Parte Teórica

ALUMNO: CANO, ANDRÉS

1) Identifique qué elementos constituyen los LEs de la FPGA Cyclone III a emplear en el laboratorio y qué estructura tienen las LABs.

Los elementos lógicos (LE) también llamados Celdas Lógicas son las unidades lógicas más pequeñas de la familia de dispositivos Cyclone III. Cada LE tiene las siguientes elementos:

-Multiplexor

-Registros

-Lógica de acarreo

-Look up table (LUT) (Son memorias que almacenan una tabla de verdad y también se las llama generadores de funciones; Una LUT de 4 entradas simula una RAM de 16x1)

Por otro lado cada LAB (Bloque de matriz lógica) consta de las siguientes características:

-Cadenas de transporte de LE

-Registros en cadena

-16 LE

-Señales de control para todo el LAB.

2) ¿De qué se trata el Nios® II?

En cuanto a lo que es el procesador Nios® II podemos decir que es el más utilizado en la industria de FPGA. El procesador Nios® II es un integrado de 32 bits y a su vez es compatible con todas las familias Intel® FPGA.

En comparación con otros procesadores su costo es muy bajo lo cual es la principal razón de por qué es tan utilizado, y a su vez realiza el procesamiento de aplicaciones de una manera muy rápida y segura.

3) ¿Qué diferencia existe entre IP cores y los bloques embebidos (ej multiplicador embebido) disponibles en la FPGA?

Los IP cores son programas hechos el VHDL que cuando se lo instancia en la FPGA se llevan a cabo las conexiones requeridas para el diseño del hardware que cumpla con la operación que se quiere realizar. Para realizar solamente una multiplicación requiere compuertas, celdas de memoria, registros y LUT entre otras cosas. Se puede ver a simple vista que debido a los retardos de todas estas cosas no será muy eficiente esta aplicación.

En cambio los bloques embebidos solucionan este inconveniente de los retardos ya que son exclusivamente utilizados para realizar la función para la cual fueron diseñados, realizan esa función y solo esa, como lo es el multiplicador embebido. También el consumo es mucho menor que en los IP cores.

4) ¿Qué tipo de celda de programación posee el dispositivo FPGA Cyclone III?

Posee celdas que son chips de memoria SDRAM (Synchronous Dynamic RAM) de 8 MB.

5) Realice la descripción en VHDL de un Flip Flop JK.

```
library ieee;
use ieee.std_logic_1164.all ;
use ieee.std_logic_arith.all ;
use ieee.std_logic_unsigned.all ;
--creo un FF-JK
entity FF_JK is   -- el nombre de la entidad debe coincidir con el del archivo
port  ( J,K,CLOCK: in std_logic;  --declaro entradas y salidas como siempre
        Q : out std_logic ) ;
end FF_JK ;
architecture prueba of FF_JK is
begin
 process (J,K,CLOCK)    --el proceso se dispara cuando cambia el CLOCK/J/K  (lista de sensibilidad)
 begin
  if (rising_edge(CLOCK)) then
   Q <= ( J and not(Q)) or (not(K) and Q) ;      -- describe el funcionamiento de un FF JK
  end if;
 end process;
end prueba;
```

6) Realice la descripción en VHDL de un restador completo de un bit

```
library ieee;
use ieee.std_logic_1164.all ;
entity RESTADOR_COMPLETO_1BIT is
 Port  (A: in std_logic;    --A,B y Cin son las entradas
   B: in std_logic;
   Cin: in std_logic;
   Cout: out std_logic;   --indica el carry de salida
   S: out std_logic);     -- indica la resta
 end RESTADOR_COMPLETO_1BIT;
 architecture prueba of RESTADOR_COMPLETO_1BIT is
 begin
          S <= A xor B xor Cin;   --indica la salida cuando hay una resta
  Cout <= (not(A) and B) or (Cin and not(A)) or (Cin and B);   --indica operación para obtener carry salida
end prueba ;
```

7) Realice la descripción en VHDL del test bench del restador completo de un bit.

```
library ieee;
use ieee.std_logic_1164.all ;
use ieee.std_logic_arith.all ;
use ieee.std_logic_unsigned.all ;
entity RESTADOR_COMPLETO_TestBench is   --No declaro entradas y salidas en la entidad del testbench
 end RESTADOR_COMPLETO_TestBench;
architecture prueba of RESTADOR_COMPLETO TestBench is
component RESTADOR_COMPLETO      --llamo al componente creado en otro archivo
    port (A : in std_logic;   -- estos parámetros deben coincidir con los de la entidad
    B : in std_logic;
   Cin : in std_logic;
    clk : in std_logic;
    Cout : out std_logic;
   S : in std_logic) ;
   end component ;
   end prueba;
                                -- estas van ser señales de prueba
     signal A : std_logic := '0';      --entrada
     signal B : std_logic := '0';      --entrada
     signal Cin : std_logic := '0';    --entrada
     signal clk : std_logic := '0';          --entrada
     signal Cout : std_logic := '0';         --salida
     signal S : std_logic := '0' ;           --salida


        constant CLOCK_period : time := 20ns;
  begin
   UUT: RESTADOR_COMPLETO port map -- etiqueto los componentes
  ( A  => A,        -- izquierda puertos componente // derecha señales de prueba
   B  => B,
   Cin  => Cin,
   clk  => clk,
   Cout => Cout,
   S => S);
   CLOCK_process: process    --utilizo un proceso para crear la señal del CLK
```

```
begin                    -- CLOCK es la entrada del reloj del FF-D
clk <= '0' ;
 wait for CLOCK_period/2 ;
 clk <= '1' ;
 wait for CLOCK_period/2 ;
end process ;

stim_proc: process   --creo las entradas de prueba a través de procesos
begin  -- establezco todas las combinaciones posibles de las entradas del sumador (3bits)
input_A <= '0'; input_B <= '0' ; input_Cin <= '0'; wait for 50ns ;
input_A <= '0'; input_B <= '0' ; input_Cin <= '1'; wait for 50ns ;
input_A <= '0'; input_B <= '1' ; input_Cin <= '0'; wait for 50ns ;
input_A <= '0'; input_B <= '1' ; input_Cin <= '1'; wait for 50ns ;
input_A <= '1'; input_B <= '0' ; input_Cin <= '0'; wait for 50ns ;
input_A <= '1'; input_B <= '0' ; input_Cin <= '1'; wait for 50ns ;
input_A <= '1'; input_B <= '1' ; input_Cin <= '0'; wait for 50ns ;
input_A <= '1'; input_B <= '1' ; input_Cin <= '1'; wait for 50ns ;
      wait;
      end process ;
end;
```