Université M’hamed BOUDIAF de M’SILA

Faculté de Technologie

Département Electronique

TP n°5 Electronique Numérique Avancée

Développement d’un premier exemple de circuit : Registre à décalage.

1. Objectifs

Les registres sont principalement utilisés pour le stockage de données.
Nous sommes tous familiers avec le rôle et l'utilisation des différents types de registres utilisés à l'intérieur d'un microprocesseur, ou d'un microcontrôleur.
Un registre est un circuit séquentiel synchrone qui est formé principalement par des ***bascules D***.
Dans ce TP, nous trouverons du code VHDL pour 4 genres de compteurs. Ces compteurs sont utilisés pour des applications différentes mais sont basés sur la même structure. Inspirez-vous de ces concepts pour générer votre propre code VHDL.

2. Rappel de notions sur les registres à décalage

Registre à décalage :
Un registre à décalage est un registre qui décale ses données à droite ou à gauche à travers des bascules mises en cascade.
Registre à décalage à droite :
C'est un registre qui décale ses données à droite lorsqu'un signal d'horloge (**CLK**) est appliqué à l'ensemble de bascules qui le constituent.
Dans un tel registre, l'état logique de la sortie ***Qn*** de la $n^{ième}$ bascule est reproduite à la sortie ***Qn+1*** de la ***(n+1)*** *ième* bascule.
Prenons un exemple d'un registre à décalage à droite, formé par 3 bascules type ***D*** qui s'activent sur front montant.
Le schéma de ce registre est donné par la **figure 1**.

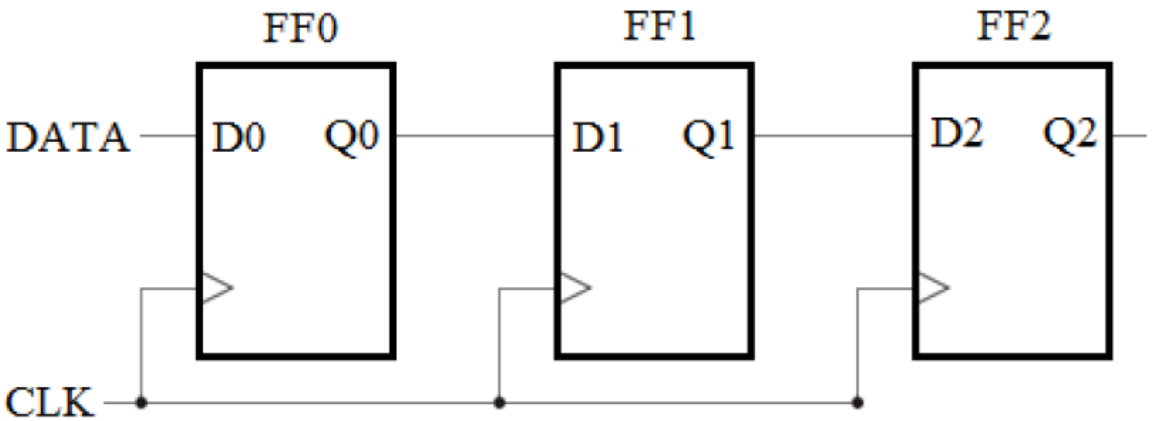


Figure 1 : Schéma d'un registre à décalage à droite.

Registre à décalage à gauche :
C'est un registre qui décale ses données à gauche lorsqu'un signal d'horloge (**CLK**) est appliqué à l'ensemble de bascules qui le constituent.
Dans un tel registre, l'état logique de la sortie ***Qn*** de la ***n****ième* bascule est reproduite à la sortie ***Qn-1*** de la ***(n-1)*** *ième* bascule.
Prenons un exemple d'un registre à décalage à gauche, formé par 4 bascules type ***D*** qui s'activent sur front montant.
Le schéma de ce registre est donné par la **figure 2**.

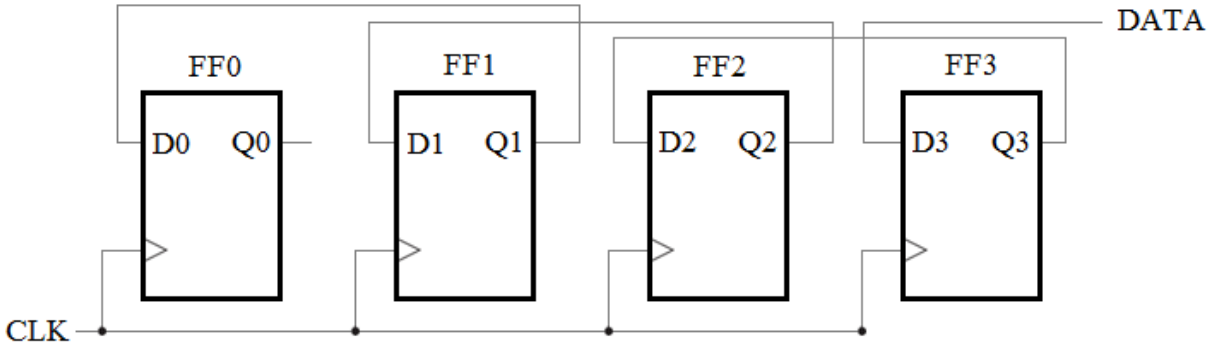


Figure 2 : Schéma d'un registre à décalage à gauche.

Registre à décalage avec mode de décalage à droite ou à gauche :
C'est un registre qui peut décaler ses données à droite ou à gauche suivant le choix du mode de décalage choisi.
On peut concevoir un tel registre en combinant les deux registres étudiés précédemment et en ajoutant une entrée supplémentaire (MODE) pour la sélection du mode de décalage tel que :

$$MODE = \begin{cases} 0 \rightarrow Décalage\ à\ droite \\ 1 \rightarrow Décalage\ à\ gauche \end{cases}$$

Le schéma de la **figure 4.45** montre un exemple d'un registre à décalage avec mode de décalage formé de quatre bascules type D qui s'activent sur front montant.
Dans la **figure 4.45**, l'entrée ***DATA_R*** représente la donnée à décaler *à droite*, et l'entrée ***DATA_L*** représente la donnée à décaler *à gauche*.

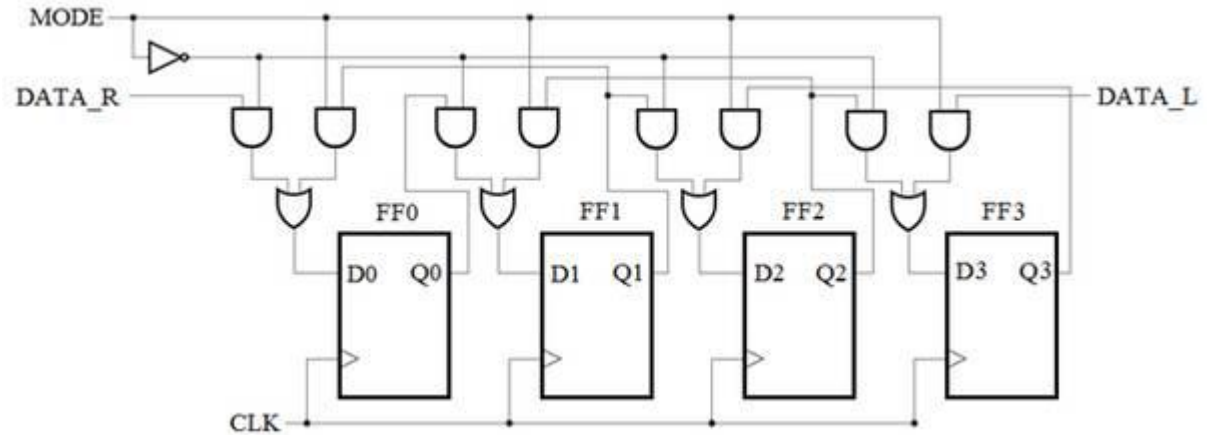


Figure 3 : Schéma d'un registre à décalage avec mode de décalage (à droite ou à gauche).

3. Codes VHDL des Registres à décalage :

a. Description VHDL d'une BASCULE D active sur front montant :

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
        entity BASCULE_D  is
                Port ( CLK_D : in  STD_LOGIC;
                       DATA_D : in  STD_LOGIC;
```

```
                        Q_D  :  out STD_LOGIC);
          end  BASCULE_D;
Architecture RTL_D of BASCULE_D is
    Signal   SIG_D    :   std_logic := '0';
         Begin
              Process (CLK_D)
        Begin
             If rising_edge (CLK_D) then
              SIG_D <= DATA_D;
         Else
           SIG_D <= SIG_D;
         End if;
     End process;
         Q_D <= SIG_D;
End  RTL_D;
```

b. Description structurelle en VHDL d'un registre à décalage à droite :

```
   library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

    entity REGISTRE is
            Port ( CLK  :  in  STD_LOGIC;
                  DATA  :  in  STD_LOGIC;
                        Q  :  out  STD_LOGIC_VECTOR (2 downto 0));
     end REGISTRE;

  Architecture RTL_REGISTRE of REGISTRE is
        Component  BASCULE_D  is
                    Port  (CLK_D   :  in std_logic;
                           DATA_D  : in std_logic;
                          Q_D          : out std_logic);
       End component;
        Signal Q0, Q1, Q2 : std_logic;
     Begin
      B1 : BASCULE_D   port map (CLK, DATA, Q0) ;
      B2 : BASCULE_D port map (CLK, Q0, Q1) ;
     B3 : BASCULE_D port map (CLK, Q1, Q2) ;
   Q(0)  <=  Q0 ;
   Q(1)  <=  Q1 ;
   Q(2)  <=  Q2 ;
 End RTL_REGISTRE ;
```

c. Description comportementale en VHDL d'un registre à décalage à droite :

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity REGISTRE_COMP is
   Port ( CLK : in  STD_LOGIC;
         DATA_in : in  STD_LOGIC;
         DATA_out : out  STD_LOGIC);
end REGISTRE_COMP;
```

```
architecture Behavioral of REGISTRE_COMP is
signal SIG_REG : std_logic_vector (2 downto 0) := "000";

begin
process (CLK)
 begin
      if rising_edge(CLK) then
            SIG_REG <= DATA_in & SIG_REG(SIG_REG'length-1 downto 1);
                                          else
                                          SIG_REG <= SIG_REG;
      end if;
   end process;
     DATA_out <= SIG_REG(0);
end Behavioral;
```

La donnée **DATA** est transmise à la première bascule (**FF0**) après quatre impulsions d'horloge (**CLK**).

d. Description structurelle en VHDL d'un registre à décalage à gauche :

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;


entity REGISTRE_GAUCHE_STRUCT is
   Port ( CLK : in  STD_LOGIC;
          DATA : in  STD_LOGIC;
          Q : out  STD_LOGIC_VECTOR (3 downto 0));
end REGISTRE_GAUCHE_STRUCT;

architecture Behavioral of REGISTRE_GAUCHE_STRUCT is


    Component  BASCULE_D  is
Port  (CLK_D   :  in std_logic;
          DATA_D  : in std_logic;
         Q_D         : out std_logic);
End component;
Signal Q0, Q1, Q2, Q3 : std_logic;
Begin
   B1 : BASCULE_D port map (CLK, Q1, Q0) ;
   B2 : BASCULE_D port map (CLK, Q2, Q1) ;
   B3 : BASCULE_D port map (CLK, Q3, Q2) ;
   B4 : BASCULE_D port map (CLK, DATA, Q3) ;
  Q(0)  <=  Q0 ;
  Q(1)  <=  Q1 ;
  Q(2)  <=  Q2 ;
  Q(3)  <=  Q3 ;
 End Behavioral;
```

e. Description comportementale en VHDL d'un registre à décalage à gauche.

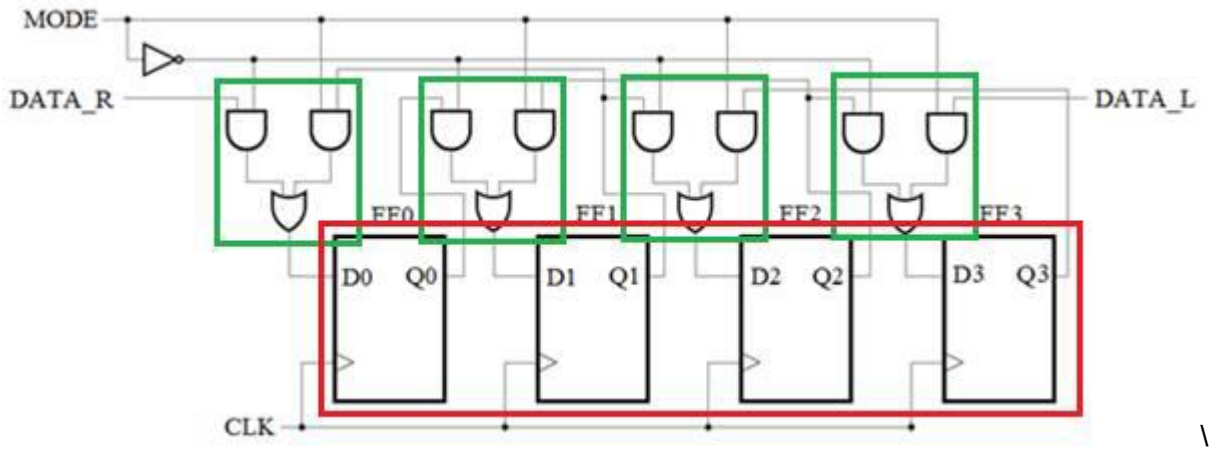
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity REGISTRE_GAUCHE_COMP is
    Port ( CLK : in  STD_LOGIC;
           DATA : in  STD_LOGIC;
           Q : out  STD_LOGIC_VECTOR (3 downto 0));
end REGISTRE_GAUCHE_COMP;

architecture Behavioral of REGISTRE_GAUCHE_COMP is
signal SIG : std_logic_vector (0 to 3) := "0000";
begin
  process(CLK)
            begin
            if rising_edge (CLK) then
                SIG <=  SIG(1 to 3 ) & DATA   ;
     end if;
           end process;
           Q <= SIG;

end Behavioral;
```

Nous allons déduire le VHDL du registre à décalage avec mode de décalage avec les deux styles de descriptions, structurelle et comportementale à partir de la figure 4.
Distinguons d'abord les composants à modéliser pour le style de *description structurelle*
Figure 4. Les composants utilisés dans la description VHDL avec le style structurel du registre à décalage de la *figure 4.*



\

Figure 4 : Les composants utilisés dans la description VHDL avec le style structurel du registre à décalage

Le bloc en rouge de la **figure 4** représente les bascules type ***D*** qu'on a déjà modélisées.
On remarque que le bloc en vert de la **figure 4** possède *quatre entrées*, et *une sortie*.
Les opérations effectuées au niveau de ce bloc sont basiques (**and**, **or**) ; modélisons-le en VHDL avec le style de description : *flot de données*.

f. Description VHDL du BLOC MODE

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity BLOC_MODE is
    Port ( E1 : in  STD_LOGIC;
           E2 : in  STD_LOGIC;
           E3 : in  STD_LOGIC;
           E4 : in  STD_LOGIC;
           SORTIE : out  STD_LOGIC);
end BLOC_MODE;

architecture Behavioral of BLOC_MODE is
signal SIG_1, SIG_2 : std_logic;
begin
  SIG_1 <= E1 and E2;
  SIG_2 <= E3 and E4;
    SORTIE <= SIG_1 or SIG_2;

end Behavioral;
```

g. Description structurelle en VHDL du registre à décalage de la *figure 4* :

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity REGISTRE_MODE_0_1 is
    Port ( CLK : in  STD_LOGIC;
           DATA_R : in  STD_LOGIC;
           DATA_L : in  STD_LOGIC;
           MODE : in  STD_LOGIC;
           Q : out  STD_LOGIC_VECTOR (3 downto 0));
end REGISTRE_MODE_0_1;

architecture Behavioral of REGISTRE_MODE_0_1 is
component BLOC_MODE is
    Port ( E1, E2, E3, E4 : in  STD_LOGIC;
           SORTIE : out  STD_LOGIC);
end component;
component BASCULE_D  is
                    Port ( CLK_D : in  STD_LOGIC;
                              DATA_D : in  STD_LOGIC;
                           Q_D  :  out STD_LOGIC);
  end  component;
  signal D0, D1, D2, D3 : std_logic;
  signal Q0, Q1, Q2, Q3 : std_logic;
  signal NMOD           : std_logic;
begin
-- Instantiation des BLOCs Mode
BM1 : BLOC_MODE port map (DATA_R, NMOD, MODE, Q1, D0);
BM2 : BLOC_MODE port map (Q0, NMOD, MODE, Q2, D1);
BM3 : BLOC_MODE port map (Q1, NMOD, MODE, Q3, D2);
BM4 : BLOC_MODE port map (Q2, NMOD, MODE, DATA_L, D3);
-- Instantiation des bascules
BD1 : BASCULE_D port map (CLK, D0, Q0);
```

```
BD2 :  BASCULE_D port map (CLK, D1, Q1);
BD3 : BASCULE_D port map (CLK, D2, Q2);
BD4 : BASCULE_D port map (CLK, D3, Q3);
NMOD <= not MODE;
Q(0) <= Q0;
Q(1) <= Q1;
Q(2) <= Q2;
Q(3) <= Q3;
 end Behavioral;
```

h. Description comportementale en VHDL du registre à décalage de la figure 4 :

```
library ieee;
use ieee.std_logic_1164.all;
entity REG_MODE01_COMP is
      port (CLK, MODE, DATA_R, DATA_L : in std_logic;
                        Q                    : out std_logic_vector(3 downto 0));
end REG_MODE01_COMP;

architecture Behav of REG_MODE01_COMP is
  begin
    process(CLK, MODE)
            variable var : std_logic_vector (3 Downto 0):= "0000";
          begin
            if (MODE = '0') then
                    if (CLK'event and CLK = '1') then
                            var := var(2 downto 0) & DATA_R;
                        end if;
                elsif (MODE = '1') then
                if (CLK'event and CLK = '1') then
                    var :=  DATA_L & var(3 downto 1);
     end if;
                end if;
        Q <= var;
        end process;
        end Behav;
```

4. Manipulations : Utilisez la plate-forme ISE 8.2 de Xilinx

- Simuler la bascule D active sur front montant
- Simuler un registre à décalage à droite en utilisant la description VHDL structurelle.
- Simuler un registre à décalage à droite en utilisant la description VHDL comportementale.
- Simuler un registre à décalage à gauche en utilisant la description VHDL structurelle.
- Simuler un registre à décalage à gauche en utilisant la description VHDL comportementale.
- Simuler un registre avec mode de décalage à gauche ou à droite en utilisant la description VHDL structurelle.
- Simuler un registre avec mode de décalage à gauche ou à droite en utilisant la description VHDL comportementale.

Donnez pour chaque cas les résultats de simulations ainsi que la consommation de ressources en termes de cellules, bascules D, tables LUT, horloges et entrées/sorties.