

TP 6

Architecture des systèmes informatiques – Info3 S5

Circuits logiques et processeur

Objectif du TP : Appliquer les principes vus en cours pour la synthèse de circuits logique, et découvrir le fonctionnement interne d'un processeur simplifié.






1 Installation du simulateur de circuit logique : logisim

Suivez les instructions données à <https://sourceforge.net/projects/logisim-ps/> afin de télécharger le logiciel logisim. Il s'agit d'un fichier exécutable windows que vous pourrez lancer depuis votre environnement de travail windows. Documentation et tutoriels sont disponibles ici : <http://www.cburch.com/logisim/docs/2.7/en/html/guide/>


2 Premier circuit : le 1/2 additionneur binaire


Reprenez les transparents vus en cours pour la synthèse du 1/2 additionneur binaire, et synthétisez-le sous logisim.

Vous aurez besoin pour cela des composants suivants :

-  Pin : pour les 2 entrées A et B
-  Pin : pour les 2 sorties S et R
- AND  et OR  et NOT  Gates

La première chose à faire est de créer un projet que vous nommerez « demi-additionneur »

Ensuite vous composerez le circuit en *mode édition* (curseur flèche ) qui permet d'ajouter des composants, de les paramétrer (*properties*), de relier les broches (*pins*) par des connexions (*wires*), et de nommer les entrées/sorties A, B, S, R.

Puis, pour l'exécuter, vous devrez passer en *mode simulation* (curseur main ). En cliquant sur les entrées pin vous pourrez en changer la valeur (0,1,x) et voir en direct la réponse du circuit.

Dans le menu « *Projet/Analyze Circuit* » vous pourrez analyser les réponses du circuit et ses tables de vérité.

3 Deuxième circuit : l'additionneur binaire

Toujours en reprenant les éléments vus en cours, composer un additionneur binaire à partir du 1/2 additionneur binaire précédent.

Pour cela vous créerez un nouveau projet intitulé « additionneur-binaire »

4 L'additionneur parallèle 4 bits

A l'aide de circuits « additionneur-binaire » composer un additionneur parallèle 4 bits :

$$A_3A_2A_1A_0 + B_3B_2B_1B_0 = S_4S_3S_2S_1S_0$$

Ai et Bi seront les entrées, Si les sorties

5 Mémoire RSC et registre à décalage

Reprenez les éléments du cours pour composer une mémoire RSC 1 bit, puis un registre 4 bits, et enfin un registre à décalage 4 bit.

6 Simulateur de Processeur : Johnny Simulator

Installez le simulateur de processeur Johnny Simulateur en téléchargeant son archive zip à partir de : <http://sourceforge.net/projects/johnnysimulator/> .

Lisez la section 4 « The Processor » dans le fichier de documentation « Johnny-Manual-EN.pdf », qui vous expliquera les différents composants graphiques du processeur (bus, ram, UC, ALU, registres). Dans un premier temps laissez l'UC masquée.

Premier programme : addition

Voici votre premier programme à écrire dans la RAM :

```
001: TAKE 010
002: ADD 011
003: SAVE 012
004: HLT 000
```

Ce programme est composé d'instructions en langage machines (assembleur), lui-même traduit en nombres entiers (code machine) ici encodés en octal (Hi Lo). Adr correspond à l'adresse en mémoire. Le processeur est très simple car il ne comporte que 10 instructions : TAKE SAVE ADD SUB INC DEC NULL TST JMP HLT. Ces instructions machines sont décrites dans la section 5 du manuel.

Une fois ce programme entré en RAM, commencer son exécution en mode pas à pas (One Macro Step) et observez à chaque pas l'effet de l'instruction sur les bus adresse et données, et sur l'accumulateur.

Faites évoluer ce programme pour réaliser une multiplication par 2.

Le micro-code de l'UC

Chaque instruction est interprétée par l'UC qui est chargée de réaliser l'opération demandée. En cliquant sur « Ctrl » (rouages), vous ferez apparaître les composants internes de l'UC. Les détails de l'UC sont décrits en section 4.3 du manuel. On y trouve principalement une mémoire interne (ROM) contenant le microcode de chaque instruction du processeur. Les instructions du microcode sont rudimentaires et consistent à manipuler les composants internes de l'UC : db->ins, ins->ab, ins->pc, pc->ab, pc++, =0:pc++, ins->mc, mc:=0, stopp, plus, minus. ab est le bus d'adresse, db le bus de donnée, mc est pointeur de microcode (adresse mémoire microcode), pc est le pointeur d'instruction (adresse RAM), ins le registre d'instruction en cours de décodage.

Répétez l'exécution du programme addition, mais cette fois-ci en faisant du pas à pas sur le microcode. Vous verrez ainsi comment chaque instruction est interprétée et exécutée par le microcode au sein de l'UC. Détaillez le contenu de la mémoire de microcode et comment chaque instruction est implémentée.

Proposez d'étendre le jeu d'instruction en ajoutant le microcode d'une nouvelle instruction, MUL2, qui fera un décalage binaire à gauche (x2) sur l'accumulateur.

Suite : répéter cette simulation sur le programme « multiplication ».