BEAL Clara
MORIN Alice
VALEMBOIS Auxane

RAPPORT

Projet Processeur

1. Composant Registres

Le registre est un composant de mémoire interne du processeur. C'est une bascule D (délai) qui retient une valeur sur tout un cycle d'horloge. C'est la mémoire la plus rapide mais aussi la plus chère à fabriquer.
Ici, les registres ont une capacité de 16 bits.
A l'aide d'un démultiplexeur on peut choisir l'entrée du registre à modifier ou accéder à sa valeur pour être utilisée par le composant ALU.

2. Composant ALU

L'ALU ou "Arithmetic & Logic Unit" est le composant permettant à l'ordinateur de mener à bien des calculs. Il renvoie le résultat des opérations sur 16 bits.
Dans notre projet nous avons rajouté comme huitième opération une multiplication. Cela semblait pour nous le choix le plus logique à faire, cela nous permet d'effectuer les multiplications et les divisions beaucoup plus rapidement.

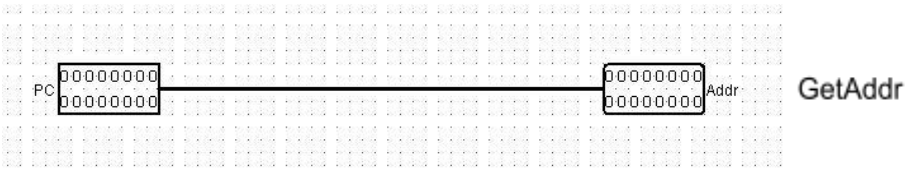
```
ADD R1 R0 3    * R1=3
ADD R2 R1 R1   * R2=6
SUB R4 R0 -7   * R4=7
SUB R3 R4 R2   * R3=1
OR R5 R1 6     * R5=7
OR R6 R1 R2    * R6=7
AND R5 R4 2    * R5=2
AND R6 R4 R1   * R6=3
XOR R5 R2 3    * R5=5
XOR R6 R2 R1   * R6=5
SL R5 R3 2     * R5=4
SL R6 R3 R1    * R6=8
SR R5 R5 2     * R5=1
SR R6 R6 R1    * R6=1
MUL R5 R1 4    * R5=12
MUL R6 R1 R4   * R6=21
```

Avec ce code assembleur nous pouvons vérifier la fonctionnalité de nos composants registres et ALU.

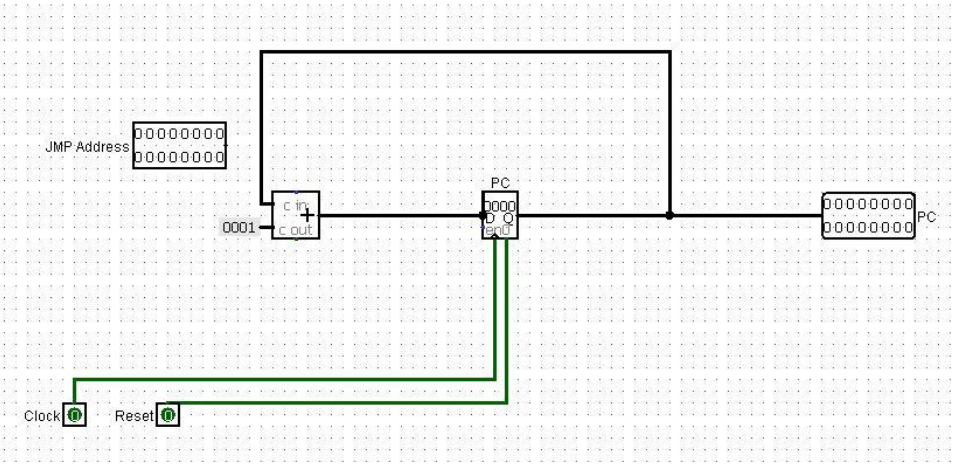
3. Decode IR

A cette étape de la conception, le processeur fonctionne sans les instructions de Mémoire.
L'implémentation de RegPC et GetAddr seront différentes lorsque nous ajouterons ces instructions au processeur.

Ci dessous l'implémentation de RegPC et GetAddr sans les instructions de la mémoire :



BEAL Clara
MORIN Alice
VALEMBOIS Auxane



RegPC

4. Instructions MEM

Une fois les instructions de mémoires implémentées RegPC et GetAddr servent respectivement à :

- pointer l'adresse de la RAM qui correspond à l'instruction que l'on veut exécuter.
- donner à la RAM comme adresse un deuxième argument ou le RegPC.

```
SUB R7 R7 R7    * R7=0
ADD R7 R0 64    * R7=64
SUB R0 R0 R0    * R0=0
ADD R0 R0 2     * R0=2
STR R7 R0       * mem[64]=2
LD  R1 R7       * R0=2 R1=mem[64]=2 R7=64
```

Nous avons vérifié le bon fonctionnement de la mémoire avec ce code assembleur ci-contre.

5. Sauts

Les instructions de sauts permettent de déplacer le choix de la prochaine instruction à exécuter.
Quand il n'y a pas de saut, le RegPC s'incrémente à chaque fin d'instruction jusqu'à son arrêt. Pour réaliser le saut dans RegPC, on rajoute un multiplexeur qui va soit choisir l'adresse de la prochaine instruction, soit l'instruction suivante.

```
       SUB  R0 R0 R0     * R0=0
       SUB  R1 R1 R1     * R1=0
       ADD  R1 R1 2      * R1=2
l0: JEQU R0 R1 b0:  * (R0==2)? b0:
       ADD  R0 R0 1      * R0++
       JMP  l0:        *
b0:  SUB  R0 R0 R0     * R0=0
l1:  ADD  R0 R0 1      * R0++
       JNEQ R0 R1 l1: * (R0!=2)? l1:
       SUB  R0 R0 R0     * R0=0
       ADD  R0 R0 4      * R0=4
l2: SUB  R0 R0 1      * R0--
       JSUP R0 R1 l2: * (R0>2)? l2:
       SUB  R0 R0 R0     * R0=0
l3: ADD  R0 R0 1      * R0++
       JINF R0 R1 l3: * (R0<2)? l3:
```

De plus, nous avons ajouté un composant JmpCond qui nous permet d'effectuer les instructions JEQU, JNEQ, JSUP et JINF. Avec un multiplexeur qui choisit l'opération, il admet en sortie un booléen qui nous informe sur la véracité de la condition.
Ci-contre le code assembleur pour tester ce nouveau composant.

BEAL Clara
MORIN Alice
VALEMBOIS Auxane

6. Appel de fonction

Nous avons ajouté un nouveau composant PCheap pour les instructions CALL et RET.
Lors des appels successifs d'instructions CALL et RET il y a une pile de mémoire pour le retour de ces appels et un registre qui correspond au haut de la pile.
Quand on utilise CALL l'adresse d'entrée est ajoutée à la mémoire et on incrémente le registre.
Quand on utilise RET on décrémente le registre et on rend la mémoire de celui-ci disponible à la sortie. Pour cela on utilise l'opération ADD avec les valeurs 1 ou -1.
Nous avons utilisé un multiplexeur en entrée de la sélection de l'adresse de la mémoire pour éviter de décrémenter le registre puis de sélectionner la mémoire, ici, pour les instructions RET la valeur du registre sera de -1, et pour le reste cela restera sa valeur authentique.