6GEI228 – Systèmes Digitaux

Laboratoire #8

Compteurs en VHDL

Hiver 2018

1. Objectifs

- Se familiariser avec le VHDL séquentiel
- Apprendre à concevoir des compteurs en VHDL

2. Méthodologie

Le but de ce laboratoire est de vous montrer comment concevoir des compteurs en VHDL. Le début du laboratoire sera un court tutoriel où vous ferez un compteur. Dans la deuxième partie du laboratoire, vous aurez à concevoir un transmetteur AM qui envoie la note LA (440Hz) sur un poste AM de votre choix. Comme vous allez le découvrir, le transmetteur sera principalement deux compteurs dont les sorties sont combinés ensemble par une porte ET.

3. Tutoriel

Un compteur est un bloc séquentiel qui incrémente sa valeur en sortie de 1 à chaque coup d'horloge. À son état le plus simple, un compteur peut être conçu avec des flip-flops et un additionneur.

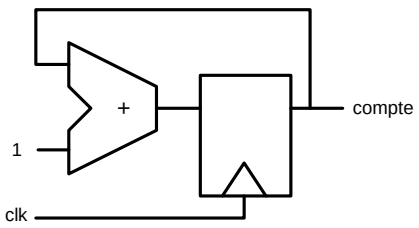

Figure 1. Bloc séquentiel représentant un compteur.

Vu de l'extérieur, ce compteur aurait 1 port d'entrée et 1 de sortie; une horloge en entrée et le compte en sortie. On pourrait donc écrire la partie du code associée aux librairies et à l'entité :

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;

ENTITY compteur IS
   PORT (
        clk    : IN  STD_LOGIC;
        compte : OUT STD_LOGIC_VECTOR(3 DOWTO 0)
       );
END compteur;
```

Pour implémenter ce compteur en VHDL, on utilise un process séquentiel: la liste de sensibilité contient seulement clk et doit avoir une structure assez spécifique.

```
PROCESS (clk)
BEGIN
   IF clk'EVENT AND clk = '1' THEN
      …
   END IF;
END PROCESS;
```

Sachant que chaque assignation dans un process séquentiel donne des flip-flops, on pourrait simplement écrire que le compte est égal au compte qui a été incrémenté de 1 :

```
PROCESS (clk)
BEGIN
   IF clk'EVENT AND clk = '1' THEN
      s_compte <= s_compte + 1;
   END IF;
END PROCESS;
```

Remarquez que le signal utilisé n'est pas la sortie *compte* mais plutôt un signal intermédiaire *s_compte*. La raison est que *compte* est une sortie et on ne peut pas LIRE une sortie. C'est-à-dire que la commande `compte <= compte + 1` ne fonctionnerait pas parce que, pour modifier la sortie *compte,* il faut LIRE cette sortie. On utilise donc un signal intermédiaire *s_compte* qu'on va connecter à la sortie en dehors du process. Au final, le compteur ressemblerait à ceci :

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;

ENTITY compteur IS
   PORT (
        clk : IN STD_LOGIC;
        compte : OUT STD_LOGIC_VECTOR(3 DOWTO 0)
       );
END compteur;

ARCHITECTURE rtl OF compteur IS

SIGNAL s_compte : STD_LOGIC_VECTOR(3 DOWNTO 0);

BEGIN

PROCESS (clk)
BEGIN
   IF clk'EVENT AND clk = '1' THEN
      s_compte <= s_compte + 1;
   END IF;
END PROCESS;

compte <= s_compte;
END;
```

Avec un compteur, il est aussi possible de faire un diviseur d'horloge. Un diviseur d'horloge est un système qui prend une horloge d'une certaine fréquence en entrée et fait sortir une horloge de plus basse fréquence à la sortie.

Pour ce tutoriel, il sera question de prendre une horloge de 50MHz en entrée et de générer une horloge de 5MHz à la sortie. Pour ce faire, il est possible de s'inspirer du design précédent. Sachant que le but est de diviser l'horloge d'entrée par 10, notre approche sera de faire un compteur qui compte jusqu'à 10 (plutôt allant de 0 à 9). Il faudrait donc regarder le compteur à chaque cycle et si le compte est rendu à 9, nous devons le remettre à 0 au prochain cycle. Sinon, il faut incrémenter la valeur du compteur.

```
PROCESS (clk)
BEGIN
   IF clk'EVENT AND clk = '1' THEN
      IF s_compte = 9 THEN
         s_compte <= (OTHERS => '0');
      ELSE
         s_compte <= s_compte + 1;
      END IF;
   END IF;
END PROCESS;
```

Avec un compteur, il est possible de générer une horloge en sortie de la fréquence voulue. Avec la valeur du compte, il est possible de générer une horloge en mettant la sortie à 0 la moitié du temps et 1 durant l'autre moitié.

Il est possible de générer cette horloge de façon combinatoire ou séquentielle. De la façon combinatoire, il faudrait utiliser une commande comme celle-ci à l'extérieur du process :

```
sortie <= '1' WHEN s_compte > 5
          ELSE
         '0';
```

Cette approche fonctionne bien pour certaines applications mais moins bien pour d'autres. La raison est qu'une comparaison combinatoire peut parfois générer des aléas de fonctionnement « glitchs » (ça change entre 0 et 1 une ou plusieurs fois avant de se stabiliser à la valeur finale). Ceci est dû au fait que les bits ne se propagent pas tous à la même vitesse. Certains bits arrivent avant d'autres et donc, la sortie peut changer de valeur plusieurs fois avant que tous les bits du compte arrivent.

Figure 3. Schéma d'un dysfonctionnement soudain (Gitch).

Si ce genre de comportement n'était pas souhaitable, il est possible d'utiliser un process séquentiel. Pour ce faire, il suffit d'utiliser un IF à l'intérieur d'un process séquentiel.

```
PROCESS (clk)
BEGIN
   IF clk'EVENT AND clk = '1' THEN
      IF s_compte > 5 THEN
         sortie <= '1';
      ELSE
         sortie <= '0';
      END IF;
   END IF;
END PROCESS;
```

4. Théorie

La modulation en amplitude (AM) est une technique pour la transmission de signaux par les ondes radio. L'idée derrière la transmission d'ondes radio est que TOUT signal « émet » une certaine énergie par les ondes radio. Les signaux de hautes fréquences vont tendre à « émettre » plus qu'un signal de basses fréquences. De plus, un signal qui circule dans un LONG fil (une antenne, par exemple) propagera plus d'énergie par les ondes radio qu'un signal qui circule dans un fil court. Un signal de basse fréquence devra donc avoir une antenne plus longue tandis qu'un signal de haute fréquence n'aura besoin que d'une antenne plus courte.

Certains des signaux que nous voulons transmettre (la voix ou la musique, par exemple) sont considérés comme étant de basses fréquences et ne propagent donc pas très bien avec des tailles d'antenne raisonnables. Pour pouvoir bien transmettre ce signal, il serait possible d'allonger l'antenne ou sinon MODULER ce signal avec une fréquence beaucoup plus élevée. Dans la figure suivante, on voit un signal de basse fréquence et le même signal qui est modulé avec un signal de plus haute fréquence. En connectant ce deuxième signal à un long fil, il est possible de transmettre ce signal par les ondes radio. À la réception, après démodulation, le système recevrait le signal de basse fréquence (premier signal).

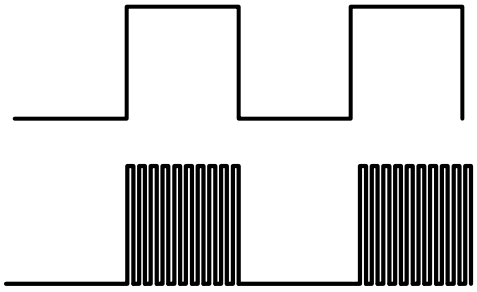

Figure 2. Signal de basse fréquence.

Dans un monde numérique, la modulation AM s'implémente facilement avec une porte ET. En faisant une fonction logique ET entre le signal à envoyer (basse fréquence) et le signal modulateur (haute fréquence), nous obtenons le signal modulé qui est prêt à être envoyé.

5. Travail demandé

Dans ce laboratoire, vous aurez à créer un transmetteur AM qui envoie la note LA au poste 1000KHz (1MHz) de la radio lorsqu'on pèse sur un bouton. La note LA peut être vue comme étant une onde carrée (une horloge) de 440Hz.

L'idée de base est donc de créer une horloge de 1MHz, de créer une horloge de 440Hz, de les faire passer par une porte ET et de connecter ce signal a un long fil. Vous pourrez alors écouter votre son au poste 1000KHz de n'importe quelle radio AM qui est proche.

Montrez votre design fonctionnel au chargé de laboratoire.