TD n°4.
Cycle d'intstruction, Jeu d'instructions,
Machine à états finis.

Exercice $n°$ 1 **:**
On considère une instruction en double opérandes ayant le format suivant :

4 bits	4 bits	4 bits
opcode	**opérande 1**	**opérande 2**

Supposons que ce format doit représenter 12 instructions en double-opérandes et 30 instructions ayant un opérande (mono-opérande). Calculer le nombre total des instructions sans opérandes qui peuvent être représenter.

Exercice $n°$ 2 **:**
On veut réaliser une machine traitant les données à travers six registres selon 12 différents modes d'adressage. Elle doit être conçu pour executer 10 instructions arithmétiques, 15 instructions logiques, 24 instructions de transferts, 6 instructions de branchement et 5 instructions de contrôle. De ces instructions, respectivement 20%, 60%, 50%, 50% et 60% sont soit des instructions mono-opérande ou sans-opérande soit de type double-opérandes.
Quelle est la taille minimale nécessaire pour définir toutes les instructions de cette machine.

Exercice $n°$ 3 **:**
On considère une machine de Van-Neuman ayant les caractéristiques suivantes :

- le premier cycle machine est toujours composé de 4 états machine,
- les autres cycles machines sont constitués soit de 3 ou 4 états machine,
- un cycle d'instruction se compose au minimum de 2 cycles machines et au maximum de 5 cycles machines,
- la fréquence d'horloge du processeur est 50 MHz.

Calculer le plus court et le plus long temps d'execution d'une instruction sachant que chaque accès mémoire introduit un état machine d'attente.

Exercice $n°$ 4 **:**
A partir d'une instruction de taille 36 bits, concevoir un format extensible d'opcode pour permettre le codage de l'ensemble suivant :

- 7 instructions ayant deux opérandes adresses sur 15-bits et un code de registre sur 3-bit.
- 500 instructions ayant un opérande adresse sur 15-bits et un code de registre sur 3-bit.
- 40 instructions sans opérandes.

Réponse :
Il faut allouer les 3 bits de poids fort pour l'opcode des 7 instructions ayant 3 opérandes. Ces sept instructions utilisent les opcodes de 000 à 110, laissant 33 bits pour les deux adresses et le registre. L'opcode 111 est ensuite associé au premier champ d'adresse de 15 bits pour définir les 500 instructions ayant deux opérandes. On obtient 32768 opcodes disponibles, et on utilisent 500. Les instructions sans-opérandes sont codées à partir de l'opcode principale 111 auquel on associe les 32768 - 500 = 32268 codes inutilisés dans le premier champ d'adresse.

Exercice $n°$ 5 **:**
On considère une machine ayant des instructions sur 16-bit et des adresses sur 6-bits. Certaines

instructions ont une adresse comme opérande et d'autres en ont deux. S'il y a n instructions à deux opérandes adresses, quel est le nombre maximum d'instructions ayant un opérande adresse.

Réponse :
Chaque instruction ayant deux opeérandes adresses utilise jusqu'à $2^{12} = 4096$ combinaisons binaires pour représenter leurs opérandes. Avec n d'entre eux, $4096 * n$ opcodes sont nécessaires. Le nombre de codes d'instruction restants ayant un opérande adresse, est alors $2^{16} - 4096 * n = 65536 - 4096 * n$. Chaque instruction ayant un opérande adresse utilise $2^6 = 64$ combinaisons binaires pour représenter l'opérande. On a donc au maximum $(65536 - 4096 * n)64 = 1024 - 64 * n$ codes qui sont disponibles.

Exercice n° 6 :
Est-il possible de concevoir un format extensible d'opcode pour permettre de coder ce qui suit dans une instruction à 12 bits sachant qu'un registre est codé sur 3 bits.

- 4 instructions avec 3 registres.
- 255 instructions avec 1 registre.
- 16 instructions sans registres.

Réponse :
Non. Pour définir les codes instructions associés aux trois types d'instructions , il faut $4 * 8 * 8 * 8 = 2048$ codes pour la prmier type, $255 * 8 = 2040$ codes pour le deuxième type, et 16 codes pour le troisième type, ce qui fait un totale de 4104 codes. Sur un mot à 12-bit, on ne peut représenter que 4096.

Exercice n° 7 :
On considère une machine ayant 16 registres pour représenter les entiers, dénotés par $R_1, \cdots, R_{15}$ et 16 registres pour représenter les flottants, dénotés par $F_1, \cdots, F_{15}$. Son jeu d'instructions est donné par la table suivante :

Instruction	Détails
`OP` R_i, R_j, R_k	$R_i \leftarrow R_j$`OP`R_k `OP`= `add, sub, mull, div, and, or, xor`
`FOP` F_i, F_j, F_k	$F_i \leftarrow F_j$`FOP`F_k `FOP`= `addf, subf, mullf, divf`
`Mov` R_i, R_j	$R_i \leftarrow R_j$
`Mov` $[R_i], R_j$	$[R_i] \leftarrow R_j$
`Movf` $F_i, [F_j, F_k, 4]$	$F_i \leftarrow [F_j + F_k + 4]$
`Movf` $[F_j, F_k, 4], F_i$	$[F_j + F_k + 4] \leftarrow F_i$
`jeg` R_i, R_j, n	effectue un branchement à l'adresse n plus adresse courante si $R_i == R_j$, sinon continue l'execution. n est une constante entiŕe signée codée sur 5 bits.

Proposer un codage de toutes ces instructions qui minimise le nombre de bits nécessaire.