فصل یازدهم

دستورات کنترل برنامه و دستورات منطقی

مقدمه

تا به حال برنامه‌های نوشته شده، دستورات را پشت سر هم اجرا می‌نمودند ولی بعضی از اوقات نیاز است که اجرای دستورات از محل دیگری در برنامه ادامه یابد که در این موقع می‌توان از دستورات کنترل برنامه شامل دستورات پرش استفاده نمود. علاوه بر این اکثر برنامه‌ها شامل تعدادی عملیات مقایسه اطلاعات هستند که متناسب با نتیجه مقایسه می‌بایستی به محل بخصوصی از برنامه مراجعه شود و دستورات مورد نظر اجرا گردند. در این مواقع نیز از دستورات پرش شرطی استفاده می‌گردد. بسیاری از مواقع نیر نیاز داریم قسمتی از برنامه چندین بار پشت سر هم اجرا گردد. در این مواقع از دستور حلقه بهره می‌گیریم. علاوه بر این با دستورات منطقی AND، OR، XOR، TEST، NOT، شیفت، چرخش و غیره می‌توان یک یا تعدادی بیت را یک یا صفر نمود و یا بیت بخصوصی را تست کرد.

دستورات کنترل برنامه

دستورات کنترل برنامه عبارتند از دستورات پرشJMP و دستورات LOOP که هر یک را ذیلاً مورد بررسی قرار می‌دهیم.

دستور پرش JMP

دستور پرش بصورت زیر تعریف می‌شود:

JMP برچسب یا آدرس دستور هدف

این دستور برای کنترل برنامه استفاده می‌شود، یعنی کنترل از نقطه‌ای از برنامه به نقطه دلخواه دیگر از برنامه منتقل می‌شود (این دستور بر بیت‌های پرچم اثر ندارند). در حقیقت عملوند یا اپراند دستورJMP، برچسب یا آدرس دستور دیگری می‌باشد که برنامه از آن دستور می‌بایستی شروع و ادامه یابد.

به عنوان مثال در دستورات:

```
JMP BEGIN
------
BEGIN: ADD AL, BL
-------
```

دستور LOOP

دستورLOOP برای کنترل برنامه و تکرار دستورات به تعداد معین بکار می‌رود. برای این کار می توان از دستورLOOP استفاده نمود، به شرطی که تعداد تکرار حلقه را، در ثبات CX قرار دهیم. در این صورت هر بار که حلقه اجرا می‌شود، دستورLOOP به طور خودکار از ثبات CX یک واحد کم می‌کند، به محض اینکه CX صفر شد، اجرای حلقه پایان می‌یابد و دستور بعد از LOOP اجرا می‌شود. شکل کلی این دستور به صورت زیر است:

Mov Cx , مقدار

آدرس هدف یا ابتدای حلقه LOOP

که آدرس جلوی دستور یک بایتی است و حداکثر بین 127+ و 128- می‌باشد. در ضمن این دستور بر روی بیت‌های پرچم تاثیر نمی‌گذارد.

دستور مقایسه CMP

این دستور محتویات دو عملوند را با هم مقایسه می‌کند و روی بیت‌های پرچم اثر می‌گذارد. شکل کلی آن بصورت زیر است:

CMP Operand1,Operand2

CMP 1عملوند، عملوند2

اجرای این دستور باعث می‌شود که عملوند2 از عملوند1 کسر گردد و بر روی بیت‌های پرچم OF، SF، ZF، AF، DF و CF اثر بگذارد که از روی وضعیت بیت‌های پرچم، پردازنده تشخیص می‌دهد که آیا دو عملوند مساوی یا عملوند1 بزرگتر از عملوند2 می‌باشد و بالعکس هستند.

طبق شکل زیر داریم:

CF	ZF	
0	0	عملوند - 1>عملوند - 2
0	1	عملوند - 1=عملوند - 2
1	0	عملوند - 1< عملوند - 2

عملوند1 می‌تواند یک ثبات یا یک خانه حافظه باشد و عملوند2 نیز می‌تواند هر یک از ثبات‌ها، یک خانه حافظه و یا یک عدد ثابت باشد.

ایجاد حلقه تاخیر با دستور LOOP

با دستورLOOP می‌توان یک تاخیر زمانی نرم‌افزاری ایجاد کرد. برای این کار دستورات زیر را می‌نویسیم

```
MOV CX, N
AGAIN: LOOP AGAIN
```

دستورات پرش شرطی

کامپیوترها دارای تعداد زیادی دستورات پرش شرطی برای کاربردهای متفاوت می‌باشند. اصولاً دستورات پرش شرطی بعد از دستوراتی که بر روی بیت‌های پرچم اثر می‌گذارند، قرار داده می‌شوند. معمول‌ترین این دستورات دستور مقایسه CMP و دستورات منطقی یا محاسباتی می‌باشند. انواع دستورات پرش شرطی مبتنی بر بیت‌های پرچم در سه جدول زیر اشاره شده‌اند.

بیت‌های پرچمی که بررسی می‌شود	توضیح	دستور پرش
بیت علامت SF	Jump Sign (Negative)	JS Tarjet اگر نتیجه محاسبات منفی (SF = 1) است عمل پرش انجام شود.
بیت علامت SF	Jump Not Sign (Positive)	JNS Tarjet اگر نتیجه محاسبات مثبت (SF = 0) است عمل پرش انجام شود.
بیت نقلی CF	Jump Carry	JC Tarjet اگر بیت نقلی CF برابر یک است پرش انجام شود.
بیت نقلی CF	Jump Not Carry	JNC Tarjet اگر بیت نقلی CF برابر صفر است پرش انجام شود.
بیت سرریز OF	Jump Overflow	JO Tarjet اگر بیت سرریز OF یک است عمل پرش انجام شود
بیت سرریز OF	Jump Not Overflow	JNO Tarjet اگر بیت سرریز OF صفر است عمل پرش انجام شود.
بیت توازن PF	Jump Parity یا Jump Parity Even	JP/JPE Tarjet* اگر بیت توازن P زوج** است عمل پرش انجام شود.
بیت توازن PF	Jump Not Parity یا Jump Parity Odd	JNP/JPO Tarjet اگر بیت توازن P فرد** است. عمل پرش انجام شود.

مقدار بیت‌های پرچم	توضیح	دستور
ZF = 1	(Jump Equal/or Jump Zero)	JE/JZ Tarjet اگر اُپراندها* مساوی است، یا بیت تشخیص صفر ZF = 1 است عمل پرش انجام شود.
ZF = 0	(Jump Not Equal/ or Jump Not Zero)	JNE/JNZ Tarjet اگر اُپراندها* مساوی نیستند، یا اگر بیت تشخیص صفر ZF = 0 است عمل پرش انجام شود.
CF = 0 , ZF = 0	(Jump Above or Jump Not Below or Equal)	JA/JNBE Tarjet اگر اُپراند اوّل بالاتر است (اگر بزرگتر است) یا، اگر کمتر، یا مساوی نیست عمل پرش انجام شود.
CF = 0	(Jump Above or Equal/or Jump Not Below)	JAE/JNB Tarjet اگر اپراند* اول بالاتر (اگر بزرگتر است) یا مساوی است، یا اگر کمتر نیست عمل پرش انجام شود.
CF = 1	(Jump Below/or Jump Not Above or Equal)	JB/JNAE Tarjet اگر اُپراند* اوّل کمتر است، یا اگر بالاتر یا مساوی نیست، عمل پرش انجام شود.
CF = 1 یا ZF = 1	(Jump Below or Equal/or Jump Not Above	JBE/JNA Tarjet اگر اُپراند* اوّل کمتر یا مساوی است، یا اگر بالاتر نیست، عمل پرش انجام شود.

دستور	توضیح	مقدار بیت‌های پرچم
JE/JZ Tarjet اگر دو عدد مساوی است، یا نتیجه محاسبات صفر است، عمل پرش انجام شود.	(Jump Equal/or Jump If Zero)	ZF = 1
JNE/JNZ Tarjet اگر دو عدد مساوی نیستند، یا نتیجه محاسبات صفر نیست، عمل پرشی انجام شود.	(Jump Not Equal/or Jump If Not Zero)	ZF = 0
JG/JNLE Tarjet اگر اُپراند اوّل* بزرگتر است، یا کوچکتر یا مساوی نیست، عمل پرش انجام شود.	(Jump Greater/or Jump If Not Less or Equal	SF ⊕ OF = 0 ZF = 0
JGE/JNL Tarjet اگر اُپراند* اول بزرگتر یا مساوی است، یا کوچکتر نیست، عمل پرش انجام شود.	Jump Greater or Equal/or Jump Not Less	SF ⊕ OF = 0
JL/JNGE Tarjet اگر اُپراند* اوّل کوچکتر است یا بزرگتر و مساوی نیست عمل پرش انجام شود.	(Jump Less/or Jump Not Greater or Equal)	SF ⊕ OF = 1
JLE/JNG Tarjet اگر اُپراند اوّل* کوچکتر یا مساوی است و یا بزرگتر	(Jump Less or Equal /or Jump Not Greater)	SF ⊕ OF = 1 ZF = 1

دستورات منطقی

این دستورات عبارتند از AND، OR، XOR، TEST که توسط این دستورات عملیات منطقی بر روی هر یک از بیت‌های ثبات‌ها، یا خانه‌های حافظه انجام می‌شود. با این عملیات می‌توان بر روی یک بیت یا گروهی از بیت‌های ثبات‌ها یا خانه‌های حافظه عملیاتی انجام داد و آنها را متناسب با کاربرد، یک یا صفر نمود. به این دلیل در بعضی از کامپیوترها، دستورات منطقی را دستورات عملیات روی بیت نیز می‌نامند. شکل کلی این دستور به صورت زیر است:

Operation Destination, Source

که کد اپراند مقصد می‌تواند هر یک از ثبات‌ها یا خانه‌های حافظه باشد و منبع نیز می‌تواند هر یک از ثبات‌ها، خانه‌های حافظه یا یک عدد ثابت باشد. این عملیات می‌توانند روی اطلاعات 8بیتی، 16بیتی و یا 32بیتی اجرا شوند و بر روی بیت‌های پرچم نقلیCF، سرریزOF، توازنPF و تشخیص صفرZF اثر می‌گذارند.

الف) دستور AND

در دستور AND اگر هر دو بیت نظیر به نظیر دو اپراند یک باشند، بیت نظیر نتیجه نیز یک می‌باشد ولی اگر هر یک از دو بیت دو اپراند صفر باشند بیت نظیر نتیجه، صفر خواهد بود. به عنوان مثال داریم:

0101= اپراند1 یا اطلاعات 1

0011= اپراند2 یا اطلاعات 2

0001= نتیجه AND دو اپراند یا دو اطلاعات

ب) دستور OR

در دستور OR اگر هر دو بیت نظیر به نظیر اطلاعات صفر باشند، بیت نظیر نتیجه صفر می‌شود در غیر این‌صورت بیت نظیر نتیجه یک خواهد بود. به عنوان مثال داریم:

0101= عملوند 1 یا اطلاعات 1

0011= عملوند 2 یا اطلاعات 2

0111= نتیجه OR دو عملوند یا دو اطلاعات

ج) دستور XOR

در دستور XOR اگر فقط یکی از دو بیت اطلاعات یک باشد، خروجی یک است ولی اگر هر دو بیت اطلاعات یک یا هر دو بیت صفر باشند، خروجی صفر خواهد بود. به عنوان مثال داریم:

0101= عملوند 1 یا اطلاعات 1

0011= عملوند 2 یا اطلاعات 2

0110= نتیجه XOR دو عملوند یا دو اطلاعات

د) دستور NOT

این دستور باعث می‌شود عملوند یا محتوای ثبات یا خانه حافظه، بیت به بیت معکوس می‌گردد، به عبارت دیگر مکمل یک آن بدست می‌آید.

ح) دستور TEST

این دستور دارای دو عملوند است که باعث می‌شود این دو عملوند بیت به بیت باهم AND شوند و روی بیت‌های پرچم اثر بگذارند ولی محتویات هیچ‌کدام از عملوندها عوض نمی‌شوند. شکل کلی این دستور به صورت زیر است:

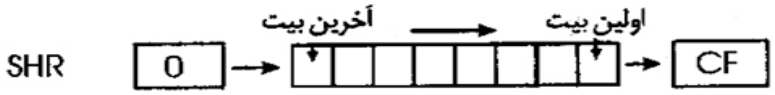
TEST 1 عملوند، 2 عملوند

دستورات شیفت و چرخش

این دستورات عبارتند از SHL، SAL، SHR، SAR، ROR، RCR، ROL و RCL که در ذیل هر یک از آنها بررسی می‌گردد. توسط دستور شیفت، محتوای ثبات‌های پروسسور یا خانه حافظه، یک یا چند بیت به طرف راست، یا چپ شیفت داده می‌شود و دو نوع شیفت ریاضی و منطقی وجود دارند.

شیفت به چپ SHL و SAL

در عملیات شیفت به چپ، صفر وارد اولین بیت ثبات خانه حافظه می‌شود و هر بیت به خانه‌ی سمت چپ خود منتقل می-گردد، این نوع شیفت برای شیفت ریاضی و منطقی یکسان است و این دستور بر بیت‌های OF، SF، ZF، PF و CF اثر می-گذارد.



شکل (۱۱-۲۱) نمایش دستور شیفت منطقی به راست

شکل کلی این دستور برای شیفت منطقی به صورت ذیل است:

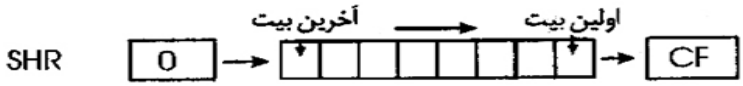
SHR	۱ ، عملوند
	و
SHR	CL ، عملوند

شکل کلی شیفت ریاضی به چپ به صورت ذیل است:

SAR	۱ ، عملوند
	و
SAR	CL ، عملوند

شیفت به راست SHR و SAR

شیفت به راست دو نوع می‌باشد. شیفت منطقی برای اعداد بدون علامت و دیگری شیفت ریاضی جهت اعداد علامت‌دار. این دستورات بر بیت‌های پرچم OF، SF، ZF، PF و CF اثر می‌گذارند. در شیفت منطقی به راست (SHR) محتویات ثبات یا خانه حافظه یک بیت یا به اندازه عددی که داخل ثبات CL است به راست شیفت می‌کند یعنی هر بیت، به خانه‌ی سمت راست خود منتقل می‌شود.

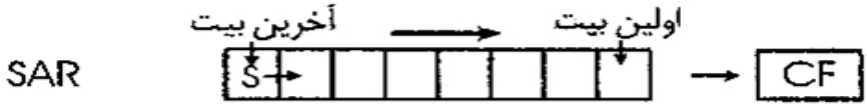


در این دستور، یک بار و یا به اندازه محتوای ثبات CL، به آخرین بیت صفر وارد می‌شود و اولین بیت نیز وارد بیت پرچم CF می‌گردد

شکل کلی این دستور به صورت زیر است:

SHR	۱ ، عملوند
	و
SHR	CL ، عملوند

شیفت ریاضی به راست SAR، فرق مهمی که با شیفت منطقی به راست دارد در این است که بیت علامت تغییر نمی‌کند و از چپ وارد بیت‌های دیگر می‌شود. در این صورت اعداد مثبت و منفی، علامت خود را حفظ می‌کنند. در شیفت ریاضی به راست SAR، محتویات ثبات یا خانه‌ی حافظه یک بیت، یا به اندازه عددی که داخل ثبات CL است، به راست شیفت می‌کند (یعنی هر بیت به خانه‌ی سمت راست خود منتقل می‌شود.) ولی علامت آن عوض نمی‌شود.



شکل کلی این دستور به صورت زیر است:

SAR عملوند ، ۱
و
SAR عملوند ، CL

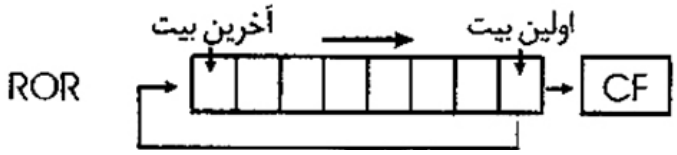
دستورات چرخش

دستورات چرخش شبیه دستورات شیفت هستند، یعنی محتویات یک ثبات یا یک خانه حافظه 8بیتی یا 16بیتی یا 32 بیتی، یک بیت، یا به تعداد عددی که در ثبات CL است، به راست یا به چپ شیفت داده می‌شوند، در دستورات چرخش، اولین بیت وارد آخرین بیت و بالعکس می‌شود. این دستورات بر بیت‌های پرچم‌های CF وOF اثر می‌گذارند

دستور چرخش راست ROR

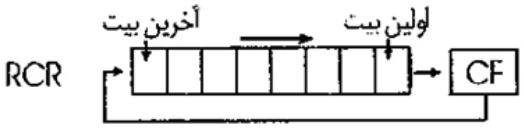
شکل کلی دستور چرخش به راست به صورت زیر است:

ROR عملوند ، 1	(۱)
و	
ROR عملوند ,CL	(۲)



دستور چرخش به راست از طریق بیت پرچم نقلی RCR

دستور چرخش به راست می‌تواند از طریق بیت پرچم نقلی CF نیز انجام پذیرد. یعنی محتوای بیت صفر عملوند (اولین بیت سمت راست) وارد بیت نقلی CF می‌گردد و هر بیت به خانه‌ی سمت راست خود منتقل می‌شود و بیت نقلی CF وارد آخرین بیت، یعنی بزرگترین بیت می‌گردد



دستور چرخش به راست از طریق بیت نقلی CF

شکل کلی این دستور به صورت زیر است:

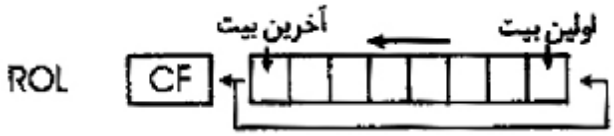
RCR عملوند ، 1	(۱)
و	
RCR عملوند ، CL	(۲)

دستور چرخش به چپ ROL

شکل کلی این دستور به صورت زیر است:

RCL عملوند ، 1	(۱)
و	
RCL عملوند ،CL	(۲)

این دستور بر بیت های پرچم OF، CF، اثر می گذارد. به این ترتیب در دستور (1) عملوند که یک ثبات یا یک خانه حافظه است، یک بیت به طرف چپ شیفت می‌کند یعنی هر بیت به خانه‌ی سمت چپ خود منتقل می‌شود و آخرین بیت وارد اولین بیت و همچنین بیت نقلی CF می‌گردد .



ساختار روال یا سابروتین

برنامه‌ی اسمبلی می‌تواند دارای تعدادی روال یا سابروتین باشد که هر روال با شبه‌دستور PROC شروع و با شبه دستور ENDP پایان می‌یابد و وظیفه‌ی خاصی را انجام می‌دهد. (مانند گرفتن اطلاعات از صفحه کلید و ...) از لحاظ اصول برنامه-نویسی بهتر است برنامه‌ی اسمبلی را به چندین روال کوچکتر تقسیم نمائیم که در این صورت برنامه‌ی اسمبلی دارای مزایای ذیل خواهد بود.

1- تعداد دستورات برنامه حداقل می‌شود چون یک روال می‌تواند از محل‌های مختلف در سگمنت کد فراخوانی شود.

2- تشکیلات برنامه بهتر می‌گردد و آسان‌تر می‌توان محل اشکالات برنامه را پیدا نمود.

3- نگهداری و به روز درآوردن و اصلاحات آتی برنامه نیز راحت می‌شود.

لذا در نوشتن برنامه‌های اسمبلی سعی می‌شود که حتی الامکان از روال یا سابروتین ساده و کوچک استفاده شود.

دستور فراخوانی روال CALL و برگشت از روال RET

برنامه‌ی فرعی یا سابروتین در کامپیوترهای شخصی روال یا Procedure نیز نامیده می‌شود. هر روال دارای یک نام است و ساختار کلی آن به طریق زیر مشخص می‌شود.

```
نام روال[3]   PROC
              —
              —
              —
دستورات       —
              —
              —
              —
              RET
نام روال      ENDP
```

برای فراخوانی یک روال از پیش نوشته شده کافی است نام روال را بعد از کلمه رزرو شده Call بنویسید. در این حالت اجرای برنامه اصلی به صورت موقت قطع شده و اجرای برنامه از ابتدای روال آغاز خواهد شد. بعد از اتمام برنامه روال، با اجرای دستور RET دوباره برنامه از محل قطع برنامه اصلی از سر گرفته می‌شود.