N7_SN_1A

Architecture des ordinateurs - Semestre 5

BE-TP4 - Circuits séquentiels

L'objectif est d'implanter l'algorithme de tri par sélection dans un circuit logique, sur la base de l'algorithme suivant :

Pour I de **1 à N-1** Faire
    Calculer Max et Indice_Max de Tab (I...N)
    Echanger Tab(I) et Tab(Indice_Max)
Fin Pour

La première étape consiste à transformer cet algorithme pour l'adapter à une implantation de bas niveau : remplacer le Pour par un Tantque et les indices de tableau par des adresses (le tableau se trouve stocké entre l'adresse Ad1, adresse du premier élément, et Ad2, adresse du dernier élément. Pour simplifier l'implantation (moins d'états), on utilisera un Répéter à la place du Tant que en supposant que le tableau contient au moins deux élément (vérifiable avec Ad2 > Ad1. On aura ainsi :

Ad_courante ← Ad1
Répéter
    Calculer Max et Ad_Max de Tab (Ad_Courante … Ad2)
    Echanger Tab (Ad_Courante) et Tab (Ad_Max)
    Ad_Courante ← Ad_Courante + 1
Jusqu'à Ad_Courante >= Ad2

On adoptera la même démarche qu'en TD3/TP3, et on commence par élaborer un graphe d'états pour cet algorithme.

1- On utilisera le module Cal_Max déjà fait en TD3/TP3 pour réaliser l'action « Calculer Max et Ad_Max de Tab (Ad_Courante … Ad2) », (cette action prendra plusieurs cycles en fonction de la taille du tableau).

2- Comme pour le module Cal_Max, on utilisera un état de départ dans lequel on restera tant que le signal cal (qui commande le calcul) est à 0, et dès que cal est remis à 0, on revient à l'état de départ quel que soit l'état courant.

3- Dès que l'ordre de calcul est donné (cal = 1), on passe à l'état suivant en exécutant l'action «Ad_courante ← Ad1 »

4- Depuis cet état, on réalise les actions de la boucle « Répéter » :

a- Calculer Max et Ad_Max de Tab (Ad_Courante … Ad2) : cette action sera réalisée par le module Cal_Max, et se fera en bouclant dans ce même état que l'on pourra nommer « CALMAX » jusqu'à ce que le module Cal_Max indique que le calcul du max est terminé.

b- Quand le calcul du max est terminé, on sort de l'état « CALMAX » en se dirigeant vers de nouveaux états qui nous permettront de réaliser les actions élémentaires qui implantent l'actions complexe « Echanger Tab (Ad_Courante) et Tab (Ad_Max) ». Il faut savoir que « Echanger » nécessitera une lecture mémoire et deux écritures mémoires, et que ces 3 opérations ne peuvent pas se faire en même temps, et consomment donc 3 cycles.

c- On peut s'arranger pour que l'action « Ad_Courante ← Ad_Courante + 1 » puisse être réalisée en même temps que la dernière opération de « Echanger ».

d- après avoir réalisé ces actions, on se trouve devant deux possibilités :

d1- Ad_Courante < Ad2 : On repart vers l'état « CALMAX » pour réaliser de nouveau les actions de la boucle Répéter

d2- Ad_Courante >= Ad2 on part vers l'état final, qui indique que le calcul est terminé et on y reste tant que cal = 1

1- Représenter le graphe d'états

Dans un module nommé « Etats_Tri_Tab », représenter le Graphe d'états sous la forme suivante (exemple : etats_cal_max, dans sa version non optimisée). Donner des noms significatifs aux différents états :

Transition	Condition	opérations
* -> INITAD	/cal	aucune
INITAD -> INITMAX	cal	adCourante <- ad1
INITMAX -> MAJMAX	cal	max <- Tab (adCourante) adMax <- adCourante adCourante ← adCourante + 1
MAJMAX -> MAJMAX	cal*/adCouranteSupAd2	Si Max > Tab (adCourante) Max <- Tab (adCourante) adMax <- adCourante adCourante ← adCourante + 1
MAJMAX -> FINI	cal* adCouranteSupAd2	aucune
FINI -> FINI	cal	aucune

Montrez votre travail à votre enseignant de TP, avant d'en faire l'implantation. ***3 points***

2- Réalisez le module Etats_Tri_Tab

en utilisant une bascule D pour chaque état.

Ce module ne gère pas les actions et ne connaît pas Ad1 et Ad2. On lui fournira donc deux signaux en en entrée : maxOK pour lui indiquer que le calcul du max est terminé, et finTab pour lui indiquer que l'on a atteint la fin du tableau (Ad_Courante >= Ad2).

Testez bien ce module : au départ, l'état initial (et seulement cet état) est = 1, bouclage dans l'état l'état « CALMAX » (nombre de cycles dépend de Ad1 et Ad2), sortie de « CALMAX » quand maxOK=1, retour sur l'état « CALMAX » après l'échange si finTab=0, transition vers l'état final et maintient dans cet état tant que cal est à 1.

Bloquez ce module. ***3 points***

3- Module Tri_Tab

On souhaite réaliser le module tri_tab qui effectue le tri d'un tableau (stocké dans une RAM interne au module, entre les adresses ad1 et ad2)

module Tri_Tab (rst, clk, cal, ad1[7..0], ad2[7..0] :

etats[N-1..0],	// les N états fournis par Etats_Tri_Tab, l'état initial à l'indice 0
calMax,	// le signal donnant l'ordre du calcul du max (c'est l'entrée cal du module // cal_max). Il reste à 1 jusqu'à la fin du calcul du max
adCourante[7..0]	// adresse courante
etatsMax[3..0],	// les 4 états de cal_max // on peut écrire : etatsMax[3..0] = FINMAX & MAJMAX & INITMAX & INITAD
max[31..0],	// la valeur du max courant
adMax[7..0],	// l'adresse du max courant
adRam[7..0],	// l'adresse à l'entrée de la RAM
inRam[31..0],	// la donnée à l'entrée de la RAM
wRam,	// le signal d'écriture dans la RAM
outRam[31..0])	// la donnée à la sortie de la RAM

On met les états sous forme de vecteur (etats[N-1..0], et etatsMax[3..0]) seulement dans la sortie du module, et gardera les états avec leur nom à l'intérieur de tritab.

Pour pouvoir vérifier le résultat dans la RAM, on fera de sorte de pouvoir lire le contenu de l'adresse entrée sur ad1 sur la sortie outRam lorsque cal = 0. Ce qui revient à écrire :

```
adRam[7..0] =  /cal*ad1[7..0]          // pour lire la ram quand cal=0
               + cal* …                // opérations sur la RAM lors du tri
```

A- Pour éviter des difficultés de mise au point, vous **devez réaliser un module intermédiaire** que l'on appellera tri_tab_int, et qui implante le calcul des max sans réaliser l'action « Echanger Tab (Ad_Courante) et Tab (Ad_Max) ».
Dans cette étape, on n'accède à la mémoire qu'en lecture lors du calcul du max, et on aura les entrées de la Ram comme suit :

```
wRam = 0
adRam[7..0] =  /cal*ad1[7..0]              // pour lire la ram quand cal=0
               + cal*CALMAX*/maxOK*adCourCalMax [7..0]  // calcul du max en cours
```

Testez ce module selon le scénario suivant :

- Charger dans la Ram le fichier tableauRam sur moodle
- Mettre l'entrée rst à 1 puis à 0
- Mettre cal à 1, ad2 à 00000011 et toutes autres entrées à 0
- Cliquez sur clk plusieurs fois jusqu'à avoir maxOK=1
- Vérifier que max = 0…01 0001 0001 et que adMax = 0000 0001
- Répéter les clics en vous arrêtant à chaque passage à maxOK=1, et en vérifiant à chaque fois les valeurs de max et de adMax
- Vérifier qu'après le 3ème passage à maxOK=1, les clics suivants conduisent à l'état final du tri.
- Vérifier que le contenu de la mémoire n'a pas changé.

Indiquez en commentaire si ce module fonctionne comme demandé. Bloquez le. ***7 points***

B- Réalisez le module Tri_Tab en : ***7 points***

- reprenant une copie du code précédent
- en y indiquant en commentaire :
  - les cas où wrRam = 1
  - les termes de de l'équation de adRam (les différents adresse d'accès à la Ram en lecture ou écriture)
  - les termes de de l'équation de inRam (ce qu'on écrit dans la RAM)
- Compléter avec ce qui est nécessaire pour réaliser l'action : Echanger Tab (Ad_Courante) et Tab (Ad_Max)

Bloquez ce module avant de quitter la séance de TP

Vous pouvez poursuivre votre travail dans un autre module Tri_Tab_2, qu'il faudra ***bloquer au plus tard le samedi de la semaine de ce TP avant minuit.***