لغة الاسمبلي

4

ألأوامر المنطقية

The Logical Family

AND , OR , NOT , XOR , TEST

الأمر AND

يقوم بتعديل محتوى احد مسجلات الميكروبروسيسور على الصورة التي بالمثال التالي:

AND AX,BX

AX قبل تنفيذ الأمر	0	0	0	0	1	0	1	0	1	1	1	0	0	0	1	1
BX قبل تنفيذ الأمر	1	0	0	1	1	0	0	0	0	0	1	0	0	0	0	1
AX بعد تنفيذ الأمر	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	1

وتتأثر أيضا كل الأعلام الرياضية Arth. flags والقاعدة هي:

1< 1,1	0< 1,0	0< 0,1	0< 0,0

من حالات استخدام هذه الأوامر الكشف عن قيمة بت معين بالذاكرة ولنفرض أن البرنتر عندما لا يكتشف وجود ورق سيقوم بتعديل البت التاسع في كلمة سعة 2 بايت إلى واحد لا تأتي رسالة الخطأ التي تخبرك بعدم وجود ورق من برنامجك ولكن تأتي بعد أن يقوم البرنامج بمساعدة نظام التشغيل بالكشف عن البت المفترض انه تغير إلى 1 وهذا مثال لتطبيق الافتراض السابق وكذلك لكيفية استخدام الأمر AND

سنقوم بتحميل المسجل BX برقم ثنائي كل بتاته صفر ماعدا البت التاسع يكون ب 1

MOV BX , 0000000010000000B

نقوم بتحميل الرقم المطلوب اختباره إلى AX ثم نكتب أمر AND

AND AX , BX

إذا كان البت التاسع المطلوب اختباره في AX = واحد فان الرقم الناتج بعد تنفيذ الأمر سيكون هو نفس الرقم الموجود ب XB ويكون ال Z-FLAG مساويا للصفر لان ناتج العملية ليس صفرا ، أما إذا كان البت التاسع في AX مساويا للصفر فسيكون الناتج بعد تنفيذ الأمر هو 0000000000000000B أي أن العملية صفرية ويكون Z-FLAG مساويا للصفر وعندها يمكنك تنفيذ أي أمر من أوامر القفز السابق الإشارة إليه في الدرس السابق لنقل وإعادة توجيه البرنامج إلى نقطة معينة أو طبع رسالة معينة.

سنكتفي بالمثال السابق حيث ينطبق استخدامه نوعا ما على باقي الأوامر التي سيلي ذكرها فيما بعد لضيق الوقت ولكن سنذكر قواعد استخدام كل أمر.

OR AX,BX

AX قبل تنفيذ الأمر	0	0	1	1	0	0	1	1	0	1	0	1	1	0	1	0
BX قبل تنفيذ الأمر	0	1	0	1	0	1	0	1	0	0	0	0	0	1	1	1
AX بعد تنفيذ الأمر	0	1	1	1	0	1	1	1	0	1	0	1	1	1	1	1

1< 1,0	1< 0,1	1< 1,1	0< 0,0

التشابه يظل كما هو والاختلاف يكون ب 1

NOT BX

BX قبل تنفيذ الأمر	0	1	0	0	1	1	0	1	1	1	0	0	0	1	0	1
BX بعد تنفيذ الأمر	1	0	1	1	0	0	1	0	0	0	1	1	1	0	1	0

1< 0	0< 1

XOR AX,BX

AX قبل تنفيذ الأمر	0	0	1	0	0	0	1	0	0	1	0	0	1	1	1	1
BX قبل تنفيذ الأمر	1	0	1	0	1	1	1	0	0	0	1	1	0	0	0	1
AX بعد تنفيذ الأمر	1	0	0	0	1	1	0	0	0	1	1	1	1	1	1	0

1< 1,0	1< 0,1	0< 1,1	0< 0,0

TEST DL,AH

مشابه تماما للأمر AND ولكنه يؤثر فقط في الأعلام Flags ويقوم بوضع z-flag بواحد إذا كانت قيم البتات ذات الترتيب 2و4و6و8 مساوية للصفر

DL قبل تنفيذ الأمر	1	0	0	0	1	1	0	0
AH قبل تنفيذ الأمر	0	1	0	1	0	1	0	1
DL بعد تنفيذ الأمر	0	0	0	0	0	1	0	0

The Shift Family

SHL , SHR , SAL , SAR

الأمر SHL

SHL REG. or mem. ,1

يقوم بعمل إزاحة لليسار بمقدار واحد بت ثم يترك البت أقصى اليمن بصفر وهذه العملية عبارة عن مضاعفة عدد بدون إشارة أي ضربه في 2

العدد	0	1	0	0	1	1	0	0
SHL	1	0	0	1	1	0	0	0

إذا نتج عن عملية المضاعفة رقم اكبر من سعه حيز التخزين سيتم حمل البت الزائد إلى CARRY-FLAG ، مثال :

MOV AL , 01001010B

SHL AL , 1D

ستصبح محتويات AL بعد تنفيذ الأمر 10010100B ويتم وضع قيمة C-FLAG بـ 1

الأمر SHR

SHR REG. or mem. ,1

يقوم بعمل إزاحة لليمين بمقدار واحد بت ثم يترك البت أقصى اليسار بصفر وهذه العملية عبارة عن مناصفة عدد بدون إشارة أي قسمته على 2

العدد	0	1	1	0	0	1	1	1
SHR	0	0	1	1	0	0	1	1

مثال :

MOV AL , 01100111B

SHR AL , 1D

ستصبح محتويات AL بعد تنفيذ الأمر 00110011B ويتم وضع قيمة C-FLAG بـ 1

الأمر SAL

نفس استخدام وتأثير SHL ولكن بالنسبة للأعداد ذات الإشارة ويتأثر S-FLAG

الأمر SAR

نفس استخدام وتأثير SHR ولكن بالنسبة للأعداد ذات الإشارة ويتأثر S-FLAG

ملاحظات هامة جدا:

SHL AL , 1D (عملية ضرب في 2)

SHL AL , 2D (عملية ضرب في 4)

SHL AL , 3D (عملية ضرب في 8)

SHL AL , 4D (عملية ضرب في 16)

SHL AL , 5D (عملية ضرب في 32)

وهكذا الأمر بالنسبة لـ SHR ولكن العملية تكون قسمة وليست ضرب

The Rotate Family

ROR , ROL , RCL , RCR

كما شرحنا من قبل عند تنفيذ SHL AL , 4D فإن الأربعة بتات اليسرى تفقد نهائيا ولكن عند الرغبة في عدم فقد أي بتات علينا استخدام هذه العائلة الرائعة من الأوامر الدوارة.

الأمر ROR : للدوران إلى اليمين

الأمر ROL : للدوران إلى اليسار

مثال :

```
MOV AL , 01010111B

ROR AL , 1
```

سيصبح AL حاملا للقيمة 10101011B وذلك بعد تنفيذ أمر الدوران

الأمر RCR : للدوران إلى اليمين مع استخدام C-Flag

الأمر RCL : للدوران إلى اليسار مع استخدام C-Flag

العدد	0	1	0	1	0	1	1	1
RCL	1	0	1	0	1	0	1	1

البت الزائد في العدد يرحل إلى C-Flag ثم الذي بال C-Flag يرحل إلى القيمة الجديدة ناحية اليمين وهكذا

مع الأمر RCL تتم نفس العملية السابقة تماما ولكن ناحية اليسار.

هل أدركت الآن لماذا لم أتطرق بالشرح إلى C-Flag عندما كنت اشرح ال Flags في الدرس الثالث ، السبب أني كنت اخطط لشرحه مع أوامر الدوران حيث يمكن أن تتضح بسهولة وظيفته.

يجب أن تفهم الدروس السابقة قبل الانتقال إلى الدروس التالية فهي لغة تعتبر بحق صعبه وبالغة الصعوبة فعليك أن تمشي بخطوات بطيئة جدا.

والآن إلى الدرس الخامس