

Les langages d'assemblage ou assembleur.

Mamma Mohand Fabre Clement

Département informatique Université de Montpellier 2

TCCP, 2014

Sommaire

- 1 Introduction
- 2 Histoire
- 3 Principes de programmation en Assembleur
 - Programmer en assembleur
 - La syntaxe de l'assembleur
 - Instruction
 - Exemple de code
 - Assembleur et langages de haut niveau
- 4 Usage
 - Souci d'efficacité
 - Cas d'utilisation
- 5 Conclusion

Introduction

Un langage assembleur (ou d'assemblage) est un langage de bas niveau qui représente le langage machine sous une forme lisible par un humain. Le terme assembleur est aussi utilisé pour désigner le programme qui traduit le langage assembleur pour le convertir en langage machine.

Histoire

- En 1949, les programmes du premier ordinateur à programmes enregistrés étaient rédigés en utilisant des mnémoniques alphabétiques. La traduction était alors faite à la main par les programmeurs.
- Le premier programme assembleur a été écrit par Nathaniel Rochester en 1954.
- Les langages assembleur ont éliminé une grande partie des erreurs commises par les programmeurs. On a donc commencé à les utiliser dans toutes sortes de programmes.

Programmer en assembleur

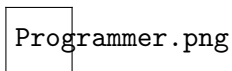


Figure : Programmer en assembleur.

Petit Rappel

Processeur

Definition

Composant de l'ordinateur qui exécute les instructions machines des programmes informatique.

Les parties essentielles d'un processeur:

- Unité arithmétique et logique.
- Unité de controle.
- Les registre.

Petit rappel

Registre

Definition

C'est la mémoire la plus rapide d'un ordinateur .

L' architecture externe du processeur définit un ensemble de registre.

Ces registres sont accessible par leur jeu d'instruction .

Petit rappel

Registre

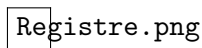


Figure : Différents registres existants.

Section

Un programme en assembleur peut etre divisé en 3 sections:

- .txt :contient les instructions et les constantes du programme.
- .data :décrit comment allouer l'espace mémoire.
- .bss :contient les variables non initialisées.

Declaration

Peuvent prendre quatre formats .
Se termine par un saut a la ligne ou un ;.

Illustration:

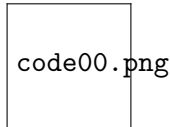


Figure : Exemple de declaration.

Important

le symbole . : peut etre utilisé comme référence a une adresse lors de l'assemblage.

Symbole

Séquences de caractères choisie parmi :

- Les lettres de l'alphabet.
- Les chiffres décimaux.
- Les caractères :.\$

Symbole

à connaître :

- Les Symboles ne doivent jamais commencer par un chiffres.
- Un symbole doit commencer en début de ligne
- La casse est significatives.
- Un symbole suivie de : est appelé étiquette.

Important

Le symbole spécial `.` peut être utilisé comme une référence à une adresse au moment de l'assemblage.

Exemple:

`len = . - msg`

Symbole

Symbole local

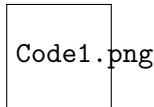
Le GNU propose 10 symbole locaux [0,1,..... ,9].

Annotation


Nb:se réfère au dernier symbole N défini

Nf:se réfère au premier symbole N suivant

Exemple:



Nombre entier



Nombreen entier.png

Figure : Representation des constantes.

Chaine de caractères

Definition

C'est une séquence de caractères écrite entre guillemets.
Elle représente un tableau contigu d'octets en mémoire.

Exemple de Code:


```
msg : .asciz "Hello, World !"
```

Instruction

Definition

L'élément clé de l'ordinateur car c'est elle qui permet de spécifier au processeur l'action à effectuer.

Toutes les instructions sont stockées en mémoire.



Instruction.png

Figure : Une Instruction.

Instruction

Instruction d'affectation

Definition

Permettent de faire des transferts de données entre les registres et la mémoire.

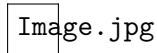


Figure : Une Instruction d'affectation.

Instruction

Instruction arithmétique et logique

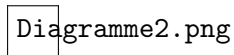


Figure : Instruction arithmétique et logique.

Instruction

Liste d'instruction

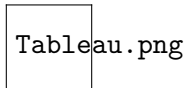


Figure : Liste d'instruction.

Exemple de code

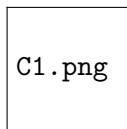


Figure : Programmer en assembleur.

Exemple de code

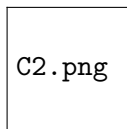


Figure : Programmer en assembleur.

Assembleur et langages de haut niveau

- Certains langages permettent d'ajouter du code ASM dans la source d'un programme.
- Cette technique est utilisée principalement par souci de rapidité.

Exemple en C++:

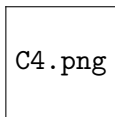


Figure : Liste d'instruction.

Souci d'efficacité

Dans beaucoup de cas, des compilateurs-optimiseurs peuvent transformer du langage de haut niveau en un code qui tourne aussi efficacement qu'un code assembleur écrit à la main par un très bon programmeur.

Manuel FORTRAN (1956) :

Object programs produced by Fortran will be nearly as efficient as those written by good programmers

Cas d'utilisation

- Calculs complexes
- Routines (ou drivers)
- Tâches effectuées dans l'espace mémoire du système d'exploitation
- Debogage et profilage
- Micro-contrôleurs limités en ressources

Conclusion

La programmation en assembleur est beaucoup plus longue, plus délicate et donc plus coûteuse que la programmation en langage de haut niveau. Il est donc conseillé de ne l'utiliser que si on peut pas faire autrement.

Merci de votre attention