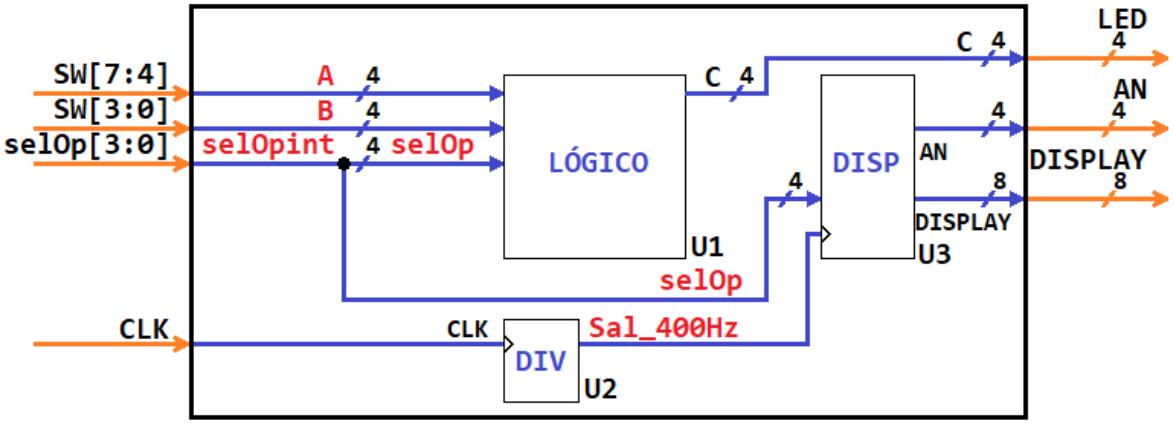
Operaciones Lógicas en VHDL con diseño de alto nivel

Se presenta un ejemplo de un circuito que realiza las operaciones lógicas AND, OR XOR y NOT bit a bit para entradas de 4 bits con salida a display y leds, implementado en la tarjeta nexys 2, utilizando diseño de alto nivel TLD, compuesto por 3 códigos para los tres bloques escritos en VHDL, como se presenta en el diagrama de bloques siguiente y el diseño del proyecto en el ISE de Xilinx:

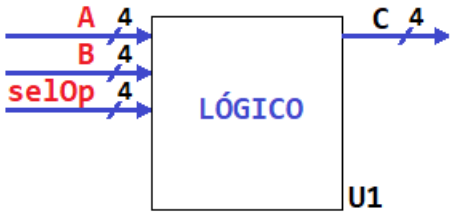



```
----------------------------------------------------------------------------------
-- Top Level Design
-- Lógico AND OR XOR NOT
-- con salida a display.
----------------------------------------------------------------------------------

Library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

---------------------------------------------------------------
-- Declaración de la entidad

entity logico_top is
Port (
        CLK: in STD_LOGIC; -- reloj de 50MHz para la nexys 2 y 100MHz para nexys 3
        SW: in STD_LOGIC_VECTOR(7 DOWNTO 0); -- A SW[7:4], B SW[3:0]
```

```
        selOp: in STD_LOGIC_VECTOR(3 DOWNTO 0); -- selector de operación lógica
        LED: OUT STD_LOGIC_VECTOR(3 DOWNTO 0); -- salida a leds testigos
        DISPLAY: out STD_LOGIC_VECTOR(7 DOWNTO 0); -- segmentos del display "abcdefgP"
        AN: out STD_LOGIC_VECTOR(3 DOWNTO 0)); -- ánodos del display

end logico_top;


------------------------------------------------------------------
-- Declaración de la arquitectura

architecture Behavioral of logico_top is

-- Declaración de señales del divisor
signal SAL_400Hz: std_logic; --salidas 2.5ms

-- Declaración del selector
signal selOpint: std_logic_vector (3 DOWNTO 0); -- op lógica

BEGIN

selOpint <= selOp;

-------------------------------------------------------------------------------------------
-- Declaración del cto lógico -- U1
        U1: ENTITY WORK.logico PORT MAP(
                A               =>      SW(7 DOWNTO 4),     -- a SW
                B               =>      SW(3 DOWNTO 0),     -- a SW
                C               =>      LED,                -- a señal LD
                selOp           =>      selOpint            -- a señal selOp (U3) y BTN
        );
-------------------------------------------------------------

-------------------------------------------------------------------------------------------
-- Declaración del componente del divisor (2.5ms=400Hz) -- U2
        U2: ENTITY WORK.DIV_CLK PORT MAP(
                CLK             =>      CLK,            -- a reloj 50MHz p/nexys2
                SAL_400Hz       =>      SAL_400Hz       -- a señal p/displays (U4)
        );
-------------------------------------------------------------

-------------------------------------------------------------------------------------------
-- Declaración del controlador de display -- U3
        U3: ENTITY WORK.DISPLAYS PORT MAP(
                selOp           =>      selOpint,       -- a señal selOp (U1)
                SAL_400Hz       =>      SAL_400Hz,      -- a señal p/div_clk (U4)
                DISPLAY         =>      DISPLAY,        -- a segmentos del display
                AN              =>      AN              -- a ánodos del display
```

```
        );
----------------------------------------------------------------

end Behavioral; -- fin de la arquitectura TLD
```

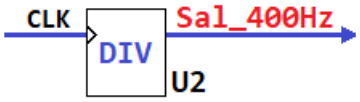
El primer componente (U1) a implementar es el bloque que realiza las funciones lógicas AND, OR XOR y NOT entre las dos A y B de 4 bits, junto con el selector de operación (selOp) para visualizar el resultado en los leds (bit a bit) y el nombre de la operación en los displays, empleando el código siguiente.




```
--------------------------------------------------------------------------------
-- programa que realiza la operación lógica bit a bit entre 2entradas de
-- 4 bits con salida a display y leds
--------------------------------------------------------------------------------

Library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

----------------------------------------------------------------
-- Declaración de la entidad

entity LOGICO is
Port (
        A,B: in STD_LOGIC_VECTOR(3 DOWNTO 0);       -- entradas con interruptores
        C: OUT STD_LOGIC_VECTOR(3 DOWNTO 0);        -- salida a leds testigos
        selOp: in STD_LOGIC_VECTOR(3 DOWNTO 0));    -- a botones
end LOGICO;

----------------------------------------------------------------
-- Declaración de la arquitectura

architecture Behavioral of LOGICO is

BEGIN

----------------------OPERACIONES---------------------------------
-- en este proceso realiza las operaciones según el selector selOp
```

```
PROCESS(A,B,selOp)
begin
        case selOp is                   -- función lógica
        when "0001" => c <= A AND B;  -- AND
        when "0010" => c <= A OR B;    -- OR
        when "0100" => c <= A XOR B;  -- XOR
        when "1000" => c <= NOT A;     -- NOT
        when others => c <= (others => '0');
        end case;
end process; --termina el proceso de operaciones

end Behavioral; -- fin de la arquitectura

----------------------------------------------
```

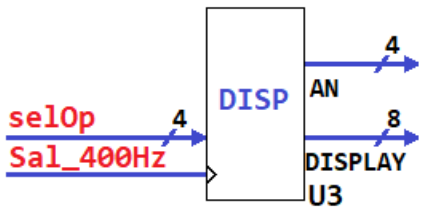
El segundo componente a declarar es U2, un divisor de frecuencia para generar la señal de barrido de los displays a 400Hz (2.5ms). Y el código viene enseguida.




```
----------------------------------------------------------------------------------
-- Divisor de 2.5ms (400Hz)
----------------------------------------------------------------------------------

Library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

-----------------------------------------------------------------
-- Declaración de la entidad

entity DIV_CLK is
Port (
                CLK: in STD_LOGIC; -- reloj de 50MHz para la nexys 2 y 100MHz para nexys 3
                SAL_400Hz: inout STD_LOGIC); --salida 2.5ms,

end DIV_CLK;

-----------------------------------------------------------------
-- Declaración de la arquitectura

architecture Behavioral of DIV_CLK is

-- Declaración de señales de los divisores
```

```
signal conta_1250us: integer range 1 to 62_500:=1; -- pulso1 de 1250us@400Hz (0.25ms)

BEGIN

-----------------DIVISOR 2.5ms=400Hz----------------
-------------------DIVISOR ÁNODOS--------------------
process (CLK) begin

        if rising_edge(CLK) then
                if (conta_1250us = 62_500) then --cuenta 1250us (50MHz=62500)
                        -- if (conta_1250us = 125000) then --cuenta 1250us (100MHz=125000)
                        SAL_400Hz <= NOT(SAL_400Hz); --genera un barrido de 2.5ms
                        conta_1250us <= 1;
                else
                        conta_1250us <= conta_1250us + 1;
                end if;
        end if;

end process; -- fin del proceso Divisor Ánodos

------------------------------------------------
end Behavioral; -- fin de la arquitectura
```

El componente 3 (U3) es el encargado de desplegar cada nombre de la operación lógica seleccionada (selOp) en un display de 4 dígitos conectado en bus (los datos de los segmentos llegan a todos los dígitos) con control de 4 ánodos habilitados en bajo para la Nexys 2.




```
-------------------------------------------------------------------------------
-- Controlador del display de 4 digitos para las letras
-------------------------------------------------------------------------------

Library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

-------------------------------------------------------------
-- Declaración de la entidad
```

```
entity DISPLAYS is
Port (
        selOp: in std_logic_vector (3 DOWNTO 0); -- selector de operación lógica
        SAL_400Hz: in STD_LOGIC; -- reloj de 400Hz
        DISPLAY: out STD_LOGIC_VECTOR(7 DOWNTO 0); -- seg dsply "abcdefgP"
        AN: out STD_LOGIC_VECTOR(3 DOWNTO 0)); -- ánodos del display

end DISPLAYS;

----------------------------------------------------------------
-- Declaración de la arquitectura

architecture Behavioral of DISPLAYS is

-- Declaración de señales de la multiplexación y asignación de letras al disp

signal SEL: std_logic_vector (1 downto 0):="00"; -- selector de barrido

-- Declaración de constantes para las letras (AND, OR, XOR, NOT) al disp

constant A: std_logic_vector (7 downto 0):= "00010001"; -- A
constant N: std_logic_vector (7 downto 0):= "11010101"; -- n
constant D: std_logic_vector (7 downto 0):= "10000101"; -- d
constant V: std_logic_vector (7 downto 0):= "11111111"; -- V sin dato

constant O: std_logic_vector (7 downto 0):= "00000011"; -- O
constant R: std_logic_vector (7 downto 0):= "11110101"; -- r

constant X: std_logic_vector (7 downto 0):= "00111011"; -- ^

constant T: std_logic_vector (7 downto 0):= "11100001"; -- t

BEGIN

--------------------MULTIPLEXOR---------------------

PROCESS(SAL_400Hz, sel, selOp)
BEGIN

        IF SAL_400Hz'EVENT and SAL_400Hz='1' THEN SEL <= SEL + '1';

                CASE(SEL) IS
                when "00" => AN <="0111";       --
                        if    selOp = x"1" then DISPLAY <= V;       -- andV
                        elsif selOp = x"2" then DISPLAY <= V;     -- orvV
                        elsif selOp = x"4" then DISPLAY <= V;     -- ^orV
                        elsif selOp = x"8" then DISPLAY <= V;     -- notV
                        else                            DISPLAY <= V;     -- sin dato
```

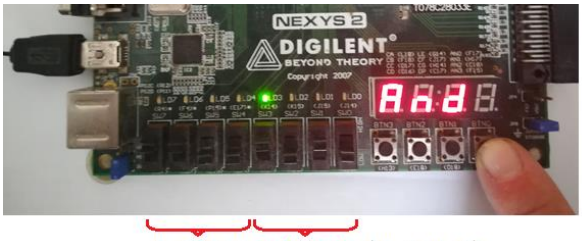
```
                end if;

            when "01" => AN <="1011";      --
                if    selOp = x"1" then DISPLAY <= D;      -- anDv
                elsif selOp = x"2" then DISPLAY <= V;      -- orVv
                elsif selOp = x"4" then DISPLAY <= R;      -- ^oRv
                elsif selOp = x"8" then DISPLAY <= T;      -- noTv
                else                    DISPLAY <= V;      -- sin dato
                end if;

            when "10" => AN <="1101";      --
                if    selOp = x"1" then DISPLAY <= N;      -- aNdv
                elsif selOp = x"2" then DISPLAY <= R;      -- oRvv
                elsif selOp = x"4" then DISPLAY <= O;      -- ^Orv
                elsif selOp = x"8" then DISPLAY <= O;      -- nOtv
                else                    DISPLAY <= V;      -- sin dato
                end if;

            when "11" => AN <="1110";      --
                if    selOp = x"1" then DISPLAY <= A;      -- Andv
                elsif selOp = x"2" then DISPLAY <= O;      -- Orvv
                elsif selOp = x"4" then DISPLAY <= X;      -- ^orv
                elsif selOp = x"8" then DISPLAY <= N;      -- Not
                else                    DISPLAY <= V;      -- sin dato
                end if;

            when OTHERS=>AN <="1111";  --
                if    selOp = x"1" then DISPLAY <= V;      -- and
                elsif selOp = x"2" then DISPLAY <= V;      -- or
                elsif selOp = x"4" then DISPLAY <= V;      -- xor
                elsif selOp = x"8" then DISPLAY <= V;      -- not
                else                    DISPLAY <= V;      -- sin dato
                end if;

            END CASE;

        end if;

END PROCESS; -- fin del proceso Multiplexor

-------------------------------------------------
end Behavioral; -- fin de la arquitectura
```

// Archivo de restricciones de usuario (UCF) //

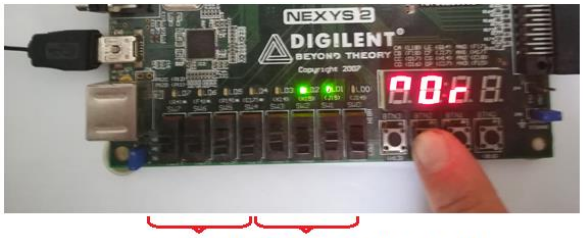
```
########################################
##Asignación de terminales de la NEXYS 2 NEXYS 2 NEXYS 2

//DISPLAYS
NET "DISPLAY(7)" LOC = "L18"; // a
NET "DISPLAY(6)" LOC = "F18"; // b
NET "DISPLAY(5)" LOC = "D17"; // c
NET "DISPLAY(4)" LOC = "D16"; // d
NET "DISPLAY(3)" LOC = "G14"; // e
NET "DISPLAY(2)" LOC = "J17"; // f
NET "DISPLAY(1)" LOC = "H14"; // g
NET "DISPLAY(0)" LOC = "C17"; // P

//ANODOS
NET "AN(3)" LOC = "F17"; // AN0
NET "AN(2)" LOC = "H17"; // AN1
NET "AN(1)" LOC = "C18"; // AN2
NET "AN(0)" LOC = "F15"; // AN3

//selOp--PUSH BUTTON
NET "selOp(0)" LOC = "B18"; // btn0 and
NET "selOp(1)" LOC = "D18"; // btn1 or
NET "selOp(2)" LOC = "E18"; // btn2 xor
NET "selOp(3)" LOC = "H13"; // btn3 not

//CLK reloj 50MHz
NET "CLK" LOC = "B8";

// entrada A y B a SW
NET "SW(0)" LOC = "G18"; // SW0 B(0)
NET "SW(1)" LOC = "H18"; // SW1 B(1)
NET "SW(2)" LOC = "K18"; // SW2 B(2)
NET "SW(3)" LOC = "K17"; // SW3 B(3)

NET "SW(4)" LOC = "L14"; // SW4 A(0)
NET "SW(5)" LOC = "L13"; // SW5 A(1)
NET "SW(6)" LOC = "N17"; // SW6 A(2)
NET "SW(7)" LOC = "R17"; // SW7 A(3)

// salida a led testigos del resultado C
NET "led(0)" LOC = "J14"; // LD0
NET "led(1)" LOC = "J15"; // LD1
NET "led(2)" LOC = "K15"; // LD2
NET "led(3)" LOC = "K14"; // LD3
#NET "led(4)" LOC = "E17"; // LD4
#NET "led(5)" LOC = "P15"; // LD5
```

#NET "led(6)" LOC = "F4" ; // LD6
#NET "led(7)" LOC = "R4" ; // LD7

A continuación, se presentan las fotos del funcionamiento de las salidas lógicas para cada función.

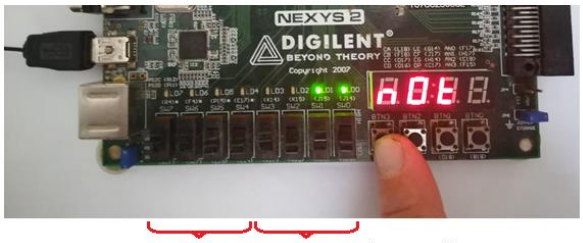


1 1 0 0 1 0 1 0 (A and B)
A B



1 1 0 0 1 0 1 0 (A xor B)
A B

1 1 0 0 1 0 1 0 (A or B)
A B



1 1 0 0 1 0 1 0 (not A)
A B