

INF1600

Architecture des micro-ordinateurs

Laboratoire 2

Soumis par:

Nguyen Nicolas- 2031636

Do Minh-Tri – 2030231

Groupe de laboratoire 05

Le 25 février 2020

Exercice 1

a)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| RTN concret | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Hexa |
| MA <- PC; | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0x3060 |
| MD <- M[MA] :  PC <- PC + 4; | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0x6CC0 |
| IR <- MD; | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0x82E0 |

b)

(IR<31..27>) = opcode ) -> R[IR<26..22>]<- R[R<21..17>] oper M[R[IR<16..12>] + IR<11..0>];

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| RTN concret | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Hexa |
| A <- R[rc]; | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0x006E |
| MA <- A + IR<11..0>; | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0x1021 |
| MD <- M[MA] :  A <- R[rb]; | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0x0CEA |
| R[ra] <- A oper MD; | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0x8210 |

c)

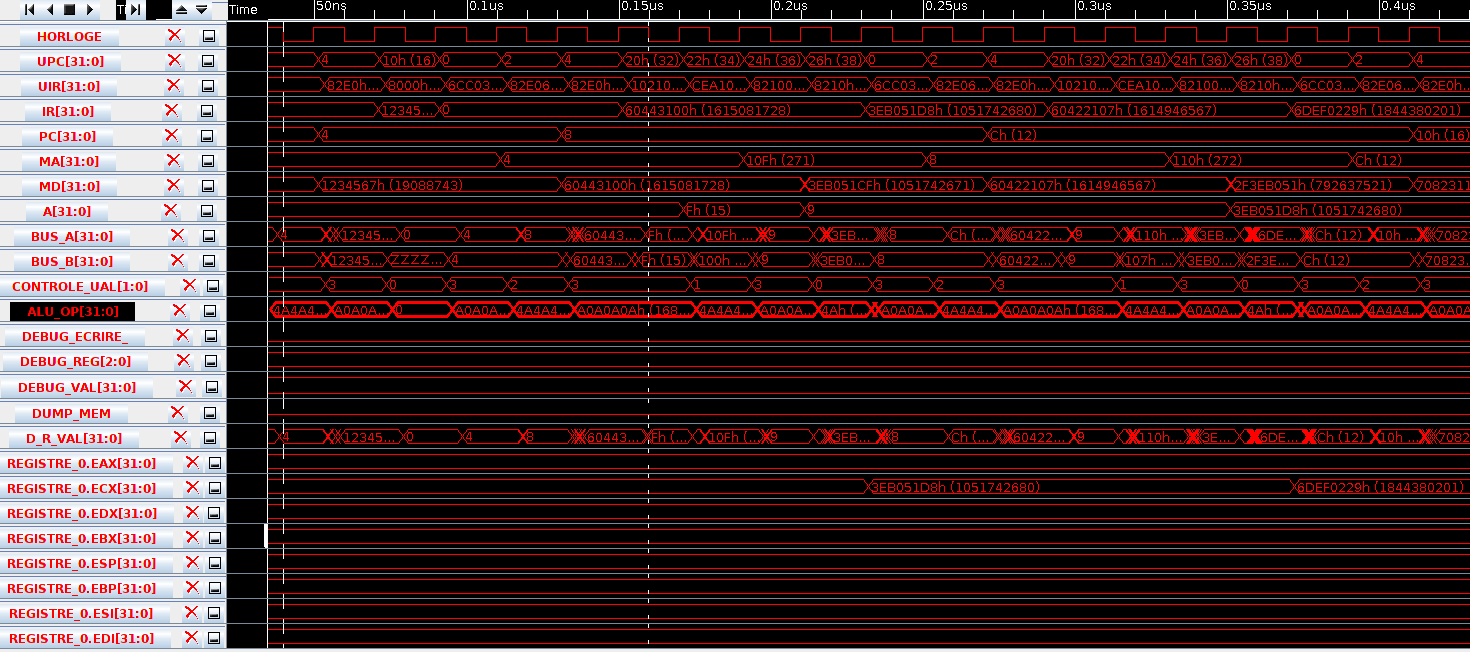


Figure 1 : Simulation du no. 1 et 2

L’instruction pour "add" (opcode 0xc) : R[1] <= R[2] add M[R[3] + 0x100] est affiché après 4 cycle, et celle du "add" (opcode 0xc) : R[1] <= R[1] add M[R[2] + 0x107] est affiché 7 cycles après la première. On a les 4 cycles pendant laquelle l’instruction du tp2mem.txt est effectué, puis 3 cycles pour passer à la prochaine instruction, puis 4 cycles de la prochaine instruction, et ainsi de suite. Étant donné que les instructions affichent leurs résultats au 4e cycle, alors les deux réponses sont séparées de 7 cycles d’horloge.

d)

opcode du NAND : 0x07, car [3..0], 0111, est la table de vérité d’un NAND.

e)

a.

01010 10101 01010 10101 010101010101

Dans l’opcode, les deux derniers octets, [IR<15..0>], représentent les 4 derniers bit da l’adresse du registre rc [IR<16..12>] et la valeur d’une constante stocké dans [IR<11..0>].

Si on échange rb et rc, on a une instruction à peu près équivalente, si l’opcode correspond à une opération commutative :

01010 10101 10101 01010 010101010101

0x556AAEEE

b.

Un avantage est de pouvoir faire deux opérations en même temps. Notamment, dans le no 2, on accède à la mémoire en même temps (même cycle) qu’on effectue une opération d’addition.

MD <- M[MA] : PC <- PC + 4;

c.

Dans la première architecture, on pouvait lire des registres ra, rb et rc alors que dans la deuxième, on ne peut pas lire de ra. La première avait aussi deux constantes, donc étant donné que la première architecture a plus de variables qu’on peut utiliser que la deuxième, la deuxième est moins flexible que la première.