دستورات اسمبلی ـ دستورات انتقال داده

دستورالعمل های انتقال داده مقادیر را از یک محل به محل دیگر کپی می کنند .

MOV
XCHG
LEA

MOV

ساده ترین دستورالعمل mov است که دارای دو عملوند است. این دستورالعمل محتوای دومین عملوند خود را در اولین کپی می کند. فرم کلی آن به صورت زیر است :

mov Dest, Source

دستور mov یک کپی از Source را گرفته و آنرا در Dest ذخیره می کند .محتوای Source بعد از اجرای دستور تغییر نمی کند ولی مقدار قبلی Dest رونویسی می شود .

دستور mov مشابه دستور انتساب در زبان های سطح بالا است (Dest := Source; در زبان Pascal یا Dest=Source; در زبان C).

با توجه به نوع عملوندها، انواع مختلفی از دستورالعمل mov را می توان داشت. متداولترین آنها عبارتند از :

mov register, register
mov memory, register
mov register, memory
mov memory, immediate data
mov register, immediate data
mov AX/AL, memory
mov memory, AX/AL
mov segment register, memory 16
mov segment register, register 16
mov register 16, segment register
mov memory 16, segment register

چند موضوع مهم درباره دستور mov را باید همواره بخاطر داشت :

۱. انتقال حافظه به حافظه وجود ندارد. یعنی هردو عملوند همزمان نمی توانند عملوند حافظه ای باشند.
۲. عملوندها می تواند از نوع بایت یا کلمه باشند. اما هردو عملوند حتما باید هم اندازه باشند (برای مثال دستور mov AX,BL اشتباه است). این برای عملوند های حافظه و ثابت هم باید رعایت شود (اگر متغیری را یک بایتی تعریف کنید و آنرا در ثبات AX منتقل کنید اسمبلر پیغام خطا صادر می کند).
۳. با این دستور نمی توان یک داده فوری را در یک ثبات سگمنت منتقل کرد.
۴. هر دو عملوند نمی توانند ثبات سگمنت باشند.
۵. گونه هائی از دستور mov سریع تر و کوتاهتراز بقیه هستند. برای مثال هر دو دستور mov ax, mem و mov reg, mem داده ای را از حافظه به ثبات کپی می کنند اما دستورالعمل اول کوتاهتر و سریع تر از دومی است.
۶. می توان یک مقدار فوری را در یک محل حافظه منتقل کرد. در این حالت داده فوری به اندازه عملوند مقصد گسترش داده می شود (مگراینکه بزرگتر از مقصد باشد که خطا صادر می شود). البته اسمبلر نمی تواند اندازه عملوند حافظه را تعیین کند مگر اینکه عملوند حافظه ای به صورت یک متغیر در برنامه اعلان شده باشد. برای حل این مشکل از عملگر های byte ptr و word ptr برای تعیین اندازه عملوند حافظه ای می توان استفاده کرد.

مثال. دستور زیرداده فوری 10h را به اندازه یک کلمه گسترش داده و در محلی که BX به آن اشاره می کند ذخیره می کند .
mov word ptr [bx], 10h

مثال. دستورات زیر داده فوری 40h را در ثبات سگمنت ES ذخیره می کند .ثبات AX به عنوان واسطه بکار رفته است. هر کدام از ثبات همه منظوره را می توان به جای AX بکار برد .

```
mov AX, 40h
mov ES, AX
```

دستور mov روی هیچکدام از فلگ ها تاثیری ندارد.

XCHG

دستورالعمل xchg محتوای دو عملوند خود را جابجا می کند. فرم کلی آن به صورت زیر است :

```
xchg Operand1, Operand2
```

مقدار هردو عملوند در اثر اجرا تغییر می کند .

چهار شکل خاص برای این دستور وجوددارد:

```
xchg register, memory
xchg register, register
xchg ax, register16
```

ترتیب علموندها اهمیت ندارد. می توانید xchg mem,reg یا xchg reg,mem را بنویسید نتیجه فرقی ندارد. اکثر اسمبلرها بطور خودکار کد کوتاهتر را انتخاب می کنند.

هردو عملوند باید یک اندازه باشند .

دستور xchg روی هیچیک از فلگ ها تاثیر نمی گذارد.

LEA

دستورالعمل lea (load effective address) برای مقداردهی اشاره گرها استفاده می شود. فرم خاص آن به صورت زیر است :

```
lea register16, memory
```

این دستور آدرس موثر یک محل خاص از حافظه را درون یک ثبات همه منظوره ذخیره می کند. منظور از آدرس موثر آدرس نهائی حافظه بعد از کلیه محاسبات آدرسی است .

مثل .دستور زیر مقدار 1234h را در ثبات AX قرار می دهد .

```
lea   AX, DS:[1234h]
```

دستور mov ax, immediate data هم همین عمل را انجام می دهد. تفاوت آنها در این است که دستورالعمل lea محاسبه آدرسی و انتقال داده را همزمان انجام می دهد .

مثال. دستور زیر آدرس حاصل از محاسبه BP+SI+4 را درثبات AX قرار می دهد. ابتدا مقادیر را باهم جمع کرده سپس در ثبات BX منتقل می کند .

```
lea   BX, 4[bp+si]
```

دستورالعمل lea روی فلگ ها تاثیر ندارد.