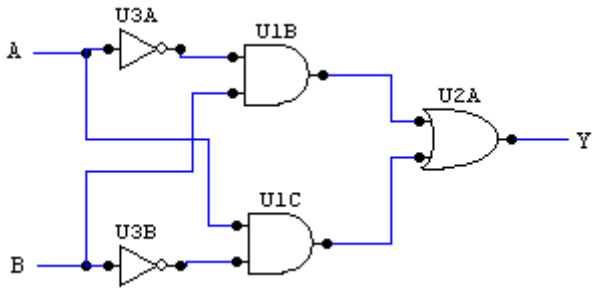
# Lab. Script 9

## VHDL

### COMBINATORIAL CIRCUITS

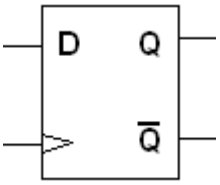
Observe the following combinatorial circuit composed only by elemental logic gates:




a) Implement the circuit in VHDL.
b) Establish a "*testbench*" to the circuit and simulate its response forcing all the possible combinations to its input.
c) Establish an alternative architecture to the circuit that emulates only its behaviour (suggestion: use the reserved word WHEN)

### SEQUENTIAL CIRCUITS

Using VHDL draw an architecture that describes the behaviour of a D flip-flop active to the ascending edge.

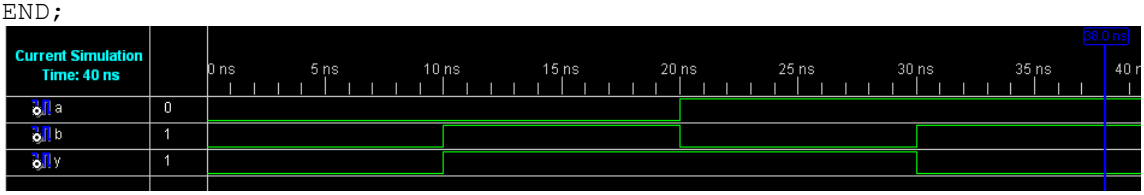


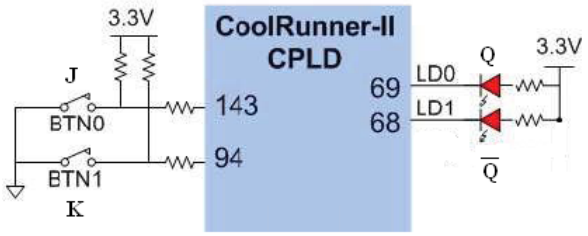

Write a "testbench" to the previous circuit and simulate its operation.

RESOLUTION

```
-- CIRCUIT ------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity circuitoxor is
    Port ( A : in  STD_LOGIC;
           B : in  STD_LOGIC;
           Y : out  STD_LOGIC);
end circuitoxor;

architecture Behavioral of circuitoxor is

begin

Y<=(A and not B) or (not A and B);

end Behavioral;

-- TESTBENCH ----------------------------------------------------------
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.std_logic_unsigned.all;
USE ieee.numeric_std.ALL;

ENTITY testbench IS
END testbench;

ARCHITECTURE behavior OF testbench IS

    -- Component Declaration for the Unit Under Test (UUT)

    COMPONENT circuitoxor
    PORT(
         A : IN  std_logic;
         B : IN  std_logic;
         Y : OUT  std_logic
        );
    END COMPONENT;


   --Inputs
   signal A : std_logic := '0';
   signal B : std_logic := '0';

      --Outputs
   signal Y : std_logic;

BEGIN

      -- Instantiate the Unit Under Test (UUT)
   uut: circuitoxor PORT MAP (
          A => A,
          B => B,
          Y => Y
```

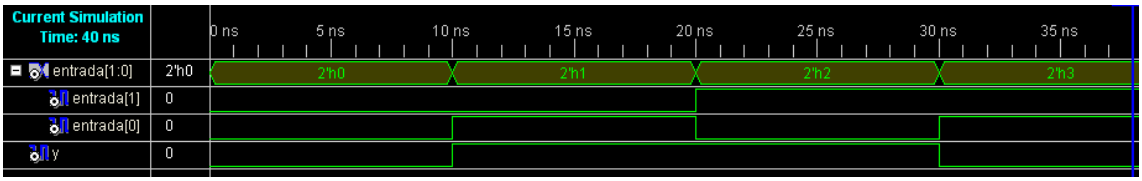
```
        );

   A<='0', '1' after 20 ns, '0' after 40 ns;
   B<='0', '1' after 10 ns,'0' after 20 ns,'1' after 30 ns;

END;
```




```
-- CIRCUIT - Behaviour --------------------------------

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity circuitoXor is
    Port ( entrada : in  STD_LOGIC_VECTOR (1 downto 0);
           Y : out  STD_LOGIC);
end circuitoXor;

architecture Behavioral of circuitoXor is

begin

Y<= '0' WHEN entrada="00" ELSE
       '1' WHEN entrada="01" ELSE
       '1' WHEN entrada="10" ELSE
       '0';
end Behavioral;



-- TESTBENCH ------------------------------------------------------

LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.std_logic_unsigned.all;
USE ieee.numeric_std.ALL;

ENTITY testbench IS
END testbench;

ARCHITECTURE behavior OF testbench IS
```

```
    COMPONENT circuitoXor
    PORT(
         entrada : IN  std_logic_vector(1 downto 0);
         Y : OUT  std_logic
        );
    END COMPONENT;


   --Inputs
   signal entrada : std_logic_vector(1 downto 0) := (others => '0');

   --Outputs
   signal Y : std_logic;

BEGIN

      -- Instantiate the Unit Under Test (UUT)
   uut: circuitoXor PORT MAP (
          entrada => entrada,
          Y => Y
        );

   entrada<="00","01" after 10 ns,"10" after 20 ns,"11" after 30
ns,"00" after 40 ns;

END;
```




```
-- FLIP-FLOP D ----------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity flipflopD is
    Port ( D : in  STD_LOGIC;
           CLK : in  STD_LOGIC;
           Q : out  STD_LOGIC;
           nQ : out  STD_LOGIC);
end flipflopD;

architecture Behavioral of flipflopD is

begin

process(CLK)

begin
if(CLK'EVENT AND CLK='1') THEN
      Q<=D;
      nQ<=not D;
end if;
```

```
end process;

end Behavioral;


-- TESTBENCH -----------------------------------------------------------

LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.std_logic_unsigned.all;
USE ieee.numeric_std.ALL;

ENTITY tb IS
END tb;

ARCHITECTURE behavior OF tb IS

    -- Component Declaration for the Unit Under Test (UUT)

    COMPONENT flipflopD
    PORT(
         D : IN  std_logic;
         CLK : IN  std_logic;
         Q : OUT  std_logic;
         nQ : OUT  std_logic
        );
    END COMPONENT;


   --Inputs
   signal D : std_logic := '0';
   signal CLK : std_logic := '0';

      --Outputs
   signal Q : std_logic;
   signal nQ : std_logic;

BEGIN
```