Architecture des ordinateurs

TP1 : circuits combinatoires

C. Alias

Le but de ces TPs est de construire un petit processeur avec le logiciel de simulation logique Diglog. Dans ce premier TP, nous allons construire les circuits combinatoires de base ainsi que l'unité arithmétique et logique.

Exercice 0. *Prise en main de* Diglog

Ajouter le répertoire `/home/tpedu/INF2003L/diglog/bin` à votre variable d'environnement `PATH`. Ouvrir un terminal et entrer `diglog`.

- Diglog est composé de deux fenêtres appelées *mylib* et *mycrt*. *mylib* permet de construire le circuit et *mycrt* est une fenêtre annexe utilisée pour le dialogue (chargement/sauvegarde, etc). Diglog vous permet de dessiner des circuits de manière intuitive à l'aide d'une interface graphique.

- Au bas de la fenêtre *mylib*, il y a une zone contenant quelques portes logiques et des points. Cette barre vous permet de stocker les portes logiques que vous utilisez le plus souvent. Pour placer une porte logique dans cette barre, cliquez simplement sur une porte et amenez-la à l'endroit desiré. Si vous placez une porte sur une porte déjà existante, elle la remplace.

- A gauche et à droite se trouvent des menus, mais la plupart des commandes peuvent être lancées à partir du clavier, ce qui est souvent plus rapide.

- Pour **placer une porte logique**, cliquez sur la porte dans la barre du bas, et faites glisser à l'endroit désiré.

- Pour **tracer un fil**, utilisez la souris comme dans un logiciel de dessin. En cliquant sur le bouton gauche vous commencez un fil, vous marquez un angle ou la fin du fil en cliquant à nouveau et quand vous avez terminé, vous cliquez sur le bouton droit.

- Pour **imposer une valeur sur un fil**, on peut utiliser un *générateur*. C'est le composant en bas à gauche de la barre. En cliquant au centre du générateur, on fait passer sa valeur de 0 à 1.

- Pour **visualiser la valeur sur un fil**, on peut utiliser une *sonde*. C'est le composant carré à droite du générateur. On peut également passer en *glow mode* en appuyant sur la touche g. On peut sortir le glow-mode en appuyant à nouveau sur la touche g.

- Pour **utiliser d'autres portes logiques**, on peut utiliser le *catalogue*. Pour çà, cliquer sur CAT en bas à gauche et cliquer sur le composant voulu. Si çà ne suffit pas, on peut chercher dans la bibliothèque de composants en cliquant sur LIBR. Une liste de composants s'affiche alors dans la fenêtre *newcrt*. On peut passer à la catégorie suivant en appuyant sur la touche + et revenir à la catégorie précédente en appuyant sur la touche -. Pour choisir un composant, il suffit de cliquer dessus. Pour quitter, appuyer sur la touche espace.

- Diglog se compose d'**onglets**, accessible en appuyant sur les touches 1, 2, 3, etc.

- Pour **sauvegarder l'onglet courant dans un fichier**, faites Maj+S. Entrez le nom du fichier dans la fenêtre *newcrt*.

- Pour **charger un fichier dans l'onglet courant**, faites Maj+L. Entrez le nom du fichier dans la fenêtre *newcrt*.

- Pour **quitter** Diglog, faites Maj+Q. Confirmez ensuite dans la fenêtre *newcrt*.

Question 1.

- Allez dans CAT, et sélectionnez le clavier (KEYPAD) et l'afficheur hexadécimal (7SEG).
- Reliez-les par des fils. Passez en *glow-mode* en expérimentez.
- Construisez un circuit qui réalise le ET de 4 bits en entrée par un bit de contrôle. A quoi peut servir ce circuit ?
- Placez le entre le clavier et l'afficheur, et testez.
- Sauvegardez dans le fichier `tp1.lgf`.

Question 2.

Encapsulez votre circuit dans un composant. Pour cela:

- Cliquez sur CAT et et choisissez un circuit d'instance INSTn en fonction du nombre de pattes dont vous avez besoin (le nombre d'entrées et de sorties).
- Connectez les entrées et les sorties de votre circuit aux points de connexion du circuit INSTn. Par exemple, connectez les entrées sur le coté gauche du circuit d'instance, et les sorties à droite. DIGLOG sait calculer si les connexions sont des entrées ou des sorties en fonction de votre circuit.
- Donnez un nom à votre circuit. Pour cela, appuyez sur la touche ℓ pour créer un label et entrez le nom de votre circuit entre guillemets "AND8". Faites glisser le label dans le composant d'instance INSTn.
- Passez en mode configuration en cliquant en bas à droite de *mylib* jusqu'à ce que CNFG apparaisse.
- Cliquez dans le composant d'instance INSTn. Dans la fenêtre *newcrt*, entrez "AND8" (avec les guillemets) dans *Instance of*. Appuyez sur la flèche du bas et entrez y. Appuyez sur la touche espace.
- Dans le menu Frills, cliquez sur Box. Tracez une boite autour de votre circuit.
- Enfin, ajoutez des labels pour commenter.

Question 3.

Créez une instance de votre circuit. Pour cela:

- Choisissez un circuit d'instance INSTn identique.
- Passez en mode configure, et cliquez dans INSTn. Dans *newcrt*, entrez "AND8" (entre guillemets). Appuyez sur la flèche du bas, puis y, puis espace.

Question 4.

Utilisez votre composant AND8 entre le clavier et l'afficheur. Testez.

Exercice 1. *Circuits combinatoires importants*

On étudie plusieurs circuits élémentaires fréquemment utilisés dans les composants des processeurs. Chacun de ces circuits sera encapsulé dans un composant.

- Construire un décodeur 3 bits vers 8 bits
- Construire un multiplexeur 2 bits vers 1 bit.
- Construire un multiplexeur 3 bits vers 1 bit.
- Construire un multiplexeur 8 bits vers 4 bits.

Exercice 2. *Unité arithmétique et logique*

Il s'agit ici de construire une unité arithmétique et logique. Là encore, chaque circuit sera encapsulé dans un composant. Autant que possible, on réutilisera les composants déjà définis.

- Construire un **additionneur 1 bit** avec retenue entrante.
- Construire un **additionneur 4 bits**, puis **un additionneur 8 bits**.
- Construire un **additionneur/soustracteur 8 bits**, c'est à dire un circuit qui prend deux entrées de 8 bits et un signal qui vaut 0 s'il faut les additionner ou bien 1 s'il faut les soustraire. Le résultat est un mot de 8 bits et une retenue. On utilisera le complément à deux 8 bits pour la soustraction.
- Construire un **opérateur d'extension signée 8 bits vers 16 bits**.
- En utilisant les composants précédement définis, construire une **ALU 8 bits** capable de faire une addition, une soustraction et un test d'égalité. L'opération sera choisie avec un signal qui vaut 00 pour une addition, 01 pour une soustraction et 10 pour un test d'égalité.

Exercice 3. *Addition avec retenue anticipée*

L'inconvenient de l'additionneur 8 bits en cascade est que chaque additionneur 1 bit doit attendre que sa retenue soit disponible pour pouvoir réellement commencer son addition. Par conséquent, son temps de traversée est 8 fois celui d'un additionneur.

L'idée de l'additionneur avec retenue anticipée (*carry-select*) est de diviser le calcul avec plusieurs blocs de 2 additionneurs classiques, le premier avec une retenue entrante de 0 et le deuxième avec une retenue entrante de 1. On sélectionne ensuite le bon résultat (avec des multiplexeurs) quand la retenue entrante est connue. Comme les blocs peuvent être traversés en même temps, le temps de traversée s'en trouve réduit.

Question 1. Sur papier, construire un additionneur avec retenue anticipée 8 bits. On utilisera deux blocs de 4 bits.

Question 2. Le temps de traversée de l'additionneur 8 bit est 8 fois celui de l'additionneur 1 bit. Quel est le temps de traversé de votre additionneur avec retenue anticipée ?

Question 3. Implémentez votre additionneur avec retenue anticipée dans DIGLOG et testez.

Exercice 4. *Addition avec conservation de retenue*

Question 1. Pour la somme de 3 entiers sur 4 bits, proposez un circuit dont le temps de traversée vaut 6 fois celui de l'additionneur 1 bit.

Question 2. Pour la somme de 3 entiers sur 4 bits, proposez un circuit dont le temps de traversée ne vaut que 5 fois celui de l'additionneur 1 bit. *Indication : utiliser l'associativité de l'addition pour retarder la propagation des retenues jusqu'a la dernière étape du calcul. Le circuit obtenu est appelé additionneur avec conservation de retenue* (carry-save).

Question 3. Implémentez votre additionneur avec conservation de retenue dans DIGLOG et testez.