Chapitre IV :

Microprocesseur MIPS 3000

4.1. Introduction

Le **processeur** est le véritable **cerveau** de l'ordinateur. Dans ce chapitre, nous expliquerons son **rôle** ainsi que sa **puissance**, caractérisée par le nombre d'instructions qu'il est capable de traiter par seconde, en calculant le CPI (Cycle Per Instruction). Nous allons étudier la programmation en assembleur d'un microprocesseur. Pour cette partie nous verrons une présentation d'une version légèrement simplifiée de l'architecture externe et interne du processeur **MIPS R3000**, puis nous décrirons le langage d'assemblage du processeur MIPS, ainsi que différentes conventions relatives à l'écriture des programmes en langage d'assemblage.

4.2. Processeur

Aussi appelé CPU (*Central Processing Unit*), c'est un circuit électronique cadencé au rythme d'une horloge interne, grâce à un cristal de quartz qui, soumis à un courant électrique, envoie des impulsions, appelées « **top** ». Toutes les avancées technologiques se concentrent sur ce composant, qui travaille toujours plus vite et effectue des opérations de plus en plus compliquées.

Autrefois agglomérat de circuits physiquement séparés, le microprocesseur est né en 1971 (là encore, le préfixe « micro », à l'origine synonyme de petite taille par rapport aux processeurs sur les gros systèmes, a disparu des dénominations courantes). On est passé de deux mille trois cents transistors (les composants de base des circuits informatiques) pour le premier microprocesseur à plusieurs centaines de millions actuellement.

4.2.1. Rôle du processeur

Le rôle du processeur est d'exécuter les instructions composant le programme. Il se charge de tous les calculs mathématiques et des transferts de données internes et externes. Il décide (en fonction bien sûr des instructions du programme en cours d'exécution) de ce qui se passe à l'intérieur de l'ordinateur, car il permet de manipuler des informations numériques, c'est-à-dire des informations codées sous forme binaire et d'exécuter les instructions stockées en mémoire.

4.2.2. Calcul de CPI (Cycle Per Instruction)

A chaque **top d'horloge**, le processeur exécute une action correspondant à une instruction ou une partie d'instruction. Donc chaque **instruction** nécessite un certain **nombre de cycles** d'horloge pour s'effectuer :

- Le **nombre de cycles** dépend de la **complexité de l'instruction.**
- La **durée d'un cycle** dépend de la fréquence d'**horloge du séquenceur.**

On peut caractériser la **puissance d'un microprocesseur** par le nombre d'instructions qu'il est capable de traiter par seconde. Pour cela, on définit :

- Le **CPI** (*Cycle Par Instruction*) qui représente le **nombre moyen de cycles d'horloge** nécessaire pour **l'exécution d'une instruction** pour un microprocesseur donné.
- Le **MIPS** (*Millions d'Instructions Par Seconde*) qui représente **la puissance de traitement** du microprocesseur.

L'indicateur appelé **CPI** (Cycles Par Instruction) permet de représenter le **nombre moyen de cycles d'horloge** nécessaire à **l'exécution d'une instruction** sur un microprocesseur. La moyenne des cycles par instruction dans un processus donné est définie comme suit :

$$CPI = \frac{\sum_i (IC_i)(CC_i)}{IC}$$

Ou $\mathbf{IC_i}$ est le ***nombre d'instructions*** pour un type d'instruction donné i, $\mathbf{CC_i}$ est les ***cycles d'horloge*** pour ce type d'instruction et $\mathbf{IC=\sum_i(IC_i)}$ est le ***nombre total d'instructions***. La sommation résume tous les types d'instructions pour un processus d'analyse comparative donné.

La **puissance du processeur** peut ainsi être caractérisée par le nombre d'instructions qu'il est capable de traiter par seconde. L'unité utilisée est le MIPS (Millions d'Instructions Par Seconde) correspondant à la fréquence (en MHz) du processeur que divise le CPI et il est défini comme suit :

$$MIPS = \frac{Fréquence}{PCI}$$

Remarque :

Pour augmenter les performances d'un microprocesseur, on peut donc soit augmenter la fréquence d'horloge (limitation matérielle), soit diminuer le **CPI** (choix d'un jeu d'instruction adapté).

Exercice 1 : Pour une processeur MIPS multicycle, il existe cinq types d'instructions :

- Load (5 cycles)
- Store (4 cycles)
- R-type (4 cycles)
- Branch (3 cycles)
- Jump (3 cycles)

Si un programme a :

- 50% load instructions
- 25% store instructions
- 15% R-type instructions
- 8% branch instructions
- 2% jump instructions

Calculer CPI nécessaire pour l'exécution d'une instruction.

Solution :

CPI = (5*50+4*25+4*15+3*8+3*2) / (50+25+15+8+2) = **4.4**
Alors le CPI est égal à 4.4

Exercice 2 : Un processeur à 400 MHz a été utilisé pour exécuter un programme de référence avec le mélange d'instructions et le nombre de cycles d'horloge suivants :

Type instruction	Nombre instruction	Nombre de cycle d'horloge
Arithmétique entière	45000	1
Transfert de données	32000	2
Point flottant	15000	2
Transfert de contrôle	8000	2

Déterminer l'effectif CPI et le taux MIPS pour ce programme

Solution :

CPI = (45000*1+32000*2+15000*2+8000*2) / (45000+32000+15000+8000) =**1.55**

Alors le CPI est égal à 1.55

MIPS = 400/1.55= **258 MIPS**

Alors le MIPS est égal à 258

4.2.3. Processeur CISC et RISC

Actuellement l'architecture des microprocesseurs se compose de deux grandes familles:

- N'implanter en machine que les mécanismes réellement utiles (dont on a statistiquement montré l'utilité), c'est l'approche **RISC** (Reduced Instruction Set Computer).
  Exemple : PowerPC, MIPS, Sparc

- Faire correspondre à chaque structure de données exprimées dans le langage de haut niveau un mode d'adressage adapté dans le langage machine, c'est l'approche **CISC** (Complex Instruction Set Computer)
  Exemple : Pentium

Si dans l'approche **CISC**, la volonté était de séparer la machine matérielle de l'implantation, dans l'approche, c'est la volonté d'une optimisation globale (matérielle et logicielle) qui en est le moteur : on souhaite réaliser des architectures efficaces pour l'exécution des programmes.

a- Caractéristiques :

Leurs caractéristiques sont les suivantes :

RISC	CISC
• Jeu d'instructions de taille limitée • Instructions simples ne prenant qu'un seul cycle • Format des instructions petit et fixé • Modes d'adressage réduits • Décodeur simple (câble) • Beaucoup de registres • Seules les instructions LOAD et STORE ont accès à la mémoire • Compilateur complexe	• Jeu d'instructions de taille importante • Instructions pouvant être complexes prenant plusieurs cycles • Format d'instructions variables (de 1 à 5 mots) • Modes d'adressages complexes. • Décodeur complexe (microcode) • Peu de registres • Toutes les instructions sont susceptibles d'accéder à la mémoire • Compilateur simple

b- Avantages et inconvénients :

Ces deux types ont leurs avantages et leurs inconvénients :

	Avantages	Inconvénients
CISC	+ programmation de plus haut niveau. + programmation plus compacte (écriture plus rapide et plus élégante des applications) + moins d'occupation en mémoire et à l'exécution	- complexifie le processeur - taille des instructions élevée et variable : pas de structure fixe - exécution des instructions : complexe et peu performante.
RISC	+ instructions de format standard + traitement plus efficace + possibilité de pipeline plus efficace	- programmes plus volumineux - compilation plus compliquée