Université de Bordeaux
Licence STS

ARCHITECTURE DES ORDINATEURS

TD : **09**

Chemin de données d'un processeur (partie 2)

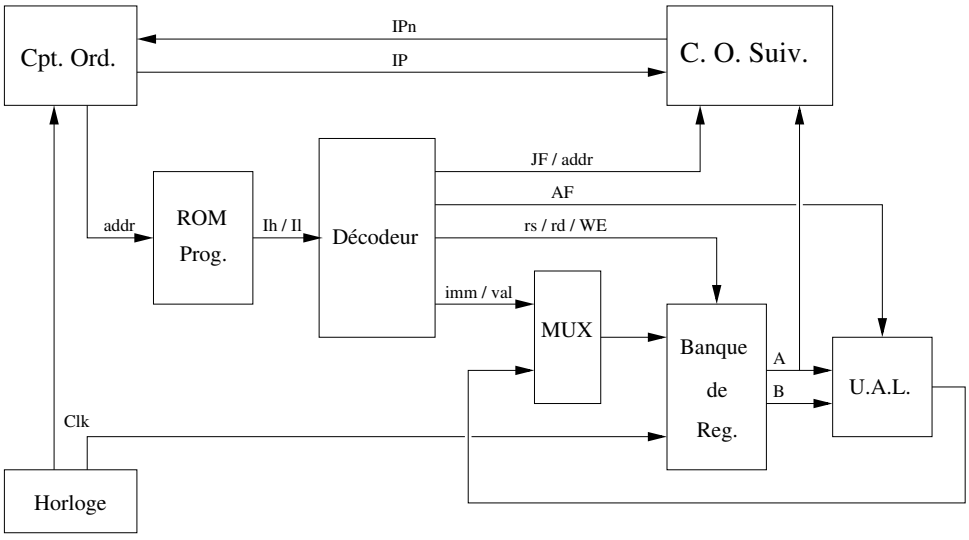
Introduction

On veut construire un processeur sur 4 bits, à l'image du processeur historique « 4004 » d'Intel. Le processeur à construire possède les caractéristiques suivantes :

- Il dispose de 4 registres, numérotés de 0 à 3, c'est-à-dire adressés sur 2 bits ;
- Le compteur ordinal est sur 3 bits, permettant l'existence de $2^3 = 8$ instructions dans la mémoire de programme (oui, c'est peu !) ;
- Chaque instruction est formatée sur un octet, à savoir 2 mots de 4 bits. Le tableau suivant recense les instructions constituant ce langage machine, ainsi que leur codage :

Instruction	7	6	5	4	3	2	1	0
`halt`	0	0	—					
`li` $\texttt{val}_{(4)}$`,`$\texttt{rd}_{(2)}$	0	1	$\texttt{rd}_{(2)}$		$\texttt{val}_{(4)}$			
`add` $\texttt{rs}_{(2)}$`,`$\texttt{rd}_{(2)}$	1	0	0	0	$\texttt{rs}_{(2)}$		$\texttt{rd}_{(2)}$	
`dec` $\texttt{rd}_{(2)}$	1	0	1	1	—		$\texttt{rd}_{(2)}$	
`jmp` $\texttt{addr}_{(3)}$	1	1	—		0	$\texttt{addr}_{(3)}$		
`jz` $\texttt{rs}_{(2)}$`,`$\texttt{addr}_{(3)}$	1	1	$\texttt{rs}_{(2)}$		1	$\texttt{addr}_{(3)}$		

L'architecture du processeur susceptible d'exécuter les instructions de ce langage est la suivante :



Exercice 1 : Passage à l'instruction suivante

En règle générale, la prochaine instruction à exécuter est l'instruction suivante, sauf lorsqu'un branchement doit être pris. Dans le cas du processeur décrit plus haut, il peut s'agir soit d'un branchement

inconditionnel, soit d'un branchement si la valeur du registre source est nulle. Cette information est fournie par le circuit de décodage au moyen de son signal `JF` (« *jump function* »), dont on rappelle ci-dessous les valeurs et leur signfication :

Valeur	Signification
0	Même instruction (cas de `halt`)
1	Passage à l'instruction suivante
2	Branchement inconditionnel
3	Branchement conditionnel

Soient $\texttt{IP}_{(3)}$ (« `IP` », pour « *instruction pointer* ») la valeur courante du compteur ordinal, $\texttt{addr}_{(3)}$ l'adresse de destination du branchement éventuel, $\texttt{val}_{(4)}$ le contenu du registre éventuellement à tester, et $\texttt{JF}_{(2)}$ la fonction à appliquer. Soit $\texttt{IPn}_{(3)}$ la valeur de sortie.

Question 1

Au moyen d'un incrémenteur à quatre bits, câblez la fonction permettant de calculer la valeur $\texttt{IP}'$ correspondant à l'incrémentation de `IP`. Notez que comme les numéros d'instruction vont de 0 à 7 seulement, la valeur suivant 7 doit être 0.

Question 2

Câblez la fonction f_Z valant 1 lorsque la valeur `val` est égale à 0. Complétez-la pour que la fonction résultante f'_Z ne vaille 1 que si également `JF` vaut 3.

Question 3

En vous appuyant sur la fonction précédente, câblez la fonction qui renvoie la valeur du nouveau compteur ordinal `IPn`. On peut voir cette fonction comme un multiplexeur qui, selon la valeur de `JF` (de 0 à 3), renvoie la nouvelle valeur de compteur ordinal correspondante. Le cas de l'instruction `jz` doit alors être traité comme un cas particulier : si `JF` vaut 3 et que `val` vaut zéro, alors la valeur du signal `JF` doit être changée en 2 en amont du multiplexeur, pour déclencher un branchement inconditionnel. Dans tous les autres cas, la valeur du signal restera identique. De fait, la valeur 3 non modifiée doit conduire, au niveau du multiplexeur, à ce que l'on passe à l'instruction suivante (branchement conditionnel non pris).

Exercice 2 : Banque de registres

Une banque de registres est une catégorie de mémoire dite « *multi-ports* ». En effet, au cours d'une instruction, on doit pouvoir être capable de lire en même temps deux registres à la fois, identifiés par les signaux d'entrée $\texttt{r0}_{(2)}$ et $\texttt{r1}_{(2)}$, et éventuellement d'écrire dans un registre identifié par le signal $\texttt{rw}_{(2)}$ (pas au moment exact où on lira, mais un peu plus tard).

Question 1

Câblez les deux sorties associées aux signaux $\texttt{r0}_{(2)}$ et $\texttt{r1}_{(2)}$.

Question 2

Câblez le dispositif d'écriture associé à l'entrée $\texttt{rw}_{(2)}$ et au signal `WE` (pour « *write enable* »).

Exercice 3 : Mémoire ROM

Une mémoire ROM peut être vue comme une fonction logique qui, en fonction de la valeur d'une adresse (donnée d'entrée), renvoie une valeur fixe, codée « en dur ». Chaque bit du mot renvoyé en sortie du circuit mémoire peut donc être vu comme une fonction logique dépendant des variables codant chacun des bits de l'adresse. Plus simplement, on peut considérer une mémoire ROM comme un grand multiplexeur qui, pour chacune des valeurs d'adresse, renvoie une donnée fixe correspondant à l'une des voies d'entrée du multiplexeur.

Question 1

À base de multiplexeurs, construisez une mémoire ROM à quatre mots de quatre bits.

Question 2

Étendez cette mémoire pour en faire une mémoire ROM à huit mots de quatre bits.