Ecole d'ingénieurs Sup Galilée

Spécialité Instrumentation

TP projet FPGA 3

Réalisation des dispositifs (RPPE)

simulation numérique

Carte électronique moteur pas à pas

Sommaire

Simulation fonctionnelle des principaux modules VHDL

1. Gestion horloge : stimuli et test bench
2. State machine moteur pas à pas : stimuli et test bench
3. Encodeur numérique: stimuli et test bench
4. Liaison RX moteur: stimuli et test bench
5. Multiplexage-décodage : stimuli et test bench

Ecriture des stimuli test bench : **gestion d'horloge moteur pas à pas**

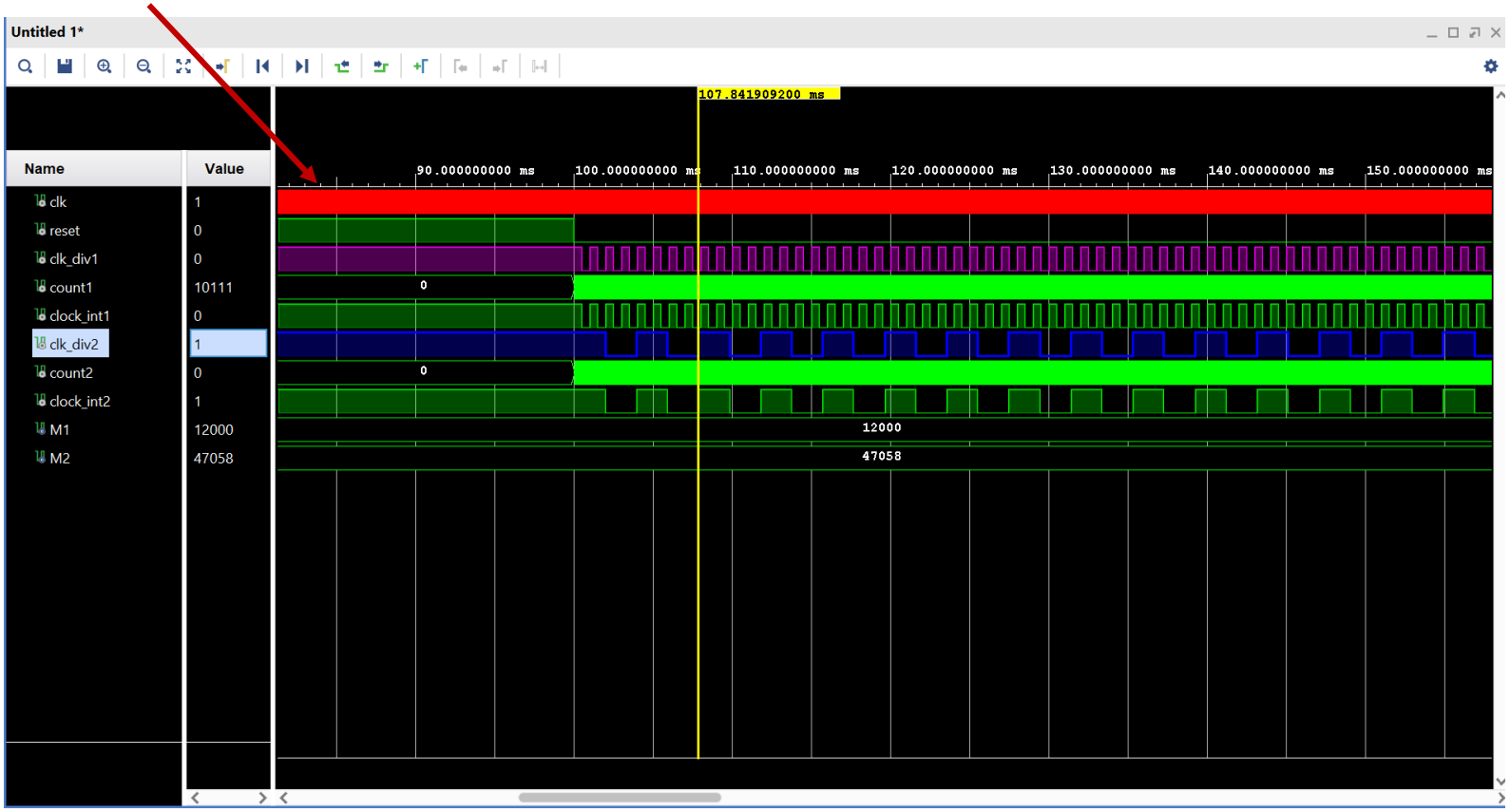
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity gestion_horloge_tbw is
--  Port ( );
end gestion_horloge_tbw;

architecture Behavioral of gestion_horloge_tbw is

component clock_manager_CmodA7
      port ( clk : in STD_LOGIC;   --12MHz
             reset : in STD_LOGIC;
             clk_div1 : out STD_LOGIC;  --1000Hz
             clk_div2 : out STD_LOGIC);  --255Hz
end component;

    signal clk : std_logic;
    signal reset : std_logic;
    signal clk_div1: STD_LOGIC;
    signal clk_div2 : STD_LOGIC;

begin
```

Déclaration du composant

```
-- Instantiate the Unit Under Test (UUT)
  uut: clock_manager_CmodA7 PORT MAP (
        clk => clk,
        reset => reset,
        clk_div1 => clk_div1,
        clk_div2 => clk_div2
        );

  --generation d'un signal d'horloge @ 12MHz
  clkgen : process
  begin
  clk <='1';
  wait for 41.66ns;
  clk <='0';
  wait for 41.66ns;
  end process;

  resetgen : process
  begin
  reset <='1';
  wait for 100ms;
  reset <='0';
  wait;
  end process;
  end Behavioral;
```

Fichier de simulation

Test bench : gestion d'horloge moteur pas à pas



Fabrice Wiotte LPL

Ecriture des stimuli test bench : **gestion state machine moteur pas à pas**

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity state_machine_moteur_tbw is
--  Port ( );
end state_machine_moteur_tbw;
architecture Behavioral of state_machine_moteur_tbw is

component state_machine_stepper_motor
Port (
        clk : in  STD_LOGIC;  --255Hz
        bouton_UP : in  STD_LOGIC;
        bouton_DOWN : in  STD_LOGIC;
        change_frequency_motor : in  STD_LOGIC_VECTOR (7 downto 0);
        Commandes_demi_pas : out  STD_LOGIC_VECTOR (3 downto 0);
        vers_change_frequency_motor : out  STD_LOGIC_VECTOR (7 downto 0);
        reset : in  STD_LOGIC;
        visu_UP : out  STD_LOGIC;
        visu_DOWN : out  STD_LOGIC);
end component;
```

```
    signal clk : std_logic;
    signal reset : std_logic;
    signal visu_UP : std_logic;
    signal visu_DOWN : std_logic;
    signal bouton_UP : std_logic;
    signal bouton_DOWN: STD_LOGIC;
    signal change_frequency_motor : STD_LOGIC_VECTOR (7 downto 0);
    signal Commandes_demi_pas : STD_LOGIC_VECTOR (3 downto 0);
    signal vers_change_frequency_motor : STD_LOGIC_VECTOR (7 downto 0);

begin

    -- Instantiate the Unit Under Test (UUT)
       uut: state_machine_stepper_motor PORT MAP (
       clk => clk,
       reset => reset,
       visu_UP => visu_UP,
       visu_DOWN => visu_DOWN,
       bouton_UP => bouton_UP,
       bouton_DOWN => bouton_DOWN,
       change_frequency_motor => change_frequency_motor,
       Commandes_demi_pas => Commandes_demi_pas,
       vers_change_frequency_motor => vers_change_frequency_motor
       );
```

Fichier de simulation

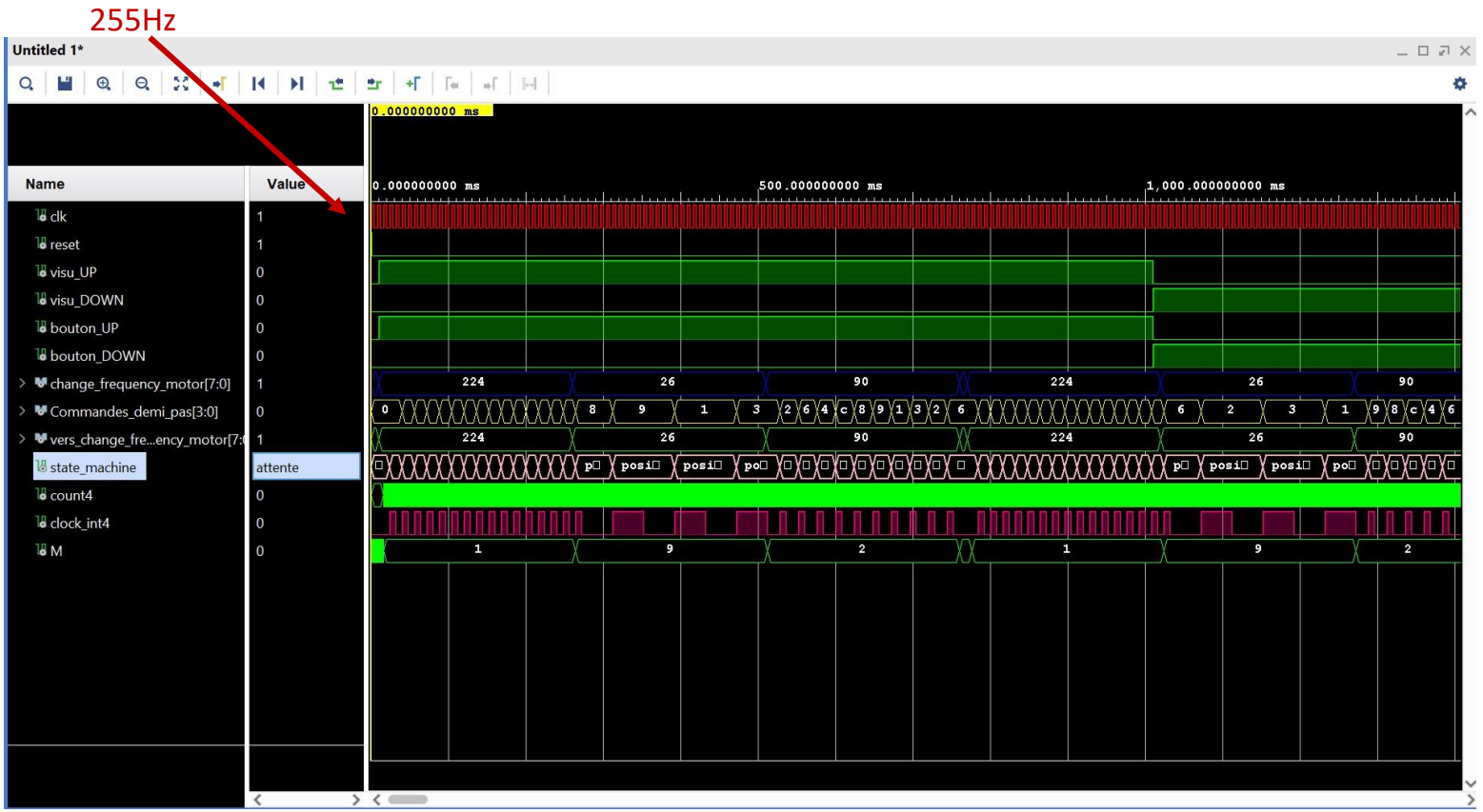
Ecriture des stimuli test bench : **gestion state machine moteur pas à pas**

```
--generation d'un signal d'horloge @ 255Hz
  clkgen : process
  begin
  clk <='1';
  wait for 4ms;
  clk <='0';
  wait for 4ms;
  end process;

  resetgen : process
  begin
  reset <='1';
  wait for 1ms;
  reset <='0';
  wait;
  end process;
```

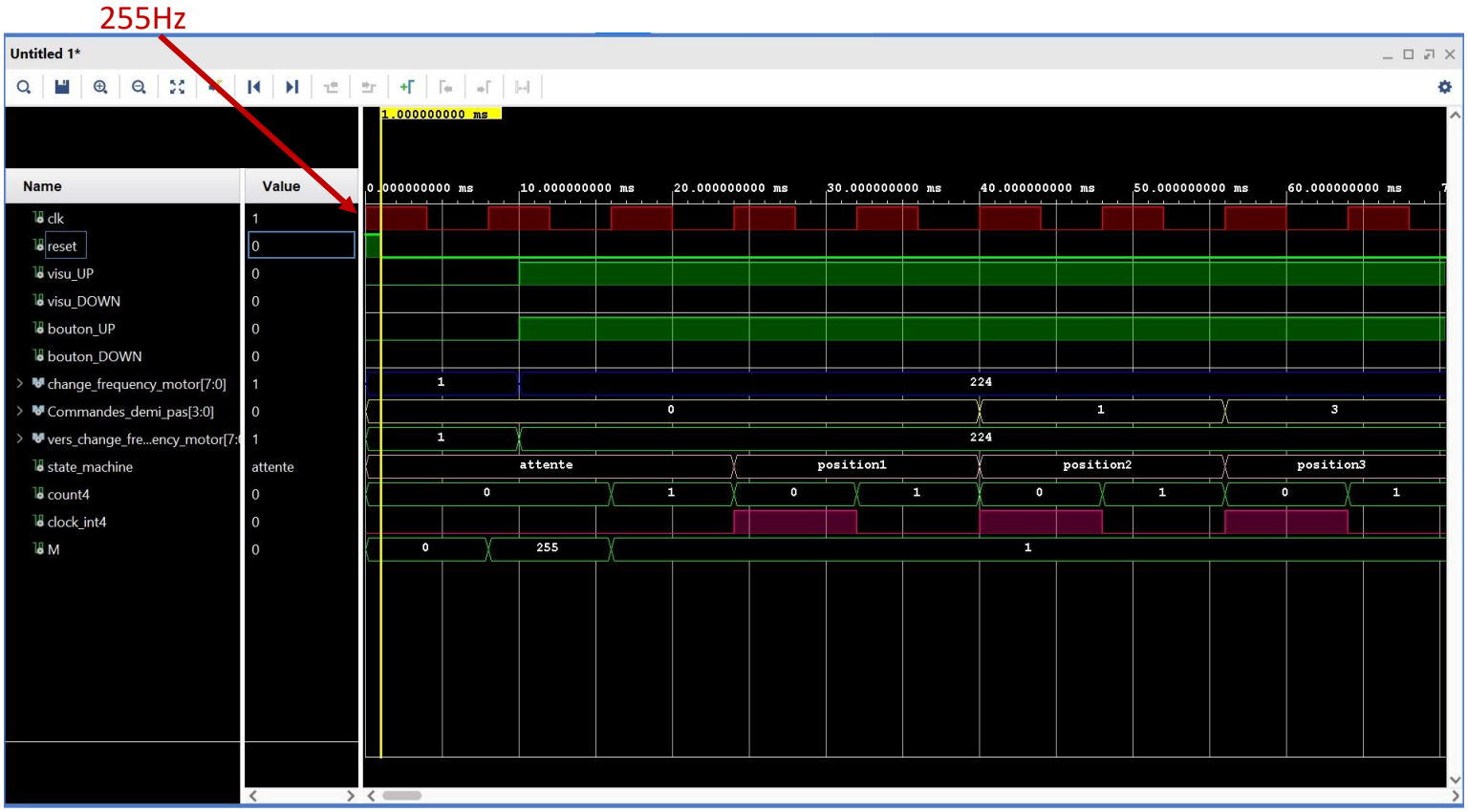
```
buttonupgen : process
  begin
  bouton_UP <='0';
  bouton_DOWN <='0';
  wait for 10ms;
  bouton_UP <='1';
  bouton_DOWN <='0';
  wait for 1000ms;
  bouton_DOWN <='1';
  bouton_UP <='0';
  wait for 1000ms;
  end process;

  change_frequency_motorgen : process
  begin
  change_frequency_motor <="00000001";
  wait for 10ms;
  change_frequency_motor <="11100000";
  wait for 250ms;
  change_frequency_motor <="00011010";
  wait for 250ms;
  change_frequency_motor <="01011010";
  wait for 250ms;
  end process;
  end Behavioral;
```

Fichier de simulation

test bench : gestion state machine moteur pas à pas



test bench : gestion state machine moteur pas à pas



Ecriture des stimuli test bench : **encodeur numérique moteur pas à pas**

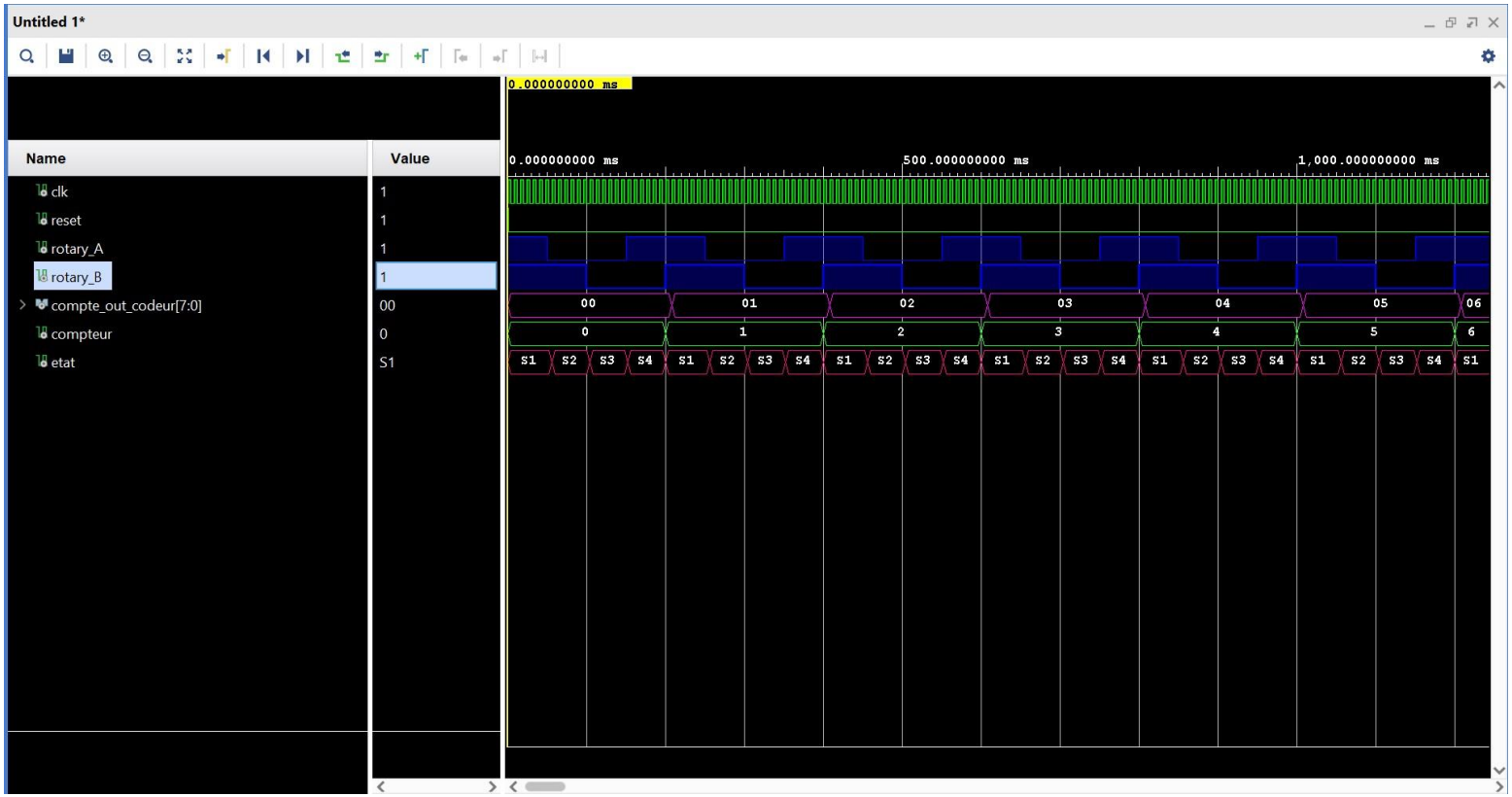
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity codeur_num_tbw is
--  Port ( );
end codeur_num_tbw;

architecture Behavioral of codeur_num_tbw is
 component codeur_num
    Port (CLK : in  STD_LOGIC; --255Hz
          reset: in  STD_LOGIC;
          rotary_A : in  STD_LOGIC;
          rotary_B : in  STD_LOGIC;
          compte_out_codeur : out STD_LOGIC_VECTOR(7 downto 0));
end component;

   signal clk : std_logic;
   signal reset : std_logic;
   signal rotary_A: STD_LOGIC;
   signal rotary_B : STD_LOGIC;
   signal compte_out_codeur : STD_LOGIC_VECTOR(7 downto 0);
begin
 -- Instantiate the Unit Under Test (UUT)
  uut: codeur_num PORT MAP (
       clk => clk,
       reset => reset,
       rotary_A => rotary_A,
       rotary_B => rotary_B,
       compte_out_codeur => compte_out_codeur
       );
```

```
 --generation d'un signal d'horloge @ 255Hz
clkgen : process
begin
clk <='1';
wait for 4ms;
clk <='0';
wait for 4ms;
end process;
resetgen : process
begin
reset <='1';
wait for 1ms;
reset <='0';
wait;
end process;
rotary_Agen : process
begin
rotary_A <='1';
wait for 100ms;
rotary_A <='0';
wait for 100ms;
end process;
rotary_Bgen : process
begin
rotary_B <='1';
wait for 50ms;
rotary_B <='0';
wait for 100ms;
rotary_B <='1';
wait for 50ms;
end process;
end Behavioral;
```

Fichier de simulation

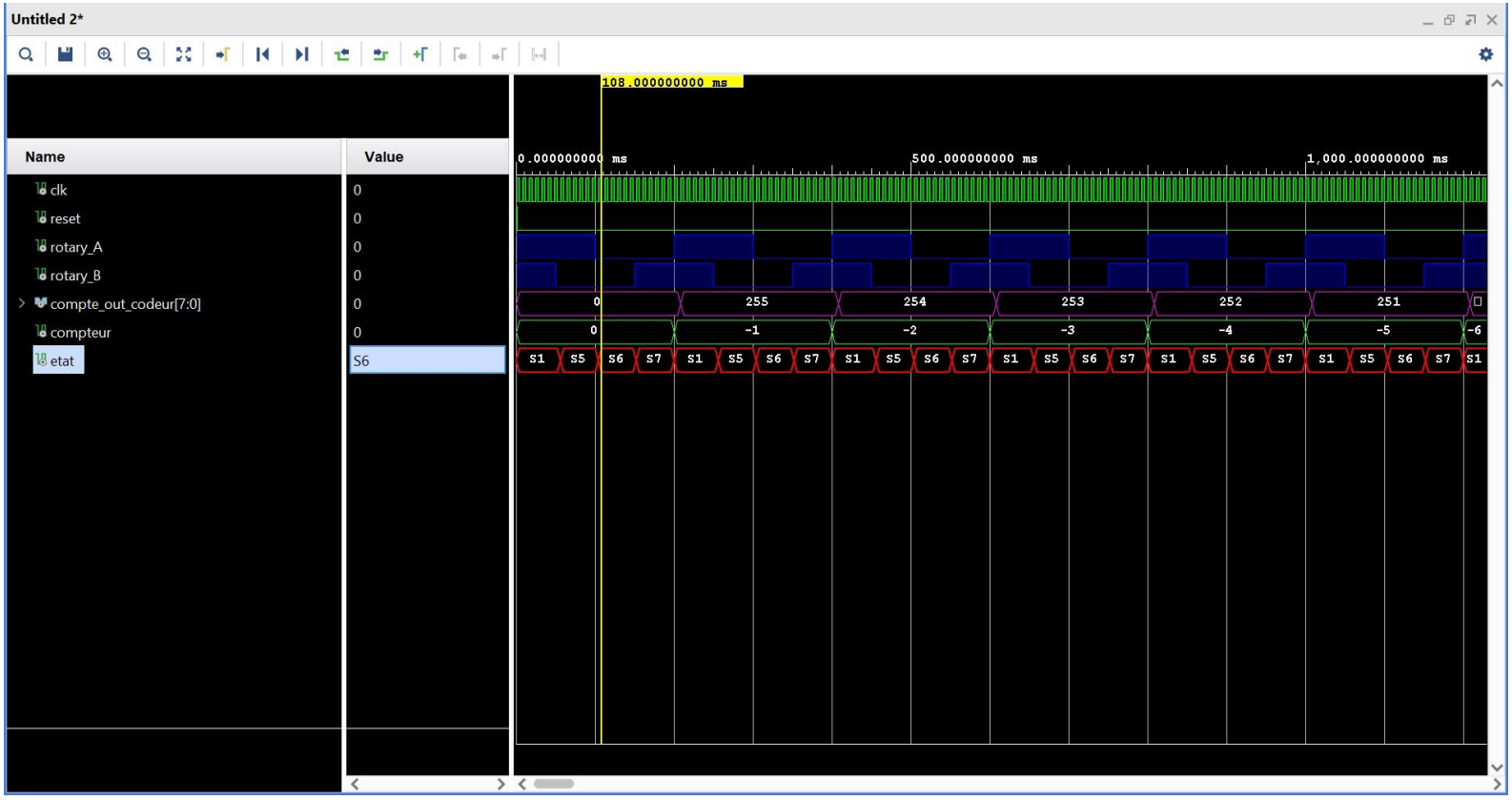
test bench : **encodeur numérique moteur pas à pas**

Rotation CW



test bench : **encodeur numérique moteur pas à pas**

Rotation CCW



Ecriture des stimuli test bench : **liaison RX moteur pas à pas**

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;

entity UART_rx_tbw is
--  Port ( );
end UART_rx_tbw;

architecture Behavioral of UART_rx_tbw is

component UART_RX
port (
   clk        : in  std_logic;
   RX_Serial   : in  std_logic;
   data_send   : out std_logic;
   RX_Byte     : out std_logic_vector(7 downto 0)
                 );
end component;

    -- Test Bench uses a 12MHz Clock
    -- Want to interface to 9600 baud UART
    -- 12000000 / 9600 = 1250 Clocks Per Bit.
    constant c_CLKS_PER_BIT : integer := 1250;
    constant c_BIT_PERIOD : time := 104.16 us; --9600bps
```

```
 signal RX_Byte : STD_LOGIC_VECTOR (7 downto 0);
 signal clk : std_logic;
 signal data_send : std_logic;
 signal RX_Serial : std_logic;

 -- Low-level byte-write
 procedure UART_WRITE_BYTE (
 data_in       : in  std_logic_vector(7 downto 0);
 signal serial  : out std_logic)
 is
 begin
-- Send Start Bit
serial <= '0';
wait for c_BIT_PERIOD;

-- Send Data Byte
 for i in 0 to 7 loop
  serial <= data_in(i);
  wait for c_BIT_PERIOD;
  end loop;

-- Send Stop Bit
  serial <= '1';
  wait for c_BIT_PERIOD;
  end UART_WRITE_BYTE;
```

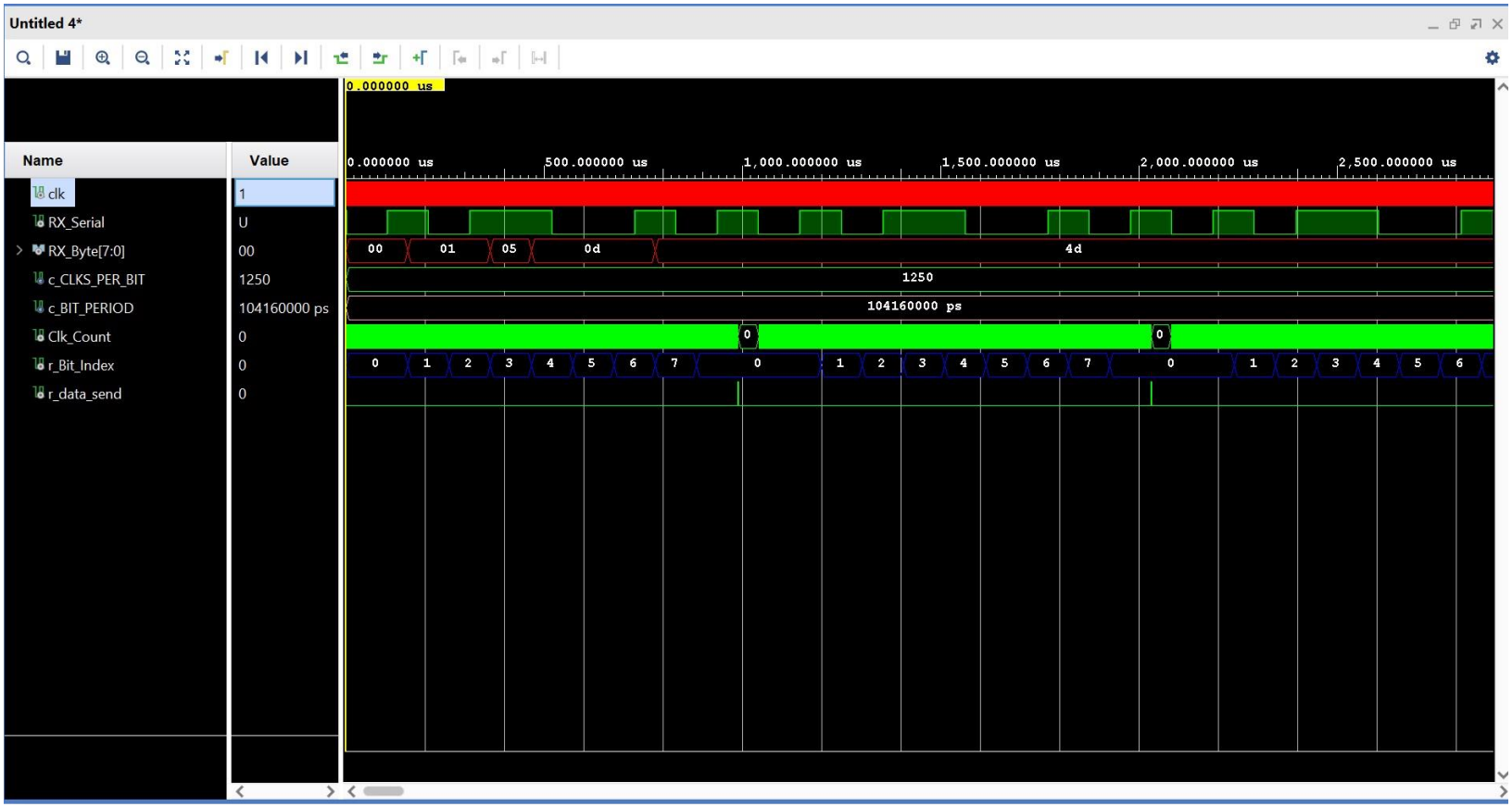
Fichier de simulation

Ecriture des stimuli test bench : liaison RX moteur pas à pas

```
begin
 -- Instantiate the Unit Under Test (UUT)
        uut: UART_RX PORT MAP (
        RX_Byte => RX_Byte,
        data_send => data_send,
        RX_Serial => RX_Serial,
        clk => clk
        );

    --generation d'un signal d'horloge @ 12MHz
   clkgen : process
   begin
   clk <='1';
   wait for 41.66ns;
   clk <='0';
   wait for 41.66ns;
   end process;

   process
   begin
   -- Send a command to the UART
   wait until rising_edge(clk);
   UART_WRITE_BYTE(X"4D", RX_Serial);
   wait until rising_edge(clk);
   end process;
   end Behavioral;
```

Fichier de simulation

test bench : **liaison RX moteur pas à pas**



Ecriture des stimuli test bench : **multiplexage décodage**

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

entity mux_decode_tbw is
end mux_decode_tbw;

architecture Behavioral of mux_decode_tbw is
component mux_decode

Port (  A : in  STD_LOGIC_VECTOR (3 downto 0);
        B : in  STD_LOGIC_VECTOR (3 downto 0);
        C : in  STD_LOGIC_VECTOR (3 downto 0);
        D : in  STD_LOGIC_VECTOR (3 downto 0);
        SEL : in  STD_LOGIC_VECTOR (1 downto 0);
        afficheur_0 : out  STD_LOGIC;
        afficheur_1 : out  STD_LOGIC;
        afficheur_2 : out  STD_LOGIC;
        afficheur_3 : out  STD_LOGIC;
        sortie_mux : out  STD_LOGIC_VECTOR (3 downto 0)
      );
end component;
   signal SEL : STD_LOGIC_VECTOR (1 downto 0):= "00";
   signal A : STD_LOGIC_VECTOR (3 downto 0):= "0000";
   signal B : STD_LOGIC_VECTOR (3 downto 0):= "0000";
   signal C : STD_LOGIC_VECTOR (3 downto 0):= "0000";
   signal D : STD_LOGIC_VECTOR (3 downto 0):= "0000";
   signal sortie_mux : STD_LOGIC_VECTOR (3 downto 0):= "0000";
   signal afficheur_0 : STD_LOGIC;
   signal afficheur_1 : STD_LOGIC;
   signal afficheur_2 : STD_LOGIC;
   signal afficheur_3 : STD_LOGIC;
```

```
begin
-- Instantiate the Unit Under Test (UUT)
      uut: mux_decode PORT MAP (
      SEL => SEL,
      A => A,
      B => B,
      C => C,
      D => D,
      afficheur_0 => afficheur_0,
      afficheur_1 => afficheur_1,
      afficheur_2 => afficheur_2,
      afficheur_3 => afficheur_3,
      sortie_mux => sortie_mux
      );

   stim_proc: process
   begin
   A <= "0011";
   B <= "1100";
   C <= "0101";
   D <= "1111";
   SEL <="00";
   wait for 10ms;
   SEL <="01";
    wait for 10ms;
   SEL <="10";
    wait for 10ms;
   SEL <="11";
   wait for 10ms;
   end process;
   end Behavioral;
```

Fichier de simulation

test bench : **multiplexage décodage**