UNIVERSIDAD AUTONOMA METROPOLITANA

UNIDAD AZCAPOTZALCO

DIVISIÓN DE CIENCIAS BÁSICAS E INGENIERÍA

DEPARTAMENTO DE ELECRÓNICA

INSTRUCTIVO PARA EL USO DEL PROGRAMA QUARTUS2 V7.2 DE ALTERA

DR. ISAAC SCHNADOWER BARÁN

MARZO DE 2011

ÍNDICE

INTRODUCCIÓN

Se presenta un instructivo para el uso del programas Quartus II V7.2, que se utiliza en cursos de diseño lógico, arquitectura de computadoras y para la tablilla de desarrollo DE2 de Altera.

El instructivo resume las principales operaciones para obtener producir, compilar y simular programas VHDL, de acuerdo con el manual de usuario y tutorial que acompaña al programa Quartus II, a consultarse en HELP.

CAPÍTULO 1. CREACIÓN DE UN PROYECTO EN VHDL

Para abrir Quartus II, pulse sobre su ícono. Aparece entonces la pantalla mostrada en la Figura 1. Si aparecen otras pantallas auxiliares, puede cerrarlas.

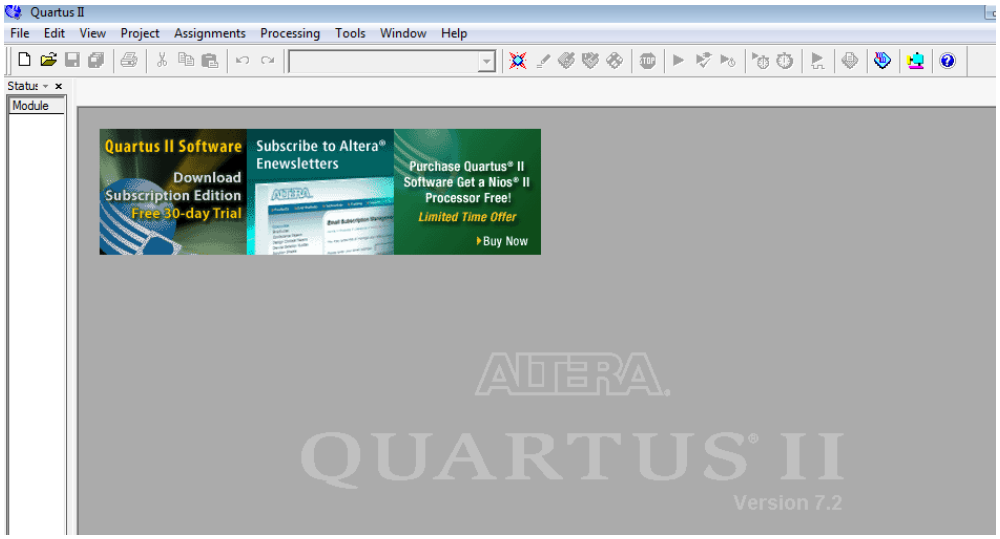


Figura 1. Pantalla de inicio.

Pulse **File→Open project** para abrir un proyecto existente, o bien **File→New Project Wizard** para generar un nuevo proyecto. En este caso, se abre la ventana de la Figura 2.

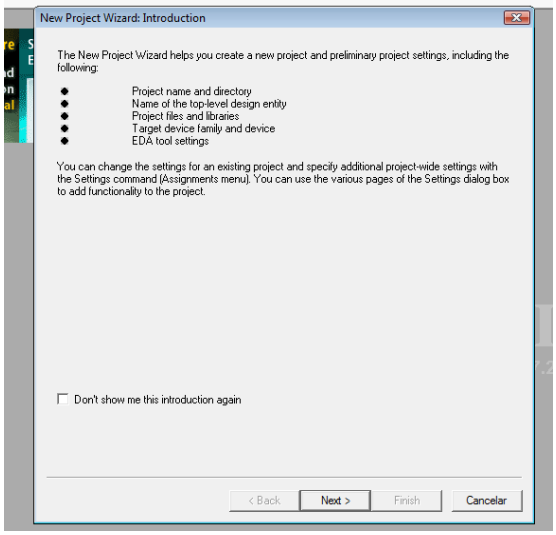


Figura 2. Pantalla de inicio de un nuevo proyecto.

Pulse **Next**, y aparece la ventana de la Figura 3. **Todo proyecto requiere un directorio** (carpeta en Windows), así que presione el ícono de exploración (…) para crear su ruta y nombre. En nuestro caso es C:\logicos\sum1. El nombre sum1 aparecerá también en nombre del proyecto y en la entidad top. Pulse ahora **Next**, y aparece la ventana de La Figura 4.

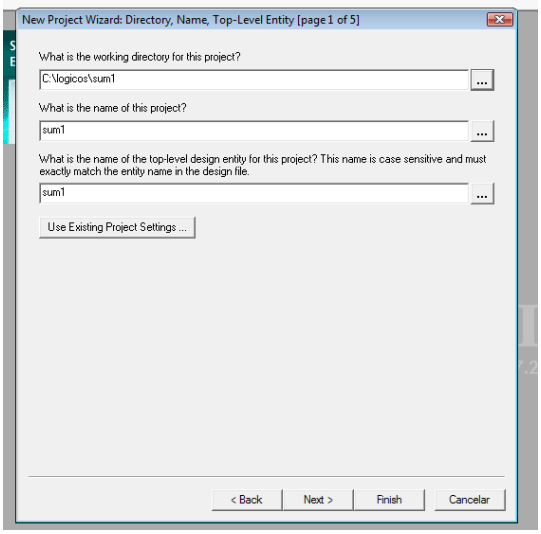


Figura 3. Asignando directorio para el proyecto sum1.

Puede insertar archivos de otros proyectos en el directorio del proyecto presente. En este caso no añadimos ninguno, pulsamos Next y aparece la ventana de la Figura 5.

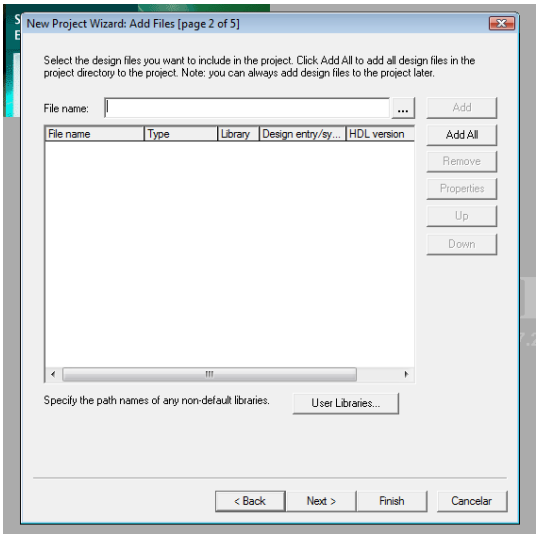


Figura 4. Ventana para añadir otros archivos.

En la ventana de la Figura 5 se indica el dispositivo en el cual se implementará el diseño. La tablilla DE2 de desarrollo cuenta con un dispositivo Cyclone II. Indicarlo así. Pulsamos ahora **NEXT** 2 veces para finalizar con el resumen del proyecto, como en la Figura 6.

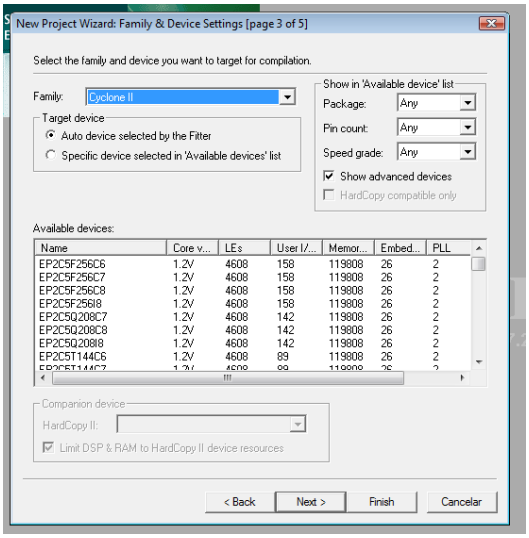


Figura 5. Selección del dispositivo FPGA a utilizar.

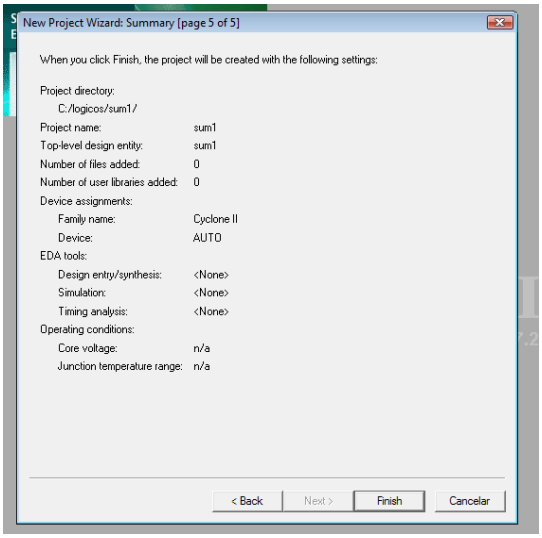


Figura 6.- Resumen del proyecto.

El siguiente paso es crear un archivo de diseño. Pulsamos **File→New**, para abrir la ventana de la Figura 7. Escogemos diseñar **por medio de VHDL**; pulsamos VHDL, y se abre la ventana de trabajo de la Figura 8. En ésta se teclea el código del programa sum1.

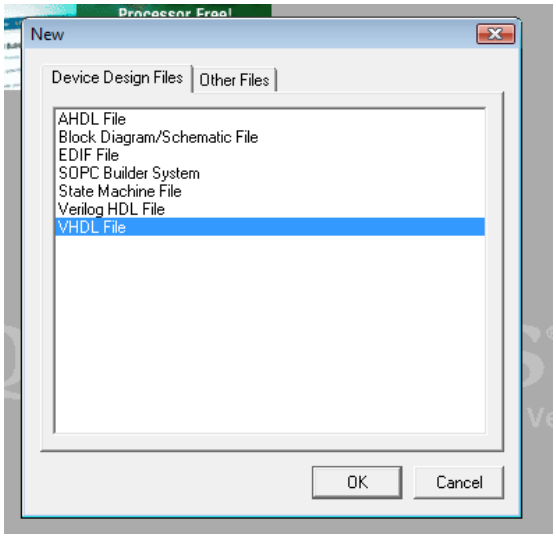


Figura 7. Selección del tipo de archivo.

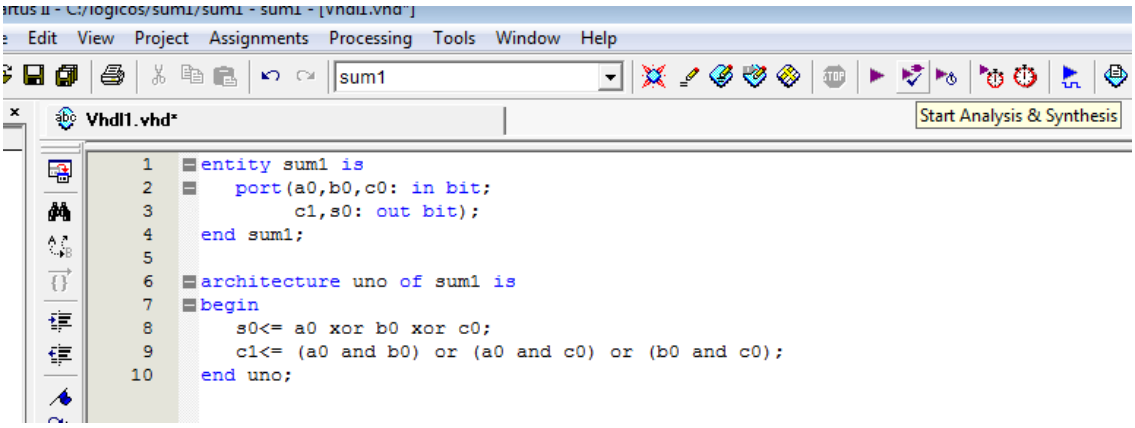


Figura 8. Creación del archivo del programa sum1.

El programa debe ahora compilarse; pulsar **Processing→Compiler tool.** Aparece la ventana de la Figura 9, la herramienta de compilación

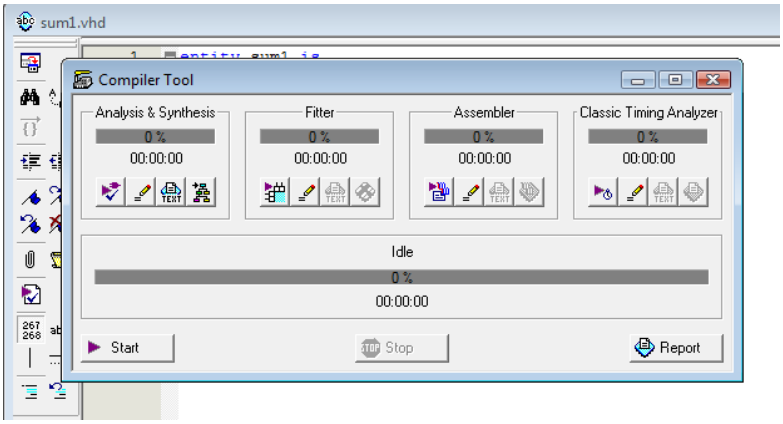


Figura 9. Herramienta de compilación.

Para iniciar la compilación, Pulsar **Start.** Otra forma es pulsar el ícono que aparece en la barra de herramientas justo arriba del letrero “Start Analysis &Synthesis” de la Figura 8. De no existir errores, aparece la ventana de la Figura 10 con el mensaje de éxito, un resumen y otros mensajes en la parte inferior. De lo contrario, aparecen mensajes de error. Al pulsar sobre ellos, se resaltan las líneas que lo generan, para facilitar su corrección.

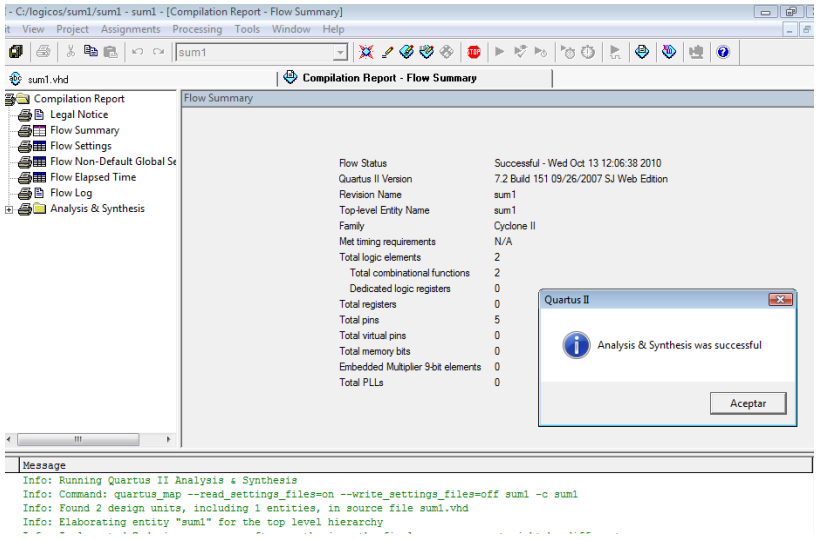


Figura 10. Reporte de una compilación exitosa.

Para visualizar el circuito sintetizado, pulse **Tools→Netlist Viewers→ RTL Viewer**. Aparece la ventana de la Figura 11. Ud. podrá reconocer el circuito de un sumador.

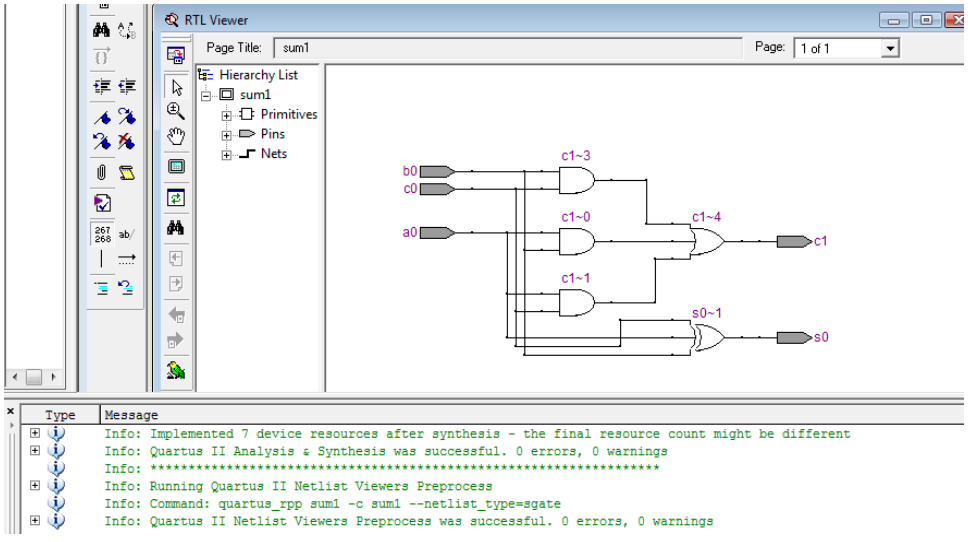


Figura 11. Diagrama RTL del circuito sintetizado por el programa sum1.

Una compilación exitosa no garantiza un funcionamiento satisfactorio, pues podrían existir errores de lógica. Por ello es conveniente simular el comportamiento del circuito. Pulse ahora **Processing→Simulation Tool.** En **Simulation Mode** seleccione **Functional** en vez de **Timing**, y pulse **Generate Functional Simulation Netlist.** De no existir errores, aparecerá el mensaje de éxito de la Figura 13.

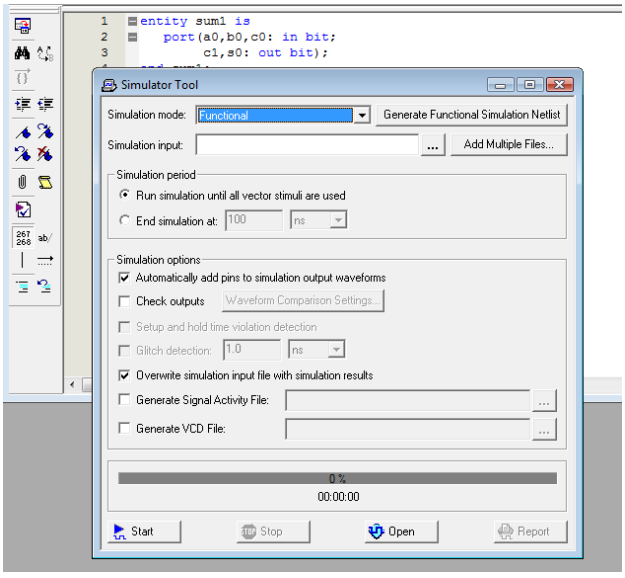


Figura 12. Herramienta de simulación.

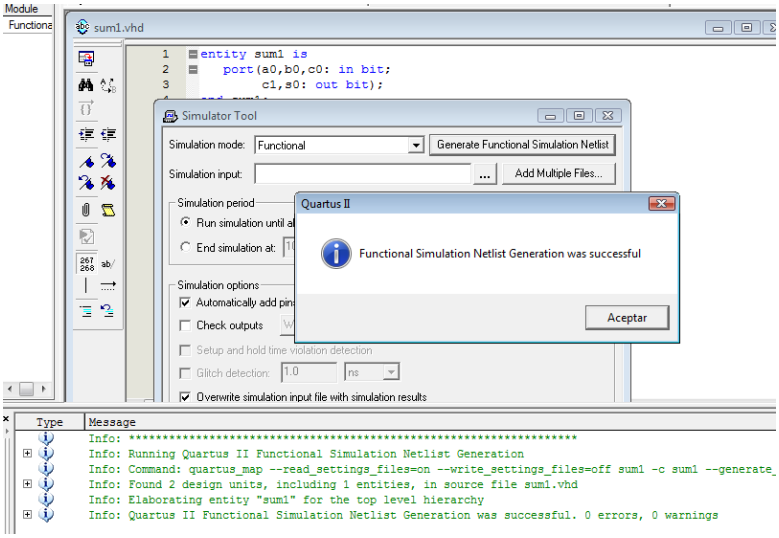


Figura 13. Ventana correspondiente a una generación de lista de nodos exitosa.

Para facilitar la visualización de la simulación, marque el cuadro **“Overwrite simulation input file with simulation results”**. Pulse **Open** para abrir la ventana de forma de ondas (Figura 14).

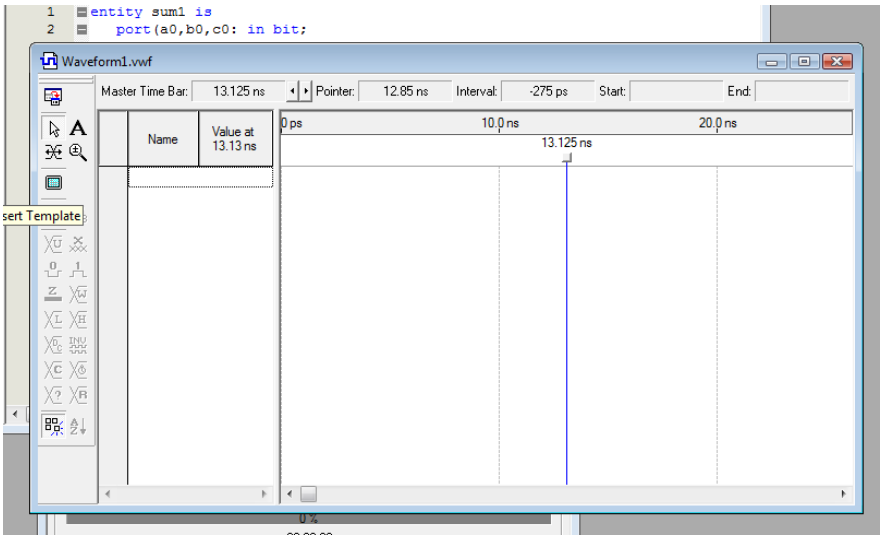


Figura 14. Ventana de formas de onda para simulación.

Requerimos ahora insertar las señales de entrada y salida. Pulse **Edit→Insert Node or Bus**. En la Figura 15 puede Ud. seleccionar la base numérica, el nombre de la señal (si desea insertarlas una por una), o bien desplegar los nombres de todas las señales de la entidad si pulsa **Node Finder.**

Esto es más sencillo, sobre todo si existen mucha señales. Aparece entonces la ventana de la Figura 16.

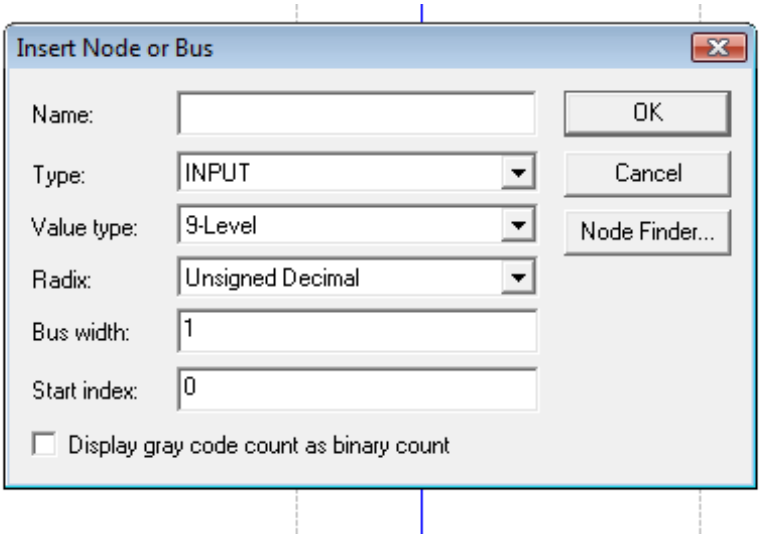


Figura 15. Selección de señales a insertar.

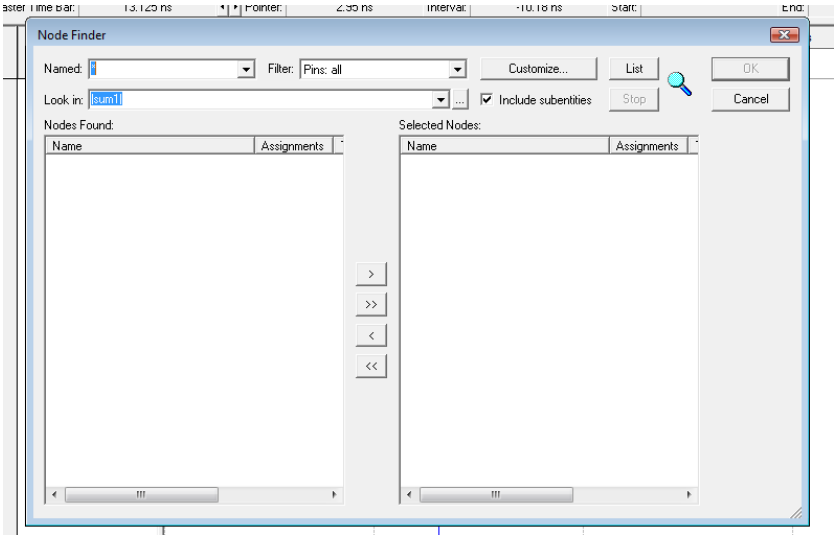


Figura 16. Ventana para enlistar todos los nodos.

En la Ventana seleccionamos **Pins:all,** y pulsamos **List.** Aparecen entonces **todos los nodos** del proyecto (señales de la entidad). Los podemos seleccionar en cualquier orden mediante la flecha >, o todos simultáneamente mediante >> (Copy all to Selected Nodes list). Los nodos seleccionados aparecen en el lado derecho de la ventana (Figura 17) y resta tan sólo pulsar **OK.**

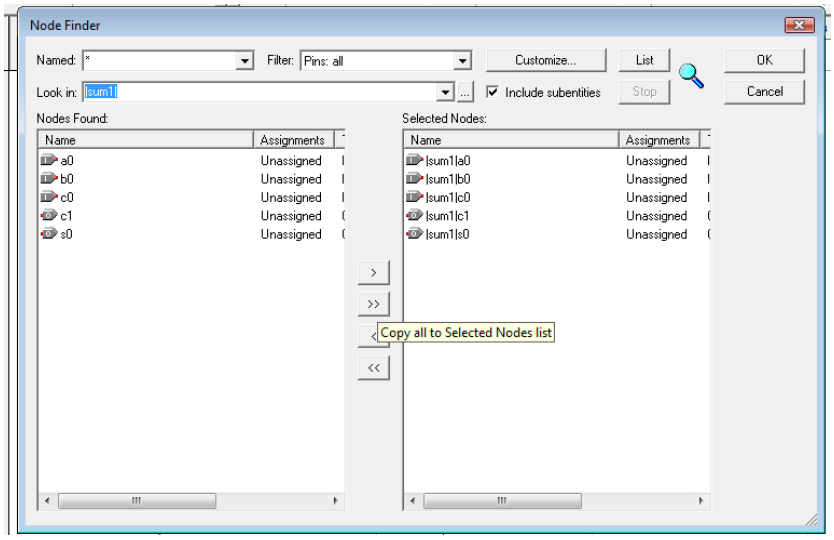


Figura 17. Lista de nodos de sum1 seleccionados.

La ventana de forma de onda incluye ahora los nodos seleccionados. Los de entrada (a0,b0 y c0) listos para recibir valores (todos en cero originalmente), y los de salida indefinidos (Figura 18). Los valores de las entradas pueden ahora asignarse señalando segmentos con el ratón y marcando los íconos correspondientes a 0 o 1 colocados a la izquierda de la ventana. (parte inferior), que se aprecian en la Figura 19.

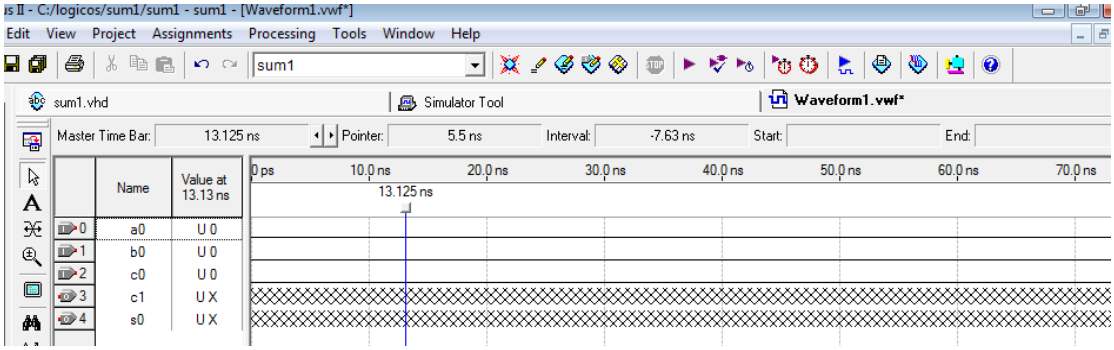


Figura 18. Formas de onda previas a la simulación.

Una vez que se han asignado valores a los nodos de entrada, procedemos a la simulación, pulsando el ícono **Start** de la herramienta de simulación (Figura 12) o el correspondiente que se encuentra en la barra de herramientas de Quartus II. Dado que está marcado en la herramienta el cuadro de sobreescritura, aparecerá el cuadro de diálogo de la Figura 19; pulse **Sí,** y se obtiene la hoja de resultados de la Figura 20, en la cual se verifica el comportamiento del circuito.

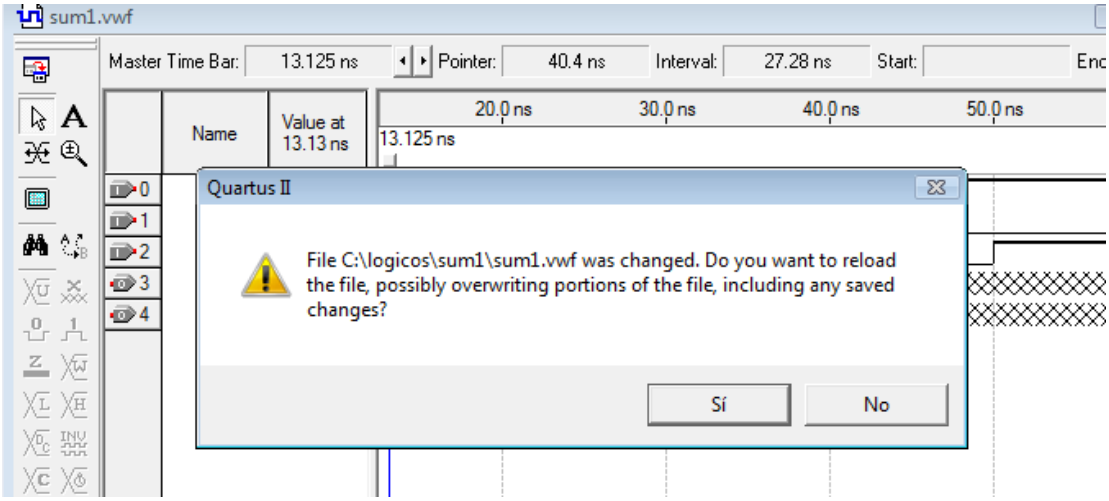


Figura 19. Ventana de sobreescritura.

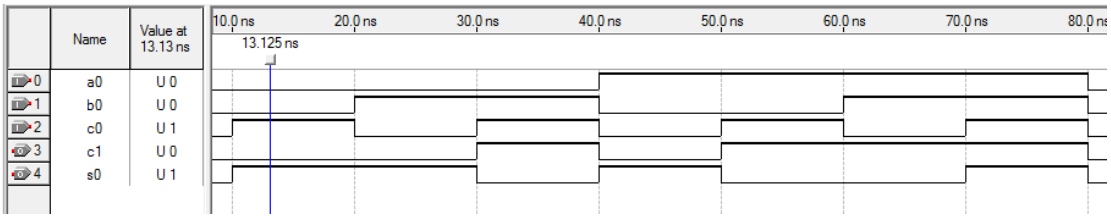


Figura 20. Resultados de la simulación de sum1.

En caso de que se desee efectuar otras simulaciones, crear otras hojas al pulsar **File→New,** seleccionar **Other Files** y **Vector Waveform File.** (Figura 21).

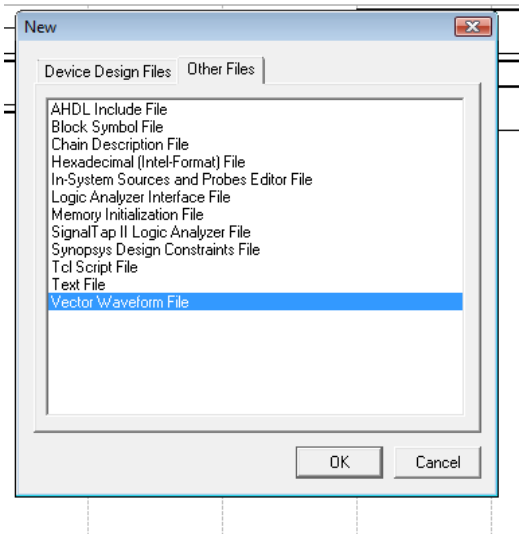


Figura 21. Selección de archivo de formas de onda.

CAPÍTULO 3. USO DE COMPONENTES

Veamos ahora el caso en que un proyecto utiliza componentes que fueron sintetizadas en otro proyecto. Por ejemplo, un sumador de 4 bits, sum4, que utiliza 4 instancias de sum1. En el directorio de sum4 se requiere el código compilado de la componente sum1. Este se puede agregar desde el inicio (Fig. 4), o bien después, pulsando **Project→Add/remove Files in Project.** Se abre la misma ventana de la Figura 4; se utiliza la exploración para localizar el proyecto por añadir, sum1, se abre y se pulsa **Add**, y **OK.** Ver la Figura 22.

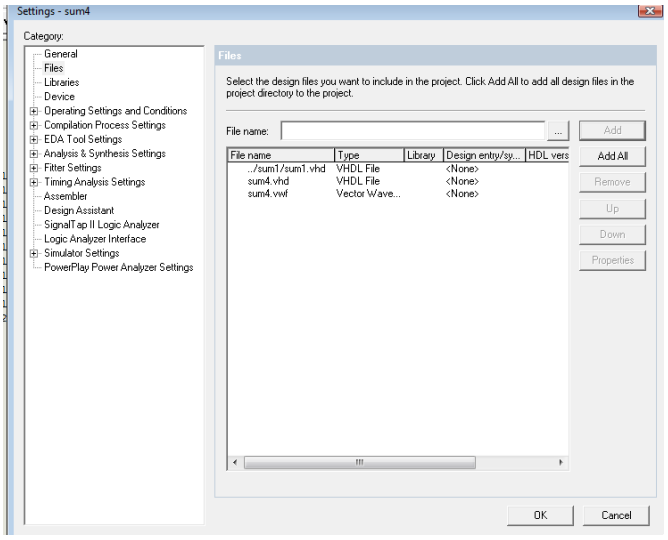


Figura 22. Adición del archivo VHDL de sum1 al proyecto sum4.

Creando un símbolo. El símbolo de un proyecto corresponde con la caja negra descrita por su entity. Dicho símbolo puede utilizarse para el diseño de un proyecto mayor que lo contiene mediante un esquema. Para obtener el símbolo, abrir **File→Create/Update→Create Symbol Files For Current File. Quartus responde con la ventana de la Figura 23.**

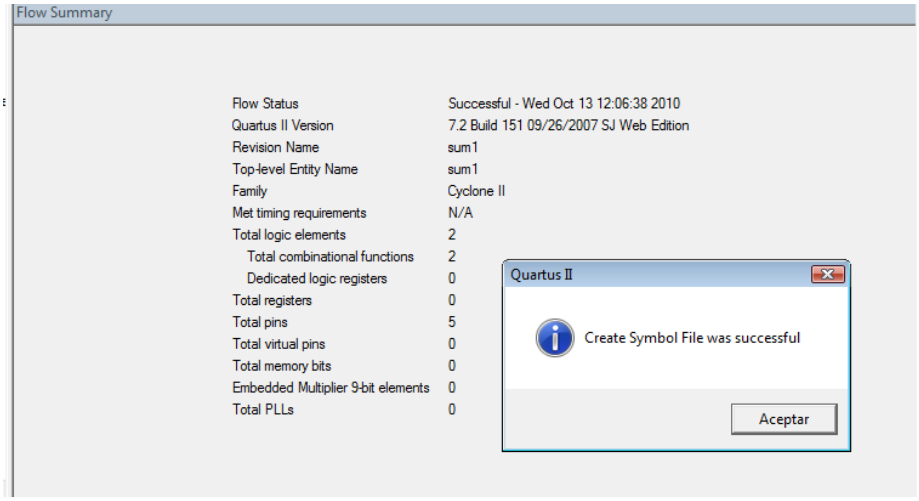


Figura 23. Notofocación de éxito por la creación de un símbolo.

Necesitamos ahora crear un archivo para guardar el símbolo. Pulsar **File→New→Block Diagram/Schematic File** (Figura 24). Se abre ahora una ventana para captura esquemática como la de la Figura 25.

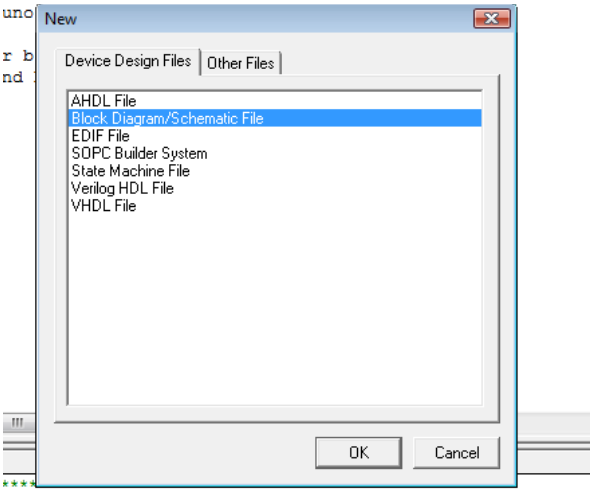


Figura 24. Selección de un archivo de bloques y esquemático.

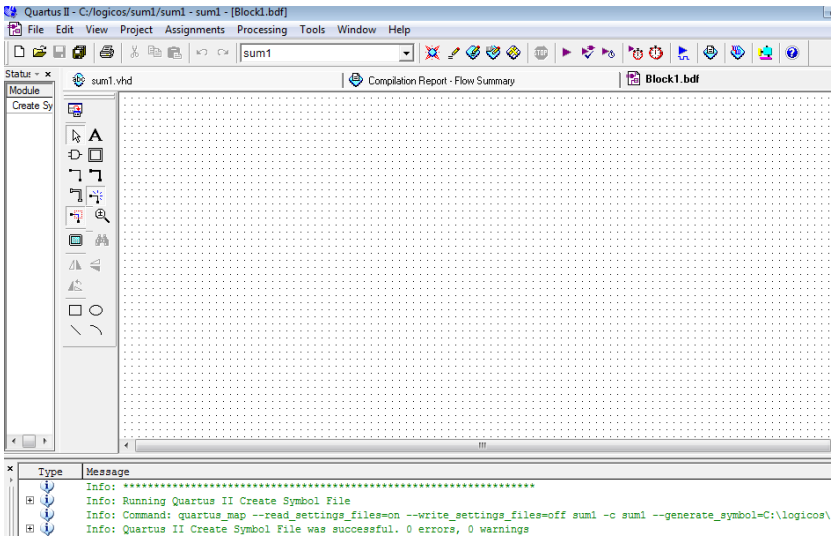


Figura 25. Ventana para captura esquemática.

Pulsamos ahora **Edit/Insert Symbol.** Se superpone entonces otra ventana para seleccionar el proyecto (en este caso, sum1), como en la Figura 26. En la figura aparece el símbolo de sum1 al seleccionar sum1 debajo de Project. Pulse ahora **OK**, y guarde el archivo. En nuestro caso, block1.

Este se inserta ahora en la carpeta del proyecto, como un archivo de diseño junto con el original del tipo VHDL.

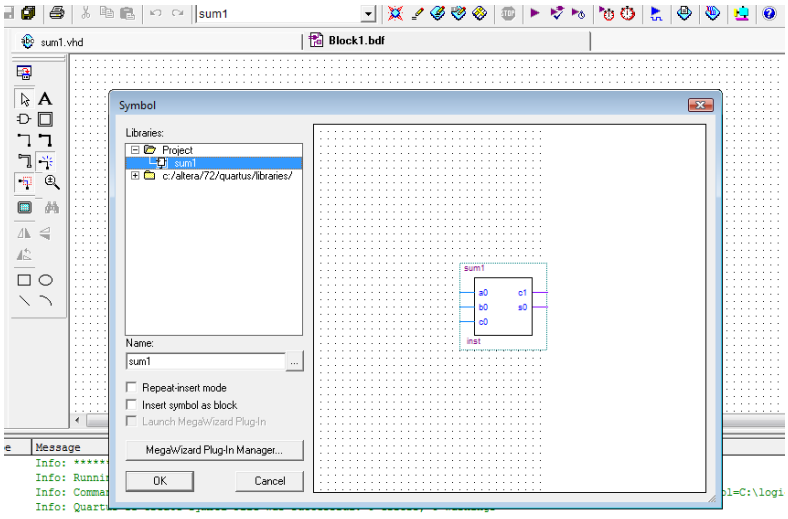


Figura 26. Selección del símbolo del proyecto sum1.

Quartus II incluye una librería de módulos LPM (*Library of parameterized modules*) de diferentes tipos, que constituyen diseños optimizados y que puede invocarse por los usuarios. Una manera es mediante el asistente **Tools→Mega Wizard Plug in Manager.** Se abre así la ventana de la Figura 27. Seleccionar el cuadro de creación como se muestra.

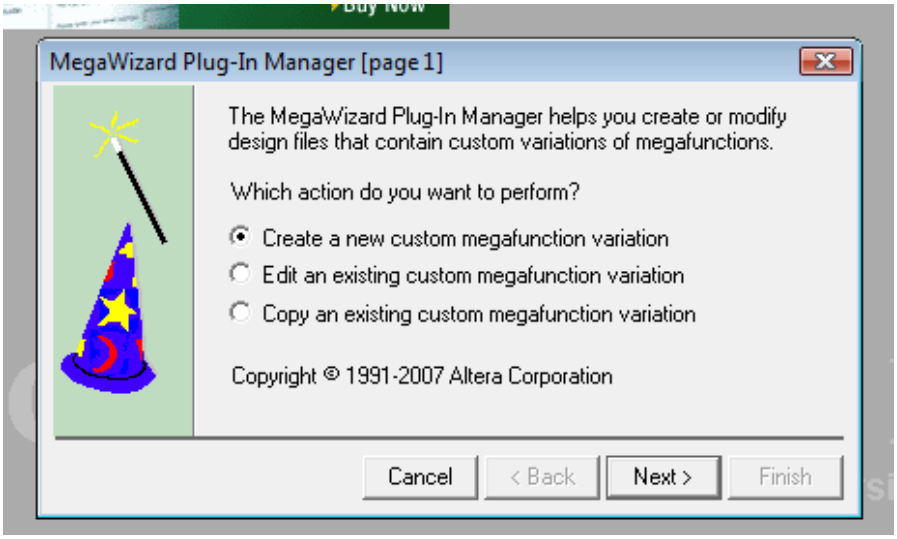


Figura 27. Creación de una megafunción.

Pulse **Next,** y se obtiene la ventana de la Figura 28. En ésta seleccionamos el dispositivo (Cyclone II), el tipo de archivo por crear (VHDL), el tipo de función (Arithmetic), el módulo LPM_ADD_SUB, el nombre del archivo donde se almacenará el código (sumrest8) para

obtener un circuito de suma y resta de 8 bits (Figura 29). Este archivo, sumrest8, se almacena en un directorio creado con anterioridad, como C:\logicos\sumrest8\.

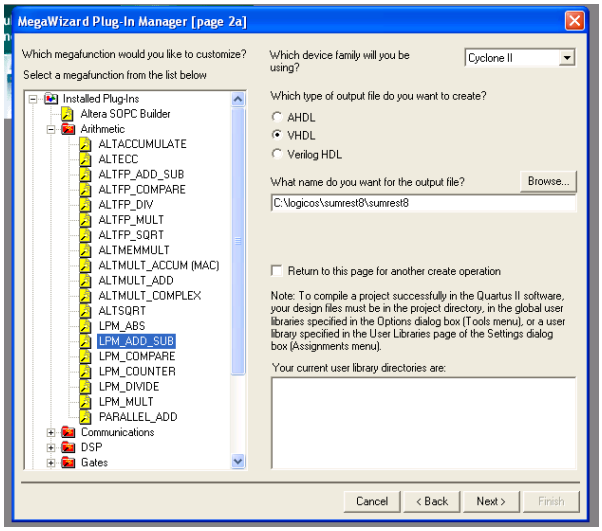


Figura 28. Selección del módulo por crear.

Pulse **Next,** y se abre así la ventana de la Figura 29, que muestra las opciones parametrizables (suma, resta, suma y resta), número de bits. En las siguientes ventanas que aparecen después de pulsar **Next**, seleccionamos más opciones (Valores fijos o variables, acarreos, sobreflujo, pipeling(n0), así como una lista de archivos por generar deseados(Figura 30 a 33).

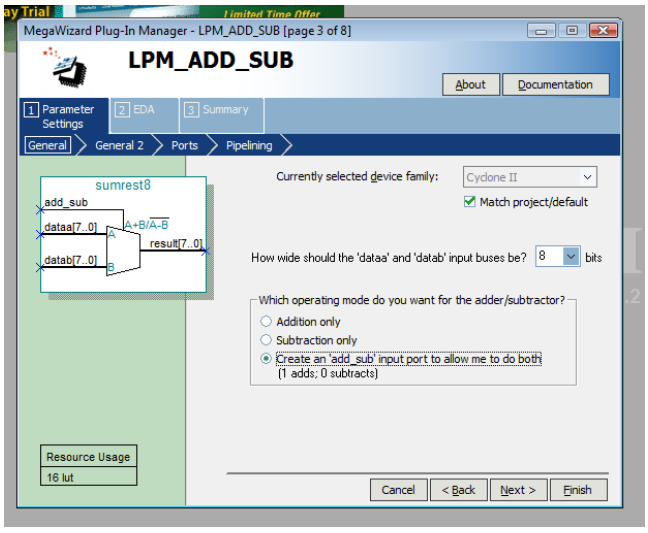
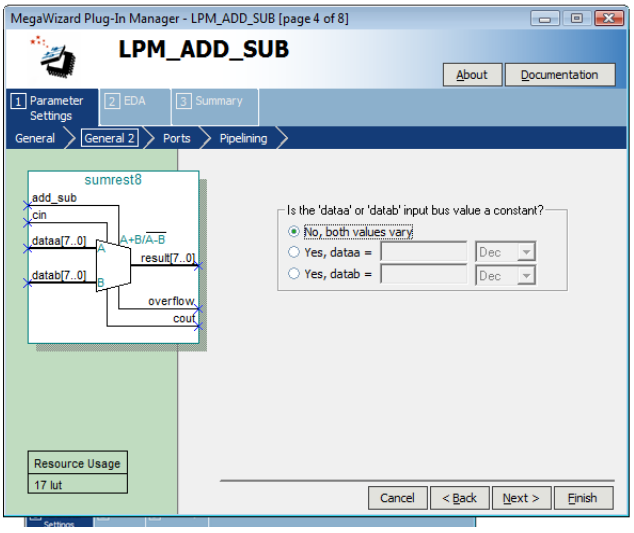
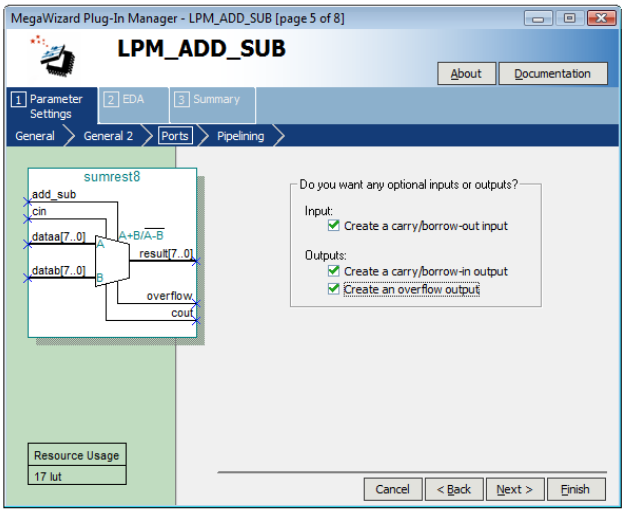


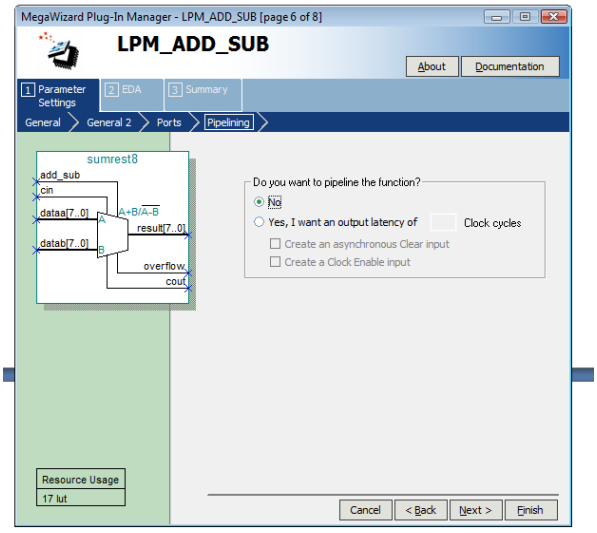
Figura 29.



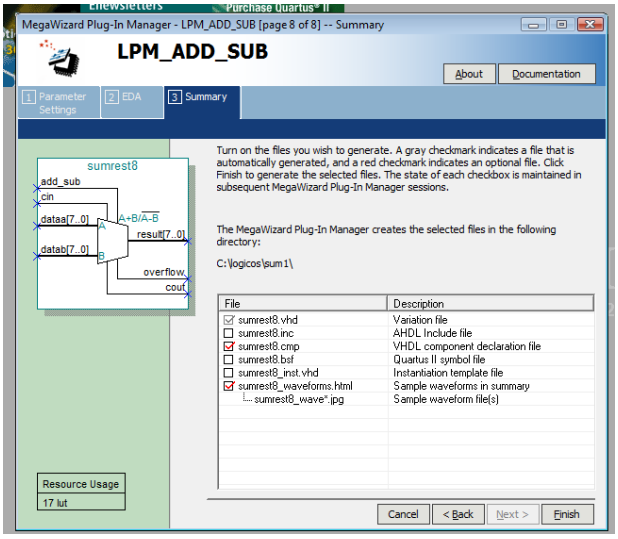
Figuras 30. Opciones del sumador –restador.



Figuras 31. Opciones del sumador –restador.



Figuras 32. Opciones del sumador –restador.



Figuras 33. Opciones del sumador –restador.

Al pulsar **Finish,** se crea el código de sumrest8, que utiliza la componente lpm_add_sub construída en la librería lpm de Quartus II. Este código se abre a partir del directorio, como se muestra en la Figura 35.

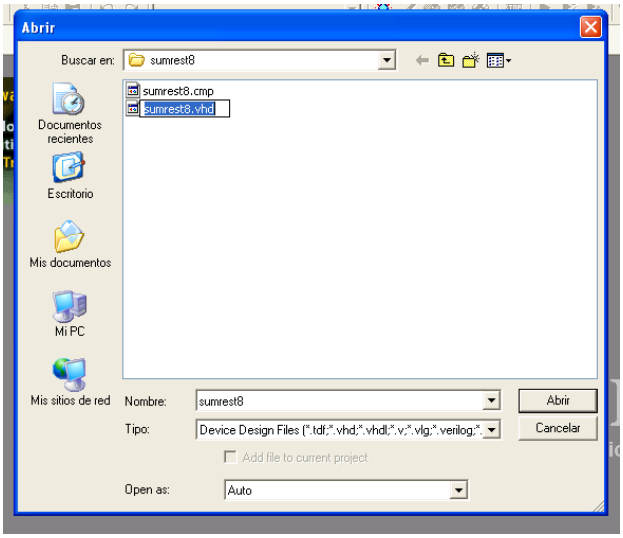


Figura 35.

El código generado, con los comentarios suprimidos, se muestra en la Figura 36 .

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
LIBRARY lpm;
USE lpm.all;
ENTITY sumrest8 IS
    PORT
    (add_sub       : IN STD_LOGIC ;
        cin      : IN STD_LOGIC ;
        dataa       : IN STD_LOGIC_VECTOR (7 DOWNTO 0);
        datab       : IN STD_LOGIC_VECTOR (7 DOWNTO 0);
        cout        : OUT STD_LOGIC ;
        overflow        : OUT STD_LOGIC ;
        result      : OUT STD_LOGIC_VECTOR (7 DOWNTO 0));
END sumrest8;
```

```
ARCHITECTURE SYN OF sumrest8 IS
    SIGNAL sub_wire0    : STD_LOGIC ;
    SIGNAL sub_wire1    : STD_LOGIC ;
    SIGNAL sub_wire2    : STD_LOGIC_VECTOR (7 DOWNTO 0);
    COMPONENT lpm_add_sub
    GENERIC (
        lpm_direction          : STRING;
        lpm_hint       : STRING;
        lpm_type       : STRING;
        lpm_width          : NATURAL
    );
    PORT (
            dataa  : IN STD_LOGIC_VECTOR (7 DOWNTO 0);
            add_sub   : IN STD_LOGIC ;
            datab  : IN STD_LOGIC_VECTOR (7 DOWNTO 0);
            overflow   : OUT STD_LOGIC ;
            cin : IN STD_LOGIC ;
            cout    : OUT STD_LOGIC ;
            result  : OUT STD_LOGIC_VECTOR (7 DOWNTO 0));
    END COMPONENT;
BEGIN
    overflow    <= sub_wire0;
    cout    <= sub_wire1;
    result    <= sub_wire2(7 DOWNTO 0);
    lpm_add_sub_component : lpm_add_sub
    GENERIC MAP (
        lpm_direction => "UNUSED",
        lpm_hint => "ONE_INPUT_IS_CONSTANT=NO,CIN_USED=YES",
        lpm_type => "LPM_ADD_SUB",
        lpm_width => 8
    )
    PORT MAP (
        dataa => dataa,
        add_sub => add_sub,
        datab => datab,
        cin => cin,
        overflow => sub_wire0,
        cout => sub_wire1,
        result => sub_wire2)

END SYN;
```

Figura 36. Código generado para sumrest8.

Para utilizar el código, hay que crear el proyecto correspondiente, pues hasta el momento no es más que un archivo VHDL. Utilizando el asistente abrimos la ventana de la Figura 37, denominamos sumrest8 al proyecto, y compilamos el archivo VHDL.

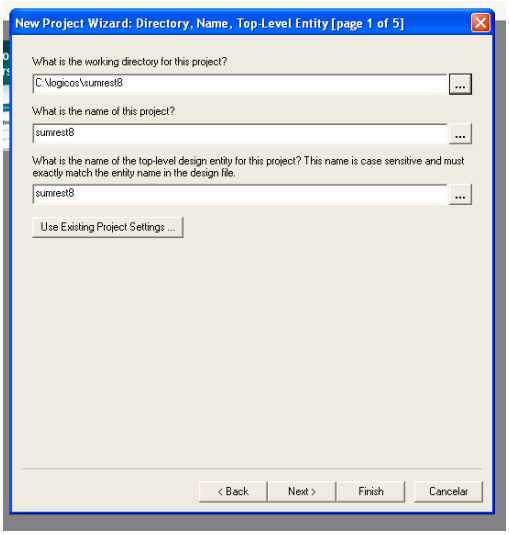


Figura 37. Creación del directorio de sumrest8.

CAPÍTULO 5. MÓDULOS PREDISEÑADOS

Quartus II permite utilizar módulos prediseñados (plantillas) de uso común además de los incluidos en la librería LPM. Si abre un archivo VHDL nuevo (New), Pulse **Edit→Insert Template**, y se abre la ventana de la Figura 38. Si expande VHDL, se observan las opciones **Full Designs, Logic, etc.**

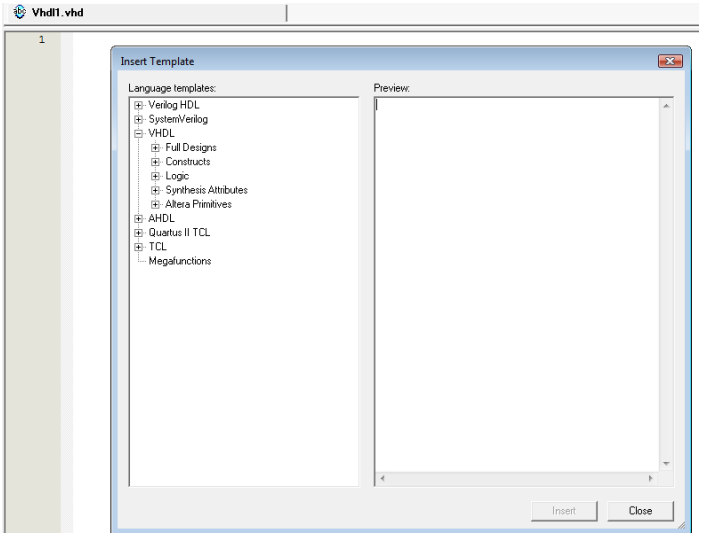


Figura 38. Ventana de módulos de uso común (plantillas).

En la Figura 39 se muestra la ventana obtenida al seleccionar **Full Designs→Arithmetic→Signed Adder /Substracter.** Su código VHDL aparece en la parte derecha, y puede insertarse en el archivo.

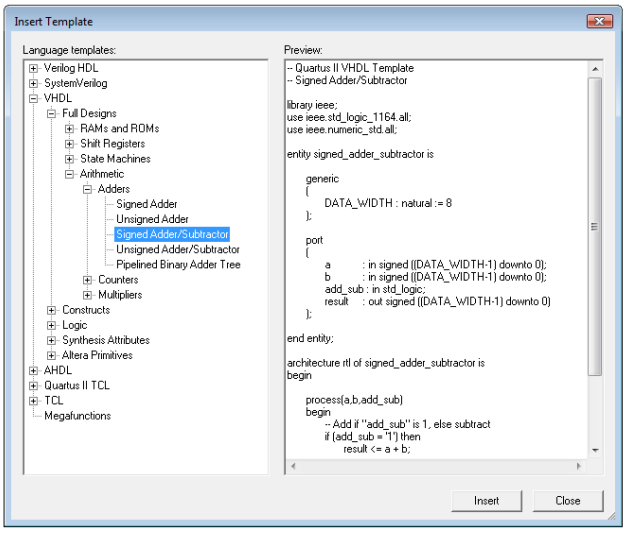


Figura 39. Selección de un módulo sumador/restador.

Diseño por captura esquemática. En vez de utilizar el lenguaje VHDL, mostramos ahora el diseño de un sistema utilizando captura esquemática. Creamos primeramente el directorio de un proyecto sum22 (sumador de 2+2 bits, que se diseñará con 2 bloques del proyecto sum1)), y generamos un archivo nuevo abriendo la ventana de la Figura 7 y pulsando **Block diagram/Schematic File**. Aparece así la ventana de la Figura 40 (hoja de diseño esquemático).

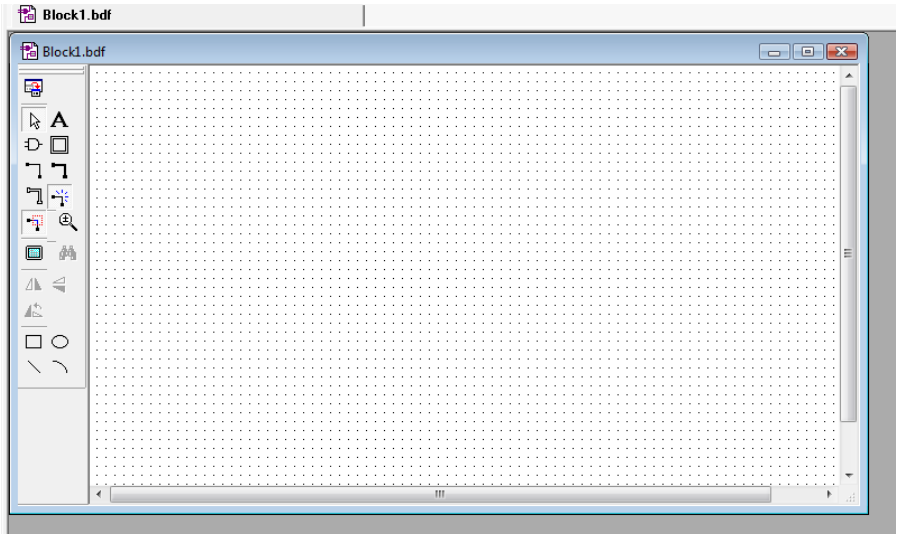


Figura 40. Ventana de diseño esquemático.

Pulsamos ahora un doble click en la ventana para acceder a la biblioteca de símbolos, y se abre la ventana de la Figura 41. Las librerías incluyen megafunciones, primitivos (como conectores, compuertas, etc.) y "otros". Otra manera de acceso consiste en pulsar **Edit→Insert Symbol.**

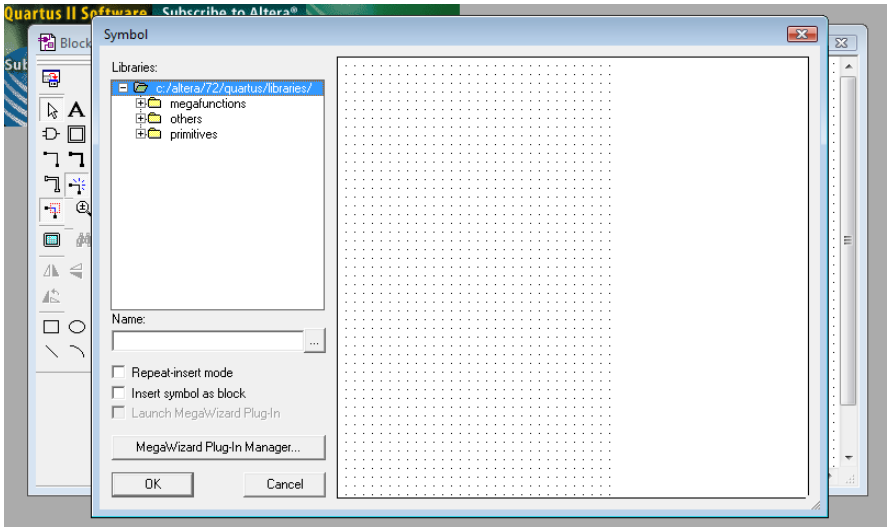


Figura 41.

Para acceder al símbolo del bloque de sum1, accesamos al proyecto sum1 en el cuadro de exploración . Al abrirlo, aparece el símbolo en la ventana de la Figura 42. Al pulsar OK, el símbolo se transfiere a la hoja de diseño esquemático (Figura 43).

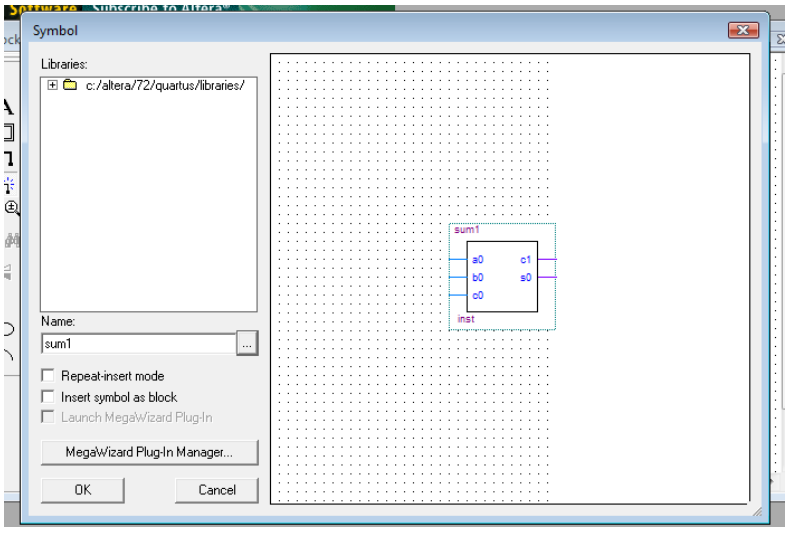


Figura 42. Abriendo el símbolo de sum1.

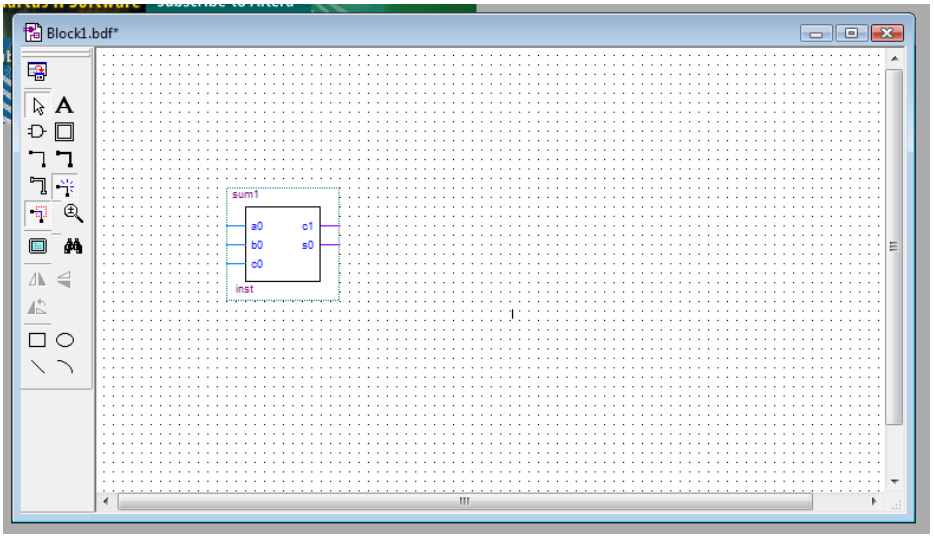


Figura 43. Símbolo de sum1 en la pantalla de diseño esquemático.

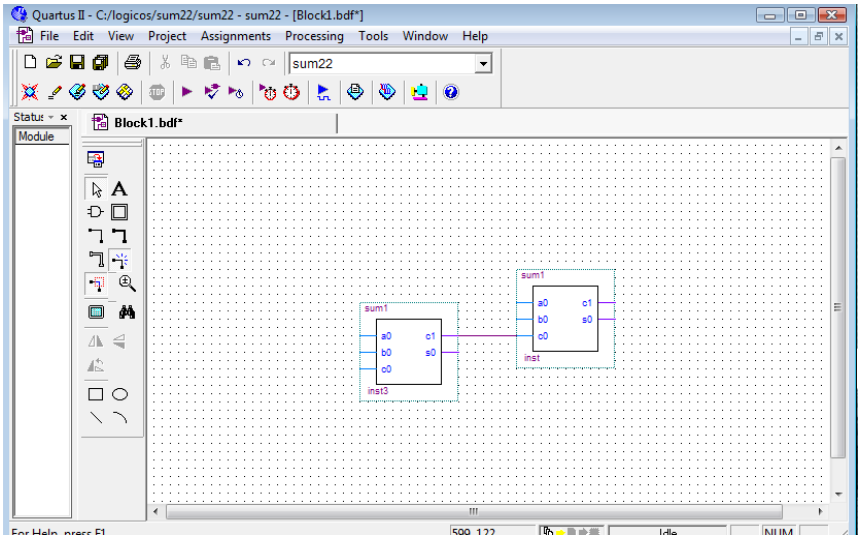


Figura 44. Conexión de los 2 bloques sum1 por medio de los acarreos.

El diseño del sumador de 2 bits requiere 2 instancias del sumador sum1; el acarreo de entrada del sumador que implementa el bit más significativo proviene del acarreo de salida del sumador que implementa el bit menos significativo. El esquema se muestra en

la Figura 44. Para obtener un segundo bloque de sum1, puede repetirse el proceso anterior, o bien señalar con el ratón el bloque original arrastrándolo mientras se pulsa **control.** Otra opción es utilizar el símbolo de selección y dibujo (la flecha blanca del editor de bloques situado en la parte superior izquierda del editor de bloques), seleccionar el bloque y copiarlo (**Edit→copy, Edit→Paste).** Con la misma flecha blanca seleccionada, use el ratón para unir las terminales s1 de un bloque con la s0 del segundo bloque. Tenemos así el esquema de la Figura 44.

Falta sólo importar los símbolos de las terminales. Haga doble click en la pantalla para abrir las librerías; expanda **primitives** , **pines** y seleccione **input.** Se obtiene así la pantalla de la Figura 45.

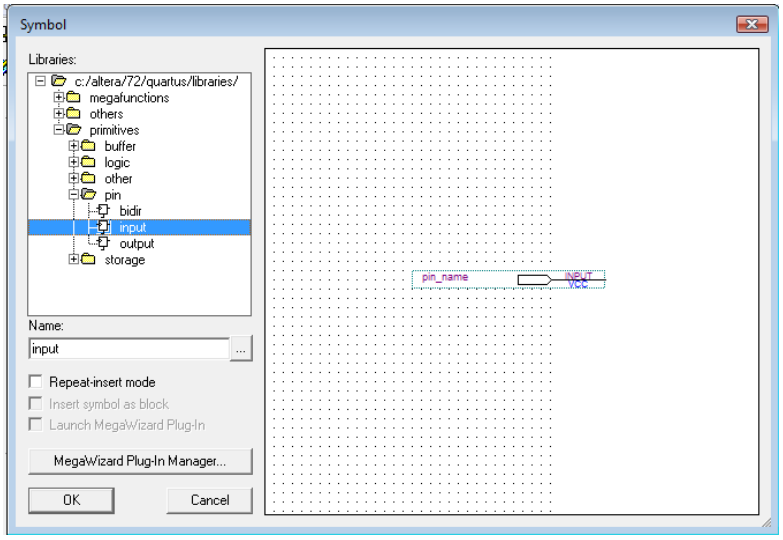


Figura 45. Símbolo de un conector de entrada.

Pulse OK, y el símbolo del conector aparece en la pantalla de diseño. Repita el procedimiento para un pin de salida (output), y transfiéralo también a la pantalla de diseño. Copie y pegue por cualquier método los símbolos de los conectores hasta tener 5 entradas y 3 salidas, conectadas a los bloques sum1 como en la Figura 46.

Asignación de nombres a los símbolos de entrada y salida. Falta sólo asignar nombres a cada símbolo de entrada o salida. Apunte con el ratón a cada palabra pin_name y haga doble click. Aparece la ventana de asignación de nombres de la Figura 47.

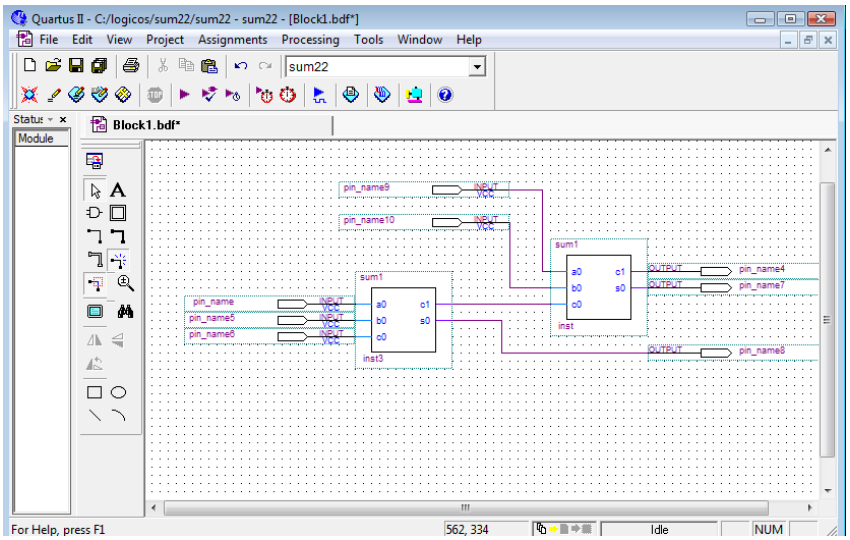


Figura 46. Esquema para sum22 sin asignación de nombres a entradas y salidas.

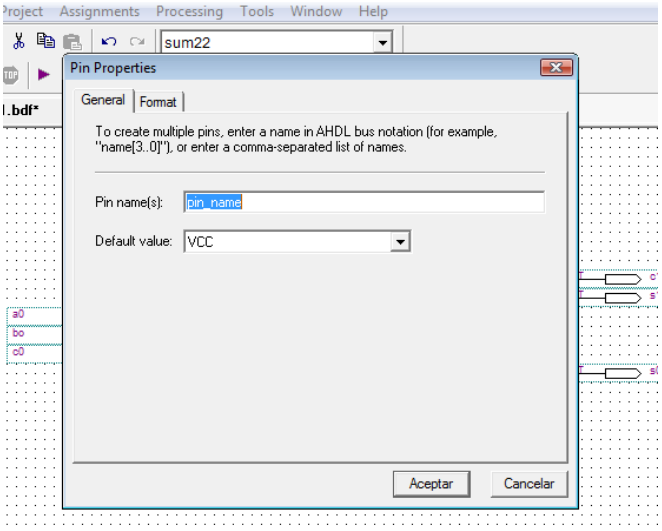


Figura 47. Asignación de nombres.

Repita el procedimiento para cada entrada y salida. Obtenemos así el esquema final de la Figura 48, con entradas a(1:0), b(1:0), c0 y salidas s(1:0) y c1. Guarde el archivo tipo bloque como sum22.

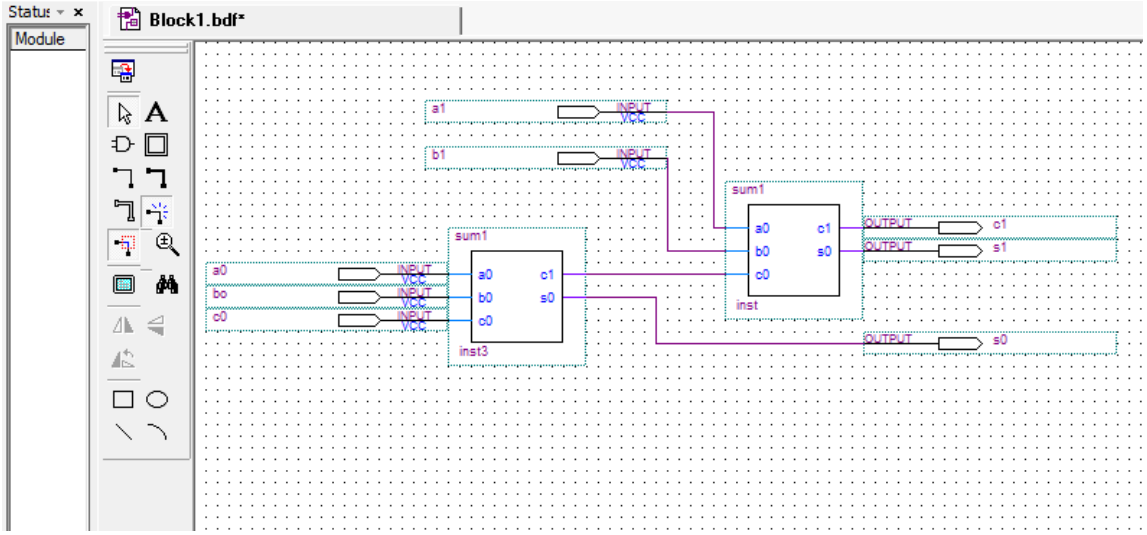


Figura 48. Esquema final de sum22.

Resta tan sólo compilar el proyecto, no sin antes añadirle el archivo VHDL de sum1 como ya lo hemos mostrado. En la Figura 49 se muestra el reporte correspondiente.

Flow Status	Successful - Mon Jan 10 09:29:46 2011
Quartus II Version	7.2 Build 151 09/26/2007 SJ Web Edition
Revision Name	sum22
Top-level Entity Name	sum22
Family	Cyclone II
Met timing requirements	N/A
Total logic elements	4
Total combinational functions	4
Dedicated logic registers	0
Total registers	0
Total pins	8

Figura 49. Reporte de compilación.

En la Figura 50 se muestra, por último, una hoja de simulación. El lector podrá comprobar que los resultados de la suma s y acarreo final c1 son correctos.

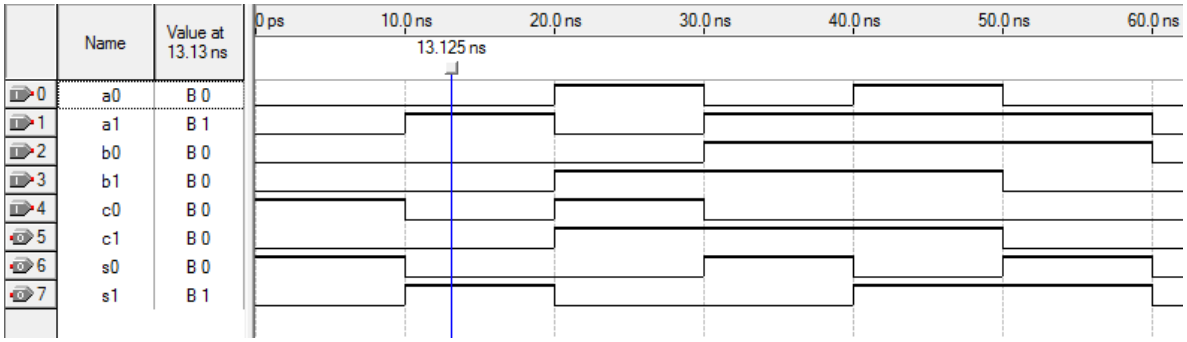


Figura 50. Hoja de formas de onda simuladas.

REFERENCIAS

"PDF Tutorial for VHDL users", en HELP del programa Quartus II.

Tocci, Widmer, Moss. "Sistemas Digitales, Principios y Aplicaciones, décima edición, Pearson