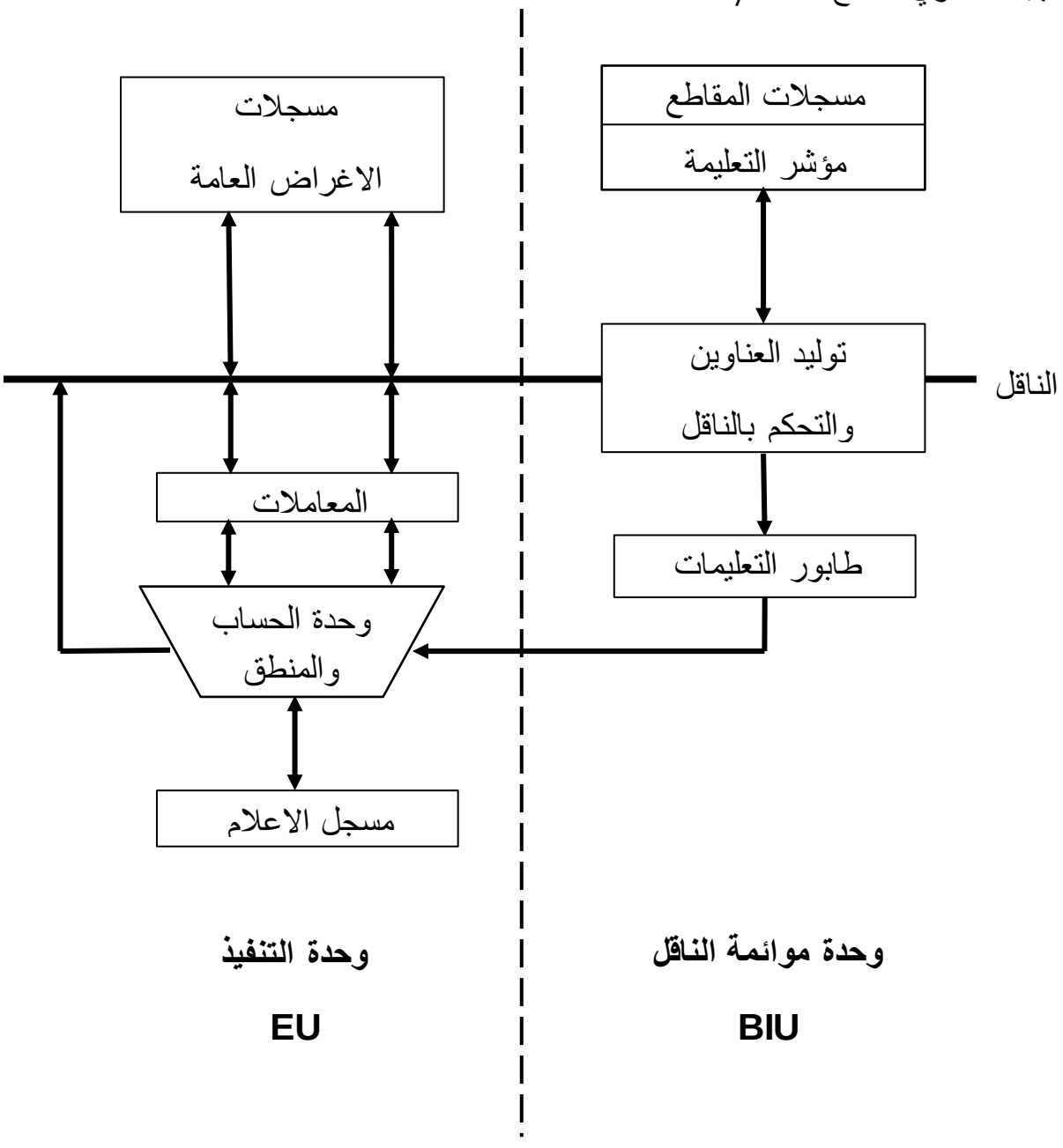
معمارية المعالج 8086/8088

ان التقدم الكبير في تقنية نظام الحاسب الذي حدث في العقدين الاخيرين هو ناتج للتطور الذي حدث في معمارية المعالجات ذات $(16bit, 32bit, 64bit)$ وكذلك انظمة الحاسب المبنية بهذه المعالجات, خلال هذه الفترة حدث تغير كبير في استخدام الحاسبات الكبيرة الغالية الثمن الى الحاسبات الشخصية الاقل كلفة.
ان اول حاسب شخصي ظهر في بداية الثمانينيات حيث كان يستخدم المعالج (8086) ذات $16bit$ كوحدة معالجة له وبعد عدة سنوات ظهر جيل اخر في الحاسب الشخصي (PC/AT) الذي مثل باستخدام معالج اكثر تطور وهو (80286) وفي عام 1985 ظهر المعالج (80386) ذات $(32bit)$ الذي حسن في اداء الحاسب الشخصي في السنوات التي تلتها وسعت شركة انتل معمارية الشخصي في السنوات التي تلتها وسعت شركة انتل معمارية $32bit$ في المعالج (80486) وعوائل المعالج $(Pentium)$ اعطت هذه المعالجات مستويات جديدة من الاداء والسعات للحاسب الشخصي.

التركيب المعماري للمعالج 8086/8088



الشكل (1)

يوضح الشكل (1) التركيب المعماري للمعالج والذي يتكون من الوحدتين

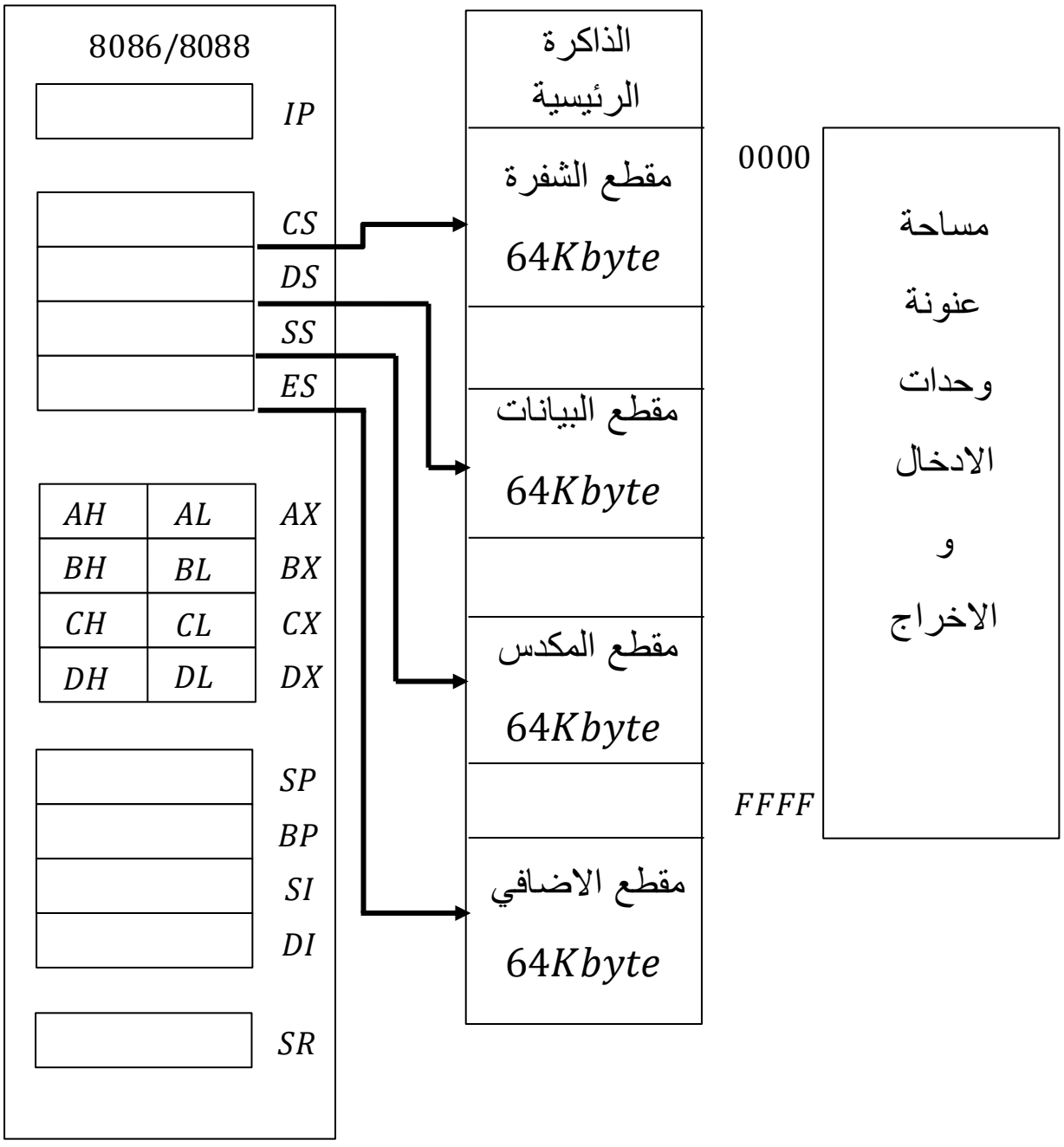
1- وحدة موائمة الناقل $Bus\ Interface\ Unit\ (BIU)$

وهي وحدة ربط المعالج بالعالم الخارجي وهي المسؤولة عن جلب التعليمة وقراءة البيانات من والى الذاكرة. تحتوي وحدة موائمة الناقل على مسجلات المقاطع ومؤشر التعليمة ووحدة التحكم بالناقل وتوليد العناوين وكذلك طابور التعليمات.

2- وحدة التنفيذ $Execution\ Unit\ (EU)$

وهي المسؤولة عن تفسير وتمثيل التعليمات. تتكون وحدة التنفيذ من وحدة الحساب والمنطق ومسجلات الاغراض العامة ومسجل الاعلام.

النموذج البرمجي للمعالج 8086/8088



الشكل (2)

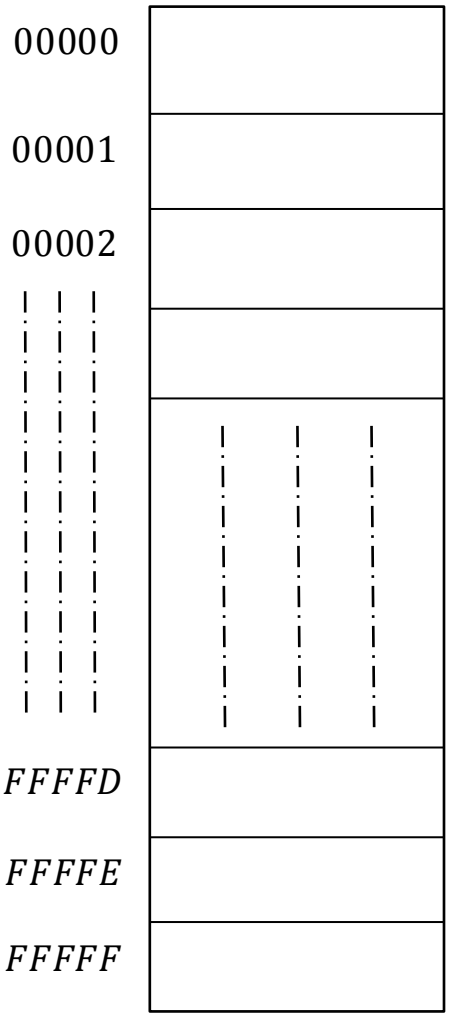
ان الشكل (2) يوضح النموذج البرمجي للمعالج. الغرض من هذا النموذج هو المساعدة المبرمج على فهم مبدأ عمل نظام الحاسب من وجهة نظر برمجية.

يتكون هذا النموذج من مجموعة من المسجلات الداخلية ذات $16bit$.

مؤشر التعليمة (IP) ومسجلات الاغراض العامة (DX, CX, BX, AX) واربعة مسجلات مقاطع (ES, SS, DS, CS) ومسجلي تأشير (BP, SP) ومسجلي فهرسة (DI, SI) ومسجل الحالة (SR).

يحتوي النموذج البرمجي ايضاً على $(1M\ byte)$ من الذاكرة و $(64K\ byte)$ من العنونة المعزولة للادخال والاخراج.

عنونة الذاكرة

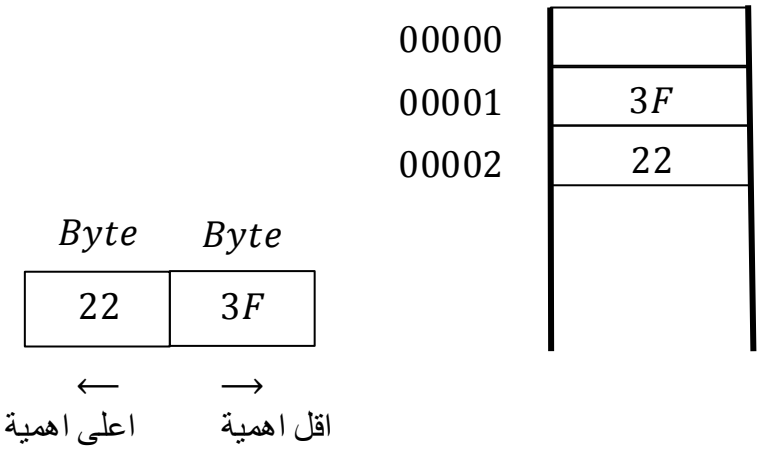


الشكل (3)

المعالج (8086) يدعم $1M\ byte$ من الذاكرة الخارجية كما موضح في الشكل (3).

يعامل المعالج اي موقع من الذاكرة كبايت واحد واي موقعين متتاليين من الذاكرة ككلمة $(1word)$ في هذه حالة يمثل البايت الاقل عنوان من الذاكرة البايت الاقل اهمية من الكلمة والبايت الاعلى عنوان من الذاكرة هو البايت الاكثر اهمية من الكلمة.

مثلاً من الشكل (3):-

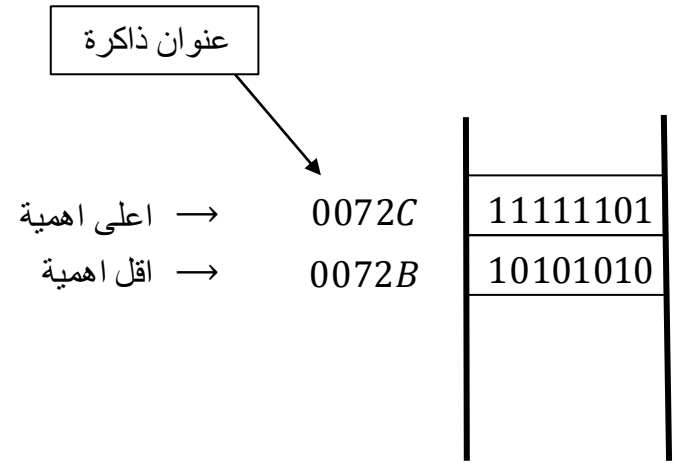


$\therefore Word = 223F$

الكلمة المضاعفة $Double\ Word$

هي الشكل الاخر من البيانات والتي يتم معالجتها من قبل المعالج. تتمثل الكلمة المضاعفة بأربعة مواقع متتالية من البيانات في الذاكرة.

مثال/ ماهي كلمة البيانات الناتجة من الشكل (4) عبر عن الكلمة بالنظام السادس عشر.



الحل/

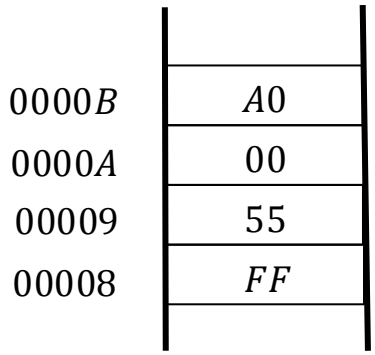
توضيح:

$$\underbrace{1111}_{F}\underbrace{1101}_{D}\underbrace{1010}_{A}\underbrace{1010}_{A}$$

$\therefore Word = (1111111011010110)_2$

$\therefore Word = FDAA_{16}$

Ex / ماهي كلمة البيانات المضاعفة الناتجة من الشكل (5) عبر عن الكلمة بالنظام السادس عشر.



$double\ word(DW) = (A00055FF)_{16}$

Ex / وضح بالرسم كيفية توزيع كلمة البيانات $3C8E$ بالنظام الثنائي على موقعي الذاكرة $(0000A)_{16}$ و $(00009)_{16}$

مثال/ وضح بالرسم كيفية توزيع كلمة البيانات المضاعفة $(00D01CA2)_{16}$ بالنظام الثنائي على موقع الذاكرة $(0001B)_{16} \rightarrow (00018)_{16}$

الحل/

$DW = (00D01CA2)_{16}$

بالنظام الثنائي

$DW = (00000000110100000001110010100010)_2$

المسجلات $Registers$

هو مخزن البيانات وتخزن على شكل بتات.(وهي مكان لخزن البيانات)

تتحكم المسجلات بطبيعة مهام التعليمات التي تنفذ فمن خلالها نستطيع عنونة الذاكرة وتطبيق العمليات الحسابية عليها, يشار الى المسجلات من خلال اسمائها وترقم خاناتها الثنائية ابتدأ من اليمين الى اليسار علماً ان كل المسجلات في المعالج تكون بطول $16bit$ وهي على عدة انواع.

1/ مسجلات المقاطع $Segment\ Registers$

تستخدم هذه المسجلات لعنونة منطقة من الذاكرة تعرف بالمقطع الحالي وهي عبارة عن:-

أ/ مسجل مقطع التعليمات $Code\ Segment\ Registers$

يحتوي هذا المسجل على عنوان بداية مقطع التعليمات الذاكرة ونستطيع من خلال القيمة الموجودة في هذا المسجل والقيمة الموجودة في مسجل مؤشر التعليمة ان تصل الى عنوان اي تعليمة موجودة ضمن مقطع التعليمات.

ب/ مسجل مقطع البيانات $Data\ Segment\ Register$

يحتوي هذا المسجل على عنوان بداية مقطع البيانات في الذاكرة ونستطيع من خلال القيمة الموجودة في التعليمة ان تصل الى محتويات اي خلية في الذاكرة ضمن مقطع البيانات.

ج/ مسجل مقطع المكدس $Stack\ Segment\ Register$

يحتوي هذا المسجل على بداية المقطع المكدس في الذاكرة والذي يستخدم لتخزين المؤقت للبيانات.

د/ مسجل المقطع الاضافي $Segment\ Register\ Extra$

حجمه $16\ Bit$ يحتوي هذا المسجل على عنوان بداية المقطع الاضافي والذي يستخدم في معالجة السلاسل الرمزية $Strings$.

مقطع الشفرة CS
مقطع البيانات DS
مقطع المكدس SS
مقطع الإضافى ES

2/ مسجل مؤشر التعليمة (IP) $Instruction\ Pointer$

حجمه $16\ Bit$ يستخدم هذا المسجل لتحديد عنوان التعليمة القادمة التي سينفذها المعالج والموجود في مقطع التعليمات. العنون الفعلي للتعليمة القادمة يحدد من خلال قيمة مسجل مؤشر التعليمة وقيمة مسجل مقطع التعليمات.

3/ مسجل التأشير $Pointer\ Register$

يستخدم هذه المسجلات للوصول الى البيانات الموجودة ضمن مقطع المكدس وهي على نوعين:-

أ/ مسجل مؤشر المكدس (SP) $Stack\ Pointer\ Registers$

يحتوي هذا المسجل على العنوان الحالي للتخزين او الاسترجاع ضمن مقطع المكدس.

العنوان الفعلي لمقطع المكدس يحدد من خلال قيمة مسجل مؤشر وقيمة مسجل مقطع المكدس.

ب/ مسجل مؤشر القاعدة (BP) $Base\ Pointer\ Register$

يستخدم هذا المسجل للتعامل مع البيانات الموجودة ضمن مقطع المكدس.

القيمة الفعلية لعنوان ضمن مقطع المكدس يحدد من خلال القيمة الموجودة ضمن مسجل مؤشر القاعدة ومسجل مقطع المكدس.

4/ مسجلات الاغراض العامة $General\ Purpose\ Register$

تستخدم هذه المسجلات في معظم التعليمات التي ينفذها المعالج. يتكون كل مسجل من هذه المسجلات من كلمة بيانات $word = 2byte = 16bit$ والتي يمكن التعامل معها على انها مسجل بطول $2byte$ او تجزئتها الى مسجلين بطول $1bit$ فمثلا المسجل AX الذي طوله $16bit$ يجزئ الى المسجل الاقل اهمية.

أ/ مسجل المركم (AX) $Accumulator\ Register$

يستخدم هذا المسجل في معظم العمليات الحسابية وعمليات ادخال واخراج البيانات من و الى موانئ الادخال والاخراج وكذلك في عمليات نقل البيانات.

ب/ مسجل القاعدة (BX) $Base\ Register$

يستخدم هذا المسجل كمؤشر ودليل لأغراض العنونة وكذلك يستخدم في العمليات الحسابية.

ج/ مسجل العداد (CX) $Counter\ Register$

يستخدم هذا المسجل كعداد اثناء التعامل مع العمليات المراد تكرارها وكذلك يستخدم في العمليات الحسابية.

د/ مسجل البيانات (DX) $Data\ Register$

يستخدم هذا المسجل في العمليات الادخال والاخراج والعمليات الحسابية.

5/ مسجلات الدليلية $Index\ Register$

تستخدم هذه المسجلات للتعامل مع العمليات الحسابية وعمليات السلاسل الرمزية. يوجد نوعان من المسجلات الدليلية وهي:-

أ/ مسجل دليل المصدر $Source\ Index\ Register\ (SI)$

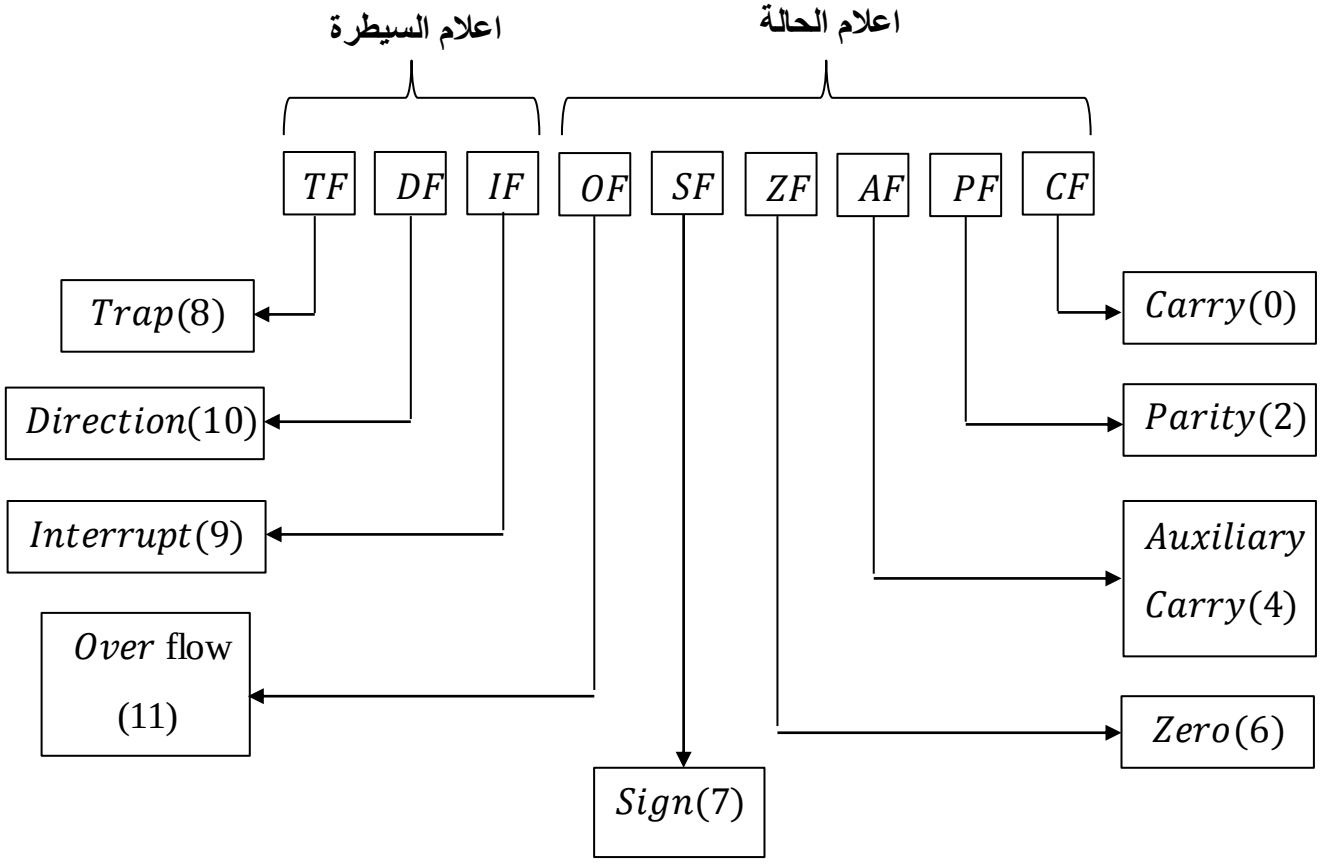
يستخدم هذا المسجل في عنونة مقطع البيانات وكمسجل مصدري في عمليات معالجة السلاسل الرمزية.

ب/ مسجل دليل الهدف $Destination\ Index\ Register\ (DI)$

يستخدم هذا المسجل في عنونة المقطع الاضافي وكمسجل هدفي في عمليات معالجة السلاسل الرمزية.

6/ مسجل الاعلام $Flay\ Register\ (FR)$

يسمى ايضاً بمسجل الحالة $Status\ Register$ ويكون من 16 بت Bit الشكل التالي يوضح الخانات الثنائية الكثر اهمية لمسجل الاعلام. لاحظ هناك 9 خانات ثنائية مهمة 6 منها تمثل اعلام الحالة وهي الحمل $Carry$ والتحقق والحمل المساعد والصفر والإشارة والفيض والتي تبين الحالة الناتجة من تنفيذ التعليمة اما 3 الخانات الباقية فهي اعلام الاتجاه والمقاطعة والتتبع.



أ - اعلام الحمل (CF) $Carry\ Flag$

يتم تمكين الخانة الثنائية الخاصة بهذا الاعلان اذا كان هناك حمل او استعارة من الخانة الثنائية الأكثر اهمية للنتيجة خلال تنفيذ التعليمة.

ب - اعلام التحقيق / Parity Flag (PF)

يتم تمكين الخانة الثنائية الخاصة بهذا الاعلام اذا كانت نتيجة تنفيذ التعليمة تحتوي على تحقق زوجي اي تحتوي على عدد زوجي من القيم الثنائية (1).

ج - اعلام الحمل المساعد (AF) $Auxiliary\ Carry\ Flay$

يتم تمكين الخانة الثنائية الخاصة بهذا الاعلام $(AF = 1)$ اذا كان هناك حمل او استعارة بين الخانة الثنائية 3 و4 للنتيجة خلال تنفيذ التعليمة.

د - اعلام الصفر (ZF) $Zero\ Flay$

يتم تمكين الخانة الثنائية الخاصة بهذا الاعلام $(ZF = 1)$ اذا كانت نتيجة تنفيذ التعليمة يساوي صفر.

هـ - اعلام الاشارة (SF) $Sgin\ Flay$

يتم تمكين الخانة الثنائية الخاصة بهذا الاعلام $(ZF = 1)$ اذا كانت نتيجة تنفيذ التعليمة هي قيمة سالبة.

ز- اعلام الفيض (OF) $Over\ flow\ Flay$

يتم تمكين الخانة الثنائية الخاصة بهذا الاعلام $(OF = 1)$اذا كان هناك فيض في نتيجة تنفيذ التعليمة.

ح - اعلام المقاطعة (IF) $Interupt\ Flay$

يتم تكمين الخانة الثنائية الخاصة بهذا الاعلام $(IF = 1)$ اذا سمح المعالج بطلب مقاطعة عند طرف الادخال $(INTER)$ عند تصغير الخانة الثنائية $(IF = 0)$ يتم اهمال طلب المقاطعة.

ط - اعلام الاتجاه (DF) $Direction\ Flay$

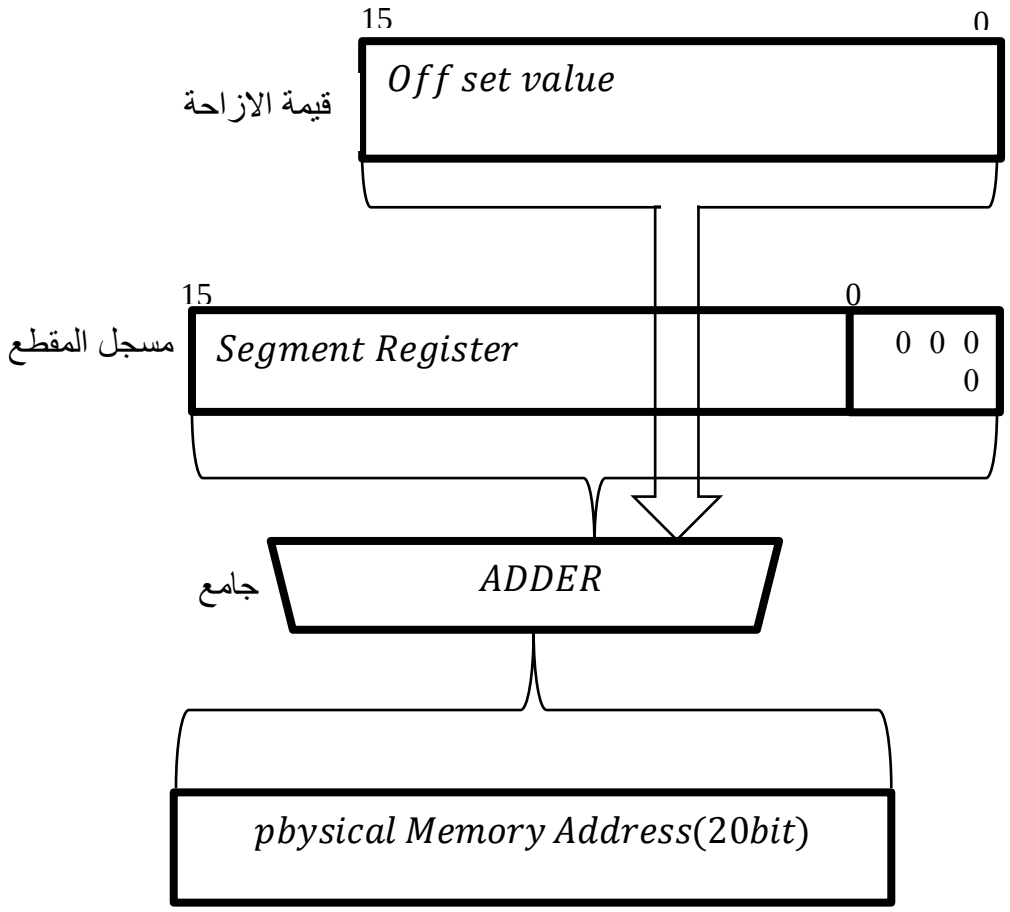
يحدد هذا الاعلام تجاه معالجة السلاسل الرمزية عند تمكين الخانة الثنائية الخاصة بهذا الاعلام $DF = 1$ يتم معالجة عناوين السلاسل الرمزية بصورة تنازلية وعندما $(DF = 0)$ يتم معالجة العناوين بصورة تصاعدية.

ي - اعلام التتبع (TF) $Trap\ Flay$

عند تمكين الخانة الثنائية الخاصة بهذا الاعلام $(TF = 1)$ يصبح المعالج في نمط الخطوة المفردة اي يقوم المعالج تنفيذ التعليمة ثم ينتقل الى البرنامج فرعي لحساب تأثير تنفيذ هذه التعليمة هذا النوع من العمليات يخدم في تنقيح البرامج.

توليد عنوان الذاكرة *Generating Memory Address*

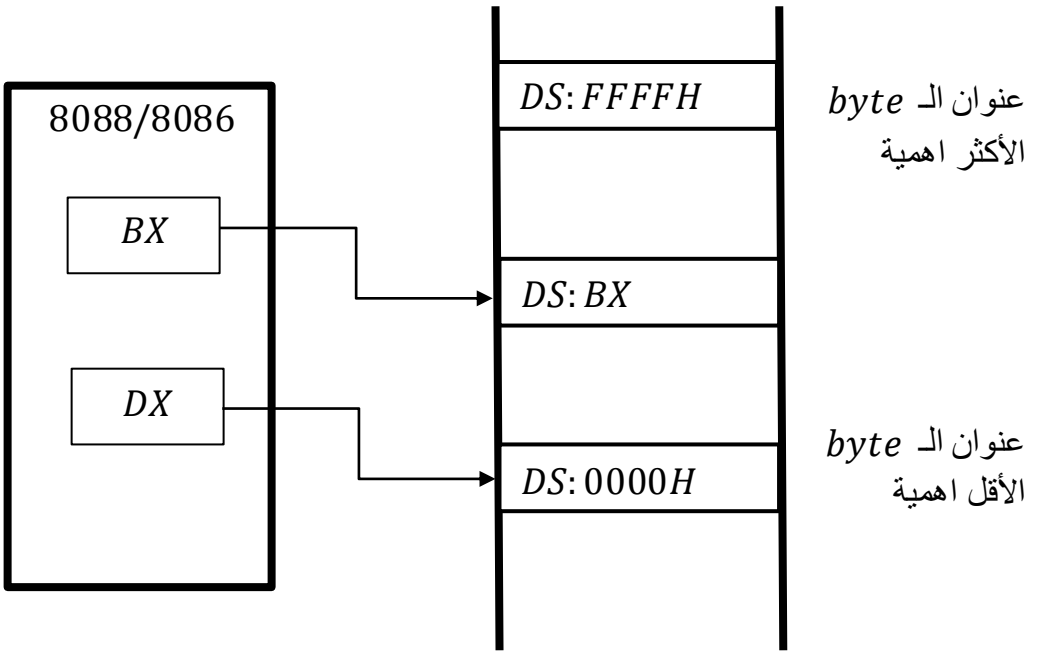
يحدد العنوان الفيزيائي للمعالج بالاعتماد على العنوان الاساس للمقطع $(Segment\ Base)$ والازاحة $(Off\ Set)$



العنوان الفيزيائي للذاكرة

العنوان الفيزيائي المستخدم للعنونة الذاكرة هو بطول $20\ Bit$ توليد العنوان الفيزيائي يتطلب تجميع قيمة الازاحة $16\ Bit$ الموجودة في مؤشر التعليمة (IP) او في مسجل القاعدة (BR) او في احد المسجلات الدليلية او مسجلات التأشير مع قيمة المقطع الاساس $16\ Bit$ الموجودة في احد مسجلات المقاطع.

عنوان المقطع الاساس / يمثل عنوان البداية بالنسبة للمقطع في الذاكرة علماً بان حجم كل مقطع هو $64Kbyte$ الشكل التالي يوضح بأن الازاحة تمثل المسافة بـ $byte$ لموقع التخزين عن عنوان بداية المقطع.



الفيزيائي العنوان $=$ مسجل المقطع $\times 10_{16}$ $+$ الازاحة مقدار

$$Pbysical\ address = Segment\ Register\ *\ 10_{16} + Offset$$

Ex / اوجد القيم المجهولة لكل من العناوين الفيزيائية التالية:-

	$Segment$	$off\ set$	$physical\ address$
1	$A000\ H$	?	$A0123\ H$
2	?	$14DA\ H$	$235DA\ H$
3	$D765\ H$	?	$DABC0\ H$
4	$0100\ H$	$ABCD\ H$	?
5	$B2C0\ H$	$FA12\ H$	?

الحل/

$$Pbysical\ address = Segment\ Register\ *\ 10_{16} + Offset$$

1/ $A0123\ H = A000\ H * 10 + offset$

$\therefore offset = A0123\ H - A0000H = 0123\ H$

2/ $235DA\ H = segment * 10 + 14DA$

$\therefore\ segment * 10 = 235DA\ H - 14DA\ H = 22100\ H$

$\therefore\ segment = 2210\ H$

3/ $DABC0\ H = D765\ H * 10 + offset$

$\therefore\ offset = DABC0\ H - D7650\ H = 3270\ H$

4/ $Pbysical\ address = 0100\ H * 10 + ABCD\ H$

$\therefore Pbysical\ address = 01000\ H + ABCD\ H = BBCD\ H$

5/ $Pbysical\ address = B2C0\ H * 10 + FA12\ H$

$\therefore Pbysical\ address = B2C00\ H + FA12\ H = C2612$

Ex / اوجد قيم المجهول لكل من العناوين الفيزيائية التالية:-

	$Segment$	$off\ set$	$physical\ address$
1	$B002\ H$	?	$B0123\ H$
2	?	$11C0\ H$	$2AB00\ H$
3	$1D2A\ H$	$AC3B\ H$	?

الحل/

$Pbysical\ address = Segment\ Register\ *\ 10_{16} + Offset$

1/ $B0123\ H = B002\ H * 10 + offset$

$\therefore offset = B0123\ H - B0020\ H = 0103$

2/ $\text{segment} * 10 = 2\text{AB00 H} - 11\text{C0 H} = 29940$

$\therefore segment = 2994$

3/ $Pbysical\ address = 1D2A\ H * 10 + AC3B\ H = 1D2A0\ H + AC3B\ H$

$Pbysical\ address = 27EDB$

الذاكرات $Memories$

يمكن اعتبار ذاكرات الحاسبة بانها تلك المكونات القادرة على خزن المعلومات واسترجاعها حيث تستجيب هذه الذاكرات للأوامر التي غالباً ما تكون بإحدى الصيغتين التاليين:-

$1/\ load\ from$

$2/\ store\ to$

الوحدة الاساسية لأي ذاكرة هي الخلية والتي تمثل الوحدة القياسية لتناقل البيانات فيما بين المعالج والذاكرة او بقية الأجزاء المكونة للحاسب على وسائل النقل التي تعرف بالمسارات. هناك ميزتان اساسيتان لكل خلية ذاكرة هما عنوان الخلية ومحتوياتها يمكن التمييز بين الذاكرات بطريقتين اساسيتين:-

1/ التمييز بين الذاكرات من حيث الوظيفة :حيث تقسم الى

أ- **ذاكرة اولية** : نقصد بالخزن الاولى (الرئيسي) هو ذلك الخزن الذي يمكن المعالج من الوصول مباشرتاً الى التعليمات والبيانات.

ب- **ذاكرة ثانوية** : هو ذلك الخزن الذي يمكن الوصول اليه من خلال ايعازات الخزن والاسترجاع بتحديد اجهزة خزن دائميه.

2/ التمييز بين الذاكرات حيث طريقة الوصول.

أ- **الذاكرة الوصول العشوائي**: في هذا النوع من الوصول يمكن للمعالج ان يصل مباشرتاً الى موقع محدد دون طلب الوصول الى المواقع البينية.

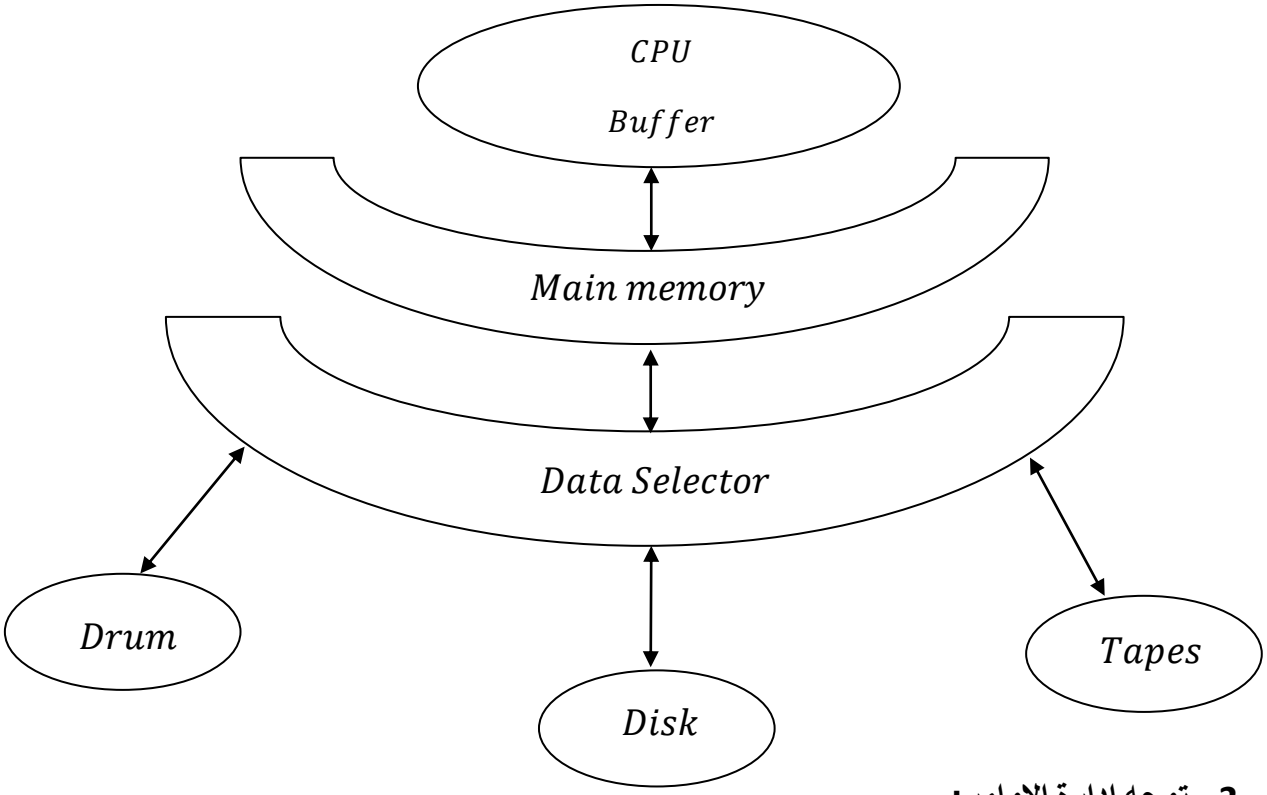
ب- **الذاكرات الوصول المتسلسل**: في هذا النوع يكون الوصول فقط الى موقع السابق او اللاحق للمؤشر الحالي اي اذا كان الموقع هو n فانه يمكن الوصول منه اما الى الموقع $(n-1)$ او الموقع $(n+1)$.

أدارة الذاكرة $Memory\ Management$

ان اداء اي حاسبة يعتمد بشكل رئيسي على عاملين مهمين هما السرعة والكلفة.

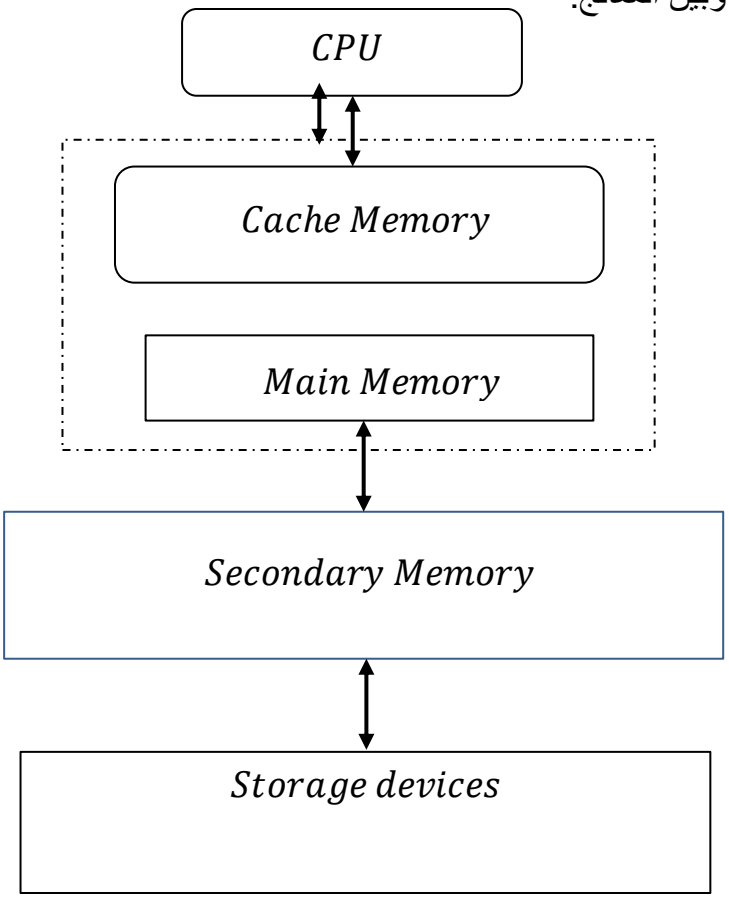
السرعة هي قابلية المعالج على تنفيذ العمليات المخزونة في مواقع الذاكرة فكلما كانت سرعة التنفيذ عاليا نسبياً كلما كانت الحاسبة اكثر ثمناً ولهذا السبب يجب تنظيم عملية الوصول الى البيانات والبرامج في الذاكرة لضمان الأداء الأمثل للحاسبة هذا التنظيم يعرف بإدارة الذاكرة. يوجد توجهان لإدارة الذاكرة

1- **توجه ادارة البيانات** :- اثناء التنفيذ يحصل وقت يمثل فجوة في اداء المعالج وهذه الفجوة تمثل الوقت المستغرق للحصول على البيانات وانتقالها بين المعالج والذاكرة هذا الوقت يؤدي الى الفصل بين زمن الوصول وكلفة تخزين البيانات حيث يمكن للمستخدم من الوصول مباشراً الى كمية محدودة من البيانات المخزنة عشوائيا بينما لكي نصل الى وحدات التخزين الدائمة على المستخدم ان يطلب من منتقي البيانات ان يخصص الوحدة التخزينية الحاوية على بيانات المطلوبة.

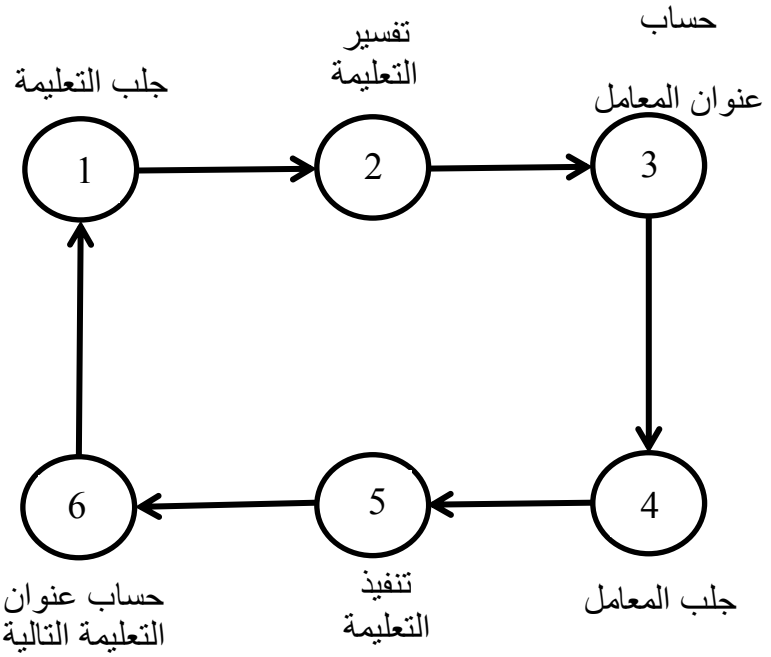


2- توجه ادارة الاوامر:-

يمكن تصميم نظام نقل البيانات بين المعالج والذاكرات بأكمله تحت سيطرة خوارزميات مبنية داخلياً لذلك فأن المستخدم يحتاج الى معلومات مسبقة عن توزيع البيانات بين مختلف المستويات في الذاكرة المستخدمة لتوفير هذا الاحتياج تم استخدام ما سمي بالذاكرة المخبئة $Cache\ Memory$ التي توفر السرعة المطلوبة بين الذاكرة الرئيسية وبين المعالج.



تنفيذ التعليمة:-



يتطلب تنفيذ أي تعليمة من قبل المعالج اجراء الخطوات التالية:-

1- وحدة المعالجة المركزية تطلب التعليمة من الذاكرة.
2- وحدة المعالجة المركزية تفسر التعليمة.
3- اعتماد على الخطوة 2 ينجز التالي:-

أ - جلب المعاملات من الذاكرة وتخزينها في سجل وحدة الحساب والمنطق مع اعطاء السيطرة الى وحدة الحساب والمنطق لكي تنفذ التعليمة.

ب - تخزين ناتج تنفيذ وحدة الحساب والمنطق.

ج - وحدة الادخال والاخراج تنقل الناتج الى خارج النظام.

4- عند انتهاء الخطوة 3 يعاد تكرار الخطوة 1.

المسارات (نوقل) $Buses$:-

تتم عملية نقل البيانات بين المعالج والمكونات المختلفة للحاسب عن طريق مجموعة من النواقل تعرف بالمسارات وهي على 3 انواع.

1- مسارات السيطرة $Control\ Bus$:-

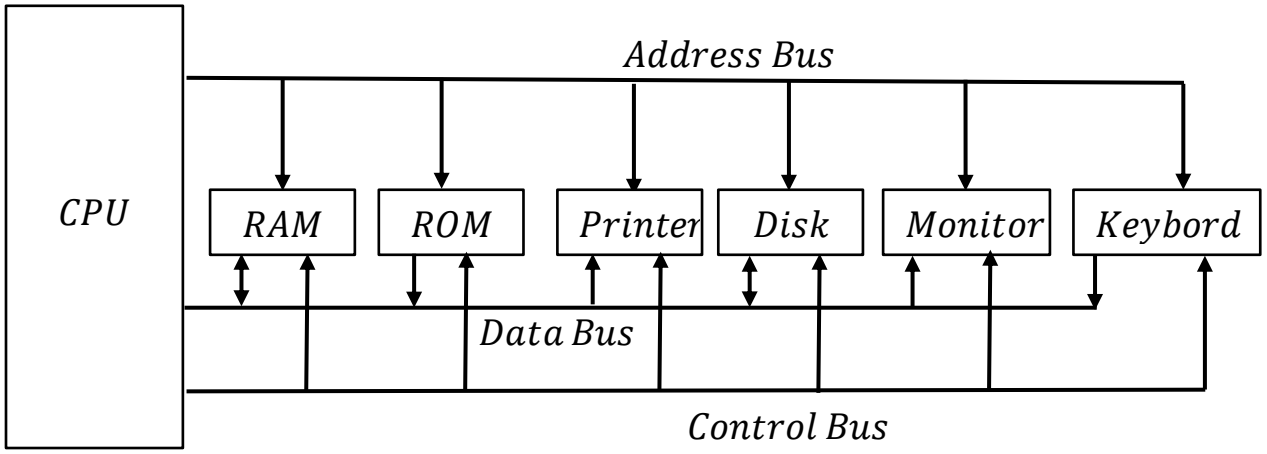
يحدد هذا المسار رغبة وحدة المعالجة المركزية بأرسال او استلام البيانات من والى اجهزة المكونة للحاسب وذلك عن طريق تعليمات القراءة والكتابة.

2- مسار البيانات $Data\ Bus$:-

يقوم هذا المسار بنقل البيانات من والى وحدة المعالجة المركزية فكلما كان عدد المسارات اكثر كلما زادت كفاءة وسرعة المعالج.

3- مسار العناوين $Address\ Bus$:-

يحدد هذا المسار الجهاز او موقع الذاكرة الذي يتصل في وحدة المعالجة المركزية فكلما زادت عدد خطوط هذا المسار كلما زاد حجم الذاكرة او عدد الاجهزة التي يمكن ان تستخدم من قبل المعالج.



مقدمة الى البرمجة بلغة التجميع:-

تعمل وحدة المعالجة المركزية بنظام الثنائي وبسرعة عالية جداً ان برمجة الحاسبات بالنظام الثنائي تعتبر عملية شاقة وبطيئة يصعب على الانسان القيام فيها, يسمى البرنامج المكون من شفرات ثنائية بلغة الالة $machine\ language$ لقد قام المبرمجون في بداية الامر برمجة الحاسبات بهذه اللغة وعلى الرغم من استخدام النظام السداسي عشر كطريقة فعالة في تنفيذ لغة الالة الا ان هذا النظام مرهق وشاق ومن الصعب استخدامه واخيراً تم تطوير لغة تعرف بلغة التجميع $Assembly\ Languague$ تعطي هذه اللغة اسماء رمزية لشفرات لغة الالة الخاصة بالتعليمات وهذا ما اعطى للمبرمجين امكانية كتابة برمجهم بشكل اسرع وتعرض اقل للأخطاء تتم عملية ترجمة برنامج لغة التجمع الى لغة الالة من قبل برنامج اخر يسمى بالمجمع $Assembler$.

كما يشار الى اللغة التجمع بانها لغة واطئة المستوى $Low\ Level\ Languague$ وكونها تتعامل مباشراً مع البنية الداخلية للمعالج لكي نستطيع البرمجة بلغة التجميع فأننا لابد ان نكون على معرفة بعدد المسجلات واحجامها وتفاصيل وحدة المعالجة المركزية والمعالج. يتوفر حالياً العديد من لغات البرمجة التي يمكن استخدامها مثل لغة باسكال وبيسك ولغة سي هذه اللغات تسمى بالغات العالية المستوى $High\ Level\ Languague$ وذلك لأن المبرمج لا يحتاج الى معرفة البنية الداخلية للمعالج لترجمة البرنامج المكتوب بلغة عالية المستوى الى لغة الالة يستخدم برنامج المترجم $Compiler$ فعلى سبيل المثال لكي نستطيع كتابة برنامج بلغة باسكال لابد من توفر مترجم يقوم بتحويل هذا البرنامج الى لغة الالة.

البرمجة بالغة التجميع:-

يتكون برنامج لغة التجميع من عدد من الاسطر تمثل مجموعة من تعليمات هذه اللغة تتكون تعليمات لغة التجميع من الاسم الرمزي الذي قد يكون متبوع بمعامل واحد او اكثر تمثل معاملات البيانات التي سيتم معالجتها ام الاسم الرمزي فهو الامر الذي يوجه الى المعالج لأخباره عن العملية التي تجري على البيانات. تعتبر عملية نقل البيانات بين مختلف مكونات الحاسب من اهم العمليات لغة التجميع حيث تتم هذه العملية باستخدام التعليمة MOV تستخدم هذه التعليمة لنقل البيانات بين المعاملات وتأخذ الصيغة التالية

$$\text{MOV}\ D,S$$

فعلى سبيل المثال التعليمة ($\text{MOV}\ DX,AX$) تقوم بنقل محتويات المسجل AX الى المسجل DX هناك مجموعة من الحالات التي يجب تجنبها عند التعامل مع التعليمة MOV وهي:

1/ لا يسمح بالنقل المباشر للبيانات الى مسجلات المقاطع $\{CS,DS,ES,SS\}$ لكي نقوم بهذا الامر يجب تحميل القيم من مسجل اخر ثم تنقل الى مسجلات المقاطع.

2/ اذا تم تحميل قيمة اقل من $1Bit$ في مسجل بطول كلمة فأن بقية الخانات الفارغة من الكلمة تملأ بالأصفار مثلاً التعليمة ($\text{MOV}\ BX,05H$) ستجعل قيمة المسجل BX $(BX = 0005H)$.

3/ ان تحميل المسجلات بقيم اكبر من حجمها سيؤدي الى حدوث خطأ بالتعليمة هي تعليمة خاطئة $\Leftarrow MOV\ \underbrace{BL}_{\text{بايت}},\underbrace{73D}_{\text{اكثرمن بايت}}H$

الموجهات $Directives$

ان اي برنامج مكتوب بلغة التجميع يتم ترجمته بواسطة المجمع وهذا البرنامج هو عبارة عن سلسلة من العبارات التي تكون اما تعليمات مثل MOV, ADD او عبارات تعرف بالموجهات والتي تعطي توجيهات الى المجمع عن كيفية ترجمة تعليمات لغة التجميع الى لغة الألة.

تتألف التعليمات في لغة التجميع من اربعة حقول اساسية

$$[Label:]Name[Operands][;Comments]$$

تشير الاقواس الى ان هذه الحقول هي حقول اختيارية عند كتابة التعليمة.

- حقل العنوان $Label$: يستخدم هذا الحقل كمؤشر لسطر محدد من البرنامج من خلال اعطاءه اسم معين.
- حقل الاسم الرمزي $Name$: يمثل هذا الحقل اسم التعليمة المراد تنفيذها مثل (MOV, ADD).
- المعاملات $Operands$: يمثل هذا الحقل المعاملات المراد اجراء التعليمات عليها وهي قد تكون اسم مسجل او عنوان موقع ذاكرة او قيمة فورية تذكر ضمن التعليمة.
- حقل ملاحظات $Comment$: يبدأ هذا الحقل بفاصلة منقوطة ويمثل بعبارة تصف وظيفة التعليمة الحالية وهي عبارة غير قابلة للتنفيذ قد تكون في نهاية السطر او في سطر مستقل.

Ex/ البرنامج التالي برنامج في لغة التجميع يحتوي على مجموعة من الموجهات ويستخدم لإيجاد مجموع قيمتين عدديتين من خلال استخدام مجموعة تعليمات النقل والجمع

```
;Simple program for Summing two Numbers

• Model small  <——— حجم الذاكرة التي تستخدم لهذا البرنامج
• stack 64
• Data
• Data1 DB 52H
• Data2 DB 29H
• Sum
   • Code
  Main Proc Far
     Mov  Ax, @ Data
     Mov  Ds, Ax
     Mov  BL,Data1
     Mov  BL, Data2
     Add  AL,BL
     Mov  Sum,AL
     Main End P
          End Main
```

اول عبارة هي عبارة ملاحظات للتعريف للبرنامج. العبارة الثانية هي لتوجيه $(Model)$ الذي يحدد حجم الذاكرة التي تخصص للبرنامج يأخذ هذا التوجيه احد الاختيارات التالية:-

$Name$	$Data$	$Code$
$Small$	$64\ KB$	$64\ KB$
$Medium$	$64\ KB$	$> 64KB$
$Compact$	$> 64KB$	$64\ KB$
$Large$	$> 64KB$	$> 64KB$

يمتلك المعالج اربعة مسجلات مقاطع (مقطع البيانات ومقطع التعليمات ومقطع المكدس والمقطع الإضافي) يجب على كل سطر من اسطر التعليمات ان يتبع احد هذه المقاطع فمثلاً التوجيه(Stack•) يتبع المقطع المكدس والتوجيه $(Data•)$ يتبع لمقطع البيانات والتوجيه $(Code•)$يتبع المقطع التعليمات . يمكن للبرنامج الواحد ان يحتوي على اكثر من مقطع واحد.

مقطع البيانات يعرف بثلاث متغيرات $(Data1, Data2, Sum)$ وكل منها معرف من خلال التوجيه DB اي انها من النوع بايت $Byte$.

انماط العنونة $Addressing\ Modes$

تمثل انماط العنونة طرق الوصول الى البيانات التي تستخدم مع التعليمات قد تكون هذه البيانات موجودة بصورة فورية ضمن التعليمة او مخزونة في احد مسجلات المعالج او ضمن مواقع الذاكرة, سوف تستخدم التعليمة MOV لتوضيح انماط العنونة والتي تشتمل.

1/ العنونة بالمسجلات $Register\ Addressing\ Mode$

تستخدم لنقل بايت او كلمة من مسجل المصدر الى مسجل الهدف مثل

MOV $AX\ ,BX$

MOV $CL\ ,AL$

2/ العنونة الفورية $Immediate\ Addressing\ Mode$

تستخدم لنقل البايت او الكلمة المصدر الموجودة ضمن التعليمة الى المسجل الهدف مثل

MOV $AX, 1234H$

MOV $AL.32H$

اكثر من من رقمين فهو $Word$ اما رقمين فهو بايت.

3/ العنونة المباشرة *Direct Addressing Mode*

تستخدم لنقل بايت او كلمة بين احد مسجلات المعالج وبين مواقع الذاكرة المعنونة بالقيمة المذكورة ضمن التعليمة مثل/

Mov AX , $[325DH]$

MOV $[44C2H], BL$

الاقواس المربعة تشير الى موقع ذاكرة.

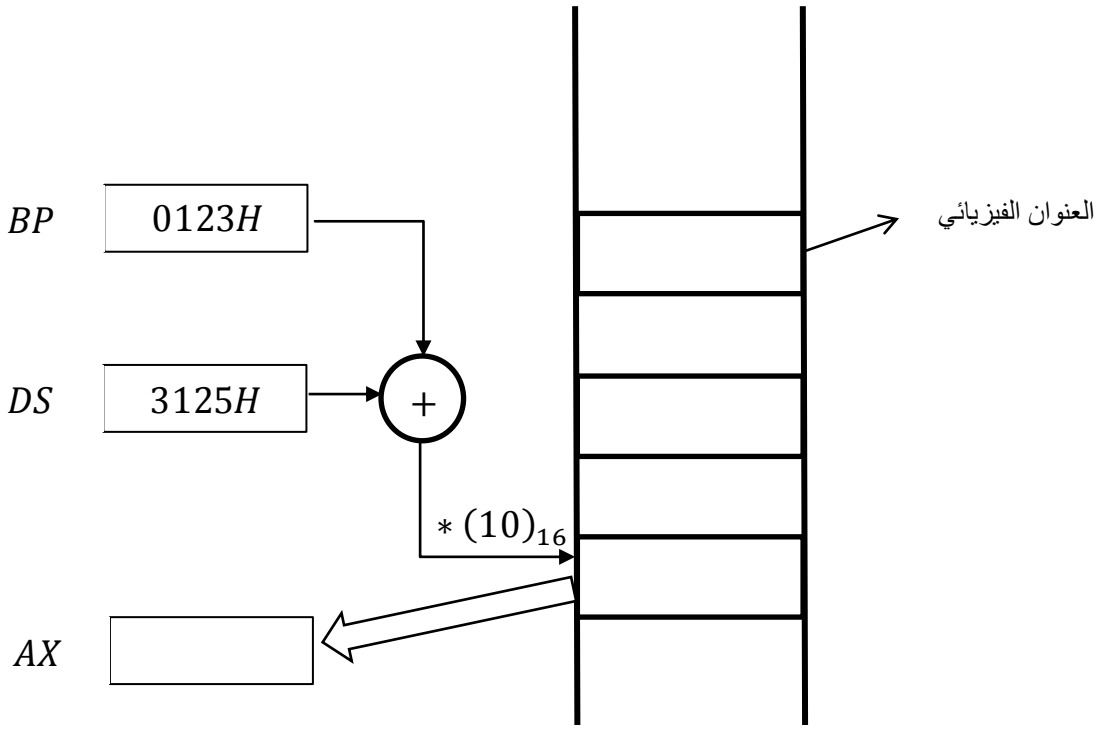
4/ العنونة غير المباشرة بالمسجلات *Register Indirect Addressing Mode*

تستخدم لنقل بايت او كلمة بين احد مسجلات المعالج وبين مواقع الذكرة المعنونة بالقيمة الموجودة في احد المسجلات الدليلية $(Index\ Register:\ SI, DI)$

او مسجلات القاعدية $(Base\ Register:\ BX, BP)$ مثل

MOV AX , $[BP]$

MOV $[SI], AL$

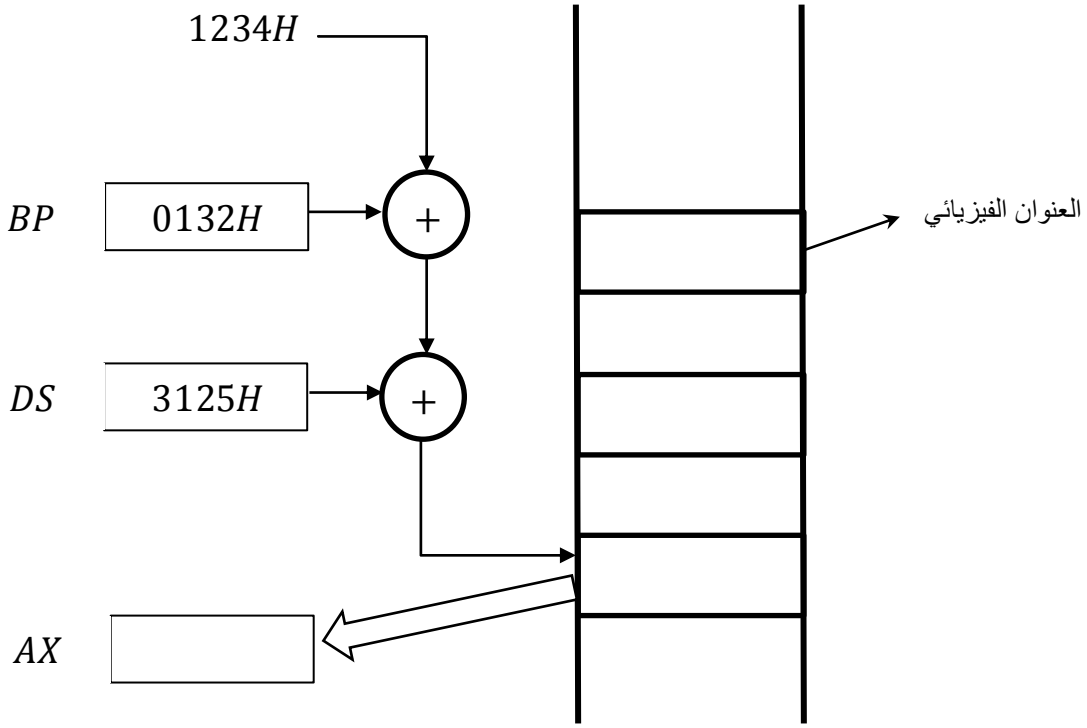


5/ العنونة القاعدية ***Based Addressing mode***

تستخدم لنقل بايت او كلمة بين احد مسجلات المعالج وبين مواقع الذاكرة المعنونة بالقيمة الموجودة في احد مسجلات القاعدة $(Base\ Register: BX, BP)$ زائداً قيمة الازاحة ضمن التعليمة مثل

$\text{MOV}\ AX, [BP] + 1234H$

$\text{MOV}\ [BX] + 12H, AL$



6/ العنونة الدليلية ***Index Addressing Mode***

تستخدم لنقل بايت او كلمة بين احد مسجلات المعالج وبين مواقع الذاكرة المعنونة بالقيمة الموجودة في احد المسجلات الدليلية $(Index\ Register: SI, DI)$ زائداً قيمة الازاحة ضمن التعليمة مثل

$\text{MOV}\ AX, [SI] + 1234H$

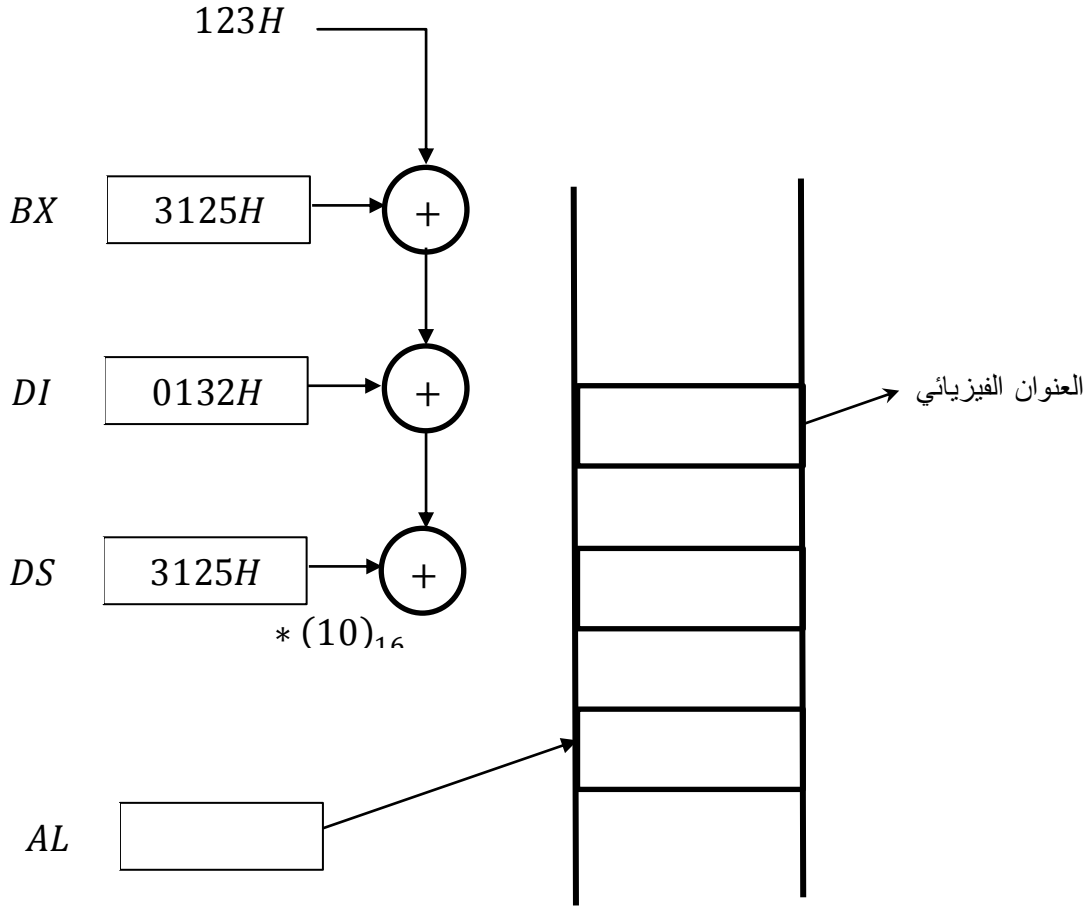
$\text{MOV}\ [DI] + 13H, AL$

7/ العنونة الدليلية القاعدية $Based - Index\ Addressing\ Mode$

تستخدم لنقل بايت او كلمة بين احد مسجلات المعالج وبين مواقع الذاكرة المعنونة بالقيمة الموجودة في احد المسجلات القاعدة (BX, BP) وأحد المسجلات الدليلية (SI, DI) زائد قيمة الازاحة ضمن التعليمة مثل

$$\text{MOV } AX, [BP][SI] + 1234H$$

$$\text{MOV } [BX][DI] + 123H, AL$$



Ex / اذكر نمط العنونة المستخدم لكل من التعليمات التالية

$1/\ \text{MOV } AL, BL$

$2/\ \text{MOV } AX, 0325H$

$3/\ \text{MOV } [DI], AX$

$4/\ \text{MOV } DI, [SI]$

$5/\ \text{MOV } [BX] + 0400H, CX$

$6/\ \text{MOV } [DI] + 0400H, AH$

$7/\text{MOV } [BX][DI] + 0400H, AL$

الحل/

1/ بالمسجلات 2/ الفورية 3/ غير مباشرة 4/ غير مباشرة 5/ قاعدية 6/ الدليلية 7/ دليلية قاعدية

Ex / اذكر نمط العنونة للتعليمات الصحيحة الموجودة ضمن التعليمات التالية

1/ MOV AX , BL

2/ MOV $AX, 0FFH$

3/ MOV $CL, 1FDH$

4/ MOV $[1234H], AX$

5/ MOV $DS, 1234H$

6/ MOV $[BP] + 0400H, CL$

7/ MOV $[DI]$ $[BX] + 0400H, DX$

1/ خطا 2/ فورية 3/ خطا 4/ مباشرة 5/ خطا (لا يجوز تخزين البيانات مباشرة في مسجلات

6/ قاعدية 7/ دليلية قاعدية

تحويل ايعازات لغة التجميع الى لغة الالة

Converting Assembly Language Instructions to Machine Code

لتحويل برنامج لغة التجميع الى لغة الالة يجب تحويل كل تعليمة من تعليمات لغة التجميع على حدة الى تعليمات لغة الالة حيث تعتمد عملية التحويل على الشكل التالي.

يحتوي الـ $byte1$ لنموذج التحويل على الحقول التالية

/1 $OP\ Code\ (16bit)$

يمثل الشفرة الخاصة بالاسم الرمزي للتعليمة $(MOV, ADD, \dots)$

/2 $D\ (1bit)$

تحدد هل المسجل (REG) هو المصدر ام الهدف (اذا المصدر $D = 0$)

/3 $W\ (1bit)$

يحدد هل العملية تطبق على معاملات بطول $(8bit)$ او بطول $(16bit)$. $(if\ 8bit\ W = 0)$

فمثلا التعليمة $(ADD\ AX, BX)$ شفرة الاسم الرمزي ADD هي $OP\ Code = 000000$ فاذا كان المسجل AX هو REG $(REG = AX)$ وموقعه هدف بالنسبة للتعليمة تكون $(D = 1)$ وبما ان التعليمة تطبق على معاملات بطول $(16bit)$ اذن تكون $(W = 1)$.

يحتوي $Byte\ 2$ لنموذج التحويل على الحقول التالية

/1 $REG\ (3bit)$:-

يمثل الشفرة الخاصة بالمسجل REG والجدول التالي يوضح الشفرة الخاصة في كل مسجل من مسجلات المعالج بالاعتماد على قيمة W

REG	$W = 1$	$W = 0$
000	AL	AX
001	CL	CX
010	DL	DX
011	BL	BX
100	AH	SP
101	CH	BP
110	DH	SI
111	BH	DI

جدول الشفرة REG لمسجلات المعالج

$MOD\ (2bit)$ /2

يحدد هل المعامل الثاني في التعليمة هو مسجل او موقع ذاكرة فاذا كان مسجل يكون $MOD = 11$

$MOD = 11$			طريقة العنونة للذاكرة			
$\boldsymbol{R/M}$	$W = 0$	$W = 1$	$\boldsymbol{R/M}$	$MOD = 00$	$MOD = 01$	$MOD = 10$
000	AL	AX	000	$[BX] + [SI]$	$[BX] + [SI] + D_8$	$[BX] + [SI] + D_{16}$
001	CL	CX	001	$[BX] + [DI]$	$[BX] + [SI] + D_8$	$[BX] + [DI] + D_{16}$
010	DL	DX	010	$[BP] + [SI]$	$[BX] + [SI] + D_8$	$[BP] + [SI] + D_{16}$
011	BL	BX	011	$[BP] + [DI]$	$[BX] + [SI] + D_8$	$[BP] + [DI] + D_{16}$
100	AH	SP	100	$[SI]$	$[\mathrm{SI}] + D_8$	$[SI] + D_{16}$
101	CH	BP	101	$[DI]$	$[DI] + D_8$	$[DI] + D_{16}$
110	DH	SI	110	$Direct\ address$	$[BP] + D_8$	$[BP] + D_{16}$
111	BH	DI	111	$[BX]$	$[BX] + D_8$	$[BX] + D_{16}$

جدول الشفرة (R/M) والشفرة (MOD) للمعامل الثاني في التعليمة

$R/M(3bit)$ /3

تحدد الشفرة الخاصة بالمعامل الثاني هل هي موقع ذاكرة او مسجل بالاعتماد على قيمة (MOD).

فمثلاً بالنسبة الى $2byte$ للتعليمة السابقة بما ان المعامل الثاني هو مسجل BX اذن تكون $MOD = 11$ وتكون قيمة $R/M = 011$.

- دائما نعتبر المسجلات $(AX, DX, CX, \dots)$ هي REG.
- دائما نعتبر المسجل $(AL, , AH, AX, \dots)$ هو REG.
- L او $H \leftarrow Byte$.
- X او P او $I \leftarrow Word$.

Ex/ مثل الايعاز التالي الى شفرة الالة. $ADD\ \ AX\ ,\ BX$

0	0	0	0	0	0	1	1	1	1	0	0	0	0	1	1
$OP\ Code$						D	W	MOD		REG			R/M		

$Byte\,1 = (00000011)_2 = (03)_{16}$

$Byte\,2 = (11000011)_2 = (C3)_{16}$

$\therefore\ ADD\ \ AX\ , BX = 0C03\ H$

Ex/ مثل الايعاز التالي بشفرة الالة $MOV\ \ [BX][SI]\ , CX$ علماً بأن $(MOV = 100010)$

1	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0
$OP\ Code$						D	W	MOD		REG			R/M		

$Byte\,1 = (10001001)_2 = (89)_{16}$

$Byte\,2 = (00001000)_2 = (08)_{16}$

$\therefore\ MOV\ \ [BX][SI]\ , CX = 8908\ H$

Ex / مثل الايعاز التالي بشفرة الالة $MOV\ \ BL\ , AL$

1	0	0	0	1	0	0	0	1	1	0	0	0	0	1	1
$OP\ Code$						D	W	MOD		REG			R/M		

$Byte\,1 = (10001000)_2 = (88)_{16}$

$Byte\,2 = (11000011)_2 = (C3)_{16}$

$\therefore\ MOV\ \ [BX][SI]\ , CX = 88C3\ H$

تحويل ايعازات لغة الالة الى لغة التجميع

Converting Assembly Language Instructions to Machine Code

Ex/ مثل الايعاز بلغة التجميع المكافئ لشفرة الألة $(0304H)$ علماً بأن $ADD = 000000$

$byte1 = (03)_{16} = (00000011)_2$

$byte2 = (04)_{16} = (00000100)_2$

$byte1$								$byte2$							
0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0
$OP\ Code$						D	W	MOD		REG			R/M		

$OP\ Code = 000000 \implies ADD$

$REG = 000\ ,\ W = 1\ , D = 1$ $\implies$ المعامل الاول AX وهو هدف

$R/M = 100\ ,\ MOD = 00$ $\implies$ المعامل الثاني $[SI]$ وهو مصدر

$\therefore (0304)_{16} = ADD\ \ AX\ , [SI]$

Ex/ مثل الايعاز بلغة التجميع المكافئ لشفرة الألة $8915H$ علماً بأن $MOV = 100010$

$byte1 = (89)_{16} = (10001001)_2$

$byte2 = (15)_{16} = (00010101)_2$

$byte1$								$byte2$							
1	0	0	0	1	0	0	1	0	0	0	1	0	1	0	1
$OP\ Code$						D	W	MOD		REG			R/M		

$OPCode = 100010 \implies MOV$

$REG = 010\ , W = 1\ , D = 0 \implies$ المعامل الاول DX وهو مصدر

$R/M = 101 , MOD = 00 \implies$ المعامل الثاني $[DI]$ وهو هدف

$\therefore (8915)_{16} = MOV\ \ [DI]\ , DX$

- يقوم بعملية نقل محتويات المسجل DX الى مواقع الذاكرة المعنونة بمحتوى المسجل $[DI]$ والموقع الذي يليه.

Ex/ مثل الايعاز التالي بشفرة الألة $ADD\ \ CX\ , BX$ علماً بأن $ADD = 000000$

$byte1$								$byte2$							
0	0	0	0	0	0	0	1	1	1	0	1	1	0	0	1
$OP\ Code$						D	W	MOD		REG			R/M		

$byte1 = (00000001)_2 = (01)_{16}$

$byte2 = (11011001)_2 = (D9)_{16}$

$\therefore ADD\ \ CX\ , BX = (01D9)_{16}$

Ex/ مثل الإيعاز بلغة التجميع المكافئ لشفرة الألة $8908H$ علماً بأن $MOV = 100010$

$byte1 = (89)_{16} = (10001001)_2$

$byte2 = (08)_{16} = (00001000)_2$

$byte1$								$byte2$							
1	0	0	0	1	0	0	1	0	0	0	1	1	0	0	0
$OP\ Code$						D	W	MOD		REG			R/M		

$OPCode = 100010 \implies MOV$

$REG = 001 , W = 0 , D = 1 \implies$ المعامل الاول CX وهو مصدر

$R/M = 000 , MOD = 00 \implies$ المعامل الثاني $[BX][SI]$ وهو هدف

$\therefore MOV \ [BX][SI] , CX = (8908)_{16}$

Ex/ جد الايعاز بلغة التجميع المكافئ لشفرة الألة $03C3H$ علماً بأن $ADD = 000000$

$byte1 = (03)_{16} = (00000011)_2$

$byte2 = (C3)_{16} = (11000011)_2$

$byte1$								$byte2$							
0	0	0	0	0	0	1	1	1	1	0	0	0	0	1	1
$OP\ Code$						D	W	MOD		REG			R/M		

$OPCode = 000000 \implies ADD$

$REG = 000 \ , \ W = 1 \ , \ D = 1 \implies$ المعامل الاول هو AX وهو هدف

$R/M = 011 , MOD = 11 \implies$ المعامل الثاني هو BX وهو مصدر

$\therefore ADD \ AX , BX = (03C3)_{16}$

تعليمات لغة التجميع $Transfer\ Instructions$

1- التعليمة MOV

الصيغة العامة

$$MOV\ Destination\ ,Source$$

بحيث /

$Destination$ تمثل تعليمة الهدف.

$Source$ تمثل تعليمة المصدر.

- ملاحظة/ الحالات التي يجوز النقل فيها.

$D.$ المصدر	$\longrightarrow$	$S.$ الهدف
$R.$ مسجل	$\longrightarrow$	$R.$ مسجل
$R.$ مسجل	$\longrightarrow$	$M.$ موقع ذاكرة
$M.$ موقع ذاكرة	$\longrightarrow$	$R.$ مسجل
$R.$ مسجل	$\longrightarrow$	$Immediatly$ فوري
$M.$ موقع ذاكرة	$\longrightarrow$	$Immediatly$ فوري

- ملاحظة/ لا يجوز النقل من ذاكرة $M.$ الى ذاكرة $M.$

Ex/ اكتب برنامج بلغة التجميع لنقل محتويات الذاكرة من $(44400\,H \longrightarrow 44404\,H)$ الى مواقع الذاكرة $(4440A\,H \longrightarrow 4440E\,H)$ علماً بأن المسجل $DS = 4400\,H$

/Sol

$$Pbysical = Segment * 10 + Offset$$

$$44400 = 4400 * 10 + Offset$$

$$Offset = 44400 - 44000 = 400$$

$$44404 = 4400 * 10 + Offset$$

$Offset = 44404 - 44000 = 404$

$4440A = 4400 * 10 + Offset$

$Offset = 4440A - 44000 = 40A$

$4440E = 4400 * 10 + Offset$

$Offset = 4440E - 44000 = 40E$

$MOV\ AX, [400]$ ينقل التعليمة 400 و 401 الى AX

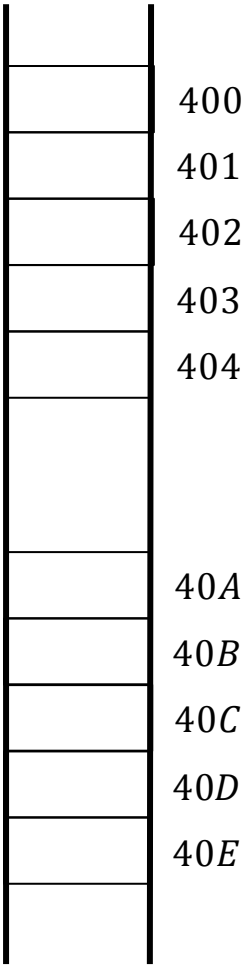
$MOV\ [40A], AX$ ينقل من AX الى $40A$ و $40B$

$MOV\ , [402]$ و هكذا

$MOV\ [40C], AX$

$MOV\ AL\ , [404]$

$MOV\ [40E], AL$



2- التعليمة $XCHG$

الصيغة العامة

$XCHG\ \ Destination\ ,Source$

بحيث/

$Destination$ تمثل تعليمة الهدف.

$Source$ تمثل تعليمة المصدر. الحالات التي يجوز النقل فيها

$S.$ الهدف	$\longrightarrow$	$D.$ المصدر
$R.$ مسجل	$\longrightarrow$	$R.$ مسجل
$M.$ موقع ذاكرة	$\longrightarrow$	$R.$ مسجل
$R.$ مسجل	$\longrightarrow$	$M.$ موقع ذاكرة

Ex/ اكتب برنامج بلغة التجميع لتبديل محتويات مواقع الذاكرة ($60010\ H \longrightarrow 60013\ H$) من اعلى الى اسفل او من اسفل الى اعلى باستخدام تعليمة $XCHG$ فقط علماً بأن المسجل $DS = 6000\ H$

$Pbysical = Segment * 10 + Offset$

$60010 = 6000 * 10 + Offset$

$Offset = 60010 - 60000 = 10\ H$

$60011 = 6000 * 10 + Offset$

$Offset = 60011 - 60000 = 11\ H$

$60012 = 6000 * 10 + Offset$

$Offset = 60012 - 60000 = 12\ H$

$60013 = 6000 * 10 + Offset$

$Offset = 60013 - 60000 = 13\ H$

$XCHG\ \ AL\ ,[10]$

$XCHG\ \ AL\ ,[13]$

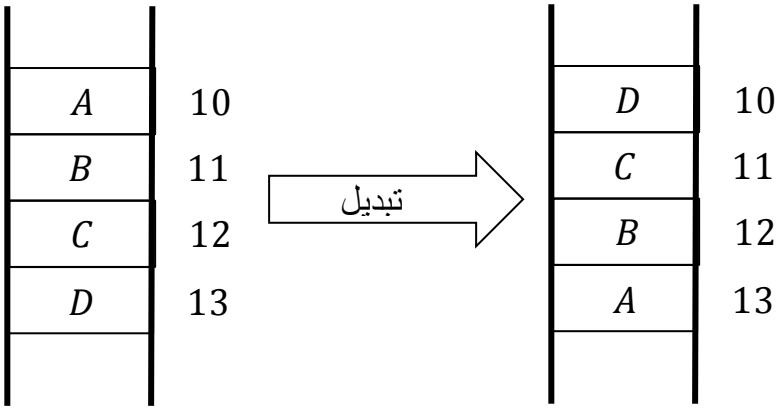
$XCHG\ \ AL, [10]$

$XCHG\ \ AL, [11]$

$XCHG\ \ AL, [12]$

$XCHG\ \ AL, [11]$

- يفضل في $XCHG$ استخدام الـ $Byte$ بدل الـ $Word$



Qu/ فسر الايعازات التالية:-

$MOV\ \ AL, [400\ H]$ /1

نقل محتويات موقع الذاكرة المعنون بالقيمة $400\ H$ الى المسجل AL

$MOV\ \ [SI] + 0200\ H, AL$ /2

نقل محتوى المسجل AL الى مواقع الذاكرة المعنون بمجموع قيمة الإزاحة ومحتوى المسجل $[SI]$.

$XCHG\ \ AX, [300\ H]$ /3

تبديل محتوى المسجل AX بموقعي الذاكرة المعنونين بالقيمة $300\ H$ و $301\ H$

Qu/ ما هو محتوى AX بعد تنفيذ التعليمات التالية:-

$MOV\ \ AX, 1000\ H$

$MOV\ \ BL, AL$

$MOV\ \ BH, 20\ H$

$XCHG\ \ AL, BH$

$XCHG\ \ AX, BX$

/Sol

	AH	AL
AX	10	00

	BH	BL
BX	20	00

$AX = 1000\,H\,, AH = 10\,, AL = 00$

$AH = 10\,, AL = 00, BL = 00$

$AH = 10\,, AL = 00\,, BL = 00\,, BH = 20$

$AH = 10\,, AL = 20\,, BL = 00\,, BH = 00$

$AH = 00\,, AL = 00\,, BL = 20, BH = 10$

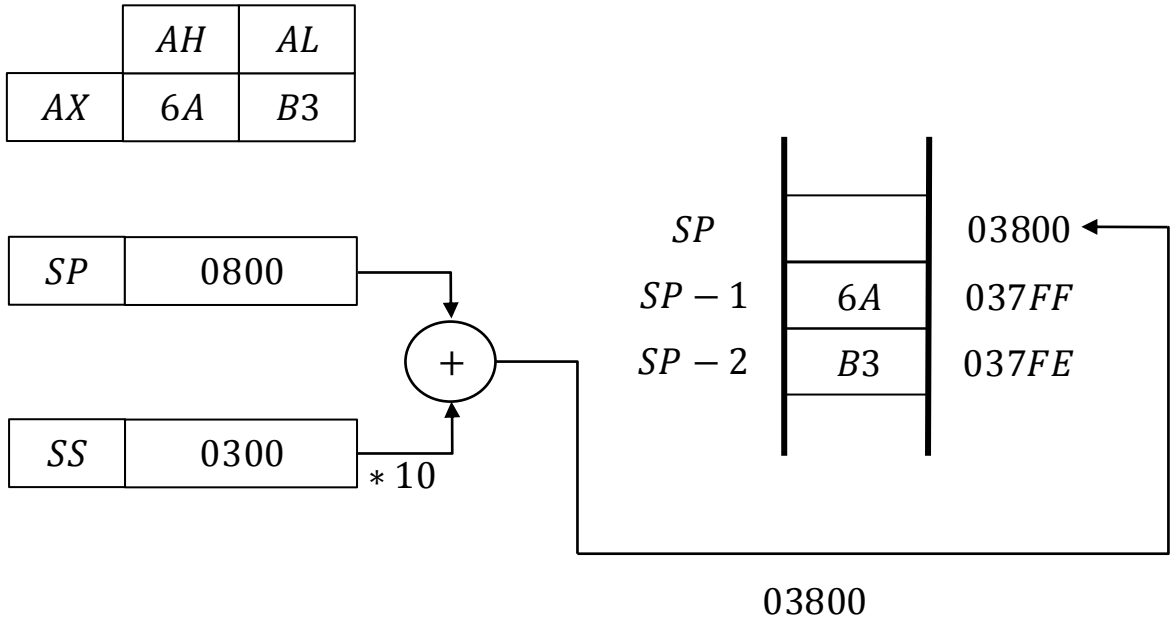
$\therefore AX = 0000\,H$

3- التعليمة $Push$

تستخدم لتخزين البيانات في المكدس

$Push\ Source$
$R.$ مسجل
$R.$ مسجل
$M.$ موقع ذاكرة
$Immediatly$ فوري

$Push\ \ AX$ /Ex

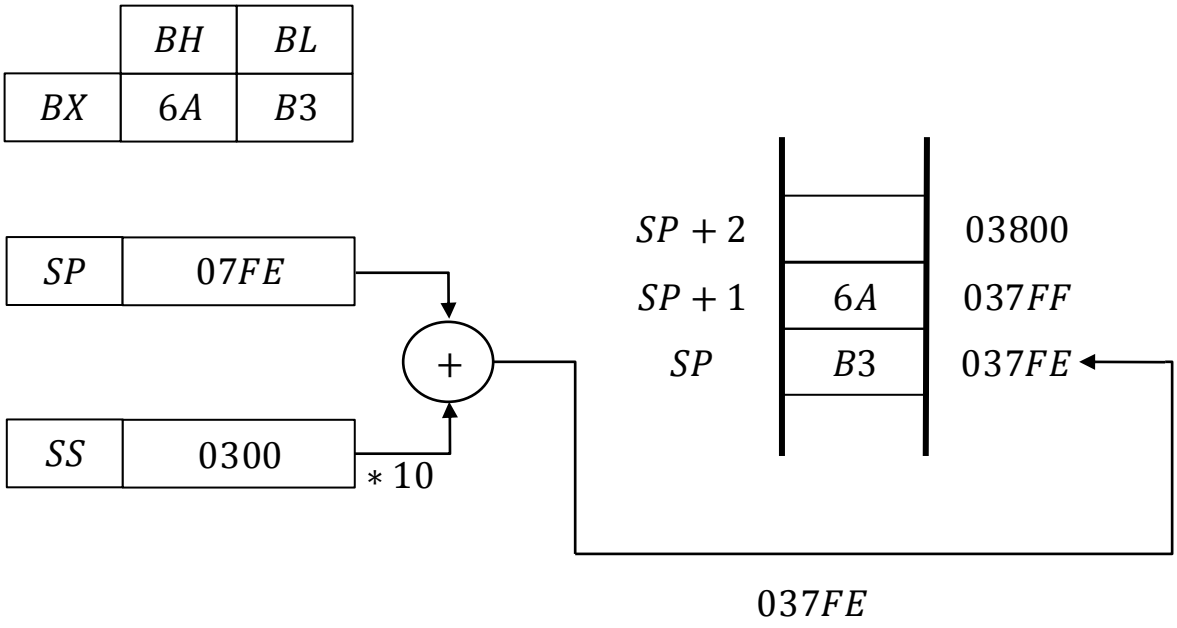


4- التعليمة *Pop*

استرجاع البيانات من مقطع المكدس

$Pop\ Source$
$R.$ مسجل
$M.$ موقع ذاكرة

Ex/ $Pop\ \ BX$



$037FE$

Ex/ اكتب برنامج بلغة التجميع لتبديل محتوى المسجل AX بمحتوى المسجل BX ومحتوى المسجل CX بمحتوى المسجل DX مستخدماً التعليمات $Pop\ ,Push$ فقط.

$Push\ \ AX$

$Push\ \ BX$

$Push\ \ CX$

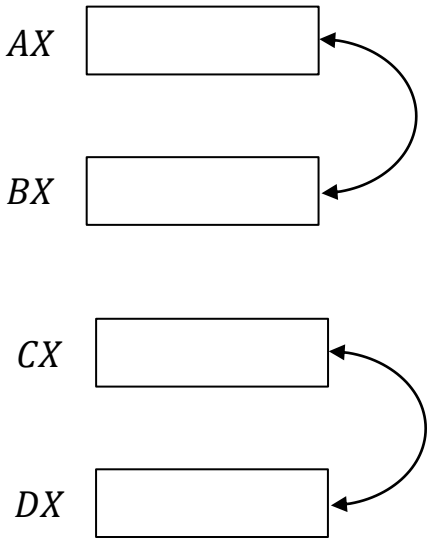
$Push\ \ DX$

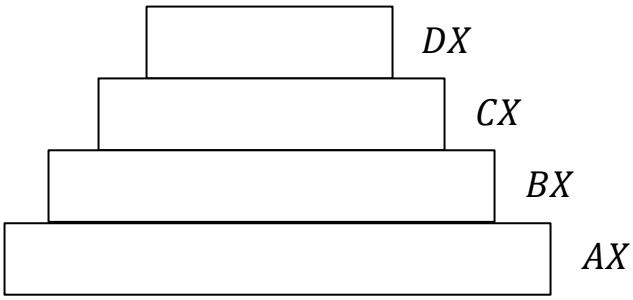
$Pop\ \ CX$

$Pop\ \ DX$

$Pop\ \ AX$

$Pop\ \ BX$





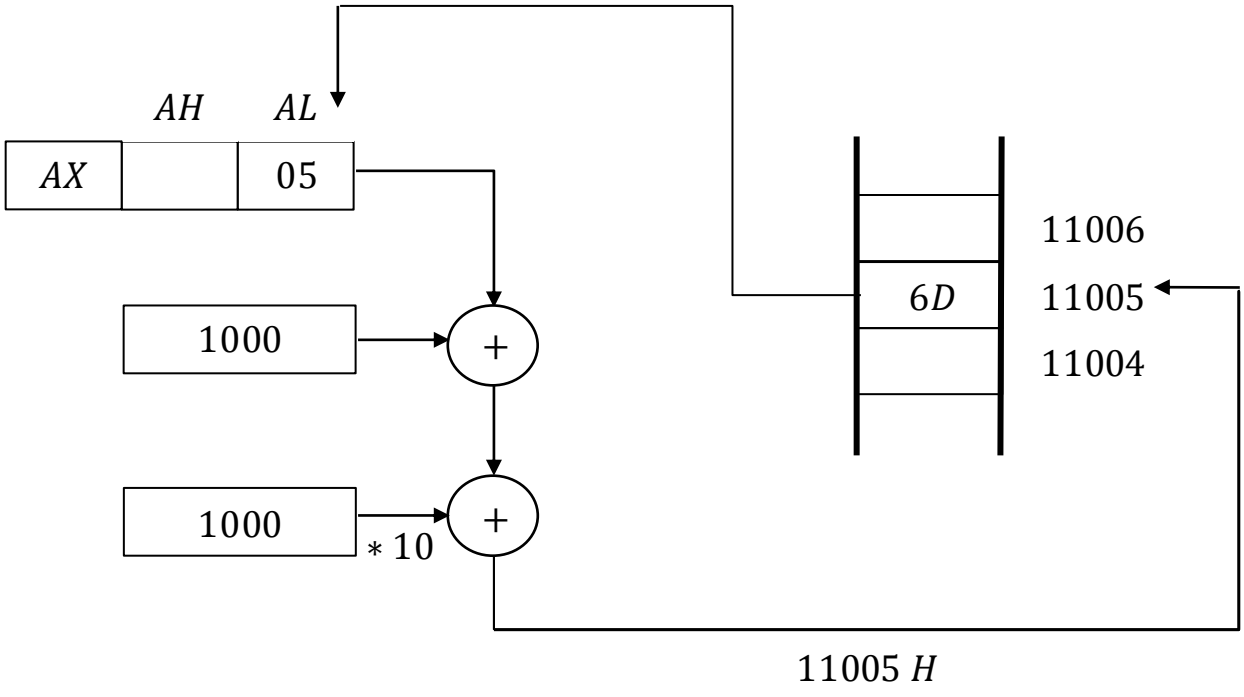
5- التعليمات $SAHF$, $LAHF$

$LAHF$ تستخدم لنقل الجزء الايمن من محتوى مسجل الاعلام الى المسجل AH.

$SAHF$ تستخدم لنقل محتوى المسجل AH الى الجزء الايمن من محتوى مسجل الاعلام.

6- التعليمة $XLAT$

تستخدم محتوى المسجل AL مع محتوى المسجل BX لتكوين عنوان موقع الذاكرة ضمن مقطع البيانات.
ثم تنقل محتويات ذلك الموقع الى المسجل AL.



7- التعليمات IN , OUT

تستخدم لنقل $1byte$ و $2byte$ بين المسجل AL او AX وموانئ الادخال والاخراج

هنالك طريقتان لعنونة موانئ الادخال والاخراج

1/ الطريقة الثابتة $8bit\ port$

يذكر رقم الميناء مباشراً ضمن التعليمة مثل

$IN\ \ AX\ ,19\ H$

نقل محتوى الميناء 19 $(1Word)$ الى المسجل AX.

$Out\ \ 17\ H\ ,AL$

نقل محتوى المسجل AL $(1byte)$ الى الميناء 17.

2/ الطريقة المتغيرة

يوضع رقم الميناء ضمن المسجل DX مثل

$IN\ \ \ AX\ ,DX$

نقل محتوى الميناء المعنون بمحتوى المسجل DX الى المسجل AX.

$Out\ \ \ DX\ ,AL$

نقل محتوى المسجل AL الى الميناء المعنون بمحتوى المسجل DX.

Ex/ اكتب برنامج بلغة التجميع لإيجاد مجموع محتوى الميناء 10 والميناء 11 وتخزين النتيجة في الميناء 12 علماً بأن محتوى الموانئ بحجم $Word$.

$IN\ \ \ AX\ ,10\ H$

نقل محتوى الميناء 10 الى المسجل AX

$MOV\ \ \ BX\ ,AX$

نقل محتوى المسجل AX الى المسجل BX

$IN\ \ \ AX\ ,11\ H$

نقل محتوى الميناء 11 الى المسجل AX

$ADD\ \ \ AX\ ,BX$

جمع محتوى المسجل AX مع محتوى المسجل BX

$Out\ 12, AX$

اخراج محتوى المسجل AX الى الميناء 12

التعليمات المنطقية OR, NOT, XOR,

$D.$ المصدر	⟶	$S.$ الهدف
$R.$ مسجل	⟶	$R.$ مسجل
$R.$ مسجل	⟶	$M.$ موقع ذاكرة
$M.$ موقع ذاكرة	⟶	$R.$ مسجل
$R.$ مسجل	⟶	$Immediatly$ فوري
$M.$ موقع ذاكرة	⟶	$Immediatly$ فوري

A	B	$A\ AND\ B$
0	0	0
0	1	0
1	0	0
1	1	1

A	B	$A\ OR\ B$
0	0	0
0	1	1
1	0	1
1	1	1

A	B	$A\ XOR\ B$
0	0	0
0	1	1
1	0	1
1	1	0

A	$NOT\ A$
0	1
1	0

Ex/ اذا كانت $AL = 55\ H$ و $BL = AA\ H$ اوجد ناتج ما يلي

$AND\ \ AL, BL$

$OR\ \ AL, BL$

$XOR\ \ AL\ ,BL$

/Sol

$AL = (55)_{16} \implies (01010101)_2$

$BL = (AA)_{16} \implies (10101010)_2$

$AND\ \ AL\ ,BL \quad (00000000)_2$

$OR\ \ AL\ ,BL \quad (11111111)_2$

$XOR\ \ AL\ ,BL \quad (11111111)_2$

Ex/ ما هو محتوى المسجل AX بعد تنفيذ التعليمات التالية

$MOV\ \ AL\ ,44\ H$

$MOV\ \ BL\ ,25\ H$

	AH	AL
AX	00	44

	BH	BL
BX	00	00

$XCHG\ \ BL\ ,AH$

$XOR\ \ AL\ ,AL$

	AH	AL
AX	25	44

	BH	BL
BX	00	25

$OR\ \ AH\ ,AH$

$AL = (44)_{16} \implies (01000100)_2$

$(01000100)_2$

$XOR\ \ AL\ ,AL \quad (00000000)_2 = (00)_{16}$

$AH = (25)_{16} \implies (00100101)_2$

$(00100101)_2$

$OR\ \ AH\ ,AH \quad (00100101)_2 = (25)_{16}$

محتوى المسجل AX بعد تنفيذ التعليمات اعلاه

$\therefore AX = 2500\ H$

Ex/ ما هو محتوى المسجل AX بعد تنفيذ التعليمات التالية

$MOV \quad AX, 2211\,H \Longrightarrow AH = 22, AL = 11$

$MOV \quad BX, 1122\,H \Longrightarrow BH = 11, BL = 22$

$XCHG \quad BL, AH \Longrightarrow AX = 2222, BX = 1111$

$XOR \quad BH, AH \Longrightarrow BH = 33$

$AND \quad BL, AH \Longrightarrow BL = 00$

تعليمات الإزاحة والدوران

تعليمات الإزاحة $Shift\ Instruction$

الازاحة لليمين

SHR

الازاحة لليسار

SHL

الصيغة العامة

$SHR \quad AX, 1$

$R.$ مسجل	1
$R.$ مسجل	CL
$M.$ موقع ذاكرة	1
$M.$ موقع ذاكرة	CL

Ex/ وضح نتيجة تنفيذ التعليمة التالية $SHR \quad BX \, , CL$ علماً ان $= EE33\,H$, $CL = 6$.

قبل التنفيذ BX

1	1	1	0	1	1	1	0	0	0	1	1	0	0	1	1
16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1

بعد التنفيذ BX

0	0	0	0	0	0	1	1	1	0	1	1	1	0	0	0
						16	15	14	13	12	11	10	9	8	7

CF

1
6

$BX = (0000001110111000)_2 = (03B8)_{16}$

- ملاحظة/

تزحيف الرقم مرتبة واحدة لليسار تعني ضرب الرقم في 2

تزحيف الرقم مرتبتين لليسار تعني ضرب الرقم في 4

تزحيف الرقم ثلاثة مراتب لليسار تعني ضرب الرقم في 8

تزحيف الرقم مرتبة واحدة لليمين تعني قسمة الرقم على 2

تزحيف الرقم مرتبتين لليمين تعني قسمة الرقم على 4

تزحيف الرقم ثلاثة مراتب لليمين تعني قسمة الرقم على 8

Ex/ اكتب برنامج بلغة التجميع لإيجاد قيمة التعبير الحسابي التالي $AX = 4BX$.

```
MOV  CL , 2
SHL  BX , CL
MOV  AX , BX
```

Ex/ اكتب برنامج بلغة التجميع لإيجاد قيمة التعبير الحسابي التالي $AX = 0.125BX$..

$$0.125 = \frac{1}{8}$$

```
MOV  CL , 3
SHR  BX , CL
MOV  AX , BX
```

تعليمات الدوران $Insturction$

ROR التدوير لليمين

ROL التدوير لليسار

RCR التدوير لليمين من خلال CF

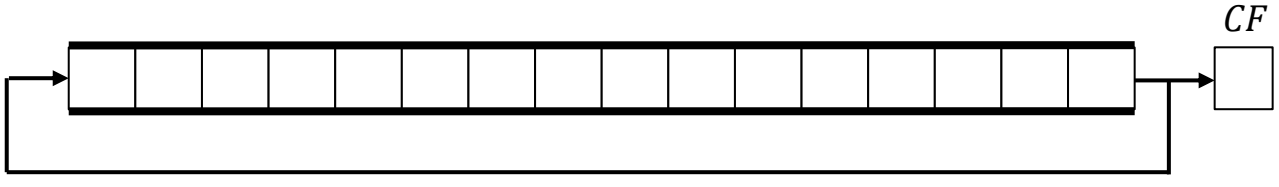
RCL التدوير لليسار من خلال CF

الصيغة العامة

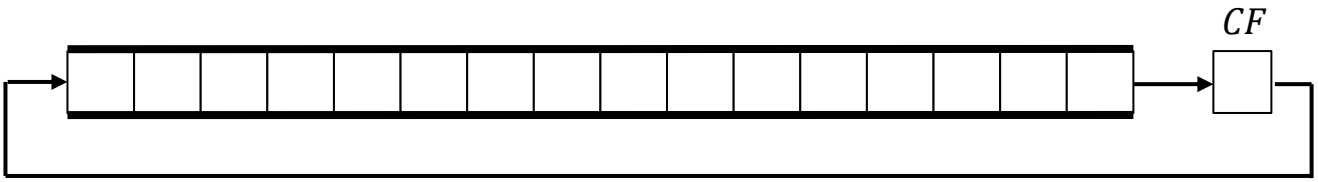
```
ROL  BX , 1
```

$R.$ مسجل	1
$R.$ مسجل	CL
$M.$ موقع ذاكرة	1
$M.$ موقع ذاكرة	CL

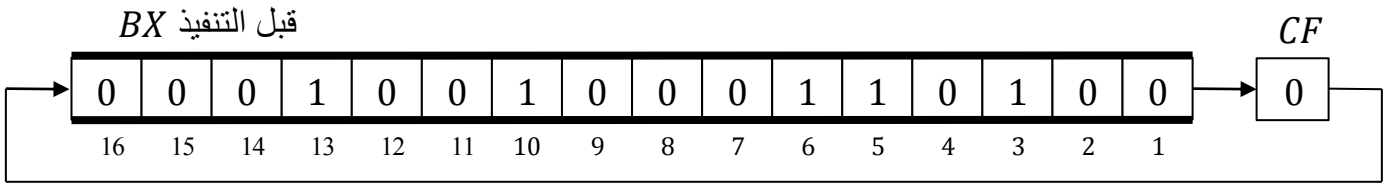
```
ROR  AX , 1
```

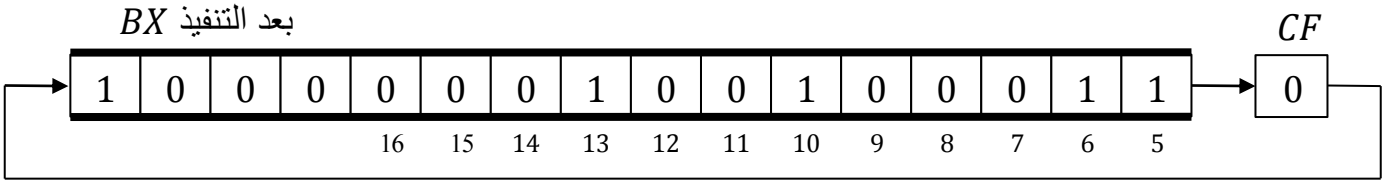


$RCR\ \ AX,1$



Ex/ ماهي نتيجة تنفيذ التعليمة $RCR\ \ BX,CL$ على المسجل BX و CF علماً بأن $BX = 1234\,H$ و$CL = 4$ و $CF = 0$.

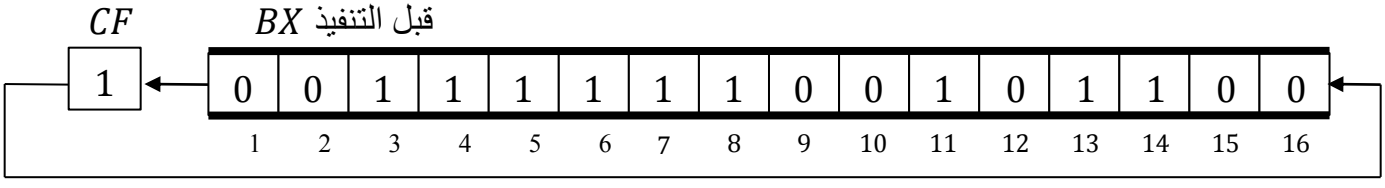


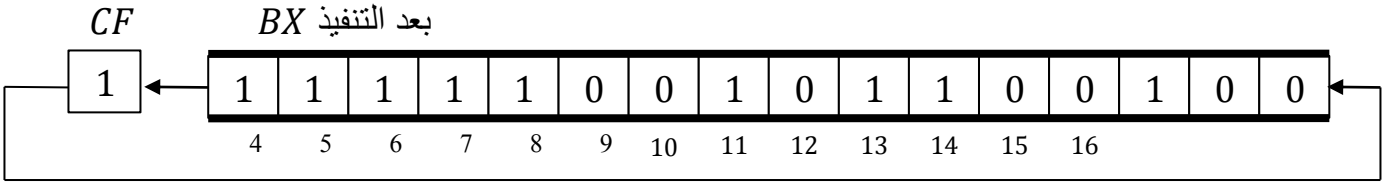


$BX = 8123\,H\,, CF = 0$

$BX = 8123\,H\ \,, CF = 0$

Ex/ ماهي نتيجة تنفيذ التعليمة $RCL\ \ BX,CL$ على المسجل BX و CF علماً ان $BX = 3F2C$ و $CL = 3$ و $CF = 1$.





$$BX = F964\,H\,, CF = 1$$

التعليمات الحسابية *Instruction*

1/ الجمع:-

أ- التعليمة *ADD*

المصدر $D.$	$\longrightarrow$	الهدف $S.$
مسجل $R.$	$\longrightarrow$	مسجل $R.$
مسجل $R.$	$\longrightarrow$	موقع ذاكرة $M.$
موقع ذاكرة $M.$	$\longrightarrow$	مسجل $R.$
مسجل $R.$	$\longrightarrow$	فوري $Immediatly$
موقع ذاكرة $M.$	$\longrightarrow$	فوري $Immediatly$

مثال/ ماهي نتيجة تنفيذ التعليمة التالية علما بأن

$AX = (1100)_{16}$

$BX = (0ABC)_{16}$

الحل/

ADD AX,BX

(نجمع AX مع BX والخزن في AX)

$\therefore AX = (1BBC)_{16}$

ب- التعليمة *ADC*

($carry$ الكري مع ADD)

$ADC\ D,S$

$D = D + S + CF$ | $CF(carry\ flag)$

عمل هذه التعليمة مشابه لعمل التعليمة ADD مع اضافة محتوى CF.

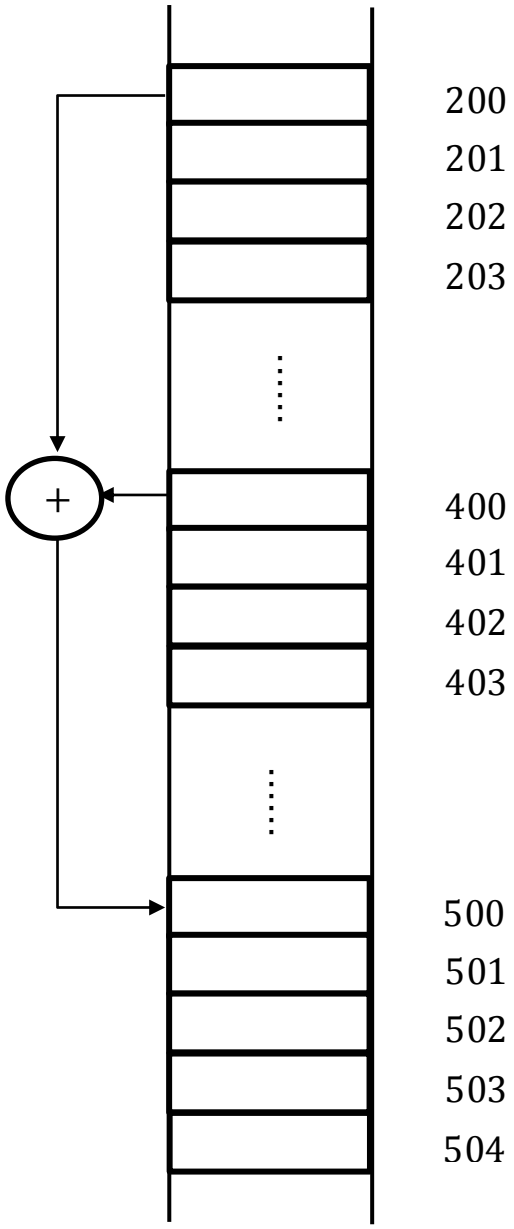
ج- التعليمة INC

$INC\ AH$ | نزيد محتوى المعامل مقدار 1

تستخدم لأضافه 1 الى قيمة المعامل (نستخدم لزيادة قيمة عداد او عنوان بـ1).

مثال/ اكتب برنامج بلغة التجميع لأضافه محتويات مواقع الذاكرة $(200H) \leftarrow (203H)$ الى محتويات مواقع الذاكرة $(400H) \leftarrow (403H)$ ثم تخزن النتيجة في مواقع الذاكرة $(500H) \leftarrow (504H)$ علماً بأن (مسجل $DS = 1000H$)

```
MOV  AX,[200H]
ADD  AX,[400H]
MOV  [500H],AX
MOV  AX,[202H]
ADC  AX,[402H]
MOV  [202H],AX
MOV  AL,00H
ADC  AL,00H
MOV  [504H],AL
```



2/ التعليمة SUB:-

(طرح المصدر من الهدف والتخزين يتم في الهدف)

المصدر $D.$	$\longrightarrow$	الهدف $S.$
مسجل $R.$	$\longrightarrow$	مسجل $R.$
مسجل $R.$	$\longrightarrow$	موقع ذاكرة $M.$
موقع ذاكرة $M.$	$\longrightarrow$	مسجل $R.$
مسجل $R.$	$\longrightarrow$	فوري $Immediatly$
موقع ذاكرة $M.$	$\longrightarrow$	فوري $Immediatly$

مثال/ ما هي نتيجة تنفيذ التعليمة التالية علماً ان $(BX = (1234)_{16})$ $(CX = (0123)_{16})$

$SUB \quad BX, CX$

الحل/

$SUB \quad BX, CX$

$BX = (1111)_{16}$

التعليمة SBB

$SBB \quad D, S$

عملها مشابه للتعليمة SUB مع طرح ($CF = 1$) من محتويات الهدف.

$$D = D - S - CF$$

(الاستعارة $CF = \ Carry\ Flag$)

التعليمة DEC

$DEC \ BX$

تستخدم لطرح واحد من قيمة المعامل وتستخدم للعداد والعناوين.

التعليمة NEG

$NEG\ \ BX$

تعطي القيمة السالبة لقيمة المعامل.

Ex/ اكتب برنامج بلغة التجميع لحساب قيمة التعبير الحسابي التالي:-

$AX = -2(BX + 1) - 0.5(CX - 1)$

الحل/

$INC\ \ BX \implies BX = BX + 1$

$SHL\ \ BX, 1 \implies BX = 2(BX + 1)$

$NEG\ BX \implies BX = -2(BX + 1)$

$MOV\ \ AX, BX \implies AX == -2(BX + 1)$

$DEC\ \ CX \implies CX = CX - 1$

$SHR\ \ CX \implies 0.5(\ CX - 1)$

$SUB\ \ AX, CX \implies AX = -2(BX + 1) - 0.5(\ CX - 1)$

مثال/ استخدم تعليمات الازاحة والتعليمات الحسابية لإيجاد حاصل ضرب محتوى المسجل AX بالعدد (4.5).

الحل/

توضيح:-

$4.5 = (4 + 0.5)$

$\therefore 4.5AX = (4AX + 0.5AX)$

عمل نسخة ثانية من (AX) في(BX)

$MOV\ \ , AX \implies AX = BX$

$SHR\ \ BX, 1 \implies BX = 0.5BX$

$MOV\ \ CL, 2$

$SHL\ \ AX, CL \implies AX = 4AX$

$ADD\ \ AX, BX \implies AX = 4.5AX$

مثال/ اكتب برنامج بلغة التجميع لحساب حاصل ضرب محتوى المسجل AX بالرقم 7.

الحل/

نرجع 7 على الاساس اي $(111)_2 = 7$ وذلك لأنه لا يمكن الضرب مباشرة مع الرقم 7.

$7 = 1 + 2 + 4$

$MOV\ BX\ AX \Longrightarrow BX = AX$

$SHL\ BX, 1 \Longrightarrow BX = 2BX$

$ADD\ AX, BX \Longrightarrow AX = 3AX$

$SHL\ BX, 1 \Longrightarrow BX = 4BX$

$ADD\ AX, BX \Longrightarrow AX = 7AX$

مثال/ استخدم تعليمات الازاحة والتعليمات الحسابية لإيجاد حاصل ضرب المحتوى المسجل AX بالعدد 5.625

الحل/

توضيح $0.625 = 0.5 + 0.125$

$MOV\ BX\ , AX \Longrightarrow BX = AX$

$MOV\ DX\ , AX \Longrightarrow DX = AX$

$MOV\ CL\ , 2 \Longrightarrow CL = 2$

$SHL\ BX\ , CL \Longrightarrow BX = 4BX$

$ADD\ AX, BX \Longrightarrow AX = 5AX$

$SHR\ DX\ , 1 \Longrightarrow DX = 0.5DX$

$ADD\ AX\ , DX \Longrightarrow AX = 5.5AX$

$SHR\ DX\ , CL \Longrightarrow DX = 0.625AX$

$ADD\ AX\ , DX \Longrightarrow 5.625AX$

عملية الضرب :-

- **التعليمة *MUL***

$MUL\ BL$

ضرب محتوى المسجل BL في محتوى المسجل AL وتخزن النتيجة في المسجل AX.

$MUL\ BX$ ضرب محتوى المسجل BX في محتوى المسجل AX وتخزن النتيجة في المسجلين AX و DX.

- **عملية القسمة *DIV***

$DIV\ BL$

قسمة محتوى المسجل AX على محتوى المسجل BL وتخزن خارج القسمة في المسجل AL وباقي القسمة في المسجل AH.

$DIV\ BX$

قسمة محتوى المسجلين AX و DX على محتوى المسجل BX وتخزن خارج القسمة في المسجل AX وباقي القسمة في المسجل DX.

مثال/ اكتب برنامج بلغة التجميع لضرب محتويات مواقع الذاكرة 100,101 بمحتويات مواقع الذاكرة 200,201 وتخزين النتيجة بمواقع الذاكرة (300,301,302,303,).

توضيح: AX اقل مرتبة من DX

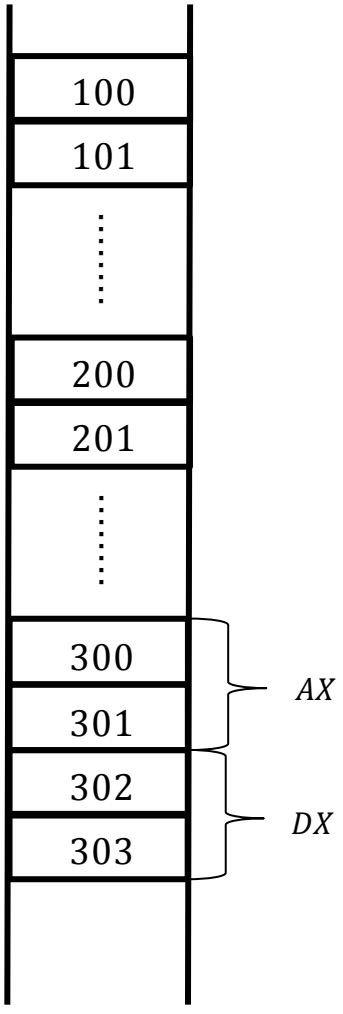
$MOV\ AX,[100]$

$MOV\ BX,[200]$

$MUL\ BX$

$MOV\ [300],AX$

$MOV\ [302],DX$



مثال/ اكتب برنامج بلغة التجميع لضرب محتوى مواقع الذاكرة 100 و 101 بمحتوى المسجل BL وتخزين النتيجة في مواقع الذكرة 200 و 201 و202 ؟

$MOV\ AL,[100]$

$MUL\ BL$

$MOV\ [200],AL$

$MOV\ BL\ ,AH$

$MOV\ AL,[101]$

$MUL\ \ BL$

$ADD\ AL\ ,BH$

$MOV\ [201]\ ,AL$

$ADC\ AH\ ,00$

$MOV\ [202]\ ,AH$

احل اخر

$MOV\ BH,00$

$MOV\ AX,[100]$

$MUL\ \ BX$

$MOV\ \ [200]\ ,AX$

$MOV\ \ [202]\ ,DX$

مثال/ اكتب برنامج بلغة التجميع لحساب قيمة التعبير الحسابي التالي مستخدما تعليمات الازاحة

$$AX = 0.25(3BX + 0.75DX)$$

الحل/

$$AX = \frac{1}{4}\left(BX + 2BX + \frac{1}{2}DX + \frac{1}{4}DX\right)$$

$MOV\ \ AX\ ,BX \qquad \Longrightarrow \qquad (AX = BX)$

$SHL\ BX\ ,1 \quad \Longrightarrow \quad (BX = 2BX)$

$ADD\ AX\ ,BX \quad \Longrightarrow \quad (AX = 3BX)$

$SHR\ DX\ ,1 \quad \Longrightarrow \quad \left(DX = \frac{1}{2}DX\right)$

$ADD\ AX\ ,DX \quad \Longrightarrow \quad \left(AX = 3BX + \frac{1}{2}DX\right)$

$SHR\ DX\ ,1 \quad \Longrightarrow \quad \left(DX = \frac{1}{4}DX\right)$

$ADD\ AX\ ,DX \quad \Longrightarrow \quad (AX = 3BX + 0.75DX)$

$MOV\ CL\ ,2 \quad \Longrightarrow \quad (CL = 2)$

$SHR\ AX\ MCL \quad \Longrightarrow \quad \left(AX = 0.25(3BX + 0.75DX)\right)$

مثال/ اكتب برنامج بلغة التجميع لحساب قيمة التعبير التالي

$AX = 9.625BX$

الحل/

$AX = BX + 8BX + 0.5BX + 0.125BX$

$MOV\ AX\ ,BX$

$MOV\ CL\ ,3$

$MOV\ DX\ ,BX \quad \Longrightarrow BX = 8BX$

$ADD\ AX\ ,BX \quad \Longrightarrow AX = 9BX$

$SHR\ DX\ ,1 \quad \Longrightarrow DX = 0.5BX$

$ADD\ AX\ ,DX \quad \Longrightarrow AX = 9BX + 0.5BX$

$MOV\ CL\ ,2$

$SHR\ DX\ ,CL$

$ADD\ AX\ ,DX \quad \Longrightarrow AX = 9.625BX$

5/ تعليمات التحكم بالبرامج

تعليمات القفز $JUMP$

1- القفز غير المشروط $Uncondition\ Jump\ (JMP)$

هناك ثلاثة انواع من القفز غير المشروط

1/ القفز القصير $Short\ Jump$

ياخذ ايعاز هذا النوع من القفز $(2byte)$ ويسمح بالقفز الى مواقع ذاكرة ضمن من عنوان القفز $(-128byte\ \ \&\ +127byte)$.

2/ القفز القريب $Near\ Jump$

يأخذ هذا النوع من القفز ($3byte$) ويسمح بالقفز الى موقع ذاكرة ضمن $(\pm 32\text{Kbyte})$ ضمن نفس المقطع الحالي.

3/ القفز البعيد $Far\ Jump$

يأخذ ايعاز هذا النوع من القفز $(5byte)$ ويسمح بالقفز الى اي موقع للذاكرة ضمن ذاكرة الحاسب ولأي مقطع.

2- القفز المشروط $Condition\ Jump$

وهو القفز الذي يحدث وفق شرط معين كما موضح بالجدول التالي:-

التعليمة	الوظيفة
JA	$Jump\ If\ Above$
JAE	$Jump\ If\ Above\ or\ Equle$
JB	$Jump\ Below$
JBE	$Jump\ Below\ or\ Equle$
JC	$Jump\ If\ Carry$
$JE\ or\ JZ$	$Jump\ Equle, Jump\ Zero$

6/ الايعاز CMP

يستخدم للمقارنة ولا يؤثر على الهدف, فقط يؤثر على مسجل الاعلام $(Flag\ Register)$ ويستخدم قبل ايعازات القفز المشروط.

الصيغة العامة $CMP\ \ D\ ,S$

مثال/ اكتب برنامج بلغة التجميع لإيجاد اكبر رقم من بين ثلاثة ارقام مخزونة في المسجلات AX و BX وDX ثم خزن النتيجة في المسجل CX.

```
CMP  AX ,BX
JAE  L1
XCHG  AX ,BX
L1:CMP  AX ,DX
JAE  L2
XCHG  AX ,DX
L2:MOV  CX ,AX
```

توضيح: وهو يشبه $ELSE\ ,THEN$

- في حالة ايجاد الاصغر

```
CMP  AX ,BX
JBE  L1
XCHG  AX ,BX
L1:CMP  AX ,DX
JBE  L2
XCHG  AX ,DX
L2:MOV  CX ,AX
```

مثال/ اذا كان محتوى AX يساوي $2222\ H$ ضع في المسجل AX القيمة $2BX$ واذا كان محتوى المسجل BX يساوي $2222H$ ضع في المسجل BX القيمة $2AX$.

$$AX = 2222H \Longrightarrow AX = 2BX$$

$$BX = 2222H \Longrightarrow BX = 2AX$$

$/Solution$

$CMP\ AX, 2222$

$JE\ L1$

$L4: CMP\ BX, 2222$

$JE\ L2$

$JMP\ L3$

$L1: SHL\ BX, 1$

$MOV\ AX, BX$

$JMP\ L4$

$L2: SHL\ AX, 1$

$MOV\ BX, AX$

$L3: END$

مثال/ اكتب برنامج بلغة التجميع لقسمة محتوى مواقع الذاكرة من 100 الى 103 على محتوى مواقع الذاكرة $111 - 110$ وتخزين ناتج القسمة في مواقع الذاكرة $200 - 201$ وباقي القسمة في مواقع $203 - 202$

Sol

$MOV\ AX, [100]$

$MOV\ DX, [102]$

$MOV\ BX, [110]$

$DIV\ BX$

$MOV\ [200], AX$

$MOV\ [202], DX$

مثال/ اكتب برنامج بلغة التجميع لتخزين الرقم 10 في المسجل BL اذا كانت محتوى المسجل AX اكبر من $2222H$, وتخزين الرقم 20 في المسجل BL اذا كان محتوى المسجل AX اكبر من $1111H$, والا يخزن الرقم 30.

Sol

$CMP\ AX,2222$

$JBE\ L1$

$MOV\ BL,10$

$JMP\ L3$

$L1:CMP\ AX,1111$

$JBE\ L2$

$MOV\ BL,20$

$JMP\ L3$

$L2:MOV\ BL,30$

$L3:END$

حل اخر

$CMP\ AX,2222$

$JBE\ L1$

$L1:MOV\ BL,10$

$JMP\ L3$

$CMP\ AX,1111$

$JAE\ L2$

$L2:MOV\ BL,20$

$JMP\ L3$

$MOV\ BL,30$

$L3:END$

تعليمات الدوران $Loop\ Instruction's$

- **الدوران غير المشروط $Loop\ L1$**

تنفيذ هذه التعليمة يؤدي الى الانتقال الى المواقع $L1$ مع نقصان العداد (المسجل CX) بمقدار 1.

وتتكرر عملية الدوران لحين وصول العداد الى الصفر.

$Ex/$ لكتب برنامج بلغة التجميع لمعرفة عدد الوحدات في الرقم $(03EE)$

$Sol/$

$MOV\ \ DX\ ,03EEH \Longrightarrow DX = 03EE\ H$

$MOV\ \ CX\ ,10H \Longrightarrow CX = 10\ H$

$MOV\ \ AL\ ,00 \Longrightarrow AL = 0$

$L1{:}SHR\ \ DX\ ,1$

$ADC\ \ AL\ ,00 \Longrightarrow (AL = 0) + 0 + CF = CF$

$Loop\ L1$

في حالة طلب حساب عدد الاصفار

$MOV\ \ BL\ ,10H$

$SUB\ \ BL\ ,AL$

$Ex/$ اكتب برنامج بلغة التجميع لحساب مجموع الارقام في المواقع الذاكرة من 400 الى 405 وخزن النتيجة في المسجل AX.

$MOV\ \ CX\ ,6$

$MOV\ \ AX\ ,00$

$MOV\ \ SI\ ,400$

$L1{:}ADD\ \ AL\ ,[SI]$

$ADC\ \ AH\ ,00$

$INC\ \ SI$

$Loop\ L1$

توضيح:

$CX = 6$

$SI = 400$

$AL = 9\ , CF = 0$

$SI = 401$

$CX = 5$

$AL = 7\ , CF = 1$

$AH = 1$

$SI = 402$

$CX = 4$

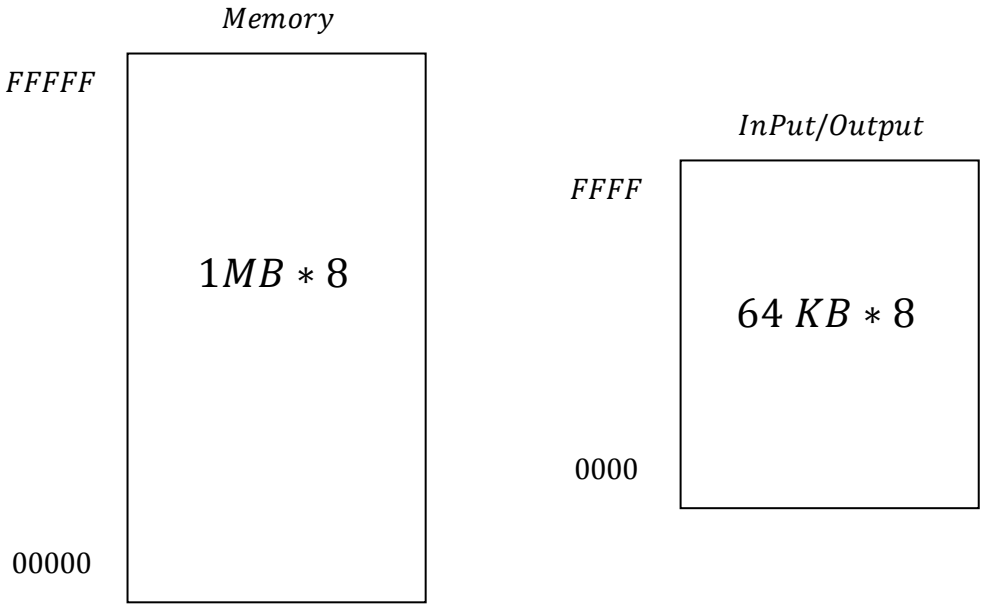
$AL = 4\ , CF = 1$

$AH = 0$

عنونة موانئ الإدخال والاخراج $I/O\ Port\ Address$

هناك طريقان للعنونة:-

1- العنونة المعزولة $isolated\ I\ /\ O\ Addressing$

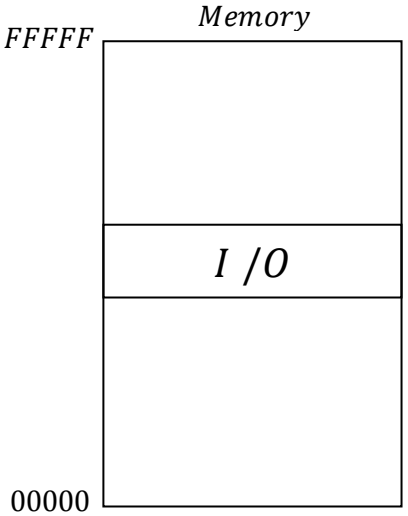


هي الطريقة الاكثر استخداما بالنسبة لمعالجات شركة $intel$. مصطلح معزول يصف الكيفية التي يتم فيها عزل مواقع الادخال والاخراج عن نظام الذاكرة.

ميزة هذه الطريقة هي امكانية استخدام الذاكرة بصورة كاملة من قبل المستخدم بدون استغلال بعض عناوين الذاكرة لمواقع الادخال والاخراج.

عيوب هذه الطريقة هي ان اي عملية وصول للبيانات الموجودة في موانئ الادخال والاخراج تتم عن طريق الايعازات (التعليمات) $IN\ ,\ OUT$, ادخال سلسلة رمزية INS, اخراج سلسلة رمزية $OUTS$

2- العنونة من خلال الذاكرة $Memory\ Mapped\ I\ /\ O$



في هذه الطريقة تكون عنونة موانئ الادخال والاخراج من خلال عناوين الذاكرة. وفي هذه الطريقة يتم استخدام كافة التعليمات التي تتعامل مع الذاكرة للوصول الى موانئ الادخال والاخراج, كذلك البيانات المنقولة من موانئ الادخال والاخراج يمكن ان تخزن في المسجلات الداخلية للمعالج ولكن من عيوبها هو استغلال قسم من عناوين الذاكرة لموانئ الادخال والاخراج.

النمط المحمي لعنونة الذاكرة *Protected Mode Memory Addressing*

يستخدم هذا النمط ابتداء من المعالج 80286 ولغاية $pentium$ ويسمح هذا النمط بالوصول الى البيانات والبرامج الموجودة في مواقع الذاكرة التي هي اكبر من $1Mbyte$ ويختلف هذا النمط عن النمط الحقيقي $Real\ Mode$ الذي يستخدم مع المعالج 8088, 8086

في النمط الحقيقي يحدد عنوان الذاكرة من خلال قيمة مسجل المقطع مضافاً الية قيمة الازاحة.

اما في النمط المحمي فان مسجل المقطع يحتوي على المنتقى $(Selector)$ الذي ينتقي واحد من 8192 واصف $Descriptor$ في جدول الواصف

يوجد هنالك جدولان للمواصفات

الاول- يمثل جدول الواصفات العامة $(Global\ Descriptor)$ والتي تحتوي على تعريفات لمقاطع البرامج.

الثاني – يمثل جدول الواصفات المحلية $(Local\ Descriptor)$ والتي تخص تطبيقات محددة.

يحتوي كل موقع في جدول الواصف على العنوان الأساس$Base\ Address$ والذي يمثل عنوان بداية مقطع الذاكرة وكذلك يحتوي على العنوان المحدد $Limit\ Address$ والذي يمثل قيمة الازاحة داخل المقطع.

فمثلا اذا كانت بداية المقطع هي $F00000H$ وتنتهي عند $F000FF$ فيكون

$$Base\ Address = F00000$$

$$Limit\ Address = FF$$

7	$Base(B24-B31)$	G	D	O	AV	$Limit\ (L16-L19)$	6
5	$Access\ Rights$	$Base\ (B16-B23)$					4
3	$Base\ (B0-B15)$						2
1	$Limit\ (L0-L15)$						0

الواصف $Descriptor(80386 \longrightarrow Pentium4)$

المقاطعة $Interrupt$

توفر المقاطعة ميكانيكية سريعة لتغيير مجرى البرنامج عند استلام اشارة المقاطعة من جهاز خارجي تبين ان الجهاز الخارجي يحتاج الى خدمة, فالمعالج يجب ان يوقف ما يعمل به في الجزء الرئيسي في البرنامج ويمرر السيطرة الى البرنامج الفرعي يمثل الدالة المطلوبة من قبل الجهاز الذي احدث المقاطعة. ويدعى هذا البرنامج بالبرنامج الفرعي الخدمي للمقاطعة $(Interrupt\ Service\ Routine)$.

جدول المقاطعة $Interrupt\ Vector\ Table$

يستخدم هذا الجدول لربط نوع المقاطعة بمقطع البرنامج الفرعي الخاص لخدمته والموجود في ذاكرة تخزين البرامج.

يحتوي الجدول على 256 متجه عنوان يبدا من 0 ولغاية 255 فكل عنوان يتطلب كلمتان $2word$ واحد للمسجل Ip والاخرى للمسجل CS.

فمثلا اذا اردنا الوصول الى العنوان $IP_{50}CS_{50}$ فبما ان كل عنوان يأخذ $4Byte$ فيكون العنوان المطلوب

$$Address = 4 * 50 = (200)_{10} = (C8)_{16}$$

$$\therefore IP_{50} = (C8)_{16}$$

$$CS_{50} = (CA)_{16}$$

توضيح:

$CS_{50} = (CA)_{16}$ بعد الزيادة مقدار 2

$$2 + IP_{50} = 2 + C8 = CA$$

الذاكرة الافتراضية (التخيلية) $Virtual\ Memory$

تستخدم ذاكرة الوصول العشوائي RAM لتخزين الأوامر والبيانات التي تخص البرامج التي يعمل عليها المعالج في الوقت الحاضر عند فحص سلوك المعالج يلاحظ في كثير من الحالات ان المعالج لا يشترط ان يكون كامل البرنامج موجود في ذاكرة الوصول العشوائي RAM وحتى نتجنب حجز مكان ثمين في الذاكرة لأجزاء قد لا تستحقها ثم الاستعانة بتقنية الذاكرة الافتراضية التي تمكن النظام من تنفيذ برامج لا تقع في الكامل بالذاكرة العشوائية RAM بل الاجزاء الضرورية منها يتم تخزين البرامج على الذاكرة الرئيسية باستخدام مبدأ تقسيم البرامج الى صفحات $(Pages)$ والذي يعمل على جعل المساحة الواقعية للبرنامج مقسمة الى صفحات كما ان تخزين هذه الصفحات ليس متصلاً بل يحسب ما يتوفر لنا من الأماكن الشاغرة في الذاكرة الرئيسية.

وبذلك نستنتج ان هذا النوع من الذاكرة يعطي صورة للمستخدم بأن البرنامج قد تم تحميله بالكامل على الذكرة الرئيسية خلال فترة المعالجة.

مميزات استخدام تقنية الذاكرة الافتراضية

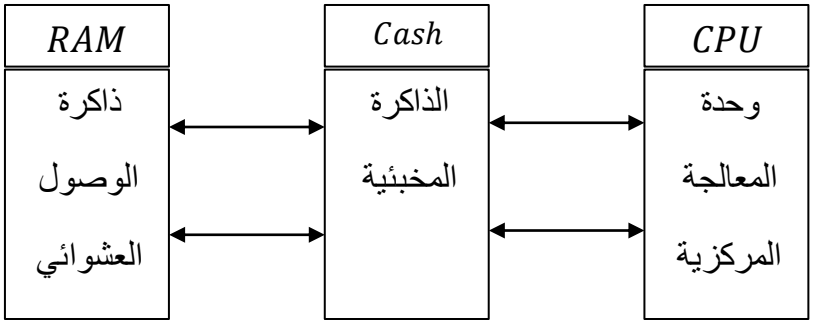
1- النظام الذي يطبق هذا المبدأ يعمل بشكل جيد مع بيئة البرامج المتعددة.

2- يتم تقليل وقت الانتظار بشكل كبير في ظل وجود هذه الذاكرة.
3- ان حجم البرنامج لا يؤثر كثيراً في عدد البرامج التي يمكن تحميلها للذاكرة مما يعطينا استخدماً اكثر كفاءة للذاكرة.
4- تقدم هذه التقنية تسهيلاً لعملية مشاركة الأوامر او البيانات بين برامج التي تقع في الذكرة.

عيوب استخدام تقنية الذاكرة الافتراضية

1- قد تكون هذه التقنية مكلفة اما من ناحية المساحة الداعمة او من ناحية الوقت.
2- تزيد بشكل لا يمكن تجاهله عدد مرات مقاطعة البرامج.
3- تزيد من تعقيدات مهمة البرمجة.

الذاكرة المخبئية $Cash\ Memory$

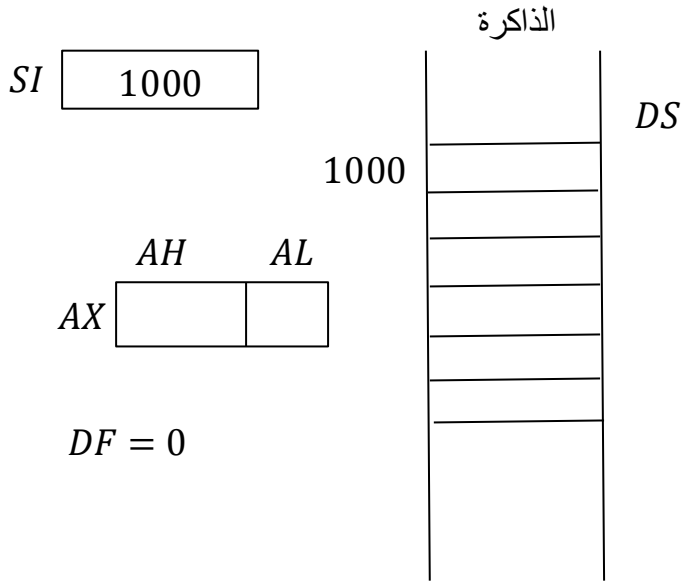


وهي ذاكرة بينية موجودة بين وحدة المعالجة المركزية وذاكرة الوصول العشوائي RAM تمتاز هذه الذاكرة بصغر حجمها وسرعتها العالية مقارنة مع ذاكرة الوصول العشوائي ان الاستدعاء المتكرر للبيانات من قبل وحدة المعالجة المركزية يجعل نسخ احتياطية للبيانات في ذاكرة المخبئية مما يسهل من سرعة وصول وحدة المعالجة اليها.

تعليمات السلاسل الرمزية *String Instruction*

1- التعليمة *LODS* :-

LODSW , LODSB

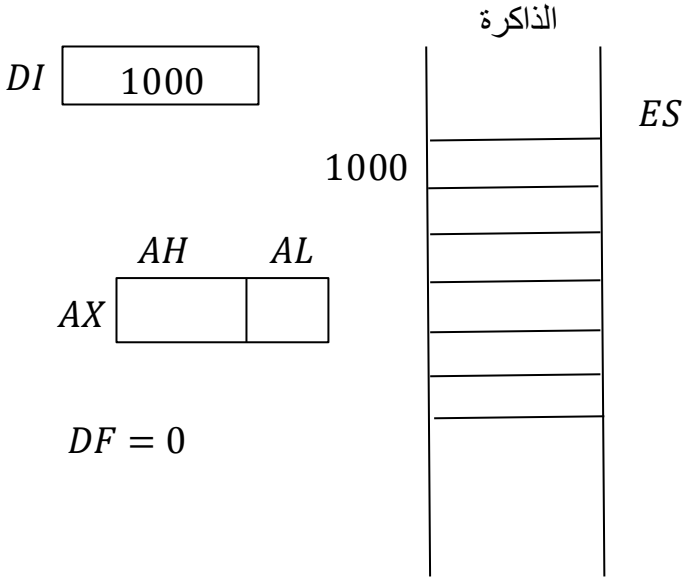


LODSB:- نقل محتوى الذاكرة المعنون في محتوى المسجل *SI* الى المسجل *AL* مع زيادة محتوى المسجل *SI* بمقدار 1 اذا كانت ($DF = 0$) او نقصان بمقدار 1 اذا كان $DF = 1$.

LODSW:- تعمل على نقل محتوى موقع الذاكرة المعنون بالمسجل *SI* والذي يليه الى المسجل *AX* وكذلك يزداد او ينقص *SI* بمقدار 2 اعتماداً على *DF*.

2- التعليمة *STOS*:-

STOSW , STOSB

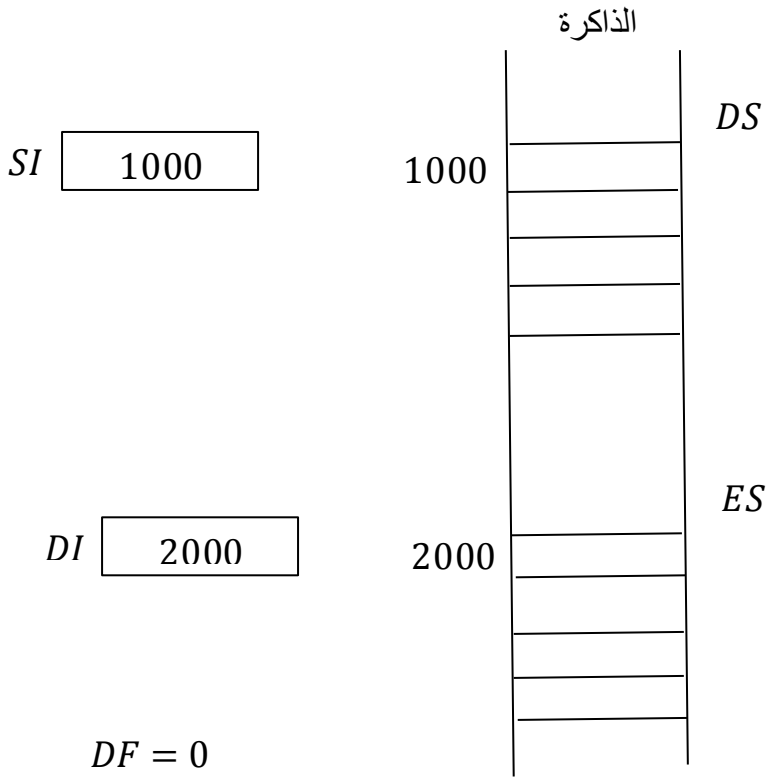


$STOSB$:- وظيفتها نقل محتوى المسجل AL الى مواقع الذاكرة المعنون للمسجل DI ضمن المقطع الاضافي ES ويزداد محتوى المسجل DI او ينقص بمقدار 1 اعتماداً على قيمة DF ($DF = 0$ معناه زيادة والعكس هو الصحيح).

- ملاحظة/ $STOSW$:- وظيفتها نقل محتوى المسجل AX الى مواقع الذاكرة المعنون بالمسجل DI والذي يليه بزيادة ونقصان DI بـ 2.

3- التعليمة $MOVS$

$MOVSB$, $MOVSW$



$MOVSB$:-وظيفتها نقل محتوى موقع الذاكرة المعنون بالمسجل SI ضمن مقطع البيانات DS الى موقع الذاكرة المعنون بالمسجل DI ضمن المقطع الاضافي ES مع زيادة او نقصان محتوى المسجلين SI, DI بـ1 اعتماداً على قيمة DF ($DF = 0$ تعني زيادة).

$MOVSW$:- عملها يشبه لعمل $MOVSB$ مع نقل محتوى موقعين للذاكرة وزيادة والنقصان SI, DI بمقدار 2.

مثال/ اكتب برنامج بلغة التجميع لنقل $12Byte$ من مقطع البيانات عند العنوان 100 الى مقطع الاضافي عند العنوان 200 مستخدماً تعليمات السلاسل الرمزية.

الحل/

1- نقل بـ $Word$

$MOV\ SI,100H$

$MOV\ DI,200H$

$MOV\ CX,06$

CLD

$L1:MOVSW$

$Loop\ L1$

2- نقل بـ $Byte$

$MOV\ SI,100H$

$MOV\ DI,200H$

$MOV\ CX,OCH$

CLD

$L1:MOVSB$

$Loop\ L1$

- ملاحظة/ لتصفير DF في مسجل الاعلام نستخدم التعليمة (CLD) $Clear\ Dircetion\ Flag$.