ch  ;argument przerwania - koniec programu
int 21h     ;przerwanie
END start
```

Źródło i cel

Przystępny opis w rozdziale 10 książki [Duntemann(1993)].
Łańcuch – spójny obszar pamięci. Można operować na dwóch łańcuchach jednakowej długości równocześnie.
Założenia:

- łańcuch źródłowy wskazywany jest przez DS:SI,
- łańcuch docelowy wskazywany jest przez ES:DI,
- wspólna długość łańcuchów wskazywana jest przez CX,
- dane pobierane z łańcucha źródłowego lub przekazywane z do łańcucha docelowego muszą przechodzić przez rejestr AX.

Wypełnienie obszaru pamięci. Adres segmentu w ES, przesunięcie (offset) w DI, słowo umieszczone w AX, liczba słów w CX.

Łańcuchy – przykład

Zakładamy, że ES, DI, AX, CX mają już prawidłowe wartości.

```
kopiuj: mov es:[di],ax   ;kopiuj AX pod adres ES:DI
inc di                   ;adres zwiększamy o 2
inc di
dec cx                   ;licznik zmniejszamy o 1
jnz kopiuj               ;sprawdzamy, czy koniec
```

Inaczej

```
rep stosw
```

Instrukcja STOSW

- słowo z AX kopiowane jest pod adres ES:DI,
- DI jest zwiększane (gdy DF=0) lub zmniejszane (gdy DF=1) o 2.

Przedrostek REP powoduje, że zawartość CX zmiejszana jest o 1, a STOSW jest powtarzane, aż do osiągnięcia CX=0.

Porównywanie łańcuchów

Porównywanie dwóch łańcuchów może być wykonane instrukcją CMPS, CMPSB, CMPSW.

Przykład Porównanie pierwszych 50 znaków tablic umieszczonych pod adresami tab1 i tab2.

Porównywanie łańcuchów – przykład

```
mov si,OFFSET tab1   ;kompletujemy adres tab1
mov ax,SEG tab1
mov ds,ax
mov di,OFFSET tab2   ;kompletujemy adres tab2
mov ax,SEG tab2
mov es,ax
mov cx,50            ;pierwszych 50 znaków
cld                  ;wyzerowanie znacznika kierunku
repe cmpsw           ;porównujemy póki są równe
jne tab_ne
.............
tab_ne:              ;wracamy z adresami na pierwszy
dec si               ;element nierówny
dec si
dec di
dec di
```

Procesory 386 i 486

Procesory 386 i 486 są zgodne ze swoimi poprzednikami. Rejestry AX, BX, CX, DX, SI, DI, BP, SP zostały rozszerzone do 32-bitowych, jako EAX, EBX, ECX, EDX, ESI, EDI, EBP, ESP. Ich mlodsze połówki (słowa) są dostępne pod starymi nazwami, starsze połówki nie są dostępne oddzielnie. Rejestry segmentowe CS, DS, ES, SS pozostały jako 16-bitowe. Doszły dwa nowe 32-bitowe rejestry segmentowe FS i GS.
Lista rozkazów została istotnie rozszerzona, np. PUSHAD kładzie, a POPAD zdejmuje ze stosu wszystkie rejestry ogólnego przeznaczenia. PUSHFD kładzie, a POPFD zdejmuje ze stosu rejestry znaczników. Wiele instrukcji operujących na bajtach i słowach, ma wersje działające na podwójnych słowach, np. do STOSB i STOSW doszła STOSD.

Rejestry Pentium i adresowanie

Rejestry Pentium i tryby adresowania, to dalsze rozwinięcie procesorów 386 i 486. W porównaniu do 8086 są 32-bitowe. Rejestry podstawowe są takie same jak w 386. Wskaźnik rozjazów EIP i rejestr znaczników (flag) EFAGS.

Nowe rejestry, to koprocesor numeryczny: R0 – R7 (taki jak w 486) i MMX (o tym później).
W Pentium III następne 8 rejestrów 128-bitowych SSE.
W Pentium 4 nie ma nowych rejestrów, są nowe rozkazy.

Rejestry – schemat

31 16	15 8	7 0	16-bit	32-bit
	AH	AL	AX	EAX
	BH	BL	BX	EBX
	BH	BL	CX	ECX
	DH	DL	DX	EDX
	BP			EBP
	SI			ESI
	DI			EDI
	SP			ESP

Rejestry segmentowe 16-bitowe: CS, DS, SS, ES, FS, GS.

Adresowanie pamięci segment:offset

Pentium PRO do Pentium 4: magistrala adresowa 36 bitów, pamięć operacyjna do 64 GB, offset 32 bity, segment 4 GB.
Pentium 4 EE (Prescot): magistrala adresowa 40 bitów, pamięć operacyjna do 1 TB, offset 32 bity, segment 4 GB.
Płaski model pamięci (Windows) – operujemy jedynie offsetem w tym samym segmencie.

Adresowanie argumentów

64 K ośmiobitowych układów wejścia-wyjścia (portów). Można je adresować bezpośrednio lub za pomocą DX. Instrukcje IN i OUT

Przykład.

```
in al,3f8h
mov dx,0df0h
in ax,dx
out 61j,bl
mov dx,378h
out dx,al
```

Obliczanie adresu efektywnego w trybie 32-bitowym

$$\begin{bmatrix} EAX \\ EBX \\ ECX \\ EDX \\ ESP \\ EBP \\ ESI \\ EDI \end{bmatrix} + \left[\begin{pmatrix} EAX \\ EBX \\ ECX \\ EDX \\ ESP \\ EBP \\ ESI \\ EDI \end{pmatrix} * \begin{pmatrix} 1 \\ 2 \\ 4 \\ 8 \end{pmatrix} \right] + \begin{bmatrix} none \\ 8-bit \\ 16-bit \\ 32-bit \end{bmatrix}$$

Reguły adresowania

- jeśli rejestrem bazowym jest EBP lub ESP, standardowym rejestrem segmentowym jest SS, w pozostałych przypadkach jest DS,
- jeśli występują dwa rejestry, to tylko jeden może mieć współczynnik skalowania – jest on rejestrem indeksowym,
- jeśli nie ma skalowania, to pierwszy rejestr jest bazowy.

Składnia

Przemieszczenie – liczba lub zmienna symboliczna, adres efektywny jest sumą rejestrów i przemieszczenia.
Schemat adresu: $[x][y] \equiv [x + y]$, $[x][y][z] \equiv [x + y + z]$ itp.

Przykład [Wróbel(2011)], str.45

```
mov bx,ds:[10]
add cx,licznik
mov dl,[bx]
add bx,[bp+di]
mov ax, zmienna[bx][di]
add cx,[bp+si+8]
mov 6[bp][di],ch
```

Architektura

https://pl.wikipedia.org/wiki/Architektura_64-bitowa
https://pl.wikipedia.org/wiki/Intel_Core_i7
https://pl.wikipedia.org/wiki/Intel_Core_i9

Publicystyka

https://ithardware.pl/artykuly/jaki_procesor_w_2021_2022_kupic-19234.html

https://www.morele.net/wiadomosc/ranking-procesorow-2024-top-10-najwydajniejszych-cpu/745/

https://xgp.pl/artykuly/jaki-procesor-do-gier-kupic-najlepsze-modele/

Posumowanie . . . 13th and 14th generation Core

https://en.wikipedia.org/wiki/List_of_Intel_processors

Literatura dla procesorów 80x86

J. Duntemann.

Zrozumieć asembler.

Wiley – Translator, Warszawa, 1993.

D. Farbaniec.

Asembler.

Helion, Gliwice, 2012.

V. Pirogov.

Asembler, Podręcznik programisty.

Helion, Gliwice, 2005.

W. Stallings.

Organizacja i architektura systemu komputerowego.

PWN, Warszawa, wydanie XI, 2022.

E. Wróbel.

Praktyczny kurs asemblera.

Helion, Gliwice, 2011.