TD10 : Contrôle de l'exécution des instructions

Dans ce TD, on utilise le jeu d'instruction MIPS32

1. Version 1 : structure 3 bus

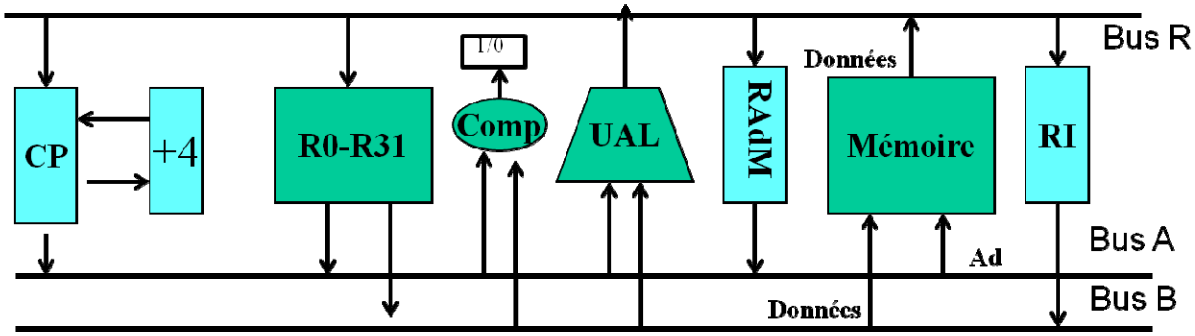

Figure 1 : Chemin de données (version 1).

Pour le chemin de données de la Figure 1, on veut réaliser le contrôle de l'exécution d'un certain nombre d'instructions du processeur NIOS.

- ADD $rk,$ri, $rj
- OR $rk,$ri,$rj
- LW $rk, ($ri + IMM16)
- BEQ $ri,$rj, IMM16
- JR $ri
- JALR $ri

Définir l'ensemble des commandes pour contrôler l'exécution des instructions
Remplir la table donnant les actions à exécuter à chaque cycle d'horloge pour exécuter les instructions.

INST	OP	FONCTION	Barrière BUS	MEM	UAL	Comp	BUS R	Mux CP	Ecriture registre

Remarques

1. Pour les commandes TARi et TBRj, les numéros de registres source 1 et 2 (contenus dans RI) servent d'adresse pour la « mémoire » de 32 registres et les barrières de bus contrôlent l'accès du contenu du registre sélectionné vers le bus
2. Pour les commandes WRk, le numéro de registre destination sert d'adresse pour la mémoire de 32 registres et WR sert de commande d'écriture.

Définir la partie Etat futur = fonction (Etat présent, Entrées) de l'automate de contrôle
Donner les sorties de l'automate en fonction des entrées et de l'état présent.

2. Version 2

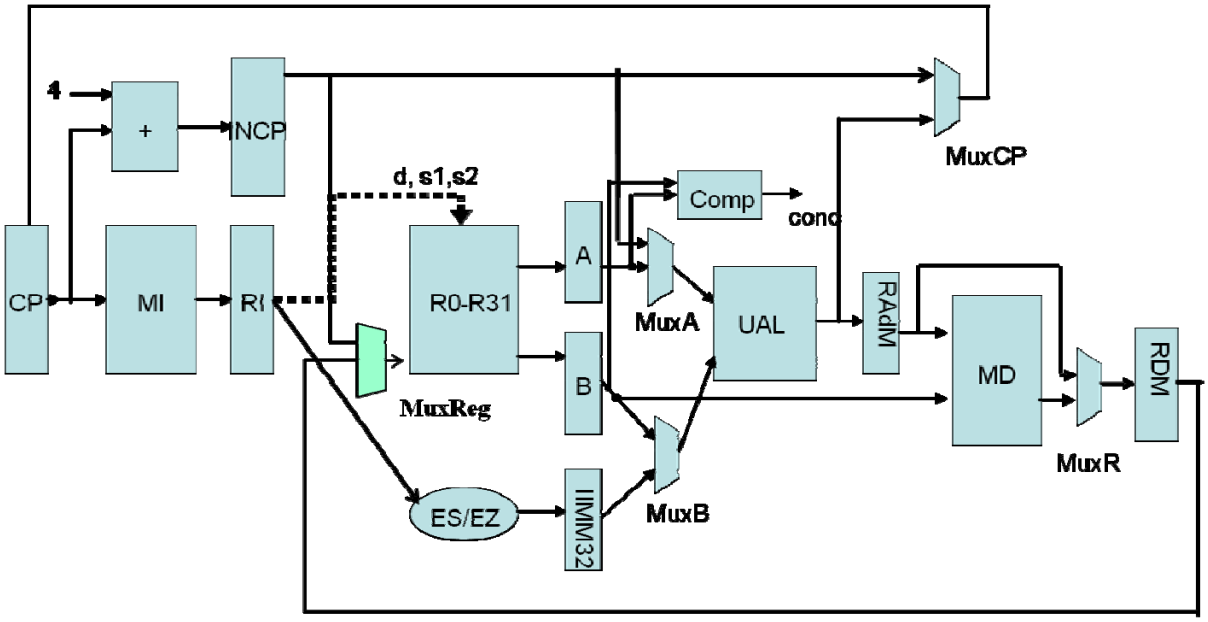

Figure 2 : chemin de données (version 2)

Quels sont les différentes étapes pour l'exécution des instructions suivantes

- ADD $rk,$ri, $rj
- OR $rk,$ri,$rj
- LW $rk, ($ri + IMM16)
- BEQ $ri,$rj,IMM16
- JR $ri
- JALR $ri

Définir la partie Etat futur = fonction (Etat présent, Entrées) de l'automate de contrôle
Définir l'ensemble des commandes pour contrôler l'exécution des instructions

3. Codes opération et fonctions des instructions utilisées

INST	OP RI $_{31-26}$	Fonction RI $_{5-0}$
ADD	00	32
OR	00	37
LW	23	
BEQ	04	
JR	00	08
JALR	00	09