Instituto Politécnico Nacional

Centro de Investigación en Computación

Lenguajes de descripción de hardware

Tarea 5 - Descripción estructural y IPCATALOG

Profesor:

M. en C. Osvaldo Espinosa Sosa

Por:

Ing. Ricardo Aldair Tirado Torres

Ciudad de México, 9 de mayo de 2024

Tabla de contenido

1. Objetivos **2**

2. Descripción estructural de un multiplexor **3**

3. Descripción estructural de un multiplexor con *IP Catalog* **5**

4. Conclusiones **11**

5. Anexos **12**

5.1. Descripciones del hardware 12

5.2. Bancos de pruebas (*Test Benches*) 12

1. Objetivos

- Implementar la descripción en forma estructural para generar un multiplexor sencillo e implementar la instanciación de un módulo creado por el usuario.
- Aprender sobre el uso de la herramienta *IP Catalog* y como se utiliza para instanciar módulos ya proporcionados por el propietario de la herramienta de Quartus.

2. Descripción estructural de un multiplexor

Actividad 1

Capturar el código para describir en forma estructural el multiplexor en el lenguaje de su elección. Instanciar el módulo o componente que se encuentre en el mismo archivo. Simular el multiplexor.

La visualización RTL del multiplexor, usando descripción estructural en Verilog, se muestra en la Figura 1. La implementación se hace con la instanciación de un multiplexor, denominado "MyMux", visualizado en el interior del módulo (Ver Figura 2). Las simulaciones se visualizan en la Figura 3, en donde se muestra que el multiplexor descrito opera de manera correcta.

En los Anexos se localiza la descripción en Verilog de este multiplexor. En el código se tienen dos módulos, siendo el primero, el de la jerarquía más alta y en donde se realiza la declaración de las entradas y la salida, para luego instanciar al módulo llamado "MyMux" con la etiqueta "u0". Cabe señalar que los argumentos de la instancia se deben colocar en el orden correcto. El segundo módulo unicamente es la descripción de un multiplexor sencillo, utilizando el operador condicional "?".

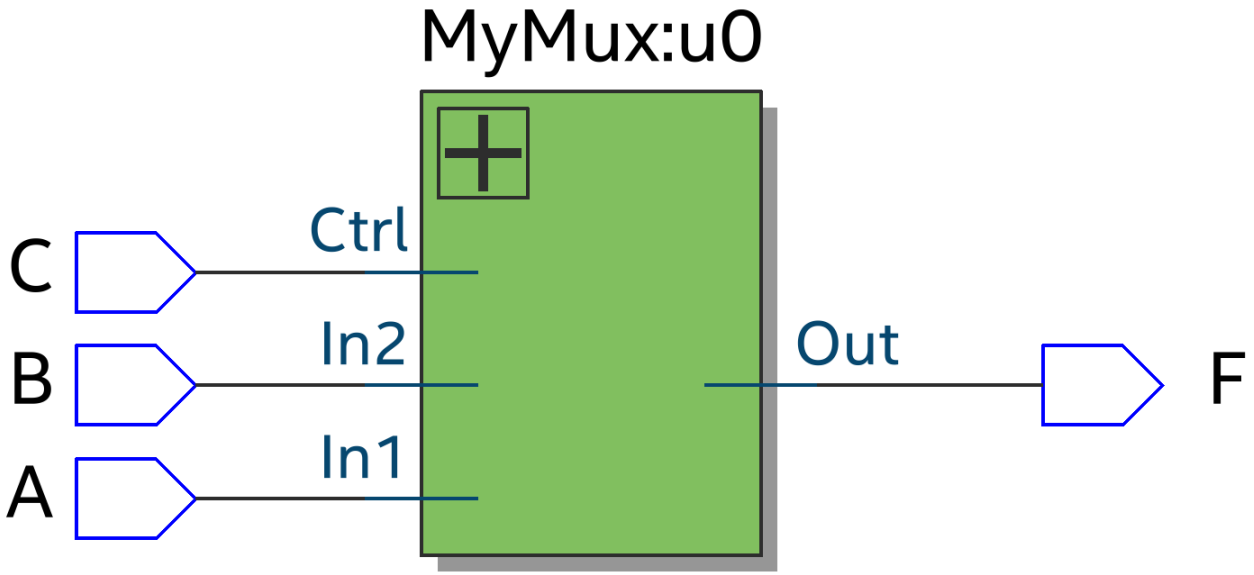

Figura 1: Diagrama RTL del multiplexor, descrito en forma estructural en Verilog.

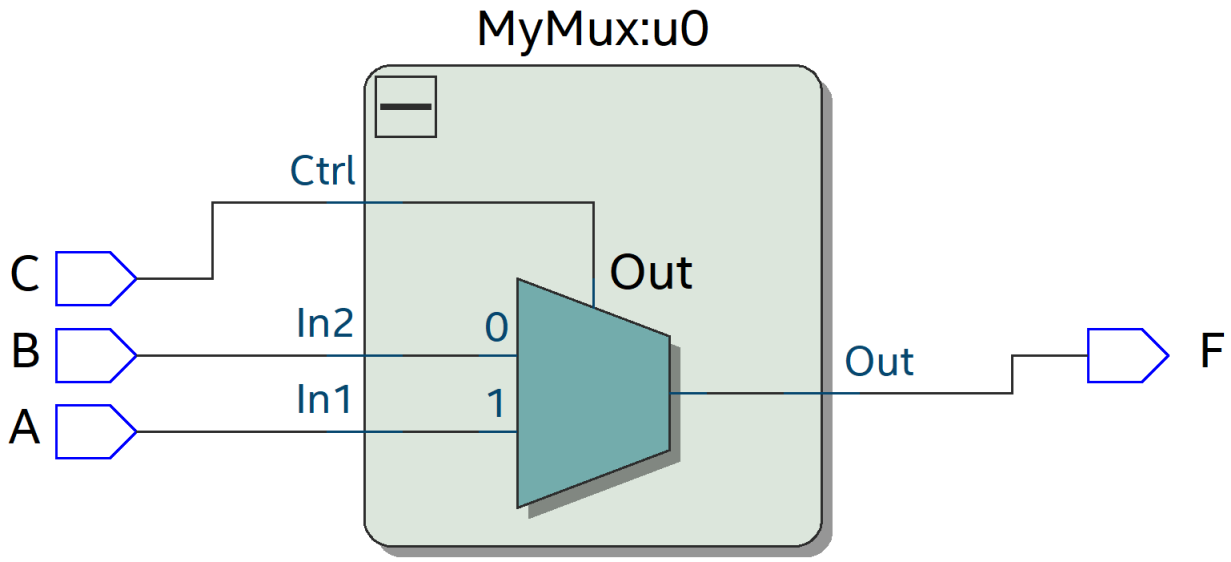

Figura 2: Diagrama RTL del multiplexor, descrito en forma estructural en Verilog (visualización de la instancia interna).

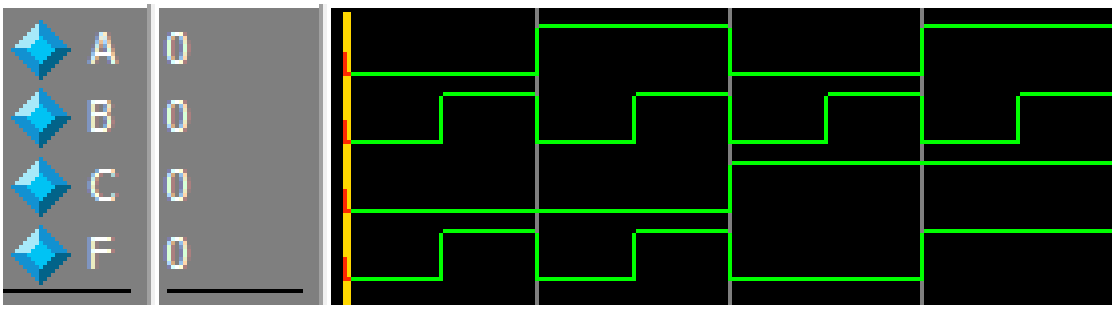

Figura 3: Simulación del multiplexor, descrito en forma estructural en Verilog, con el visor de formas de onda de ModelSim.

3. Descripción estructural de un multiplexor con *IP Catalog*

Actividad 2

Describir el multiplexor en forma estructural usando una función creada por el *IP Catalog* siguiendo los pasos descritos en el documento o presentación. Simular el multiplexor.

Para generar la instancia con *IP Catalog*, que se va a utilizar en el módulo principal, se realizaron los siguientes pasos:

- En el apartado de *Library*, se seleccionó *Basic Functions* $->$ *Miscellaneous* $->$ *LPM_MUX* (Ver Figura 4).
- Se coloco el nombre de la instancia, se seleccionó como archivo tipo Verilog y se dio clic en el botón *OK* (Ver Figura 5).
- Se indicó que el módulo debe tener 2 entradas, una salida y que no se iba a dividir en etapas, conocidas como *pipelines* (Ver Figura 6).
- Se dio clic en el botón *Next* (Ver Figura 7).
- Se seleccionaron las opciones para la creación de los archivos *MyMux.cmp*, *MyMux_inst.v* y *MyMux_bb.v* (Ver Figura 8).
- Finalmente, se dio clic en el botón *Yes*, para instanciar el archivo en el proyecto (Ver Figura 9).

Como se observa en la Figura 10 se generó la instancia del archivo en la carpeta del proyecto.

La visualización RTL del multiplexor, usando descripción estructural en Verilog, y con ayuda de *IP Catalog*, se muestra en la Figura 11. La implementación con la herramienta, anteriormente mencionada, hace la instanciación de varios módulos, vistos en la Figura 12, Figura 13 y Figura 14. Las simulaciones se visualizan en la Figura 15, en donde se muestra que el multiplexor descrito opera de manera correcta.

En los Anexos se localiza la descripción en Verilog de este multiplexor. Debido a que se utilizó la herramienta *IP Catalog*, el código solo tiene la descripción del módulo de más alta jerarquía, en donde se realiza la declaración de las entradas y la salida, para luego instanciar al módulo llamado "MyMux" con la etiqueta "u0". Cabe señalar que los argumentos de la instancia se deben colocar en el orden correcto.

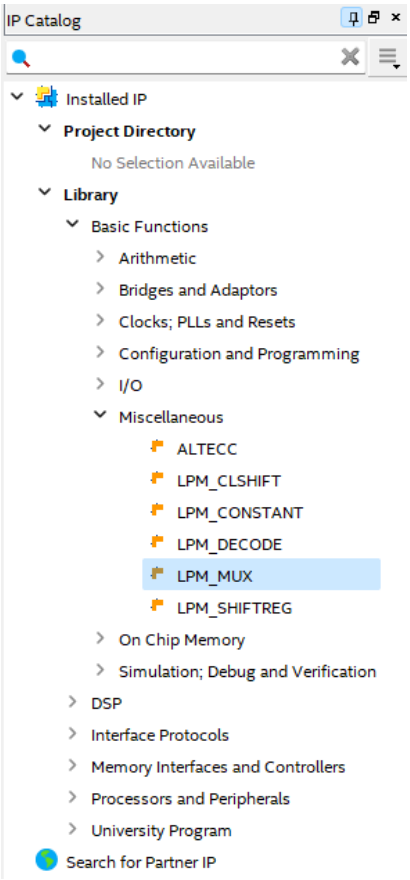

Figura 4: Instanciación del multiplexor con *IP Catalog* (Paso 1).

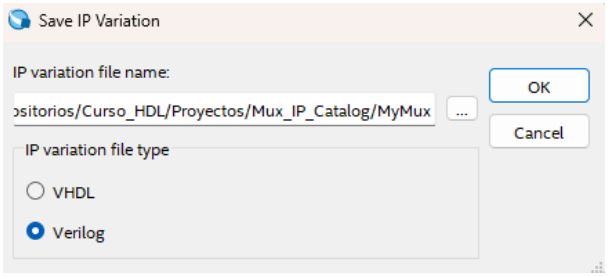

Figura 5: Instanciación del multiplexor con *IP Catalog* (Paso 2).

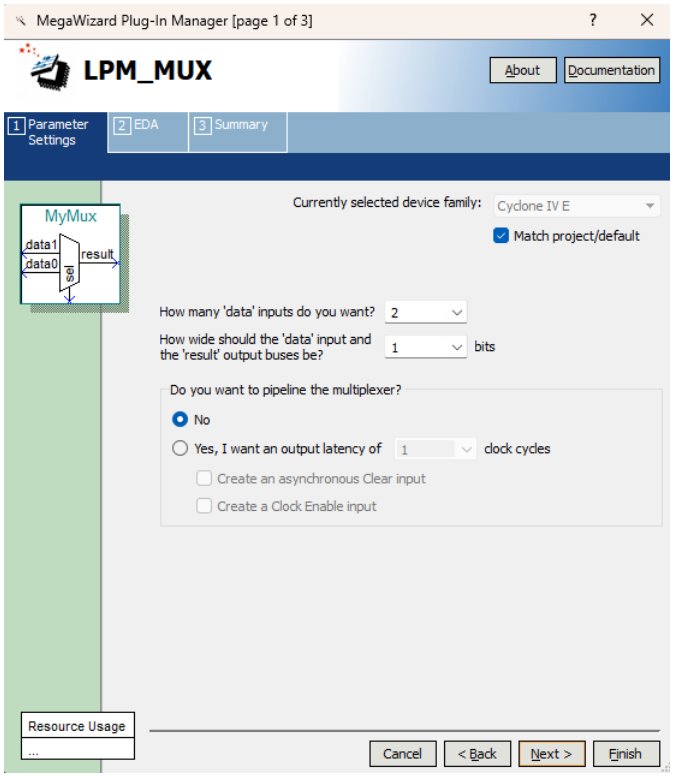

Figura 6: Instanciación del multiplexor con *IP Catalog* (Paso 3).

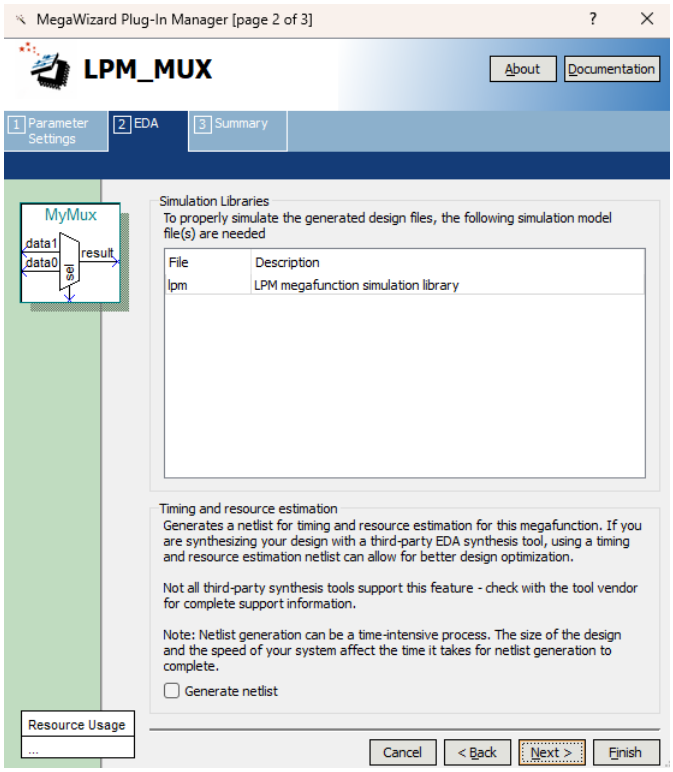

Figura 7: Instanciación del multiplexor con *IP Catalog* (Paso 4).

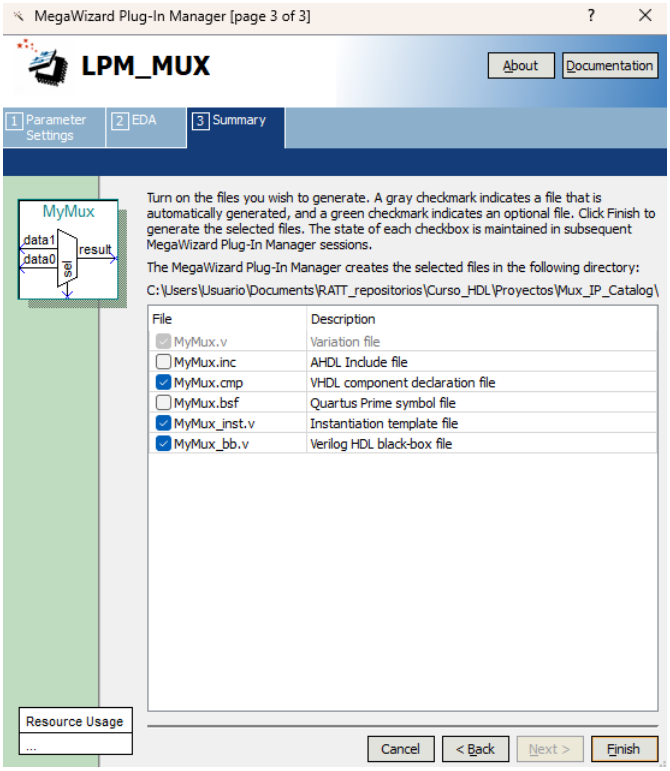

Figura 8: Instanciación del multiplexor con *IP Catalog* (Paso 5).

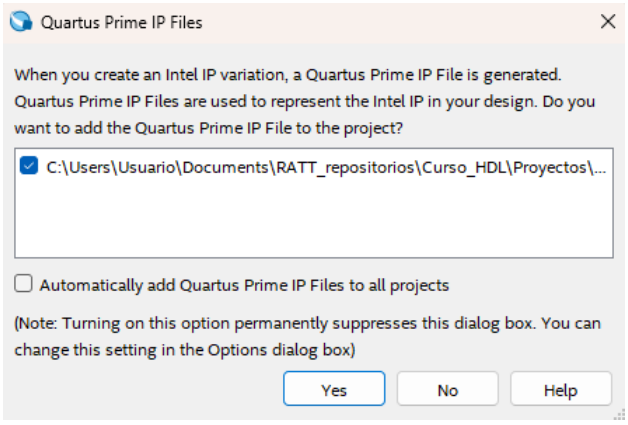

Figura 9: Instanciación del multiplexor con *IP Catalog* (Paso 6).

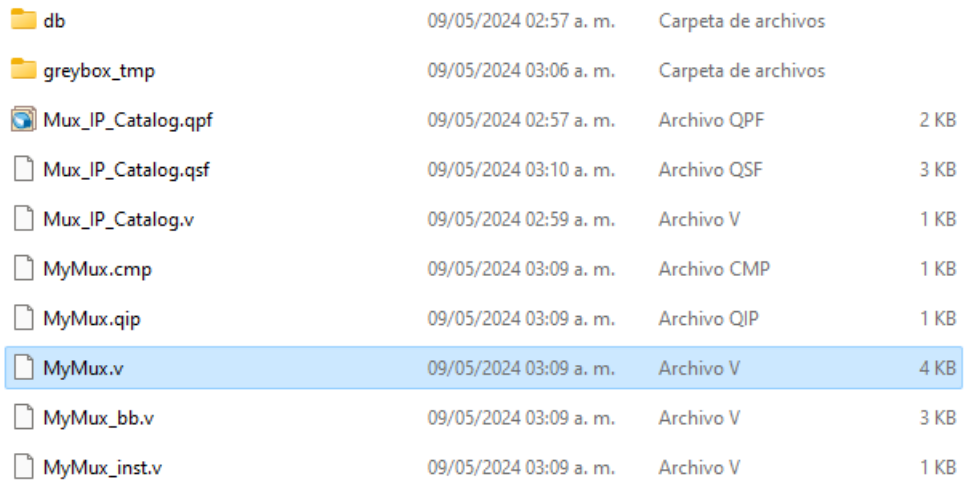

Figura 10: Instanciación del multiplexor con *IP Catalog* (Paso 7).

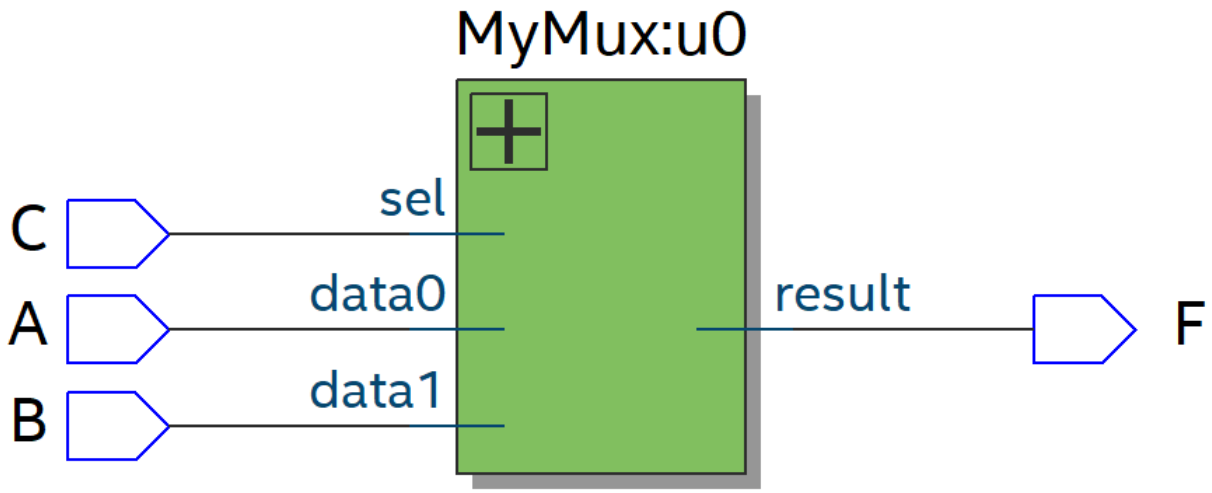

Figura 11: Diagrama RTL del multiplexor, descrito en forma estructural con *IP Catalog*.

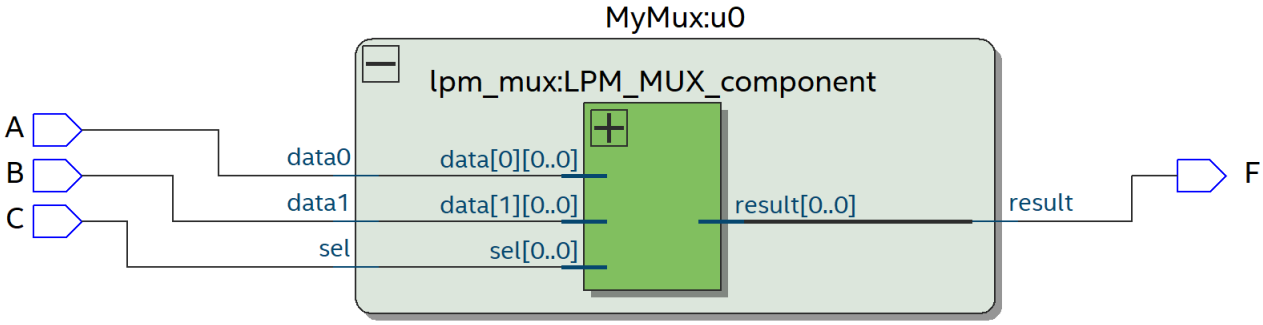

Figura 12: Diagrama RTL del multiplexor, descrito en forma estructural con *IP Catalog* (visualización de la primera instancia interna).

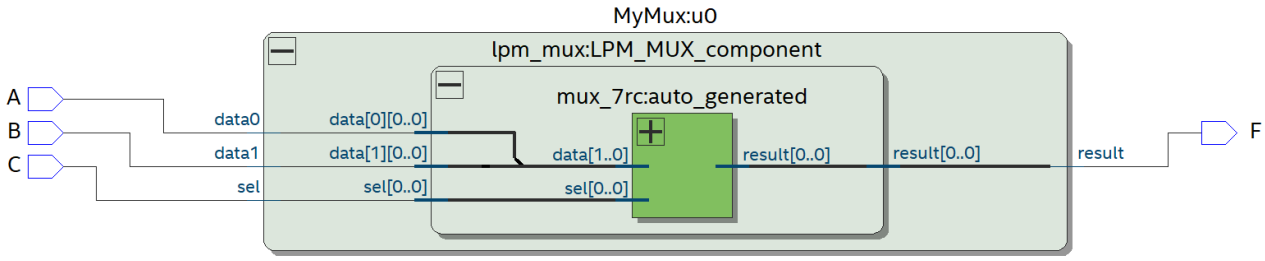

Figura 13: Diagrama RTL del multiplexor, descrito en forma estructural con *IP Catalog* (visualización de la segunda instancia interna).

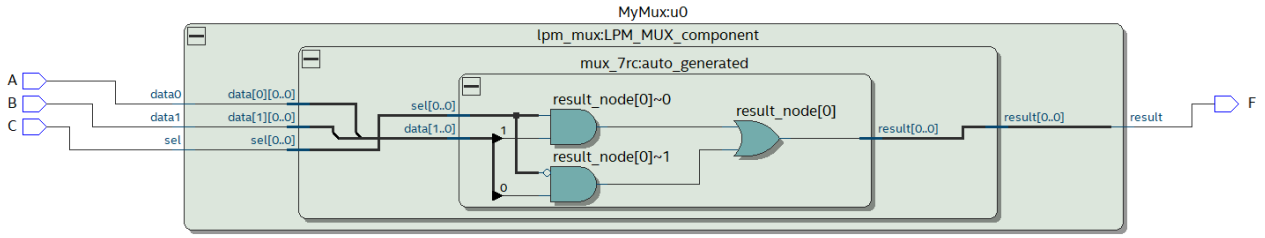

Figura 14: Diagrama RTL del multiplexor, descrito en forma estructural con *IP Catalog* (visualización de la tercera instancia interna).

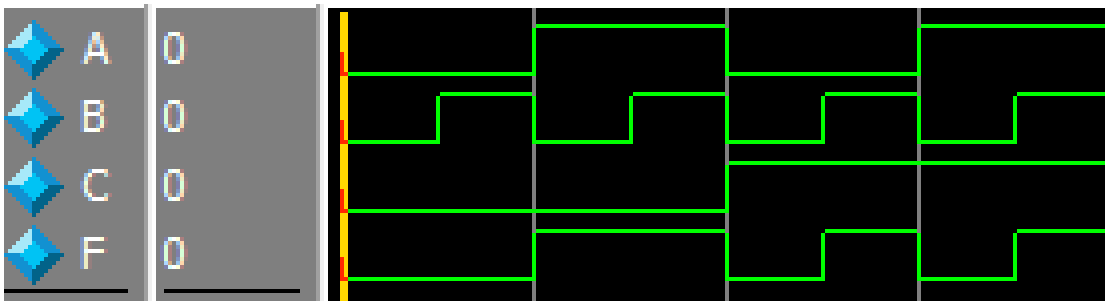

Figura 15: Simulación del multiplexor, descrito en forma estructural con *IP Catalog* en Verilog, con el visor de formas de onda de ModelSim.

4. Conclusiones

En conclusión, se implementó el circuito del multiplexor, en Verilog, de manera correcta.

Se comprendió como se utiliza la descripción estructural para instanciar módulos ya creados por el usuario, con el fin de generar hardware más complejo.

Se comprendió la función de la herramienta *IP Catalog* y se instanció un módulo de multiplexor simple como parte de otro módulo de mayor jerarquía.

En ambos casos se implementó un multiplexor sencillo con descripción estructural y se observó con el visor RTL a los circuitos instanciados dentro del módulo principal, y por medio de las simulaciones de forma de onda en ModelSim, se visualizó la correcta operación del dispositivo.

En los Anexos se pueden encontrar los códigos implementados junto con sus respectivos bancos de pruebas.

5. Anexos

5.1. Descripciones del hardware

```
module Mux_Structural_Description(
 input  A, B, C,
 output  F);

MyMux u0(A, B, C, F);

endmodule

//Modulo a instanciar
module MyMux(
 input  In1, In2, Ctrl,
 output  Out);

assign Out = Ctrl ? In1 : In2;

endmodule
```

Programa 1: Descripción en Verilog del multiplexor en forma estructural.

```
module Mux_IP_Catalog(
 input  A, B, C,
 output  F);

MyMux u0(A, B, C, F);

endmodule
```

Programa 2: Descripción en Verilog del multiplexor en forma estructural, instanciando un archivo creado con *IP Catalog*.

5.2. Bancos de pruebas (*Test Benches*)

```
`timescale 1 ns/ 1 ps
module Mux_Structural_Description_vlg_tst();
 reg  A;
 reg  B;
 reg  C;
 wire  F;

 Mux_Structural_Description i1 (
```

```
  .A(A),
  .B(B),
  .C(C),
  .F(F)
 );

 initial
 begin
  A = 0; B = 0; C = 0;
  $display("Running testbench at CIC");
 end

 always
 begin
  #10; A = 0; B = 1; C = 0;
  #10; A = 1; B = 0; C = 0;
  #10; A = 1; B = 1; C = 0;
  #10; A = 0; B = 0; C = 1;
  #10; A = 0; B = 1; C = 1;
  #10; A = 1; B = 0; C = 1;
  #10; A = 1; B = 1; C = 1;
 end

endmodule
```

Programa 3: Banco de prueba para el Programa 1.

```
`timescale 1 ns/ 1 ps
module Mux_IP_Catalog_vlg_tst();
 reg   A;
 reg   B;
 reg   C;
 wire   F;

 Mux_IP_Catalog i1 (
  .A(A),
  .B(B),
  .C(C),
  .F(F)
 );

 initial
 begin
  A = 0; B = 0; C = 0;
  $display("Running testbench at CIC");
 end
```

```
20
21  always
22  begin
23   #10; A = 0; B = 1; C = 0;
24   #10; A = 1; B = 0; C = 0;
25   #10; A = 1; B = 1; C = 0;
26   #10; A = 0; B = 0; C = 1;
27   #10; A = 0; B = 1; C = 1;
28   #10; A = 1; B = 0; C = 1;
29   #10; A = 1; B = 1; C = 1;
30  end
31
32 endmodule
```

Programa 4: Banco de prueba para el Programa 2.