به نام خدا

دانشکده فنی و حرفه ای مائده

گروه کامپیوتر

جزوه درس زبان ماشین و اسمبلی

قابل توجه دانشجویان عزیز:

به همراه این فایل جهت توضیحات تکمیلی مطالب گفته شده از فصل دو و سه کتاب "برنامه نویسی به زبان اسمبلی" نوشته جعفرنژاد قمی نیز باید مطالعه شود.

تمرین های پایان هر فصل که در این فایل آمده است جواب داده شود و در اولین جلسه کلاس حضوری تحویل داده شود.

مدرس:

نگار مهدی نژاد

فصل2

ساختار کامپیوتر و ریزپردازنده های80X86

واژه میکروپروسسور در صنعت نیمه هادی توسط شرکت اینتل ابداع شد . آنها این واژه را برای توصیف یک مدار مجتمع ماشین حساب چهار بیتی که تازه طراحی کرده بودند به کار بردند . امروزه میکروپروسسور به آی سی هایی گفته می شود که اساس یک میکروکامپیوتر را تشکیل می دهند.

بعضی سازندگان به کار بردن چند میکروپروسسور در یک کامپیوتر را مفید تشخیص داده اند . یکی از میکروپروسسورها برای کنترل صفحه کلید ، دومی برای پرداختن به عملیات ورودی خروجی ، سومی برای کنترل وسایل ذخیره سازی انبوه(دیسک گردانها)و- چهارمی به عنوان پروسسور اصلی سیستم می توانند به کار روند . این تکنیک پردازش توزیع شده Distributed Processing نام دارد.

شاید بتوان علت ساخت پردازنده ها را در سادگی ، کم بودن حجم و برنامه پذیری خلاصه کرد اما در واقع هر سیستم کامپیوتری بر اساس نحوه دریافت و پردازش اطلاعات به صورتهای زیر تقسیم می شود:

کامپوتر آنالوگ : دارای ورودی آنالوگ

کامپیوتر دیجیتال : دارای ورودی دیجیتال

کامپیوتر ترکیبی : دارای ورودی آنالوگ و دیجیتال (یک کاربرد در هواشناسی)

رایانه نسل اول لامپ خلاء زبان ماشین

رایانه نسل دوم ترانزیستور زبان اسمبلی

رایانه نسل سوم آی سی زبان سطح بالا

رایانه نسل چهارم VLSI زبان سطح بالا

رایانه نسل پنجم هوش مصنوعی

رایانه نسل ششم شبکه های عصبی مصنوعی

ریز پردازنده مدار الکترونیکی بسیار گسترده و پیچیده ای میباشد که دستورات برنامه های ذخیره شده را انجام می دهد . پردازنده دو عمل مهم انجام می دهد:

1 کنترل تمام محاسبات و عملیات-

2 کنترل قسمت های مختلف-

پردازنده ها وظایف اصلی زیر را برای رایانه انجام می دهد:

1دریافت داده هااز دستگاه های ورودی-

2انجام عملیات و محاسبات وکنترل ونظارت بر آنها-

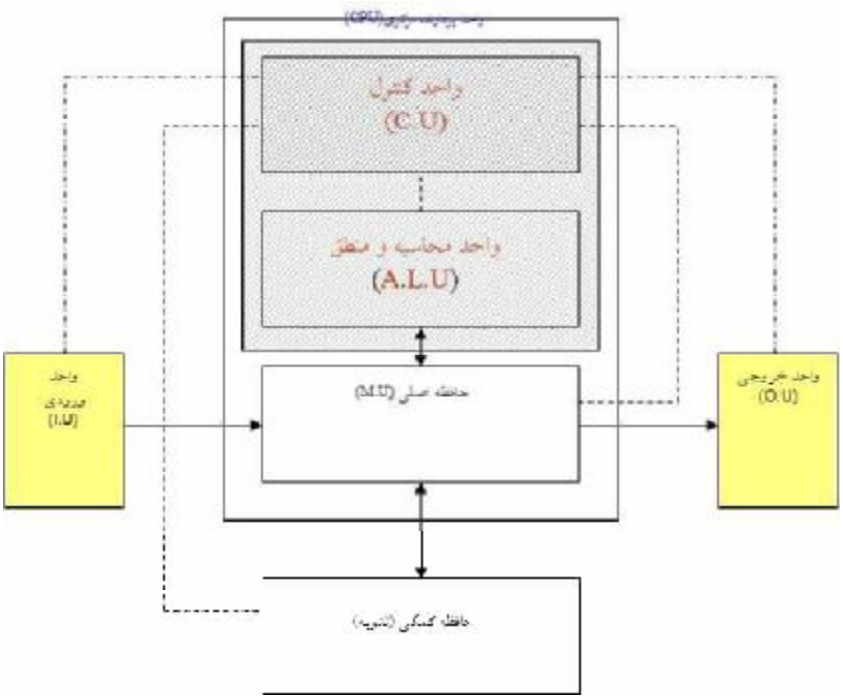
3ارسال نتایج عملیات با دستگاه های خروجی-

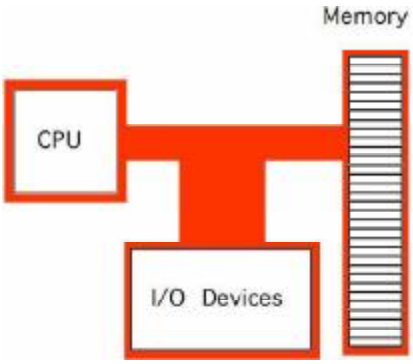
همچنین عملکرد یک پردازنده از نظر فنی با دو ویژگی تعیین می شود:

1-تعداد بیت هایی که یک پردازنده در هر لحظه پردازش می کند . طول این کلمات معمولا4 و8 و 16 و 32 و 64بیتی می باشد.

2تعداد پالس های الکترونیکی که در یک ثانیه تولید شده است و با واحد مگاهرتز سنجیده می شود– .

اجزای یک سیستم کامپیوتری





نکته : 1 پردازنده بطور مستقیم با حافظه اصلی در ارتباط است.

نکته : 2 ظرفیت حافظه های اصلی خیلی بالا نیست بنابراین لازم است از حافظه های کمکی نیز استفاده شود که دارای سرعت کمتریمی باشد.

بخشهای داخلی یک ریزپردازنده:

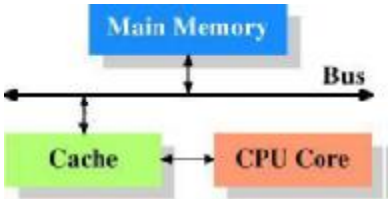
v ALU (ARITHMATIC LOGIC UNIT) وظیفه این بخش انجام عملیاتهای محاسباتی (ریاضی) و منطقی است.

v CU (CONTROL UNIT) این قسمت فرمانهای لازم جهت کنترل و نظارت بر سیستم را به قسمتهای مختلف ارسال و یا از قسمتهای مختلف دریافت می کند.

v REGISTER ثباتها حافظه های کوچک و سریعی هستند که در درون پردازنده جهت ذخیره موقت اطلاعات بکار می روند

و بسته به نوع ریزپردازنده می توانند 8 بیتی ، 16 بیتی ، 32 بیتی و 64 بیتی و ... باشند.

v CACHE این قسمت جهت ذخیره اطلاعاتی که بطور مداوم استفاده می شود بکار می رود.

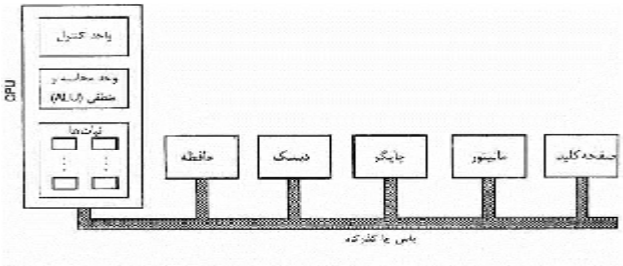


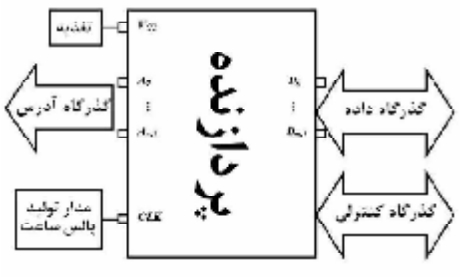
v ACCUMULATOR :ثباتی است که از سایر ثباتها متمایز بوده و عمدتاً در متن اجرای برخی دستورات قرار دارد که جهت نگهداری نتجه مورد استفاده قرار می گیرد (. برگه کار موقت)

v IR (INSTRUCTION REGISTER) این قسمت معنای هر دستور و مراحلی که یک پردازنده بایستی برای اجرای آن در پیش بگیرد را مشخص می کند . دستورات پس از دریافت از حافظه وارد این قسمت شده تا پس از رمزگشایی اجرا شوند .

v PC (PROGRAM COUNTER) از آنجایی که دستورات یک برنامه بایستی به ترتیب اجرا شوند پردازنده بایستی به طریقی بداند که دستور بعدی را که بایستی اجرا کند کدام است . وظیفه این ثبات نگهداری آدرس دستور بعدی است که قرار است اجرا شود . با اجرای هر دستور پردازنده به طور خودکار یک واحد به این ثبات اضافه می کند تا به دستور بعدی اشاره کند (. همان ثبات IP در برخی کامپیوترها)

v BUS سیمهای ارتباطی بین عده ای وسایل در کامپیوتر باس یا گذرگاه نامیده می شود.





پردازنده های 80X86 اینتل

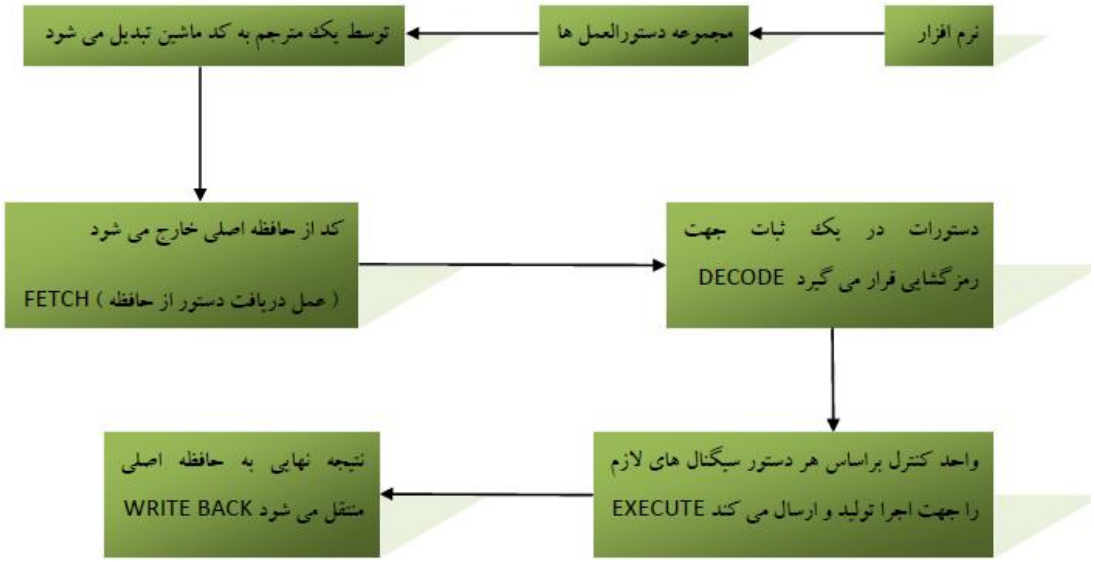
4004	این پردازنده یک پردازنده ۴ بیتی بود که جهت ماشین حسابهای دستی طراحی شده بود و از ۲۳۰۰ ترانزیستور PMOS ساخته شده بود.
8008	این پردازنده یک پردازنده ۸ بیتی ۱۸ پایه بود که از ۳۰۰۰ ترانزیستور PMOS ساخته شده بود و دارای ۶۶ دستورالعمل بوده و قابلیت آدرس دهی 16KB را دارا می باشد .
8080	این پردازنده ۴۰ پایه بود که از تکنولوژی NMOS ساخته شده بود و دارای ۱۱۱ دستورالعمل بوده و قابلیت آدرس دهی 64KB را دارا می باشد .
8085	اینتل ۳ پردازنده فوق را در یک تراشه تحت عنوان 8085 قرار داد .
8086	این پردازنده یک پردازنده ۱۶ بیتی بوده و قابلیت آدرس دهی 1MB حافظه را دارا بود .
8088	تا قبل از استفاده شرکت IBM از پردازنده های اینتل ، شرکت اینتل بعنوان تولید کننده تراشه های حافظه شناخته می شد . اینتل این پردازنده را برای کاهش هزینه های بوردهای 8085 طراحی نمود . پردازنده ای که در درون ۱۶ بیتی ولی دارای ۸ پایه گذرگاه داده بود و از ۲۹۰۰۰ ترانزیستور ساخته شده بود .
80286	تمامی پردازنده های قبلی بصورت بسته بندی DIP ساخته شده بودند . این پردازنده یک پردازنده ۱۶ بیتی ، دارای ۲۴ پایه گذرگاه آدرس و بصورت بسته بندی LCC و از ۱۳۰۰۰۰ ترانزیستور ساخته شده بود که توانایی اجرای کلیه دستورات ۸۰۸۶ و ۸۰۸۸ را دارا بود .
80386	قبل از ساخت این پردازنده اینتل تولید تراشه های حافظه را متوقف کرد . این پردازنده ۳۲ بیتی دارای ۳۲ پایه گذرگاه آدرس و ۱۳۲ پایه که بصورن بسته بندی PGA و از ۲۷۵۰۰۰ ترانزیستور CMOS ساخته شده بود .
80486	قبل از ساخت این پردازنده اینتل مجوز تولید تراشه های ۸۰۸۶ و ۸۰۸۸ را برای سایر شرکت ها صادر و خود تنها به ساخت تراشه ۸۰۳۸۶ پرداخت . این تراشه اولین ریزپردازنده ای بود که شامل ۱/۲ میلیون ترانزیستور بود . یک پردازنده ۳۲ بیتی با ۱۶۸ پایه در بسته بندی PGA و قابلیت آدرس دهی 4GB حافظه .
PENTIUM	اولین پردازنده پنتیوم با بیش از ۳ میلیون ترانزیستور BiCMOS که با پردازنده های قبلی سازگار بود ساخته شد . این پردازنده دارای گذرگاه داده ۶۴ بیتی و ثباتهای داخلی ۳۲ بیتی بود . دارای دو عدد حافظه نهان داخلی (یکی برای داده و یکی برای آدرس) و مدارهای پیش بینی پرش و ۲۷۳ پایه ای و تقریباً دو برابر سریعتر از ۸۰۴۸۶ بود .

انواع RAM به طورکلی دو نوع وجود دارد:

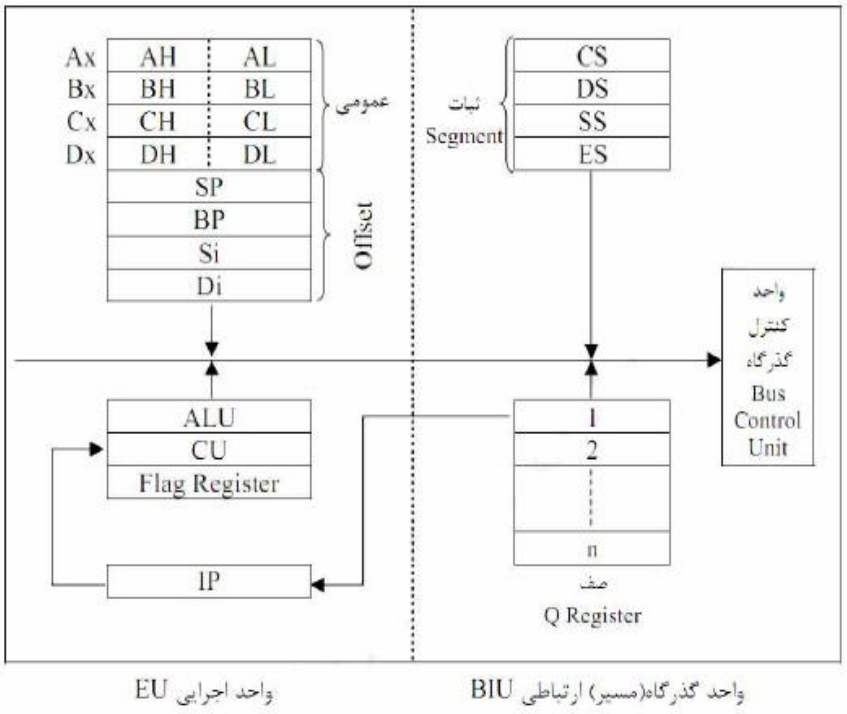
SRAM در ساخت این نوع حافظه از فلیپ فلاپها استفاده می شود . جهت ساخت هر سلول حافظه از 6 ترانزیستور استفاده میشود (امروزه 4 ترانزیستور) در ساخت حافظه های CACHE به دلیل سرعت بالاتر این نوع حافظه ها استفاده می شود.

DRAM در ساخت این نوع حافظه از خازن استفاده می شود . جهت ساخت هر سلول حافظه از 1 ترانزیستور استفاده می شود.

کار پردازنده :



آشنایی با معماری پردازنده های 80X86



ثباتها :

ثباتهای پردازنده ۸۰۸۶ به شش دسته تقسیم می شوند که همه این ثباتها شانزده بیتی هستند :

۱- ثباتهای عمومی (همه منظوره) AX , BX , CX , DX

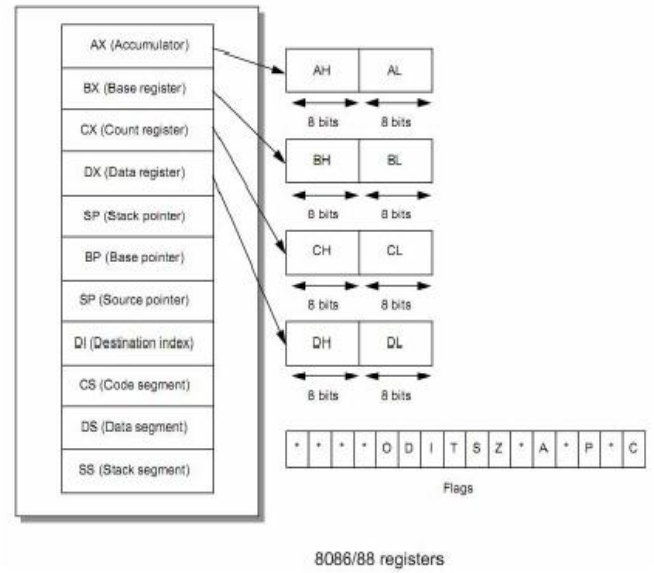
۲- ثباتهای قطعه ES , SS , CS , DS

۳- ثباتهای اشاره گرها BP , SP

۴- ثبات دستور IP

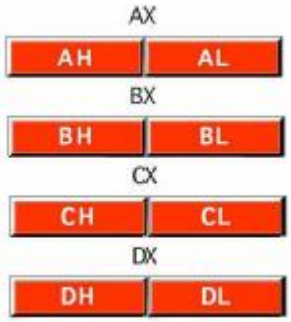
۵- ثباتهای شاخص (اندیس) SI , DI

۶- پرچم FR



8086/88 registers

ثباتهای همه منظوره : این ثباتها برای مقاصد مختلفی بکار می روند . ویژگی منحصر به فرد این ثباتها قابلیت استفاده آنها بصورت 8 بیتی و16 بیتی است . مثلاً ثبات 16 بیتی AX شامل یک بخش (AH هشت بیت بالا) و (ALهشت بیت پایین) است.

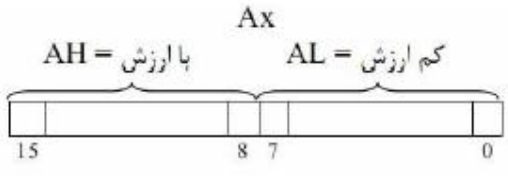


ثبات: (ACCUMULATOR) AX این ثبات در دستورالعملهای محاسباتی و ورودی /خروجی به عنوان ثبات نتیجه عملیات به کار می رود.

ثبات : (BASE) BX این ثبات جهت نگهداری آدرس پایه حافظه به کار می رود . کاربرد دیگر آن در انجام محاسبات است.

ثبات : (COUNTER) CX از این ثبات معمولاً برای شمارش دفعات تکرار یک حلقه و نیز محاسبات استفاده می شود.

ثبات : (DATA) DX از این ثبات در عملیاتهای ورودی/خروجی به عنوان آدرس پورت استفاده می شود.



AX=39F7H ➔ AL=7FH , AH=39H

قطعه یا سگمنت:

هر برنامه اسمبلی حداقل از سه قطعه تشکیل شده است . قطعه کد که دستورات برنامه در آن قرار دارد ، قطعه داده که اطلاعات موجود در برنامه در آن قرار دارد و قطعه پشته که برای اطلاعات موقت بکار می رود . فرض کنید شما می خواهید عملیات ضرب زیر را انجام دهید : ابتدا 5 را در 7 ضرب کرده و از آنجایی که جواب آن 35 می شود 5 را نوشته و 3 را در ذهن خود می سپارید سپس 5 را در 1 ضرب کرده و با 3 ذهن جمع کرده و حاصل را که 8 هست می نویسید . این الگو محاسبه در کامپیوتر به شکل قطعه بصورت زیر نوشته می شود:

۱۷
× ۵

قطعه کد	دستورات	× و + و -
قطعه داده	اطلاعات	۱۷ و ۵ و ۸۵
قطعه پشته	اطلاعات موقت	۳

نکته : از آنجایی که 8086 دارای بیست خط آدرس می باشد بنابراین حداکثر تا 1MB را پشتیبانی می کند این مقدار در واقع بیشترین حافظه قابل دسترس فیزیکی می باشد.

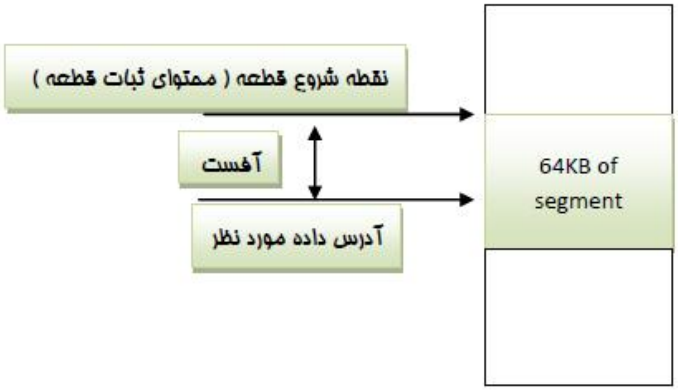
وظیفه ثبات : CS محل نگهداری آدرس ابتدای قطعه کد (ثبات سگمنت کد)

وظیفه ثبات : DS محل نگهداری آدرس ابتدای قطعه داده (ثبات سگمنت دیتا)

وظیفه ثبات : SS محل نگهداری آدرس ابتدای قطعه پشته (ثبات سگمنت پشته)

وظیفه ثبات : ES محل نگهداری آدرس ابتدای قطعه اضافی (ثبات سگمنت اکسترا)

آدرس تفاوت مکان (آفست) : این آدرس در محدوده 64KB قطعه قرار داشته و یک آدرس ۱۶ بیتی می باشد . معمولاً فاصله نقطه شروع قطعه تا مکانی که داده مورد نظر درآنجا قرار دارد را آفست گویند . محدوده آن از 0000^H الی $FFFF^H$ می باشد .

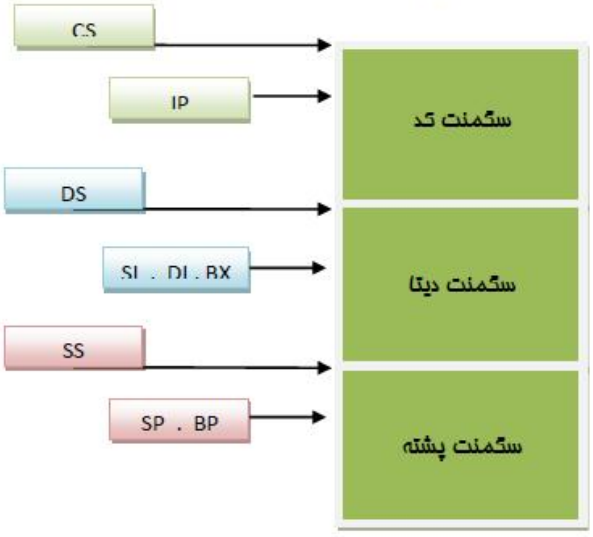


اشاره گرها POINTER : ثباتهای اشاره گر ۱۶ بیتی نگهدارنده بخش آفست در آدرس دهی هستند و همراه یکی از ثباتهای قطعه به محلی از حافظه اشاره می کنند . (قبلاً دیدیم که ثبات IP با ثبات CS اشاره به یک آدرس منطقی یک دستور در حافظه می کند .)

IP : همراه با ثبات CS به دستورالعمل بعدی که بایستی توسط پردازنده اجرا شود اشاره می کند .

SP : آفست مکانی از محدوده قطعه پشته است که عمل قرار گرفتن داده در پشته صورت می گیرد . SS : SP

BP : برای دسترسی به متغیرهایی که در پشته قرار دارند استفاده می شود .



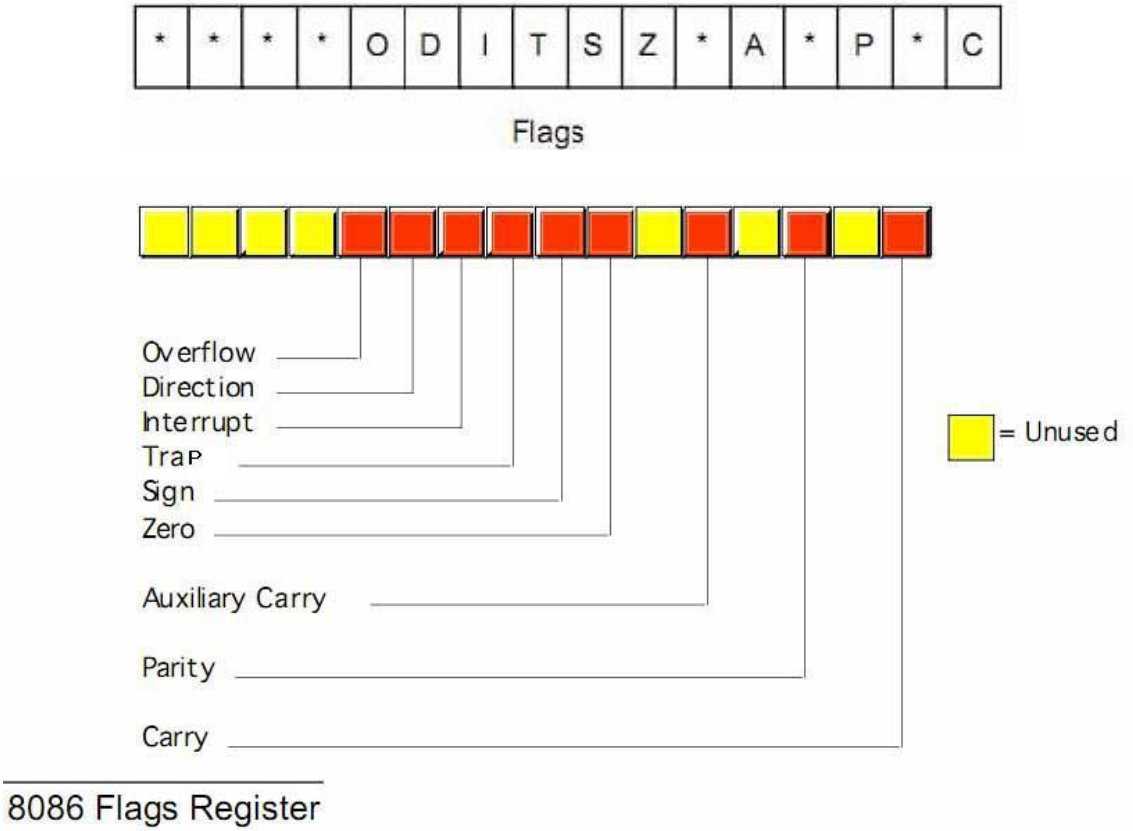
ثبات های شاخص : INDEX این دو ثبات 16 بیتی اغلب به عنوان اشاره گر به همراه DS به کار می روند تا به داده های موجود در محدوده قطعه داده دسترسی شود . اما به منظورهای دیگری نیز استفاده می شوند مثلاً

اگرچه نمی توانند به دو بخش هشت بیتی تجزیه شوند اما میتوانند مانند ثباتهای همه منظوره به کار روند (. آفست قطعه اضافی ثبات DI می باشد)

SI : برای آدرس دهی و در عملیاتهای رشته ای به عنوان مبداء استفاده می شوند.

DI : برای آدرس دهی و در عملیات های رشته ای بعنوان مقصد استفاده می شوند.

ثبات پرچم : FLAG REGISTER بیتهای این ثبات وضعیت CPU را بعد از انجام یک عمل نشان می دهند.



8086 Flags Register

رقم نقلی : Carry (C)در عملیاتهای محاسباتی و همینطور چرخش و شیفت این بیت جهت ذخیره رقم نقلی به کار می رود (. با ارزش ترین رقم نقلی در محاسبات ذخیره می شود) در واقع اگر نتیجه یک عمل محاسباتی بدون علامت آنقدر بزرگ باشد که در مقصد جا نشود این پرچم برابر یک خواهد شد.

توازن : Parity : P این پرچم توازن بایت اول 8 (بیت کم ارزش) را چک می کند . اگر پس از انجام عملی تعداد زوجی از یک ها وجود داشته باشد این پرچم یک می شود (توازن فرد) در واقع این پرچم مخصوص تست انتقال دیتا می باشد.

رقم نقلی کمکی : Auxiliary Carry : A در انجام عملیات ریاضی به صورت BCD اگر رقم نقلی از بیت d3 به d4 وجود داشته باشد این پرچم یک و در غیر این صورت صفر است.

پرچم صفر : Zero : Z هرگاه نتیجه یک عملیات حسابی یا منطقی صفر گردد این پرچم یک می شود و در غیر این صورت صفر است.

علامت : Sign : S این بیت مستقیماً به بیت پرارزش نتیجه عملیات (MSB) حاصل عملیات متصل است اگر نتیجه منفی باشد این پرچم برابر یک و اگر نتیجه عملیات مثبت باشد برابر صفر است.

تله : Trap/Trace : T این پرچم برای اجرای برنامه بصورت گام به گام یا دستور به دستور کاربرد دارد . در واقع این پرچم مشخص میکند که آیا پردازنده پس از اجرای هر دستور متوقف می شود یا خیر . اگر این بیت 1 باشد برنامه به صورت گام به گام اجرا می شود و اگرصفر باشد بصورت طبیعی اجرا می شود.

وقفه : Interrupt : I جهت تشخص وقفه کاربرد دارد . اگر یک باشد فعال و اگر صفر باشد وقفه غیر فعال است.

جهت : Direction : D جهت کنترل عملیاتهای رشته ای از قبیل انتقال یا مقایسه کاربرد دارد . اگر یک باشد عمل انتقال یا مقایسه از راست به چپ یا از پایین به بالا است و اگر صفر باشد عمل انتقال یا مقایسه از چپ به راست یا از بالا به پایین است.

سرریز : Overflow : O اگر در حاصل یک عملیات (علامت دار) نتیجه ای بیش از ظرفیت بدست آید سرریز رخ داده است و این پرچم یک می شود.

مثلاً در عملیات جمع خطای سرریز زمانی رخ می دهد که علامت حاصلجمع دو عدد مثبت ، منفی شود و یا علامت حاصلجمع دو عدد منفی ، مثبت شود . برای برطرف کردن این خطا تعداد بیتها را افزایش می دهند مثلاً بجای هشت بیت از ده بیت استفاده می شود. در جمع دو عدد مختلف العلامت سرریز رخ نمی دهد.

نکته مهم : تفاوت بین نقلی و سر ریز $OF=CF\oplus Cin$

: CF نقلی خارج شده از بیت علامت : Cin نقلی وارد شده به بیت علامت است.

اگر OF=1 عدد در محدوده علامتدار نادرست است و اگر CF=1 عدد در محدوده بدون علامت نادرست است.

مثال : ضمن بدست آوردن حاصل جمع های زیر مقادیر ثبات پرچم را نشان دهید .

$38^H + 2F^H =$

0011 1000+0010 1111 = 0110 0111 ⟶ 67^H

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

$2345^H + 3219^H =$

0010 0011 0100 0101+0011 0010 001 1001=0101 0101 0101 1110 ⟶ $555E^H$

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

	علامتدار	بدون علامت	
از آنجایی که هم CF و هم OF برابر صفر است نتیجه در محدوده علامتدار وبی علامت صحیح است .	+4 -5 -1	4 +251 255	0000 0100 +1111 1011 1111 1111 CF=0,Cin=0 ➔ OF=0
OF=0 فقط محدوده علامتدار صحیح است .	-4 + 5 +1	252 + 5 1	1111 1100 +0000 0101 0000 0001 CF=1,Cin=1 ➔ OF=0
CF=0 فقط محدوده بدون علامت صحیح است .	+121 + 11 -124	121 + 11 132	0111 1001 +0000 1011 1000 0100 CF=0,Cin=1 ➔ OF=1
از آنجایی که هم CF و هم OF برابر یک است هر دو نتیجه در محدوده علامتدار وبی علامت نادرست است .	-10 -119 +127	246 +132 127	1111 0110 +1000 1001 0111 1111 CF=1,Cin=0 ➔ OF=1

صف دستورات : **INSTRUCTION QUEUE** در ریزپردازنده ها یک صف دستورالعمل وجود دارد که طول آن برای پردازند های اولیه محدود بود و به تدریج در پردازنده های بعدی افزایش یافت که نهایتاً منجر به ایجاد CACHE گردید (. در 8086 طول صف دستور(4 بایت) دستور می باشد) از این صف برای ذخیره تعدادی دستورالعمل که از حافظه به ریز پردازنده منتقل می شوند استفاده می گردد (. به منظور افزایش سرعت) CPU در پردازنده های 8085 و ماقبل آن برای اجرای یک برنامه پردازنده می بایست دستور را از حافظه دریافت آنرا اجرا و سپس مجدداً به حافظه مراجعه کرده و دستور بعدی را به پردازنده منقل نماید و این عمل بصورت پشت سرهم انجام می شد . اما در پردازنده 8086 با ایجاد صف دستور همزمان با اجرای یک دستور (بطور موازی) تا 4 دستور بعدی نیز از حافظه دریافت و در این صف ذخیره می شد) FIFO (. با این کار پردازنده برای اجرا مجبور نبود هربار به حافظه مراجعه کند ودر نتیجه زمان کلی اجرا کاهش و سرعت اجرا بالاتر خواهد رفت.

تمرینهای فصل 2:

1-کدام یک از ثباتهای زیر نمی توانند به دو بایت بالا و پایین تقسیم شوند.

الف CS ب AX پ DS ت SS ث BX ج DX

2-وضعیت CF,PF,AF,ZF,SF را برای اعمال زیر مشخص کنید.

الف MOV BL,9FH ADD BL,61H

ب MOV DX,10FFH ADD DX,1H

3- روند اجرای یک دستور توسط پردازنده به چه شکل است؟

4- ثباتهای پردازنده 8086 را نام ببرید و کاربرد هر یک را توضح دهید.

5- اصطلاحات زیر را توضیح دهید:

سگمنت: آفست: ثبات: گذرگاه:

فصل 3:

مقدمات زبان اسمبلی

معمولاً هر برنامه زبان اسمبلی از تعدادی دستورالعمل ساخته شده است که بیانگر عملیاتی است که بایستی انجام شود . مجموعه دستورالعملهای یک میکروپروسسور لیست تمام فرمانها یی است که CPU می تواند تشخیص دهد و اجرا کند.

شبه دستورات : اسمبلر دارای فرمانهایی است که کاربر را در کنترل ترجمه و تهیه لیست های برنامه یاری می کند . این فرمان ها به شبه دستورات معروف هستند.

انتخاب اسم : اسم با رقم شروع نمی شود . از حروف و اعداد انگلیسی تشکیل میشود.

همچنین اسم نبایستی از کلمات ذخیره شده در اسمبلی باشد.

انواع ثابت ها:

1-باینری : شامل صفر و یک ها می باشد که در انتهای آن حرف B نوشته می شود.

2-دسیمال : شامل ارقام صفر الی 9 می باشد (اضافه کردن حرف D در انتهای آن اختیاری می باشد)

3-هگزادسیمال : شامل ارقام 1 الی 9 و حروف A الی F که در انتهای آن حرف H نوشته می شود.

4-اکتال : شامل ارقام 1 الی 7 می باشد که در انتهای آن حرف O نوشته می شود (. به جای O از Q نیز استفاده می شود)

5-کاراکتر : شامل هر کاراکتراز کدهای اسکی می باشد که بین علامت نقل قول ' یا " قرار می گیرد.

نکته : هرگاه مقداری در مبناء شانزده با حروف A تا F شروع شود بایستی به اول آن یک 0 اضافه شود تا کامپیوتر آنرا با یک برچسب یا متغیر اشتباه نگیرد.

داده ها : معمولاً عرض یک داده برابر سایز ثباتهای داخلی پردازنده می باشد (در اینجا شانزده بیتی) شبه دستورهای داده ها برای همه خانواده 80x86 یکسان و استاندارد می باشد.

تعریف متغیرها : شامل آدرس ، نوع داده و اندازه آن می باشد (. آدرس اطلاعات در حافظه)

شبه دستور : DB به متغیر مورد نظر یک بایت اختصاص می دهد.

شبه دستور : DW به متغیر مورد نظر دو بایت اختصاص می دهد.

شبه دستور : DD به متغیر مورد نظر چهار بایت اختصاص می دهد.

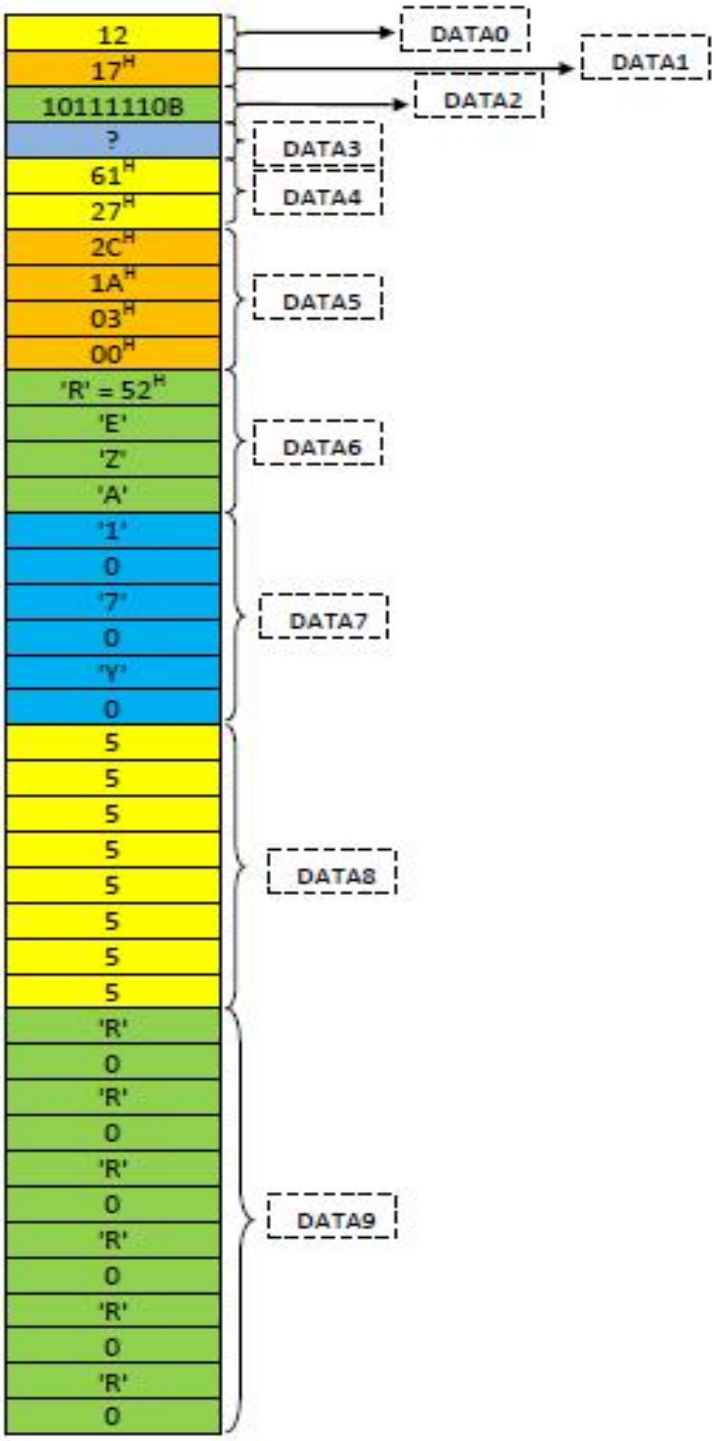
DBتنها شبه دستوری است که می تواند برای تعریف یک رشته اسکی بزرگتر از 2 کاراکتر مورد استفاده قرار گیرد .

شبه دستور (DUP کپی کردن :) با این شبه دستور می توان اطلاعات مساوی را در متغیرهای مختلف کپی نمود.

M1 DB 5DUP(4H) == M1 DB 4H, 4H, 4H, 4H, 4H

مثال : ضمن رسم نقشه حافظه بنویسید که هر متغیر تعریف شده چند بایت حافظه اشغال می کند . به کل قطعه نوشته شده چند بایت حافظه اختصاص می یابد ؟

```
DATA0  DB  12
DATA1  DB  17H
DATA2  DB  10111110B
DATA3  DB  ?
DATA4  DW  2761H
DATA5  DD  31A2CH
DATA6  DB  'REZA'
DATA7  DW  '1','7','Y'
DATA8  DB  8DUP(5)
DATA9  DW  2DUP3DUP('R')
```



همانطور که دیده می شود برای کل متغیرها ۴۰ بایت درنظر گرفته شده است

قالب کلی هر دستورالعمل زبان اسمبلی :

[lable] op-code operand ; comment

برچسب : معمولاً از برچسب به عنوان آدرس دستورالعمل در برنامه استفاده می شود . (آدرس دستور در حافظه)

برخی از انواع برچسب :

۱- برچسب دستور

```
BACK :
        MOV AL, 3C^H
        .
        .
        LOOP BACK
```

۲- برچسب متغیر

```
ALI DB 85
DATA DW ?
```

۳- برچسب روال

```
START PROC FAR
.
.
START ENDP
```

۴- برچسب قطعه

```
CDSEG SEGMENT
.
.
CDSEG ENDS
```

کد دستور : این قسمت حاوی عملی است که پردازنده بایستی آنرا انجام دهد .

عملوند : این قسمت حاوی اطلاعات مورد نیاز کد عمل می باشد . برخی از دستورات دارای دو عملوند و برخی شامل یک عملوند و برخی نیز بدون عملوند هستند . در شرایطی که دستور شامل دو عملوند باشد عملوند اول را مقصد و دومی را مبداء گویند .


```
MOV     AX,BX
ADD     BL,CH
```

توضیحات : این قسمت شامل توضیحات مورد نیاز دستورالعمل یا برنامه می باشد که بوسیله ; جدا می شود .

```
MOV AX,DATA1          ;move the first word into AX
```

روشهای آدرس دهی :

روش دست یافتن پردازنده به عملوند را آدرس دهی گویند .

۱- آدرس دهی فوری : در این روش عملوند مبداء یک مقدار عددی ثابت است .

```
MOV     AL,20
```

نکته : جهت ثباتهای قطعه و پرچم از این روش آدرس دهی استفاده نمی شود .

۲- آدرس دهی ثباتی : این روش مربوط به عملیاتهای ثباتهای داخلی ریز پردازنده می باشد . (سرعت آن از سایر روشهای آدرس دهی بالاتر است)

```
MOV     BX,CX
MOV     CL,DH
```

۳- آدرس دهی مستقیم : در این روش عملوند مبداء یا داده مورد نظر در حافظه وجود دارد و آدرس آن داده موجود است .

```
MOV     AX,[2000H]
MOV     CX,DATA3
```

نکته : ثباتهای شاخص SI , DI آدرس دهی مستقیم نمی شوند .

۴- آدرس دهی غیر مستقیم : در این روش آدرس داده مورد نظر در خود دستور نیست بلکه در یکی از ثباتهای BX , BP , DI , SI است .

```
MOV     AX,[BX]
```

۵- آدرس دهی غیر مستقیم نسبی پایه : در این روش آدرس موثر اطلاعات در حافظه برابر محتوای یکی از ثباتهای پایه BX , BP بعلاوه مقداری جابجایی قرار دارد .

```
MOV     AX,[BX+5]
MOV     AX,[BX]+5
MOV     AX,5 [BX]
```

۶- آدرس دهی غیر مستقیم نسبی شاخص (اندیس) : در این روش آدرس موثر اطلاعات در حافظه برابر محتوای یکی از ثباتهای شاخص SI , DI بعلاوه مقداری جابجایی قرار دارد .

```
MOV   AX, [SI+3]
```

۷- آدرس دهی غیر مستقیم نسبی پایه و شاخص : در این روش آدرس موثر اطلاعات در حافظه برابر مجموع محتوای یکی از ثباتهای شاخص SI , DI بعلاوه محتوای یکی از ثباتهای پایه BP , BX بعلاوه مقداری جابجایی قرار دارد .

```
MOV   AX, 4[DI][BX]
```

۸- آدرس دهی ضمنی : به دستوراتی که به طور مشخص به یک آدرس اطلاعات یا عملوند اشاره نمی کنند گفته می شود .

```
STC
NOP
CLD
```

دستورات زبان اسمبلی

اشکال مختلف دستورالعملها:

1-انتقال اطلاعات 2 محاسبات ریاضی 3 مقایسه و پرش 4 محاسبات منطقی 5 ورودی و خروجی 6 کنترل پرچم وتوقف کامپیوتر 7 زیر برنامه

1) انتقال اطلاعات:

دستور: MOV

MOV مبداء , مقصد

با اجرای این دستور محتوای عملوند مبداء به عملوند مقصد منتقل خواهد شد.

-هر دو عملوند بایستی از نوع بایت یا از نوع کلمه باشند.

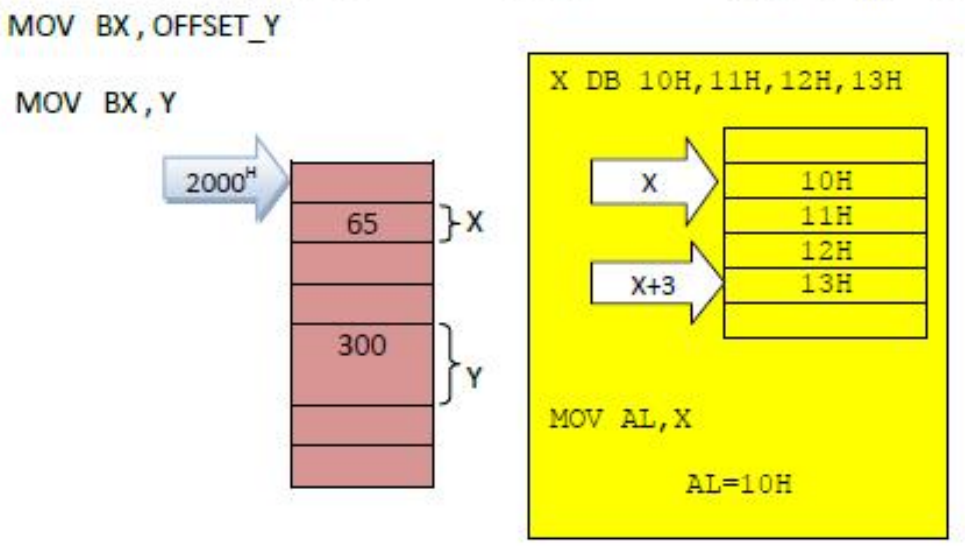
-هر دو عملوند نمی توانند متغیر باشند.

-هیچکدام از عملوندها نمی توانند ثبات های IP و FR باشند.

-محتویات دو ثبات قطعه را نمی توان مستقیماً به همدیگر منتقل نمود.

```
MOV CL , -50
MOV X , 34H
MOV AX , DATA1
MOV ALI , BX
```

در مثال زیر دو دستور نوشته شده است . با توجه به نقشه حافظه نتیجه هر دستور چیست ؟



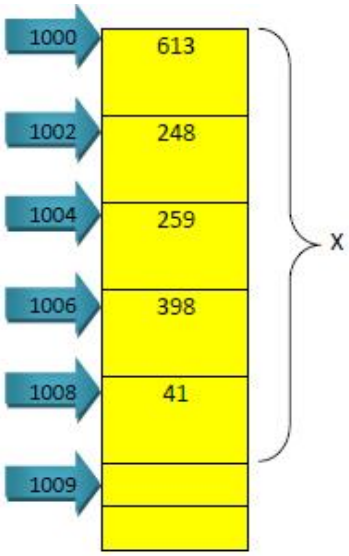
دستور اول آدرس متغیر Y را به ثبات BX منقل می کند . از آنجایی که متغیر Y دو بایت از حافظه را اشغال کرده است بنابراین آدرس آن برابر 2004H خواهد بود BX=2004H .

دستور دوم محتوای متغیر Y را به ثبات BX منقل می کند . بنابراین BX=300

مثال : با توجه به قطعه برنامه زیر محتوای AX و BX را پس از اجرای دستورات زیر بنویسید.

```
ORG 1000
X DW 613 , 248 , 259 , 398 , 41
MOV BX , OFFSET_X
MOV AX , [BX]+4
MOV DX , X+3
```

جواب : در خط اول آفست یا نقطه شروع از آدرس 1000 شروع شده است . در خط دوم یک متغیر به نام X تعریف شده است با 5 عدد دو بایتی که اگر بخواهیم برای این متغیر نقشه حافظه را رسم کنیم بصورت زیر می شود (: البته در بسیاری از موارد رسم نقشه حافظه لازم نیست)



در خط سوم آدرس متغیر X به ثبات BX منتقل می شود بنابراین : BX = 1000

در خط چهارم یک روش آدرس دهی غیر مستقیم نسبی پایه مشاهده می کنید . در این دستور محتوای آدرس 4+1000 به ثبات AX منتقل می شود . در آدرس 1004 عدد 259 ذخیره شده است بنابراین : AX = 259

DX=398

دستور : LEA : با اجرای این دستور آدرس متغیر مبداء در یکی از ثباتهای SI , DI , BP , BX قرار می گیرد.

Lea مقصد , مبداء

آفست مبداء --------> مقصد

-مبداء می تواند یک متغیر از نوع بایت یا کلمه باشد.

-از نظر نتیجه و اجرا این دو دستور زیر با هم معادل هستند:

```
LEA  BX , X
MOV  BX , OFFSET X
```

مثال : محتوای ثبات AL پس از اجرای قطعه برنامه زیر چیست ؟

```
ORG 100
DATA3 DB 23 , 47 , 35 , 83
LEA SI , DATA3 + 1
MOV AL , [SI]
```

جواب : خط اول آفست برنامه یا نقطه شروع را برای متغیر DATA3 تعیین می کند.

خط دوم یک متغیر از نوع بایت می باشد با چهار رقم که هر رقم با توجه به نوع متغیر بایت می باشند . بدین ترتیب عدد 23 در آدرس 100و عدد 83 در آدرس 103 ذخیره خواهد شد.

دستور مبادله : XCHG با اجرای این دستور محتوای عملوندهای مبداء و مقصد با هم مبادله می شوند.

مبداء ⟷ مقصد

XCHG مبداء , مقصد

-مبداء و مقصد نمی توانند ثابت باشند.

-عملوندها بایستی از نوع بایت یا از نوع کلمه باشند.

-هر دو عملوند با هم نمی توانند متغیر باشند.

مثال : محتوای ثبات AX و متغیر X را پس از اجرای قطعه برنامه زیر بنویسید:

```
MOV  AX , 1000
MOV  X , 3000
XCHG  X , AX
```

جواب : دو خط اول دستورهای انتقال است پس نتیجه:

AX=1000

X=3000

در خط سوم دستور مبادله بین ثبات AX و متغیر X داده شده که محتوای این دو با هم عوض می شوند

بنابراین نتیجه نهایی:

AX=3000

X=1000

مثال : محتوای ثبات AL و متغیرهای X و Y را پس از اجرای دستورات زیر بنویسید.

```
X DB 65
Y DB 48
MOV AL , X
XCHG AL , Y
MOV X , AL
```

جواب : خط اول X=65

خط دوم Y=48

خط سوم AL=65

خط چهارم AL=48 , Y=65

خط پنجم X=48

نتیجه : آخرین تغییرات AL=48 , X=48 , Y=65

دستور کاهش : با اجرای این دستور یک واحد از محتوای عملوند کم شده و نتیجه در عملوند قرار می گیرد.

DEC عملوند

عملوند - > عملوند -1

دستور افزایش : با اجرای این دستور یک واحد به محتوای عملوند اضافه شده و نتیجه در عملوند قرار می گیرد.

INC عملوند

عملوند <- 1+ عملوند

-عملوند می تواند از نوع بایت یا از نوع کلمه باشد.

-عملوند نمی تواند یک عدد ثابت باشد.

مثال : نتیجه خط آخر در قطعه برنامه زیر چیست ؟

```
Z DB 130
DEC Z
```

جواب:

در خط اول مقدار Z برابر با 130 میشود و در خط دوم از مقدار Z یک واحد کم میشود. پس Z=129 .

تمرین : نتیجه قطعه برنامه زیر چیست و مکان ذخیره نتیجه کجاست ؟

```
X DB 10 , 20 , 26 , 44 , 6
MOV BX , X
INC BX
```

پشته STACK

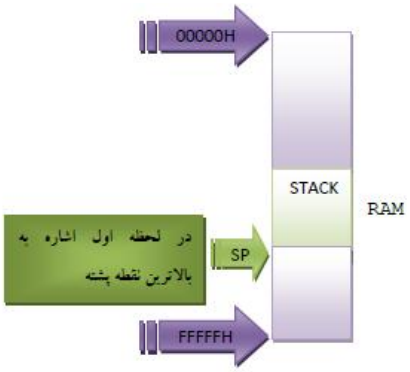
هر ثبات در 8086 بجز ثباتهای قطعه و SP قابل ذخیره سازی در پشته و دریافت از آن می باشند.

: PUSHذخیره در پشته (درج)

: POPبار کردن (دریافت) از پشته (بازیافت)

پشته بصورت LIFO است یعنی آخرین داده ای که در آن قرار گرفته اول خوانده می شود مانند چند کتاب که روی هم قرار گرفته شده باشد . اطلاعات در حافظه پشته بصورت یک کلمه یک کلمه (دو بایت دو بایت) نوشته می شود و یا از آن خوانده می شود.

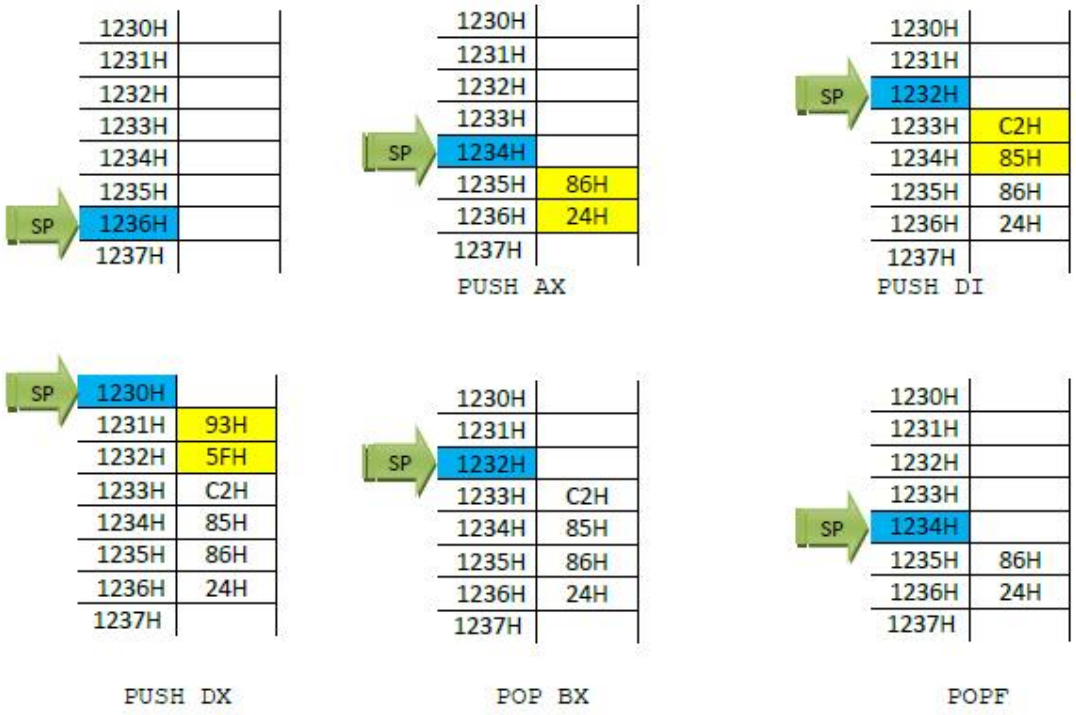
در 8086 ثبات SP به مکان حافظه جاری بکار رفته در بالای پشته اشاره می کند و به محض درج داده کاهش خواهد یافت 2 (واحد کاهش می یابد) و برعکس هنگام بازیافت این اشاره گر افزایش می یابد 2 (واحد افزایش)



مثال : با فرض AX=2486H,DX=5F93H,DI=85C2H,SP=1236H با اجرای دستورات زیر محتوای پشته و اشاره گر آن
و همچنین محتوای ثبات BX و پرچم را نشان دهید.

```
PUSH AX
PUSH DI
PUSH DX
POP BX
POPF
```

جواب :



تمرینهای فصل 3:

1) دستورات **push** و **pop** را توضیح دهید و بگویید که چگونه عمل میکنند؟
2) در مورد حافظه پشته چه میدانید؟
3) آدرس دهی مستقیم و آدرس دهی غیرمستقیم پایه را شرح دهید.
4) مقدار ثبات مقصد را در هر یک از دستورات زیر بیابید؟

```
X db 12
Y db 0d2 h
Z  db “ali”
Mov   al , x
Inc  al
Mov   bl, y
Dec  bl
Mov  ch, 92
Xchg  ch, bl
Mov  dh, z
```

توضیحات و مثالهای تکمیلی برای چند دستور:

دستورالعمل های انتقال داده مقادیر را از یک محل به محل دیگر کپی می کنند.

MOV
XCHG
LEA

MOV

ساده ترین دستورالعمل mov است که دارای دو عملوند است .این دستورالعمل محتوای دومین عملوند خود را در اولین کپی می کند .فرم کلی آن به صورت زیر است:

mov Dest, Source

دستور mov یک کپی از Source را گرفته و آنرا در Dest ذخیره می کند .محتوای Source بعد از اجرای دستور تغییر نمی کند ولی مقدار قبلی Dest رونویسی می شود.

دستور mov مشابه دستور انتساب در زبان های سطح بالا است ; Dest := Source (در زبان Pascalیا; Dest=Source در زبان C .)

با توجه به نوع عملوندها، انواع مختلفی از دستورالعمل mov را می توان داشت .متداولترین آنها عبارتند از:

mov register, register mov memory, register mov register, memory mov memory, immediate data mov register, immediate data mov AX/AL, memory mov memory, AX/AL mov segment register, memory 16 mov segment register, register 16 mov register 16, segment register mov memory 16, segment register

چند موضوع مهم درباره دستور mov را باید همواره بخاطر داشت:

1.انتقال حافظه به حافظه وجود ندارد .یعنی هردو عملوند همزمان نمی توانند عملوند حافظه ای باشند.

2.عملوندها می تواند از نوع بایت یا کلمه باشند .اما هردو عملوند حتما باید هم اندازه باشند(برای مثال دستور mov AX,BL اشتباه است .)این برای عملوند های حافظه و ثبات هم باید رعایت شود (اگر متغیری را یک بایتی تعریف کنید و آنرا در ثبات AX منتقل کنید اسمبلر پیغام خطا صادر می کند.)

3. با این دستور نمی توان یک داده فوری را در یک ثبات سگمنت منتقل کرد.

4.هر دو عملوند نمی توانند ثبات سگمنت باشند.

5.هر کدام از ثبات همه منظوره را می توان به جای AX بکار برد.

mov AX, 40h **mov ES, AX**

6.دستور mov روی هیچکدام از فلگ ها تاثیری ندارد.

XCHG

دستورالعمل xchg محتوای دو عملوند خود را جابجا می کند .فرم کلی آن به صورت زیر است:

xchg Operand1, Operand2

مقدار هردو عملوند در اثر اجرا تغییر می کند.

چهار شکل خاص برای این دستور وجوددارد:

xchg register, memory xchg register, register xchg ax, register16

ترتیب علموندها اهمیت ندارد .می توانید xchg mem,reg یا xchg reg,mem را بنویسید نتیجه فرقی ندارد .اکثر اسمبلرها بطور خودکار کد کوتاهتر را انتخاب می کنند.

هردو عملوند باید یک اندازه باشند.

دستور xchg روی هیچیک از فلگ ها تاثیر نمی گذارد.

LEA

دستورالعمل lea (load effective address) برای مقداردهی اشاره گرها استفاده می شود .فرم خاص آن به صورت زیر است:

lea register16, memory

این دستور آدرس موثر یک محل خاص از حافظه را درون یک ثبات همه منظوره ذخیره می کند . منظور از آدرس موثر آدرس نهائی حافظه بعد از کلیه محاسبات آدرسی است.

مثال .دستور زیر مقدار 1224 h را در ثبات AX قرار می دهد.

lea AX, DS:[1234h]

دستور mov ax, immediate data هم همین عمل را انجام می دهد .تفاوت آنها در این است که دستورالعملlea محاسبه آدرسی و انتقال داده را همزمان انجام می دهد.

مثال .دستور زیر آدرس حاصل از محاسبه BP+SI+4 را درثبات AX قرار می دهد .ابتدا مقادیر را بهم جمع کرده سپس در ثبات منتقل می کند.

lea bx, 4[bp+si]

دستورالعمل lea روی فلگ ها تاثیر ندارد.

INC

دستورالعمل inc (increment) یک واحد به عملوند خود اضافه می کند .شکل کلی آن به صورت زیر است:

inc dest

این دستورالعمل عدد 1 را با dest جمع و حاصل را در خود dest ذخیره می کند.

دستور inc به شکل های زیر می تواند باشد:

inc register inc memory

عملوند دستور می تواند ثبات یا مکانی از حافظه باشد .اندازه عملوند می تواند 0 یا 18 بیتی باشد.

دستور inc فشرده تر و اغلب سریع تر از دستور add است.
به استثنای فلگ Carry بقیه فلگ ها مشابه دستورالعمل add تغییر می کنند .توجه کنید که این دستور بر روی فلگ
Carryتاثیر ندارد و برای تاثیر روی فلگ Carry باید از دستورالعمل ADD استفاده شود.
افزایش شمارنده حلقه و اندیس آرایه یکی از متداولترین کاربردهای دستور inc است.

DEC

دستورالعمل dec (decrement) یک واحد از عملوند خود کم می کند و حاصل را در خود عملوند دخیره می نماید.

dec dest

عملوند دستور dec می تواند ثبات یا حافظه باشد.
به استثنای فلگ Carry بقیه فلگ ها مشابه دستورالعمل sub تغییر می کنند.