

TP 4 : Compteurs et chronomètre

Pour ce TP, il est demandé de ne créer qu'un seul projet Quartus dans le répertoire TP4

Partie I : les compteurs

1) Compteurs asynchrones

2) Compteurs synchrones


Partie II : Application : Réalisation d'un chronomètre à affichage digital

Partie I : les compteurs

Les compteurs sont classés en deux catégories suivant leur mode de fonctionnement et leur structure. On distingue :

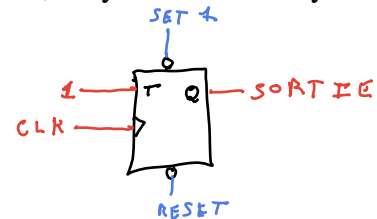
- les compteurs asynchrones
- les compteurs synchrones

Un système séquentiel est synchrone lorsque tous les changements d'états du système sont liés à l'activité du même signal d'horloge. Si cette condition n'est pas vérifiée, le système est dit asynchrone. Commençons par l'étude de ce second type de système.

Si $T=1$: Si les montants 
 \Rightarrow change de valeur

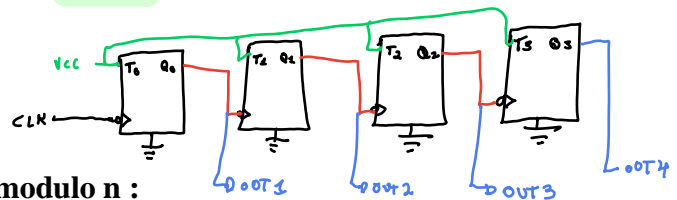
1) Compteurs asynchrones

Si $T=0$: Rien ne passe



Un **compteur asynchrone** peut être obtenu par la **mise en cascade** de bascules TFF où la sortie de la bascule précédente vient alimenter le signal d'horloge de la bascule suivante. La structure des compteurs asynchrones permet ainsi de propager en cascade l'ordre de changement d'état des bascules.

Réaliser un compteur asynchrone fonctionnant sur 4 bits.



2) Compteurs synchrones

a) Création en VHDL d'un compteur synchrone modulo n :

Ce type de compteur est destiné à compter les fronts montants d'un signal d'horloge (CLK_in : IN STD_logic). Le signal de sortie du compteur (CLK_out : Out STD_logic) est à '0' tant que le nombre d'impulsions de la clock est inférieur à n, puis à '1' lorsque le nombre de fronts montants = n. Le compteur pourra être réinitialisé par un signal 'Reset', signal 'reset' : remise à zéro

- Compléter l'architecture de l'entité *Compteur_n.vhd* ci-dessous, afin de réaliser la fonction de comptage, avec 'Reset' Synchrone.

```

ENTITY Compteur_n IS
--Passage de la valeur Max du compteur en parametre
GENERIC (n:integer range 0 to 15:=10); --parametre generique fixant la valeur max du compteur
PORT (
    CLK_in:    IN STD_LOGIC;    -- Clock en entree
    RESET:     IN STD_LOGIC;    -- reset le compteur à la valeur 0

    V_COUNT:   OUT STD_LOGIC_VECTOR ( 3 Downto 0); -- Valeur du Compteur
    CLK_Out:   OUT STD_LOGIC    -- signal Clock en Sortie
);
END ENTITY;

ARCHITECTURE Arch of Compteur_n is

End Arch;
    
```

Indications :

- Il sera nécessaire d'implémenter un Process dans l'architecture :

```

Process (CLK_in)
Begin
    
```

```

End process ;
    
```

- Afin que le process intervienne sur le front montant de l'horloge, il faut tester la condition suivante à l'intérieur du process :

```

    if (CLK_in'event and CLK_in='1') then

    --Ou    IF (RISING_EDGE (CLK_in)) THEN
    
```

- Pour mettre tous les bits à zéro d'un signal :

Exemple :

```

Signal Ncont : STD_LOGIC_VECTOR ( 3 Downto 0);
    
```

Ecrire :

```

Ncont<= (Others=>'0');
    
```

➤ **Faire une simulation fonctionnelle de l'entité Compteur_n.vhd**

Vérifier que le signal 'CLK_out' passe bien à la valeur '1' lorsque le compteur atteint la valeur n-1. (comptage de 0 à 9 correspond à un compteur modulo 10)

b) Modifier l'entité Compteur_n.vhd afin que le reset soit Asynchrone

c) En s'inspirant de l'entité précédente, créer une nouvelle entité 'Timer.vhd' qui génère un signal de sortie de type horloge à 1hz à partir du signal d'entrée *CLK_in*

Tenir compte du fait que le signal d'entrée peut avoir une fréquence très importante. (Clock à 50 Mhz présente sur la carte DE1-SoC : CLOCK_50) et que l'on souhaite un signal de sortie à une fréquence de 1 Hz.

- Modifier l'entité précédente 'Timer.vhd' afin que le comptage s'effectue lorsqu'un signal externe 'start' est égal à '1'
- Faire une simulation de l'entité Timer.VHD pour une valeur faible du paramètre générique n
- Créer un Block Symbol de ce code et insérer-le dans un Block Diagram Schematic.
- Placer des pins d'entrée et sortie afin de tester ce circuit sur la carte.

Utiliser 'CLOCK_50' comme nom pour le signal se connectant sur 'CLK_in '

Utiliser une led afin de visualiser le signal de sortie CLK_OUT

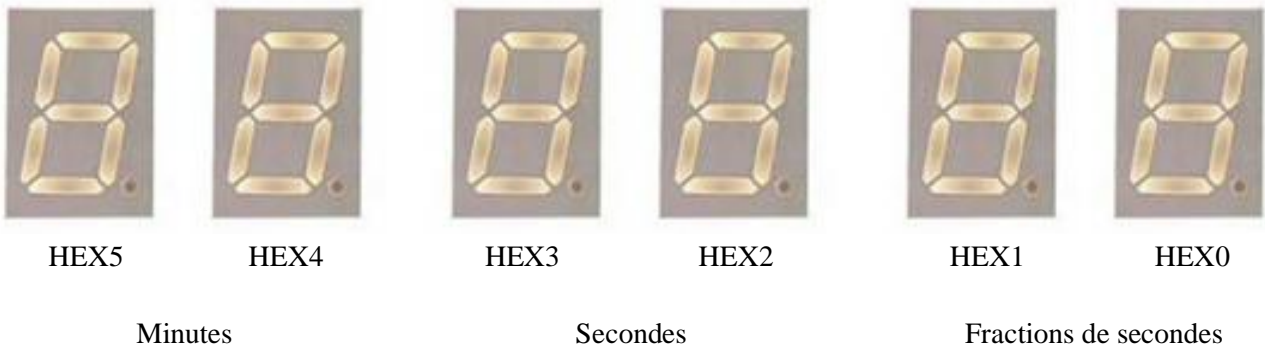
- Effectuer l'assignation des pins afin d'implanter ce design sur le FPGA
- Pour une valeur faible du paramètre générique n, refaire une simulation mode real time.
- Régler le paramètre générique n de façon à générer une Clock de fréquence 1Hz
- Compiler et Implémenter sur la carte DE1-SoC et vérifier le clignotement de la LED à 1HZ.

Partie II : Réalisation d'un chronomètre à affichage digital

Le but final de ce TP est de réaliser sur la carte DE1-SoC un Chronomètre à affichage digital. Ce chronomètre devra afficher : les minutes, les secondes, 10ème de seconde et 1/100ème de secondes.

- 1) Créer un block symbole à partir de l'unité : 'Compteur_n.vhd'
- 2) Recopier le code vhdl du TP3 précédent correspondant au décodeur 7Segment dans le répertoire du projet. L'ajouter au projet, le compiler et créer un bloc symbole.
- 3) A partir de l'unité décrivant le compteur, re créer une nouvelle unité VHDL de ce compteur en version simplifié : ce compteur sera destiné à compter des nombres de 0 à 15, donc la sortie sera sur 4bits. **Suppression du signal 'start'**. Créer également un bloc symbole de ce composant.
- 4) Vous devez ensuite réaliser un système de comptage de type chronomètre (minutes : secondes : centièmes). Ce système affichera le temps écoulé depuis le lancement. On demande également de prévoir une remise à zéro. Pour cela, il est conseillé de traiter le problème par étapes successives, en affichant d'abord les millièmes, les centièmes, ...puis les secondes, et enfin les minutes.

Vous pouvez au choix réaliser des blocs fonctionnels élémentaires (compteur modulo 10, compteur modulo 6), permettant de commander chacun un afficheur Hexadecimal. Chaque bloc disposera d'une sortie compteur sur 4 bits qui pourra être connecté à un décodeur 7 segment.



Fin du TP 4