ELECINF102
Processeurs et Architectures Numériques

Partie théorique

Vendredi 15 juin 2012

Sans calculatrice

Document autorisé : une feuille recto-verso

Durée: 45 minutes

Les exercices de ce contrôle sont **indépendants** et peuvent donc être traités dans n'importe quel ordre. Lisez bien l'énoncé en entier avant de commencer !

N'oubliez pas d'inscrire vos nom, prénom, et numéro de casier sur votre copie.

Bon courage !

1 Questions de cours

1. Qu'est-ce qu'un mapping mémoire (cartographie mémoire) pour un processeur ?
2. Donner la représentation en binaire CA2 de $-27,6_{10}$ sur 8 bits de partie entière + 5 bits de partie décimale.
3. Sur n bits, quel est l'intervalle des nombres signés représentables en CA2 ?
4. Complétez la phrase : un compteur sur 8 bits est un automate (une machine à états finis) à au moins ..... états.
5. Qu'est-ce qu'un chemin critique ?

2 Multiples de m et n

On rappelle qu'on peut utiliser toutes les portes vues en cours / TP / TD. L'horloge et le signal d'initialisation asynchrone des bascules D ne seront pas représentés. On attachera une attention particulière à la clarté des schémas.

Soit deux nombres `m` et `n` sur 8 bits. Tracez le schéma d'un circuit prenant en entrée `m` et `n` et produisant à chaque cycle, sur 32 bits, les multiples de `m` ou `n` par ordre croissant. Par exemple, si $m = 2$ et $n = 3$, le circuit produira la sortie suivante : 0, 2, 3, 4, 6, 8, 9, 10, 12, 14, 15, 16, ...

3 Fibonacci

On demande d'écrire le code SystemVerilog d'un circuit calculant sur 8 bits les nombres de la suite de Fibonacci : $U_n = U_{n-1} + U_{n-2}$, avec $U_1 = 1$ et $U_0 = 0$.

On propose le code SystemVerilog ci-dessous.

```
module fibonacci(clk,
                 reset_n,
                 U);

   input logic        clk;
   input logic        reset_n;
   output logic [7:0] U;

   // Un-1 (= Un au cycle précédent)
   logic [7:0]        U_1;

   // Un-2 (= Un-1 au cycle précédent)
   logic [7:0]        U_2;

   always @(posedge clk or negedge reset_n)
     if(reset_n==0)
       begin
          U   <= 1;
          U_1 <= 1;
          U_2 <= 0;
       end
     else
       begin
          // Un = Un-1 + Un-2
          U    <= U_1 + U_2;
          // Un-1 = Un retardé d'un cycle
          U_1  <= U;
          // Un-2 = Un-1 retardé d'un cycle
          U_2  <= U_1;
       end
endmodule
```

1. Donnez dans un tableau les valeurs de `U`, `U_1` et `U_2` aux 10 premiers cycles d'horloge (après relâchement du `reset_n`).
2. Tracez le schéma du circuit modélisé par ce code (l'horloge et le signal d'initialisation ne seront pas représentés).
3. Quelle est l'équation de la suite U_n générée ?
4. Corrigez le code SystemVerilog de façon à générer la suite de Fibonacci.

4 Horloge

On cherche à réaliser un compteur de secondes / minutes. On demande pour cela d'écrire le code SystemVerilog d'un compteur modulo 10, possédant les entrées-sorties suivantes :

- entrée `clk` : horloge système
- entrée `reset_n` : signal d'initialisation asynchrone, actif à l'état bas
- entrée `enable` : à l'état haut le module compte, à l'état bas le module garde sa valeur.
- sortie `s` : sortie sur 8 bits du compteur.
- sortie `cout` : sortie sur 1 bit valant 1 pendant un cycle lorsque le compteur vaut 9 et passe à 0 au prochain cycle.

On propose le code SystemVerilog ci-dessous.

```
module compteur10(input logic  clk,
                  input logic  reset_n,
                  input logic  enable,
                  output logic q[8:0],
                  output logic cout);


   // Entrée du compteur
   logic                d[8:0];

   always @(posedge clk or reset_n)
     if(reset_n=0)
       d <= 0;
     else if (enable)
       begin
          // Par défaut, on compte
          d <= q+1;
          q <= d;

          // Si on arrive à 10, on repart à 0
          if(q=10)
            d <= 0;
       end;

   always@ ( * )
     // Si le compteur vaut 10 et qu'enable est haut, le compteur
     //  va boucler. On passe donc cout à 1.
     if (enable)
       cout <= (q==10);

endmodule;
```

1. Corrigez le code.