2.1 Processor Organization

We will consider the processor organization of 8086 processor, which can see it diagram in Figure -1-

سيتم اعتماد هيكيلة معالج 8086 وكما موضح في الرسم ١

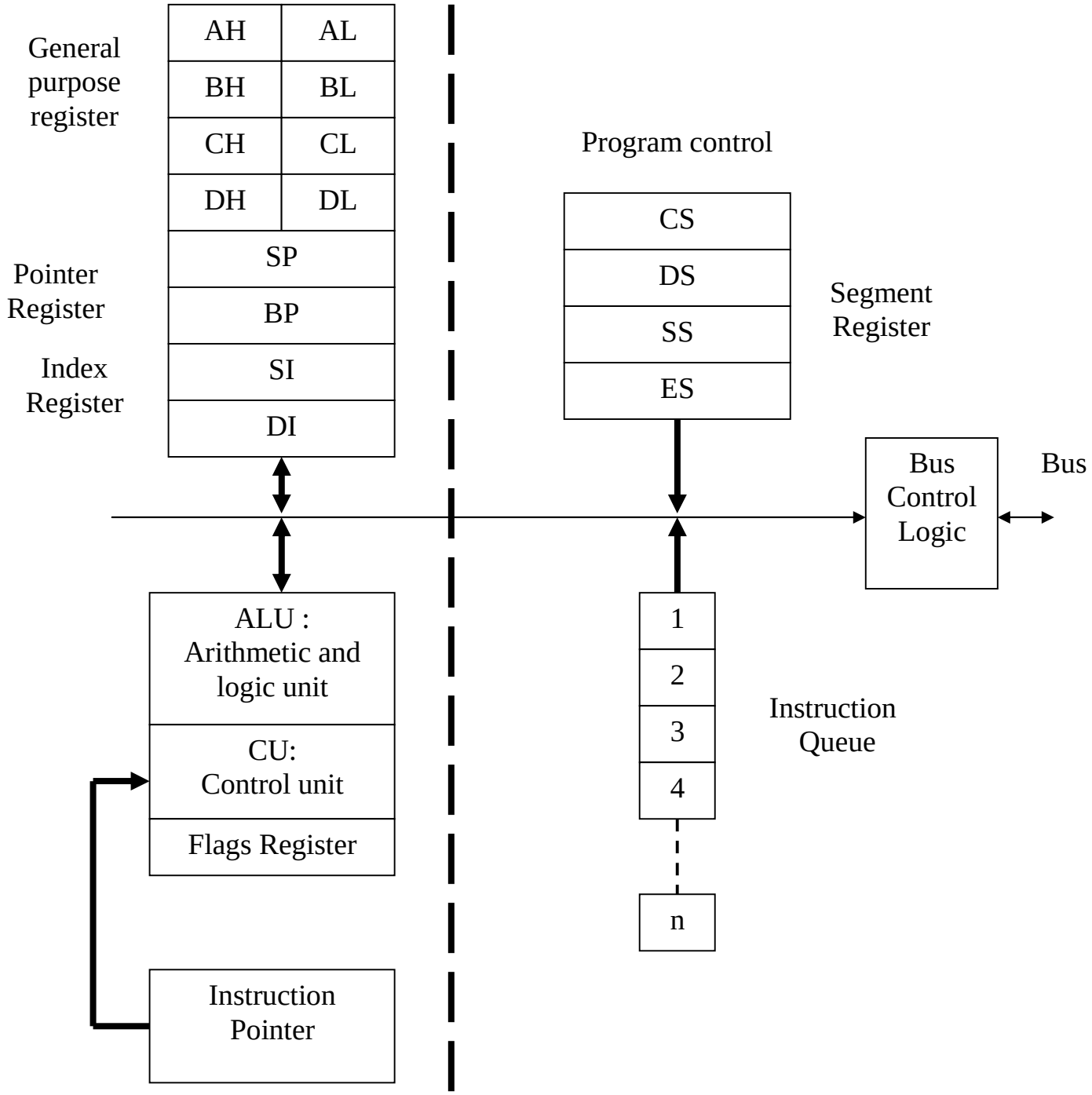


Fig -1- Execution Unit and Bus Interface Unit
8086 Processor Organization

The organization of 8086 Processor is partitioned into two logical units

a. Bus Interface unit (BIU)

b. Execution Unit (EU)

يتم تقسيم هيكيلة المعالج 8086 الى وحدتين منطقتين هما:

أ. وحدة جلب الايعاز

ب. وحدة التنفيذ

2.1.a. Bus Interface Unit (BIU)

The role of the BIU delivers instructions and data to the EU. The most important function of the BIU is to manage the Bus control unit which response of making synchronization between the CPU and the Device is connected to it, by reading it specification. And the other function of the Bus Control Unit is specify which device is connected when more than one device is request an service from CPU at the same time with respect to priority specified from the designer or the user.

أ. وحدة جلب الايعاز

ان الدور الرئيسي لوحدة جلب الايعاز هو جلب الايعازات والبيانات الى وحدة التنفيذ . ان الوظيفة الرئيسية هو تنظيم وحدة السيطرة على الـــBus والتي تكون مسوؤلة عن عمل تزامن بين المعالج والجهاز المربوط معه، من خلال قراءة المواصفات الخاصة بالجهاز. والوظيفة الاخرى لوحدة السيطرة على الـــ Bus هو تحديد اي جهاز يرتبط عندما يطلب اكثر من جهاز خدمة في نفس الوقت من خلال افضليات تحدد من قبل المصمم او المستخدم.

Another function of the BIU is to provide access to instructions. Because the instructions for a program that is executing are in memory, the BIU must access instructions from memory and place them in an instruction queue, which varies on size from 5 instructions to 7 instructions depending on the processor. This feature enables the BIU to look ahead and prefetch instructions so that there is always a queue of instructions ready to execute.

ان الوظيفة الاخرى لوحدة جلب الايعاز هو عمل معالجة للايعازات. وبسبب ان الايعازات لبرنامج والذي يتم تنفيذه يكون موجود في الذاكرة ، وبالتالي يجب ان تقوم وحدة جلب الايعاز بجلب الايعازات من الذاكرة ووضعها في طابور الايعازات والذي يترواح طوله بين ٥ ايعازات الى ٧ ايعازات بالاعتماد على نوع المعالج. وهذه الخاصية تمكن وحدة

جلب الايعازات من تحضير مجموعة ايعازات جاهزة للتنفيذ وقد تم جلبها مسبقا من الذاكرة وبالتالي توفير طابور من الايعازات الجاهزة للتنفيذ.

Manage the segment register and controlling the process of addressing the memory is another function of BIU. The segment register is considering the most important set of registers in 8086 CPU and we have four types of segments:

تنظيم مسجلات التجزئة والسيطرة على عملية عنونة الذاكرة هو وظيفة اخرى لوحدة جلب المعلومات . تعتبر مسجلات التجزئة هي اهم مجموعة مسجلات ضمن مسجلات المعالج 8086 وهناك اربعة انواع من مسجلات التجزئة :

1. Code segment register

شريحة ومسجل تجزئة البرنامج

2. Data segment register

شريحة ومسجل تجزئة البيانات

3. Stack segment register

شريحة ومسجل تجزئة المكدس

4. Extra segment register

شريحة ومسجل تجزئة الاضافي

Let us first define the segment as a special area defined in a program that begins on paragraph boundary, that is, at a location evenly divisible by 16, or hex 10. Although a segment may be located, almost anywhere in memory and in real mode (executing one program in one time) the size of segment is 64 KB and the size of the space it takes in memory is equal to the space required for the program execution. In addition, we note that there is another mode of operating is called protected mode (executing more than one program in one time) which can process working in it.

في البداية لنعرف المقطع والذي هو عبارة عن منطقة خاصة معرفة من فبل البرنامج والتي تبدا بما يعرف بال Paragraph boundary وهوعنوان موقع في الذاكرة يقبل القسمة على 16 او 10 في النظام السادس العشري . ويمكن

اين يكون المقطع في اي مكان بالذاكرة وفي نمط التشغيل المعالج (real mode) (تنفيذ نرنامج واحد في وقت واحد فان حجم المقطع سيكون 64KB وسيكون حجم الماخوذ من الذاكرة

Let us exam the four types of segment and begin with

1. Code segment

The code segment contains the machine instructions that are to execute. Typically, the first execute instruction is at start of this segment, and the operating system links to that location to begin program execution. As the name implies, the code segment (CS) register addresses the code segment. If your code area requires more than 64KB, your program may need to define more than one code segment.

ان مقطع الــCode يحتوي ايعازات الالة التي سيتم تنفيذها . عمليا ان الايعاز التنفيذي الاول سيكون في بداية المقطع وسيرتبط نظام التشغيل مع هذا الموقع المخزون به الايعاز الاول ليتم تنفيذ البرنامج. وسيكون هناك مسجل بنفس الاسم يحتفظ بعنوان بداية المقطع. واذا كان حجم البرنامج اكبر من 64KB فيتم تعريف اكثر من مقطع للــCode .

2. Data Segment

The data segment contains a program's defined data, constants, and work areas. The data segment (DS) register addresses the data segment. If your data area requires more than 64KB, your program need to defines more than one data segment.

يحتوي مقطع البيانات على البيانات المعرفة من قبل البرنامج ، الثوابت ، مناطق العمل. ويعنون مسجل خاص لهذا الغرض مقطع البيانات . واذا كان مقطع يحتاج اكثر من 64KB فيتم توليد اكثر من مقطع بيانات.

3. Stack segment

In simple terms, the stack contains any data and addresses that you need to save temporarily during an execute a subroutines related to the main program. The stack segment register addresses the stack segment.

بكلمات بسيطة، ان المكدس (Stack) يحتفظ بشكل مؤقت باي بيانات او عنوانين يحتاجها البرنامج الرئيسي اثناء تنفيذ برنامج فرعي مرتبط به. وهناك مسجل خاص يعنون مقطع الــ Stack .

Segment Boundaries

The segment is begin with an address called paragraph boundary which mean an address begin with digit 0 from the right side of 5 digits address .Therefore, it will be choose an location with address starting with zero digit. Because of the length of any segment is 64KB, which equal to 65536 locations and when convert to hexa after subtract one of the number the result, is FFFFH.

يبدا المقطع بعنوان يسمى paragraph boundary والذي يعني ان العنوان يبدا برقم صفر من جهة اليمين لعنوان من خمسة ارقام ، وبالتالي فانه سيختار عنوان موقع يبداء بالرقم صفر. وبسبب ان حجم المقطع هو 64KB والذي يساوي ٦٥٥٣٦ موقع وعندما تحول الى النظام السادس عشري مطروح منها واحد فان الناتج مساوي الى FFFFH.

The segment begins with address 0000H to FFFFH as the offset inside the segment. Therefore, we will use a set of register to locate the staring address of segment and can define as:

A segment register is 16 bits long and provides for addressing an area of memory known as the current segment, to reduce the complexity of processor register it store 4 digits instead of 5 digits and insert 0 digit from right of the address of segment register.

يبدا المقطع بالعنوان 0000H الى FFFFH وكمافة ازاحة داخل المقطع. وبالتالي سيتم استخدام مجموعة من المسجلات والتي تحتفظ بعنوانين البداية للمقاطع ويمكن تعريف مسجل المقطع: بانه مسجل بطول 16bit والتي تستخد لعنونة مساحة من الذاكرة تسمى المقطع الحالي ، ومن اجل تقليل تعقيد مسجل المعالج فانه يخزن اربعة ارقام بدلا من خمسة مع اضافة صفر من جهة اليمين عند التعامل مع القيمة الموجودة ضمن المسجل.

Let ' take an overview about segment registers:

لنلقي نظرة على انواع مسجلات المقطع

1. Code Segment register (CS): contains the starting address of a program's code segment. For normal programming purpose, you need not reference the CS register.

مسجل مقطع البرنامج وهو يحوي على عنوان البداية لمقطع البرنامج . وفي البرمجة العادية لايتم الاشارة الى هذا المسجل.

2. Data Segment register (DS): contains the starting address of a program data segment. Instructions use this address to locate data.

مسجل مقطع البيانات وهو يحوي على عنوان البداية لمقطع البيانات والايعازات تستخدم هذا العنوان لتحديد موقع البيانات.

3. Stack Segment register (SS) Permits the implementation of a stack in memory, which a program uses for temporary storage of address of a program's stack segment in the SS register. For normal programming purpose, you need not directly reference the SS register.

مسجل مقطع المكدس والذي يسمح باستخدام المكدس في الذاكرة والذي يستخدم من قبل البرنامج للخزن المؤقت وهو يحتفظ بعنوان بداية مقطع المكدس.وفي البرمجة العادية لايتم الاشارة الى هذا المسجل.

4. Extra Segment register (ES): used by some string (character data) operations to handle memory addressing. The ES (Extra segment) register is associated with DI (index register). A program that requires the use of the ES may initialize with an appropriate segment address.

مسجل المقطع الاضافي وهو يستخدم مع عمليلات السلاسل الحرفية عندما تستخدم عنوانين الذاكرة. وترتبط مسجل ES مع مسجل الفهرسة DI .وهي تولد ان احتاج البرنامج التعامل مع السلاسل الحرفية.

5. FS AND GS registers Additional extra segment registers on the 80386 and later processors.

Management OF Generated Segments

تنظيم توليد المقاطع

Example: Find the configuration of segments generated by two programs , first Program A deal with arithmetic operations and has 800KB in its programming section and 600KB in its data sections and 900 KB in stack sections while second Program B has 1048 KB code and 660KB data and 2066 KB as stack memory. And Find the total number of segments generated.

جد كيف يتم تنظيم المقاطع المولدة من خلال برنامجين الاول يتعامل مع عمليات رياضية ويمتلك 800KB في جزء البرمجة و 600KB لجزء البيانات و 900KB لجزء Stack بينما يمتلك البرنامج الثاني 1048 كـــ Code و 660KB كـــ Data و 2066KB كـــ Stack . وجد عدد المقاطع الكلية المتولدة من قبل البرنامجين.

Sol

The default size of any 64 KB and we have three types of segments

ان الحجم الاصلي للمقطع هو 64KB ولدينا ثلاثة انواع

Code segment , Data segment , Stack segment

Therefore the Program A has these segments

Number of segments = size of section / size of one segment

No. of code segments = 800KB / 64 KB= 12.5 =13 segments

No. of data segments = 600KB / 64 KB= 9.375 =10 segments

No. of stack segments = 900KB / 64 KB= 14.0625 =15 segments

And for Program B has these segments

No. of code segments = 1048KB / 64 KB= 16.375 =17 segments

No. of data segments = 660KB / 64 KB= 10.3125 =11 segments

No. of stack segments = 2066KB / 64 KB= 32.28125 =33 segments

2.1.b. Execution Unit (EU)

The role of the EU is to execute instructions, whereas the BIU delivers instructions and data to the EU. The EU contains an arithmetic and logic unit (ALU), a control unit (CU), and a number of registers. These features provide for execution of instruction and arithmetic and logical operations.

ان الهدف الرئيسي من وحدة التنفيذ هو تنفيذ الايعازات ، وحيث ان وحدة جلب الايعازات تقوم بجلب الايعازات والبيانات من الذاكرة الى وحدة التنفيذ. وتحتوي هذه الوحدة على وحدة الحساب والمنطق ووحدة السيطرة وعدد من المسجلات ، وهذه المكونات توفر امكانية تنفيذ العمليات الرياضية والمنطقية .

We can classify the components of EU to :

1. Pointers Registers

مسجلات التاشير

The Three pointer registers are the IP, SP, and BP.

هناك ثلاثة انواع من مسجلات التاشير وهي IP,SP,BP.

Instruction Pointer (IP) register

The 16-bit IP register contains the offset address of the next instruction that is to execute. The IP is associated with CS register in that The IP indicates the current instruction within the currently executing code segment. You do not normally reference the IP register in program, but you can change its value when using the DEBUG program to test a program. The 80386 and later processors have an extended 32-bit IP called the EIP.

يحتوي مسجل IP 16-bit على عنوان الازاحة للايعاز التالي في التنفيذ ، وبارتباط مسجل IP مع مسجل CS حيث يتولد عنوان الايعاز الحالي ضمن مقطع البرنامج الذي ينفذ حاليا. ولايمكن التحكم بقيمة مسجل IP بشكل اعتيادي من قبل المستخدم، ولكن يمكن تغيير قيمته في حالة استخدام برنامج DEBUG والذي يقوم بتدقيق البرنامج من الاخطاء حيث يمكننا من جعل البرنامج ينفذ خطوة خطوة والتحكم بالمسجل IP. ومعالجات 80386 والمعالجات التي بعدها تحتوي على مسجل موسع بطول 32-bit ويسمى EIP.

Stack Pointer (SP) register

The 16-bit SP register provides an offset value, which when associated with the SS register, refers to the current word being processed in the stack. The 80386 and later processors have an extended 32-bit stack pointer, the ESP register. The system automatically handles these registers.

يقدم مسجل SP bit – 16 الازاحة والتي تربط مع مسجل SS لتعنون الكلمة الحالية الموجودة في stack. تمتلك معالجات 80386 والمعالجات التي بعدها مسجل ب32 بت اي مسجل موسع من نوعية stack pointer والمعروف ESP. والنظام اليا يتعامل مع هذه المسجلات.

Base Pointer (BP) register

The 16 – bit BP facilitates referencing parameters, which are data and addresses that a program passes via the stack. The processor combines the address in the SS with the offset in the BP. The 80386 and later processors have an extended 32-bit BP called the EBP register

يستخدم مسجل BP ذو 16 bit يؤشر المعاملات، وهي مجموعة البيانات والعنوانين والمررة الى stack. ان المعالج يجمع بين العنوان في المسجل SS مع الازاحة الموجود في BP. والمعالجات 80386 والتي بعدها تمتلك النوع الموسع من هذا المسجل ذو 32 بت والمسمى EBP.

2.Addressing Using Pointer Registers and Segment Registers

We have rules for determine the final address of information founded in Memory and we have four different kinds of as:

1. Final address for CODE: value in CS with digit 0 from right + IP
1. Final address for DATA: value in DS with digit 0 from right + Address in the instruction
1. Final address for WORD in Stack: value in SS with digit 0 from right + SP
1. Final address for PARAMETER in Stack: value in SS with digit 0 from right + BP

EX1 Find the final address for these information's:

1. CODE if you know that CS=1FE4 and IP=BB6A
2. DATA if you know that DS=55862_{10} and the address in the instruction is 4237_8
3. WORD in Stack if you know that SS=110110111100110_2 and SP=5678_{16} and found the Final address for PARAMETER if BP=36985_{10}.

SOL:

1. Final address for CODE: CS0 + IP

```
  1FE40
+ BB6A
-------
  2B9AA
```

2. Final address for Data : DS0+Address in the instruction

The DS is in the decimal system and we must convert it to the hexadecimal as fllow

16	55862	
16	3491	6
16	218	3
16	13	A
16	0	D

Therefore the value of DS=DA36

And the of the address in the instruction is in Octal and we must converted into Hexadecimal

4 2 3 7

100 010 011 111

8 9 F

Therefore, the value of address in the instruction is 089F

And the final of address of Data is:

```
   DA360
+   089F
---------
  DABFF
```

3. Final address of WORD in the Stack: SS0 + SP

The value of SS in the binary Form and we must converted into Hexadecimal form

0110110111100110

6 D E 6

Therefore, the value of SS=6DE6

```
   6DE60
+   5678
---------
   734D8
```

And final address for PARAMETER is SS0+BP

BP= $36985_{10} \longrightarrow 9079_{16}$

```
   6DE60
+   9079
---------
   76ED9
```

3.General – Purpose Registers

The AX, BX, CX and DX general – purpose registers are the workhorses of the system. They are unique in that you can address them as one word or as a 1- byte portion. The leftmost byte is the "high" portion and the rightmost byte is the "low" portion. For example, the AX register consists of an AH (high) and an AL (low) portion, and you can reference any portion by its name. The 80386 and later processors support all the general – purpose registers, plus 32-bit extended versions of them: the EAX, EBX, ECX, and EDX.

تعتبر المسجلات AX, BX, CX and DX مسجلات الاستخدام العامة ومناطق العمل في النظام . وهي المسجلات الوحيدة ضمن مجموعة مسجلات معالج 8086 التي يمكن ان تقسم الى جزئين كل منها بحجم بايت واحد ويسمى الجزء الى اليسار بالـــhigh والجزء الى اليمين بال low . ولو اخذنا مسجل AX كمثال فانه يتكون من AH(high) وجزء اخر هو AL(low) ويتم الاشارة الى اي جزء من خلال اسمه. والمعالج 80386 والمعالجات التي بعدها تدعم كل انواع مسجلات الاستخدام العام اضافة نسخ 32-bit الموسعة من هذه المسجلات وهي EAX, EBX, ECX, and EDX.

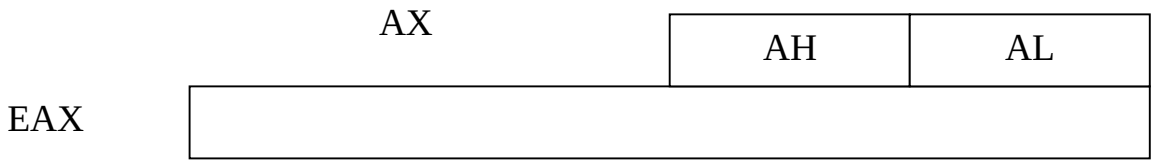
The following assembler instructions move zeros to the AX, BH, and ECX registers, respectively:

وايعازات ايسمبلي التالية تنقل الصفر الى مسجلات AX, BH, and ECX على التوالي

```
MOV   AX,00
MOV   BH,00
MOV  ECX,00
```

AX register

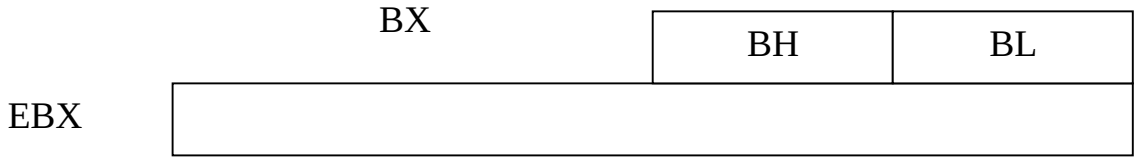
The AX register, the primary accumulator, is used for operations involving input / output and most arithmetic. For example, the multiply, divide, and translate instructions assume the use of the AX. Also, some instructions generate more efficient code if they reference the AX rather than another register.



يعتبر المسجل AX هو المركم الاساسي وهو يستخدم لعمليات الادخال والاخراج ومعظم العمليات الرياضية. وكمثال على هذه العمليات الضرب ، القسمة ، وايعازات النقل تفترض استخدام AX. بعض الايعازات تولد code اكثر كفاءة اذا كان التعامل مع AX بدلا من مسجلات اخرى.

BX register

The BX is known as the base register since it is the only general – purpose register that can be used as an index to extend addressing. Another common purpose of the BX is for computations.

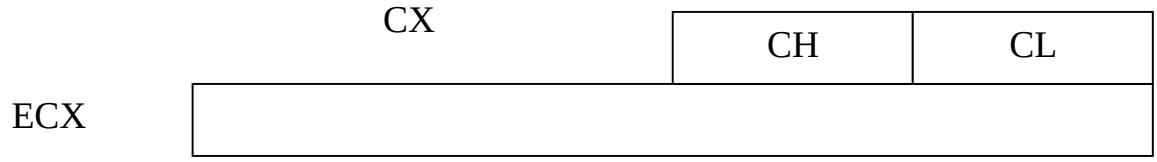


يسمى مسجل BX بمسجل القاعدة حيث انه المسجل الوحيد الذي يمكن ان يستخدم كمؤشر ضمن العنونة الموسعة.

والوظيفة العامة الاخرى هي الحسابات.

CX register

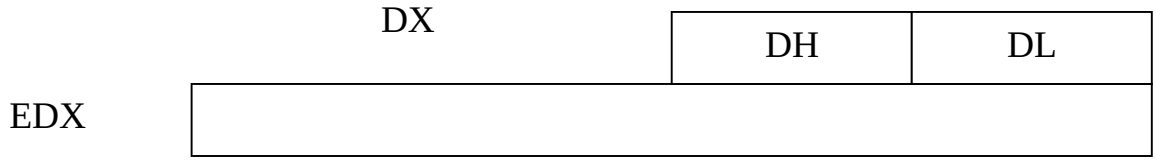
The CX is known as the count register. It may contain a value to control the number of times a loop is repeated or a value to shift bits left or right. The CX may also be used for many computations.



يعرف المسجل CX بانه مسجل العداد . وهو قد يحتوي على قيمة تسيطر على عدد مرات تكرار تنفيذ حلقة تكرارية او قيمة تزحيف بتتات الى اليمين او اليسار ، ويمكن ان يستخدم في العمليات الرياضية.

DX register

The DX is known as the data register. Some input / output operations require its use, and multiply and divide operations that involve large values assume the use of the DX and AX together as a pair.



يعرف مسجل DX بانه مسجل البيانات ، وتستخدم في بعض عمليات الادخال والاخراج وكذلك في عمليات الضرب والقسمة لاعداد كبيرة حيث يتم استخدام كل من المسجلين AX و DX كمسجلين مزدوجين.

You may use any of these general – purpose registers for addition and subtraction of 8 - bit, 16 – bit, or 32 – bit values.

ويمكن ان يستخدم اي من هذه المسجلات ذات الاستخدام العام في عمليات الجمع والطرح وباحجام بيانات ٨ بت و١٦ بت و٣٢ بت.

4.Index Registers

The SI and DI registers are available for indexed addressing and for use in addition and subtraction.

تسخدم مسجلات الفهرسة لعمليات العنونة المفهرسة وكذلك في عمليات الجمع والطرح.

SI register

The 16 – bit source index register is required for some string (character) operations. In this context, the SI is associated with the DS register, The 80386and later processors support a 32 – bit extended register, the ESI.

ان مسجل فهرسة المصدر ذو ١٦ بت يستخدم مع بعض عمليات السلاسل الحرفية. وفي هذه الحالة يرتبط مسجل SI مع مسجل DS ، ومعالج 80386 والمعالجات التي بعده تدعم النوع الموسع من هذ المسجل وبطول ٣٢بت وباسم ESI.

DI register

The 16 – bit destination index register is required for some string (character) operations. In this context, the DI is associated with the ES register, The 80386and later processors support a 32 – bit extended register, the EDI.

ان مسجل فهرسة المستقرذو ١٦ بت يستخدم مع بعض عمليات السلاسل الحرفية. وفي هذه الحالة يرتبط مسجل DI مع مسجل ES ، ومعالج 80386 والمعالجات التي بعده تدعم النوع الموسع من هذ المسجل وبطول ٣٢بت وباسم EDI.

Flags Register

Nine of the bits of the flags register are common to all 8086 – family processors to indicate the current status of the computer and the results of processing. Many instructions involving comparisons and arithmetic change the status of the flags, which some instructions may test to determine subsequent action.

تسعة من البتتات لمسجل flags هي مشتركة لكل عائلة معالج 8086 والتي تبين الحالة الحالية للمعالجة ، والكثير من الايعازات الرياضية وايعازات المقارنة تغير حالة flags والنوع الاخر يدقق قيمتها ليحدد على اساس التدقيق اجراء عملية معينة.

The following briefly describes the common flag bits:

سيتم شرح هذه البتات الان

1. OF(overflow). Indicates overflow of a high – order (leftmost) bit following arithmetic, or indicates a carry into and out of the high order (leftmost) sign bit following a signed arithmetic operation . JO and JNO test this flag.

يبين حالة وجود بتين اضافيين بعد اخر بت من جهة اليسار او وجود بتين اضافيين من جهة اليسار بعد بت الاشارة في حالة العمليات الرياضية بارقام مع اشارات . والايعازين JO and JNO يدققان هذا flag.

2. DF(direction). Determine left or right direction for moving or comparing string (character) data. When the flag is 0, the string operation performs left – to – right data transfer, when the flag is 1, the string operation performs right – to – left data transfer.

يحدد اتجاه اليمين اواليسار لعمليات النقل اوالمقارنة لبيانات سلاسل حرفية . عندما يكون قيمة flag هو صفر، فان عملية السلسلة الحرفية سوف تنقل البيانات من اليسار الى اليمين ، عندما يكون قيمة flag هو واحد، فان عملية السلسلة الحرفية سوف تنقل البيانات من اليمين الى اليسار.

3. IF(interrupt). Indicates that all external interrupts, such as keyboard entry, are to be processed or ignored. Which mean all interrupts are disables when 0, and enables when 1.

يبين ان كانت كل عمليات المقاطعة الخارجية مثل ادخال من قبل لوحة المفاتيح سوف يتم التعامل معها او اهمالها ، وهذا يعني فان كل المقاطعات سوف يتم اهمالها عندما تكون قيمة هذا flag تساوي صفر. وعندما تكون قيمة flag تساوي واحد فان كل المقاطعات سوف يتم التعامل معها.

4. TF(trap). Permits operation of the processor in single – step mode. Debugger programs such as DEBUG set the trap flag so that you can step through execution a single instruction at a time to examine the effect on registers and memory.

تسمح بتحويل عمل المعالج الى حالة الخطوة الخطوة . وبرامج التصحيح مثل برنامج DEBUG يحول قيمة هذا flag الى واحد حيث يسمح بتحويل تنفيذ البرنامج من تنفيذ برنامج متكامل الى تنفيذ ايعاز واحد في كل مرة وذلك لفحص تاثير تنفيذ الايعاز على المسجلات والذاكرة.

5. SF(sign). Contains the resulting sign of an arithmetic operation (0 = positive and 1 = negative) or (the first bit from left from unsigned number). JG(Jump if Greater) and JL (Jump if Less) test this flag.

يحتوي هذا flag على قيمة الاشارة لناتج عملية رياضية حيث يمثل(صفر الاشارة الموجبة والواحد تمثل الاشارة السالبة) او اول بت من جهة اليسار في حالة كونها عملية رياضية ولكن لارقام بدون اشارات . وتستخدم الايعازين JG اقفز اذا اكبر و JL اقفز اذا اقل لتدقيق flag.

6. ZF (zero). Indicates the result of an arithmetic or comparison operation (0 = nonzero and 1 = zero result). JE (Jump if Equal) and JZ(Jump if Zero) test this flag.

يبين ان كان الناتج لعملية رياضية او مقارنة هو صفر فتصبح قيمة flag هو واحد. وتصبح قيمة flag هو صفر عندما يكون قيمة الناتج اي عدد غير الصفر ، والايعازين JE اقفز اذا كان هناك تساوي و JZ اقفز اذا كان هناك صفر تدقق هذا flag.

7. AF(auxiliary carry). Contains a carry out of bit 3 on 8 – bit data, for specialized arithmetic. It concerned with arithmetic on ASCII and BCD packed fields.

تاخذ قيمة carry الذي يولد عندما يعبر carry من اول اربع بتات من bit 3 خلال عملية رياضية لبيانات بطول ٨ بتات وضمن العمليات الرياضية الخاصة وهي مختصة بالعمليات التي تجرى على ASCII و BCD.

8. PF (parity).Indicates even or odd parity of a low – order (rightmost) 8 – bit data operation. JPO(Jump if parity Odd) and JPE(Jump if parity Even) test this flag.

يبين حالة even or odd parity لعمليات رياضية من ٨ بتات وفقط للجهة اليمنى . والايعازين JPO اقفز اذا كان parity هو فردي و JPE اقفز اذا كان parity هو زوجي يدقق هذا flag.

9. CF (carry). Contains carries from a high – order (leftmost) bit following an arithmetic operation; also, contains the contents of the last bit of a shift or rotate operation. JC and JNC test this flag.

هي حالة البت الاضافي بعد اخر بت من جهة اليسار او البت النـــاتج من عملية تزحـــيـــف او دوران والايعـــــازين JC and JNC تدقق هذا flag.

The flags are in the flags register in the following locations (which you need memorize):

Flag					O	D	I	T	S	Z		A		P		C
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Bit no.

The flags most relevant to assembly programming are OF, SF, ZF, CF for comparison and arithmetic operations, and DF for the direction of string operations. The 80286 and later processors have some flags used for internal purposes, concerned primarily with protected

mode. The 80386 and later processors have a 32 – bit extended flags register known as Eflags.

ان الاكثر flags المستخدمة هي OF, SF, ZF, CF والتي تستخدم في العمليات الرياضية وعمليات المقارنة ، و كذلك فان DF flag يستخدم لتحديد الاتجاهات في عمليات التعامل مع السلاسل الحرفية ، والمعالجات 80286 فما بعدها تمتلك flags تستخدم في اغراض داخلية ، وخاصة عندما تعمل في حالة protected ، والمعالجات 80386 فما بعدها مسجل flag الموسع وبطول 32-bit وباسم Eflags.

Question Show the effect of below operations on these flags DF,IF,TF,SF,CF,SF,AC,ZF,PF

1. an arithmetic operation as add AH+BH and store result in AL with these numbers

4E

8A +

2. a program deal with four subroutine one of them is Arabic program for bank customer.

SOL:

1.	1	
4E	0100	1110
8A	1000	1010 +
	1101	1000

DF , IF , TF not effected

ZF=0

SF=1

CF=0

AC=1

PF=1

2. ZF,SF,CF,AC,PF not effected

DF=1

IF=1

TF=0