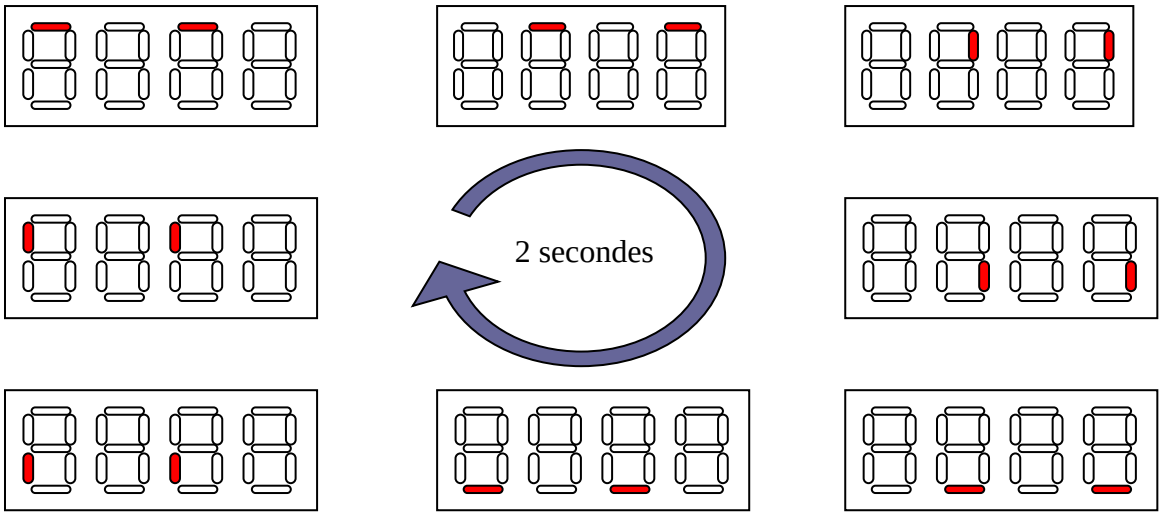
V H D L

Durée 1H45

Programme fonctionne : +14 points
Façon de programmer (présentation, clarté, solution optimisé, commentaire, …) : +6 points
Programme ne fonctionne pas : -6 points systématiquement.

Travail demandé

Nous souhaitons afficher le motif ci-dessous sur les afficheurs sept-segments. L'horloge en entrée est de 50 MHz et nous souhaitons que le temps d'un tour complet soit environs 2 secondes (0.25 secondes par changement).



Nom : **Prénom :**

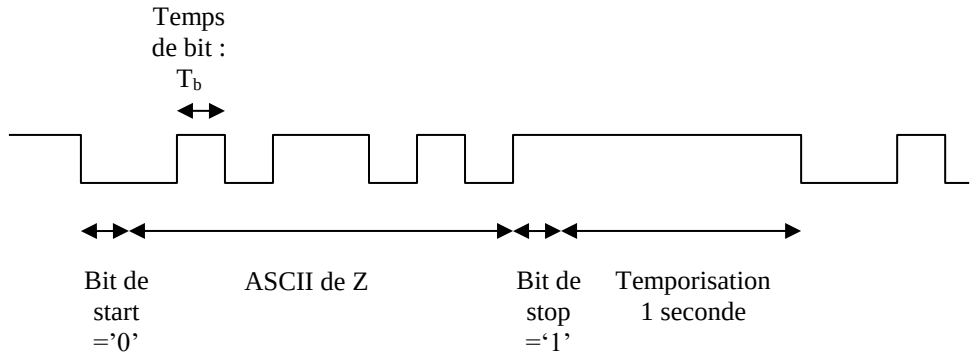
V H D L

Durée 1H15

Programme fonctionne : +14 points
Façon de programmer (présentation, clarté, solution optimisé, commentaire, …) : +6 points
Programme ne fonctionne pas : -6 points systématiquement.

Travail demandé

Nous souhaitons envoyer la lettre Z (code ASCII= $5A) sur le port série. Votre programme en VHDL doit fournir le signal ci-dessous



C'est un signal qui se répète toutes les secondes. Pour tester votre programme, utilisez le câble sériel et le programme hyper terminal de windows. La vitesse est fixée à 19200 bits par seconde. C'est à dire que T_b=1/19200.
La fréquence d'horloge de la carte est de 50 MHz.

Remarque :

Vous pouvez déclarer une constante :

```
constant ascii : std_logic_vector(9 downto 0) := "1010110100"
```

qui contient l'octet $5A encadré par le bit de start et de stop. Ensuite, définir un compteur allant de 0 à 9 pour pointer chaque bit de cette constante et envoyer ce bit sur la broche TX (R13 du FPGA).

Corrigé

```
entity envoi_serie is
   Port ( MCLK : in  STD_LOGIC ;
          TX : out  STD_LOGIC) ;
end envoi_ serie;

architecture Behavioral of envoi_ serie is
        constant ascii : std_logic_vector(9 downto 0) := "1010110100" ;
        type T_etat is (attente, envoi) ;
        signal etat : T_etat ;
        signal count : integer range 0 to 4095;
        signal uart_clk : std_logic;
        signal compt_1sec : integer range 0 to 32767;
        signal compt_bit : integer range 0 to 15;

begin
        process(MCLK)
        begin
                if MCLK'event and MCLK='1' then
                        uart_clk <= '0';
                        count <= count+1;
                        if count = 2604 then
                                uart_clk <= '1';
                                count <= 0;
                        end if;
                end if;
        end process;

        process(MCLK)
        begin
                if MCLK'event and MCLK='1' then
                    if uart_clk = '1' then
                        case etat is
                        when attente =>
                                TX <= '1' ;
                                compt_bit <= 0 ;
                                compt_1sec <= compt_1sec + 1 ;
                                if compt_1sec = 19200 then
                                        etat <= envoi;
                                end if;
                        when others =>
                                compt_bit <= compt_bit + 1;
                                TX <= ascii(compt_bit);
                                if compt_bit = 9 then
                                        etat <= attent;
                                        compt_bit <= 0;
                                end if;
                        end case;
                    end if;
                end if;
        end process;

end Behavioral;
```

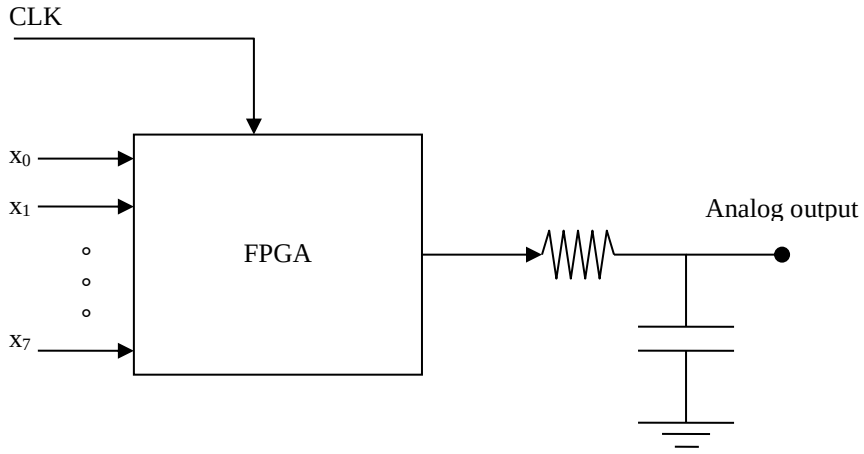
V H D L

Durée 1H45

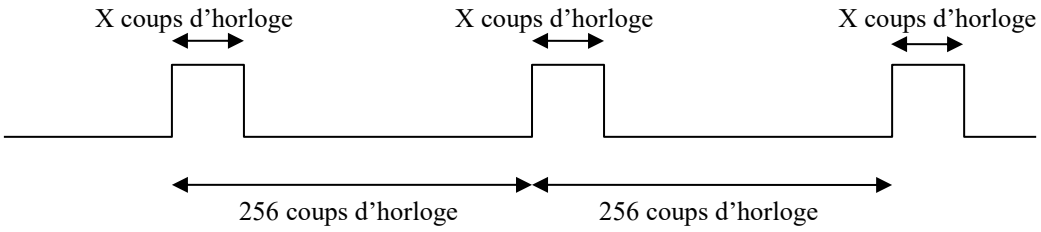
Programme fonctionne : +14 points
Façon de programmer (présentation, clarté, solution optimisé, commentaire, …) : +6 points
Programme ne fonctionne pas : -6 points systématiquement.

Travail demandé

Nous allons synthétiser un convertisseur numérique analogique simple dont le schéma est donné ci-dessous :



Le principe est le même que PWM. C'est-à-dire que vous devrez générer le signal suivant en fonction de X.



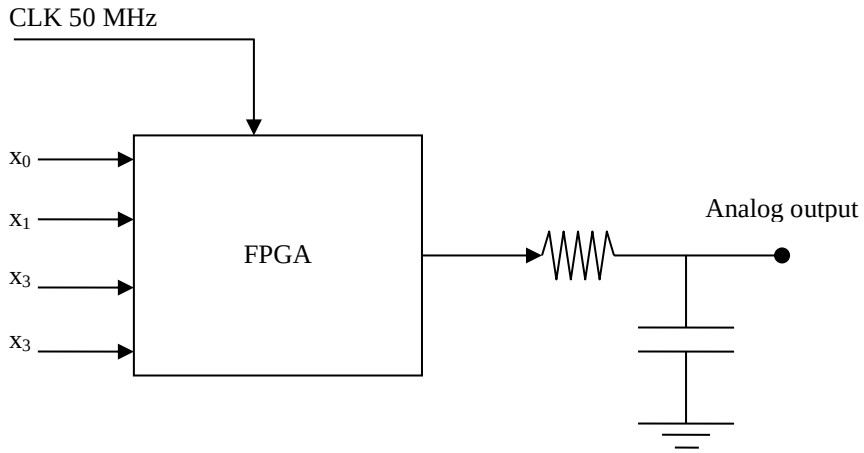
Les entrées de votre système sont les 8 interrupteurs de la carte. Faites sortir la sortie sur la broche 4 du connecteur A2. On utilisera un oscilloscope pour visualiser la tension en sortie.

V H D L

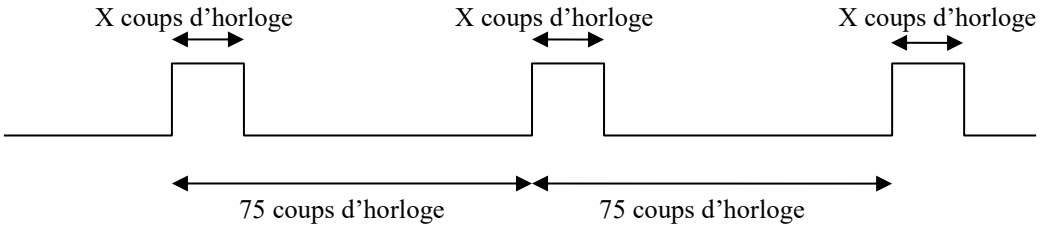
Durée 1H45

Programme fonctionne : +14 points
Façon de programmer (présentation, clarté, solution optimisé, commentaire, …) : +6 points
Programme ne fonctionne pas : -6 points systématiquement.

Nous allons synthétiser un convertisseur numérique analogique simple dont le schéma est donné ci-dessous :



Le principe est le même que PWM. C'est-à-dire que vous devrez générer le signal suivant en fonction de X.



Les entrées de votre système sont 4 interrupteurs de la carte. Faites sortir la sortie sur la broche 4 du connecteur A2. Si les 4 entrées sont à zéro, la sortie reste à 0, si les 4 sont à 1, la sortie reste à 1, et pour d'autres possibilités on génère un rapport cyclique proportionnel à la tension souhaitée. On utilisera un oscilloscope pour visualiser la tension en sortie.

Puis, utiliser votre circuit pour générer un signal de dent de scie de période (approximativement) 1 msec.

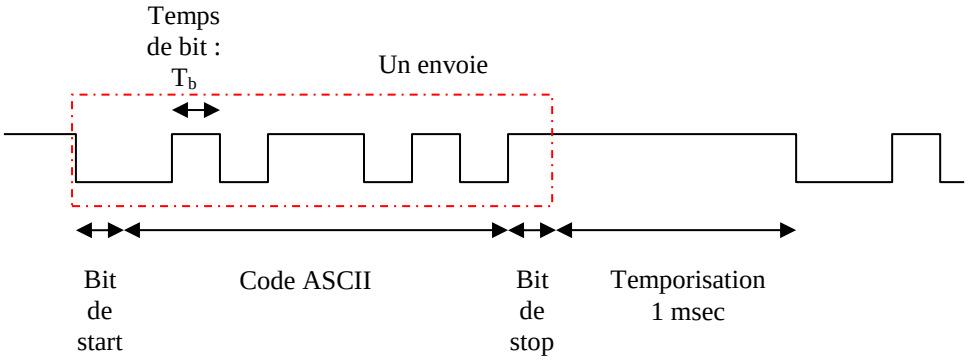
V H D L

Durée 1H45

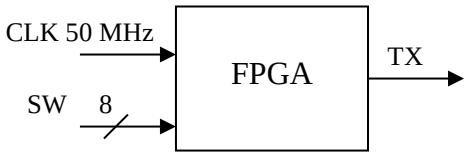
Programme fonctionne : +14 points
Façon de programmer (présentation, clarté, solution optimisé, commentaire, …) : +6 points
Programme ne fonctionne pas : -6 points systématiquement.

Nous souhaitons envoyer les codes ASCII respectant le format RS232. Le code ASCII est fourni par les 8 interrupteurs de la carte Spartan III. Le programme doit répéter l'envoie du code toutes les 1 msec.

Rappel sur le protocole RS232.



Le LSB d'abord, pas de parité, et la vitesse est fixée à 19600 bits par seconde. C'est à dire que T_b=1/19200. La fréquence d'horloge de la carte est de 50 MHz.



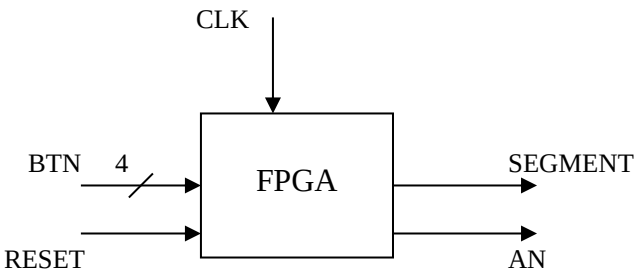
V H D L

Durée 1H45

Programme fonctionne : +14 points
Façon de programmer (présentation, clarté, solution optimisé, commentaire, …) : +6 points
Programme ne fonctionne pas : -6 points systématiquement.

Pour un jeu, on nous a demandé de faire un circuit qui détecte parmi les 4 candidats, celui qui a appuyé sur son bouton plutôt que les autres et afficher son numéro sur un des 7-segments. C'est-à-dire que l'entrée du circuit est 4 boutons poussoirs et la sortie indique un nombre allant de 1 à 4 désignant le gagnant (la première personne).

Utiliser un interrupteur pour remettre à zéro le circuit.



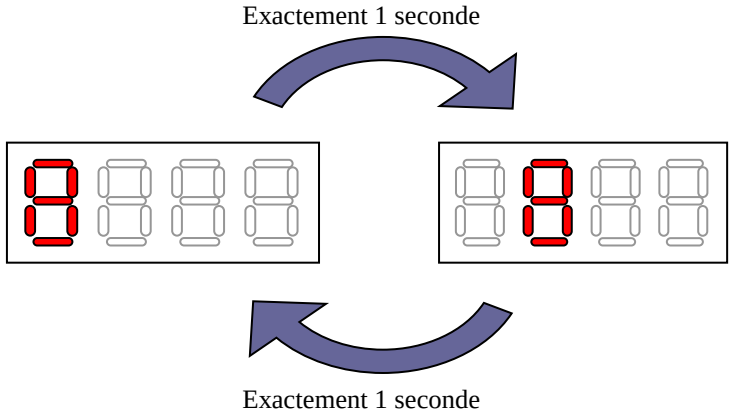
Nom : **Prénom :**

V H D L

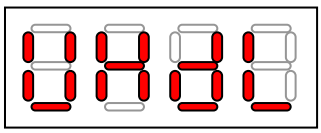
Durée 1H15

Programme fonctionne : +14 points
Façon de programmer (présentation, clarté, solution optimisé, commentaire, …) : +6 points
Programme ne fonctionne pas : -6 points systématiquement.

Faire clignoter les segments comme présente la figure ci-dessous :



Quand le circuit fonctionne, faites le voir par l'examinateur.
Ensuite, modifiez le circuit pour obtenir le message suivant sur les 7-segments.



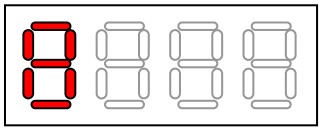
Nom : **Prénom :**

V H D L

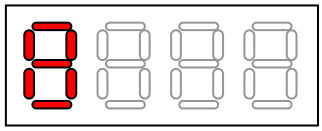
Durée 1H15

Programme fonctionne : +14 points
Façon de programmer (présentation, clarté, solution optimisé, commentaire, …) : +6 points
Programme ne fonctionne pas : -6 points systématiquement.

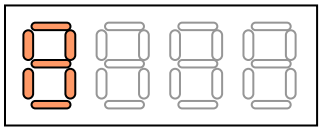
Nous allons changer la luminosité d'un des 7-segments en jouant sur le rapport cyclique.
1- D'abord faites afficher



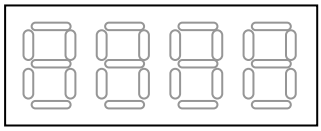
2- Le principe est d'allumer la LED pendant x.T second et de l'éteindre pendant (255-x)T. En faisant varier le x entre 0 et 255, la luminosité va être modifiée. Pour pouvoir varier le x, utilisez les 8 interrupteurs de la carte. Quand toutes sont à '1' nous devrons obtenir la luminosité maximale et quand toues sont à zéros, les LEDs sont éteintes.



SW= "11111111"



SW= "10000000"



SW= "00000000"

Nom : **Prénom :**

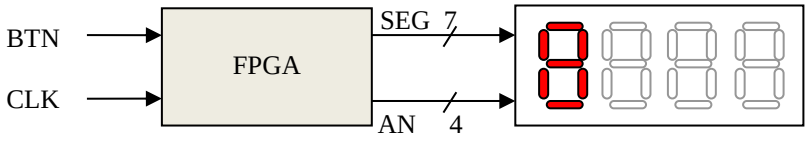
V H D L

Durée 1H15

Programme fonctionne : +14 points
Façon de programmer (présentation, clarté, solution optimisé, commentaire, …) : +6 points
Programme ne fonctionne pas : -6 points systématiquement.

1- Le but est de créer un dé avec la carte NEXYS3. A chaque appuie sur un des boutons de la carte, une valeur entre 1 et 6 doit être affichée sur le 7-Segments de gauche. Pour cela, créer un compteur entier allant de 1 à 6 à la vitesse max (100 MHz). Puis quand vous appuyez sur le bouton, la valeur de compteur est sauvegardée dans un registre qui est toujours affiché sur le segment de gauche.

2- Pour donner un effet à cette réalisation, à chaque appuie du bouton, on fait d'abord clignoter le segment de gauche 6 fois (visible à l'œil) avec la valeur de dé à afficher avant de fixer cette valeur.



Au dos de cette feuille dessiner l'architecture globale de votre design.

A chaque étape, quand le circuit fonctionne, faites le voir par l'examinateur.

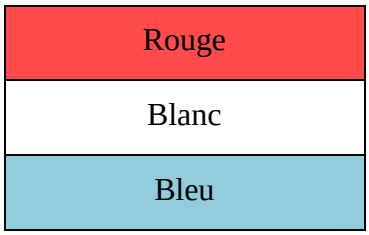
V H D L

Durée 1H45

Programme fonctionne : +14 points
Façon de programmer (présentation, clarté, solution optimisé, commentaire, …) : +6 points
Programme ne fonctionne pas : -6 points systématiquement.

Le but de ce TP est d'afficher des motifs sur l'écran VGA.

1- Afficher le drapeau du Pays-Bas sur l'écran



2- Afficher une ligne verticale blanche qui se déplace à l'écran. Il parcourt la largeur de l'écran de gauche à droite en 2 secondes (la taille de l'écran est 640x480) et il recommence.

3- Faire un programme pour donner à l'utilisateur le choix entre les deux affichages que vous avez fait : si SW='1' c'est le drapeau qui s'affiche et si SW='0 c'est la ligne.

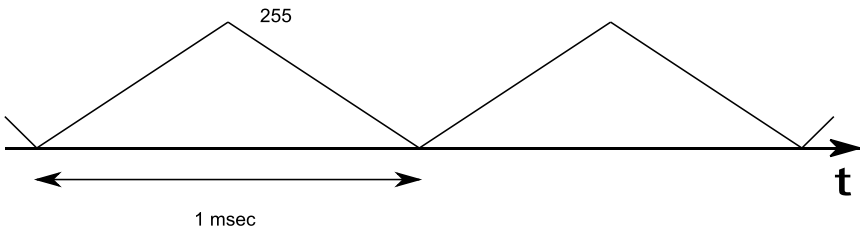
V H D L

Durée 1H45

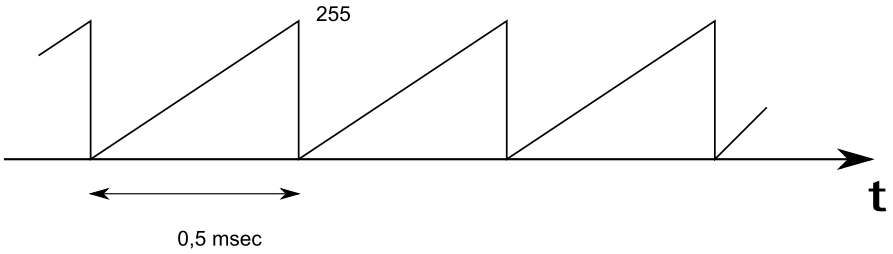
Programme fonctionne : +14 points
Façon de programmer (présentation, clarté, solution optimisé, commentaire, …) : +6 points
Programme ne fonctionne pas : -6 points systématiquement.

Le but de cet exercice est de générer un signal analogique.

1- Générez le signal ci-dessous que vous vérifiez sur l'oscilloscope.



2- L'utilisateur peut, en appuyant sur le BTND de la carte NEXYS3, changer la forme d'onde en :



3- A chaque appuie du BTND, on change de forme d'onde : forme 1, forme 2, tout à 255, tout à zéro, et on reboucle.

Les broches du FPGA qui sont connectées au port d'extension JD.

```
#Net "CNA<0>" LOC = G11
#Net "CNA<1>" LOC = F10
#Net "CNA<2>" LOC = F11
#Net "CNA<3>" LOC = E11
#Net "CNA<4>" LOC = D12
#Net "CNA<5>" LOC = C12
#Net "CNA<6>" LOC = F12
#Net "CNA<7>" LOC = E12
```

V H D L

Durée 1H15

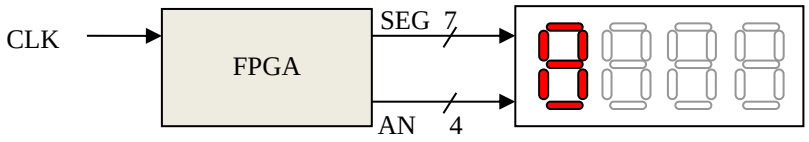
Programme fonctionne : +14 points

Façon de programmer (présentation, clarté, solution optimisé, commentaire, …) : +6 points

Programme ne fonctionne pas : -6 points systématiquement.

Le but est créer un compteur 1-digit allant de 0 à F sur le 7-segment de gauche de la carte nexys3. Ce travail se fait en étape et chaque étape est notée.

1- Afficher le chiffre 0 sur le 7-segment de gauche.
2- Afficher le compteur 1-digit qui compte toutes les secondes sur le 7-segment de gauche. Commencer par créer un compteur 4 bits qui s'incrémente toutes les secondes
3- Arrêter le comptage quand on arrive à F.



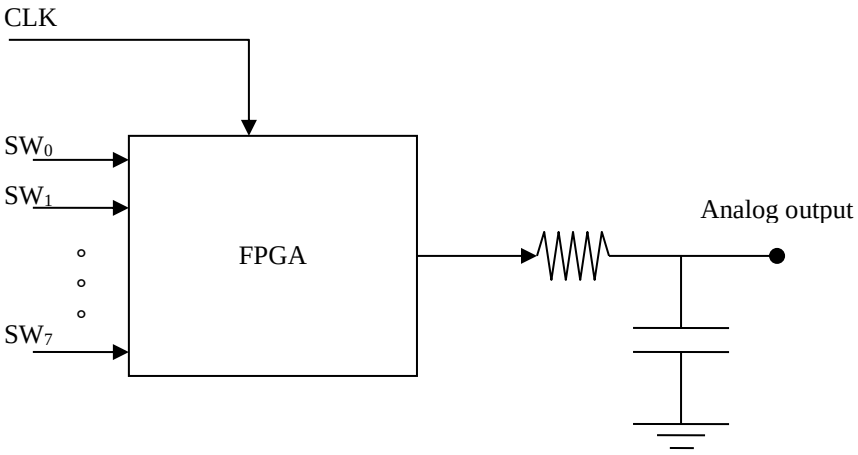
Nom : **Prénom :**

V H D L

Durée 1H15

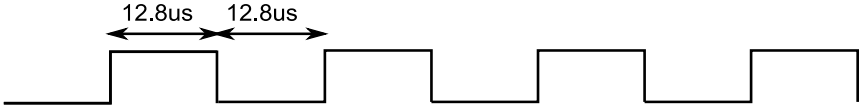
Programme fonctionne : +14 points
Façon de programmer (présentation, clarté, solution optimisé, commentaire, …) : +6 points
Programme ne fonctionne pas : -6 points systématiquement.

Nous allons synthétiser un convertisseur numérique analogique simple dont le schéma est donné ci-dessous :

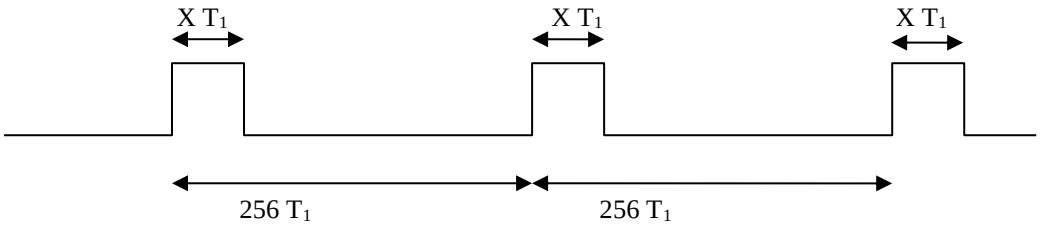


Le principe est le même que PWM (Pulse Width Modulation). C'est-à-dire que vous devrez générer le signal suivant en fonction de l'état des switch.

1- Dans un premier temps générez le signal ci-dessous (à vérifier avec l'oscillo)



2- Générez le signal ci-dessous où X est la valeur numérique représentée par les switch.



T_1 = 100 nano secondes

Les entrées de votre système sont les 8 interrupteurs de la carte. Faites sortir la sortie sur la broche G11 de FPGA (à prendre sur le connecteur D) que l'on visualisera sur l'oscilloscope.

Nom : **Prénom :**

V H D L

Durée 1H15

Programme fonctionne : +14 points
Façon de programmer (présentation, clarté, solution optimisé, commentaire, …) : +6 points
Programme ne fonctionne pas : -6 points systématiquement.

Le but de ce TP est d'afficher des motifs sur l'écran VGA.

1. Afficher une ligne vertical blanche au milieu de l'écran (la taille de l'écran est 640x480, donc quand x=320)

2. Afficher une ligne verticale blanche qui se déplace à l'écran. Il parcourt la largeur de l'écran de gauche à droite en 2 secondes et il recommence.

E.N.S.I.L **Deuxième Année ELT** **2015-2016**

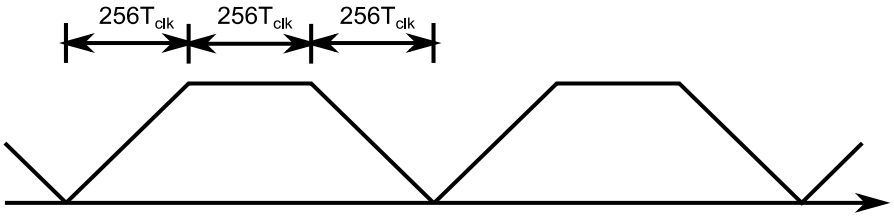
Nom : **Prénom :**

V H D L

Durée 1H15

Programme fonctionne : +16 points
Façon de programmer (présentation, clarté, solution optimisé, commentaire, …) : +4 points
Programme ne fonctionne pas : -6 points systématiquement.

Le but recherché est de générer le signal présenté ci-dessous



Pour cela, vous utiliserez un convertisseur numérique analogique que vous connecterez au connecteur d'extension JD1.

```
Net "clk" LOC=V10;

Net "JD<0>" LOC = G11;
Net "JD<1>" LOC = F10;
Net "JD<2>" LOC = F11;
Net "JD<3>" LOC = E11;
Net "JD<4>" LOC = D12;
Net "JD<5>" LOC = C12;
Net "JD<6>" LOC = F12;
Net "JD<7>" LOC = E12;
```

E.N.S.I.L **Deuxième Année ELT** **2015-2016**

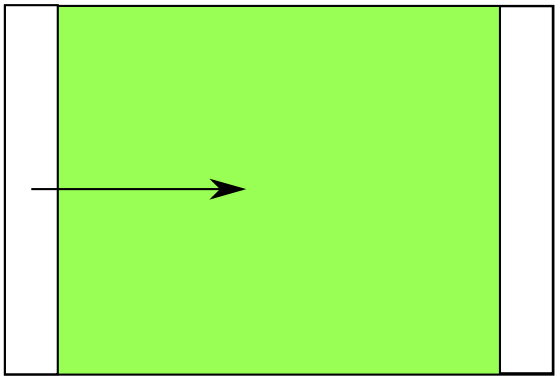
Nom : **Prénom :**

V H D L

Durée 1H15

Programme fonctionne : +16 points
Façon de programmer (présentation, clarté, solution optimisé, commentaire, …) : +4 points
Programme ne fonctionne pas : -6 points systématiquement.

1- Nous souhaitons afficher sur un écran VGA une bande blanche de largeur 16 pixels sur un fond vert.



2- Ensuite, nous souhaitons faire bouger cette bande vers la droite à raison de 100 pixels par seconde. Un fois arrivé au fond (X=640-16=624), la bande reprend son mouvement dès le début.

```
Net "clk" LOC = V10;

# VGA Connector
NET "RGB<0>" LOC = U7 ; --Rouge
NET "RGB<1>" LOC = V7 ; --Rouge
NET "RGB<2>" LOC = N7 ; --Rouge
NET "RGB<3>" LOC = P8 ; -- Vert
NET "RGB<4>" LOC = T6 ; -- Vert
NET "RGB<5>" LOC = V6 ; -- Vert
NET "RGB<6>" LOC = R7 ; -- bleu
NET "RGB<7>" LOC = T7 ; -- bleu

NET "HS" LOC = N6 ;
NET "VS" LOC = P7 ;
```

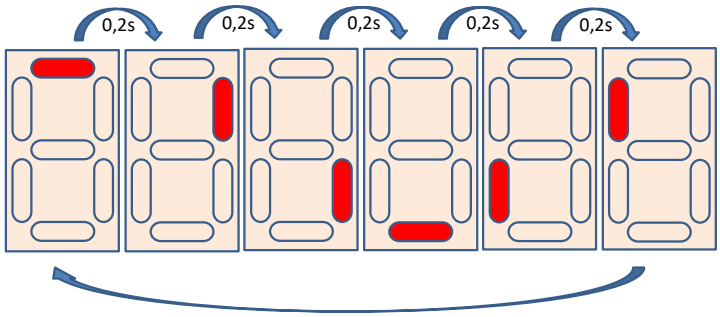
Nom : **Prénom :**

V H D L

Durée 1H15

Programme fonctionne : +16 points
Façon de programmer (présentation, clarté, solution optimisé, commentaire, …) : +4 points
Programme ne fonctionne pas : -6 points systématiquement.

1- Afficher sur chaque 7-segment de la carte le chenillard circulaire suivant :



(Pour plus de détails sur les 7-segments de la carte nexys 3, référez-vous au manuel de la carte disponible dans la salle)

2- Ajouter un switch pour pouvoir inverser le sens de chenillard.

```
Net "clk" LOC=V10 ; horloge de la carte 100 MHz

Net "seg<0>" LOC = T17 ;
Net "seg<1>" LOC = T18 ;
Net "seg<2>" LOC = U17 ;
Net "seg<3>" LOC = U18 ;
Net "seg<4>" LOC = M14 ;
Net "seg<5>" LOC = N14 ;
Net "seg<6>" LOC = L14 ;
Net "seg<7>" LOC = M13 ;

Net "an<0>" LOC = N16 ;
Net "an<1>" LOC = N15 ;
Net "an<2>" LOC = P18 ;
Net "an<3>" LOC = P17 ;


# Switche de droite
Net "sw" LOC = T10;
```