سازمان سما

وابسته به دانشگاه آزاد اسلامی

دانشگاه سما واحد حاجی آباد

حل مسئله
معماری کامپیوتر

منبع : معماری کامپیوتر-منوچهر بابایی

WWW.HREZAPOUR.IR

حمیدرضا رضاپور

معماری کامپیوتر

درس چهارم: برنامه نویسی کامپیوتر پایه-اسمبلی (براساس کتاب مانو)

(زبان ماشین-زبان اسمبلی)

- برنامه:مجموعه ای از دستورالعمل هابرای هدایت کامپیوتر در اجرای یک کار خاص می باشد.
- زبان ماشین: برنامه های دودویی قابل اجرا توسط کامپیوتر به زبان ماشین هستند
- زبان اسمبلی:کد سمبلیک برنامه را که بخش های مختلف یک دستور شامل بخش عمل،آدرس ثبات ها و ... را بصورت سمبلیک و کلمات بامعنی لاتین مشخص می کند را برنامه به زبان اسمبلی گویند و ترجمه این کد به زبان ماشین توسط اسمبلر انجام می شود.

ADD A,B A ← A+B

Symbol	*Hex code*	*Description*
AND	0 or 8	AND M to AC
ADD	1 or 9	Add M to AC, carry to E
LDA	2 or A	Load AC from M
STA	3 or B	Store AC in M
BUN	4 or C	Branch unconditionally to m
BSA	5 or D	Save return address in m and branch to m+1
ISZ	6 or E	Increment M and skip if zero
CLA	7800	Clear AC
CLE	7400	Clear E
CMA	7200	Complement AC
CME	7100	Complement E
CIR	7080	Circulate right E and AC
CIL	7040	Circulate left E and AC
INC	7020	Increment AC, carry to E
SPA	7010	Skip if AC is positive
SNA	7008	Skip if AC is negative
SZA	7004	Skip if AC is zero
SZE	7002	Skip if E is zero
HLT	7001	Halt computer
INP	F800	Input information and clear flag
OUT	F400	Output information and clear flag
SKI	F200	Skip if input flag is on
SKO	F100	Skip if output flag is on
ION	F080	Turn interrupt on
IOF	F040	Turn interrupt off

(برنامه دودویی-شانزده شانزدهی جمع دو عدد)

Location	Instruction Code
0	0010 0000 0000 0100
1	0001 0000 0000 0101
10	0011 0000 0000 0110
11	0111 0000 0000 0001
100	0000 0000 0101 0011
101	1111 1111 1110 1001
110	0000 0000 0000 0000

Binary code

Location	Instruction
000	2004
001	1005
002	3006
003	7001
004	0053
005	FFE9
006	0000

Hexadecimal code

(برنامه سمبلیک جمع دو عدد)

Location		Instruction	Comments
000	Ac ← 53	LDA 004	Load 1st operand into AC
001	Ac ← Ac+FFE9	ADD 005	Add 2nd operand to AC
002		STA 006	Store sum in location 006
003		HLT	Halt computer
004		0053	1st operand
005		FFE9	2nd operand (negative)
006		0000	Store sum here

(برنامه اسمبلی جمع دو عدد)

```
        ORG 0      /Origin of program is location 0
Ac←A    LDA  A     /Load  operand from location A
Ac←Ac+B ADD  B     /Add  operand from location B
C← A+B  STA  C     /Store sum in location C
        HLT        /Halt computer
A,  DEC (83)10     /Decimal operand
B,  DEC (-23)10    /Decimal operand
C,  DEC  0         /Sum stored in location C
        END        /End of symbolic program
```

(قواعد زبان اسمبلی)

هرخط از برنامه به زبان اسمبلی بصورت سه ستون ،که به هرستون میدان گویند و بصورت زیر می باشد:

1. میدان عنوان(برچسب)که می تواند خالی باشد یا یک آدرس سمبلیک باشد
2. میدان دستورالعمل که یک دستورالعمل ماشین یا یک شبه دستورالعمل است
3. میدان توضیحات که میتواند خالی و یا حاوی یک توضیح باشد.

میدان دستورالعمل در زبان اسمبلی یکی از حالت های زیر است:

1. دستورالعمل های حافظه ای(MRI)
2. دستورالعمل های ثباتی یا ورودی-خروجی از نوع غیرارجاع به حافظه(non-MRI)
3. شبه دستورالعمل ها باعملوند یا بدون عملوند

non – MRI	MRI	Pseudo Instruction
CLA	ADD A	ORG N
INC	ADD B I	END
CLE		HEX N – DEC N

(برنامه اسمبلی تفریق دو عدد)

$MIN + \overline{SUB} + 1$

		ORG 100	/ Origin of program is location 100
$AC \leftarrow SUB$		LDA SUB	/ Load subtrahend to AC
$\overline{SUB}$		CMA	/ Complement AC
$\overline{SUB} + 1$		INC	/ Increment AC
$MIN + \overline{SUB} + 1$		ADD MIN	/ Add minuend to AC
		STA DIF	/ Store difference
		HLT	/ Halt computer
	MIN,	DEC 83	/ Minuend
	SUB,	DEC -23	/ Subtrahend
	DIF,	HEX 0	/ Difference stored here
		END	/ End of symbolic program
	Label	Instruction	comment

(اسمبلر)

- اسمبلر برنامه ای است که برنامه به زبان اسمبلی را به معادل باینری (زبان ماشین) تبدیل می کند.
- به برنامه زبان اسمبلی ،برنامه منبع و برنامه به زبان ماشین ،برنامه مقصد گفته می شود.
- اسمبلر برای تولید کد زبان ماشین دو مرور روی کد منبع دارد:
  - مرور اول:تولید یک جدول شامل معادل دودویی همه سمبل های تعریف شده در برنامه منبع
  - مرور دوم:ترجمه به کد دودویی در مرور دوم انجام می شود

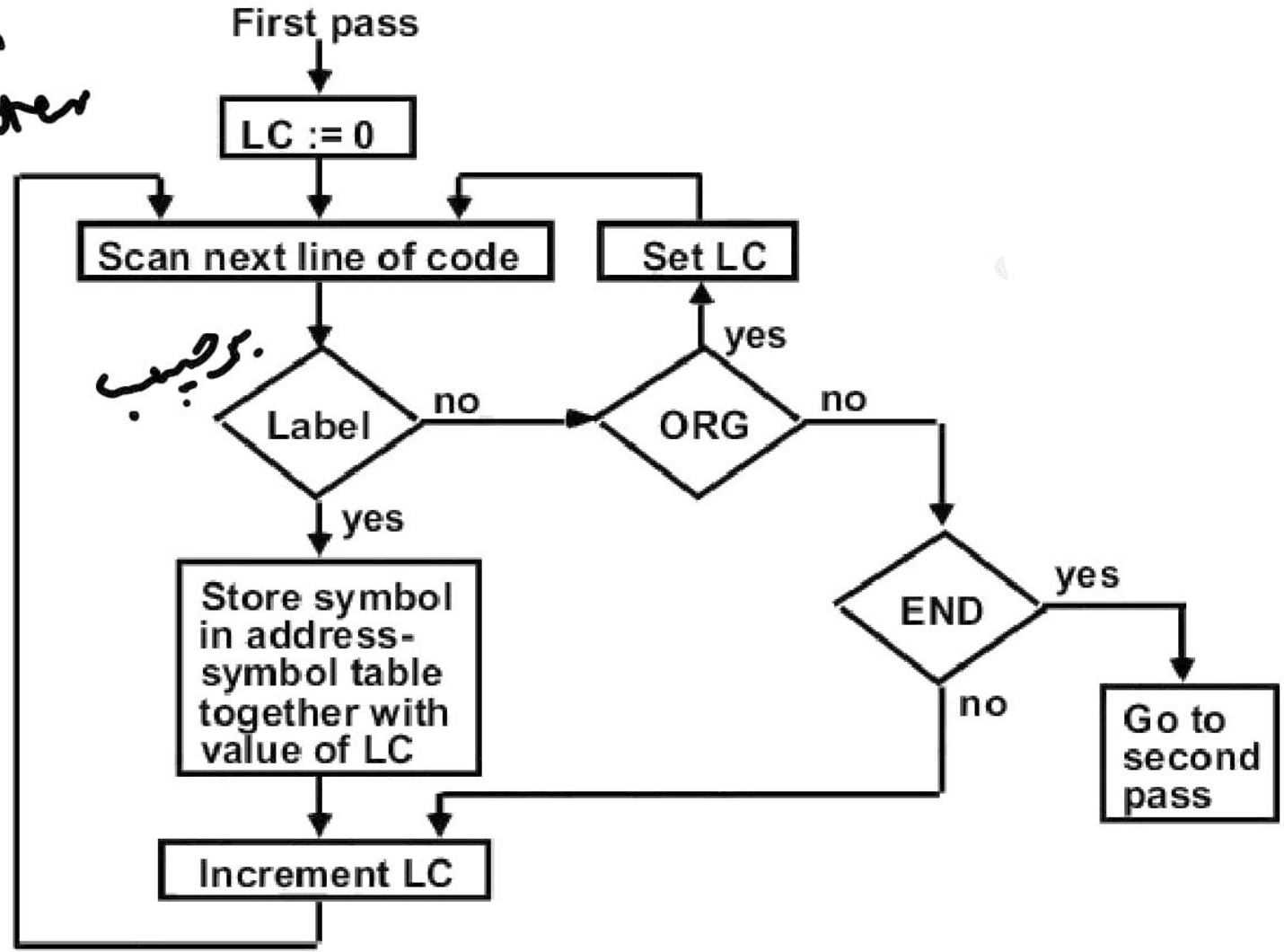
- PL3, LDA SUB I
- By referring to the ASCII code table, we get:

Memory word	Symbol	Hex code
1	P L	50 4C
2	3 ,	33 2C
3	L D	4C 44
4	A ~~A~~	41 20
5	S U	53 55
6	B ~~B~~	42 20
7	I CR	49 0D

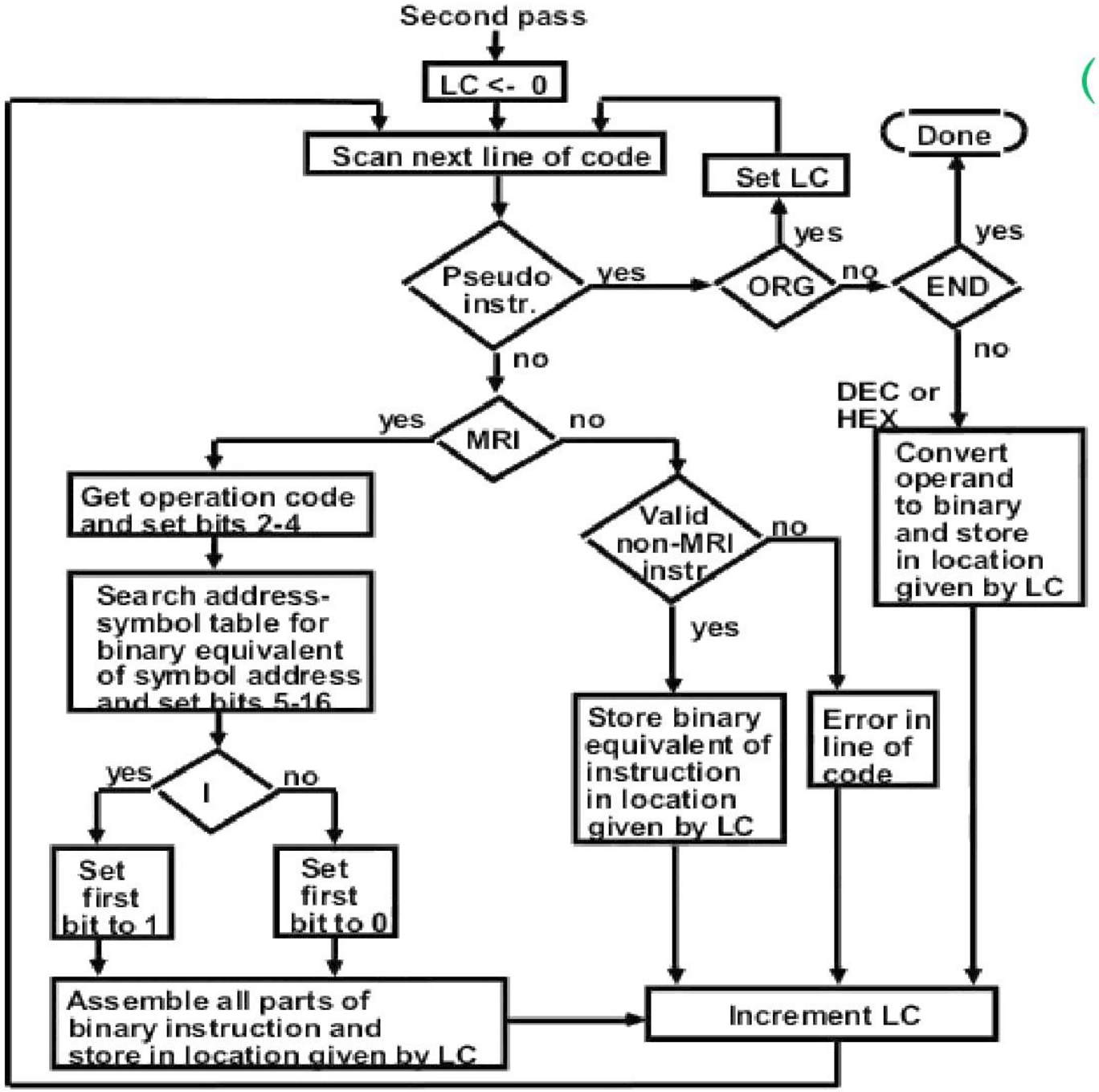
Carriage

(مرور اول اسمبلر)

Location counter



(مرور دوم اسمبلر)



HEX 50
DEC

(حلقه در برنامه نویسی اسمبلی)

```
int i;
int x[100];
int S=0;
for(i=0;i<100;i++)
{
    S=S+x[i];
}
```


```
        ORG  100      / Origin of program is HEX 100
        LDA  ADS      / Load first address of operand
        STA  PTR      / Store in pointer
        LDA  NBR      / Load -100
        STA  CTR      / Store in counter
        CLA           / Clear AC
LOP,    ADD  PTR  I   / Add an operand to AC
        ISZ  PTR      / Increment pointer
        ISZ  CTR      / Increment counter
        BUN  LOP      / Repeat loop again
        STA  SUM      / Store sum
        HLT           / Halt
ADS,    HEX  150      / First address of operands
PTR,    HEX  0        / Reserved for a pointer
NBR,    DEC  -100     / Initial value for a counter
CTR,    HEX  0        / Reserved for a counter
SUM,    HEX  0        / Sum is stored here
        ORG  150      / Origin of operands is HEX 150
150 →   DEC  75       / First operand
        .
        .
        DEC  23       / Last operand
        END           / End of symbolic program
```

مثال (۱):برنامه ای به زبان اسمبلی بنویسید که دو عدد را با استفاده از جمع در هم ضرب کند.

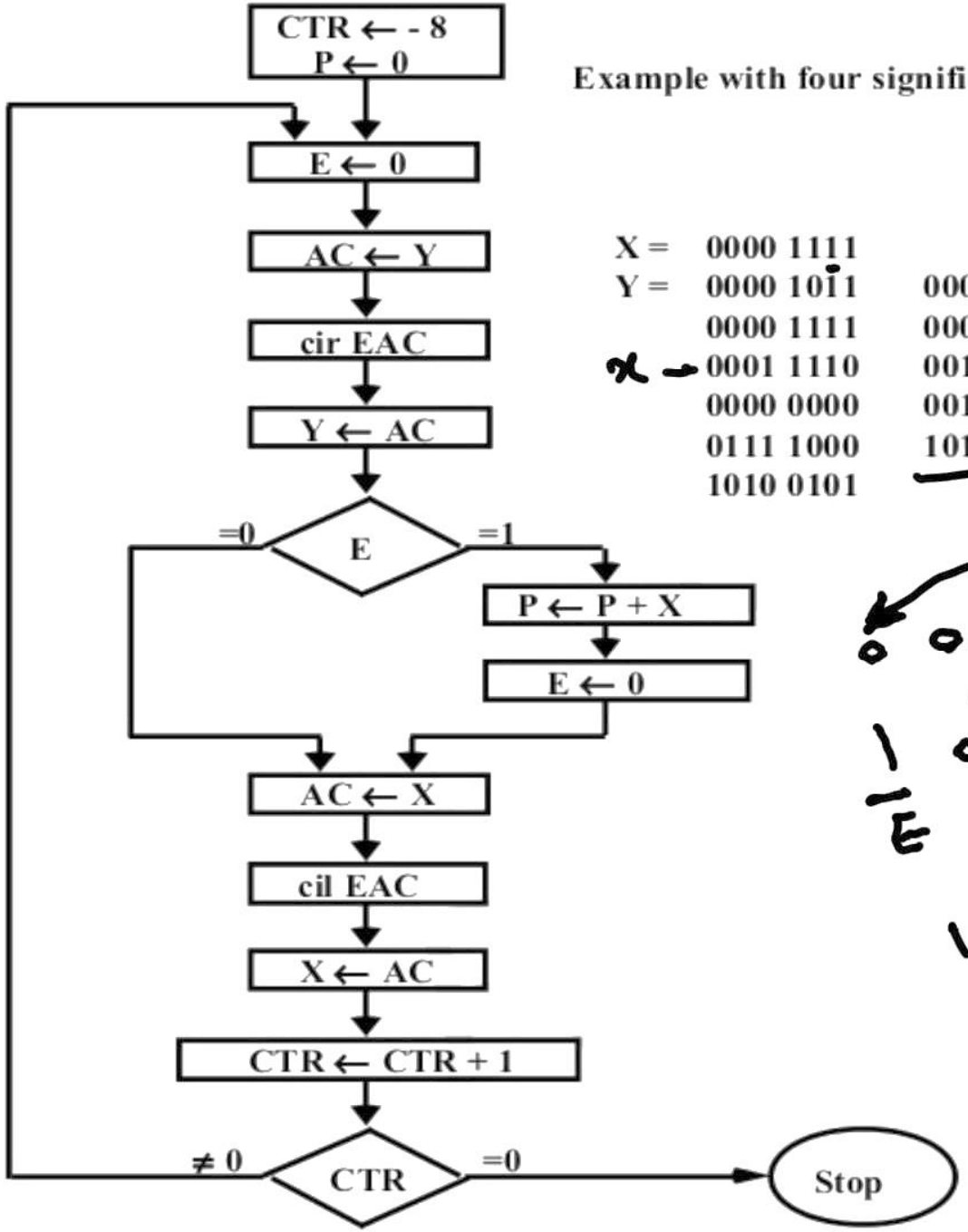
```
        ORG 100
        LDA B
B̄       CMA
B̄+1     INC
        STA CTR
zero,   SZA
        BUN ST
        BUN STP   R
ST,     CLA
LOOP,   ADD A
        ISZ CTR
        BUN LOOP
R       STA Res
STP,    HLT
A,      -
B,      -
CTR,    HEX 0
Res,    HEX 0
```

A × B

A + A + A + ⋮ + A با B

$-B = \overline{B} + 1$

(فلوچارت ضرب دو عدد)



Example with four significant digits

		P
X =	0000 1111	
Y =	0000 1011	0000 0000
	0000 1111	0000 1111
	0001 1110	0010 1101
	0000 0000	0010 1101
	0111 1000	1010 0101
	1010 0101	

X holds the multiplicand مضروب
Y holds the multiplier ضرب کننده
P holds the product

0 0000 1011

1 00000101

E Ac

110000010

Ac

x 101
y 110
000
1010
10100
0111110

(برنامه ضرب دو عدد)

```
        ORG  100
LOP,    CLE          / Clear E
        LDA  Y       / Load multiplier
        CIR          / Transfer multiplier bit to E
        STA  Y       / Store shifted multiplier
        SZE          / Check if bit is zero
        BUN  ONE     / Bit is one; goto ONE
        BUN  ZRO     / Bit is zero; goto ZRO
ONE,    LDA  X       / Load multiplicand
        ADD  P       / Add to partial product
        STA  P       / Store partial product
        CLE          / Clear E
ZRO,    LDA  X       / Load multiplicand
        CIL          / Shift left
        STA  X       / Store shifted multiplicand
        ISZ  CTR     / Increment counter
        BUN  LOP     / Counter not zero; repeat loop
        HLT          / Counter is zero; halt
CTR,    DEC  -8      / This location serves as a counter
X,      HEX  000F    / Multiplicand stored here
Y,      HEX  000B    / Multiplier stored here
P,      HEX  0       / Product formed here
        END
```

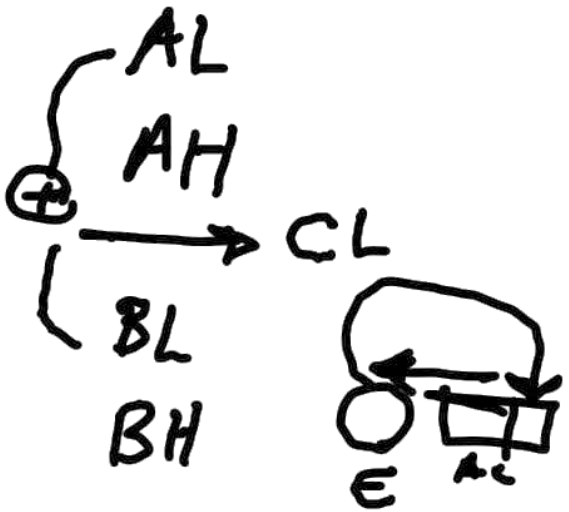
E=1

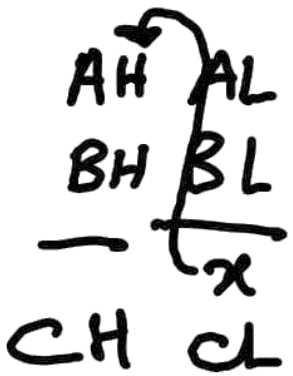
P=P+X

E=0

X را یک واحد به چپ شیفت

(برنامه جمع دو عدد با دقت مضاعف)






```
LDA   AL        / Load A low
ADD   BL        / Add B low, carry in E
STA   CL        / Store in C low
CLA             / Clear AC
CIL             / Circulate to bring carry into AC(16)
ADD   AH        / Add A high and carry
ADD   BH        / Add B high
STA   CH        / Store in C high
HLT
                / Location of operands
AL,   _____
AH,   _____
BL,   _____
BH,   _____
CL,   _____
CH,   _____
```

(انجام عمل منطقی OR با استفاده از AND)

$$x+y=(x'y')'$$

$\bar{A}$	LDA	A	/ Load 1st operand
	CMA		/ Complement to get A’
	STA	TMP	/ Store in a temporary location
$\bar{B}$	LDA	B	/ Load 2nd operand B
	CMA	$\bar{A}$	/ Complement to get B’
$\bar{A}\bar{B}$	AND	TMP	/ AND with A’ to get A’ AND B’
	CMA		/ Complement again to get A OR B

$A+B$

A or B

مثال (۱):برنامه ای بنویسید که XOR منطقی دو عملوند منطقی را حساب کند.

$$A \text{ xor } B = A'B + AB' = ((A'B)'(AB')')'$$

```
LDA  A
CMA
AND  B
CMA
STA  Temp
LDA  B
CMA
AND  A
CMA
AND  Temp
CMA
```

$\overline{(A'B)}$

$\overline{(AB')}$

(شیفت های منطقی)

- Logical shift right

  CLE

  CIR

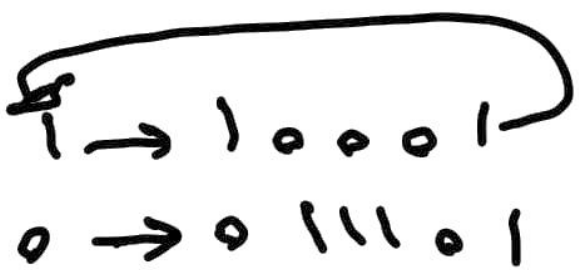
- Logical shift left

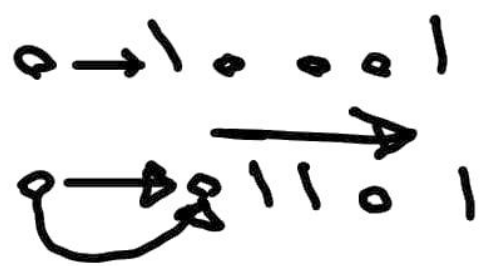
  CLE

  CIL

(شیفت حسابی به راست)

CLE	/ Clear E to 0
SPA	/ Skip if AC is positive, E remains 0
CME	/ AC is negative, set E to 1
CIR	/ Circulate E and AC





(شیفت حسابی به چپ)

در این نوع شیفت،بیت علامت تغییر نمی کند.برای این شیفت سمت راست ترین بیت(کم ارزش ترین) را صفر میکنیم.برای این کار بیت موجود در ثبات **E** را صفر میکنیم.و سپس یک چرخش به چپ انجام می دهیم.که در اثر آن بیت علامت به **E** منتقل می شود.حال باید وجود سرریز را بررسی کنیم.که در صورت تغییربیت علامت می گوییم سرریز رخ داده است.

CLE

CIL

(استفاده از زیرروال در اسمبلی)

1010
0100

```
Loc.                ORG  100      / Main program
100                 LDA  X        / Load X
101                 BSA  SH4      / Branch to subroutine
102                 STA  X        / Store shifted number
103                 LDA  Y        / Load Y
104                 BSA  SH4      / Branch to subroutine again
105                 STA  Y        / Store shifted number
106                 HLT
107       X,        HEX  1234
108       Y,        HEX  4321
                                  / Subroutine to shift left 4 times
109       SH4,      HEX  0        / Store return address here
10A                 CIL           / Circulate left once
10B                 CIL
10C                 CIL
10D                 CIL           / Circulate left fourth time
10E                 AND  MSK      / Set AC(13-16) to zero
10F                 BUN  SH4  I   / Return to main program
110       MSK,      HEX  FFF0     / Mask operand
                    END
```

مثال (۲):برنامه مقابل چه کاری انجام می دهد؟

```
        ORG  100
        CLE
        CLA
        STA CTR
        LDA WRD
        SZA
        BUN ROT
        BUN STP
ROT,    CIL
        SZE
        BUN AGN
        BUN ROT
AGN,    CLE
        ISZ CTR
        SZA
        BUN ROT
STP,    HLT
CTR,    HEX 0
WRD,    HEX 62C1
        END
```

CTR

2

⋮

6

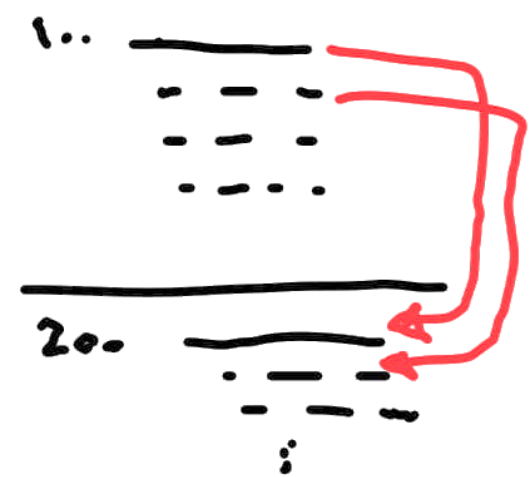
E

0

0 0010 1100 0001000

این برنامه تعداد یک های محتوای خانه حافظه WRD را محاسبه می کند

(برنامه نویسی I/O-زیرروال انتقال بلاک داده)




```
        /Main program
        BSA  MVE      / Branch to subroutine
        HEX  100      / 1st address of source data
        HEX  200      / 1st address of destination data
        DEC  -16      / Number of items to move
        HLT
MVE,    HEX  0        / Subroutine MVE
        LDA  MVE  I   / Bring address of source
        STA  PT1      / Store in 1st pointer
        ISZ  MVE      / Increment return address
        LDA  MVE  I   / Bring address of destination
        STA  PT2      / Store in 2nd pointer
        ISZ  MVE      / Increment return address
        LDA  MVE  I   / Bring number of items
        STA  CTR      / Store in counter
        ISZ  MVE      / Increment return address
LOP,    LDA  PT1  I   / Load source item
        STA  PT2  I   / Store in destination
        ISZ  PT1      / Increment source pointer
        ISZ  PT2      / Increment destination pointer
        ISZ  CTR      / Increment counter
        BUN  LOP      / Repeat 16 times
        BUN  MVE  I   / Return to main program
PT1,    --
PT2,    --
CTR,    --
```

مثال (۱):برنامه ای بنویسید که دو کاراکتر از مکان WRD را بردارد.و آنها را در بیت های 0تا 7 از مکان های CH1 وCH2 ذخیره کند،بیت های 8تا15 باید صفر باشند.

```
ORG 100
LDA WRD
AND LSB
STA CH1
LDA WRD
AND MSB
CLE
BSA Rotate
STA CH2
HLT
Rotate : Hex 0
         CIR
         CIR
         CIR
         CIR
         CIR
         .
         .
         .
         CIR
         BUN Rotate I
WRD, HEX A23F
CH1, HEX 00
CH2, HEX 00
LSB , HEX 00FF
MSB , HEX FF00
END
```