TP2 : Architecture matérielle [12]

TP – CE312/CE318

Préliminaires :
Le travail demandé dans ce TP repose sur le travail à maison TM2 **qui doit être préparé avant la séance de TP. Ce travail au format papier sera ramassé à la fin de la séance n°2**.

Un compte-rendu final par binôme sera à déposer sur Chamilo dans les 3 jours qui suivent la dernière séance de TP du dernier groupe. Il contiendra les réponses aux questions de cet énoncé TP2 couvrant les séances de TP 2 et 3 sur la partie matérielle du cours CE311/CE317.

Format PDF uniquement

Le nom du fichier doit respecter le format suivant : CE31X_TP*numerodugroupe_nom1_nom2.*pdf*.*

Ce qui est attendu dans vos comptes rendus :

- **Des explications claires et concises sur vos manipulations** contenant des **schémas** et/ou des **captures d'écrans commentées**.
- Les **réponses aux questions posées** en respectant la numérotation
- Les **codes sources et/ou chronogrammes réalisés**
- Une attention particulière sera apportée au choix des noms des différents signaux et les codes devront être commentés (succinctement)
- La notation portera sur ce compte-rendu (les points entre crochet donne un barème indicatif) **et** sur **votre travail en séance**
- Barème /20: TP2 [12] (sachant que les travaux à la maison TM1 et TM2 comptent pour [4] et le travail en séance [4] également)

1 Introduction

Le travail consistera à réaliser au cours des deux séances de TP un chronomètre. Ce chronomètre affichera les secondes et les minutes. Il s'incrémentera quand l'entrée start sera au niveau haut. Il pourra être remis à 0 avec un signal rst synchrone actif au niveau haut.

Ce chronomètre s'appuiera sur les compteurs modulo N préparés dans le travail à la maison.

Voici l'entité du chronomètre que vous allez réaliser :

```
entity chronometer is
    port(
        clk             : in STD_LOGIC;
        rst             : in STD_LOGIC;
        start           : in STD_LOGIC;
        seconds         : out STD_LOGIC_VECTOR (7 downto 0);
        minutes         : out STD_LOGIC_VECTOR (7 downto 0)
    );
end chronometer;
```

2 Validation des compteurs [1]

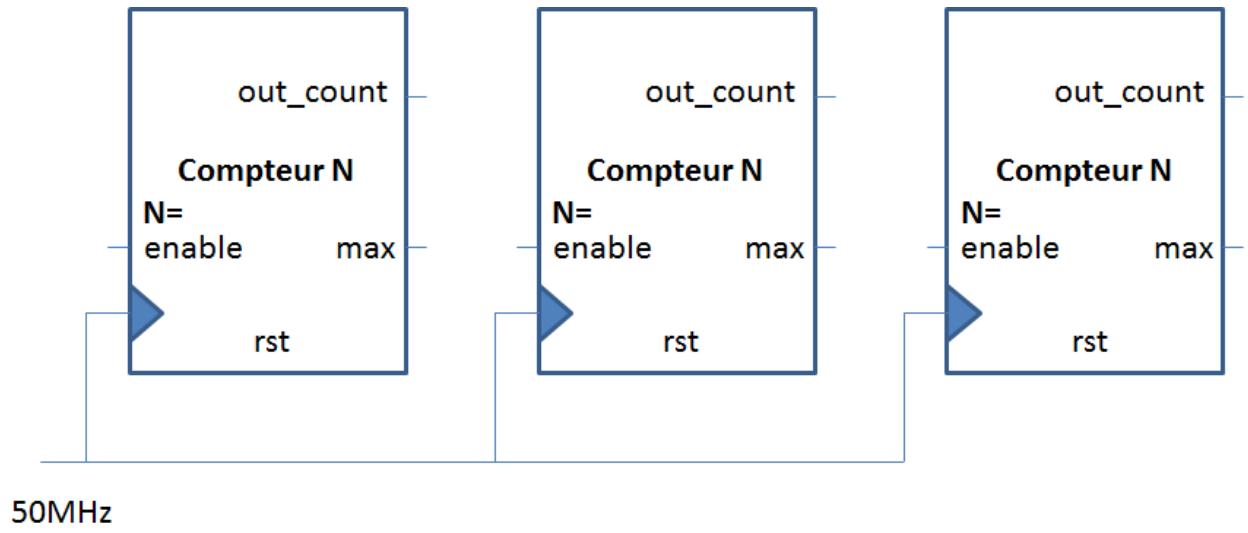
2.1 Vérifiez que les chronogrammes de votre compteur **modulo N** avec N=10, obtenus en simulation (à faire valider par l'enseignant), sont bien identiques à ceux dessinés manuellement [1].

3 Architecture du chronomètre [8]

3.1 Complétez le schéma ci-dessous de l'achitecture du chronomètre « secondes / minutes » basée sur 3 compteurs modulo N et d'une horloge systéme (commandant ces 3 compteurs) à 50MHz [2].

Remarques :

- Faire apparaitre toutes les entrées/sorties et les interconnexions entre les deux compteurs.
- Ajouter d'éventuels composants si nécessaire…
- Dans la figure ci-dessus, précisez bien les valeurs N des modulos des 3 compteurs.



3.2 Combien de bascules D seront nécessaires afin d'implémenter cette architecture du chronomètre [1]?

3.3 Codez cette architecture du chronomètre en instanciant les 3 compteurs modulo N comme représenté dans la figure précédente [3].

3.4 Codez un testbench puis vérifiez en simulation l'architecture précédente du chronomètre (Faire valider vos chronogrammes par l'enseignant) [3].

Remarque : la simulation complète du chronomètre peut prendre beaucoup de temps, il est donc impossible de simuler la totalité du fonctionnement du chronomètre. Pour finaliser la vérification, vous allez directement implémenter le chronomètre sur FPGA.

3.5 Avant de passer à l'étape suivante, faire une synthèse de votre chronomètre et vérifiez dans le rapport de synthèse que le nombre de bascules correspond bien au nombre de bascules théoriques indiqué dans la question 3.3 [1].
Si ce n'est pas le cas corrigez le problème dans votre code et vérifiez à nouveau que la simulation fonctionne avant de passer à la suite.

Remarque : le rapport de synthèse ne doit contenir aucune Latch (bascule verrou active sur niveau), et doit contenir uniquement des bascules de type Flip-Flop actives sur front.

4 Implémentation sur FPGA [3]

Afin de valider votre chronomètre sur FPGA (sur la carte de développement basys3) vous allez devoir utiliser un composant VHDL externe et un fichier de contrainte associé Basys3_master.xdc (voir le projet Basys3_7segdisp dans l'archive Basys3_7segmentdisplay.zip sur Chamilo) permettant de gérer les afficheurs 7 segments présents sur la carte basys3.

4.1 Créez une nouvelle entité et architecture du chronomètre incluant cette fois la gestion des 4 afficheurs 7 segments de la carte Basys3 [2].

4.2 Faire une synthèse, puis vérifiez sur la carte Basys3 que votre chronomètre fonctionne (Faire valider par l'enseignant). Si votre chronomètre ne fonctionne pas, corrigez le problème et revenez à la question 3.5 [1].