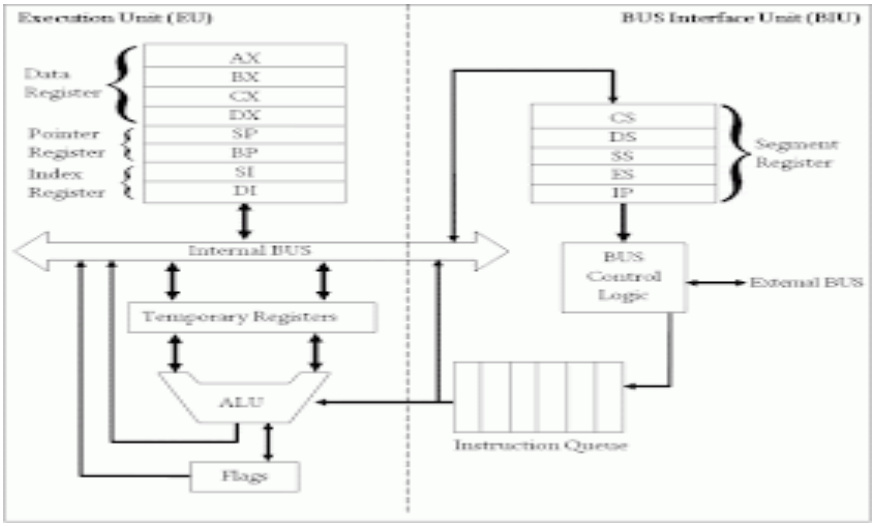
المعالج Intel 8086 Microprocessor

المعالج 8086

في العام 1978 طرحة شركة انتيل المعالج Intel 8086 ليكون أول معالج 16بت في العالم حيث يستطيع هذا المعالج تنفيذ 2.5 مليون عملية في الثانية بالإضافة إلى عنونة 1 M byteمن الذاكرة بالإضافة إلى العديد من التحديثات في بنية المعالج الداخلية ودعم تعليمات جديدة لم تكن موجودة مسبقا كعمليتي الضرب و القسمة و احتواء المعالج على مسجل رتل التعليمات "طابور التعليمات" الذي يعتبر أساس فكرة الذاكرة Cache في المعالجات الحديثة إذ كان هذا المسجل عبارة عن ست حجرات من الذاكرة حجم كل حجرة 8 بت يتم وضع التعليمات فيها ريثما ينتهي المعالج من تنفيذ التعليمة الحالية.

البنية الداخلية للمعالج 8086



بنية المعالج 8086

يتألف المعالج 8086 من وحدتين منفصلتين هما :

1) وحدة ملائمة الممرات (Execution Unit) : و سنرمز لها بالرمز EU .
2) وحدة التنفيذ (Bus interface Unit) : و سنرمز لها بالرمز BIU .

بشكل عام فإن الـ BIU مسؤولة عن معظم الأعمال مثل : إحضار التعليمة، قراءة و كتابة المتحولات في الذاكرة، إدخال و إخراج المعطيات من و إلى الأجهزة المحيطية.

أما الـ EU فهي مسؤولة عن تنفيذ التعليمات. و كلا الوحدتين تعملان بشكل متوازٍ لتخفيض الزمن المطلوب لإحضار عدة تعليمات و تنفيذها.

ملاحظة: من الجدير بالذكر بأن هنالك ثلاثة ممرات في الحاسب و هي:

1) ممر المعطيات DATA BUS : و يصل بين المعالج و الذاكرة وظيفته نقل المعطيات من و إلى الذاكرة.
2) ممر العناوين ADDRESS BUS : و يصل بين المعالج و الذاكرة أيضاً و وظيفته نقل العناوين من المعالج إلى الذاكرة.
3) ممر التحكم CONTROL BUS : لتنسيق عمل الممرين السابقين.

وحدة ملائمة الممرات Bus Interface Unit

و تستخدم لملائمة المعالج مع العالم الخارجي. و تتألف من : جامع العناوين، مسجلات المقاطع، وحدة التحكم بالمحرف، صف التعليمات.

تقوم وحدة الـ BIU بالتحكم بممر المعطيات و ممر العناوين و ممر التحكم .

تحضر BIU التعليمات من الذاكرة بايت بايت و تضعها فيما يسمى برتل التعليمات (صف التعليمات) الذي يتسع لست بايتات كحد أقصى و من الطبيعي أن التعليمة التي تدخل رتل التعليمات أولاً يتم تنفيذها أولاً للمحافظة على ترتيب التعليمات و يدعى هذا المبدأ بـ الداخل أولاً خارج أولاً First In Last Out و نرمز لهذا المبدأ بـ FIFO.

إن إحضار شيفرة التعليمة التالية يتم عندما تكون وحدة التنفيذ EU مشغولة بتنفيذ التعليمة الحالية (هذه إحدى محسنات المعالج 8086 عن أسلافه حيث كانت الـ CPU في المعالجات السابقة للمعالج 8086 تتوقف عن العمل خلال فترة تنفيذ التعليمة الحالية).

عندما تفك وحدة التنفيذ EU شيفرة تعليمة ما من رتل التعليمات و تكون هذه التعليمة تعليمة تؤدي إلى تغيير تسلسل تعليمات البرنامج (قفز إلى برنامج فرعي مثلاً) عندها يتم تصفير رتل التعليمات و إعادة ملئه من جديد بتعليمات البرنامج الفرعي (لأن وحدة ملائمة الممرات BIU تجلب التعليمات دون معرفة ما تؤديه هذه التعليمات).

وحدة التنفيذ Execution Unit

و هي مسؤولة عن فك شيفرة التعليمات و تنفيذها و تتألف من :

1) و حدة الحساب و المنطق.
2) مسجل الأعلام.
3) ثمانية مسجلات للأغراض العامة.
4) مسجلات مؤقتة.
5) منطق التحكم بـ EU.

تجلب وحدة التنفيذ EU التعليمات من مقدمة رتل التعليمات في وحدة ملائمة الممرات BIU و تفك شيفرتها و تقوم بالعمل الذي تمليه كل تعليمة فإذا احتاجت هذه الوحدة (EU) إلى معلومة مخزنة في الذاكرة فإنها تأمر وحدة ملائمة الممرات BIU بإحضارها و ذلك عن طريق إعطائها عنوان هذه المعلومة في الذاكرة.

إن من أحد أهم وظائف EU هو تنفيذ العمليات الحسابية و المنطقية على المعلومات، و أثناء سير التنفيذ تقوم EU بفحص مسجل الأعلام بعد كل تعليمة (مسجل الأعلام : هو عبارة عن ستة عشر بت تعبر عن حالة المعالج بعد تنفيذ كل تعليمة) .

مسجلات الأغراض العامة هي ثمانية مسجلات طول كل مسجل منها 2 بايت و هذه المسجلات هي AX,BX,CX,DX,SI,DI,BP,SP .

بنية الذاكرة

تتألف الذاكرة من حجرات متسلسلة سعة كل منها 8 بت (واحد بايت) ، ترقم هذه الحجرات من الصفر و حتى نهاية الذاكرة و يستخدم النظام الست عشري عادة في عملية الترقيم و بذلك يكون لكل حجرة رقم يميزها عن غيرها، يدعى هذا الرقم بعنوان تلك الحجرة.

يوضع داخل كل حجرة رقم ست عشري يتراوح بين 0 و FF و يدعى هذا الرقم بمحتوى تلك الحجرة.

يوجد بين المعالج و الذاكرة ممران هما ممر المعطيات بعرض 16 بت و ممر العناوين بعرض 20 بت.

فمثلاً عندما يحتاج المعالج إلى القيمة المخزنة في الحجرة ذات الرقم 100 (عنوانها 100) فإن الرقم 100 يمثل بشكل ثنائي و يوضع على ممر العناوين و يرسل إلى الذاكرة، و حالما تستلم الذاكرة هذا العنوان فإن محتوى الحجرة 100 يرسل إلى المعالج عن طريق ممر المعطيات.

إن كون ممر العناوين ذو عرض 20 بت (20 خط نقل) هذا يعني أنه يستطيع نقل رقم ثنائي ذو 20 خانة أي أن أكبر قيمة يمكن وضعها على ممر العناوين هي :

$$2^{20} = 1048576 \approx 1MB$$

و بذلك يستطيع المعالج 8086 عنونة واحد ميغا من الذاكرة فقط.

مقاطع الذاكرة (هذه الفقرة مرتبطة ارتباطاً وثيقاً بالمسجلات)
يتعامل المعالج كما ذكرنا مع واحد ميغا من الذاكرة، و يمكن أن نقتطع من هذه الميغا أربعة مقاطع أساسية يتعامل معها برنامجنا بشكل مباشر (أي أنه لا تتم الاستفادة من كل الذاكرة بآن واحد) و هذه المقاطع الأربعة هي:

1) مقطع الشيفرة Code Segment CS

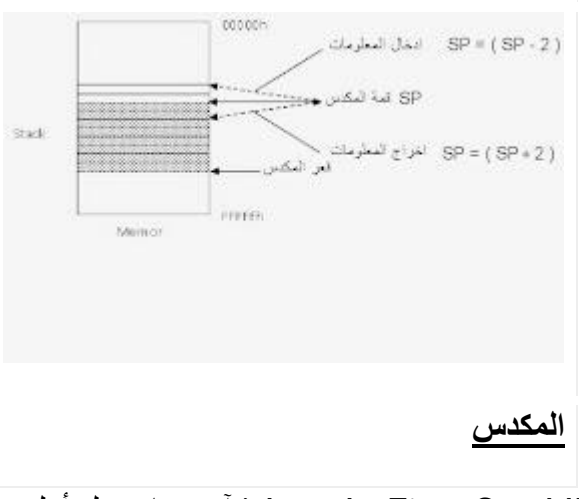
يخصص هذا المقطع من الذاكرة –كما هو واضح من تسميته- لتخزين شيفرة البرنامج. و هناك مسجل له نفس الاسم CS موجود في المعالج يحتفظ بقيمة تدل على بداية هذا المقطع في الذاكرة و يساعده المسجل IP (Instruction Pointer) الذي يحتفظ بعنوان التعليمة التي ستنفذ الآن و تعدل قيمته آلياً ليشير إلى عنوان التعليمة التالية.

2) مقطع المعطيات Data Segment DS

يخصص هذا المقطع من الذاكرة لتخزين المعطيات و المتحولات. و هناك مسجل له نفس الاسم DS موجود في المعالج يحتفظ بقيمة تدل على بداية هذا المقطع في الذاكرة و يساعده المسجل SI الذي يشير إلى الإزاحة بالنسبة إلى بدايته.

) مقطع المكدس Stack Segment SS

يخصص هذا المقطع للحفظ المؤقت لبعض المعلومات الضرورية و التي يخشى أن تضيع أو تتغير أثناء تنفيذ برنامج ما. و هناك مسجل له نفس الاسم SS موجود في المعالج يحتفظ بقيمة تدل على بداية هذا المقطع في الذاكرة.



المكدس

آلية عمل المكدس Last In First Out LIFO (آخر ما يدخل أول ما يخرج) : أي أن أول عنصر يدخل إلى المكدس يصبح في قعره و آخر عنصر يدخل المكدس يصبح في قمته و يتم سحب المعلومات من المكدس من قمته حيث لدينا مسجل اسمه Stack Pointer SP يشير دوماً إلى قمة المكدس فهو يتغير حسب الحالة التي يتم بها التعامل مع المكدس (إدخال

معلومات أو إخراج). فعند إدخال معلومة بطول 2 بايت فإن قمة المكدس تقترب من بداية الذاكرة (انظر الشكل) و بذلك تنقص قيمة SP بمقدار 2 لأن إملاء المكدس يعني الاقتراب من العنوان الأصغر و العكس بالعكس أي عندما نسحب معلومة من المكدس فإن قمته تبتعد عن بداية الذاكرة و بذلك تزيد SP بمقدار 2 لأن إفراغ المكدس يعني الاقتراب من العنوان الأكبر.

4) مقطع المعطيات الإضافي Extra Segment ES

يستخدم عند الحاجة إلى استخدام مقطعي معطيات بنفس الوقت و بذلك نستطيع الاستفادة من مساحة أكبر في الذاكرة. و يساعده المسجل Destination Index DI الموجود في المعالج و الذي يشير إلى الإزاحة بالنسبة إلى بدايته.

ملاحظة: يجب التمييز بين المقطع و مسجل المقطع حيث المقطع هو جزء من الذاكرة بينما مسجل المقطع يتألف من بايتين و هو موجود في المعالج.

عن بداية الذاكرة و بذلك تزيد SP بمقدار 2 لأن إفراغ المكدس يعني الاقتراب من العنوان الأكبر.

المسجلات Registers

يملك المعالج 8086 أربعة مجموعات من المسجلات ذات 16 بت يستطيع المبرمج الوصول إليها و هي:

1) مؤشر التعليمة IP
2) أربعة مسجلات معطيات AX,BX,CX,DX .
3) أربعة مسجلات تأشير و فهرسة SI,DI,BP,SP .
4) أربعة مسجلات مقاطع CS,DS,SS,ES .

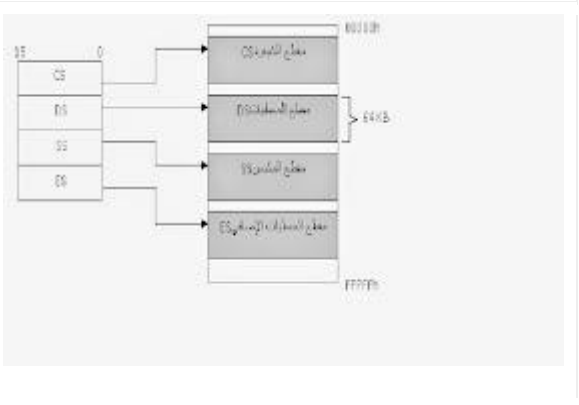
بالإضافة إلى ذلك يوجد مسجل آخر هو مسجل الأعلام و يدعى أيضاً مسجل الحالة و هو مسجل ذو 16 بت و لكن نستخدم منه 9 خانات فقط.

سنشرح كلٍ من هذه المسجلات بالتفصيل :

المجموعة الأولى : مسجلات المقاطع
و هي عبارة عن أربعة مسجلات طول كل منها 16 بت أي 2 بايت و هي :

1) مسجل مقطع الشيفرة CS : يحتوي على عنوان أول حجرة في مقطع شيفرة البرنامج في الذاكرة، أي أنه يشير إلى بداية مقطع الشيفرة.
2) مسجل مقطع المعطيات DS : يحتوي على عنوان أول حجرة في مقطع المعطيات في الذاكرة، أي أنه يشير إلى بداية مقطع المعطيات.
3) مسجل مقطع المكدس SS : يحتوي على عنوان أول حجرة في مقطع المكدس في الذاكرة، أي أنه يشير إلى بداية مقطع المكدس.

4) مسجل مقطع المعطيات الإضافي ES: يحتوي على عنوان أول حجرة



مسجل مقطع المعطيات

1) في مقطع المعطيات الإضافي في الذاكرة، أي أنه يشير إلى بداية مقطع المعطيات الإضافي.

المجموعة الثانية: مسجلات الفهرسة و التأشير

و هي عبارة عن أربعة مسجلات مساعدة تساعد في إيجاد العنوان الفيزيائي بالتعاون مع مسجلات المقاطع، و طول هذه المسجلات 16 بت أي 2 بايت، و هي :

1) مسجل دليل المصدر Source Index SI : يخزن فيه عنوان يدل على الإزاحة ضمن مقطع المعطيات DS و بمعنى آخر يستعمل في إمساك العناوين الفعالة من أجل التعليمات التي تتناول المعطيات المخزنة في مقطع المعطيات في الذاكرة.

2) مسجل دليل الهدف Destination Index DI : يخزن فيه عنوان يدل على الإزاحة ضمن مقطع المعطيات الإضافي ES ، و بمعنى آخر يستعمل مسجل دليل الهدف DI من أجل استنتاج العنوان الفيزيائي الذي يحدد حجرة متحول الهدف.

3) مسجل مؤشر المكدس Stack Pointer SP : يسمح مؤشر المكدس بوصول سهل للحجرات في مقطع المكدس الموجود في الذاكرة حيث أن القيمة في SP تمثل العنوان الفعال لحجرة المكدس التالية التي يمكن الوصول إليها نسبة إلى العنوان الحالي الموجود في مسجل مقطع المكدس SS و يحتفظ SP دوماً بقيمة تدل على قمة المكدس ، هذا و إن قيمة هذا المسجل تتعدل تلقائياً عند وضع أو سحب معلومة بالمكدس.

4) مسجل مؤشر القاعدة Base Pointer BP : يحوي قيمة تدل على الإزاحة بالنسبة لمقطع المكدس SS و هو يستخدم لقراءة المعطيات ضمن مقطع المكدس بدون إزالتها من المكدس.

المجموعة الثالثة: مسجلات المعطيات

تستخدم هذه المسجلات من أجل التخزين المؤقت للنتائج المرحلية أثناء تنفيذ البرنامج حيث أن تخزين المعطيات في هذه المسجلات يمكننا من الولوج إلى تلك المعطيات بشكل أسرع مما لو كانت في الذاكرة، و تقسم المسجلات إلى :

1) مسجل المراكم Accumulator و يرمز له بالرمز A .

2) مسجل القاعدة Base و يرمز له بالرمز B .

3) مسجل العد Count و يرمز له بالرمز C .

4) مسجل المعطيات Data و يرمز له بالرمز D .

و كل مسجل من المسجلات السابقة يمكن استعماله إما ككلمة 16 بت و يدل على ذلك بكتابة الحرف X بعد اسم المسجل أو يمكن استعماله كبايتين كل منهما 8 بت و يدل على ذلك باستخدام الحرفين H,L حيث :

L للبايت ذو العنوان الأصغر ، مثال AL .

H للبايت ذو العنوان الأكبر ، مثال BH .

هكذا و إن كلاً من هذه المسجلات يمكن استخدامه من أجل التعليمات الرياضية أو المنطقية في لغة الأسمبلي مثل And, Add .

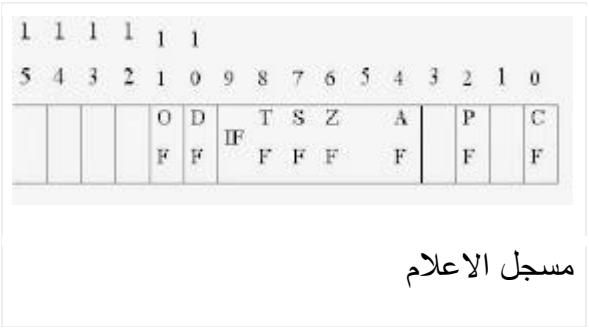
و من أجل بعض التعليمات مثل البرامج التي تحتوي على تعليمات سلاسل فإنها تستعمل مسجلات معينة مثل استعمال المسجل C لتخزين العدد الذي يمثل عدد البايتات التي ستنفذ عليها تعليمات السلاسل (عدد مرات تكرار تعليمة السلسلة)

مسجل مؤشر التعليمة Instruction Pointer IP

هذا المسجل يحدد موقع التعليمة التالية التي ستنفذ في مقطع الشيفرة و بعد جلب شيفرة التعليمة من الذاكرة فإن BIU تعدل قيمة IP بحيث تشير إلى التعليمة التالية في الذاكرة (التعديل يتم آلياً).

مسجل الأعلام Flags Register

هو مسجل ذو 16 بت موجود في وحدة التنفيذ كما هو واضح بالشكل :



مسجل الاعلام

و كما نلاحظ من الشكل السابق أنه يوجد ستة أعلام للحالة هي CF, PF, AF, ZF, SF, OF ، و كذلك يوجد ثلاثة أعلام للتحكم DF, IF, TF .

أ) أعلام الحالة

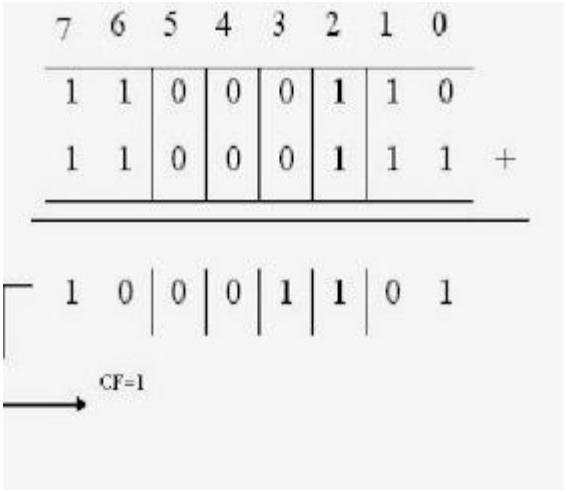
تشير إلى الحالات الناتجة كنتيجة لتنفيذ تعليمة منطقية أو رياضية حيث تكون إما في حالة واحد منطقي Set أو تكون في حالة صفر منطقي Reset ، و سنلخص فيما يلي عمل كلٍ منها:

أولاً: علم الإنزياح Carry Flag

يكون في حالة الواحد المنطقي إذا وجد انزياح خارجي (حمل) أو استعارة من أجل الخانة الأخيرة (البت الأخير) و ذلك أثناء تنفيذ التعليمات الرياضية.

و يكون في حالة الصفر المنطقي إذا لم يوجد حمل أو استعارة من أجل البت الأخير.

أمثلة:

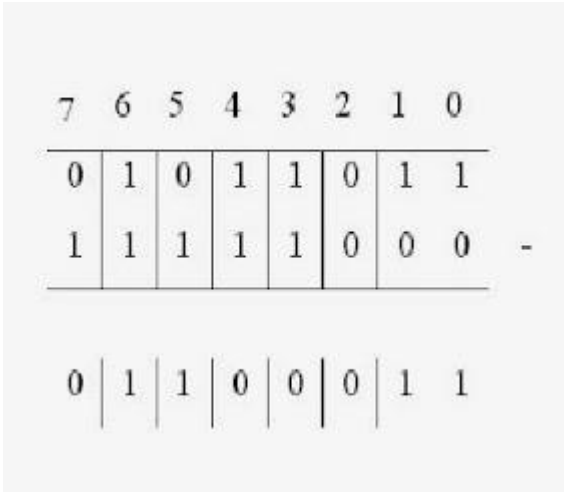


لاحظ بأن النتيجة لم تتسع في ثمانية بتات و إنما تحتاج إلى تسع بتات و نعبر عن ذلك بثمانية بتات و CF=1 أي أنه لدينا في اليد واحد.

ببساطة: فمهما كَبُرَ العددان فإن تسعة بتات يمكن أن تستوعبها.

ثانياً: حالة الاستعارة

لاحظ بأن العدد الأول الممثل ثنائياً أصغر من العدد الثاني الممثل ثنائياً أيضاً ، لذلك فعند إجراء عملية الطرح و في مثالنا هذا تخيلنا بت تاسع فيه القيمة واحد (استعرنا) و بالتالي فإن CF=1 أي لدينا استعارة من أجل البت الأعلى رتبة



و في المثالين السابقين نطبق نفس الكلام من أجل 2 بايت و لكن الإنزياح الخارج و الاستعارة تكون من أجل البت الخامس عشر (الأخير).

ثانياً: علم الازدواجية Parity Flag PF

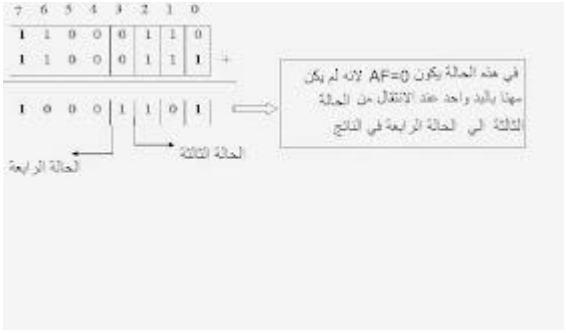
يصبح في حالة واحد منطقي إذا كانت نتيجة آخر تعليمة تحوي على عدداً زوجياً من الخانات الواحدية (بعد التحويل إلى النظام الثنائي طبعاً) و إلا يكون في حالة الصفر المنطقي.

نلاحظ أن علم PF يفحص البايت السفلي فقط حتى لو كنا نتعامل مع كلمة (2 بايت) ، أما عندما نتعامل مع بايت واحد فقط فإنه يفحصه كله.

ثالثاً: علم الإنزياح المساعد Auxiliary Flag AF

يكون في حالة الواحد المنطقي إذا وجد إنزياح من النصف السفلي إلى النصف العلوي أو استعارة من النصف العلوي إلى النصف السفلي و ذلك من أجل البايت السفلي من الكلمة (2 بايت) و بمعنى آخر أنه إذا كان لدينا إنزياح من الخانة 3 إلى الخانة 4 فإن AF=1 و ذلك في حال كانت المعطيات بايت واحد أو بايتين (كلمة)، و فيما عدا ذلك يكون AF=0 .

مثال:



رابعاً: علم الصفر Zero Flag ZF

يصبح في حالة واحد منطقي عندما يكون ناتج آخر عملية حسابية أو منطقية يساوي الصفر.

يصبح في حالة صفر منطقي عندما يكون ناتج آخر عملية حسابية أو منطقية لا يساوي الصفر.

خامساً: علم الإشارة Sign Flag SF

يكون علم SF في حالة واحد منطقي Set إذا كانت نتيجة آخر عملية حسابية عدداً سالباً.

يكون علم SF في حالة صفر منطقي Reset إذا كانت نتيجة آخر عملية حسابية عدداً موجباً.

مصطلح: من إحدى طرق تمثيل الأعداد السالبة في الكمبيوتر هي اعتبار الخانة الأخيرة مخصصة للإشارة و بما أن البايت مكون من ثمانية خانات فسيتم اقتطاع الخانة الأخيرة منه من أجل الإشارة فإن احتوت على القيمة واحد فإن الخانات السبعة الباقية هي عدد ثنائي سالب أما إذا احتوت على القيمة صفر فإن الخانات السبعة المتبقية ما هي إلا عدد موجب.

و بذلك يكون SF هو نسخة عن الخانة الأخيرة في الناتج عند اعتماد هذا النظام لتمثيل الأعداد السالبة.

لاحظ أنه انطلاقاً من هذا المبدأ في التمثيل يمكننا تمثيل المجالات التالية من الأعداد:

من أجل بايت واحد من –128 إلى +127

من أجل بايتين من –32768 إلى +32767

سادساً: علم الطفحان Overflow Flag OF

يكون في حالة واحد منطقي عندما لا تتسع النتيجة في المكان المخصص لتخزينها أي تتجاوز القدرة التخزينية، أما إذا لم تكن النتيجة خارج المجال المحدد فإن OF يبقى في حالة الصفر المنطقي.

يحدث الطفحان في الحالات التالية:

1) جمع أعداد موجبة كبيرة.

2) جمع أعداد سالبة كبيرة.

3) طرح عدد موجب كبير من عدد سالب كبير.

4) طرح عدد سالب كبير من عدد موجب كبير.

ملاحظة: جميع الأعلام السابقة ما عدا CF تُقرأ فقط أي لا نستطيع تغيير محتواها لذلك يمكن قراءتها فقط و لا يمكن تغيير محتوياتها بواسطة تعليمات برمجية مباشرة.

المعالج مزود بتعليمات تستطيع اختبار حالة هذه الأعلام لتغيير تتابع تنفيذ البرنامج فمثلاً يمكن اختبار علم ZF=1 كشرط من أجل القفز إلى جزء آخر من البرنامج.

و فيما يلي سنشرح أعلام التحكم:

أولاً: علم الخطوة الوحيدة Trap Flag TF

يوضع بالحالة واحد منطقي عندما نرغب بتنفيذ البرنامج خطوة خطوة و هو مفيد عندما نريد تصحيح برنامجنا و استكشاف مواقع الأخطاء.

ثانياً: علم المقاطعة Interrupt Flag IF

يستخدم من أجل التعبير عن إمكانية أو عدم إمكانية تنفيذ المقاطعة، فيوضع بالحالة واحد منطقي عندما لا نرغب بتنفيذ أي مقاطعة (المقاطعة محجوبة) أما عند وضعه في حالة الصفر المنطقي فإن المقاطعة مسموح بها.

ملاحظة: المقاطعة هي عبارة عن خدمة تؤدي إلى عمل معين فمثلاً المقاطعة 21 و التي من أحد خدماتها العودة إلى نظام التشغيل.

ثالثاً: علم الاتجاه Direction Flag DF

يدل على اتجاه سير العمليات التسلسلية.

عندما يكون في حالة واحد منطقي فإن السلسلة تكون من العنوان الأعلى إلى العنوان الأدنى.

عندما يكون في حالة صفر منطقي فإن السلسلة تكون من العنوان الأدنى إلى العنوان الأعلى.

مفهوم العنوان الفيزيائي و الإزاحات

توجد عدة طرق لنقل السجلات

1- MUX -2 DEC

- (MUX) :- ماهو ؟

1- عدد MUX يتناسب مع عدد البت

مثال :- 4BIT = 4 / 5BIT=5

2- حجم MUX ← يتناسب مع عدد REGISTER

عدد المداخل ← يتناسب مع عدد REGISTER

عدد SELECTOR ← يتناسب مع عدد REGISTER

ت	INPUT	SELECTOR
1	2 INPUT	1 SELECTOR
2	4 INPUT	2 SELECTOR
3	8 INPUT	3 SELECTOR
4	16 INPUT	4 SELECTOR
5	32 INPUT	5 SELECTOR
6	64 INPUT	6 SELECTOR
7	128 INPUT	7 SELECTOR

7 REGISTER

1- **حجم MUX**

حجم MUX = 1*8

2- **عدد المداخل**

عدد المداخل = 8

3- **عدد SELCTOR**

عدد (SELECTOR) = 3

- **سوف نتكلم عن :- decoder**

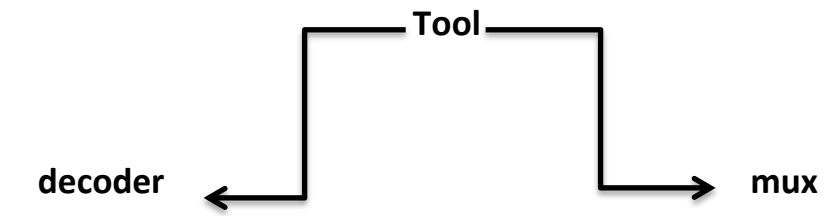
توجد عدة طرق لنقل السجلات .

1- **Mux**
2- **Decoder**

نفترض عندي 9 رجستير استخرج لي حجم mux
وكذلك حجم decoder

حجم mux = عدد المداخل عن طريق عدد الرجستر * 1

حجم decoder= عدد المداخل * عدد سلكتر



mux = register to bus

decoder = bus to register

R2 ⟵ BUS

BUS ⟵ R1

اشرح لي او فسر لي هذه العملية ؟

ج/ هي عملية نقل المحتوى من R2 الى BUS باستخدام MUX وبنفس الوقت نقل المحتوى من BUS الى R2 باستخدام DECODER