۱– دستورات محاسبات ریاضی

الف) دستور جمع ADD

با اجرای این دستور محتوای عملوند مبداء و مقصد با یکدیگر جمع و نتیجه در عملوند مقصد قرار می گیرد.

الگو:

ADD مبداء , مقصد

نکته:

–دستور جمع بر همه بیتهای حسابی ثبات پرچم اثر دارد.

مثال : در دو قطعه برنامه زیر نتیجه متغیر X را مشخص کنید:

```
X DB 18
MOV AL , -18
ADD X , AL
```

X=0

```
X DB 13
MOV AL , 10
ADD X , AL
```

X=23

تمرین: با اجرای دستورات زیر نتیجه را بدست آورده و سپس وضعیت فلگهای CF,AF,SF,ZF,PF را اعلان نمایید.

```
MOV  CL,F5H
ADD  CL,0BH
```

ب)دستور جمع با رقم نقلی ADC:

با اجرای این دستور محتوای عملوند مبداء با مقصد با رقم نقلی (CF) با یکدیگر جمع و درعملوند مقصد قرار می گیرد.

الگو:

ADC مبداء , مقصد مبداء + مقصد + رقم نقلی ← مقصد

مثال : نتیجه ثبات AX پس از اجرای قطعه برنامه زیر چقدر است ؟

```
X DW ?
MOV AX , 1000
MOV X , 3000
ADC AX , X
```

جواب : در خط چهارم محتوای ثبات AX=۱۰۰۰ و متغیر X=۳۰۰۰ با CF با همدیگر جمع شده و نتیجه در ثبات AXذخیره می گردد.

```
If CF=0 AX=4000
If CF=1 AX=4001
```

ج)دستور تفریق SUB :

با اجرای این دستور محتوای عملوند مقصد از مبداء کم شده و نتیجه در عملوند مقصد قرار می گیرد.

الگو:

SUB مبداء , مقصد

مثال : محتوای ثبات AL و BL پس از اجرای قطعه برنامه زیر چقدر است ؟

```
MOV AL , 10
MOV BL , 6
SUB AL , BL
```

جواب:

AL=4
BL=6

تمرین : در قطعه برنامه زیر محتوای ثبات AX و متغیر X را پس از اجرای دستورات زیر نشان دهید.

```
X DW 600
MOV AX , 248
SUB AX , 48
SUB X , AX
```

تمرین : در برنامه زیر محتوای ثبات AL و پرچم SF را پس از اجرای دستورات زیر نشان دهید.

```
ALI DB 26 , 126 , 64 , 13 , 40 , 60
MOV SI , 4
MOV AL , 20
SUB AL , ALI[SI]
```

د) دستور تفریق با رقم نقلیSBB :

با اجرای این دستور محتوای عملوند مقصد از مبداء و از رقم نقلی CF کم شده و نتیجه درعملوند مقصد قرار می گیرد.

الگو:

مقصد– (مبداء + رقم نقلی) ⬅مقصد **SBB مبداء , مقصد**

مثال : محتوای ثبات AX را یکبار با CF=0 و بار دیگر با CF=1 بنویسید.

```
MOV AX , 1000
SBB AX , 800
```

جواب:

If CF=0 AX=200
If CF=1 AX=199

ر) دستور ضرب :

دستور ضرب به دو صورت عملوندهای بدون علامت و عملوندهای علامتدار می باشد.

MUL → بدون علامت

IMUL → علامت دار

تفاوت این دو دستور در فلگهایی است که استفاده می کنند(در mul فلگهای cf,zf و در دستور imul فلگهای sf,of)

نکته :

-در دستور ضرب عملوند می تواند از نوع بایت یا کلمه باشد.

-عملوند می تواند متغیر یا ثبات باشد.

-عملوند نمی تواند یک عدد ثابت باشد.

عملیات ضرب دارای ۴ حالت مختلف به شرح زیر است:

الف) ضرب بایت در بایت:

الگو :

MUL بایت عملوند

بایت عملوند * AL ← AX

همانگونه که دیده می شود دستورالعمل ضرب با یک عملوند نوشته می شود . برای انجام ضرب دو عدد هشت بیتی لازم است که یکی از ارقام از قبل در ثبات AL و دیگری را در دستور بعنوان عملوند بایت قرار دهیم . نتیجه این ضرب در ثبات AX ذخیره می شود.

مثال: ۲ عدد a,b را در هم ضرب کرده و نتیجه را در result ذخیره نمایید.

```
Datasg  segment  'data'
A   db   2ch
B   db   34h
Result   dw   '?'
Datasg  ends
Codsg   segment   'code'
Start   proc   far
Mov   al,b
Mul   a
Mov   result ,ax
Start   endp
Codsg   ends
End   start
```

ب) ضرب بایت در کلمه:

الگو :

کلمه عملوند MUL

کلمه عملوند * AX ← DX : AX

همانگونه که دیده می شود ضرب بایت در کلمه می تواند جوابی بیش از شانزده بیت داشته باشد در نتیجه قسمت کم ارزش تردر ثبات AX و قسمت پرارزش تر در ثبات DX ذخیره می گردد .

مثال: ۲ عدد a,b را در هم ضرب کرده و نتیجه را در result ذخیره نمایید.

```
Datasg  segment  'data'
A   dw   2ab7h
B   db   3ch
Result   dw   2dup('?')  →  آرایه ۲ عضوی و هر عضو ۲ بایت
Datasg  ends
Codsg  segment  'code'
Start   proc   far
Mov   al,b
Sub   dx,dx   →  خالی کردن ثبات به خاطر داشتن مقدار های قبلی
Mov   cx,a
Mul   cx
Mov   result ,ax
Mov   result+2,dx   →چون دو بایتی است پس باید +۲  شود تا به محل شروع ثبات بعدی برسد
Start   endp
Codsg   ends
End   start
```

ج) ضرب کلمه در کلمه:

الگو :

کلمه عملوند MUL

کلمه عملوند * AX ← DX : AX

همانگونه که دیده می شود ضرب کلمه در کلمه می تواند جوابی بیش از شانزده بیت داشته باشد در نتیجه قسمت کم ارزش تردر ثبات AX و قسمت پرارزش تر در ثبات DX ذخیره می گردد .

مثال: ۲ عدد a,b را در هم ضرب کرده و نتیجه را در result ذخیره نمایید.

```
Datasg  segment  'data'
A   dw   2ab7h
B   dw   3c54h
```

```
Result    dw    2dup('?')  →  آرایه ۲ عضوی و هر عضو ۲ بایت
Datasg   ends
Codsg    segment   'code'
Start    proc    far
Mov    ax,b
Mov    cx,a
Mul    cx
Mov    result ,ax
Mov    result+2,dx   →چون دو بایتی است پس باید +۲  شود تا به محل شروع ثبات بعدی برسد
Start    endp
Codsg    ends
End    start
```

د) ضرب کلمه مضاعف در کلمه مضاعف:

الگو :

MUL کلمه مضاعف عملوند

EDX : EAX⬅ EAX * کلمه عملوند

ضرب کلمه مضاعف در کلمه مضاعف می تواند جوابی بیش از ۳۲ بیت داشته باشد در نتیجه قسمت کم ارزش تردر ثبات EAX و قسمت پرارزش تر در ثبات DX Eذخیره می گردد .

مثال: ۲ عدد a,b را در هم ضرب کرده و نتیجه را در result ذخیره نمایید.

```
Datasg   segment   'data'
A    DD    32AB7H
B    DD    43C54H
Result    DD    2dup('?')  →  آرایه ۲ عضوی و هر عضو ۲ بایت
Datasg   ends
Codsg    segment   'code'
Start    proc    far
Mov    eax,b
Mov    ecx,a
Mul    ecx
Mov    result ,eax
Mov    result+4,edx   →چون حافظه ها ۳۲ بیتی است پس باید +۴  شود تا به محل شروع ثبات بعدی برسد
Start    endp
Codsg    ends
End    start
```

س)دستور تقسیم :

دستور تقسیم نیز مانند دستور ضرب به دو صورت عملوندهای بدون علامت و عملوندهای علامتدار می باشد.

DIV علامت بدون

IDIV علامتدار

-در دستور تقسیم عملوند می تواند از نوع بایت یا کلمه باشد.

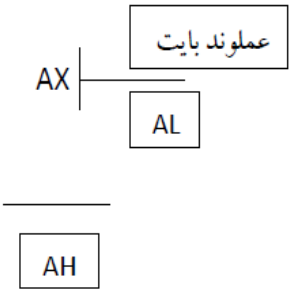
-عملوند می تواند متغیر یا ثبات باشد.

-عملوند نمی تواند یک عدد ثابت باشد.

عملیات تقسیم دارای حالتهای مختلف به شرح زیر است:

❖ الف) چنانچه عملوند بصور بایت باشد :

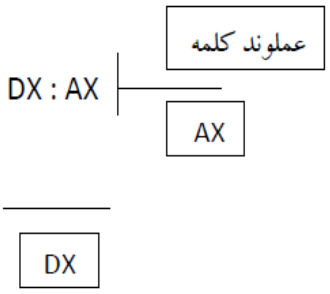
DIV عملوند بایت



نتیجه خارج قسمت و باقیمانده در ثبات AX ذخیره می شود .

❖ ب) چنانچه عملوند بصورت کلمه باشد :

DIV عملوند کلمه



نتیجه خارج قسمت و باقیمانده در ثباتهای DX:AX ذخیره می شود.

مثال: برنامه ای بنویسید عدد بایتی A را بر عدد بایتی Bتقسیم نموده و نتیجه خارج قسمت را درQ و باقیمانده را در Rقرار دهد.

```
Datasg  segment  'data'
A   db   3ch
B   db   2Ah
R   db   '?'
Q   db   '?'
Datasg  ends
Codsg   segment  'code'
Start   proc   far
Mov   al,a
Sub   ah,ah    → ah صفر کردن ثبات
Div   b
Mov   R ,ah
Mov   Q ,al
Start   endp
Codsg   ends
End   start
```

مثال: برنامه ای بنویسید عدد کلمه A را بر عدد کلمه Bتقسیم نموده و نتیجه خارج قسمت را درQ و باقیمانده را در R قرار دهد.

```
Datasg  segment  'data'
A   dw   2abch
B   dw   35efh
R   dw   '?'
Q   dw   '?'
Datasg  ends
Codsg   segment  'code'
Start   proc   far
Mov   ax,a
Sub   dx,dx    → dx صفر کردن ثبات
Div   b
Mov   R ,dx
Mov   Q ,ax
Start   endp
Codsg   ends
End   start
```

ش) دستور کاهش DEC :

با اجرای این دستور یک واحد از محتوای عملوند کم شده و نتیجه در عملوند قرار می گیرد.

الگو:

عملوند DEC

۱- عملوند ← عملوند

این دستور بر روی بیت های پرچم of,sf,zf,af,pf اثر میگذارد ولی cf تغییر نمی کند.

ف)دستور افزایش INC :

با اجرای این دستور یک واحد به محتوای عملوند اضافه شده و نتیجه در عملوند قرار می گیرد.

الگو:

عملوند INC

1+عملوند ← عملوند

این دستور بر روی بیت های پرچم of,sf,zf,af,pf اثر میگذارد ولی cf تغییر نمی کند.

ق) دستور تغییر علامت NEG :

این دستور مکمل ۲ یک عملوند را بدست می آورد. بدیهی است مکمل ۲ نمودن یک عدد یعنی تغییر علامت دادن آن(یعنی عدد مثبت را منفی و منفی را مثبت می کند)

الگو:

عملوند NEG

مثال: مقدار ثبات AL در برنامه زیر چیست؟

```
MOV   AL,35H
NEG   AL
```