الكلية التطبيقية

تنظيم الحاسوب والبرمجة بلغة التجميع

-المحاضرة الأولى-

المعالج 8086

يتم ربط المعالج مع مكونات الحاسوب الأخرى (الذاكرة , وحدات الإدخال والإخراج) عن طريق:

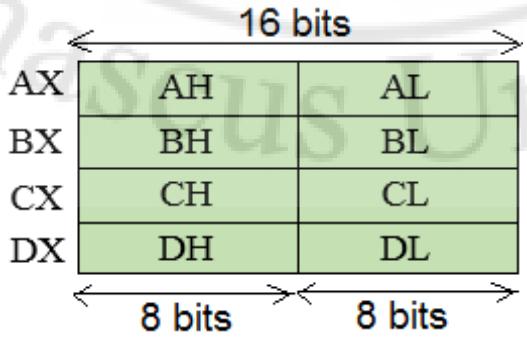
1- **ممر العناوين(Address Bus):** يستخدم لنقل العناوين , وهو ذو سعة 20 bits وبالتالي يمكن عنونة ذاكرة ذات سعة لغاية 1Mbyte (1024 kbyte).
2- **ممر المعطيات (Data Bus):** يستخدم لنقل المعطيات وهو ذو سعة تبلغ 16 bits.
3- **ممر التحكم (Control Bus):** يستخدم لنقل اشارات التحكم بين المعالج ووحدات الحاسوب الأخرى.

يمتلك المعالج 8086 أربعة مجموعات من المسجلات ذات 16 بت وهي:

1- مسجلات المعطيات (مسجلات عامة):

تستخدم هذه المسجلات لتخزين المعطيات وهي أربع مسجلات :

- المسجل AX (المراكم) :يستخدم في التعليمات التي تتعامل مع معاملات ذات 16 bits مثل الضرب والقسمة .
- المسجل BX (BASE): يستخدم غالبا في عنونة المعطيات المخزنة في الذاكرة.
- المسجل CX(العداد): يستخدم كعداد في تعليمات التكرار والمسجل CL يستخدم لتسجيل عدد مرات الإزاحة أو التدوير.
- المسجل DX(DATA): يستخدم في تعليمات الضرب والقسمة على المعطيات التي طولها 16bits.



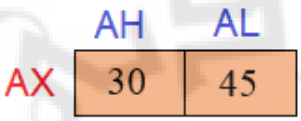
يمكن استخدام مسجلات المعطيات بطريقتين:

- يمكن استخدامها كمسجلات ذات 16 bits وعندها تحمل الاسماء :AX, BX, CX ,DX
- يمكن استخدامها كمسجلات ذات 8 bits وعندها تحمل الأسماء :

AH , AL , BH , BL, CH ,CL , DH , DL

مثال:

بفرض AX=3045H



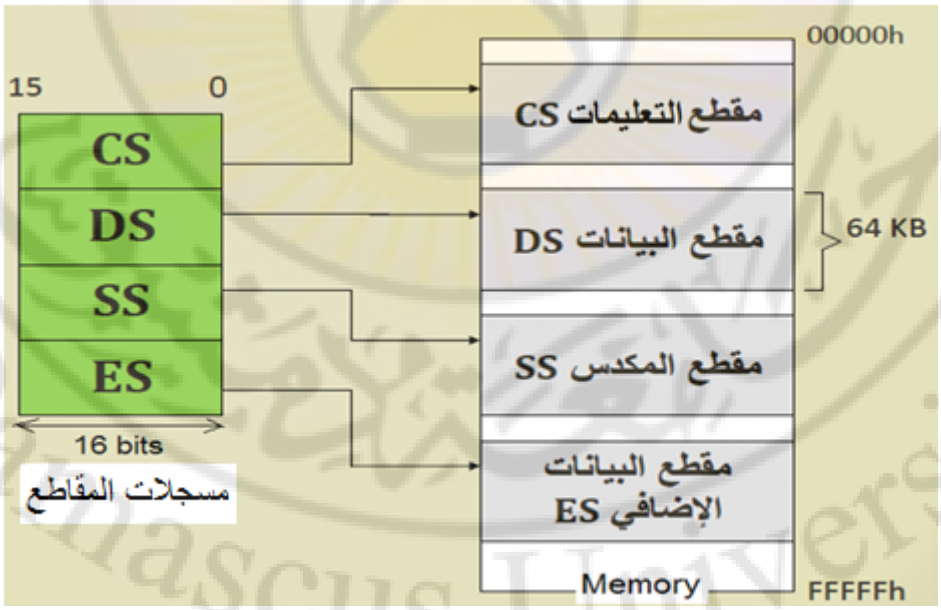
إذا AL=45 و AH=30

2- مسجلات المقاطع :

تخزن معطيات وتعليمات برنامج لغة التجميع في الذاكرة الرئيسية في مقاطع (Segment) منفصلة .

يصل طول المقطع الواحد لغاية 64 كيلوبايت.

يستطيع المعالج 8086 العمل في نفس الوقت مع أربعة مقاطع فقط.



تستخدم مسجلات المقاطع لتخزين عنوان بداية كل مقطع وهي أربع مسجلات:

1- **مسجل مقطع التعليمات CS:** يشير الى بداية مقطع التعليمات بالذاكرة.
2- **مسجل مقطع البيانات DS:** يشير الى بداية مقطع البيانات بالذاكرة.
3- **مسجل مقطع البيانات الإضافي ES:** يشير الى بداية مقطع البيانات الاضافي.
4- **مسجل مقطع المكدس SS:** يشير الى نهاية مقطع المكدس.

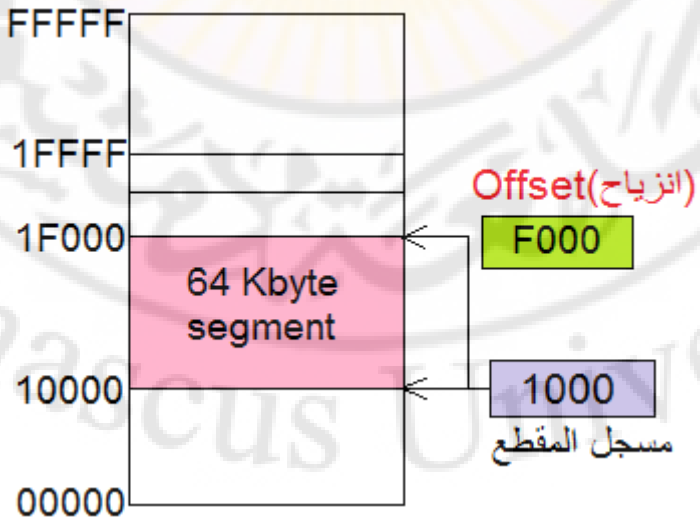
3- مسجلات التأشير والفهرسة:

- مسجل دليل المصدر (SI)
- مسجل دليل الهدف (DI)
- مسجل مؤشر المكدس (SP)
- مسجل مؤشر القاعدة (BP)

مسجل مؤشر التعليمة IP:

هو مسجل طوله 16 bits . يحوي دائما على مقدار ازاحة التعليمة التالية عن بداية مقطع التعليمات أي يشير الى التعليمة التالية بالتنفيذ.

من أجل الوصول الى المعطيات أو التعليمات المخزنة في المقاطع المختلفة في الذاكرة , يقوم المعالج بربط مسجل المقطع المناسب(يحوي عنوان بداية المقطع) مع أحد مسجلات التأشير أو الفهرسة التي تدل على عنوان الإزاحة (Offset Address) ضمن المقطع .



يبين الجدول التالي المسجلات التي تحتوي عنوان الازاحة لكل مقطع:

Offset	(المقطع)Segment
IP	CS
SP,BP	SS
BX,SI,DI	DS
DI	ES

العنوان الفيزيائي :

يتكون من 20 bit يتراوح بين 00000 H و FFFFF H

عبارة عن تركيبة من قيمتين أساسيتين :

- Segment address(عنوان المقطع): يحفظ في مسجلات المقاطع ويدل على عنوان بداية المقطع.
- Offset address (عنوان الإزاحة): يدل على عنوان الازاحة ضمن المقطع.

لايجاد العنوان الفيزيائي يتم اضافة صفر لليمين لقيمة مسجل المقطع ومن ثم جمعه مع قيمة عنوان الازاحة.

مثال :

بفرض IP=4214 H و CS=348A H ونريد ايجاد العنوان المنطقي و العنوان الفيزيائي.

العنوان المنطقي: Segment : Offset

CS:IP

348A:4214

يتم حساب العنوان الفيزيائي عن طريق اضافة صفر من اليمين لقيمة CS ومن ثم جمعه مع قيمة IP.

العنوان الفيزيائي**(PA)**= (CS)**0**+IP

$$\begin{array}{r} 348A0 \\ +\ 4214 \\ \hline 38AB4 \end{array}$$

العنوان الفيزيائي: 38AB4

4- مسجل الأعلام (Flag Register):

مسجل طوله 16 bit لتخزين 9 أعلام (Flags) كل علم عبارة عن بت واحد .

تقسم الاعلام التسعة الى نوعين:

- أعلام الحالة (Status Flags): عبارة عن ستة أعلام هي CF,PF,AF,ZF,SF,OF.
- أعلام التحكم (Control Flags): عبارة عن ثلاث أعلام للتحكم هي TF,DF,IF.

تتوضع هذه الأعلام في مسجل الاعلام على الشكل التالي:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
×	×	×	×	OF	DF	IF	TF	SF	ZF	×	AF	×	PF	×	CF

❖ أعلام الحالة:

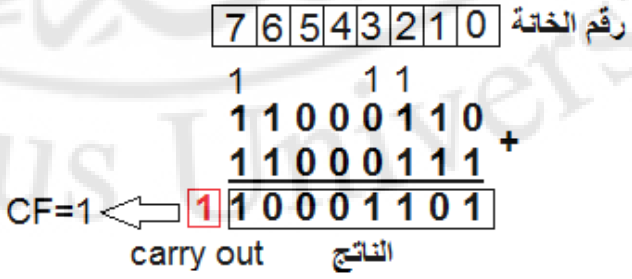
تعكس نتيجة تنفيذ العمليات الحسابية والمنطقية . حيث تكون اما في حالة واحد منطقي (set) أو في حالة صفر منطقي (reset).

1- علم الحمل CF (Carry Flag):

CF=1 اذا وجد حمل خارجي (carry out) أو استعارة (borrow) من أجل الخانة الأخيرة (الخانة 7 أو الخانة 15) .

CF=0 اذا لم يوجد حمل أو استعارة من أجل الخانة الأخيرة (البت الأخير).

مثال:



CF=1 بسبب وجود حمل من البت الأخير للخارج (carry out).

2- علم الزوجية PF (Parity Flag):

PF=1 إذا كانت أول 8 بتات (البايت السفلي) من الناتج يحتوي على عدد زوجي من الواحدات (بعد التحويل للنظام الثنائي طبعا).

PF=0 إذا كانت أول 8 بتات من الناتج تحتوي على عدد فردي من الواحدات.

مثال1:

بفرض قمنا بعملية حسابية وكان الناتج كالتالي: 1010 1100

PF=1 لأن عدد الواحدات (4) زوجي.

مثال2:

بفرض أننا قمنا بعملية حسابية وكان الناتج كالتالي : 1101 1100 1100 1010

PF=0 لأن عدد الواحدات (5) فردي في أول 8 بتات.

3- علم الحمل المساعد AF (Auxiliary Carry Flag):

AF=1 إذا وجد حمل أو استعارة بين الخانة 3 و4 (البت الرابع والخامس).

AF=0 إذا لم يوجد حمل أو استعارة بين الخانة 3 و4.

مثال:

AF=1 بسبب وجود حمل من الخانة 3 الى الخانة 4.

4- علم الصفر ZF (Zero Flag):

ZF=1 عندما يكون ناتج أخر عملية حسابية أو منطقية يساوي الصفر.

ZF=0 عندما يكون ناتج أخر عملية حسابية أو منطقية لا يساوي الصفر.

5- علم الاشارة SF (Sign Flag):

نسخة عن البت الأخير في الناتج.

SF=1 اذا كانت نتيجة أخر عملية حسابية عدد سالب (قيمة البت الأخير تساوي الواحد).

SF=0 إذا كانت نتيجة أخر عملية حسابية عدد موجب (قيمة البت الأخير تساوي الصفر).

مثال:

بفرض كان ناتج عملية حسابية : 1000 1010

SF=1 قيمة أخر بت تساوي (1).

6- علم الفيضانOF (Overflow Flag) :

OF=1 في حال عدم إمكانية تخزين القيمة الناتجة في الموقع المخصص لها.

(اذا وجد carry in فقط أو وجد carry out فقط).

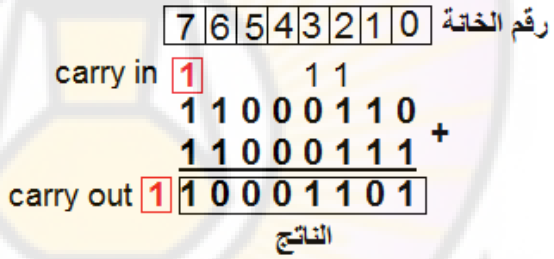
OF=0 في حال امكانية تخزين القيمة الناتجة في الموقع المخصص لها.

(اذا وجد carry out و carry in معا أو لم يوجدا).

Carry in: يحدث حمل من البت قبل الأخير الى البت الأخير.

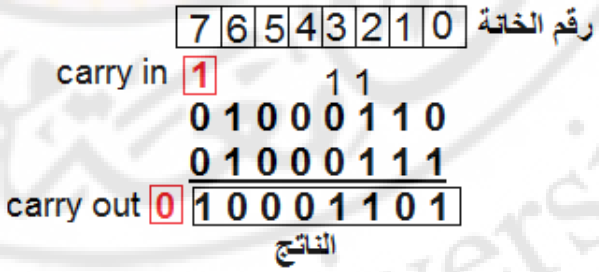
Carry out: يحدث حمل من البت الأخير إالى الخارج.

مثال1:



OF=0 بسبب وجود carry in و carry out معاً.

مثال2:



OF=1 بسبب وجود carry in فقط.

❖ أعلام التحكم:

1- علم الخطوة الوحيدة TF (Trap Flag):

يوضع TF=1 عندما نرغب بتنفيذ البرنامج خطوة بخطوة وهو مفيد عندما نريد تصحيح برنامجنا واكتشاف مواقع الأخطاء.

2- علم المقاطعة IF (Interrupt Flag):

يستخدم من أجل التعبير عن إمكانية أو عدم امكانية تنفيذ المقاطعة.

IF=1 يقوم المعالج بالاستجابة لطلب المقاطعة ويقوم بتنفيذها.

IF=0 يقوم المعالج بتجاهل المقاطعات.

3- علم الإتجاه DF (Direction Flag):

يدل على اتجاه سير العمليات التسلسلية.

DF=1 السلسة تكون من العنوان الأعلى الى العنوان الأدنى.

DF=0 السلسة تكون من العنوان الأدنى الى العنوان الأعلى.

تمرين:

جمع العددين 68D3 و 81F2 :

العددين بنظام السداسي عشر (Hex) نحولهم للنظام الثنائي (Binary) حتى تسهل عملية الجمع وايجاد قيم أعلام الحالة:

```
              11 111    1
81F2:  1000 0001 1111 0010
                                +
68D3:  0110 1000 1101 0011
EAC5:  1110 1010 1100 0101   الناتج
```

CF=0 لأنه لا يوجد carry out.

PF=1 عدد الواحدات في أول 8 بتات من ناتج الجمع زوجي.

AF=0 لأنه لا يوجد حمل من الخانة3 الى الخانة 4.

ZF=0 لأن نتيجة الجمع لا تساوي الصفر.

SF=1 لأن البت الأخير من الناتج هو 1.

OF=0 لأنه لا يوجد carry in ولا يوجد carry out

التحويل بين أنظمة العد:

النظام العشري(Decimal)	النظام الثنائي(Binary)	النظام السداسي عشر(HEX)
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

وحدات الذاكرة:

- البت (bit): هي اصغر وحدة تخزين 0 أو 1 .
- النيبل (nibble): هي أربع بتات.
- البايت (byte): عبارة عن ثمانية بتات.
- كلمة (word): عبارة عن 16bit.
- الكلمة المزدوجة (Double word): عبارة عن 32 bit.
- كيلو بايت عبارة عن 1024 بايت.
- ميجابايت عبارة عن 1024 كيلو بايت.

الكلية التطبيقية

مخبر تنظيم الحاسوب والبرمجة بلغة التجميع

- المحاضرة الثالثة -

البيانات المستخدمة في البرنامج:

1. **الأرقام:**

- تنتهي الأرقام الثنائية بالحرف b أو B للدلالة على أنه رقم ثنائي Binary مثل 11100011b أو 01010110B .
- الأرقام العشرية يمكن كتابتها بدون حرف في النهاية . كما يمكن أن تنتهي بالحرف D أو d للدلالة على أنها عشرية Decimal مثل 1234 و 1345d و -234D .
- الأرقام السداسية عشر يجب أن تبدأ برقم و أن تنتهي بالحرف H أو h للدلالة على أنها سداسية عشر Hexadecimal مثل 56h وفي حال ابتداء الرقم بحرف يجب وضع صفر قبله مثل 0abh (السبب في وضع 0 لتوضيح أن المطلوب هو الرقم السداسي عشر ab وليس المتغيرab).

أمثلة:

الرقم	ملاحظات
1001	رقم عشري
1001b	رقم ثنائي
FFFFh	خطأ(لا يبدأ برقم فيجب وضع 0)
0ab	خطأ(لم يتنهي بالحرف h أو H)
6455	رقم عشري

2. **الحروف:**

يتم وضع الحروف والجمل داخل علامات التنصيص مثل 'A' أو 'Rama' ويتم داخليا تحويل الحرف الى الأرقام المناظرة في ترميز ASCII وتخزينها في الذاكرة .

3. المتغيرات:

المتغير هو موقع في الذاكرة يستخدم لحفظ المعطيات المؤقتة.

التصريح عن متغير:

اسم المتغير	نوع المتغير	قيمة ابتدائية
Sham	db	05h

حيث:

- اسم المتغير: أي اسم شرط ألا يكون كلمة محجوزة.
- نوع المتغير: يأخذ أحد المعاملين:

db لحجز موقع واحد في الذاكرة (8 بت أو 1byte) .

dw لحجز موقعين في الذاكرة (16 بت أو word(كلمة)).

التصريح عن مصفوفة:

```
Suzan    db    48h,56h,6Ch,00h
```

(تم حجز مصفوفة اسمها Suzan بالذاكرة مؤلفة من 4 قيم حجز لكل قيمة موقع واحد بالذاكرة 8 بت)

تعليمات المعالج 8086

تعليمات تبادل المعطيات :

يوجد العديد من التعليمات التي تسمح بتبادل البيانات بين موقع ذاكرة ومسجل ومن هذه التعليمات:

MOV , XCHG, LEA , LDS , LES

1- تعليمة النقل Mov

تستخدم لنقل البيانات من المصدر (Source)الى الهدف (Destination) .

والبيانات تكون اما على صيغة 8 بت أو 16 بت.

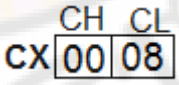
التعليمة	العملية	الأعلام المتأثرة
Mov D,S	S→D	لا يوجد

المصدر (S): إما قيمة فورية (immediately) أو مسجل (Register) أو موقع في الذاكرة(Memory).

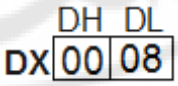
الهدف(D): اما مسجل (Register) أو موقع في الذاكرة(Memory).

أمثلة:

1- نقل قيمة فورية الى مسجل: Mov CL,08



2- نقل من مسجل الى مسجل : Mov DX,CX

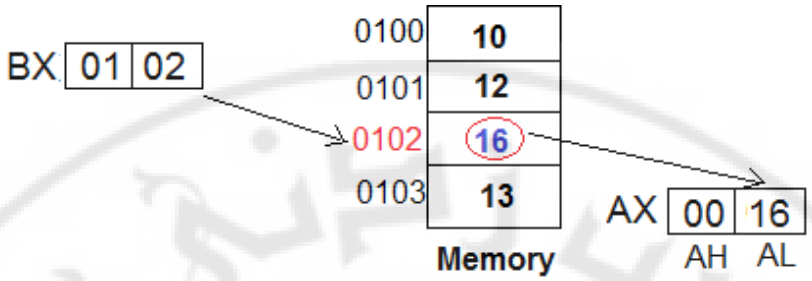


3- نقل من ذاكرة الى مسجل :

Mov BX,0102h

Mov AL,[BX]

قيمة BX تمثل عنوان بالذاكرة يتم نقل محتويات هذا العنوان الى المسجل AL.



2 - تعليمة التبديل XCHG

تقوم التعليمة بتبديل القيم بين المصدر (S) و الهدف(D).

التعليمة	العملية	الأعلام المتأثرة
XCHG D,S	S→D D→S	لا يوجد

المصدر (S): مسجل (Register) أو موقع في الذاكرة(Memory).

الهدف(D): مسجل (Register) أو موقع في الذاكرة(Memory).

- يمكن التبديل بين (مسجل و مسجل) أو (مسجل و ذاكرة) أو (ذاكرة و مسجل)

مثال:

```
Mov AL,07
Mov AH,05
XCHG AH,AL
```

- في التعليمة الأولى تم اسناد القيمة 07 إلى المسجل AL.
- في التعليمة الثانية تم اسناد القيمة 05 إلى المسجل AH.

- في التعليمة الثالثة تم التبديل بين قيم المسجل AH والمسجل AL.

أي أصبحت AL=05 و AH=0

4- تعليمة LEA:

وظيفتها تحميل عنوان فعال (EA) من الذاكرة بمسجل طوله 16 bit (reg16) .

5- تعليمة LDS:

تقوم هذه التعليمة بتحميل أربع بايتات(32 bit) من الذاكرة (Memory) وتضع أول بايتين في المسجل(reg16) المحدد كوسيط وثاني بايتين في مسجل مقطع المعطيات DS.

6- تعليمة LES :

تقوم هذه التعليمة بتحميل أربع بايتات(32 bit) من الذاكرة وتضع أول بايتين في المسجل(reg16) المحدد كوسيط وثاني بايتين في مسجل مقطع المعطيات الاضافي ES.

الكلية التطبيقية

مخبر تنظيم الحاسوب والبرمجة بلغة التجميع

- المحاضرة الرابعة -

تعليمات المعالج 8086

تعليمات القفز JMP:

الغاية من تعليمات القفز هي تعديل طريق تنفيذ التعليمات في البرنامج وهناك نوعان من تعليمات القفز:

1- تعليمات القفز غير الشرطية :

لا يوجد أي شروط من أجل حدوث القفز.

التعليمة	العملية	الأعلام المتأثرة
JMP operand	القفز الى العنوان المحدد بواسطة المتحول operand	لا يوجد

ملاحظة: المتحول operand هو العنوان الذي سيتم القفز اليه ويمكن أن يكون أي اسم بشرط ألا يكون كلمة محجوزة.

مثال:

```
Mov AL,05
Jmp Sulaf
Mov BX,100
Sulaf :
```

التعليمة Mov BX,100 لن تنفذ لأنه سوف يتم القفز مباشرة الى العنوان (sulaf) الموجود في تعليمة Jmp .

2- تعليمات القفز الشرطية:

يوجد شرط من أجل حدوث القفز. حيث يتم فحص حالة أعلام مسجل الأعلام فإذا حقق العلم شرط معين تنفذ عملية القفز الى العنوان المحدد بالتعليمة . والا يتابع المعالج تنفيذ التعليمة التي تلي مباشرة تعليمة القفز في البرنامج.

التعليمة	المعنى
JZ	اقفز إذا كان ZF=1
JNZ	اقفز إذا كان ZF=0
JC	اقفز إذا كان CF=1
JNC	اقفز إذا كان CF=0
JS	اقفز إذا كان SF=1
JNS	اقفز إذا كان SF=0
JP	اقفز إذا كان PF=1
JNP	اقفز إذا كان PF=0
JO	اقفز إذا كان OF=1
JNO	اقفز إذا كان OF=0

تستخدم تعليمات القفز الشرطية في انجاز الحلقات أو العبارات البرمجية مثل (if –else) كما تستخدم في تعليمات المقارنة لايجاد العامل الأكبر.

مثال:

```
Mov AL,05
JC Noor
Mov BX,100
Noor :
```

- في حال تحقق شرط القفز (CF=1) يتم القفز الى العنوان(Noor) وبالتالي التعليمة Mov BX,100 لن تنفذ.
- في حال عدم تحقق شرط القفز(CF=0) لن يقفز و سوف ينفذ التعليمة Mov BX,100 .

العمليات الحسابية:

وهي تشمل عمليات الجمع، الطرح، الضرب والقسمة.

- تعليمات الجمع:

1- تعليمة ADD :

تقوم هذه التعليمة بجمع المصدر (S) والوجهة (D) وتضع الناتج في الوجهة(D).

التعليمة	العملية	الأعلام المتأثرة
ADD D,S	S+D→D Carry→CF	جميع أعلام الحالة

مثال عملي:

اكتب برنامج لجمع القيمتين 68D3 و 81F2 ووضعهم في المسجل CX وماقيمة أعلام الحالة ؟

الحل:

```
Mov CX,81F2h
Mov BX,68D3h
ADD CX,BX
```

CF=0 PF=1 AF=0 ZF=0 SF=1 OF=0

ملاحظة:(الحل الرياضي موجود بالجلسة1)

2- تعليمة ADC :

تقوم هذه التعليمة بجمع المصدر (S) والوجهة (D) والحمل (CF) وتضع الناتج في الوجهة(D).

التعليمة	العملية	الأعلام المتأثرة
ADC D,S	S+D+CF→D Carry→CF	جميع أعلام الحالة

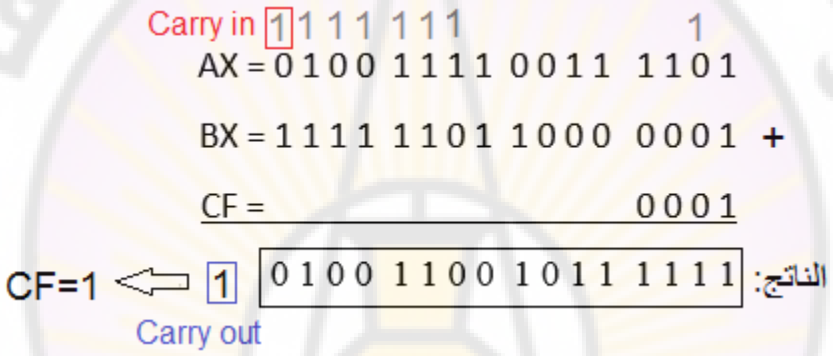
مثال: بفرض AX=4F3Dh و BX=FD81h و CF=1 فما هي نتيجة تنفيذ التعليمة

ADC AX,BX ؟

مبينا حالة أعلام الحالة بعد تنفيذ هذه التعليمة.

الحل:

العددين بنظام السداسي عشر (HEX) نحولهم للنظام الثنائي(Binary) حتى تسهل عملية الجمع وإيجاد قيم أعلام الحالة.



ناتج الجمع بالنظام السداسي عشر: 4CBF ويتم تخزين القيمة بالمسجل AX (الوجهة)

حالة أعلام الحالة بعد تنفيذ العملية:

CF=1 لأنه يوجد carry out .

PF=0 عدد الواحدات في أول 8 بتات من ناتج الجمع(7) فردي.

AF=0 لأنه لا يوجد حمل من الخانة 3 الى الخانة 4.

ZF=0 لأن نتيجة الجمع لا تساوي الصفر.

SF=0 لأن البت الأخير من الناتج هو 0 .

OF=0 يوجد carry in و carry out معا .

3- تعليمة INC :

تقوم هذه التعليمة بزيادة الوجهة (D) بمقدار واحد وتضع الناتج في الوجهة (D).

التعليمة	العملية	الأعلام المتأثرة
INC D	D+1→D	أعلام الحالة باستثناء CF

مثال: ما هو محتوى المسجل AL بعد تنفيذ البرنامج وما قيمة أعلام الحالة:

```
MOV  AL,7Fh
INC  AL
```

الحل:

```
        1111 111
AL=7F=0111 1111
                +
           1
-----------------
الناتج: 80h =1000 0000
```

وبالتالي بعد تنفيذ البرنامج تصبح قيمة المسجل AL=80

و أعلام الحالة:

AF=1 PF=0 OF=1 SF=1 ZF=0 CF=0(لا توثر تعليمة INC عليه)

الكلية التطبيقية

تنظيم الحاسوب والبرمجة بلغة التجميع

- المحاضرة الخامسة -

تعليمات المعالج 8086

- تعليمات الطرح:

1- تعليمة SUB :

تقوم هذه التعليمة بطرح المصدر(S) من الوجهة (D) وتضع الناتج في الوجهة (D).

الأعلام المتأثرة	العملية	التعليمة
جميع أعلام الحالة	D - S→D (استعارة)borrow→CF	**SUB D,S**

مثال:

```
MOV  AL,5
SUB  AL,1    ; AL=4
```

ملاحظة: العبارة المكتوبة بعد الفاصلة المنقوطة تعتبر تعليق أي لا يتم تنفيذها .

2- تعليمة SBB :

تقوم هذه التعليمة بطرح المصدر(S) و الاستعارة(CF)(الحمل في حالة الجمع) من الوجهة (D) وتضع الناتج في الوجهة(D).

الأعلام المتأثرة	العملية	التعليمة
جميع أعلام الحالة	D-S-CF→D borrow→CF	**SBB D,S**

مثال:

بفرض CF=1

```
MOV AL,5
SBB AL,3    ; AL=5-3-1 =1
```

3- تعليمة DEC:

تقوم هذه التعليمة بانقاص الوجهة (D) بمقدار واحد وتضع الناتج في الوجهة(D).

التعليمة	العملية	الأعلام المتأثرة
DEC D	D-1→D	أعلام الحالة باستثناء CF

4- تعليمة NEG :

تقوم هذه العملية بإيجاد المتتم الثنائي للوجهة(D) وتخزن الناتج في الوجهة(D).

التعليمة	العملية	الأعلام المتأثرة
NEG D	0-D→D	جميع اعلام الحالة

مثال: ما قيمة المسجل AX بعد تنفيذ البرنامج التالي :

```
MOV AX,03F8h
NEG  AX
```

الحل:

التعليمة **MOV AX,03F8h** تنقل القيمة 03F8 الى المسجل AX .

التعليمة **NEG AX** تقوم بإيجاد المتمم الثنائي لمحتوى المسجل AX أي للقيمة 03F8.

يتم الحصول على المتتم الثنائي للعدد 03F8 كما يلي:

- تحويل هذا العدد للنظام الثنائي فيصبح 0000 0011 1111 1000
- قلب الأصفار واحدات والواحدات أصفار فيصبح 1111 1100 0000 0111
- نضيف واحد للرقم الناتج فنحصل على المتمم الثنائي

```
        111
1111 1100 0000 0111
                1 +
---------------------
```

المتمم الثنائي: 1111 1100 0000 1000

نحوله للنظام السداسي عشر : FC08 وهو محتوى المسجل AX بعد تنفيذ البرنامج.

تعليمة الضرب MUL

- MUL تعليمة ضرب بدون اشارة.

MUL S

تأخذ معامل وحيد وهو المصدر (S): مسجل أو موقع في الذاكرة.

- في حال كان حجم المصدر (S) 8 بت :

AL * S →AX

- في حال كان حجم المصدر (S) 16 بت :

AX * S→ DXAX

مثال:

```
MOV   AL,2
MOV   BL,4
MUL   BL   ; AX=2*4=8
```

التعليمات المنطقية:

تقوم هذه التعليمات بإجراء العملية المنطقية بين المصدر(S) والوجهة(D) (كل بت في المصدر مع مقابله في الوجهة) وتضع النتيجة في الوجهة(D).

1- تعليمة AND:

تعمل عملية AND بين المصدر (S) والوجهة (D) وتضع الناتج في الوجهة (D)

الأعلام المتأثرة	العملية	التعليمة
اعلام الحالة	S.D→D	**AND D,S**

تذكير: 0 AND 0 = 0 0 AND 1 = 0 1 AND 0 = 0 1 AND 1 = 1

2- تعليمة OR:

تعمل عملية OR بين المصدر (S) والوجهة (D) وتضع الناتج في الوجهة (D)

الأعلام المتأثرة	العملية	التعليمة
اعلام الحالة	S+D→D	**OR D,S**

تذكير: 0 OR 0 = 0 0 OR 1 = 1 1 OR 0 = 1 1 OR 1 = 1

3- تعليمة XOR:

تعمل عملية XOR بين المصدر (S) والوجهة (D) وتضع الناتج في الوجهة (D)

الأعلام المتأثرة	العملية	التعليمة
اعلام الحالة	S⊕D→D	**XOR D,S**

تذكير: 0 XOR 0 = 0 0 XOR 1 = 1 1 XOR 0 = 1 1 XOR 1 = 0

4- تعليمة NOT:

تعمل عملية NOT على قلب الواحدات أصفار والأصفار واحدات للوجهة (D) وتخزن الناتج في الوجهة (D) .

التعليمة	العملية	الأعلام المتأثرة
NOT D	$\overline{D} \rightarrow D$	لا يوجد

مثال:

ما محتوى المسجل BL بعد تنفيذ البرنامج:

```
MOV BL,36h
MOV AH,07h
AND BL,2Ch
XOR BL,AH
NOT BL
```

- التعليمة MOV BL,36h تضع القيمة 36h في المسجل BL.
- التعليمة MOV AH,07h تضع القيمة 07h في المسجل AH.
- التعليمة AND BL,2Ch تعمل عملية AND بين محتوى المسجل BL والقيمة 2Ch

BL = 36h = 0 0 1 1 0 1 1 0

2Ch = 0 0 1 0 1 1 0 0 AND

24h = 0 0 1 0 0 1 0 0 تخزن في المسجل BL

- التعليمة XOR BL,AH تعمل عملية XOR بين محتوى المسجل AH ومحتوى المسجل BL .

$$BL = 24h = 0\,0\,1\,0\;\;0\,1\,0\,0$$
$$AH = 07h = \underline{0\,0\,0\,0\;\;0\,1\,1\,1} \quad XOR$$
$$23h = 0\,0\,1\,0\;\;0\,0\,1\,1 \text{ تخزن في المسجل BL}$$

- التعليمة NOT BL تقوم بقلب الواحدات أصفار والأصفار واحدات لمحتوى المسجل BL.

$$BL = 23h = \underline{0\,0\,1\,0\;\;0\,0\,1\,1} \quad NOT$$
$$DCh = 1\,1\,0\,1\;\;1\,1\,0\,0 \text{ تخزن في المسجل BL}$$

وبالتالي تصبح BL=DCh بعد تنفيذ البرنامج.

الكلية التطبيقية

تنظيم الحاسوب والبرمجة بلغة التجميع

- المحاضرة السادسة -

تعليمات المعالج 8086

تعليمات الازاحة:

هناك نوعان من تعليمات الإزاحة: الإزاحة المنطقية والازاحة الحسابية.

تعليمة SAL/SHL:

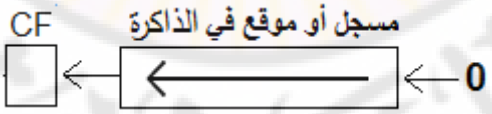
تعليمة SHL: إزاحة منطقية نحو اليسار.

تعليمة SAL : إزاحة حسابية نحو اليسار.

SHL/SAL D,S

- الوجهة(D) : موقع في الذاكرة (Memory) أو مسجل (Register).
- المصدر(S) : يمثل عدد مرات الازاحة ويكون اما قيمة فورية (immediate) أو محتوى المسجل CL .

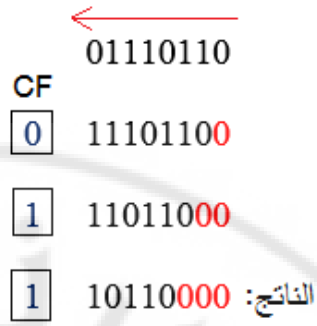
❖ التعليمتان SAL/SHL متكافئتان وتعملان على إزاحة محتويات معامل الوجهة نحو اليسار باتجاه CF عدد من الخانات مساو لقيمة المعامل الثاني وتدخل صفر في البت الأول من المعامل بعد كل إزاحة.



مثال1:

MOV BL , 01110110B

SAL BL , 3



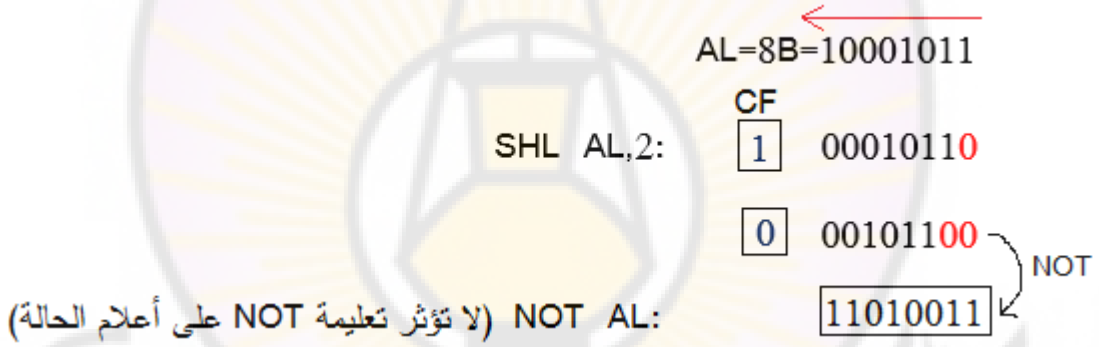
مثال2:

ما محتوى المسجل AL بعد تنفيذ البرنامج وما قيمة CF:

MOV AL ,8Bh

SHL AL,2

NOT AL



محتوى AL بعد تنفيذ البرنامج D3h=11010011 و CF=0.

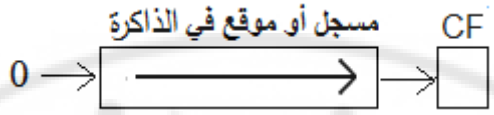
تعليمة SHR:

تعليمة SHR :إزاحة منطقية نحو اليمين.

SHR D,S

نفس الوجهة والمصدر لتعليمة SHL/SAL.

❖ التعليمة SHR تعمل على إزاحة محتويات معامل الوجهة نحو اليمين باتجاه CF عدد من الخانات مساو لقيمة المعامل الثاني وتدخل صفر في البت الأخير من المعامل بعد كل إزاحة.



مثال:

```
MOV   AL,10001011B
MOV   CL ,2
SHR   AL,CL
```

10001011 →

CF

01000101 [1]

الناتج: 00100010 [1]

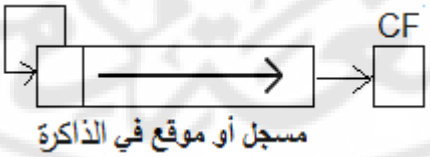
تعليمة SAR:

تعليمة SAR :إزاحة حسابية نحو اليمين.

SAR D,S

نفس الوجهة والمصدر لتعليمة SHL/SAL.

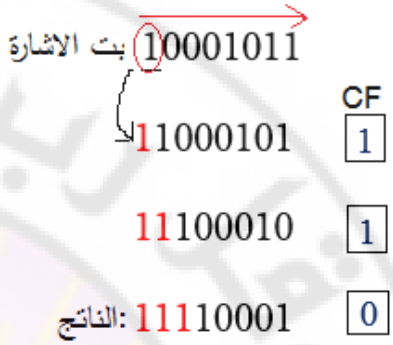
❖ التعليمة SAR تعمل على إزاحة محتويات معامل الوجهة نحو اليمين باتجاه CF عدد من الخانات مساو لقيمة المعامل الثاني ويجري دوما ادراج بت الإشارة في البت الأخير من المعامل بعد كل إزاحة.



مثال1:

MOV BL,10001011B

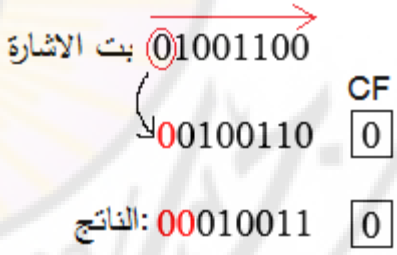
SAR BL,3



مثال2:

MOV AL,01001100B

SAR AL,2

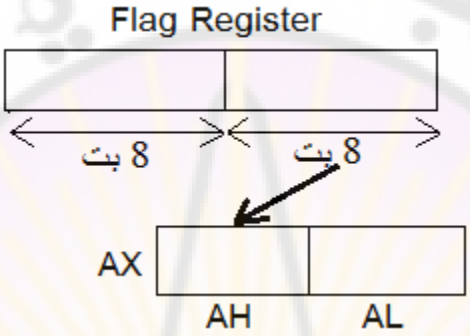


تعليمات مسجل الأعلام(Flag Register):

تعليمة LAHF:

تقوم بتحميل البايت السفلي (أول 8 بتات) من مسجل الأعلام (Flag Register) في AH.

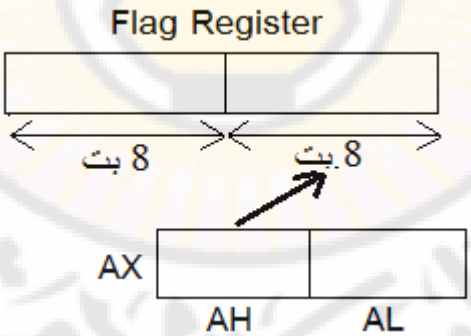
AH← Flags



تعليمة SAHF:

تقوم بتخزين قيمة AH في البايت السفلي (أول 8 بتات) من مسجل الأعلام (Flag Register).

AH → Flags



تعليمة CLC:

تضع 0 في علم الحمل (CF) : CF ← 0

تعليمة STC:

تضع 1 في علم الحمل (CF) : CF ← 1

تعليمة CMC:

تعكس حالة علم الحمل(CF):

إذا كان CF=0 بعد تنفيذ تعليمة CMC يصبح CF=1

إذا كان CF=1 بعد تنفيذ تعليمة CMC يصبح CF=0

تعليمة CLI:

تضع 0 في علم المقاطعة (IF) : IF ← 0

تعليمة STI:

تضع 1 في علم المقاطعة (IF) : IF ← 1

تعليمة CLD

تضع 0 في علم الاتجاه (DF) : DF ← 0

تعليمة STD:

تضع 1 في علم الاتجاه (DF) : DF ← 1

تمرين مهم :

حدد محتوى المسجل DL و CF بعد تنفيذ البرنامج التالي:

```
MOV   DL,10h
INC   DL
OR    DL,03h
SAL   DL,2
CMC
```

الحل:

MOV DL,10h DL=10h=00010000

INC DL DL=00010001

OR DL, 03h

```
  00010001
OR
  00000011
  --------
  00010011
```

DL=00010011

SAL DL, 2

```
  00010011
CF
0 00100110
0 01001100
  --
  DL
```

CMC تعكس حالة علم الحمل وبالتالي يصبح CF =1

وبالتالي محتوى المسجل DL بعد تنفيذ البرنامج: DL = 01001100 = 4C h

CF = 1

الكلية التطبيقية

تنظيم الحاسوب والبرمجة بلغة التجميع

- المحاضرة السابعة –

التوابع في لغة التجميع (Assembly):

لاستخدام توابع جاهزة في لغة التجميع يجب تضمين المكتبة التي تضم هذه التوابع في بداية البرنامج عن طريق التعليمة include.

include “Emu8086.inc”

تضمين المكتبة Emu8086.inc للبرنامج من أجل استخدام التوابع الموجودة ضمنها.

- تابع ادخال الأرقام من قبل المستخدم (scan_num) ويتم استدعائه عن طريق التعليمة التالية:

Call scan_num

حيث يتم تخزين القيمة التي يتم ادخالها في المسجل CX .

- تابع طباعة الأرقام مع اشارة (print_num) ويتم استدعائه عن طريق التعليمة التالية:

Call print_num

حيث يقوم بطباعة الرقم موجود بالمسجل AX .

- تابع طباعة الأرقام بدون اشارة (print_num_uns) ويتم استدعائه عن طريق التعليمة التالية:

Call print_num_uns

حيث يقوم بطباعة رقم موجود بالمسجل AX وهذا التابع لا يأخذ الإشارة بعين الاعتبار.

❖ يجب التصريح عن التوابع المستخدمة في نهاية البرنامج ويتم التصريح عن طريق كلمة define ثم نكتب اسم التابع.

define_scan_num
define_print_num
define_print_num_uns

ملاحظة:

سواء استخدمت التابع print _num أو print _num_uns يجب التصريح عن التابعين في نهاية البرنامج لأن التابعين مرتبطين ببعضهما.

برنامج1:

اكتب برنامج بلغة التجميع يسمح بإدخال رقمين من لوحة المفاتيح وجمعهما وطباعة نتيجة الجمع على الشاشة.

الحل:

```
include "Emu8086.inc"
call scan_num
mov ax,cx
putc 10
call scan_num
add ax,cx
putc 10
call print_num
define_scan_num
define_print_num
define_print_num_uns
hlt
```

شرح البرنامج:

- يتم ادخال الرقم الأول عن طريق استدعاء تابع الادخال call scan_num ويتم حفظ الرقم المدخل بالمسجل cx.
- باعتبار انه سوف يتم ادخال رقم ثاني يجب وضع الرقم الأول بمسجل أخر غير مسجل cx لذلك نستخدم التعليمة mov ax,cx لنقل الرقم الأول الى المسجل ax.
- تستخدم التعليمة putc 10 للانتقال الى سطر جديد عند ادخال الرقم الثاني.
- يتم ادخال الرقم الثاني عن طريق استدعاء تابع الادخال call scan_num ويتم حفظ الرقم الثاني بالمسجل cx.
- اصبح لدي الرقم الأول بالمسجل ax والرقم الثاني بالمسجل cx باستخدام التعليمة add ax,cx يتم جمع الرقمين ووضع الناتج بالمسجل ax.
- تستخدم التعليمة putc 10 للانتقال الى سطر جديد عند طباعة نتيجة الجمع.
- لطباعة نتيجة الجمع يتم استدعاء تابع الطباعةcall print_num الذي يطبع القيمة الموجودة بالمسجل ax (نتيجة الجمع) على الشاشة .

- يتم التصريح عن التوابع المستخدمة في نهاية البرنامج عن طريق التعليمات:

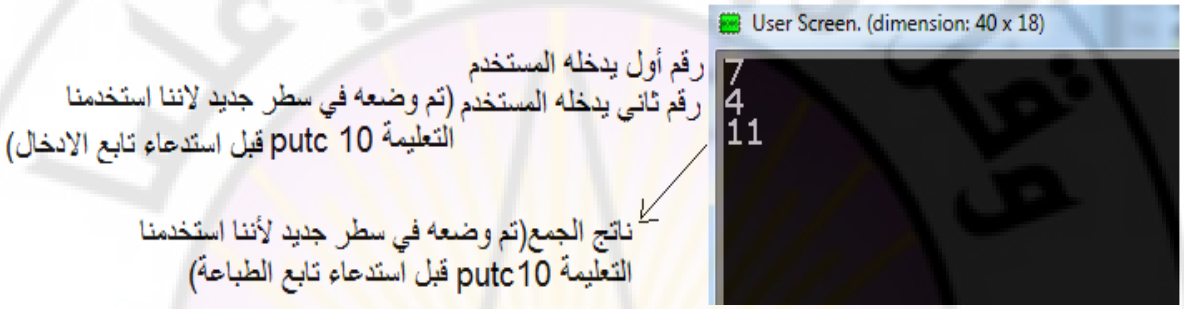
define_scan_num

define_print_num

define_print_num_uns

- hlt للإيقاف .

الخرج التنفيذي للبرنامج بعد الضغط على run:



برنامج2:

اكتب برنامج بلغة التجميع للمعالج 8086 يقوم بطلب ادخال قيمة المتحول X وحساب وطباعة القيمة Y .
حسب المعادلة: **Y=4X-8**

```
include "Emu8086.inc"
call scan_num
mov ax,4
mul cx
sub ax,8
putc 10
call print_num
define_scan_num
define_print_num
define_print_num_uns
```

- يتم ادخال قيمة المتحول x عن طريق استدعاء تابع الادخال call scan_num ويتم وضع قيمة x بالمسجل cx .
- نريد ضرب المتحول x بالعدد 4 وباعتبار تعليمة الضرب تأخذ معامل وحيد (المسجل cx لأنه يحتوي على قيمة x) والمعامل الثاني هو قيمة المسجل ax لذلك يتم وضع العدد 4 في المسجل ax عن طريق التعليمة mov ax,4.
- mul cx تضرب قيمة المسجل cx بقيمة المسجل ax ويتم وضع الناتج (4x) بالمسجل ax.
- التعليمة sub ax,8 تطرح العدد 8 من قيمة المسجل ax (4x) وتضع الناتج (4x-8) في المسجل ax أي أصبح لدينا قيمة y موجودة بالمسجل ax.
- تستخدم التعليمة putc 10 للانتقال الى سطر جديد عند طباعة القيمة y.
- لطباعة القيمة y يتم استدعاء تابع الطباعة call print_num الذي يطبع القيمة الموجودة بالمسجل ax (4x-8) على الشاشة .
- يتم التصريح عن التوابع المستخدمة في نهاية البرنامج عن طريق التعليمات:

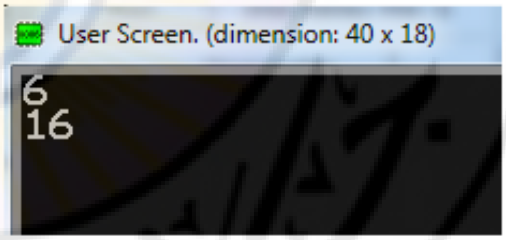
  define_scan_num

  define_print_num

  define_print_num_uns

- hlt للإيقاف .

الخرج التنفيذي للبرنامج بعد الضغط على run:



قيمة المتحول x التي يدخلها المستخدم.
قيمة y

برنامج 3:

اكتب برنامج بلغة التجميع للمعالج 8086 يقوم بطلب ادخال قيمة المتحول X وحساب وطباعة القيمة Y .
حسب المعادلة: $y=X^2 + X$

```
include "Emu8086.inc"
call scan_num
mov ax,cx
mul ax
add ax,cx
putc 10
call print_num
define_scan_num
define_print_num
define_print_num_uns
```

برنامج4:

اكتب برنامج بلغة التجميع للمعالج 8086 يقوم بطلب ادخال قيمة المتحولين x1 و x2 وحساب وطباعة القيمة Y . حسب المعادلة: $y=2X_1 + X_2$

```
include "Emu8086.inc"
call scan_num
mov ax,2
mul cx
putc 10
call scan_num
add ax,cx
putc 10
call print_num
define_scan_num
define_print_num
define_print_num_uns
```