INF1600

Architecture des Micro-Ordinateurs

Laboratoire #2 : Architecture du processeur

Soumis par :

Nathan, Ramsay-Veljens (1989944)

Cassy, Charles (1947025)

Section de labo #4

1er Mars 2020

Exercice 1: Architecture avec microcodes

Questions

a) Recherche d'instruction :

Donnez la séquence de microcodes qui correspond à la recherche d'une instruction.

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	0	1	1	0	0	0	0	0	0x8260

b) Exécution d'une instruction générique :

Écrivez la séquence de microcodes permettant d'exécuter l'instruction d'opérations arithmétiques/ logiques typiques décrite par le RTN abstrait:

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

RTN concret

A<-R[IR<16..12>];

MA <- A +IR<11..0>;

MD <- M[MA] :A<-R[IR<21..17>];

R[IR<26..22>]<-A oper MD

RTN concret	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	hexa
A<-R[IR<16..12>];	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[IR<21..17>];	0	0	0	0	1	1	0	0	1	1	1	0	1	0	1	0	0x0CEA
R[IR<26..22>]<- A oper MD	1	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0x8210

c) Simulation :

Prenez une capture d'écran et intégrez-la dans votre rapport.

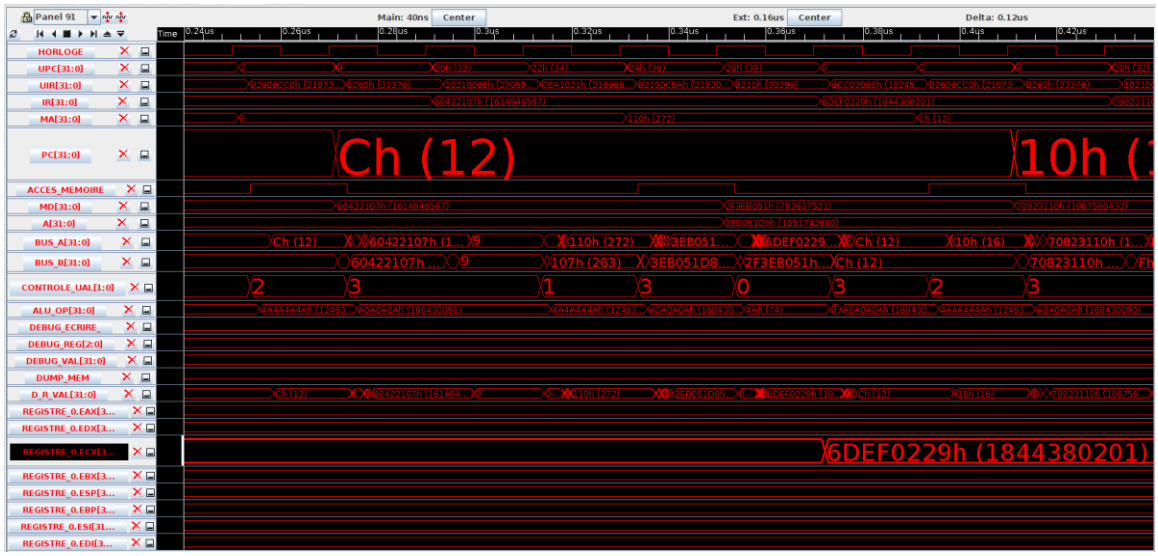

Figure 1. Simulation (Capture 1)

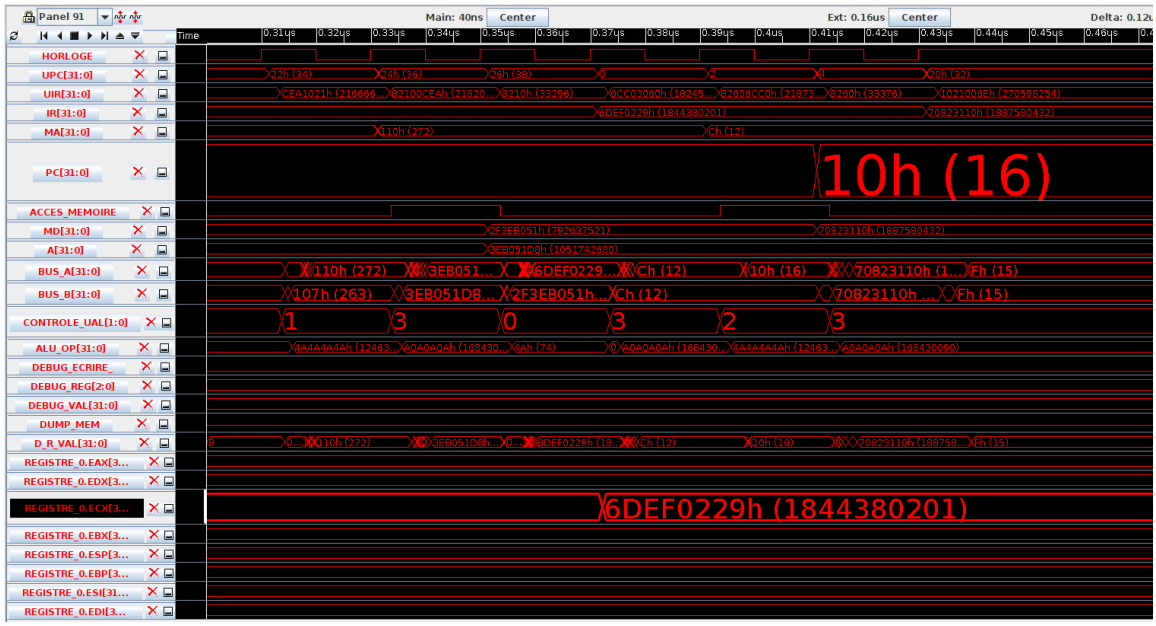

Figure 2. Simulation (Capture 2)

d) L'opération NAND :

Figure 3 : Table de vérité NAND

A	B	F
0	0	1
0	1	1
1	0	1
1	1	0

Grâce à la navigation dans Electric, il est possible de voir que les quatre bits de poids faible du opcode régissent la table de vérité de l'équation bit à bit. On sait donc que les trois bits de poids faible sont fixés à 1 et le 4^{ième} à 0. Pour ce qui est des autres bits, on ne veut pas faire le add32 alors on doit mettre le 7^{ième} bit à 0 afin que le bus ne sortie du and32 soit toujours à zéro. Pour le 6^{ième} bit, on le fixe à 0 afin de ne pas avoir de retenue et ne pas modifié le résultat. Finalement, on veut que le MUX choisisse l'entrée du haut donc le 5^{ième} bit sera fixé à 0. Cela nous donne donc 00 0111, c'est-à-dire **0x07**.

Ecrivez cette valeur à l'endroit approprié du fichier tp20pa1u. txt et relancez la simulation (redémarrez Electric pour être certain) afin de tester l'instruction NAND de l'adresse Oxc dans tp2mem. txt.

Donnez aussi cette valeur dans votre rapport.

e) Compréhension :

a. 0x555 5555=0b0101 0101 0101 0101 0101 0101 0101

Les 2 derniers octets servent à donner la constante dans IR<11..0>(1365 dans ce cas) ainsi que les 4 bits les moins significatifs de IR<16..12>. Nous pouvons donc modifier les 3 dernières valeurs de 0x555 5555 sans affecter quoi que ce soit puisque la constante n'est pas utilisée dans ce cas. Un exemple serait 0x555 5999.

b. *Nommez un avantage d'avoir une architecture à deux bus. Vous êtes -vous servi de cet avantage dans votre microprogramme développé en 2. ?*

L'avantage d'avoir 2 bus est la possibilité de transmettre des données sur les deux bus et l'exécution de davantage d'opérations dans un même cycle d'horloge. Par exemple, l'écriture dans un registre en même temps de lire dans MD comme c'est le cas lorsque nous avons réalisé l'opération suivante : R[IR<26..22>]<-A oper MD.

c. *Diriez-vous que les instructions de cette architecture peuvent être aussi flexibles, en terme d'opérations arithmétiques logiques, que celles du processeur étudié à l'exercice 4 du TPI ?*

Pourquoi ?

Les instructions de cette architecture semblent moins flexibles, en termes d'opérations arithmétiques logiques, que celle du processeur étudié à l'exercice 4 du TP1. Dans l'architecture du TP1, l'instruction possède deux emplacements pour les constantes, ce qui permet de faire des opérations plus diverses dans une même instruction.

Exercice 2: Assembleur avec processeur à pile

Voir fichier tp2_2.s

Exercice 3: Conditions et branchements

Voir fichier tp2_3.s