

مقدمة

ما هي لغة التجميع؟

تصمم وحدة المعالجة المركزية CPU لتنفيذ مجموعة صغيرة من التعليمات التي تدعى بتعليمات "لغة الآلة" machine language وكل تعليمة في هذه اللغة تحدد العملية الواجب اجراؤها على بعض المعاملات operands فمثلاً، تتم احدى العمليات النموذجية لجمع معاملين، حيث تشفر العمليات كسلسلة من البتات التي يمكن تفسيرها من قبل وحدة المعالجة المركزية CPU.

تخزن المعاملات بشكل طبيعي في الذاكرة ويحدد العنوان الذي يخزن عنده المعامل ضمن تعليمة الآلة. فمثلاً، تمثل السلسلة 0000010100001010 تعليمة نموذجية بلغة الآلة، وفيها نجد ان البتات (الخانات) الثمانية الأكثر اهمية most significant bits تمثل شفرة العملية operation code بينما يمثل العنوان بالبتات الثمانية الاقل اهمية. على هذا الاساس فان كل التعليمات هي عبارة عن تسلسل من البتات التي يمكن فهمها وتفسيرها وتنفيذها بواسطة جملة من الدارات الموجودة في وحدة المعالجة CPU.

ان هذا التسلسل من البتات يعتبر صعب الفهم من قبلنا كبشر ولذلك يتم ترميز تعليمات لغة الآلة باستخدام مجموعة مختصرات mnemonics لتمثيل شفرات العمليات ومجموعة رموز symbols لتمثيل عناوين الذاكرة بالاضافة الى المسجلات في الوحدة CPU. تكتب تعليمة الآلة السابقة مثلاً بواسطة المختصرات كما يلي:

CODE

```
mnemonic dest, src ; comments goes here
```

وتسمى هذه التعليمة بتعليمة لغة التجميع.

ان كل تعليمة بلغة التجميع هي تمثيل مقابل لتعليمة بلغة الآلة بمعنى ان كل تعليمة في لغة التجميع ستترجم الى تعليمة محددة في لغة الآلة وبالتالي فاي تعليمة تستطيع الوحدة CPU تنفيذها ستكون مكتوبة وفق تعليمة مقابلة لها بلغة التجميع. تعتمد لغة التجميع حسب طبيعتها الخاصة على الآلة المستخدمة بمعنى ان برنامج لغة التجميع المكتوب من اجل حواسيب IBM يمكن تنفيذه فقط على حواسيب متوافقة مع IBM ولا يمكن تنفيذه على حواسيب Apple.

يترجم برنامج لغة التجميع الى برنامج لغة الآلة بواسطة برنامج اخر يدعى المجمع assembler وصتمم المجمعات معا برمجيا لتلائم معظم الحواسيب الشائعة مثل حواسيب IBM والحواسيب المتوافقة معها والتي تزودنا بها شركات شهيرة مثل Microsoft و Borland.

لغة التجميع واللغة عالية المستوى:

ان اللغات عالية المستوى مثل C, Pascal هي لغات مستقلة عن الآلة فالشفرة المصدرية المكتوبة بلغة C مثلاً يمكن تنفيذها على اي حاسوب يتوفر فيه مترجم C وبما ان لغة C مستقلة عن الآلة فان تفاصيل البنية الحاسوبية الخاصة تكون مخفية عن مبرمج لغة C لانها غير مرتبطة بهذه البنية. لذلك فان برامج C لا تستطيع

مباشرة ان تعالج محتويات مسجلات الوحدة CPU لمعالج محدد.

لقد صممت برامج اللغة عالية المستوى ايضا لتكون مختصرة وقابلة للقراءة فتعلية
مثل $c = a + b$ سوف تترجم بواسطة المترجم الى عدة تعليمات من لغة الالة. لذلك فان
برامج اي لغة عالية المستوى هي مجموعة من التعليمات اقل بكثير اذا ما قابلناها
ببرامج لغة التجميع.

تعتبر البرامج المكتوبة بلغات عالية المستوى اقل فعالية عند تحويلها الى لغة الالة
بالمقارنة مع البرامج المكتوبة بلغة التجميع، لان اللغات عالية المستوى هي لغات
مستقلة عن الالة ومصممة بالدرجة الاولى لمساعدة المبرمجين على الفهم بدلا من
تحسين فعالية الالة. فمثلا البرنامج المكتوب بلغة ++C لظهار رسالة " Hello
World!" سيحتاج عند تحويله الى شفرة الالة الى حوالي 8000 بايت بينما اذا كتب
بلغة التجميع فانه سيحتاج الى حوالي 600 بايت تقريبا. على هذا الاساس تعتبر برامج
لغة التجميع اكثر فعالية من برامج اي لغة عالية المستوى. بمعنى ان برامج لغة
التجميع المكتوبة للعديد من التطبيقات تتطلب حجما اقل وتنفذ بسرعة اكبر بالمقارنة مع
البرامج عالية المستوى لنفس التطبيق. لكن كتابة برنامج بلغة التجميع يحتاج الى خبرة
عالية وجهد بشري كبير وهذا ناتج عن حقيقة ان مبرمج لغة التجميع يجب ان يكون
لديه تصور وفهم عميق لوحدة المعالجة CPU التي يبد استخدامها.

تتميز برامج لغة التجميع بطولها والصعوبة الزائدة في اكتشاف الاخطاء وتصحيحها
بدلا من تطويرها, كما انه من الصعب ايضا ادخال ميزات جديدة على برامج موجودة
بلغة التجميع بمعنى ان هناك صعوبة واضحة في تطوير برامج لغة التجميع والمحافظة
عليها او الاحتفاظ بها.

لماذا نحتاج الى لغة التجميع؟

يوجد نوعان من البرامج المكتوبة للتنفيذ على الحواسيب هما: برامج التطبيقات **application programs** وبرامج النظام **system programs**.
تكتب برامج التطبيقات بصورة طبيعية بلغة عالية المستوى لتحقيق مهام مثل لوائح الجرد والتحكم ومعالجة المتحولات والطلبات التجارية وغيرها. لكن بالمقابل فإن برامج النظام هي برامج عامة جدا مهمتها تبسيط استخدام الحاسوب ومن هذه البرامج نذكر المترجمات **compilers** والمحررات **editors** وأنظمة التشغيل **operating systems**.

تستطيع أنظمة من بين هذه البرامج إدارة موارد النظام وإذا كان المطلوب وصل أجهزة خارجية الى الحاسوب والتحكم بها فسوف نحتاج الى البرمجة عند مستوى وحدة المعالجة لان برامج اللغة عالية المستوى مستقلة عن الآلة **machine independent** ولا تستطيع العمل عند هذا المستوى وبالتالي سنحتاج الى استخدام لغة التجميع لكتابة برامج للتحكم بأجهزة الدخل / الخرج. توجد حالات أخرى أيضا تحدث فيها شروط استثنائية أثناء الحساب مثل الحصول على نتيجة كبيرة جدا أو صغيرة جدا يصعب تخزينها في سجل ال **CPU** وفي مثل هذه الحالات يتخذ فعل تصحيحي للتغلب على المشكلة فقط باستخدام برنامج لغة التجميع ولذلك تعتبر لغة التجميع أساسية لكتابة البرامج التي تعالج المشاكل البارزة عند مستوى بنية وحدة المعالجة **CPU**.

متى نستخدم لغة التجميع؟

تمت الإشارة في المقطع السابق الى ان برامج لغة التجميع تستخدم للتحكم بالأجهزة الموصولة مع ال **CPU** ولمعالجة الشروط الاستثنائية التي تحدث عند مستوى مسجلات الوحدة **CPU**. وتستخدم لغة التجميع أيضا الى جانب هذه الحالات عند اعتبار الفعالية بانها الحالة الأفضل فعلى سبيل المثال ينبغي اعطاء امر للتحكم بطائرة ضمن وقت محدد بعج انجاز بعض الحسابات وفي مثل هذه الحالة يجب تنفيذ برنامج التحكم

خلال فترة قصيرة قابلة للتنبؤ ومحددة مسبقا ولذلك يكتب البرنامج عادة بلغة التجميع. تستخدم لغة التجميع لانه الى جانب الفعالية ان يتم الحصول على تخمين جيد للزمن اللازم بواسطة برنامج لغة التجميع (حيث ان لكل تعليمة او -مختصر- في لغة التجميع زمن خاص بالتنفيذ حسب عدد الدورات وسيتم شرح هذه النقطة لاحقا).

اما الحالة الثالثة التي تعتبر فيها لغة التجميع هامة فهي عندما يكون التطبيق ملموس الكلفة وهناك عدد كبير من الانظمة التي تحتوي بداخلها على برنامج هذا التطبيق المخفي، مثل نظام التحكم والمراقبة المعتمد على معالج صغري للحركة الالية automobiles, او المتحكم المستخدم في آلات الغسيل وغيرها, وكلها تنتمي الى هذا الصنف. في مثل هذه الحالات تكتب البرامج بشفرة التجميع للتاكيد على ان برنامج شفرة الالة المترجم سوف لن يحجز اكثر من سعة الذاكرة الصغيرة ROM الموجودة في النظام.

الاجزاء الرئيسية للغة التجميع:

تتألف تعليمة لغة التجميع في شكلها الابسط من شفرة لغة التشغيل الممثلة بمختصر mnemonics متبوع بلانحة من المعاملات operands حيث ان المعنى من كلمة المختصر هو المساعدة على التذكر حيث انه من السهل تذكر المختصر mov بدلا من الشفرة الثنائية 10110000 التي هي شفرة الالة لهذا المختصر. كمثال على تعليمة بلغة التجميع:

CODE

```
mnemonic dest, src ; comments goes here
```

تستخدم الفاصلة المنقوطة هنا لفصل تعليمات لغة التجميع عن التعليقات
comments المدرجة اذ ينصح دوما باستخدام التعليقات مع كل تعليمة في لغة
التجميع لشرح الوظيفة التي تؤديها تلك التعليمة وهذا العمل هام بصورة خاصة في لغة
التجميع باعتبار ان التعليمات موجزة ومختصرة.

نجد من المثال السابق ان تعليمة لغة التجميع تتألف من المختصر ومن المعاملات كما
ذكرنا سابقا. يحدد المختصر التعليمة المراد تنفيذها مثل `add, mov, mul, jx` الخ,
بينما تتألف المعاملات من قسمين:

dest وهو اختصار ل **destination** اي الوجهة وهو المكان الذي سيتم توجيهه
القسم الثاني اليه.

src وهو اختصار ل **source** اي المصدر حيث يتم نقل المعطيات في لغة التجميع
بترتيب عكسي من ال `source <- destination`.

الان ناتي للتوسع, حيث يتألف المعامل من عدة انواع.. فهو اما ان يكون:
- قيمة ثابتة تعرف بالمعامل الفوري **immediate value** مثل القيمة 10 مثلا.
متحول **variable** مثل **sum** يمثل اسما رمزيا لموقع في الذاكرة حيث تخزن قيمة
هذا المتحول.

- اسم **name** لاحد مسجلات الوحدة مثل **ax**
- عنوان مطلق **absolute address** لموقع ذاكرة مثل **[Bh0h04]** – لاحظ ان
القيمة العددية تكون ممثلة بالنظام الست عشري.

يتألف برنامج لغة التجميع من جزأين رئيسيين الاول يقدم التوجيهات **directives**

للمجمع لتعيين مكان في الذاكرة من اجل المتحولات المراد استخدامها والقيم المراد تخزينها والفراغ المراد تحديده للمكدس stack الخ.

بينما يشكل الجزء الثاني التعليمات القابلة للتنفيذ **executable instructions** في برنامج لغة التجميع.

فيما يلي جزء صغير من برنامج بلغة التجميع (16 بت) الذي يقوم بطباعة محرف في نافذة ال DOS:

CODE

```
.data ; beginning of data segment
x db 0 ; define 1 byte x. set x = 0

.code ; beginning of code segment
main proc
    ; beginning of main procedure
    add x, 4
    ; add 4 to x (hex addition)
    mov dl, x
    ; move results to DL register
    mov ah, 2
    ; DOS service, write char to output
    int 21h
    ; call interrupt
    mov ah, 4Ch ;
```

```
DOS service, end & return to DOS
    int 21h
                                ; call interrupt
main endp
                                ; end of procedure
```

الى هنا ناتي الى ختام هذا المقال حول لغة التجميع.
ساقوم بمتابعة هذه المواضيع ان وجدت تفاعلا من الاعضاء والا فانها ستنتظر الى
وقت قريب

مراجعة للمقاطع المستخدمة في هذا المقال:

CODE

```
INT 21 - DOS 1+ - WRITE CHARACTER TO STANDARD
                                OUTPUT
                                AH = 02h
                                DL = character to write

Return: AL = last character output (despite
                                the official docs which state
                                nothing is returned) (at least DOS 2.1-7.0)
```