TD3 : Conception d'un registre à décalage SIPO de N bits

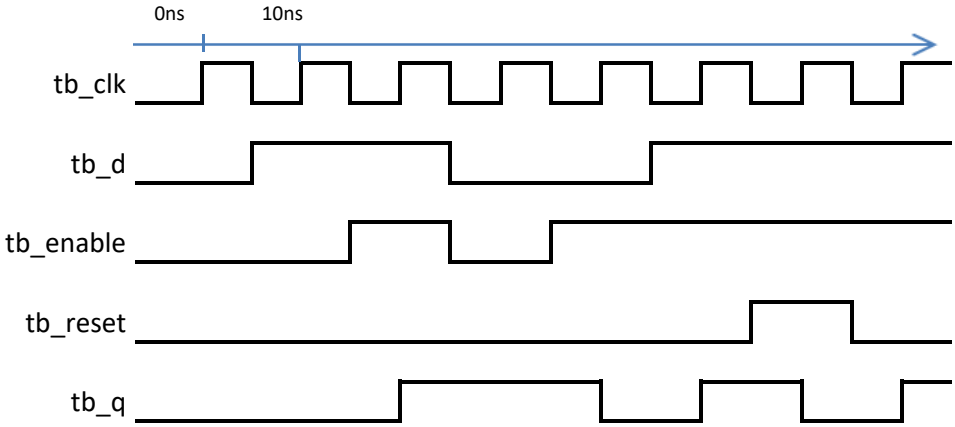
TD – CE312

Ce TD a pour but de vous familiariser avec :

- Les testbenches
- La description structurelle
- Les *generics*
- La structure *for…generate*

```
entity bascule is
    port(d, clk, reset, enable : in  std_logic ;
                 q             : out std_logic);
end entity bascule;
```

1. On dispose de la description VHDL d'une bascule D (flip-flop) dont l'entité est donnée ci-dessus. Ecrivez le testbench de ce composant de telle manière qu'il corresponde au chronogramme ci-dessous.




```
library IEEE;
use IEEE. std_logic_1164.all;

entity tb_bascule is
end tb_bascule;

architecture bhv of tb_bascule is
signal tb_clk : std_logic := '1';
signal tb_d, tb_enable, tb_reset, tb_q : std_logic := '0';
component bascule
port(d, clk, reset, enable : in  std_logic ;
        q                  : out std_logic);
end component;
```

```
begin
i1:bascule
port(tb_d,tb_clk, tb_reset, tb_enable, tb_q) ;
```

```
tb_clk <= not(tb_clk) after 5 ns;
tb_d <= '0', '1' after 5 ns, '0' after 25 ns, '1' after 45 ns;
tb_enable <= '0', '1' after 15 ns, '0' after 25 ns, '1' after 35 ns;
tb_reset <= '0', '1' after 55 ns, '0' after 65 ns;
end architecture;
```

2. On veut construire un **registre à décalage de 2 bits SIPO (Serial In – Parallel Out)** en utilisant uniquement 2 bascules D.

Ce registre à décalage comporte :

- un simple bit d'entrée (din)
- une entrée d'autorisation de décalage (enable) active à l'état haut
- une sortie parallèle (dout) sur 2 bits
- une remise à zéro (reset) synchrone active à l'état haut.

2.1. Dessinez le schéma de ce registre

2.2. Ecrivez la description structurelle du registre SIPO (entité + architecture)

```
library IEEE;
use IEEE. std_logic_1164.all;

entity SIPO is
port( din, enable, clk, reset : in std_logic;
      dout : out std_logic_vector(0 to 1);
end SIPO;

architecture struct of SIPO is
component bascule
port(d, clk, reset, enable : in  std_logic ;
     q                     : out std_logic);
end component;

signal dout_temp : std_logic_vector(0 to 1);

begin
```

```
i1: bascule
port(din, enable, clk, reset, dout_temp(0));
```

```
i2: bascule
port(dout_temp(0), enable, clk, reset, dout_temp(1));
dout<=dout_temp ;
end architecture;
```

3. Pour que le registre soit facilement réutilisable dans différents projets, on souhaite le rendre plus générique. On veut qu'il soit paramétrable pour supporter différentes tailles de vecteurs de sortie. On veut donc construire un **registre à décalage SIPO de N bits.**

   3.1. Ecrivez l'entité de ce registre

```
library IEEE;
use IEEE. std_logic_1164.all;

entity SIPO_n is
generic(N : integer :=2);
port( din, enable, clk, reset : in std_logic;
      dout : out std_logic_vector(0 to N-1);
end SIPO;
```

   3.2. Ecrivez l'architecture de ce registre

```
architecture struct of SIPO_n is
component bascule
port(d, clk, reset, enable : in  std_logic ;
     q                     : out std_logic);
end component;

signal dout_temp : std_logic_vector(0 to N);

begin
```

```
for I in 0 to N-1 loop
i: bascule
port(dout_temp(i), enable, clk, reset, dout_temp(i+1));
end loop;
dout_temp(0) <= din;
dout<=dout_temp(1 to N);
```

```
end architecture;
```

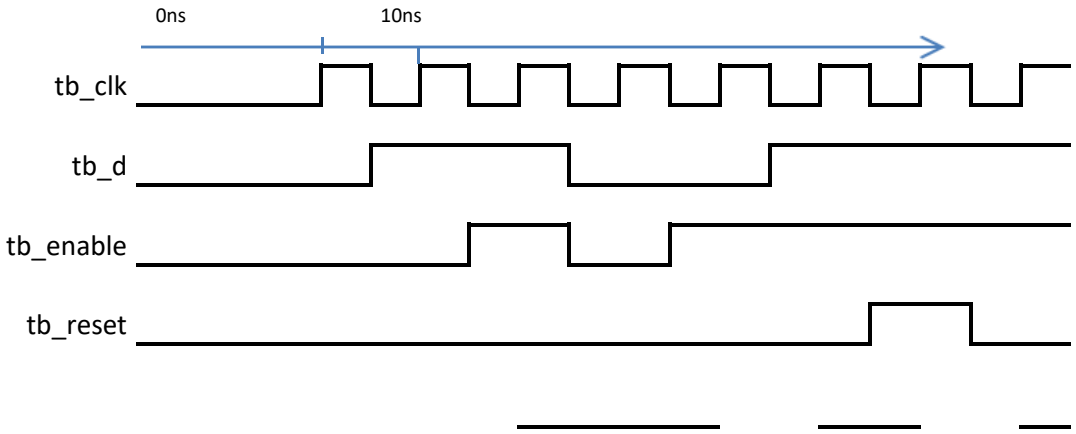
4. Ecrivez un testbench pour tester le registre SIPO avec 5 bits de sortie.

```
library IEEE;
use IEEE. std_logic_1164.all;

entity tb_SIPO_n is
end tb_SIPO_n;

architecture bhv of tb_SIPO_n is
signal tb_clk : std_logic := '1';
signal tb_din, tb_enable, tb_reset: std_logic;
signal tb_dout: std_logic_vector(0 to 4);

component SIPO_n
generic(N : integer :=2);
 port(  din, enable, clk, reset : in std_logic;
        dout : out std_logic_vector(0 to N-1);
end component;
```

```
begin
i1:SIPO_n
generic map(N=>5)
port(tb_din, tb_enable, tb_clk, tb_reset, tb_dout) ;

tb_clk <= not(tb_clk) after 5 ns;
tb_din <= '0', '1' after 5 ns, '0' after 25 ns, '1' after 45 ns;
tb_enable <= '0', '1' after 15 ns, '0' after 25 ns, '1' after 35 ns;
tb_reset <= '0', '1' after 55 ns, '0' after 65 ns;
end architecture;
```

5. Dessinez le chronogramme correspondant à votre testbench

Remarque : Pensez à indiquer clairement l'échelle des temps et les temps correspondant à chaque évènement sur l'axe temporel



A finir