

Programarea microprocesorului Intel 8086. Instructiuni logice, de deplasare si rotatie

Scopul lucrării

- a) Operanzii instructiunilor microprocesorului Intel 8086.
- b) Instructiuni logice.
- c) Instructiuni de deplasare si rotatie.

1. Operanzii instructiunilor microprocesorului Intel 8086

Microprocesorul Intel 8086 utilizeaza mai multe moduri de adresare a datelor, numite generic operanzi (sursa sau destinatie), in functie de locul in care sunt plasati acestia. Operanzii pot fi continuti in registrii, in memorie, in codul instructiunii sau in porturile de intrare/iesire.

Operanzii din registre permit, datorita plasarii lor interne, ca instructiunile care ii utilizeaza sa fie executate *mai rapid*, nemaifiind necesare transferuri cu memoria sau porturile. Registrele (de 8 sau 16 biti) pot fi operanzi sursa, operanzi destinatie sau operanzi sursa si destinatie.

Operanzii de tip imediat sunt date constante de 8 sau 16 biti continute in codul instructiunii, dupa codul operatiei. Accesul la acesti operanzi este *destul de rapid*, deoarece ei sunt incarcati in coada de instructiuni de catre unitatea de interfata odata cu instructiunea. Limitarile operanzilor de tip imediat se datoreaza faptului ca ei pot fi doar valori constante si pot servi doar ca operanzi sursa.

Operanzii din memorie sunt accesati *mai lent*, deoarece sunt necesare: mai intai calculul adresei efective a operandului (EA = offset-ul), apoi calculul adresei fizice si in final transferul lor.

Formula generala de calcul pentru EA (offset) este:

$$EA = (BX / BP)^* + (SI / DI)^* + (D8 / D16)^*$$

unde: * indica un termen optional, iar
/ separa variantele unui termen.

Observatie: *spatiul de adrese al porturilor* este 0...FFFFH (64ko).

1.1. Operanzi de tip imediat (adresarea imediata)

Operandul este explicit in codul instructiunii, dupa codul operatiei, aflandu-se in coada de instructiuni.

Exemplul 1. In instructiunea:

`ADD AL, 6`

operandul al doilea este *constanta numerica 6 de tip imediat*.

Exemplul 2. In instructiunea:

`ADC ALFA, AX`

primul operand este *constanta simbolica ALFA de tip imediat*.

1.2. Operanzi din registre (adresarea directa la registru)

Operandul este in registrul specificat in codul instructiunii, in codul operatiei.

Exemplu. In instructiunea:

`ADD AL, BL`

ambii operanzi sunt de tip registru.

1.3. Operanzi din memorie (adresarea la memorie)

1. Adresarea directa la memorie.

Operandul este in memoria principala (MP), la offset-ul specificat in codul instructiunii (dupa codul operatiei). Adresa efectiva se afla in codul instructiunii.

Exemplu. In instructiunea:

`ADD AL, VAR`

al doilea operand este variabila VAR (inlocuita la asamblare cu offset-ul variabilei VAR) declarata cu o directiva. Operandul este continutul variabilei VAR din memorie.

2. Adresare indirecta la memorie prin registru.

ARHITECTURA MICROPROCESOARELOR

LUCRAREA DE LABORATOR NR. 3

Operandul este in memoria principala (MP), la offset-ul aflat in registru de baza (BX, BP) sau index (SI, DI) specificat in codul instructiunii, in codul operatiei.

Exemplul 1. In instructiunea:

`ADD AX, [BX]`

al doilea operand este perechea de octeti (cuvantul) din memoria principala, din segmentul curent de date (specificat de DS), aflat la offset-ul continut in registrul BX.

Exemplul 2. In instructiunea:

`ADD [SI], AL`

primul operand este octetul din MP, din segmentul de date curent (specificat de DS), aflat la offset-ul continut in registrul SI.

3. Adresare indirecta la memorie prin registru cu deplasare.

Operandul este in memoria principala MP, la offset-ul calculat ca suma continutului registrului de baza sau index specificat in codul operatiei si deplasarea care urmeaza in codul instructiunii dupa codul operatiei.

Exemplul 1. In instructiunea:

`ADD AX, DS: [BP+2]`

al doilea operand este cuvantul din MP, din segmentul de date curent (specificat explicit de DS), aflat la offset-ul egal cu suma dintre continutul registrului BX si deplasarea 2 (de tip imediat).

Exemplul 2. In instructiunea:

`ADD TAB [DI], AL`

primul operand este cuvantul din MP, din segmentul de date curent (specificat de DS), aflat la offset-ul egal cu suma dintre continutul registrului DI si deplasarea TAB (de tip imediat) .

4. Adresare indirecta la memorie prin doua registre.

Operandul este in memoria principala MP, la offset-ul calculat ca suma continuturilor registrelor de baza si index specificate in codul operatiei.

Exemplu. In instructiunea:

ARHITECTURA MICROPROCESOARELOR

LUCRAREA DE LABORATOR NR. 3

`ADD AX, [BX] [SI]`

al doilea operand este cuvântul din MP, din segmentul de date curent, aflat la offset-ul egal cu suma conținuturilor registrelor BX și SI.

5. Adresare indirectă la memorie prin două registre cu deplasare.

Operandul este în memoria principală MP, la offset-ul calculat ca suma conținuturilor registrelor de bază și index specificate în codul operației și deplasarea care urmează codul operației

Exemplu. În instrucțiunea:

`MOV MATR [BX] [SI], AX`

primul operand este cuvântul din MP, din segmentul de date curent, aflat la offset-ul egal cu suma conținuturilor registrelor BX și SI cu deplasarea MATR (de tip offset).

1.4. Operanți din porturi de intrare-iesire (adresarea porturilor)

1. Adresarea directă a porturilor (pentru adrese pe 8 biți, în gama 0..255).

Operandul se află în portul aflat la adresa specificată în instrucțiune după codul operației.

Exemplul 1. În instrucțiunea:

`IN AX, 20H`

al doilea operand este portul de intrare aflat la adresa numerică 20H (de tip imediat).

Exemplul 2. În instrucțiunea:

`OUT PORT2, AL`

primul operand este portul de ieșire aflat la adresa simbolică PORT2 (de tip imediat).

2. Adresarea indirectă a porturilor prin registrul DX.

Operandul se află în portul aflat la adresa specificată în registrul DX.

Exemplu. În instrucțiunea:

`OUT DX, AL`

primul operand este portul de ieșire aflat în portul de la adresa conținută în DX.

2. Instrucțiuni logice

2.1. Instructiuni logice care utilizeaza 2 operanzi

Instructiunile care utilizeaza 2 operanzi sunt: *functia si logic intre bitii operanzilor* (AND), *functia sau logic intre bitii operanzilor* (OR), *functia sau-exclusiv logic intre bitii operanzilor* (XOR).

In cele ce urmeaza *operand1* este sursa si destinatie iar *operand2* este doar sursa.

1. *Instructiunea AND* are forma:

AND *operand1*, *operand2*

si efectul: $operand1 = operand1 \text{ AND } operand2$

Exemplu (initial $AX = 5555H$ si $BX = 6666H$):

Efect:

AND AX, BX			
0101 0101 0101 0101 B	= 5555 H x		
0110 0110 0110 0110 B	= 6666 H		

0100 0100 0100 0100 B	= 4444 H		

Valoare in urma executiei instructiunii: $AX = 4444H$.

2. *Instructiunea OR* are forma:

OR *operand1*, *operand2*

si efectul: $operand1 = operand1 \text{ OR } operand2$

Exemplu (initial $AX = 5555H$ si $BX = 6666H$):

Efect:

OR AX, BX			
0101 0101 0101 0101 B	= 5555 H +		
0110 0110 0110 0110 B	= 6666 H		

0111 0111 0111 0111 B	= 7777 H		

Valoare in urma executiei instructiunii: $AX = 7777H$.

3. *Instructiunea XOR* are forma:

XOR *operand1*, *operand2*

ARHITECTURA MICROPROCESOARELOR

LUCRAREA DE LABORATOR NR. 3

si efectul: $operand1 = operand1 \text{ XOR } operand2$

Exemplu (initial $AX = 5555H$ si $BX = 6666H$):

XOR AX, BX

Efect:

0101 0101 0101 0101 B	= 5555 H +
0110 0110 0110 0110 B	= 6666 H

0011 0011 0011 0011 B	= 3333 H

Valoare in urma executiei instructiunii: $AX = 3333H$.

2.2. Instructiuni logice care utilizeaza 1 operand

Instructiunea care utilizeaza 1 operand (sursa si destinatie) este *negarea logica* (NOT) a bitilor operandului (reprezentarea in complement fata de 1 - *CI*).

In cele ce urmeaza *operand* este sursa si destinatie.

Instructiunea NOT are forma:

NOT *operand*

si efectul: $operand = NOT \text{ operand}$

Exemplul 1 (initial $SI = 0001H$):

NOT SI

Efect:

0000 0000 0000 0001 B	= 0001 H \

1111 1111 1111 1110 B	= FFFE H

Valoare in urma executiei instructiunii: $SI = FFFE H$.

Exemplul 2 (initial $DI = 0FFFFH$):

NOT DI

Efect:

1111 1111 1111 1111 B	= FFFF H \

0000 0000 0000 0000 B	= 0000 H

ARHITECTURA MICROPROCESOARELOR

LUCRAREA DE LABORATOR NR. 3

Valoare in urma executiei instructiunii: $DI = 0000H$.

Exemplul 3 (initial $AX = 7FFFH$):

	NOT	AX	
Efect:			
		0111 1111 1111 1111 B	= 7FFF H \

		1000 0000 0000 0000 B	= 8000 H

Valoare in urma executiei instructiunii: $AX = 8000H$.

Exemplul 4 (initial $BX = 8000H$):

	NOT	BX	
Efect:			
		1000 0000 0000 0000 B	= 8000 H \

		0111 0111 0111 0111 B	= 7FFF H

Valoare in urma executiei instructiunii: $BX = 7FFFH$.

Exemplul 5 (initial $CX = 4000H$):

	NOT	CX	
Efect:			
		0100 0000 0000 0000 B	= 4000 H \

		1100 0000 0000 0000 B	= C000 H

Valoare in urma executiei instructiunii: $CX = C000H$.

Exemplul 6 (initial $DX = 0C000H$):

	NOT	DX	
Efect:			
		1100 0000 0000 0000 B	= C000 H \

		0011 1111 1111 1111 B	= 3FFF H

Valoare in urma executiei instructiunii: $DX = 3FFFH$.

3. Instructiuni de deplasare si rotatie

3.1. Instructiuni de deplasare

Instructiunile de deplasare executa operatii asupra unui singur operand (sursa si destinatie).

In cele ce urmeaza *operand* este sursa si destinatie.

1. *Instructiunea de deplasare logica la stanga* are forma:

SHL *operand*, 1

si efectul: *operand* = *operand* deplasat la stanga cu o pozitie

sau SHL *operand*, CL

respectiv *operand* = *operand* deplasat la stanga cu (CL) poz.

Exemplul 1 (initial $AX = AAAAH$):

Efect: SHL AX, 1

1010 1010 1010 1010 B	= AAAA H (<-----) ¹

0101 0101 0101 0100 B	= 5 5 5 4 H

Valori in urma executiei instructiunii: $AX = 5554H$, $CF = 1$.

Exemplul 2 (initial $AX = AAAAH$ si $CL = 4$):

Efect: SHL AX, CL

1010 1010 1010 1010 B	= AAAA H (<-----) ^{(CL)=4}

1010 1010 1010 0000 B	= AAA0 H

Valori in urma executiei instructiunii: $AX = AAA0H$, $CF = 0$.

Se observa ca *deplasarea logica la stanga cu n pozitii* are acelasi efect cu *inmultirea cu 2^n* .

Astfel, pentru primul exemplu avem ($n = 1$, $2^n = 2^1 = 2$):

ARHITECTURA MICROPROCESOARELOR

LUCRAREA DE LABORATOR NR. 3

$$AAAA\ H * 2\ H = 1\ 5\ 5\ 4\ H$$

iar pentru al doilea ($n = 4$, $2^n = 2^4 = 16 = 10\ H$):

$$AAAA\ H * 10\ H = AAAA0\ H$$

Se observa de asemenea ca in primul caz rezultatul corect este continut in CF (1) si AX (5554H), pe cand in al doilea caz se pierde primii biti ai rezultatului.

2. Instructiunea de deplasare logica la dreapta are forma:

SHR operand, 1

si efectul: operand = operand deplasat la dreapta cu o pozitie

sau SHR operand, CL

respectiv operand = operand deplasat la dreapta cu (CL) poz.

Exemplul 1 (initial AX = AAAAH):

SHR AX, 1

Efect:

1010 1010 1010 1010 B	= AAAA H	1	(----->)
0101 0101 0101 0101 B	= 5 5 5 5 H		

Valori in urma executiei instructiunii: AX = 5555H, CF = 0.

Exemplul 2 (initial AX = AAAAH si CL = 4):

SHR AX, CL

Efect:

1010 1010 1010 1010 B	= AAAA H	(CL)=4	(----->)
0000 1010 1010 1010 B	= 0AAA H		

Valori in urma executiei instructiunii: AX = 0AAAH, CF = 1.

Se observa ca deplasarea logica la dreapta cu n pozitii are acelasi efect cu impartirea la 2^n .

Astfel, pentru primul exemplu avem ($n = 1$, $2^n = 2^1 = 2$):

ARHITECTURA MICROPROCESOARELOR

LUCRAREA DE LABORATOR NR. 3

AAAA H : 2 H = 5 5 5 5 H

iar pentru al doilea ($n = 4$, $2^n = 2^4 = 16 = 10$ H):

AAAA H : 10 H = AAA H

3. Instructiunea de deplasare aritmetica la stanga are forma:

SAL operand, 1

si efectul: operand = operand deplasat la stanga cu o pozitie

sau SAL operand, CL

respectiv operand = operand deplasat la stanga cu (CL) poz.

Exemplul 1 (initial AX = AAAAH):

SAL AX, 1

Efect:

		1
1010 1010 1010 1010 B	=	AAAA H (<-----)

0101 0101 0101 0100 B	=	5 5 5 4 H

Valori in urma executiei instructiunii: AX = 5554H, CF = 1.

Exemplul 2 (initial AX = AAAAH si CL = 4):

SAL AX, CL

Efect:

		(CL)=4
1010 1010 1010 1010 B	=	AAAA H (<-----)

1010 1010 1010 0000 B	=	AAA 0 H

Valori in urma executiei instructiunii: AX = AAA0H, CF = 0.

Se observa faptul ca efectul acestei instructiuni este identic cu cel al instructiunii de deplasare logica la stanga !.

4. Instructiunea de deplasare aritmetica la dreapta (cu pastrarea semnului) are forma:

SAR operand, 1

ARHITECTURA MICROPROCESOARELOR

LUCRAREA DE LABORATOR NR. 3

si efectul: $operand = operand$ depl. *aritmetic* la dr. cu o poz.

sau SAR $operand, CL$

respectiv $operand = operand$ depl. *aritmetic* la dr cu (CL) poz

Exemplul 1 (initial $AX = AAAAH$):

Efect: SAR AX, 1

1010 1010 1010 1010 B	= AAAA H (----->)

1101 0101 0101 0101 B	= D5 5 5 H

Valori in urma executiei instructiunii: $AX = D555H$, $CF = 0$.

Exemplul 2 (initial $AX = AAAAH$ si $CL = 4$):

Efect: SAR AX, CL

1010 1010 1010 1010 B	= AAAA H (----->)

1111 1010 1010 1010 B	= FAAA H

Valori in urma executiei instructiunii: $AX = FAAA H$, $CF = 1$.

3.2. Instructiuni de rotatie

Instructiunile de rotatie executa operatii asupra unui singur operand (sursa si destinatie).
In cele ce urmeaza *operand* este sursa si destinatie.

1. *Instructiunile de rotatie la stanga fara CF* au forma:

ROL $operand, 1$

si efectul: $operand = operand$ rotit spre stanga cu o pozitie

sau ROL $operand, CL$

respectiv $operand = operand$ rotit spre stanga cu (CL) poz.

Exemplul 1 (initial $AX = AAAAH$):

Efect: ROL AX, 1

1010 1010 1010 1010 B	= AAAA H (rot. stg. cu 1)
-----------------------	---------------------------

ARHITECTURA MICROPROCESOARELOR

LUCRAREA DE LABORATOR NR. 3

0101 0101 0101 0101 B = 5 5 5 5 H

Valori in urma executiei instructiunii: $AX = 5555H$, $CF = 1$.

Exemplul 2 (initial $AX = AAAAH$ si $CL = 4$):

ROL AX, CL

Efect:

1010 1010 1010 1010 B = AAAA H (rot. stg. cu CL)

1010 1010 1010 **1010** B = AAAA H

Valori in urma executiei instructiunii: $AX = AAAAH$, $CF = 0$.

2. Instructiunile de rotatie la stanga cu CF au forma:

RCL operand, 1

si efectul: $operand = (operand, CF)$ rotit spre stanga cu o poz.

sau RCL operand, CL

respectiv $operand = (operand, CF)$ rotit spre stg. cu (CL) poz.

Exemplul 1 (initial $AX = AAAAH$ si $CF = 1$):

RCL AX, 1

Efect:

(CF = 1) 1010 1010 1010 1010 B = AAAA H (rot. stg. cu 1)

(CF = 1) 0101 0101 0101 010**1** B = 5 5 5 5 H

Valori in urma executiei instructiunii: $AX = 5555H$, $CF = 1$.

Exemplul 2 (initial $AX = AAAAH$ si $CF = 0$):

RCL AX, 1

Efect:

(CF = 0) 1010 1010 1010 1010 B = AAAA H (rot. stg. cu 1)

(CF = 1) 0101 0101 0101 010**0** B = 5 5 5 4 H

ARHITECTURA MICROPROCESOARELOR

LUCRAREA DE LABORATOR NR. 3

Valori in urma executiei instructiunii: $AX = 5554H$, $CF = 1$.

Exemplul 3 (initial $AX = AAAAH$, $CL = 4$ si $CF = 1$):

RCL AX, CL

Efect:

(CF = 1) 1010 1010 1010 1010 B = AAAA H (rot. stg. cu CL)

(CF = 0) 1010 1010 1010 1101 B = AAAD H

Valori in urma executiei instructiunii: $AX = AAADH$, $CF = 0$.

Exemplul 4 (initial $AX = AAAAH$, $CL = 4$ si $CF = 0$):

RCL AX, CL

Efect:

(CF = 0) 1010 1010 1010 1010 B = AAAA H (rot. stg. cu CL)

(CF = 0) 1010 1010 1010 0101 B = AAA5 H

Valori in urma executiei instructiunii: $AX = AAA5H$, $CF = 0$.

3. Instructiunile de rotatie la dreapta fara CF au forma:

ROR operand, 1

si efectul: operand = operand rotit spre dreapta cu o pozitie

sau ROR operand, CL

respectiv operand = operand rotit spre dreapta cu (CL) poz.

Exemplul 1 (initial $AX = AAAAH$):

ROR AX, 1

Efect:

1010 1010 1010 1010 B = AAAA H (rot. dr. cu 1)

0101 0101 0101 0101 B = 5 5 5 5 H

ARHITECTURA MICROPROCESOARELOR

LUCRAREA DE LABORATOR NR. 3

Valori în urma executiei instructiunii: $AX = 5555H$, $CF = 0$.

Exemplul 2 (initial $AX = AAAAH$ si $CL = 4$):

ROR AX, CL

Efect:

1010 1010 1010 1010 B	= AAAA H (rot. dr. cu CL)

1010 1010 1010 1010 B	= AAAA H

Valori în urma executiei instructiunii: $AX = AAAAH$, $CF = 1$.

4. Instructiunile de rotatie la dreapta cu CF au forma:

RCR operand, 1

si efectul: $operand = (operand, CF)$ rotit spre dreapta cu o poz.

sau

RCR operand, CL

respectiv $operand = (operand, CF)$ rotit spre dr. cu (CL) poz.

Exemplul 1 (initial $AX = AAAAH$ si $CF = 1$):

RCR AX, 1

Efect:

1010 1010 1010 1010 (CF = 1) B	= AAAA H (rot. dr. cu 1)

1101 0101 0101 0101 (CF = 0) B	= D 5 5 5 H

Valori în urma executiei instructiunii: $AX = D555H$, $CF = 0$.

Exemplul 2 (initial $AX = AAAAH$ si $CF = 0$):

RCR AX, 1

Efect:

1010 1010 1010 1010 (CF = 0) B	= AAAA H (rot. dr. cu 1)

0101 0101 0101 0101 (CF = 0) B	= 5 5 5 5 H

Valori în urma executiei instructiunii: $AX = 5555H$, $CF = 0$.

ARHITECTURA MICROPROCESOARELOR

LUCRAREA DE LABORATOR NR. 3

Exemplul 3 (initial $AX = AAAAH$, $CL = 4$ si $CF = 1$):

RCR AX, CL

Efect:

```

1010 1010 1010 1010 (CF = 1) B = AAAA H (rot. dr. cu CL)
-----
0101 1010 1010 1101 (CF = 1) B = 5AAA H

```

Valori in urma executiei instructiunii: $AX = 5AAAH$, $CF = 1$.

Exemplul 4 (initial $AX = AAAAH$, $CL = 4$ si $CF = 0$):

RRC AX, CL

Efect:

```

1010 1010 1010 1010 (CF = 0) B = AAAA H (rot. dr. cu CL)
-----
1010 1010 1010 0101 (CF = 1) B = AAAA H

```

Valori in urma executiei instructiunii: $AX = AAAAH$, $CF = 1$.

4. Exemple de programe

4.1. Calcule in dubla precizie

Rolul principal al bitului de transport in instructiunile de deplasare si rotatie este acela de a permite *efectuarea calculelor in dubla precizie (pe 32 biti)*.

1. Deplasarea cu o pozitie la stanga a unei valori de 32 biti - perechea de registre (AX, BX):

Initial: $AX = AAAAH$ si $BX = 8888H$.

```

SHL AX, 1      ; deplasarea MSW
SHL BX, 1      ; deplasarea LSW
ADC AX, 0      ; adunarea bitului CF la LSW

```

Efectul secventei este:

```

      1010 1010 1010 1010      1000 1000 1000 1000 B    = AAAA  8 8 8 8 H
      //// //// //// ////      //// //// //// ////
-----
(CF = 1)0101 0101 0101 0101    0001 0001 0001 0000 B    = 5 5 5 5  1 1 1 0 H

```

In urma executiei secventei: $AX = 5555H$, $BX = 1110H$, $CF = 1$ (rezultat 1 5555 1110H).

ARHITECTURA MICROPROCESOARELOR

LUCRAREA DE LABORATOR NR. 3

2. Deplasarea cu o pozitie la dreapta a unei valori de 32 biti - perechea de registre (AX, BX):

Initial: $AX = AAAAH$ si $BX = 8888H$.

SHR BX, 1	; deplasarea LSW
SHR AX, 1	; deplasarea MSW
ADC DX, 0	; adunarea bitului CF la DX
MOV CL, 15	; pregatire deplasare 15 biti
SHL DX, CL	; deplasare CF pe pozitia MSb
ADD BX, DX	; adunarea bitului CF la MSW

Efectul secvenței este:

1010 1010 1010 1010 \\\ / \\\ / \\\ / \\\ /	1000 1000 1000 1000 \\\ / \\\ / \\\ / \\\ /	B = AAAA 8 8 8 8 H
0101 0101 0101 0101	0100 0100 0100 0100 (CF = 0)	B = 5 5 5 5 4 4 4 4 H

In urma executiei secventei: $AX = 5555H$, $BX = 4444H$, $CF = 0$ (rezultat 5555 4444H).

3. Rotatia cu o pozitie spre stanga a unei valori de 32 biti - perechea de registre (AX, BX):

Initial: $AX = AAAAH$, $BX = 8888H$, $DX = 0000H$, $CL = 4$ si $CF = 1$.

ROL	AX, 1	; rotirea MSW
RCL	BX, 1	; rotirea (LSW,CF)
ADC	AX, 0	; adunarea la MSW a bitului CF
		; obtinut prin rotirea lui BX

Efectul secvenței este:

1010 1010 1010 1010	1000 1000 1000 1000 B	= AAAA 8 8 8 8 H
//// //// //// ////	//// //// //// ////	
<hr/>		
(CF = 1)0101 0101 0101 0101	0001 0001 0001 0001 B	= 5 5 5 5 1 1 1 1 H

In urma executiei secventei: $AX = 5555H$, $BX = 1111H$, $CF = 1$ (rezultat 1 5555 1111H).

4. Rotatia cu o pozitie spre dreapta a unei valori de 32 biti - perechea de registre (AX, BX):

Initial: $AX = AAAAH$, $BX = 8888H$, $DX = 0000H$, $CL = 4$ si $CF = 1$.

ROR AX, 1 ; rotirea cuvântului superior (MSW)
RCR BX, 1 ; rotirea cuvântului inferior (LSW) cu CF
; de la rotirea lui AX

ARHITECTURA MICROPROCESOARELOR

LUCRAREA DE LABORATOR NR. 3

```

ADC DX, 0      ; aducerea in DX a lui CF de la rotirea lui BX
ROR DX, 1      ; plasarea in DX a lui CF obtinut prin rotirea
                ; lui BX, pe cea mai semnificativa pozitie
ADD AX, DX     ; adunarea la cuvantul superior a bitului
                ; CF obtinut prin rotirea lui BX

```

Efectul secventei este:

```

1010 1010 1010 1010      1000 1000 1000 1000      B = AAAA 8 8 8 8 H
  \ \ \ \ \ \ \ \ \ \ \ \  \ \ \ \ \ \ \ \ \ \ \ \
-----
0101 0101 0101 0101 (CF = ) 0100 0100 0100 0100 (CF = 0) B = 5 5 5 5 4 4 4 4 H

```

In urma executiei secventei: $AX = 5555H$, $BX = 4444H$, $CF = 0$ (rezultat 5555 4444H).

5. Program de deplasare in dubla precizie a continutului unei variabile v cu 2 pozitii catre dreapta.

```

data segment
    v dw 0abcdh, 0ef12h
data ends
assume cs: cod, ds: data
cod segment
st:
    mov ax, data
    mov ds, ax
    mov ax, x          ; initializari
    mov bp, offset v
    mov ax, ds:[bp]
    mov bx, ds:[bp+2]
    mov dx, 0          ; prima deplasare
    shr ax, 1
    shr bx, 1
    adc dx, 0
    ror dx, 1
    adc ax, 0
    mov dx, 0          ; a doua deplasare
    shr ax, 1
    shr bx, 1
    adc dx, 0
    ror dx, 1
    adc ax, 0

```

ARHITECTURA MICROPROCESOARELOR

LUCRAREA DE LABORATOR NR. 3

```
        mov ds:[bp], ax          ; stocare rezultat
        mov ds:[bp+2], bx
        mov ah, 4ch
        int 21h
cod     ends
end     st
```

4.2. Alte programe

1. Program de calcul al produsului unei variabile x cu valoarea 7. Rezultatul plasat in memorie in variabila y.

```
data segment
    x dw 120h
    y dw ?
data     ends
assume cs: cod, ds: data
cod     segment
st:
    mov ax, data
    mov ds, ax
    mov ax, x          ; copiere x in AX
    mov y, ax          ; copiere x in y (y = x)
    shl ax, 1          ; calcul 2x
    add y, ax          ; adunare 2x la y (y = x+2x = 3x)
    shl ax, 1          ; calcul 4x
    add y, ax          ; adunare 4x la y (y = x+2x+4x = 7x)
    mov ah, 4ch
    int 21h
cod     ends
end     st
```

2. Program de calcul al produsului unei variabile x cu valoarea 0,75 ($=3/4$). Rezultatul plasat in memorie in variabila y.

```
data segment
    x dw 120h
    y dw ?
data     ends
assume cs: cod, ds: data
cod     segment
st:
    mov ax, data
    mov ds, ax
```

; Varianta 1

ARHITECTURA MICROPROCESOARELOR

LUCRAREA DE LABORATOR NR. 3

```
mov ax, x           ; copiere x in AX
shr ax, 1           ; calcul x/2
mov y, ax           ; copiere x/2 in y (y = x/2)
shr ax, 1           ; calcul x/4
add y, ax           ; adunare x/4 la y (y = x/2 + x/4 = 3x/4)

; Varianta 2
mov ax, x           ; copiere x in AX
mov y, ax           ; copiere x in y (y = x)
shr ax, 1           ; calcul x/2
shr ax, 1           ; calcul x/4
sub y, ax           ; scadere x/4 din y (y = x - x/4 = 3x/4)
mov ah, 4ch
int 21h
cod ends
end st
```

5. Desfasurarea lucrarii

1. Se editeaza *programul de calcul al produsului unei variabile x cu valoarea 7* (vezi §4.2.) intr-un fisier cu numele *AP31.ASM*.

Se parcurg etapele 1.a) ... 1.f) de la lucrarea nr. 2 pentru acest program.

2. Se concepe si editeaza un *program de calcul al produsului unei variabile z cu valoarea 3,25* intr-un fisier cu numele *AP32.ASM*.

Se parcurg etapele 1.a) ... 1.f) de la lucrarea nr. 2 pentru acest program.

3. Se concepe si editeaza un *program de calcul al expresiei $e = (2x + 1)(x/2 - 1)$* intr-un fisier cu numele *AP33.ASM*.

6. Teme si exercitii

1. Sa se scrie un *program care sa utilizeze calculul in dubla precizie pentru a deplasa la stanga cu 5 pozitii un cuvint din memorie folosind numai registrele de tip octet (AH, AL, BH, BL, CH, CL, DH, DL)*.

2. Sa se scrie un *program care sa utilizeze calculul in dubla precizie pentru a roti catre dreapta cu 5 pozitii un cuvint din memorie folosind numai registrele de tip octet (AH, AL, BH, BL, CH, CL, DH, DL)*.

7. Intrebari

1. Care sunt tipurile de operanzi ai instructiunilor microprocesorului Intel 8086 ?
2. Care sunt modurile de adresare a memoriei principale ?
3. Care sunt modurile de adresare a porturilor ?
4. Care este diferenta intre negatia logica (NOT) si cea aritmetica (NEG) ?
5. Care este diferenta intre deplasarea logica la dreapta (SHR) si cea aritmetica (SAR) ?
6. Care este diferenta intre deplasările logice si rotatiile fara CF ?
7. Care este diferenta intre rotatiile fara CF si rotatiile cu CF ?