DIFERENTES ENTORNOS DE APOYO PARA LA IMPARTICIÓN DE UN CURSO DE VHLD

Félix Riesco Peláez

Universidad de León. diefrp@unileon.es

RESUMEN

Los *chips* TTL y CMOS tradicionales (SLI, MSI) que se emplean en la docencia de asignaturas de diseño digital están desapareciendo tanto del mercado como de las aplicaciones. La docencia de asignaturas con contenidos de sistemas digitales debería tener como referencia el lenguaje VHDL. El lenguaje VHDL es un lenguaje estandar en el diseño de circuitos digitales. Un elemento de complejidad asociado a este lenguaje es la finalidad para la que se utiliza: para simulación, descripción y síntesis de sistemas digitales. Un curso de VHDL puede ser enfocado hacia diferentes alumnos (de electrónica, informática, de arquitectura de computadores, sistemas digitales). En cada caso el énfasis de las explicaciones recaerá sobre elementos diferentes del lenguaje. El acceso a Internet nos proporciona diferentes entornos de compilación, simulación y síntesis que trabajan con VHDL. La presente ponencia muestra tres entornos de compilación de VHDL que abarcan diferentes puntos de vista relacionados con el lenguaje

1. INTRODUCCIÓN

VHDL significa *Very High Speed Hardware Description Language*: Lenguaje de descripción de hardware de alta velocidad. Surge a partir de 1982 promovido por el Departamento de Defensa de Estados Unidos (DoD) para facilitar el diseño de sistemas digitales. En un principio la finalidad del lenguaje era la simulación de circuitos. En la actualidad la finalidad abarca la síntesis directa de circuito, la diagnosis de fallos, la documentación de sistemas, el modelado del rendimiento [1]; el lenguaje se está extendiendo también hacia la incorporación de elementos analógicos (VHDL-AMS), simulación en el entorno de microondas.

En los niveles de abstracción que se consideran en la síntesis de un circuito digital el lenguaje VHDL permite trabajar en los siguientes niveles: circuito eléctrico, circuito lógico, nivel de transferencia de registros, nivel de chip, nivel de sistema.

El lenguaje VHDL tiene una serie de particularidades que deben reseñarse: es un lenguaje sin propietario; es un lenguaje muy amplio que casi ningún fabricante soporta por completo, es un estandar, tiene estilos diferentes de descripción de sistemas digitales.

Internet proporciona una gran cantidad de software que está disponible de forma gratuita o mediante solicitud (sistemas operativos, compiladores, programas en general). Entre ellos se van a mencionar tres entornos que trabajan con el lenguaje VHDL: el programa BluePc, el entorno Alliance y el entorno Max Plus II.

BluePc está centrado en la simulación y es más próximo a cuestiones de programación, de software. Alliance está dedicado a cuestiones de síntesis de circuitos como ASICs y descompone este proceso en múltiples etapas permitiendo el acceso a cada nivel de abstracción. Max Plus II es un entorno proporcionado por Altera, fabricante de FPGAs, PLDs, que soporta el mayor subconjunto de VHDL de las tres aplicaciones y permite además llevar los resultados de descripciones de sistemas a la tarjeta del Programa Universitario de Altera, el

Trabajo subvencionado por la Universidad de León

la que se pueden programar diferentes chips para que realicen los diseños que se han compilado y simulado previamente. A continuación se describen brevemente estos entornos.

2. ESTILOS DE PROGRAMACIÓN EN VHDL

Al diseñar un sistema en lenguaje VHDL se pueden seleccionar tres tipos muy diferenciados conceptualmente de descripción del circuito: Los estilos son:

- Algorítmico
- Flujo de datos
- Estructural

El estilo algorítmico es mas próximo a los lenguajes de programación; es el más próximo al programador. Se centra en el uso de la estructura *process*.
El estilo de flujo de datos se aproxima a las descripciones lógicas de conexionado de elementos del sistema; es el estilo más próximo al diseñador de circuitos digitales.
El estilo estructural hace referencia a la interconexión de celdas de diseño (biblioteca) disponibles por la tecnología del fabricante; es el más próximo a la síntesis del circuito.

3. PRIMER ENTORNO DE TRABAJO: EL PROGRAMA BLUEPC

El primer entorno de trabajo que se va a introducir es el programa BluePc. Este programa está disponible de forma gratuita en Internet y se puede obtener en la dirección http://www.bluepc.com, soporta las tres descripciones de VHDL (algorítmica, dataflow y estructural) y proporciona los siguientes elementos de trabajo:
Compilador en un entorno gráfico (figura 1): soporta un subconjunto aceptable de VHDL; en particular soporta las construcciones de tipo process. El compilador soporta las descripciones de paquetes (*package*) y funciones. BluePc dispone de las librerías: ieee.std_logic_1164.all, ieee.std_logic_textio.all, std.textio.
Simulador temporal de sistemas (figura 3): proporciona las formas de onda de respuesta del circuito sometido a unos estímulos de entrada.
Desde el punto de vista didáctico este compilador tiene una gran ventaja respecto de los demás entornos: admite la programación de bancos de pruebas *Test Bench* para los circuitos diseñados previamente utilizando el propio lenguaje VHDL. Los demás entornos realizan la generación de patrones de pruebas en entornos diferenciados (no descritos en VHDL).
El programa proporciona las salidas como forma de onda y como fichero de texto

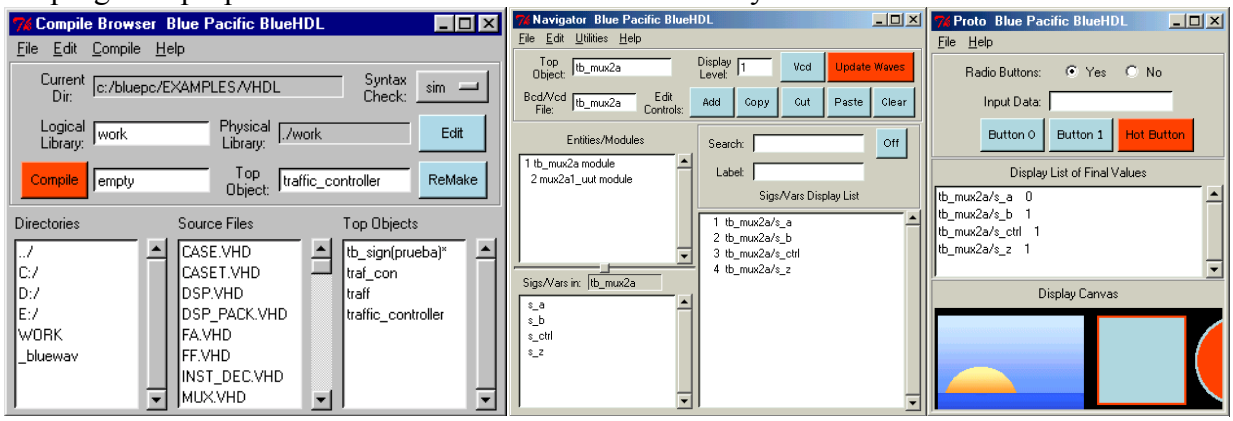

Figura 1: Ventanas correspondientes al programa BluePc

En la página siguiente se muestran las descripciones de un multiplexor 2:1 realizadas mediante el programa BluePc. Se comprueba que se admiten las tres descripciones (algorítmica, dataflow, estructural). Además se programa el banco de pruebas a partir del cual se instancia el circuito que se pretende simular (figura 2).

```
architecture algoritmo
of mux2a1 is
begin
  process (a,b,ctrl)
  begin
    if (ctrl ='0') then
      z <=a;
    else
      z <=b;
    end if;
  end process;
end algoritmo;
```

```
architecture flujo_datos of mux2a1fd
is
signal s_ctrl, n1, n2 : bit;
begin
  s_ctrl <=  not (ctrl);
  n1     <=  s_ctrl and a;
  n2     <=  ctrl and b;
  z      <=  (n1 or n2);
end flujo_datos;
```

```
architecture estructura of mux2a1es
is
signal ctrl_n, n1, n2 : bit;
component INV
port (y   : in bit;
      z   : out bit);
end component;
component AND2
port (x   : in bit;
      y   : in bit;
      z   : out bit);
end component;
component OR2
port (x   : in bit;
      y   : in bit;
      z   : out bit);
end component;
begin
U0: INV  port map (ctrl, ctrl_n);
U1: AND2 port map (ctrl_n, a,n1);
U2: AND2 port map (ctrl, b, n2);
U3: OR2  port map (n1, n2, zout);
end estructura;
```

```
library ieee;
use ieee.std_logic_1164.all;
entity tb_mux2a is
end tb_mux2a;
architecture test of tb_mux2a is
signal s_a:    std_logic;
signal s_b:    std_logic;
signal s_ctrl: std_logic;
signal s_z:    std_logic;
  component mux2a1
  port(a:    in  std_logic;
       b:    in  std_logic;
       ctrl: in  std_logic;
       z:    out std_logic);
  end component;
begin
uut: mux2a1
port map (a => s_a, b=> s_b, ctrl =>
s_ctrl, z => s_z);
  process
    begin
    wait for 10 ns;
    s_a <= '1';
    wait for 10 ns;
    s_ctrl <= '1';
    wait for 10 ns;
    ..........
  end process;
end;
```

Figura 2: Descripciones algorítmica, dataflow y estructural de un multiplexor. Banco de pruebas

El programa BluePc proporciona los resultados de la simulación como formas de onda y como fichero de texto. Previamente se ha tenido que compilar el programa fuente de descripción del circuito y el fichero de tipo *test bench* de la entidad bajo simulación.

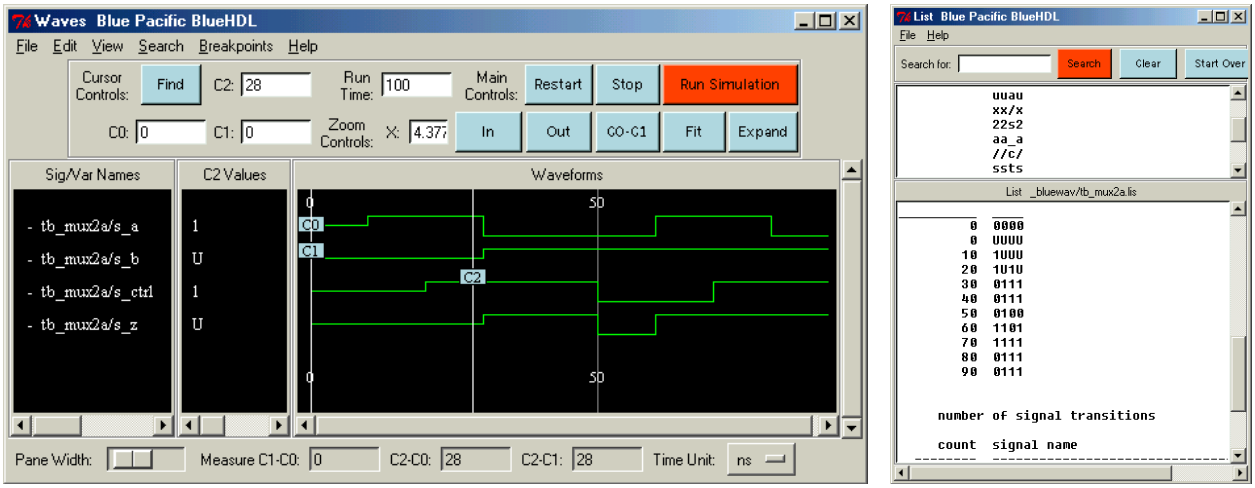

Figura 3: Resultado de una simulación en BluePC: formas de onda y fichero de texto

4. SEGUNDO ENTORNO: ENTORNO ALLIANCE

El entorno Alliance está disponible en Internet en la página http://www-asim.lip6.fr/recherche/alliance. Es un entorno muy amplio y complejo. Se puede ejecutar principalmente sobre máquinas Solaris y Linux. Está disponible desde mediados de los años 90, por lo que hay varias versiones disponibles, cada una de las cuales va incorporando nuevas herramientas de diseño. El entorno Alliance está muy encaminado a la síntesis de circuitos por lo que introduce una gran cantidad de programas de optimización y de verificación junto con utilidades de generación de módulos. Entre los módulos que genera Alliance como descripciones VHDL de sistemas se pueden mencionar: sumadores, multiplicadores, memorias, shifter. La gran ventaja de Alliance respecto de los otros entornos es que proporciona una visión muy completa del diseño ASIC [5].
El entorno Alliance tiene una biblioteca estandar de celdas (junto con otras más específicas) a partir de las cuales se obtienen como ASICs los circuitos que se diseñan.
El diseño de un chip se divide en cuatro niveles de descripción o abstracción:
La descripción behavior (VHDL), la resolución en puertas lógicas, la solución en transistores (CMOS) y el nivel de layout. Un ejemplo de descripción (VHDL: figura 4, transistor: figura 5 y layout figura 6) es la descripción en Alliance de la puerta NAND de tres entradas:

```
-- This cell caracterized by prol05.elp technologie file
-- VHDL data flow description generated from `na3_y`
--                          date : Wed Oct  7 12:14:26 1998
-- Entity Declaration
ENTITY na3_y IS
  GENERIC (
    CONSTANT area : NATURAL := 100800;              -- area
    CONSTANT transistors : NATURAL := 6;            -- transistors
    CONSTANT cin_i2 : NATURAL := 35;    -- cin_i2
    CONSTANT cin_i1 : NATURAL := 35;    -- cin_i1
    CONSTANT cin_i0 : NATURAL := 35;    -- cin_i0
    CONSTANT tplh_i0_f : NATURAL := 461;            -- tplh_i0_f
    CONSTANT rup_i0_f : NATURAL := 2450;            -- rup_i0_f
    CONSTANT tphl_i0_f : NATURAL := 442;            -- tphl_i0_f
    CONSTANT rdown_i0_f : NATURAL := 3430;          -- rdown_i0_f
    CONSTANT tplh_i1_f : NATURAL := 558;            -- tplh_i1_f
    CONSTANT rup_i1_f : NATURAL := 2450;            -- rup_i1_f
    CONSTANT tphl_i1_f : NATURAL := 401;            -- tphl_i1_f
    CONSTANT rdown_i1_f : NATURAL := 3430;          -- rdown_i1_f
    CONSTANT tplh_i2_f : NATURAL := 647;            -- tplh_i2_f
    CONSTANT rup_i2_f : NATURAL := 2450;            -- rup_i2_f
    CONSTANT tphl_i2_f : NATURAL := 313;            -- tphl_i2_f
    CONSTANT rdown_i2_f : NATURAL := 3430           -- rdown_i2_f
  );
  PORT (
  i0 : in BIT;              -- i0
  i1 : in BIT;              -- i1
  i2 : in BIT;              -- i2
  f : out BIT;              -- f
  vdd : in BIT;             -- vdd
  vss : in BIT              -- vss
  );
END na3_y;
-- Architecture Declaration
ARCHITECTURE VBE OF na3_y IS
BEGIN
  ASSERT ((vdd and not (vss)) = '1')
    REPORT "power supply is missing on na3_y"
    SEVERITY WARNING;
f <= not (((i2 and i1) and i0));
END;
```

Figura 4: Descripción VHDL de una puerta NAND

```
V ALLIANCE : 6
H na3_y,L, 7/10/98
C vss,IN,EXTERNAL,4
C vdd,IN,EXTERNAL,5
C f,OUT,EXTERNAL,2
C i2,IN,EXTERNAL,7
C i1,IN,EXTERNAL,8
C i0,IN,EXTERNAL,6
T N,0.5,6.2,3,7,4,0,1,1,14.4,14.4,5.5,6.25,tr_00001
T N,0.5,6.2,1,8,3,0,1,1,14.4,14.4,8.5,6.25,tr_00002
T N,0.5,6.2,2,6,1,0,1,1,14.4,14.4,11.5,6.25,tr_00003
T P,0.5,6.2,2,7,5,0,1,1,14.4,14.4,5.5,16.25,tr_00004
T P,0.5,6.2,5,8,2,0,1,1,14.4,14.4,8.5,16.25,tr_00005
T P,0.5,6.2,2,6,5,0,1,1,14.4,14.4,11.5,16.25,tr_00006
S 3,INTERNAL,mbk_sig3
Q 0
S 1,INTERNAL,mbk_sig1
Q 0
S 2,EXTERNAL,f
Q 0.011449
S 6,EXTERNAL,i0
Q 0.00812157
S 8,EXTERNAL,i1
Q 0.00812157
S 7,EXTERNAL,i2
Q 0.00810479
S 5,EXTERNAL,vdd
Q 0.0121456
S 4,EXTERNAL,vss
Q 0.0114237
EOF
```

Figura 5: Conexionado de transistores

El inconveniente fundamental que presenta el entorno Alliance es que el subconjunto de VHDL que se soporta es pequeño: esencialmente soporta las descripciones concurrentes de bloques (Block) y no soporta descripciones algorítmicas. Por el contrario, el nivel de detalle que alcanza en la descripción de los procesos de síntesis es muy amplia y proporciona las herramientas de paso de un nivel de abstracción a otro (y generalmente dispone de la herramienta inversa de recuperación del nivel de procedencia)

```
V ALLIANCE : 4
H na3_y,P, 7/ 9/98,100
A 500,300,2900,4500
C 500,4300,800,vdd,0,WEST,ALU1
C 2900,4300,800,vdd,1,EAST,ALU1
C 800,4500,200,i2,1,NORTH,ALU2
C 1400,4500,200,i1,1,NORTH,ALU2
C 2000,4500,200,i0,1,NORTH,ALU2
C 2600,4500,200,f,1,NORTH,ALU2
C 500,500,800,vss,0,WEST,ALU1
C 2900,500,800,vss,1,EAST,ALU1
C 800,300,200,i2,0,SOUTH,ALU2
C 1400,300,200,i1,0,SOUTH,ALU2
C 2000,300,200,i0,0,SOUTH,ALU2
C 2600,300,200,f,0,SOUTH,ALU2
S 800,200,2600,200,200,vss,RIGHT,ALU1
S 800,200,800,1400,200,vss,UP,ALU1
S 500,3900,2900,3900,2600,*,RIGHT,NWELL
S 1100,2500,1100,4000,100,*,UP,PTRANS
S 1700,2500,1700,4000,100,*,UP,PTRANS
S 2300,2500,2300,4000,100,*,UP,PTRANS
S 1100,500,1100,2000,100,*,UP,NTRANS
S 1700,500,1700,2000,100,*,UP,NTRANS
S 2300,500,2300,2000,100,*,UP,NTRANS
S 1400,300,1400,4500,200,*,UP,ALU2
S 2000,700,2000,1800,300,*,UP,NDIF
S 2600,700,2600,1800,300,*,UP,NDIF
S 1400,700,1400,1800,300,*,UP,NDIF
S 2600,2700,2600,3800,300,*,UP,PDIF
S 1400,2700,1400,3800,300,*,UP,PDIF
S 2000,2700,2000,3900,300,*,UP,PDIF
S 800,2700,800,3900,300,*,UP,PDIF
S 1100,2000,1100,2500,100,*,UP,POLY
S 800,2200,1100,2200,300,*,RIGHT,POLY
S 1700,2000,1700,2500,100,*,UP,POLY
S 1500,2200,1700,2200,200,*,RIGHT,POLY
S 2300,2000,2300,2500,100,*,UP,POLY
S 2100,2200,2300,2200,200,*,RIGHT,POLY
S 2000,300,2000,4500,200,*,UP,ALU2
S 800,700,800,1800,300,*,UP,NDIF
S 800,300,800,4500,200,*,UP,ALU2
S 2600,300,2600,4500,200,f,UP,ALU2
S 1400,2800,2600,2800,200,*,RIGHT,ALU1
S 1400,2800,1400,3400,200,*,UP,ALU1
S 800,2200,800,2800,100,*,UP,ALU1
S 2600,1400,2600,3400,200,*,UP,ALU1
S 800,200,2600,200,200,*,RIGHT,PTIE
S 800,4500,2600,4500,200,*,RIGHT,NTIE
S 1400,1700,1400,2200,200,*,UP,ALU1
S 1400,2200,1500,2200,200,*,RIGHT,ALU1
S 2000,1700,2000,2200,200,*,UP,ALU1
S 2000,2200,2100,2200,200,*,RIGHT,ALU1
S 800,3300,800,4500,200,vdd,UP,ALU1
S 500,4300,2900,4300,800,vdd,RIGHT,ALU1
S 1400,4300,1400,4500,200,vdd,UP,ALU1
S 2000,3300,2000,4500,200,vdd,UP,ALU1
S 2600,4300,2600,4500,200,vdd,UP,ALU1
S 500,500,2900,500,800,vss,RIGHT,ALU1
V 2600,2800,CONT_DIF_P
V 800,200,CONT_BODY_P
V 2600,200,CONT_BODY_P
V 800,4500,CONT_BODY_N
V 2600,4500,CONT_BODY_N
V 800,2800,CONT_VIA
V 2600,2200,CONT_VIA
V 2000,1700,CONT_VIA
V 1400,1700,CONT_VIA
V 2100,2200,CONT_POLY
V 1500,2200,CONT_POLY
V 800,2200,CONT_POLY
V 2600,3400,CONT_DIF_P
V 2000,3300,CONT_DIF_P
V 2000,3900,CONT_DIF_P
V 1400,3400,CONT_DIF_P
V 1400,2800,CONT_DIF_P
V 800,3300,CONT_DIF_P
V 800,3900,CONT_DIF_P
V 2600,1400,CONT_DIF_N
V 800,800,CONT_DIF_N
V 800,1400,CONT_DIF_N
V 2000,200,CONT_BODY_P
V 1400,200,CONT_BODY_P
V 2000,4500,CONT_BODY_N
V 1400,4500,CONT_BODY_N
EOF
```

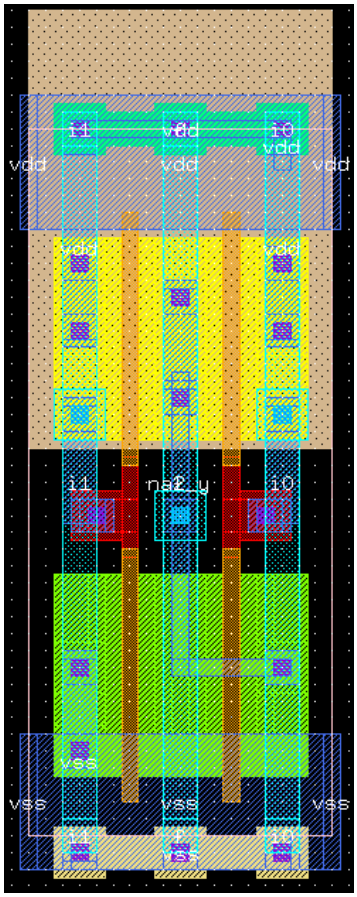

Figura 6: Descripción y visualización del layout de una puerta NAND de tres entradas

El entorno Alliance dispone además de otros programas para la visualización de resultados (formas de onda), de circuitos, de layouts, de máquinas de estados finitas. Para ello se precisa

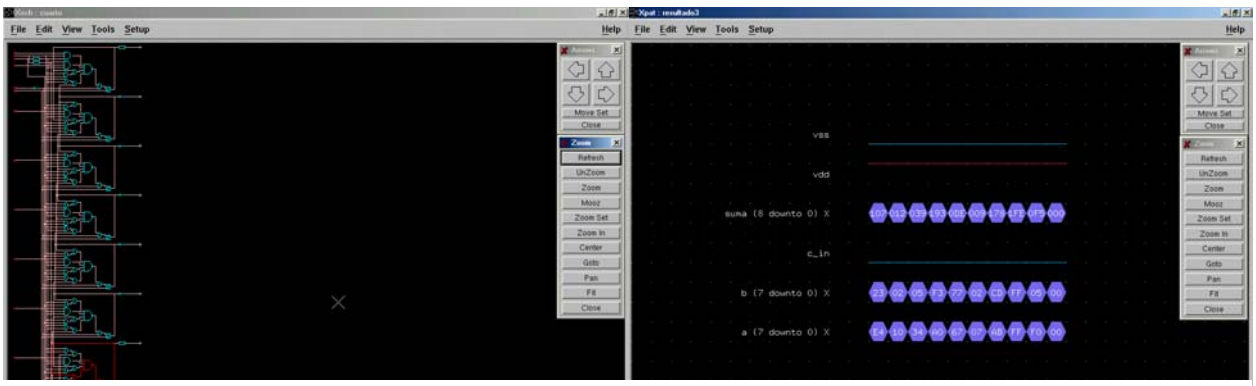

Figura 7: Ventanas de los programas xsch y xpat: visualización de circuitos (puertas) y señales

de un soporte de Xwindows con una cantidad de colores suficientemente amplia.

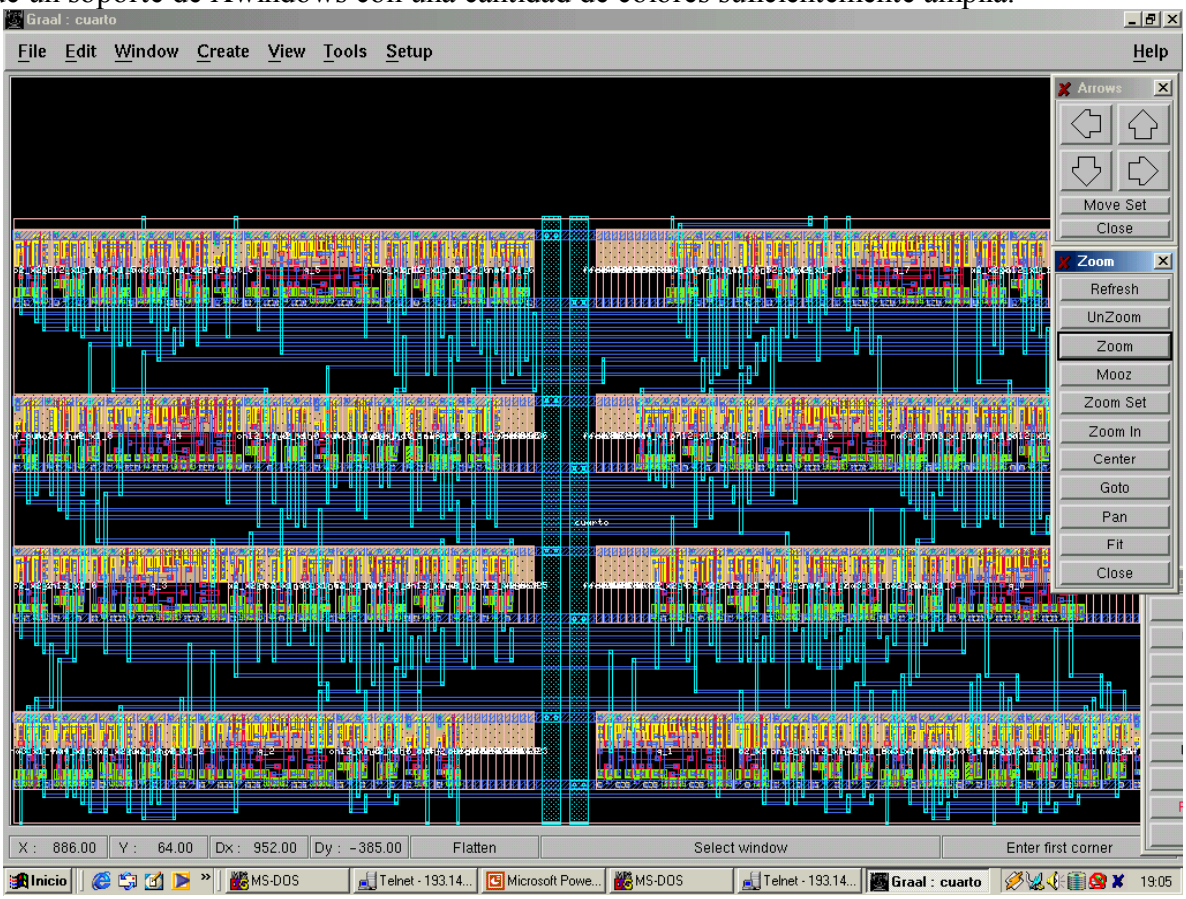

Figura 8: El programa graal: visualizador de los layouts obtenidos

Los principales programas de visualización son los siguientes:
Programa xpat: sirve para visualizar los patrones de respuesta de los circuitos simulados (figura 7a), el programa xsch sirve para ver el circuito obtenido desde el punto de vista de síntesis lógica (figura 7b). Está basado en la utilización de puertas, latches y elementos disponibles como celdas, el programa graal permite la visualización de los layouts de los circuitos obtenidos (figura 8).
El programa que realiza el compilado de los circuitos en el entorno Alliance se denomina Asimut y precisa de un fichero fuente (vbe) en el que se describe en estilo dataflow el comportamiento del circuito que se pretende diseñar. Una vez que el circuito se ha compilado libre de errores, la simulación se realiza con el mismo programa asimut, el fichero de descripción del circuito y un fichero de estímulos o patrones (fichero .pat). El fichero de resultado tiene un formato de tipo *.pat y puede ser visualizado por el programa xpat. Asimut reemplaza los símbolos * y ? por resultados de valores obtenidos mediante la simulación (en formato binario, octal y hexadecimal)
El gran inconveniente que aparece en la utilización de Alliance es que el subconjunto de VHDL soportado es muy reducido. Esencialmente se soportan descripciones basadas en la utilización de elementos concurrentes (sentencia *block*). La descripción de elementos de tipo registro a partir de sentencias block aparece desaconsejada en algunas referencias por considerarla obsoleta: es claramente una descripción encaminada a la síntesis.
Como ejemplo de utilización del entono Alliance vamos a describir un elemento combinacional: un incrementador de tres bits resuelto a partir del cálculo de sus ecuaciones lógicas :

```
Entity incrementador is

Port (e2 : in bit;
          e1: in bit;
          e0: in bit;
          out2: out bit;
          out1: out bit;
          out0: out bit;
          vdd: in bit
          vss: in bit);

End incrementador;

Architecture beh of
incrementador is

Signal s2: bit;
Signal s1: bit;
Signal s0: bit;
```

```
BEGIN
ASSERT ((Vdd and not(vss))='1')
REPORT "Fallo en la alimentación";
SEVERITY WARNING;

S2 <=(not(e2) and     e1  and     eo ) or
     (    e2  and not(e1) and not(e0)) or
     (    e2  and not(e1) and     e0 ) or
     (    e2  and     e1  and not(e0));
S1 <=(not(e2)  and not(e1) and     eo ) or
     (not(e2) and     e1  and not(e0)) or
     (    e2  and not(e1) and     e0 ) or
     (    e2  and     e1  and not(e0));S0
<=(not(e2) and not(e1) and not(eo)) or
     (not(e2) and     e1  and not(e0)) or
     (    e2  and not(e1) and not(e0)) or
     (    e2  and     e1  and not(e0));
Out2 <= s2;
Out1 <= s1;
Out0 <= s0;
End;
```

Figura 9: Descripción de un incrementador de tres bits en estilo dataflow

```
In e2;
In e1;
In e0;
Out out2;
Out out1;
Out out0;
In vdd;
In vss
Begin
Pat_1 : 0 0 0 * * * 1 0;
Pat_2 : 0 0 1 * * * 1 0;
Pat_3 : 0 1 0 * * * 1 0;
Pat_4 : 0 1 1 * * * 1 0;
Pat_5 : 1 0 0 * * * 1 0;
Pat_6 : 1 0 1 * * * 1 0;
Pat_7 : 1 1 0 * * * 1 0;
Pat_8 : 1 1 1 * * * 1 0;
End;
```

```
In e2;
In e1;
In e0;
Out out2;
Out out1;
Out out0;
In vdd;
In vss
Begin
Pat_1 : 0 0 0 0 0 1 1 0;
Pat_2 : 0 0 1 0 1 0 1 0;
Pat_3 : 0 1 0 0 1 1 1 0;
Pat_4 : 0 1 1 1 0 0 1 0;
Pat_5 : 1 0 0 1 0 1 1 0;
Pat_6 : 1 0 1 1 1 0 1 0;
Pat_7 : 1 1 0 1 1 1 1 0;
Pat_8 : 1 1 1 0 0 0 1 0;
End;
```

Una vez compilada la descripción dataflow del incrementador se somete al fichero de estímulos (figura 10a) y el programa asimut nos proporciona como salida un fichero de patrones (figura 10b). Este fichero tiene formato *.pat y puede ser representado a partir del programa xpat de visualización de formas de onda.

Figura 10: Estímulos de entrada y respuesta del circuito

4. TERCER ENTORNO: EL PROGRAMA MAX-PLUS II DE ALTERA

El tercer entorno que se puede emplear como soporte de un curso de VHDL es el programa Max Plus II, de Altera. Altera en un fabricante de circuitos integrados, en particular FPGAs. LA página web de Altera está disponible en http://www.altera.com.
En algunos de sus elementos pueden "instanciarse" procesadores enteros. De forma paralela al diseño de los componentes de hardware hay un conjunto de paquetes y aplicaciones de software que permiten la programación de los chips. Uno de los programas que permite la utilización de descripciones en VHDL de los circuitos se denomina Max Plus II. Existen otros programas más amplios para trabajar en labores de diseño de circuitos digitales de mayor capacidad como es el caso del programa Quartus II.

Max Plus II admite varios métodos de descripción de sistemas digitales [3]:
Lenguajes: AHDL (Altera Hardware Description Language) VHDL y Verilog HDL
Entorno gráfico: para describir un sistema como componentes y conexiones
Max Plus admite también herramientas de síntesis de macrofunciones: elementos comunes de memorias, (*latchs*, RAM, *shifters*, FIFO), bloques aritméticos (sumadores, restadores, contadores, divisores, comparadores).
La finalidad del software de Max Plus es el compilado, simulación y programación de los chips de la familia de Altera de FPGAs y PLDs.

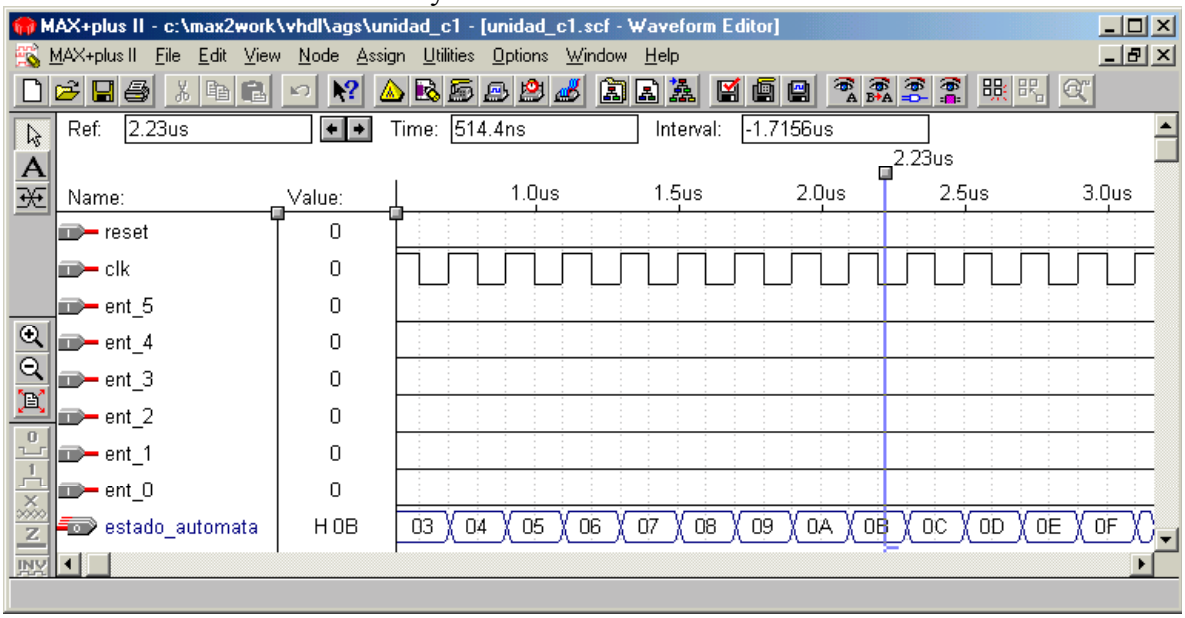

Figura 11: Ventana del simulador de Max Plus II de Altera

Para que la simulación se pueda realizar se proporciona un generador de patrones de test que describirá las señales que se van a aplicar al diseño.
Una vez que se ha realizado la compilación y se ha determinado el chip final de implantación de la solución se permite visualizar la utilización de macrocélulas y la ocupación de cada una de ellas. Max Plus II permite especificar el chip destino del diseño que se está realizando y el programa nos avisará si nuestra descripción VHDL rebasa la capacidad de síntesis del chip (tanto en número de puertas lógicas como de registros). Además el programa permite analizar los retardos debidos al uso de puertas combinacionales en los diseños entre registros. Max Plus II permite establecer el lugar físico (el número de pin) en el que se van a conectar o cablear las entradas y salidas al diseño digital.

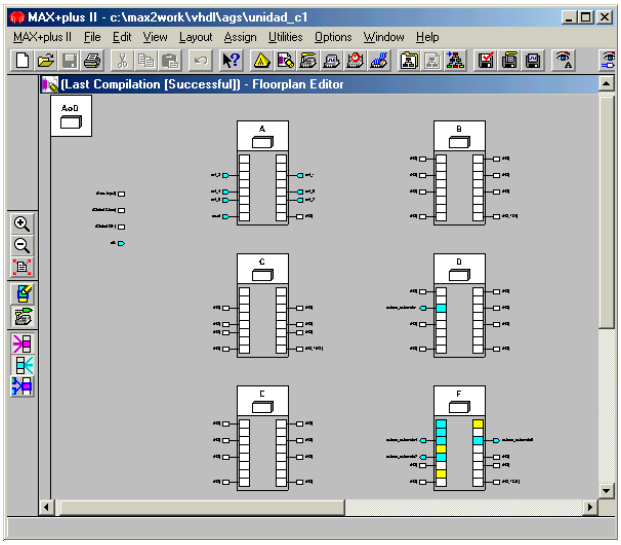

Figura 12: Macroceldas del chip EPM7128

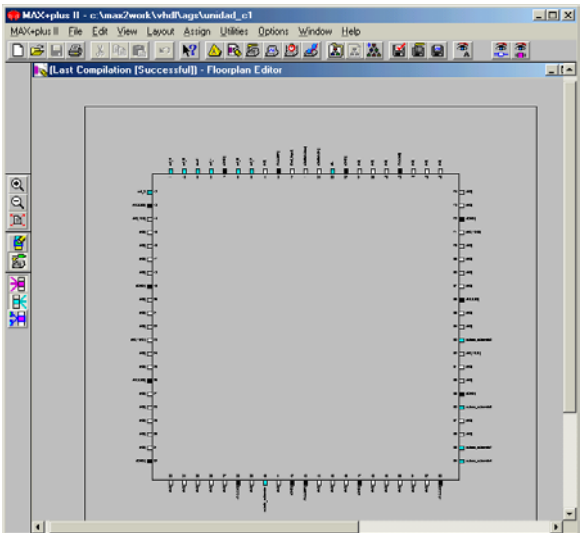

Figura 13: Asignación de pines hacia el exterior

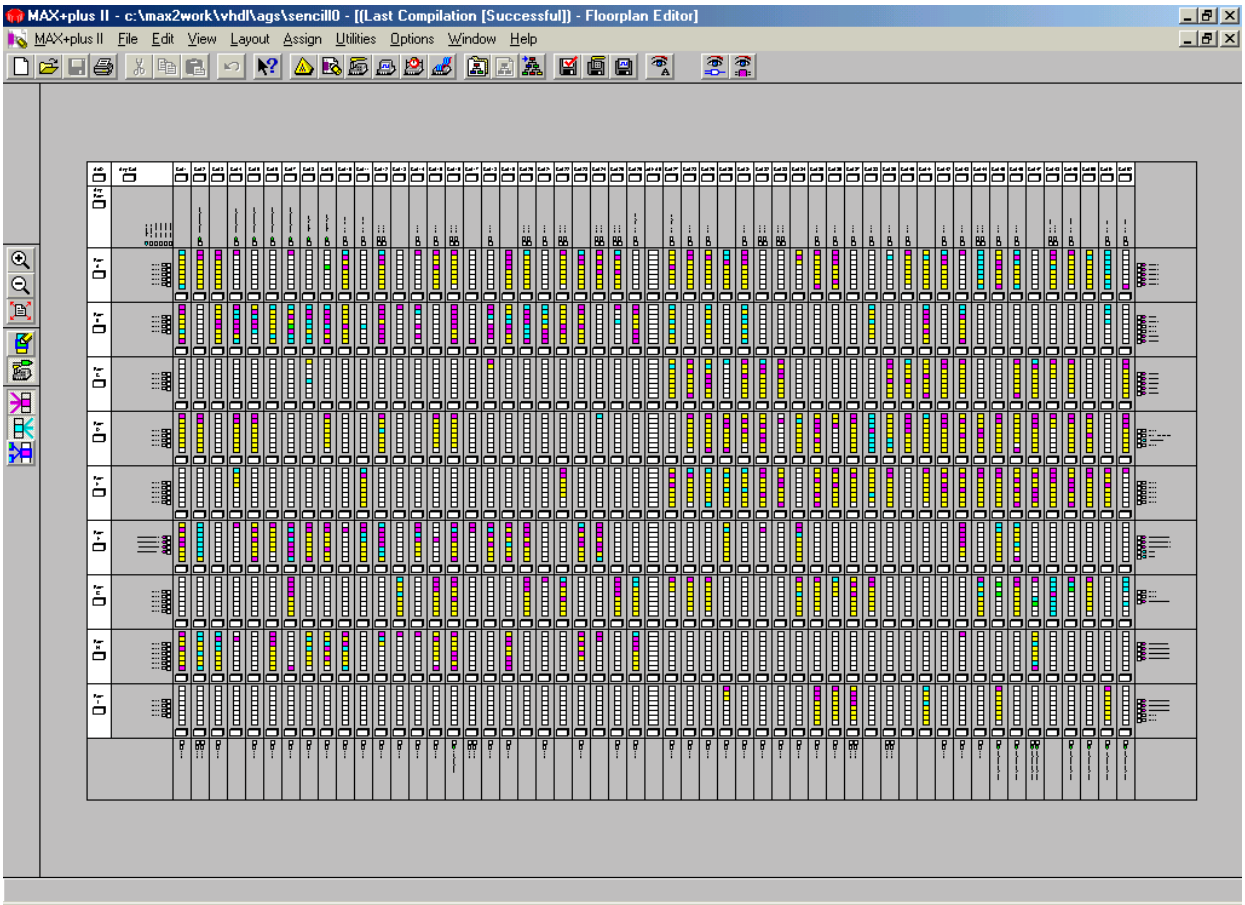

Figura 14: Utilización de celdas en el chip FLEX EPF10K70

El programa Max Plus II viene protegido por una "mochila" para su ejecución aunque se puede solicitar un fichero de licencia a Altera.

5. LA TARJETA UNIVERSITARIA DE ALTERA

Altera dispone de una tarjeta de introducción a sus componentes: la tarjeta *University Program*. Esta tarjeta dispone de dos chips de Altera: MAX EPM 7128SLC84-7 y el FLEX EPF10K70RC240-4 que permiten ser programados por el programa Max Plus II. Además del

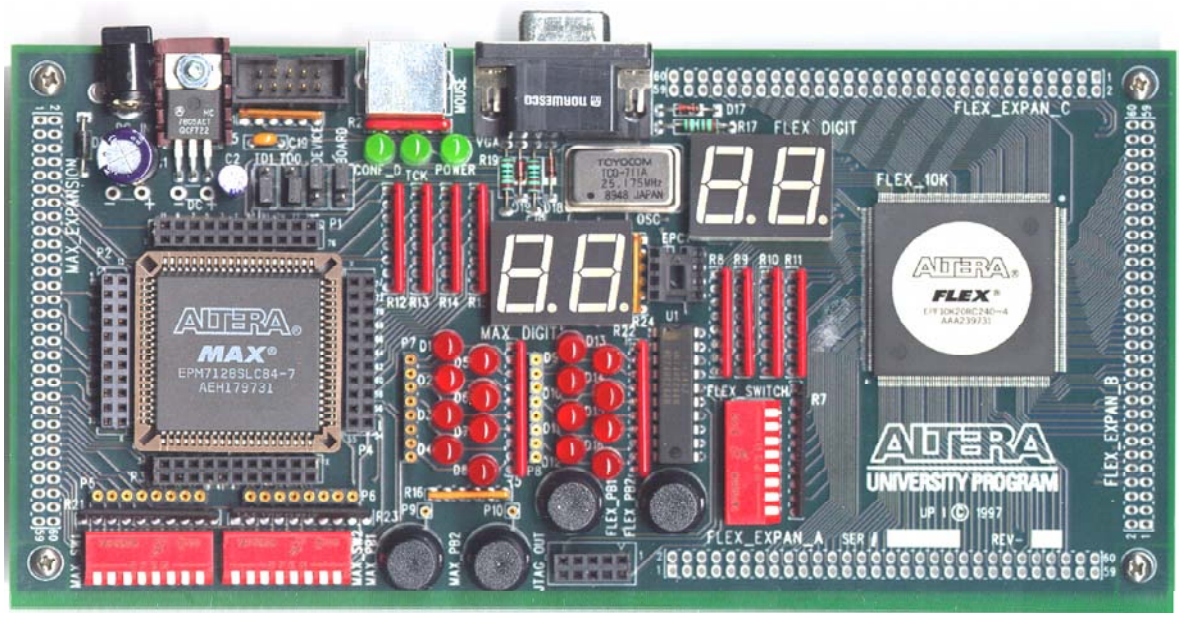

Figura 15: La tarjeta del programa universitario de Altera

software y de una mochila de licencia, la tarjeta dispone de *switchs*, botones, leds y displays así como de un generador de reloj incorporado. El reloj trabaja a una frecuencia de 25.175 MHz que no forma parte de la gama rápida de dispositivos de Altera.
Los chips en los que se basa la tarjeta son grabados individualmente mediante un conector *Byte Blaster* de manera que el circuito queda programado para realizar la tarea que se ha descrito y compilado previamente en algún lenguaje.
El *chip* de la familia MAX tiene la siguiente estructura (figura 16). Cada elemento de tipo macrocélula del chip tiene la estructura de la figura 17.

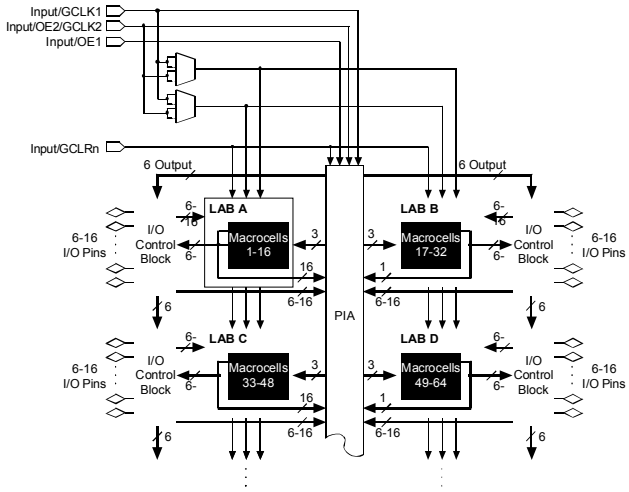

Figura 16: Macroceldas del chip MAX EPM7128

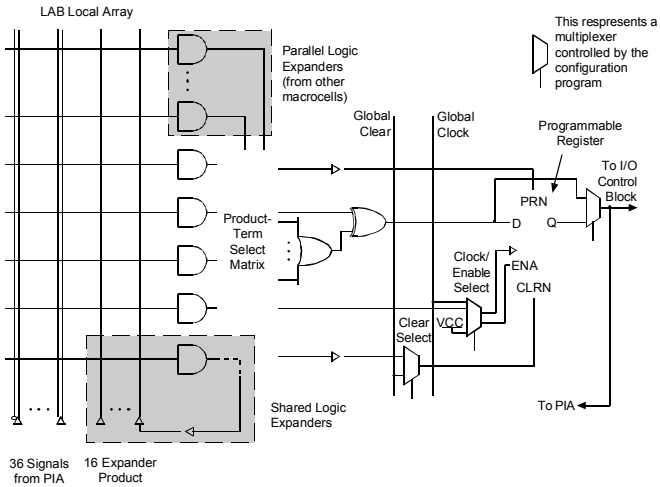

Figura 17: Elemento de macrocelda

El *chip* de la familia FLEX tiene la siguiente 128 macroceldas. Cada macrocelda dispone de un plano AND un plano OR y un registro programable, un reloj programable y la posibilidad de establecer condiciones de set y reset sobre la celda del biestable. Tiene la capacidad de disponer de unas 2500 puertas lógicas para generar la lógica de conexionado.

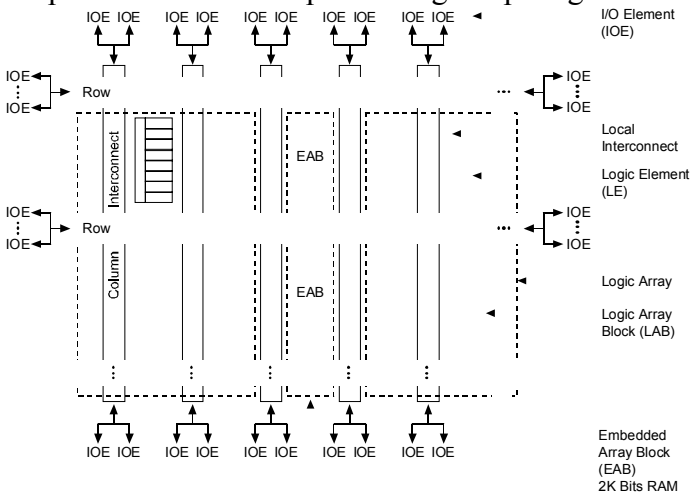

Figura 18: Interconexionado de macroceldas: EPF10K70

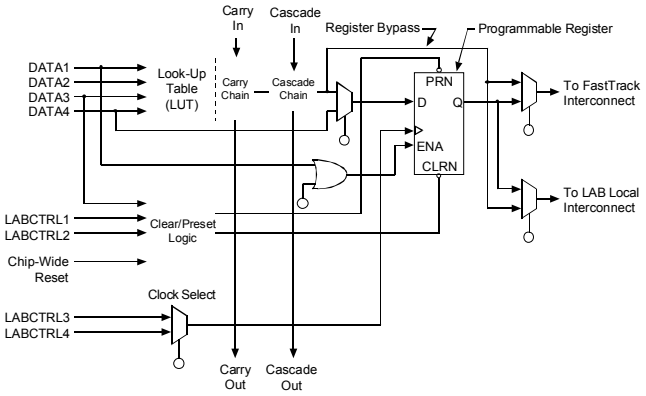

Figura 19: Detalle de la macrocelda del EPF10K70

El chip posee 3.744 LEs (elementos lógicos) y 9 alineamientos empotrados EABs con una capacidad cada bloque de 2048 bits de memoria. En total el chip dispone de la capacidad de almacenar 18432 bits de memoria de tipo SRAM. El chip en conjunto dispone de unas 70.000 puertas combinacionales que pueden emplearse en el diseño (por ejemplo, generar un bloque multiplicador de un ancho de palabra determinado).

Un aspecto importante de la tarjeta universitaria es que puede generar salidas simples hacia la un monitor VGA y recibir datos de un teclado o un ratón [2]. Mediante el programa Max Plus II y la tarjeta universitaria de Altera el prototipado de sistemas digitales se resuelve de forma sencilla. En el desarrollo de un diseño digital pueden considerase las etapas:

- Descripción del sistema (mediante algún HDL)
- Compilado
- Simulación
- Programación
- Verificación

Mediante la utilización de la tarjeta universitaria de Altera y el software de Max Plus II quedan cubiertos razonablemente todos los aspectos mencionados.

6. CONCLUSIÓN

La docencia práctica de asignaturas de diseño digital debería complementar los trabajos con componentes LSI, MSI, incorporando el lenguaje VHDL como estandar de expresión de sistemas digitales.

Existen múltiples aplicaciones software que admiten, compilan y simulan sistemas descritos en VHDL. Al ser el lenguaje VHDL muy amplio complejo y tener diferentes estilos de descripción se debe establecer como elemento discriminador de la herramienta de trabajo el tipo de alumno al que van dirigidas las prácticas que se pretendan realizar.

Aparecen tres opciones diferenciadas al considerar la impartición de prácticas:

- Alumnos con perfiles informático/software, debería considerase un programa del tipo de BluePc

- Alumnos con perfil hardware/tecnología electrónica, debería considerarse un entorno del tipo Alliance
- Alumnos con perfil desarrollo de sistemas digitales, debería considerase un entorno de tipo Max Plus II de Altera, o el equivalente de otro fabricante (Xilinx [4])

En cualquier caso los programas no son excluyentes y pueden usarse los tres (y otros más) para proporcionar una panorámica lo más amplia posible del lenguaje VHDL como método de descripción y simulación de sistemas digitales.

En el caso de la tarjeta Universitaria de Altera la capacidad de los chips de que dispone de resolver sistemas digitales es media (chip EMP7128) a media-alta (chip EPF10K70). La adopción de este tipo de tarjetas abre el camino de acceso a tarjetas mucho más robustas, rápidas, grandes (en el sentido de disponer de un número mayor de bits de tipo RAM). A partir de estas tarjetas "grandes" pueden afrontarse problemas de diseño de sistemas paralelos, DSPs, instanciación de microprocesadores, cuestiones no accesibles con los elementos tradicionales de diseño LSI y MSI.

REFERENCIAS:

[1] Cohen, Ben, VHDL Codign Styles and Metodologies, Kluwer Academic Publishers, Massachusetts, 1999
[2] Hamblen, James O. Furman Michael D. Rapid Protityping of Digital Systems. A tutorial Aproach. Kluwer Academic Publishers. USA 2001
[3] Max Plus II Programmable Logic Development System. Getting Started. Altera, 1997
[4] Mandado, Enrique, Álvarez, Jacobo, Valdés, Mª Dolores. Thomson Madrid 2002
[5] Smith, Michael John Sebastian, Application-Specific Integrated Circuits, Addisson Wesley, 1999