1. Introduction

L'exécution d'un programme consiste à donner à la machine une séquence d'instructions directement interprétables par elle. Obligatoirement, les premiers programmes étaient écrits en binaire. C'était une tâche difficile et exposée aux erreurs : C'était l'époque des langages machine (binaire). Par la suite et pour faciliter le travail, les programmes ont été écrits en donnant directement des noms (abrégés) aux opérations : On les a appelés les codes mnémoniques (ADD, SUB, MUL, DIV, MOVE , etc.). Les adresses des instructions et des variables pouvaient être données sous forme symbolique. Pour pouvoir utiliser effectivement tous ces symboles, il fallait trouver le moyen de les convertir en langage machine : Ce fut réalisé par un programme spécial, appelé outil ou traducteur "assembleur".

Bien que remplacé par les langages évolués et relégué au deuxième plan, le langage d'assemblage reste un langage important car il permet d'accéder directement à toutes les ressources de la machine. Il permet également d'optimiser l'utilisation des ressources de la machine et d'avoir des programmes performants. En général, on peut invoquer des parties en assembleur à partir des programmes en langages évolués.

En plus, le langage assembleur a une utilité pédagogique pour l'étudiant : le fait de le connaître lui permettra d'apprendre et de se familiariser avec la structure interne de l'ordinateur.

2. Traduction des programmes écrits en langage assembleur

Les traducteurs du langage assembleur sont en général des utilitaires qui ne permettent pas la création et la saisie des programmes en assembleur. De ce fait, Le code source en assembleur doit être saisi séparément à l'aide d'un *éditeur de texte* tel que le bloc-notes et sauvegardé sous forme d'un fichier texte avec l'extension `"asm"`.

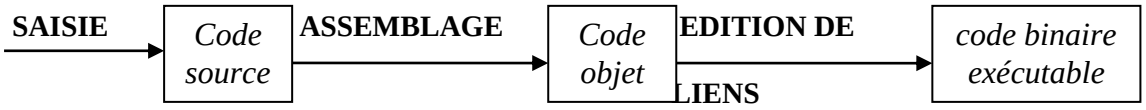
L'opération d'assemblage[1] traduit chaque instruction du programme source (ayant l'extension asm) en une instruction machine. Le résultat de l'assemblage est enregistré dans un fichier avec l'extension `.OBJ` (*fichier objet*). Le fichier objet n'est pas directement exécutable. En effet, il arrive fréquemment que l'on construise un programme exécutable à partir de plusieurs fichiers sources. Il faut ``relier" les fichiers objets à l'aide d'un utilitaire nommé *éditeur de liens*. L'éditeur de liens fabrique un fichier exécutable (généralement avec l'extension `.EXE`), directement exécutable par la machine.

Avant d'être exécuté par le processeur (CPU), le programme doit être chargé en mémoire centrale (M.C). Cette dernière tâche est réalisée par un utilitaire spécial du système d'exploitation qui est le *chargeur*.

---

[1] Dans le contexte du langage assembleur, on dit "*assembler /assemblage" et non pas "compiler/ compilation" comme c'est le cas pour les langages évolués*.

Le processus d'assemblage peut être résumé par le schéma suivant :



SAISIE : La saisie du code source avec un éditeur de texte.
ASSEMBLAGE : Cette opération est réalisée par l'outil assembleur.
EDITION de LIENS : Permet de lier plusieurs codes objets en un seul fichier exécutable.

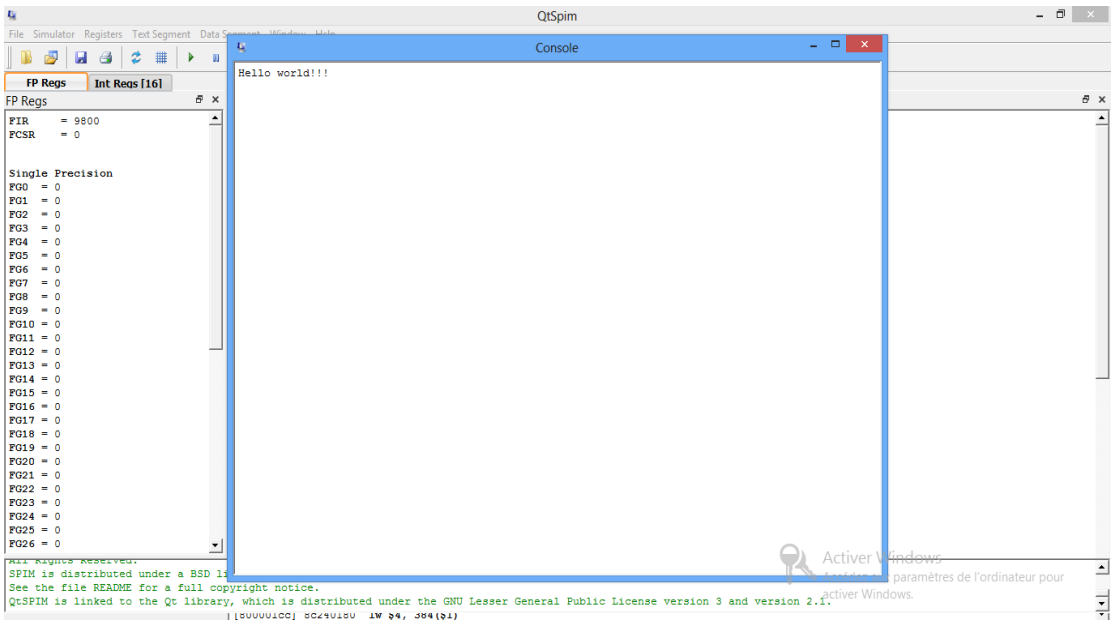
Il est à noter que le langage assembleur est lié étroitement avec le langage machine, qui à son tour dépend du processeur. En d'autres termes, chaque type de processeur a son propre langage machine et par conséquent a son propre langage assembleur. De ce fait, dans le présent manuscrit on s'intéresse à la programmation assembleur dans le cas du processeur MIPS R3000, un processeur 32 bits de type RISC.

3. Lancement d'un programme écrit en assembleur MIPS R3000

Pour les besoins de ce TP on va utiliser l'outil QtSpim (https://sourceforge.net/projects/spimsimulator/files/), un simulateur des machines MIPS 32 bits. Cet outil nous permet de lancer des programmes assembleurs destinés pour ce type de machines. Il fournit également un débogueur simple et un ensemble minimal de services du système d'exploitation.

Pour ce TP, les étapes nécessaires à l'exécution d'un programme assembleur sont comme suit :

- Lancement de l'outil QtSpim.
- Ouvrir le fichier assembleur avec la commande "**File ⇒Reinitialize and load file**"
- Démarrer l'exécution du programme avec la commande "**Simulator ⇒ Run/continue**". Une **fenêtre "console"** s'affiche alors pour donner le résultat de l'exécution du programme:



Il faut noter que l'outil QtSpim ne permet pas la saisie et la création des fichiers en assembleur. Le fichier assembleur doit être créé séparément avec un éditeur de texte tel que le bloc note.