VHDL y el método de diseño basado en descripción y síntesis.

AUTOR : Pablo Mazzara.
Grupo de Microelectrónica del IIE.
Facultad de Ingeniería.
Montevideo. Uruguay.
e-mail mazzara@iie.edu.uy

RESUMEN.

Una de las dificultades que se atribuyen al diseño con VHDL es la gran cantidad de arquitecturas posibles para un mismo circuito. El lenguaje no se halla orientado hacia ningún método en especial, ni a un tipo de lógica ni a una tecnología dada sino que deja esas elecciones a cargo de quien diseña, y en gran medida el éxito del diseño dependerá de las elecciones que este realice.

Método.-

Aquí se describe la aplicación del método que va de lo general a lo particular, y que consiste en describir el circuito (incluyendo las señales de entrada y salida) a nivel de comportamiento para luego realizar la síntesis del mismo.

El nivel de complejidad de los circuitos que vamos a tratar corresponde a la Lógica de Transferencia de Registros (RTL) y vamos a obtener una descripción VHDL de tipo comportamiento de una secuencia RTL. A partir de la misma se realizan las simulaciones y posterior síntesis del circuito.

Descripción VHDL de comportamiento.-

Esta es quizás la etapa mas importante del diseño, ya que en ella se toman decisiones claves para el resultado final.

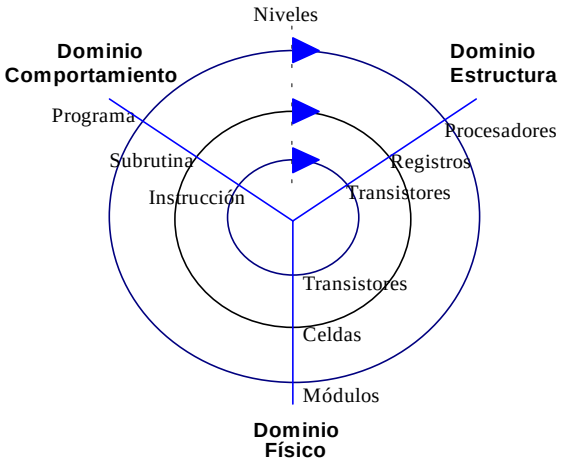
El método se clarifica con un ejemplo, en el cual se ha buscado también la aplicación de los criterios de jerarquía, modularidad y regularidad. Para ello se trata de que todos los pasos de la secuencia sean iguales, cuando ello es posible. También se siguieron las recomendaciones de no usar lógica auxiliar en las señales de reloj (todos los cambios del sistema se producen sincronizados con la señal de reloj la cual es única para todos los Flip Flops.)

Síntesis.-

Ya en la descripción inicial se halla presente la división del circuito en una parte de control y una parte de datos. Por lo tanto la síntesis consistirá en pasar de la descripción de comportamiento a una descripción de estructura (y comportamiento) que contemple los requisitos expuestos.

En esta etapa son importantes la elección del tipo de lógica a utilizar y la estrategia de reloj, esto se podrá apreciar en el ejemplo que consiste en un circuito de control para un convertidor A/D de aproximaciones sucesivas.

Dominios del proceso de diseño



En la carta Y (Gajski y Kuhn 1983) se muestra una división del proceso del diseño en tres dominios:

Comportamiento.
En una especificación de este tipo decimos cual es la función que cumple el sistema.

Estructural.- En este dominio especificamos cómo se conectan entre si las entidades (componentes) para satisfacer el comportamiento deseado.

Físico.- Especifica cómo construir una estructura física que tenga la conectividad requerida para obtener la función deseada.

En cada dominio existen varias opciones de diseño para resolver un problema particular. Por ejemplo, en el dominio del comportamiento tenemos la posibilidad de elegir entre algoritmo tipo secuencia o tipo paralelo. En el dominio de la estructura podemos optar por una familia lógica particular, por una determinada estrategia de reloj, o por un estilo de circuito. A nivel físico podemos optar por diferentes realizaciones de circuitos en términos de obleas, tarjetas, gabinetes etc.

Cada dominio puede ser especificado a una variedad de niveles de abstracción. Cada círculo concéntrico indica un nivel de abstracción usado en diseño electrónico.

Métodos de diseño.

Debido a la cantidad de transistores que se pueden colocar en una única oblea, es necesario encontrar nuevos procedimientos de diseño y se comprende la importancia de buscar métodos que permitan crear y verificar especificaciones durante las fases iniciales de concepción y partición de un sistema ya que es acá donde se decide el éxito de un proyecto.

Como forma de resolver esa complejidad se usan los recursos de jerarquía, regularidad, diseño modular, etc.

La tendencia que se ha impuesto actualmente consiste en presentar el objeto **complejo** en diversos **niveles** de descripción (**jerarquías**), partiendo desde el más **abstracto** (representación funcional o de comportamiento) hasta llegar al plano de la realización concreta.

Cada descripción sólo considera una parte de los atributos correspondientes a los elementos manipulados (**abstracción**), es decir, aquéllos que son significativos para el nivel en cuestión. Por ejemplo una descripción en alto nivel hace abstracción de los detalles correspondientes a una realización particular como son el tipo de lógica, el tipo de reloj, la tecnología etc.

La clave del método consiste en que una vez definidos los niveles se establecen los mecanismos para pasar de una descripción mas abstracta a una más detallada. La meta de este

procedimiento es la **síntesis automática** conocida como Automatización Electrónica del Diseño (**EDA)**.

Nivel RTL de los dominios de comportamiento y estructura.

SECUENCIA RTL.

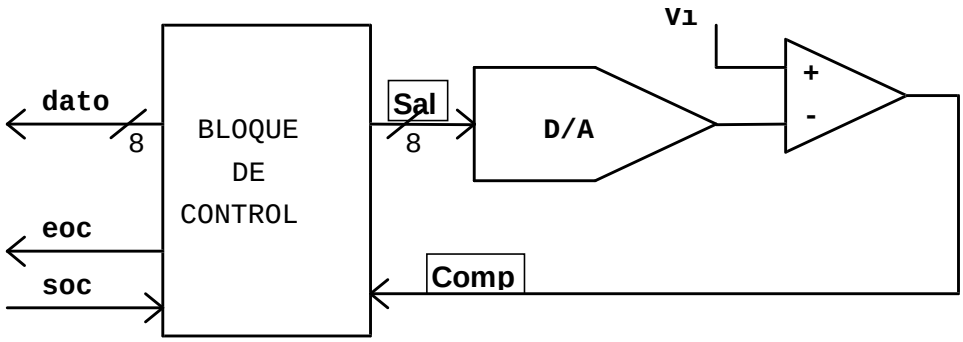
Los circuitos que se pueden describir mediante una secuencia RTL son aquellos en que se puede dividir su funcionamiento en una serie de pasos. En cada paso el circuito debe realizar una cierta función que se traduce en la transferencia de unos datos (área de datos) entre registros y evaluar ciertas condiciones para pasar al siguiente paso (área de control).

Esto implica que podemos dividir el circuito en un área de datos y un área de control.

El área de datos establecerá las transferencias que se quieren realizar en cada paso, y el área de control determinará en qué paso nos encontramos y a qué paso vamos de acuerdo a ciertas condiciones que se evalúan.

EJEMPLO.

Un convertidor A/D de aproximaciones sucesivas convierte niveles de tensión entre 0 y Vmax en la entrada **Vi** en códigos binarios de 4 bits entre 0h y 0Fh. Se construye conectando un convertidor **D/A** y un comparador analógico como se indica en el diagrama de bloques.



Para realizar una conversión, el **Bloque de Control** comienza poniendo a la entrada del **D/A** el código 1000B llevando por lo tanto la tensión de salida del **D/A** a la mitad del intervalo [0..Vmax]. Observando la salida del comparador, se puede determinar si **Vi** es mayor o menor que Vmax/2. En consecuencia puede saberse si Vi está en el intervalo [0..Vmax/2] correspondiente a códigos binarios 0xxx o en el intervalo [Vmax/2..Vmax] correspondiente a códigos binarios 1xxx.

De esta manera se determina el valor del MSB. Fijando el valor hallado para el MSB puede repetirse el procedimiento para determinar en que intervalo de tamaño Vmax/4 se encuentra Vi y así determinar el valor del segundo bit.

Iterando de esta manera, en 4 comparaciones sucesivas se completa la conversión.

La conversión debe comenzar cuando se produce un pulso a 1 en la señal **soc** de 1 período de reloj de duración.

Una vez finalizada la conversión el controlador debe poner el resultado en la salida **DATO** y llevar la señal **eoc** a 1 para indicar que hay un dato válido en la salida **DATO**. El valor en **DATO** y **eoc** debe mantenerse hasta el próximo flanco de bajada en **soc**.

Lo que se quiere diseñar es el bloque de control del convertidor. El ejemplo elegido se ajusta perfectamente a los requisitos para una descripción RTL.

Por tratarse de un convertidor por pasos sucesivos de aproximación, su funcionamiento se divide en pasos.

En cada paso se evaluará un bit del valor a convertir, por lo cual se tratará que el diseño de la parte de datos sea tipo bit-slice.

Se usará un vector **dato(I)** de 4 bits para guardar el resultado y en cada paso pueden darse una de estas tres posibilidades:

1) Si soc=1 dato(I) <= 0;
2) Si Paso_J=1 dato(I) <= comp; se carga con el resultado de la comparación
3) Si Paso_J=0 dato(I) <= dato(I); se mantiene el valor previo

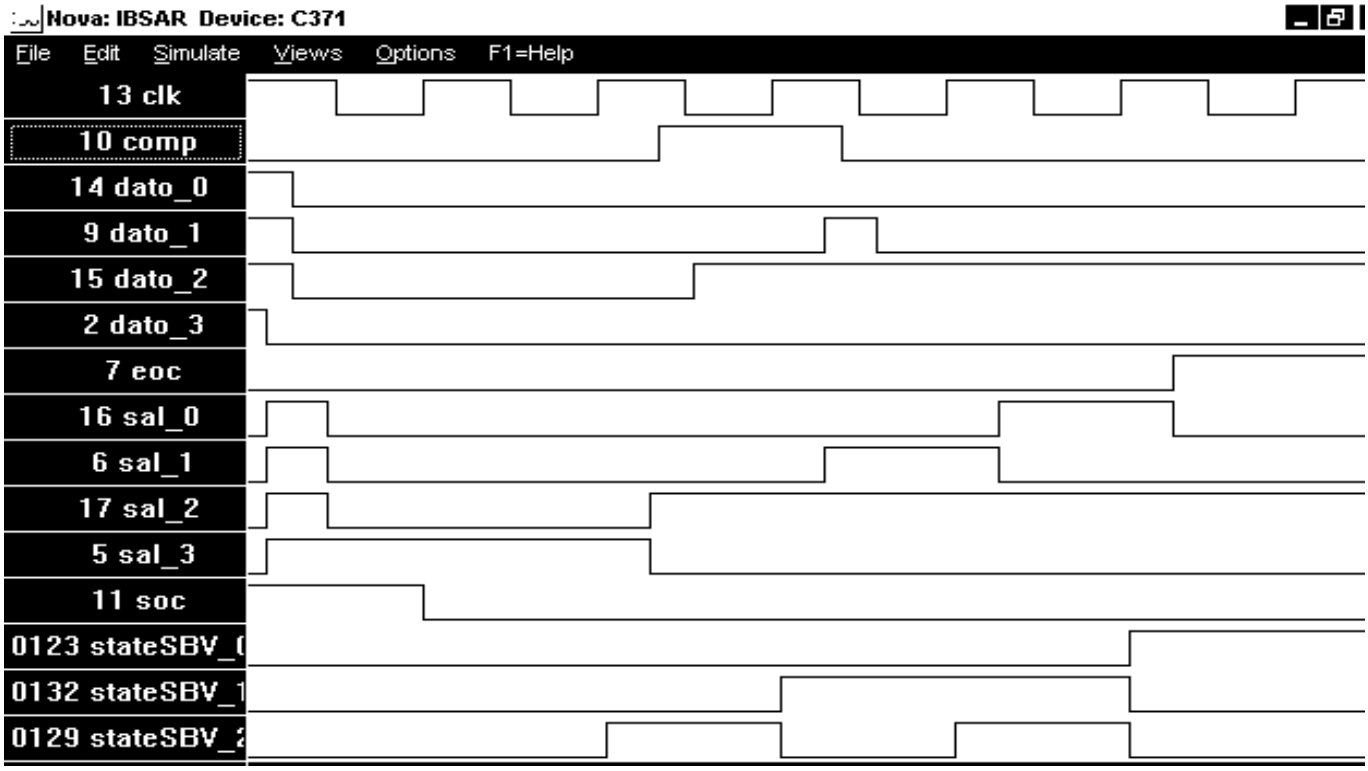
La descripción VHDL del circuito es una traducción inmediata de las especificaciones del diseño. Se usa una arquitectura con dos procesos para implementar la máquina de estados. Uno de los procesos es sensible al **clk** , mientras el segundo es sensible a los cambios de estado.

Dado que VHDL no posee la estructura STATE DIAGRAM, como es el caso del lenguaje ABEL por ejemplo, se recurre a una estructura CASE dentro de un PROCESS donde cada rama del CASE representa un estado de la máquina o un paso de la secuencia RTL.

```
--------------------------------------------------------------------------------
--Descripción del comportamiento de un registro de aproximaciones
--sucesivas de cuatro bits.
--------------------------------------------------------------------------------
LIBRARY IEEE;
use ieee.std_logic_1164.all;

ENTITY ras IS
      PORT(
          clk,soc,comp : in std_logic;
          dato: buffer std_logic_vector (3 downto 0);
          sal:   buffer std_logic_vector (3 downto 0);
          eoc: buffer std_logic
         );
END ras;


ARCHITECTURE comportamiento OF ras IS

type states is (paso_1,paso_2,paso_3,paso_4, final);
SIGNAL state, next_state : states;
```

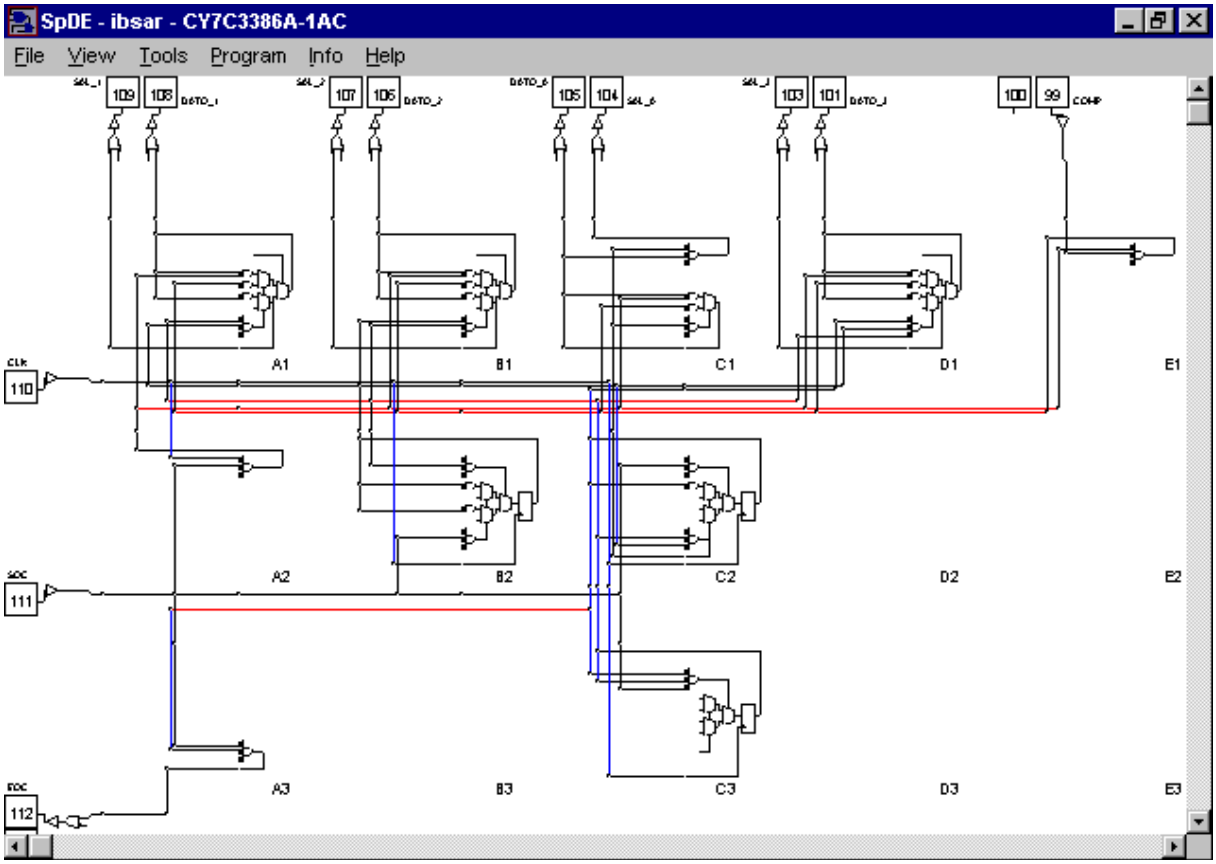
```
BEGIN

PROCESS
BEGIN
 WAIT UNTIL (clk'event and clk = '1');
 CASE soc IS
 WHEN '1' =>
      state <= paso_1;
 WHEN OTHERS =>
      state <= next_state;
 END CASE;
END PROCESS;

PROCESS(state)
BEGIN
  CASE state IS

  WHEN paso_1 =>
    next_state <= paso_2;
  WHEN paso_2 =>
    next_state <= paso_3;
  WHEN paso_3 =>
    next_state <= paso_4;
WHEN paso_4 =>
    next_state <= final;
WHEN final =>
    null;
END CASE;
END PROCESS;
--Asignaciones concurrentes de la máquina de estados


dato(3) <= '0' when ((soc='1') and (clk='1')) else
           '1' when (state=paso_1 and comp='1' ) else
           '0' when (state=paso_1 and comp='0' ) else
           dato(3);
sal(3)   <= '1' when (dato(3)='1' or state=paso_1) else
            '0';
dato(2)  <= '0' when (soc='1' and clk='1') else
            '1' when (state=paso_2 and comp='1') else
            '0' when (state=paso_2 and comp='0' ) else
             dato(2);
sal(2)   <= '1' when (dato(2)='1' or state=paso_2) else
            '0';
dato(1)  <= '0' when (soc='1' and clk='1') else
            '1' when (state=paso_3 and comp='1') else
```

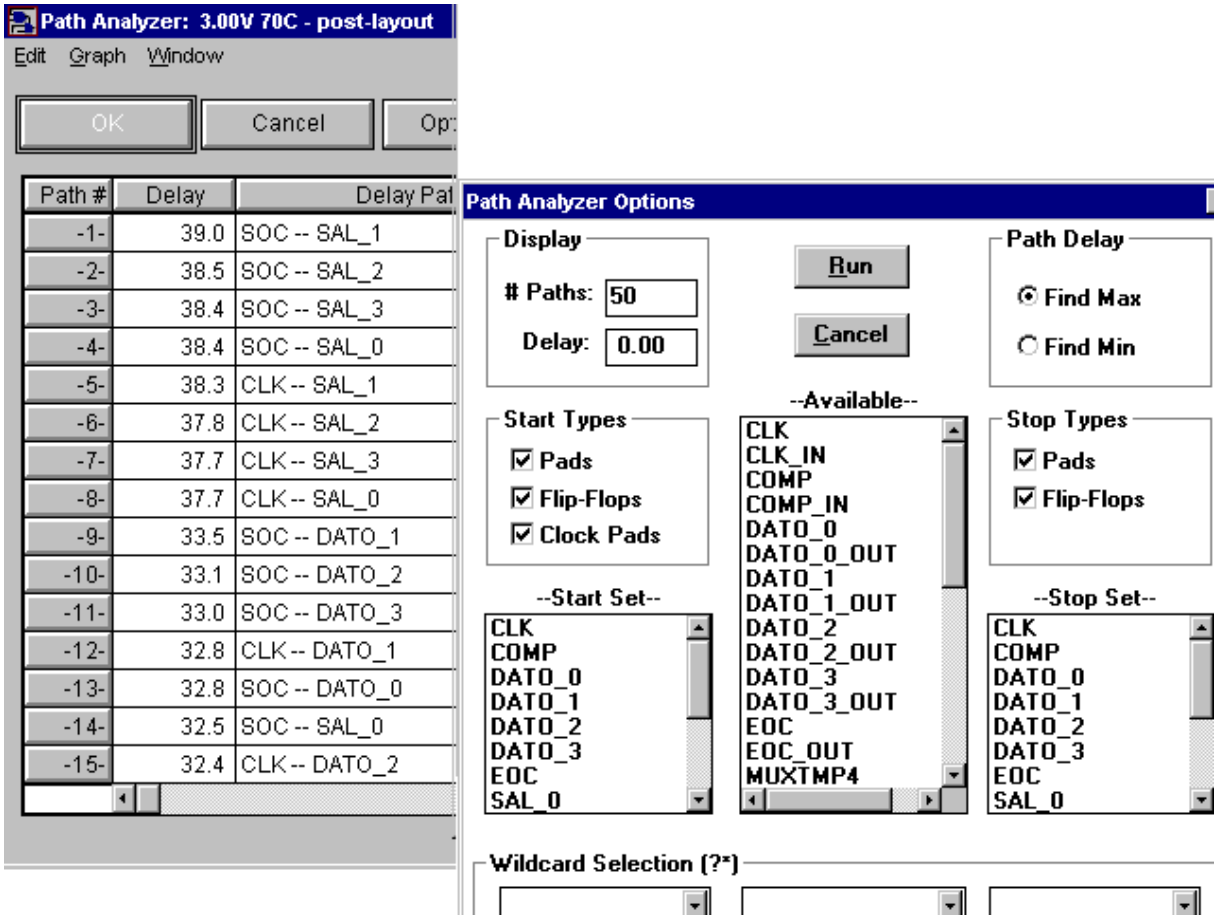
```
        '0' when (state=paso_3 and comp='0' ) else
           dato(1);
sal(1)   <= '1' when (dato(1)='1' or state=paso_3) else
        '0';
dato(0)  <= '0' when (soc='1' and clk='1') else
        '1' when (state=paso_4 and comp='1') else
        '0' when (state=paso_4 and comp='0' ) else
        dato(0);
sal(0)   <= '1' when (dato(0)='1' or state=paso_4) else
        '0';
eoc <=    '1' when (state=final and soc='0') else
       '0';
END comportamiento;
```



Síntesis.-

Con esta descripción ya se puede realizar la síntesis del circuito en lógica programable o celdas standard. Como ejemplo de esta variante se realizó la síntesis del circuito en una FPGA usando WARP de Cypress Semiconductor. Y las simulaciones se hicieron con Viewlogic (con algunos agregados y cambios en la sintaxis.)


SpDE - ibsar - CY7C3386A-1AC
File View Tools Program Info Help
109
108
107
106
105
104
103
101
100
99
A1
B1
C1
D1
E1
110
A2
B2
C2
D2
E2
111
A3
B3
C3
D3
E3
112

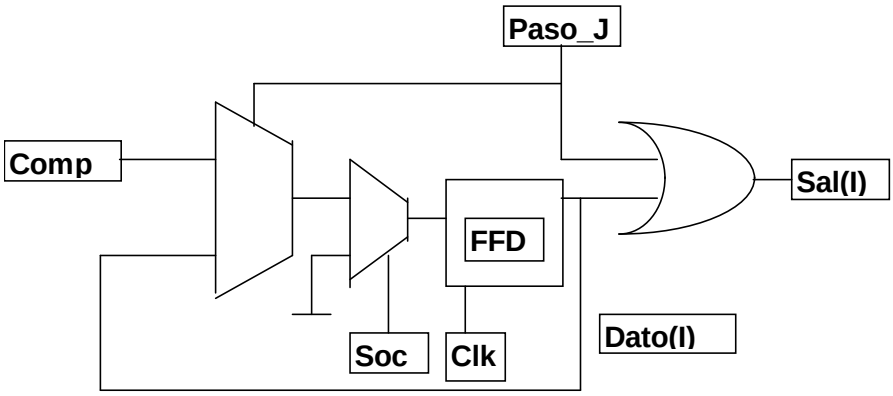


La síntesis del circuito para su realización full-custom, consistente en obtener una descripción estructural del mismo, es inmediata ya que la parte de control consiste de un registro de desplazamiento y la parte de datos son cuatro bloques iguales que implementan la función:

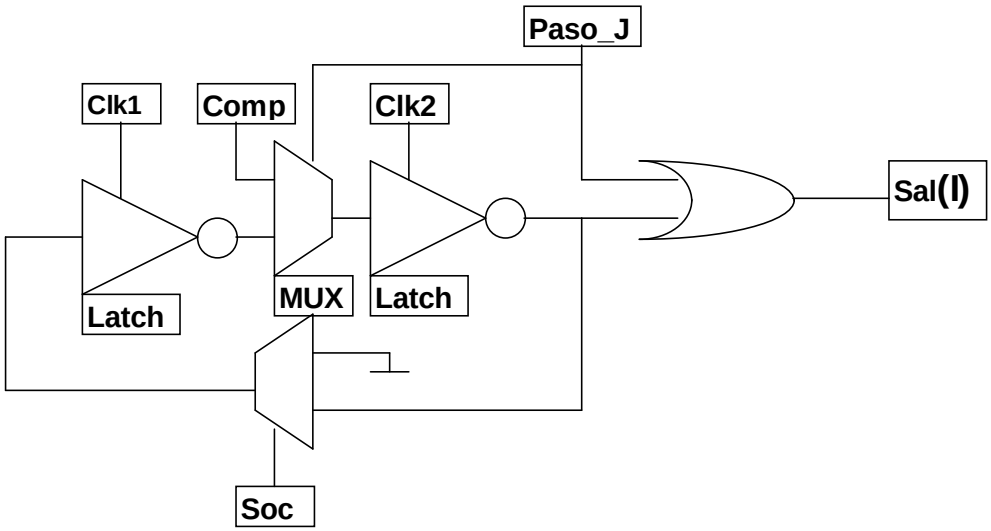
```
dato(I)  <= '0' when (soc='1' and clk='1') else
            '1' when (state=paso_I and comp='1') else
            '0' when (state=paso_I and comp='0' ) else
             dato(I);

sal(I)   <= '1' when (dato(I)='1' or state=paso_J) else
               '0';
```

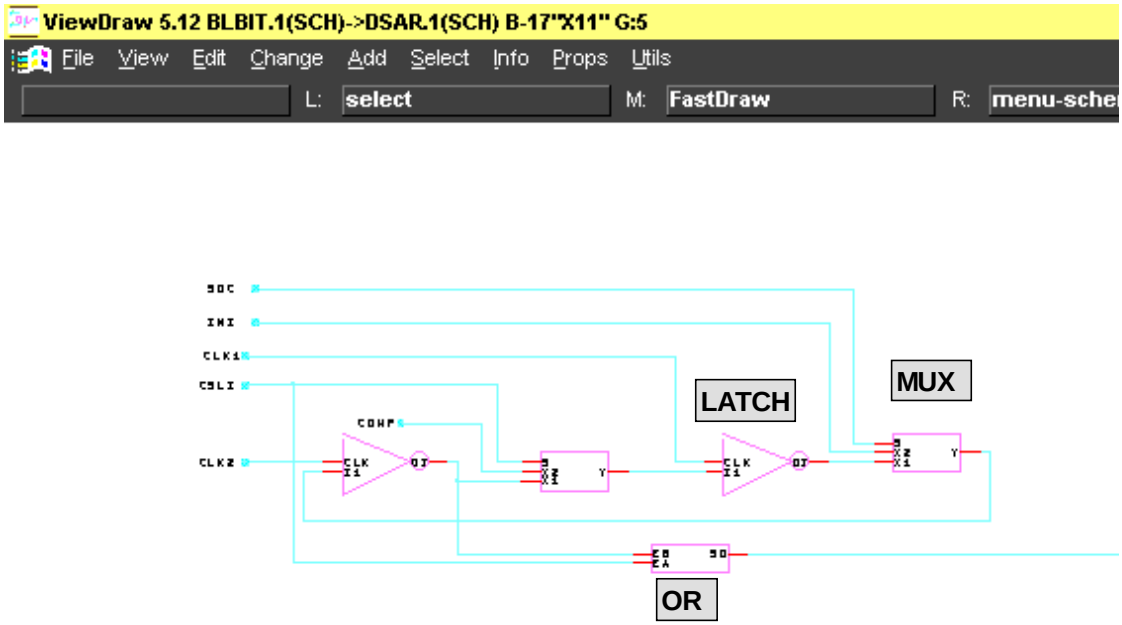
La figura muestra una síntesis posible de estos bloques implementado con un reloj de una fase. La descripción VHDL estructural se realizó usando la herramienta Viewdraw de Viewlogic en dos modalidades:

1) para celdas standard usando el Kit de diseño de ES2.
2) usando descripciones VHDL de comportamiento para los bloques FFD, multiplexor y puertas.



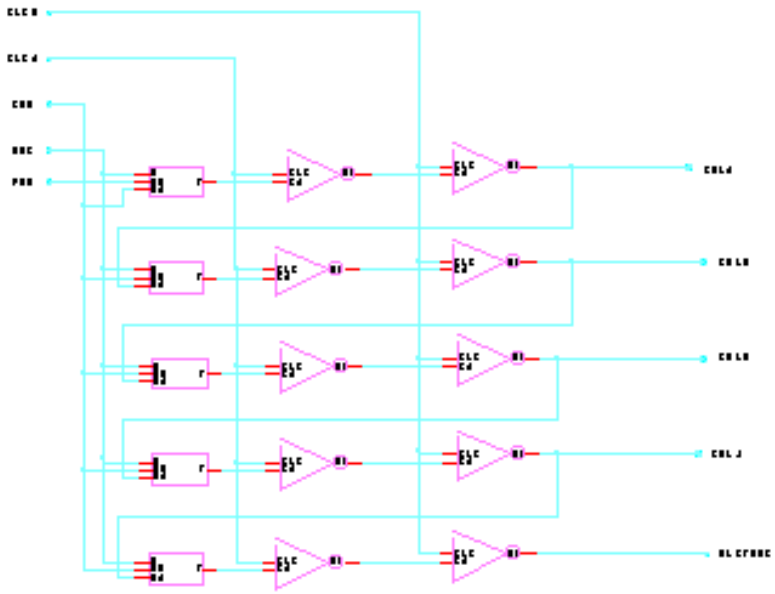
Otra solución estudiada consistió en armar el bloque usando lógica dinámica con un reloj de dos fases para implementar los Flip Flops y transistores de paso para los multiplexores. La compuerta or de salida se realiza en lógica inversora. La parte de control será un registro de desplazamiento de 5 bits realizado con lógica dinámica y el mismo reloj de dos fases. La descripción estructural VHDL y las simulaciones lógicas se hicieron también usando Viewlogic.

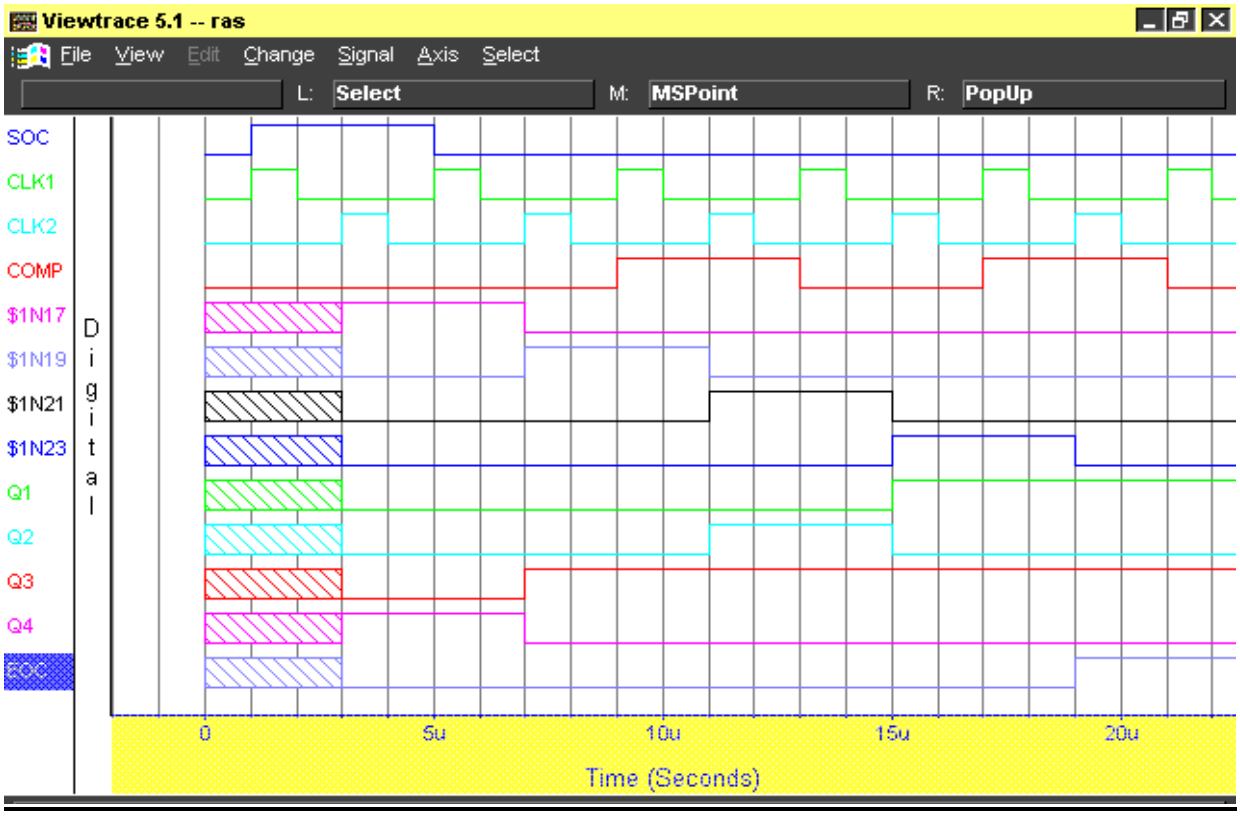





```
ENTITY latch IS
       PORT( i1,clk: in vlbit ;  oi: out vlbit );
END latch;
ARCHITECTURE complatch OF latch IS
BEGIN
PROCESS(clk, i1)
BEGIN
    if clk='1' then
       oi <= NOT ( i1 );
    else
       oi <= 'X' after 2 ms;
    end if;
END PROCESS;
END complatch;

ENTITY mux21 IS
   PORT (
   x1   : in vlbit;
   x2   : in vlbit;
   y    : out vlbit;
   s    : in  vlbit);
END mux21;


ARCHITECTURE behavior OF mux21 IS
BEGIN
WITH s SELECT
  y <= x2   WHEN '1',
       x1   WHEN '0',
```

```
        'X'   WHEN OTHERS;
END behavior;
```





Conclusiones.-

VHDL se muestra especialmente útil en las primeras etapas del diseño, a la hora de elegir arquitecturas, particiones del circuito, tipos de lógica, etc y para la realización de simulaciones a nivel funcional. Aunque es de hacer notar que las descripciones apropiadas para simulación no siempre son aptas para la síntesis.

También permite la documentación del proyecto a un nivel que es independiente de la tecnología e incluso del tipo de lógica, lo cual permite su re utilización en otros diseños.

El trabajo trata de resaltar la importancia del método, que en proyectos pequeños no se aprecia pero a medida que se incrementa la complejidad del diseño va adquiriendo mas importancia.

Bibliografía.-

Básica.-

1 Neil H. E. Weste - Kamran Eshraghian.
Principles of CMOS VLSI design.
A systems Perspective.
Second Edition.
Addison-Wesley
Massachusetts/California/New York/Ontario/Etc.

2 Manuales de Viewlogic.

De consulta.-

1 Douglas L. Perry
VHDL
Second Edition
McGraw-Hill
New York/San Francisco/Washington D.C./Bogotá/Etc.

2 Roger Lipsett, Carl F. Schaefer y Cary Ussery.
VHDL: Hardware Description and Design.
Kubler Academic Publishers
Boston/Dordrecht/London

3 Kevin Skahill (Cypress Semiconductor)
VHDL for PROGRAMMABLE LOGIC
Addison Wesley
Massachusetts - Menlo Park, California - New York - Harlow, England, etc.