Diseño de un circuito con multiplexores y su implementación en Verilog usando la FPGA Spartan 3A

En resumen, en este documento se indica paso a paso la descripción de hardware en la FPGA de un multiplexor 2 a 1, un multiplexor 8 a 1 y posteriormente el uso de estos para diseñar un circuito y su descripción a partir de la siguiente tabla de verdad:

SW [2:0]	Y
000	8'b10101010
001	8'b11001100
010	8'b10011001
011	8´b10100001
	X
	X
	X

Para empezar cabe recordar lo que es un multiplexor 2 a 1 de manera lógica:



Y cómo se representa de forma simplificada:



sel	out
0	А
1	В

Viendo así que un multiplexor de este tipo se comporta como un interruptor, siendo la salida igual a A si el selector está en 0, y la salida igual a B si el selector está en 1.

Para implementar este circuito combinacional es necesario conocer su expresión booleana (La cual se puede obtener por mapas de Karnaugh y agrupación por minterms),

$$Y = \bar{S}D_o + SD_1$$

Donde se ha llamado S al selector, Y a la salida y $\mathsf{D}_{o}\, y\, \mathsf{D}_{1}$ a cada una de las entradas.

Teniendo esto claro se puede empezar describir este circuito en Verilog.

Los pasos a seguir serán:

En primer lugar, crear un nuevo proyecto o por conveniencia usar la plantilla trabajada en la práctica anterior, para este caso se hizo sobre un proyecto en blanco (Si es la primera vez que se usa el software puede remitirse a <u>https://drive.google.com/file/d/0B_sA6IW3ej35X0c3bGo2SGQ4Q0k/view</u> donde se explica cómo introducir algunas especificaciones para esta tarjeta en particular)



Agregar el primer módulo que será el que contenga la descripción del multiplexor de 2 entradas a 1 salida



New Source Wizard	X
Select Source Type Select source type, file name and its location. IP (CORE Generator & Architecture Wizard) Schematic User Document Verilog Module Verilog Test Fixture VHDL Module VHDL Library VHDL Dackage VHDL Test Bench Embedded Processor	File name: mux2_1 Location: C:\FPGA\Multiplexores
More Info	Next Cancel

Se tendrá algo como esto:

- I	E Project I	Navigator	(P.20131	013) - C:\	FPGA\Mu	tiplexor	es\Multiple	cores.xise	[mux2_1.v]	Contraction of the owner owner of the owner
	ile Edit	View	Project	Source	Process	Tools	Window	Layout	Help	
	🖻 🗟 1	1 🕹	1 X 🕻	6×	IS CI	»	PPP) B /	2 🔊	h ⊟ ⊡ ⊡ 🥬 🖗 🕨 🗶 📌 💡
Desig	n			++	08×	E	1 'ti	mescal	1ns / 1	ps
T [#]	View: 💿	🔯 Implerr	entation	🔘 🛃 Sir	nulation	Εđ	2 ///	//////	1111111	///////////////////////////////////////
a	Hierarchy					_	3 //	Company	73	
Ch.	- 🖻 M	lultiplexor	es			=	4 //	Engine	er:	
	🖨 🗍 xc	3s700a-4f	g484			<u>=</u>	6 //	Create	Date:	16:56:55 03/01/2015
00	L V	mux	2_1 (muxi	2_1.v)		E	7 //	Design	Name :	
a.						2	8 //	Module	Name :	mux2_1
<u> </u>						-	9 //	Project	: Name:	
					:	18	10 //	Target	Devices:	
						24	12 //	Descrip	tion:	
-					1	74	13 //	200011		
						74	14 //	Depende	incies:	
							15 //			
	8) No Pr	ocassas Di	Inning			6	16 //	Revisi	on:	
-	Ca NORI	occases ra	anning			©	17 //	Revisio	on 0.01 -	File Created
ЩЩ.	Processes	: mux2_1				_	18 //	Additio	onal Comme	ents:
则	····· <u>></u>	Design S	ummary	/Reports			20 ///	11111		
ETVH.	🖻 – 🎾	Design L	Itilities		-		21 moo	ule mu	(2 1 (
14	🕀 🦉	User Cor	istraints				22);		
	H C2	Synthesi	ze - XST				23			
		Generate	Program	' Imina File			24			
	🗄 🐞	Configu	re Target	Device	-		25 end	module		
	en en	Analyze	Design U	sing Chip	Scope		20			
						4				

Donde se hará la descripción de la siguiente manera:

- S, D0 y D1 son las entradas
- Y es la salida
- Y tiene por consiguiente asignada la expresión booleana del circuito

```
module mux2_1(
input S,
input D0,
input D1,
output Y
);
```

assign Y=(~S | D0)&(S | D1);

endmodule

En este punto se puede comprobar la sintaxis del programa, si hay algún error es necesario corregirlo antes de continuar.

Si al verificar el programa ha indicado que la sintaxis es correcta, ya ha hecho su primera descripción de un multiplexor ©

Para describir el multiplexor de 8 a 1, se instanciará el multiplexor descrito anteriormente de la siguiente manera:



Con 7 multiplexores de 2 a 1 para un total de tres selectores que serán las tres entradas variables como se indican en la tabla mostrada previamente y una salida de la que hablaremos más adelante.

Ahora bien, su descripción en Verilog será:

- Un bus de ocho entradas llamado D
- Un bus de tres selectores llamados S0, S1, y S2 respectivamente
- Una salida llamada LED
- Siete cables para las respectivas conexiones

```
module mux8_1(
input [7:0]D,
input [2:0]S,
output LED
    );
wire [6:1]C;

    mux2_1 m1 (
    .S(S[2]),
    .D0(C[1]),
```

```
.D1(C[2]),
.Y(LED)
);
mux2_1 m2 (
.S(S[1]),
.D0(C[5]),
.D1(C[6]),
.Y(C[1])
);
mux2_1 m3 (
.S(S[1]),
.D0(C[3]),
.D1(C[4]),
.Y(C[2])
);
mux2_1 m4 (
.S(S[0]),
.D0(D[0]),
.D1(D[1]),
.Y(C[5])
);
mux2_1 m5 (
.S(S[0]),
.D0(D[2]),
.D1(D[3]),
.Y(C[6])
);
mux2_1 m6 (
.S(S[0]),
.D0(D[4]),
.D1(D[5]),
.Y(C[3])
);
mux2_1 m7 (
.S(S[0]),
.D0(D[6]),
.D1(D[7]),
.Y(C[4])
);
```

endmodule

Se puede ver claramente cada una de las instanciaciones denominadas según la imagen anterior y las asignaciones de la misma manera. Por ejemplo, para el primer multiplexor de 2 a 1, llamado m1 el código escrito se indicó así:

mux2_1 m1 (//Nombre asignado a la instanciación
.S(S[2]),	//Selector único en el multiplexor de dos a 1 que
	corresponde al segundo en el multiplexor de 8 a 1
.D0(C[1]),	//Primera entrada que va conectada al cable número 1
.D1(C[2]),	//Segunda entrada que va conectada al cable número 2
.Y(LED)	//Salida conectada a la salida completa del mux de 8 a 1
);	·

Comprobar la sintaxis nuevamente, de nuevo esto es indispensable para continuar con la descripción.

Si se desea, y con una tabla de verdad para un multiplexor de 8 a 1 a la mano se puede realizar la simulación de éste. (Recomendable)

En el caso de realizarla, con la vista de simulación seleccionada y el módulo a simular, seleccionar Simulate Behavioral Model



Se abre la ventana de simulación

ISim (P.20131013) - [Default.wcfg]	No. of Concession, name	And in case of the local division of the loc	a 1 <u>1</u>				Sec. 1					. 0 ×
File Edit View Simulation Win	dow Layout Help											- 8 ×
🗋 🏓 🖬 🖕 🐰 🗅 🗈 🗙 🕲	A A .	10 58		P K? 1	Ð 🔎 👰 🏓	2 2 2	in 🐴 🖬 🕨	1.00us 💌 🤇	🗉 🗔 Re-lau	nch		
Instances and Processes ↔ □ ♂ ×	Objects	++□ & ×							1,000,000 ps			· ^
	Simulation Objects for n	nux8_1										
Instance and Process Name D	THEFT	15 🐼	8 N		Value	999,997 ps	999,998 ps	999,999 ps	1,000,000 ps	1,000,001 ps	1,000,002 ps	1,000,003 p
b 🗐 mux8 1 m	Object Name	Value	1		7		LLLLLLL		1			
p 📒 gibi gi	▶ N D[7:0]	2222222	0		2							
	⊳ 📸 S[2:0]	222			-							
	Lig Y	x	0	10	-							
	C[6:1]	XXXXXX	1	L <u>in</u> [4]	z							
			d r	L [3]	z							
			-	1 [2]	z							
			ŭ	լե [1]	z							
			10	1 [0]	z							
			1	S[2:0]	222		222					
			10.9	10. v	x							
			123	CI6-11	*****		200000					
			11	- Clourl	100000				1			
			1111									
									<u>_</u>			
						X1: 1,000,000 ps						
•				E	- F	4						
🐣 Instanc 🗎 Memory 📔 Source	۰ m	F	202		D	efault.wcfg						
Console			1									↔ 🗆 🗗 ×
WARNING: A WEBPACK license was found.												
WARNING: Please use Xilinx License Configurat	ion Manager to check out	a full ISim license.										
WARNING: ISim will run in Lite mode. Please ref	er to the ISim documenta	tion for more information	n on the diff	erences betwe	en the Lite and t	he Full version.						
Time resolution is 1 ps												E
Simulator is doing circuit initialization process.												
Finished circuit initialization process.												
ISIM>	100	100										•

Forzar constante para cada una las entradas D_n y para los selectores S

dow Layout Help								- 1
🗠 🖂 🕅 🕹 🕇 🚳 🛤 🗠	🖻 🏓 k? 🏓 🏓 🎗	🖻 🔎 🗟 🗠 💇	† 🗠 🐴 🖬 🕨	🔉 1.00us 💌 🐓	🗔 Re-laund	h		
Objects ↔ □ ♂ × Simulation Objects for mux8_1 □	Name Value	1999,997 ps	999,998 ps	1999,999 ps	1,000,000 ps	1,000,001 ps	1,000,002 ps	1,000,003
Object Name Value Image: Difference of the state of the sta	1 [7] 2 1 [6] 2 1 [5] 2 1 [6] 2 1 [7] 2 1 [9] 2 1 [9] 2 1 [9] 2 1 [9] 2 1 [9] 2 1 [9] 2 1 [9] 2 1 [9] 2 1 [9] 2 1 [9] 2 1 [9] 2 1 [9] 2 1 [9] 2 1 [9] 2 1 [9] [2] 1 [9] [2] 1 [9] [2] 1 [9] [2] 1 [9] [2] 1 [9] [2] 2 [9] [9] 2 [9] [9] 2 [9] [9]	Cut Copy Paste Paste Find Radix Signal Color Divider Color Go To Source Code Show Drivers Force Clock Remove Force	Ctrl+X Ctrl+C Ctrl+C Ctrl+V Del Ctrl+F					
· · · · · ·		Default.wcfg	ظل ا	×				

A continuación algunos ejemplos, para comprobar el correcto funcionamiento del circuito.

Object Name	Value
> 📷 D[7:0]	00000000
S[2:0]	000
l <mark>la</mark> v	0
> 📲 C[6:1]	000000

0	bject Name	Value	
\triangleright	📷 D[7:0]	10101010	
⊳	📷 S[2:0]	000	
	🖫 LED	1	
⊳	🟹 C[6:1]	111111	
	-		

0	bject Name	Value	
\triangleright	📷 D[7:0]	10101010	
⊳	📸 S[2:0]	111	
	🔏 LED	0	
\triangleright	🏹 C[6:1]	000000	

Object Name	Value
> 📷 D[7:0]	10101010
> 📷 S[2:0]	101
LED LED	0
> 📸 C[6:1]	000000
_	

Si ya se ha comprobado que todo funciona correctamente, empezar a realizar la descripción del módulo para el circuito final. Viendo nuevamente la tabla

SW [2:0]	Y
000	8'b10101010
001	8'b11001100
010	8'b10011001
011	8´b10100001
	Х
	Х
	Х

Se muestra que la salida Y se hace mediante 8 bits lo que físicamente en la FPGA se representaría por medio de LEDs, y ya que el multiplexor construido (mux de 8 a 1) tiene una sóla salida, se concluye que son necesarios ocho multiplexores de este tipo para la visualización de la salida Y, se puede representar el circuito obtenido de la siguiente manera:





Ahora la descripción en Verilog

De nuevo:



 New Source Wizard Select Source Type Select source type, file name and its location. BMM File ChipScope Definition and Connection File Implementation Constraints File Implementation Constraints File Implementation Constraints File Implementation Constraints File Implementation Constraints Implementation Implementatin the state state state state	File name: Table I Location: C: \FPGA\Multiplexores
More Info	Next Cancel

Ya en la ventana del editor,



Y lo siguiente será describir el circuito completo para que funcione de la misma manera que lo indica la tabla, el código entonces:

```
module Table(
input [2:0]SW,
output [7:0]LED
  );
mux8_1 10(
.D(8'b00110000),
.S(SW[2:0]),
.LED(LED[0])
);
mux8_111(
.D(8'b1000000),
.S(SW[2:0]),
.LED(LED[1])
);
mux8_1 l2(
.D(8'b0100000),
.S(SW[2:0]),
.LED(LED[2])
);
mux8_1 I3(
.D(8'b11100000),
.S(SW[2:0]),
.LED(LED[3])
);
mux8_1 I4(
.D(8'b00100000),
.S(SW[2:0]),
.LED(LED[4])
);
mux8_1 I5(
.D(8'b10010000),
.S(SW[2:0]),
.LED(LED[5])
);
mux8_1 I6(
.D(8'b0100000),
.S(SW[2:0]),
.LED(LED[6])
);
mux8_1 I7(
.D(8'b11110000),
```

Endmodule

Donde,

- Define un bus de ocho salidas usando leds
- Define tres entradas manipuladas a través de tres interruptores
- Indica como constantes todas las entradas de los multiplexores de 8 a 1
- Instancia los ocho multiplexores asignando una configuración diferente para las entradas constantes, según sea necesario y un led diferente para cada salida Y

Ya asignado todo el código necesario, revisar que la sintaxis haya sido adecuada



Ahora es necesario hacer las respectivas referencias a las entradas y salidas, para esto se pueden asignar mediante Xilinx PlanAhead de la siguiente forma



Asignados de acuerdo a la siguiente configuración

RTL Netlist		- 0 Ľ × [Packag	e 🗙 🌸 Device	X 🕅 RT	L Schematic 🗙					
工 州王											
R Table							0	+++++++++++++++++++++++++++++++++++++++		1000	
13 Nets (13)											
Primitives (2)				→ 364 P 20124							
1) (mux8_1)											
⊕-1 12 (mux8_1) ☐											
13 (mux8_1)											
			153							10.	
I/O Port Properties	-				5			10			
P SW[2]											
Properties M Clock F	Regions		2 III								
1/O Ports											
🔍 Name	Direction	Neg Diff Pair	Site	Fixed	Bank	I/O Std	Vcco	Vref	Drive Stre Sk	lew Type	Pull Type
🔀 🖃 🐼 All ports (11)											
📥 🖨 🍓 LED (8)	Output					1 LVCMOS33*	3.3	00	12 SL	.OW	NONE
LED[7]	Output		W21	1		1 LVCMOS33*	3.3	00	12 SL	.OW	NONE
	Output		Y22	1		1 LVCMOS33*	3.3	00	12 SL	.OW	NONE
10 LED[5]	Output		V20	1		1 LVCMOS33*	3.3	00	12 SL	.OW	NONE
	Output		V19	1		1 LVCMOS33*	3.3	00	12 SL	.ow	NONE
📖 🖓 🖓 🗠 🖓	Output		U19	1		1 LVCMOS33*	3.3	00	12 SL	.OW	NONE
	Output		U20	1		1 LVCMOS33*	3.3	00	12 SL	.OW	NONE
	Output		T19	1		1 LVCMOS33*	3.3	00	12 SL	.ow	NONE
	Output		R20	1		1 LVCMOS33*	3.3	00	12 SL	.OW	NONE
🛱 🥸 SW (3)	Input					2 LVCMOS33*					NONE
SW[2]	Input		T9	2		2 LVCMOS33*					NONE
- 🐼 SW[1]	Input		US	3		2 LVCMOS33*					NONE
	Input		U10	2		2 LVCMOS33*					NONE
Galar ports (0)											

O bien, se puede editar directamente un archivo de tipo .ucf que contiene la misma información que permite relacionar el PlanAhead

```
x
  mux8_1 - Notepad
                                         Help
 File Edit Format View
                                                                                                                  .
NET "LED" LOC = R20;
NET "LED" IOSTANDARD = LVCMOS33;
        "LED[7]" LOC = W21;
"LED[6]" LOC = Y22;
"LED[5]" LOC = Y20;
"LED[4]" LOC = V19;
"LED[3]" LOC = U19;
"LED[2]" LOC = U20;
"LED[1]" LOC = T19;
"LED[0]" LOC = R20;
"SW[2]" LOC = T9;
NET "LED[7]" LOC = W21;
NET
NET
NET
NET
NET
NET
NET
NET "SW[2]" LOC = T9;
NET "SW[1]" LOC = U8;
NET "SW[0]" LOC = U10;
NET "LED[7]" IOSTANDARD = LVCMOS33;
        "LED[6]"
NET
                          IOSTANDARD = LVCMOS33:
        "LED[5]" IOSTANDARD = LVCMOS33:
NET
NET "LED[4]" IOSTANDARD = LVCMOS33;
NET "LED[3]" IOSTANDARD = LVCMOS33;
NET "LED[2]" IOSTANDARD = LVCMOS33;
NET "LED[2]" IOSTANDARD = LVCMOS33;
NET "LED[1]" IOSTANDARD = LVCMOS33;
NET "LED[0]" IOSTANDARD = LVCMOS33;
NET "SW[2]" IOSTANDARD = LVCMOS33;
NET "SW[2]" IOSTANDARD = LVCMOS33;
NET "SW[1]" IOSTANDARD = LVCMOS33;
NET "SW[0]" IOSTANDARD = LVCMOS33;
# PlanAhead Generated IO constraints
        "LED[7]" DRIVE = 8;
NET "LED[7]" DRIVE = 8;
NET "LED[6]" DRIVE = 8;
NET "LED[5]" DRIVE = 8;
NET "LED[4]" DRIVE = 8;
NET "LED[3]" DRIVE = 8;
NET "LED[2]" DRIVE = 8;
NET "LED[1]" DRIVE = 8;
NET "LED[0]" DRIVE = 8;
NET
```

NET "SW[0]" IOSTANDARD = LVCMOS33; # PlanAhead Generated IO constraints NET "LED[7]" DRIVE = 8; NET "LED[6]" DRIVE = 8; NET "LED[5]" DRIVE = 8; NET "LED[4]" DRIVE = 8; NET "LED[3]" DRIVE = 8; NET "LED[2]" DRIVE = 8; NET "LED[2]" DRIVE = 8; NET "LED[1]" DRIVE = 8;

NET "LED[0]" DRIVE = 8;

NET "LED[7]" IOSTANDARD = LVCMOS33; NET "LED[6]" IOSTANDARD = LVCMOS33; NET "LED[5]" IOSTANDARD = LVCMOS33; NET "LED[4]" IOSTANDARD = LVCMOS33; NET "LED[3]" IOSTANDARD = LVCMOS33; NET "LED[2]" IOSTANDARD = LVCMOS33; NET "LED[1]" IOSTANDARD = LVCMOS33; NET "LED[0]" IOSTANDARD = LVCMOS33; NET "SW[2]" IOSTANDARD = LVCMOS33; NET "SW[1]" IOSTANDARD = LVCMOS33; NET "SW[1]" IOSTANDARD = LVCMOS33; NET "SW[1]" IOSTANDARD = LVCMOS33;

NET "LED[7]" LOC = W21; NET "LED[6]" LOC = Y22; NET "LED[5]" LOC = V20; NET "LED[4]" LOC = V19; NET "LED[3]" LOC = U19; NET "LED[2]" LOC = U20; NET "LED[1]" LOC = T19; NET "LED[0]" LOC = R20; NET "SW[2]" LOC = T9; NET "SW[1]" LOC = U8; NET "SW[0]" LOC = U10;

NET "LED" LOC = R20; NET "LED" IOSTANDARD = LVCMOS33;

Sintetizar para comprobar que el proceso de asignación de entradas y salidas se haya hecho correctamente



Y para finalizar el proceso de descripción y antes de poder visualizarlo en la FPGA

Implementar diseño:



Generar archivo:



Y para finalizar





En la ventana del ISE iMPACT:



😵 ISE iMPACT (P.20131013) _ O X File Edit View Operations Output Debug Window Help 🗋 🏓 😻 📑 🖬 🥕 🌾 iMPACT Flows ⇔⊡∄× Boundary Scan
 SystemACE
 Create PROM File (PROM File Format...
 WebTalk Data x Automatically create and save a project Do you want the system to automatically create and save a project file for you? ? Don't show this message again, save the setting in preference. Yes No ⇔⊡∄× iMPACT Processes

Yes



With See impact (P.20131013) - [Boundary Scan] _ O X - 8 × 퉳 File 🗋 🆻 🖬 🕺 🛍 🕼 🗙 🗄 📰 🎬 🎽 🔚 🖬 🌽 💅 ? ×
 WPACT Flows
 ↔ □ ♂ ×

 ⊕ 20 Boundary Scan
 ₀ SystemACE

 □ Create PROM File (PROM File Format...
 ⊕ WebTalk Data
 ↔ 🗖 🗗 🗙 🎯 Assign New Configuration File • 3 0 0 📑 🗉 🗉 Look in: 🛛 🕌 C:\FPGA\Multiplexores My Computer hogar hog table.bit planAhead_run_5 xlnx_auto_0_xdb xst mux8_1.bit iMPACT Processes ⇔⊡₽× Available Operations are: Open File name: table.bit Cancel Bypass Cancel All Files of type: All Design Files (*.bit *.rbt *.nky *.isc *.bsd) Identify Succeeded

Open table.bit

Se iMPACT (P.20131013) - [Boundary Scan]	ATT A BEER P ATT THEY THEY. BOT MAN	
File Edit View Operations Output D	ebug Window Help	_ <i>8</i> ×
🗋 🌶 🖬 🐰 🗅 🛅 🗙 🗄 📾 🕸	11 書 😤 💷 🥜 🛠	
MPACT Processes MPACT Processes MPACT Processes MPACT Processes	def def	× Help

OK



Ya debería correr correctamente en la FPGA.