

1.1 تعلم الأسمبلي باللغة العربية - الدرس

الرئيسة | الدرس التالي | منتدى الأسمبلي | العودة الى الوحدة الأولى

| آخر تحديث : 2001/7/19 | عدد الزوار

ماهي الأسمبلي :

في قديم الزمان أيام بدايات الكمبيوتر كانت برمجة الكمبيوتر تتم بواسطة لغة الآله
Machine Language اختصاراً ML (لغة الآله هي اللغة التي تفهمها الآله مباشرة
دو ز الحاجة الى تفسير وهي تخزن بصورة ثنائية [تركيبية من الأصفار والواحد] في
الذاكرة على شكل تعليمات ووسائط تأخذ كل واحد منها عادة مقدار 8بت=1 بايت)
وكان هذا النوع من البرمجة صعب جداً عندها طور المبرمجون أو ل لغة برمجة وهذه
اللغة فكرتها بسيطة جداً حيث أنه بدل أن تكتب رموز الآله يتم كتابة كلمات مختصرة
تدل على نوع العملية مثال (MOV,ADD,CMP) ثم برنامج بسيط يتم تحويل هذه
الشفرة الى لغة الآله باستخدام تخطيط واحد-الى-واحد يأ ن كل سطر أو عبارة في
الأسمبلي تحول الى تعليمة واحدة مقابله في لغة الآله (مثال بدل كتابة
011000000000101 يتم كتابة mov al,5) يعرف البرنامج الذي يقوم بعملية التحويل
بالأسمبلر Assembler ، علماً بأن هناك عدة أنواع من الأسمبلر كل نوع يختص
بتقنية معينة وبعائلة معينة من المعالجات
ونحن هنا بصدد تعلم البرمجة بالأسمبلي للمعالجات المبنية على تقنية IBM-PC
والمنتجة من شركة إنتل وهي العائلة 80x86 ويرمز لها اختصاراً X86 وهي تضم :
(8086 / 8088 / 80186 / 80286) ل المعالجات لا 16 بت و (80386 / 80486 /
80586=بنتيوم1 / 80686=بنتيوم2 / 80786=بنتيوم3 / 80886=بنتيوم4)
لمعالجات لا 32 بت وسوف أتطرق في دروس متقدمة الى المعالج أتنيوم 64 بت
المبني بتقنية جديده كلياً لمن يرغب بمعرفة مسبقة لهذا المعالج الجديد كذلك سوف
أتطرق بأذن الله الى الكرو سب أسمبلر وهي مجموعة برامج خاصة مصممه للتحويل
من لغة أسمبلي لعائلة معالجات معينة الى عائله أخرى .

تعريف لغة الأسمبلي

الأسمبلي هي لغة برمجة تتكون من سلسلة من التعليمات المتتابعة كل تعليمة فيها تحول الى تعليمة
مقابلة بلغة الآله .

تعريف الأسمبلر

الأسمبلر هو برنامج يقوم بتحويل التعليمات المكتوبة بالأسمبلي الى لغة الآله .

لماذا أريد استخدام الأسمبلي :

بتعلمك لغة الأسمبلي فأنت تكشف النقاب عن الأسرار المختفيه وراء الكمبيوتر وتصبح
قادرأ على الفهم تماماً كيف يعمل المعالج وكيف يعمل البرنامج وبذلك تزيد خبرتك
كمبرمج وبالطبع فإن الأسمبلي أقوى من اللغات العالية المستوى في التعامل مع
العتاد وتعطيك مرونة عالية وقدرة وصول الى أشياء لم تكن تستطيع الوصول اليها من
قبل ، كذلك هناك نوعيات من البرامج لايمكن الا برمجتها بالأسمبلي مثل
الدرابفات(سواقات) الأجهزة ، كذلك فإن الأسمبلي يعطيط برامج سريعة جداً ، وبالطبع
فإن بناء برنامج متطور بالأسمبلي أشبه بحفر حفرة بواسطة الملعقه فالبرغم أنك تحفر
لا أنك أنت لحيك قليله ولكن من المحبذ جداً برمجة بعض الدلا و لأجزاء من البرلج
بالأسمبلي وبقيه البرنامج بواسطة لغة عالية المستوى مثل السي++ .

العلاقة بين الأسمبلي واللغات الأخرى :

تعتبر كل من لغة الآله و الأسمبلي لغتين منخفضتي المستوى Low-Level Language اختصاراً LLL لأنها تكتب تعليمه تعليمه (بعض الناس يعتبر لغة السي لغة منخفضة المستوى ، وهذا الكلام أرجح الى الخطأ منه الى الصواب) بينما تعتبر باقي اللغات High-Level Language اختصاراً HLL وفي هذه اللغات تختفي تقنية تخطيط واحد-الى-واحد وتفسر التعليمه الواحدة الى عدد كبير من تعليمات لغة الآله

مخطط يوضح العلاقة بين لغة الأسمبلي ولغة الآله ولغة عالية المستوى ولغة الآله

تطبيقات لغة الأسمبلي :

تتطلب كتابة البرامج بلغة الأسمبلي معرفة بالعتاد وعناية خاصة مع الأهتمام بأدق وأقل التفاصيل ، في أيام البرمجة القديمة كان المبرمجون يكتبون برامجهم بلغة الأسمبلي لأن ذاكرة الرام وقتها كانت صغيرة (أقل من 64 كيلوبايت) وهم بحاجة الى برامج أصغر وأسرع خصوصاً في معالجتهم أيضاً كانت بطيئة ، مع تطور الحاسوب وتوسع سعة ذاكرة الرام وزيادة سرعته أصبحت البرامج أكثر طولاً وتعقيداً ، هذا التعقيد أدى الى استخدام اللغات البرمجية عالية المستوى HLL مثل السي والكوبول والبيسك والباسكال والفورترن ، مرة أخرى تطور الحاسوب فأدى الى استخدام اللغات العليا الموجهة الهدف OOP مثل السي++ والجافا والتي مكنت من كتابة برامج قوامها آلاف الأسطر والتعليمات المعقدة والمتداخلة .

من الصعب أن تلاقى برامج كبيرة مكتوبة كاملة بلغة الأسمبلي لأن كتابتها صعب والأهم من ذلك تطويرها وصيانتها أصعب بكثير ، بدل ذلك يقوم المبرمجين ببرمجة مقاطع مبرمجة برمجة مثل بلغة الأسمبلي لأستخدمها في تنفيذ أسرع أو الوصول الى العتاد عن طريقها وباقي البرنامج بواسطة لغة عالية المستوى .

يفض المبرمجين لغة السي++ كلفة قياسية للبرمجة بلغة عليا لأن لها قدرة عالية وقوية جداً وموجهة الهدف مع القدرة على كتابة مقاطع السي فيها وهي لغة أقل أنخفاصاً وأكثر مرونة مع استخدام الأسمبلي كعنصر مهم في الوصول الى العتاد وبرمجة الجزئيات المحتاجة للسرعة .

لا يستخدم المبرمجون شفرة الأسمبلي وسط شفرة لغة عالية المستوى عادة وإنما يستخدمونها عن طريق واجه على شكل دالة أو كائن وتحتوي هذه الدالة أو هذا الكائن على شفرة الأسمبلي المطلوبة ، وقد تستخدم روتيناً فرعياً أو دالة في لغة عالية

المستوى وأنت لتعلم بأنك باستدعاء هذه الدالة أو الروتين الفرعي قد أستخدمت شفرة مكتوبة بلغة الأسمبلي .

لغة الآله :

لغة الآله كما ذكرنا سابقاً هي اللغة التي تستطيع الآله أو المعالج التعامل معها مباشرة ، في العائلة X86 كل معالج يحتوي يستطيع تنفيذ تعليمات المعالج الذي قبله ويملك مجموعة تعليمات موسعة وأضافة لا تستطيع المعالجات التي قبله تنفيذها وأما المعالج الذي يأتي بعد هذا المعالج فإنه يدعم التعليمات الموسعة للمعالج الذي قبله بينما يحتوي هو أيضاً على تعليمات جديدة وموسعة ، بأختصار إذا صممت برنامج لمعالج ما فإن المعالجات ما قبل هذا المعالج لن تستطيع تشغيله بينما المعالج نفسه والمعالجات التي بعده (من نفس العائلة) تستطيع تشغيل البرنامج .

لقد حافظت شركة إنتل على التوافقية في العائلة X86 ابتداء من المعالج 8086 وصولاً الى باتنتيوم 4=80886 ولكن الحفاظ على التوافقية يفرض قيود على تصميم المعالج وأستخدام تقنيات قديمة ، ومؤخراً قرر إنتل إيقاف عائلة المعالج X86 عند باتنتيوم 4 وقامت بإنشاء معالج جديد (غير متوافق مع العائلة X86) مبني على تقنية 64 بت وهو المعالج اتانيوم الجديد .

مثال على لغة الآله :

التعليمية 101100000000101 هي تعليمة بلغة الآله ومعناها أنقل الرقم 5 الى المسجل ah يقابلها بلغة الأسمبلي mov ah,5 لا 8 بت الأولى من التعليمة تشكل **شفرة التعليمة** OP-code=operation code وهي تعني نقل قيمة بطول 8 بت الى المسجل AL ، لجزء الثاني من لا 16 بت الأخرى تشكل الرقم 5 ثانياً

إضغط هنا للانتقال الى الدرس التالي

إعداد : أحمد الزياتي

تعلم الأسملي باللغه العربيه - الدرس 1.2

الرئيسة | الدرس التالي | منتدى الأسملي | العودة الى الوحدة الأولى

| آخر تحديث : 2001/7/19 | عدد الزوار

قد تتسائل ما علاقة تمثيل البيانات والعد الثنائي بالأسملي ؟ حسناً كما وضحت من قبل فإن الأسملي هي لغة قريبة جداً من لغة الآله وهي لغه منخفضة المستوى تتعامل مع العتاد والمعالج بصورة مباشرة ولكي نحقق فهماً أوسع لهذه اللغه يجب أن نفهم بعض الأشياء المهمة جداً في بنية المعالج .

العد الثنائي :

يتم تمثيل الشفرا ت والبيانات في ذاكرة الكمبيوتر كتوالييف من الشحنات الكهربائية تأخذ قيمتين الأولى وهي وجود الشحنة ويرمز لها ب ON أو صحيح TRUE أو '1' والأخرى وهي غياب الشحنة ويرمز لها ب OFF أو خطأ FALSE أو '0' ، ووجود الشحنة يكون عادة بين 4.5 الى 5.5 فولت (المعالجات الحديثه بين 2.5 الى 3.5 فولت) وغياب الشحنة يكون بين 0.5+ فولت و -0.5 فولت .

وحدا ت الذاكره الأساسيه في الذاكره والوحدا ت التي سنتعامل معها كثيراً هي :

أسم الوحدة	الأحتمالات	مايعادلها
البت	أحتمالين 0 و 1	1بت
البايت	256 احتمال	8بت
الكلمة	65536 احتمال	16بت=2بايت
الكلمة المضاعفة	4,294,967,296 احتمال	32بت=4بايت=2كلمة
الكلمة الرابعة	18,446,744,073,709,551,616 احتمال	64بت=8بايت=4كلمات=2كلمة مضاعفة

بالطبع هذه الوحدا ت الأساسيه والصغيره أما مضاعفاتنا فهي :

الوحده	الرمز	مايعادلها
الكيلوبايت	KB	1024 بايت
الميجابايت	MB	1024 كيلوبايت
الجيجابايت	GB	1024 ميغابايت
التيرابايت	TB	1024 جيجابايت
البيتابايت	PB	1024 تيرابايت
الأكسابايت	EB	1024 بيتابايت
الزيتابايت	ZB	1024 أكسابايت
اليوتابايت	YB	1024 يوتابايت

تخزين الأرقام بدو ن إشارة يأخذ النطاق الآتي لكل وحده من الوحدا ت الأساسيه باستخدام طريقه بدو ن الإشارة : unsigned:

أسم الوحدة	المجال
البايت	من 0 الى 256
الكلمة	من 0 الى 65536
الكلمة	من 0 الى 4,294,967,296

المضاعفة		
الكلمة الرباعية	0	18,446,744,073,709,551,616

الأسكي كود ASCII:

يتم في الحاسوب وبقيّة توحيد استخدام الرموز استخدام شفرة الأسكي كود (حالياً) يعمل على تبني شفرة الـ unicode وهي تسمح بتعدد اللغات في مستند واحد حيث يتم تمثيل كل حرف باستخدام كلمة واحدة=2بايت) كلمة ASCII هي اختصار لـ :
Interchange Information Code For Standard American National
ويتم استخدام هذا الكود الموحد لتسهيل تناقل البيانات ويمثل كل رمز فيه بعدد ثنائي بطول 1بايت=8بت=256أحتمال .

طريقة كتابة الأرقام في الأسميلر :

لكتابه عدد ثنائي يوضع في آخر الرمز (b) لدلالة على أنه باينري مثال :
Binary=11010010B أما العدد العشري فلايحتاج الى إضافة وأما العدد لأساس 8 فيكتب مع المرمز (Q) في نهايته Octal=1276Q أو الرمز ((O في نهايته Octal=1276O أما العدد السداسي عشر فيكتب بوضع H في نهايته hexadecimal=0AB9CDH مع مراعاة وضع 0 اذا كان العدد يبدأ بحرف كما المثال .

يجب أن تعرف الفرق بين تخزين الرقم كرقم أو تخزينه كنص فتخزين الرقم 201 مثلاً كرقم سسياخذ بايت واحد وهو جاهز للقيام بعمليات رياضية ومنطقيه عليه أما تخزينه كنص فسياخذ ثلاثة بايت في البايت الأو ل سيخزن الرقم الخاص بالأسكي كود للرمز '2' وكما قلت يخزن كرقم يدل على الرمز أما البايت الثاني فسيزن رقم الأسكي كود للرمز '0' أما البايت الثالث فيأخذ قيمه الخاصة بالرمز '1' في الأسكي كوداً يـأ ز الرقم خزن بطريقة "102" وليس 102 وهذه الطريقة ليست جاهز للجمع أو الطرح ولكنها ممتاز للطباعه على الشاشة ويمكن تحويل النص الى رقم والعكس .

الأعداد في الأشاره :

يتم تخزين الأعداد في الأشاره كالتالي :
العدد موجب إذا كانت البت الأخيره صفر وقيمة الرقم هي باقي البتات أ ي لو أخذنا رقماً من بايت واحد فإن البت رقم 7 (الثامن - الترقيم يبدأ من الصفر) يجب أن تكون صفراً ليكون العدد موجب أما البتات من 0 الى 6 (السبعه الأولى) فتشكل قيمة الرقم أما إذا كان العدد سالب فإن البت الأخيره تساو ي واحد أما قيمة الرقم فتساو ي سالب المكمل الثنائي للعدد أ ي لو أخذنا رقم مخزن في واحد بايت مثال = 11110110 بما أن البت السابعه=1 فإن الرقم سالب / نأخذ الآن المكمل الثنائي للعدد وهو 00001010 / القيمة تساو ي -00001010أ ي سالب عشرة .

وتخزين الرقم بالأشارة يختلف النطاق الذي تأخذ كل وحده :

الاسم	المجال	الاسم
الوحدة	من	الى
البايت	128-	127+
الكلمة	32,768-	32,767+
الكلمة المضاعفة	2,147,483,648-	2,147,483,647+
الكلمة الرباعية	9,223,372,036,854,775,808-	9,223,372,036,854,775,807+

تعلم الأسمبلي باللغة العربية - الدرس 1.3

الرئيسية | أسئلة الوحدة الأولى | منتدى الأسمبلي | العودة الى الوحدة الأولى

| آخر تحديث : 2001/7/19 | عدد الزوار

تعليمات وأوامر الأسمبلي :

تتكون التعليمه الواحدة في الأسمبلي من تمثيل بسيط بالأحرف الأنجليزيه يقابله بالأرقام تعليمه لغة آله ، تتكون كل تعليمه من ممابلي : أولاً جزء الأمر وهو أمر يدل على نوع العمليه المطلوبه مثل ADD (للجمع) ، الجزء الثاني هو الوسائط علمياً بأن بعض التعليمات لا يأخذ وسائط والجزء الآخر وسيطة واحدة فقط والبعض الآخر أكثر من ذلك ، تحدد هذه الوسائط الشئ الذي سيعمل عليه الأمر ، فالأمر ADD لوحده عقيم لا يدل على شئ لكن الأمر ADD AX,5 يدل على جمع الرقم 5 مع القيمة الموجوده في المسجل AX ويوضح المثال التالي بعض الأوامر

```
clc ; وسائط فقط أمر بدو ن  
dec ax ; وسيطة واحدة فقط  
mov cx,dx ; وسيطتين
```

لاحظ أن أ ي نص في شفرة الأسمبلي يأتي بعد الفاصله المنقوطة هو مجرد تعليق

الوسائط ممكن تكون عدة أنواع :

1. معلومه فوريه (مباشرة) (أ ي ثابته) مثال : 10 / 30 / a"
2. مسجل (سوف يتم شرح المسجلات بالتفصيل في الدروس القادمه) مثال : AX / EAX / BL
3. موقع ذاكره (يتم تحديده عن طريق العنوا ن) مثال : [200] / [100] / bx
4. متغير (وهو نفس السابق لكن بدل أ ن تحفظ أو تحسب العنوا ن يدوياً يقوم الأسمبلير ب استبدال المتغير برقم يدل على عنوانه) مثال : count / VAR1 / INTVAL / STR1

مدخل الى الديوغر Debugge :

ها قد وصلنا الى واحد من أقوى البرامج المبيته في النظام فيواسطه اليبغ تستطيع عمل أشياء عجيبه وغريبه ، حسناً شغل الدوس وعند محث الأوامر أطبع debug ثم أنتر وستظهر لك علامه '-' ليل على أستعداد الديوغر على أستقبال أوامرك .

الآن دعنا نكتب هذا البرنامج الصغير

```
mov ax,2 ; مباشرة الى المسجل أ ب-أكس نقل العدد 2 كمعلومه  
mov bx,3 ; المسجل بي-اكس نقل العدد 3 كمعلومه مباشرة الى  
add ax,bx ; أ ب-اكس / أ ب-اكس = أ ب-اكس + بي- جمع أ ب-اكي مع بي-اكس مع وضع الجوا ب في  
أكس
```

- كيف تقوم بأدخال هذا الكود :
1. عند المحث '-' أدخل a100 أ ي أننا سنبدأ نكتب الكود من العنوا ن 100 ثم أضغط أنتر بالطبع
 2. الآن أدخل كل تعليميه ثم أضغط أنتر ومع نهاية التعليمية الأخيرة أضغط أنتر مرتين

الآن قم بأدخال الرمز R ثم أنتر لترى حالة المسجلات
لاحظ أن المسجل AX يساو ي صفر وسترى أيضاً ظهور التعليميه MOV ax,0002 وهي
التعليميه التي عليها الدور في التنفيذ وليس المعلومة المنفذه ، الآن قم بطباعة الرمز
T ثم أنتر لتنفيذ التعليميه التي عليها الدور هنا هي MOV AX,0002 ستري الآن أن
المسجل AX أصبح يساو ي 2 وهذا ما نتوقعه بالضبط وسترى أيضاً التعليميع التي عليها
دور التنفيذ وهي MOV BX,0003 أدخل الرمز T ثم أنتر لتنفيذها لترى أن المسجل BX
أصبح يساو ي 3 وسترى أيضاً التعليميه التي عليها الدور في التنفيذ وهي ADD AX,BX
قم بأدخال الرمز T لتنفيذها ولاحظ كيف أن المسجل AX أصبحت قيمته مجموع العددين
3+2 وهو خمسة بينما بقى المسجل BX يساو ي 3 .

الآن بعدما عرفت كيف تكتب كود بسيط أخرج من الديوغر بالضغط على Q ثم أدخل مرة
أخرى بكتابة الأمر Debug حتى تصفر المسجلات مرة أخرى أدخل التعليميه A100 ثم
جرب تكتب كود من عندك ومع كل نهاية تعليميه أضغط أنتر وفي نهاية التعليميه الأخيرة
أضغط أنتر مرتين
(ملاحظة لترى شفرتك بلغة الآله والأسمبلي أدخل الرمز U ثم أنتر مباشرة بعد إدخال
الكود وقبل إدخال الرمز R)
أضغط R ثم أنتر لترى المسجلات قبل تنفيذ أ ي عملية ولترى التعليميه التي عليها
الدور في التنفيذ أضغط T ثم أنتر لتنفيذ التعليميه وترى النتائج والتعليميه التي بعدها
وهكذا ولا تنسى إذا أردت أن تدخل كود جديد الخروج والعودة مرة أخرى الى الديوغر
لتصفر المسجلات والذاكره

