

PRACTICA Nº 4: Multiplexores de forma reducida

En la práctica basándonos en el ejemplo propuesto en la anterior presentación, utilizando los conceptos de multiplexores, se puede realizar la misma programación de una manera más fácil y reducida, pero sin olvidar los fundamentos de estos. En éste ejercicio se utilizaran los 8 led con su

Objetivos:

- Identificar la cantidad de entradas que se van a utilizar y el número de bits que se van a obtener en las salidas.
- Visualizar en la FPGA el programa realizad de manera reducida para verificar que funcione igual al de la práctica anterior.
- Reforzar la programación en "ISE" junto con la conexión de los módulos de "Verilog" de cada ejercicio utilizando el instanciado.
- Aplicar este mismo método en la implementación de un display de 7 segmentos ánodo-común.

Requisitos:

- FPGA Xilinx Spartan 3A Starter Kit, con cable de alimentación y USB.
- Documentos de las prácticas anteriores
- Software instalado y licenciado

Desarrollo de la práctica:

Para esta práctica es necesario tener el ejercicio de la práctica de multiplexores.

En la plantilla anterior (Plantilla_Spartan3A.zip) a utilizaremos para crear en nuevo archivo "Module Verilog".

ISE Project Navigator (P.20131013) - C:\FPGA	s\Semiilero_ADT\Plan	tilla_Spartan3A\Plantil	lla_Spartan3	A.xise - [Design Sur	nmary]		×
File Edit View Project Source Process Tools Window Layout Help							E ×
		a π ¥					
Jesign Overview A			TOP_PO	NG Project Status			
View:	Project File:	Project File: Plantilla_Spartan3A.xise		Parser Errors:		No Errors	
Hierarchy Module Level Utilization	Module Name:	TOP_PONG	I	Implementation State:		nd Routed	
xc3s700a-4fg484	Target Device:	xc3s700a-4fg484		• Errors:			
🔠 🔄 🔽 TOP_PONG (Plantilla_top.v)	Product Version:	ISE 14.7 Balanced		• Warnings:			
instance_name - Ejerciciomuxil	Design Goal:			Routing Results:		All Signals Completely Routed	
m with the state of the state o	Design Strategy:	Xiinx Default (unlocked) System Settings		Timing Constraints:			
🔛 🗤 mux_1 (mux_1.v)	Environment:			Final Timing Score:		0 (Timing Report)	
Translation Messages							
Place and Route Messages		D	evice Utilizatio	on Summary		E.	-1
Timing Messages Bitgen Messages	Logic Utilization		Used	Available	Utilization	Note(s)	
No Processes Running All Implementation Messages	Number of 4 input LUTs			6 11,77	1%		
Processes: TOP_PONG	Number of occupied Slices			3 5,88	8 1%		-
Design Summary/Reports	Number of Slices containing only related logic			3	3 100%	5	-
Design Utilities Design Properties Design Properties Design Properties	Number of Slices containing unrelated logic			0	3 0%	5	
View Command Line Log Optional Design Summary Contents	Total Number of 4 input LUTs			6 11,77	16 1%		-
View HDL Instantiation Te Show Clock Report	Number of bonded IOBs			108 37	2 29%		
User Constraints Sow Failing Constraints Sow Failing Constraints Sow Failing Constraints	Average Fanout of Non-Clock Nets		1 1	0.45			-
Comparison Compariso	2						_
Configure Target Device			Performance	Summary		F-	-
				1	1		- ~
🖌 Start 🔍 Design 🕕 Files 🚺 Libranes 🚬 Design Su	ummary						
Console						+ □	19 ×
(J.INFO:FrojectMgmt - Parsing design hierarchy completed successfully, (J.INFO:HDLComplet:1845 - Analyzing Verilog file "C:/FPGAs/Semillero (J.INFO:HDLComplet:1845 - Analyzing Verilog file "C:/FPGAs/Semillero (J.INFO:HDC:PFCAs/Semillero (J.INFO:FFC)=FCAs/Semillero (J.INFO:FFC)=FCAs/Semillero (J.INFO:FFC)=FCAs/Semillero (J.INFO:FFC)=FCAs/Semillero (J.INFO:FFC)=FCAs/Semillero (J.INFO:FFC)=FCAs/Semillero (J.INFO:FFC)=FCAs/Semillero (J.INFO:FFC)=FCAs/Semillero (J.INFO:FFC)=FCAS/Semillero (J.INFO:FFC)=FCAS/Semillero (J.INFO:FFC)=FCAS/Semillero (J.INFO:FFC)=FCAS/Semillero (J.INFO:FFC)=FCAS/Semillero (J.INFO:FFC)=FCAS/Semillero (J.INFO:FFC)=FCAS/Semillero (J.INFO:FFC)=FCAS/Semillero (J.INFC)=FCAS/Semillero	ADT/Plantilla_Spar ADT/Plantilla_Spar ADT/Plantilla_Spar ADT/Plantilla_Spar	tan3A/Ejerciciomux tan3A/Plantilla_to tan3A/mux2_1.v" in tan3A/mux_1.v" int	i8.v" into p.v" into to library o library	library work library work work work			^
Launching Design Summary/Report Viewer							

Abrimos el programa "Xilinx Spartan 3A" y procederemos a realizar la práctica.

Una vez abierto la plantilla procedemos a crear un nuevo archivo en el "TOP_PONG" en el cual damos "click" derecho. (Ver imagen)

> ISE Project Navigator (P.20131013) - C.\FPGAs\Semiilero_ADT\Plantilla_Spartan3A\Plantilla_Spartan3A.xise - [Design Summary] – 🗖 💌									
📱 File Edit View Project Source Process Tools Window Layout Help									
□ 🖓 I 🖉 🖉 🖉 🖉 🖉 🖉 🖉 💌 🔄 🔜 🐼 スローン 🖉 🖉 🖉 🖉 🖉 🖉 🖉 🖉 🖉									
Design ↔ □ ₽ × 👩 🖨 Design Overview ^	TOP_PONG Project Status								
View:	Project File:	Plantila_Spartan3A.xise	Pa	rser Errors:	No Erro	s			
I Hierarchy Module Level Utilization	Module Name:	TOP_PONG	Im	Implementation State:		Placed and Routed			
Plantilla_Spartan3A	Target Device:	xc3s700a-4fg484		• Errors:					
🔒 🖸 🖸 TO 📑 Na Contraction of the Contraction of	Product Version:	ISE 14.7		• Warnings:					
In: New Source Static Timing Trop and Warning	Design Goal:	Balanced		Routing Results:		All Signals Completely Routed			
mux2 C Auto co	Design Strategy:	Xilinx Default (unlocked)	Xilinx Default (unlocked) • Timing Constraints:		5:				
mux_1 Add Copy of Source	Environment:	System Settings		Final Timing Score:		0 (Timing Report)			
Open Translation Messages									
Remove Remove Place and Route Messages	Remove A Place and Route Messages								
< Manual Compile Order Timing Messages	Logic Utilization	Dev	Ucod	Available	Utilization	Hete/s)	1-1		
No Processes R Set as Top Module	Number of Ainput LUTS		USEU	4 11 776	10	noce(s)	_		
Processes TOP P SmartGuide Detailed Reports	Number of occupied Slices			3 5.999	10	°			

Una vez realizado este paso nos aparece una ventana pequeña en la cual no va a pedir el tipo de archivo que vamos a crear y el nombre que le vamos a poner.

Entonces elegimos la opción "Verilog Module" como se muestra en la gráfica a continuación, para crear un módulo verilog, y lo llamaremos "multiplexreducido" en el cual vamos a realizar la correspondiente programación.

ISE Project Naviga	ator (P.20131013) - C:\FPGAs\Semiilero_ADT	\Plantilla_Spartan3A\Plantilla_Spartan3A.xise -	Plantilla_top.v] _ 🗆 🗙
Ealit View Project Source Process iools Window	Layout Help	> z 🗶 💡	<u> </u>
Design ↔ □ ♂ X € 21 modify If Vew: ○ 锁 Implementation ● 愛 Smulation 22 23 If Heinarchy □ 24 24 If Heinarchy □ 25 24 If Heinarchy □ 26 27 If □ x23700a-ffq484 27 28 If ○ YOP_PONG (Plantilla_top.v) □ 28 If ○ Ymun2_1 (mu2_1.v) 4 30 If ○ Ymun2_1 (mu2_1.v) 31 31	ule TOP_PONG (CLK_SOM, vga_h_sync, vg NORTH, SOUTH, ROT_A, ROT_B, ROT_CENTER, New Sc Celet Source Type Select Source Type Select source type, file name and its location.		
32 32 No Processes Running 34 Processes: TOP_PONG 36 Sim Simulator 37 Behavioral Check Syntax 39 Simulate Behavioral Model 41 1 42 1 44 4 47	BMM File ChipScope Definition and Connection File Implementation Constraints File IP (CORE Generator & Architecture Wizard) MeM File Schematic Schematic Verilog Module Verilog Test Fixture VHDL Module VHDL Module VHDL Vackage VHDL Package	File name: Location: C:\FPGAs\Semilero_ADT\Plantila_Spartan3A	, v
Concole	See Embedded Processor		
Launching ISim simulation engine GUI "C:/FPGAs/Semilero_ADT/Plantilla_Spartan3A/E ISim simulation engine GUI launched successfu Process "Simulate Behavioral Model" completed	More Info	Add to project	GAs/Semiilero_ADT/Plantilla_Spartan3A/Ejerc
Console Perors Add a new source to the project			Ln 21 Col 1 Verilog

Luego nos aparecerá una ventada del programa, en el cual vamos a definir los argumentos de entrada del módulo, además de un bus de tres switches definidos como "[2:0]sw" y un bus de salidas definido como "[7:0]". Seguidamente procederemos a asignar el valor de las salidas preguntando la posición donde en que estén cada el bus de switches, y si se cumple esa posición entonces definir una salida de ocho bits correspondiente a la definidas en la práctica anterior. (Ver imagen)



Después de haber realizado el programa, al igual que en las prácticas anteriores se necesita crear una instanciación, y para ello le damos "click" en el módulo que creamos y escogemos la opción de "View HDL Intantation Template", le damos doble "click" y esperamos a que nos abra la instanciación del programa que acabamos de crear, seguidamente copiamos esa instanciación.



Ya realizado el procedimiento anterior, nos vamos al "TOP_PONG" y le damos doble "click" para que nos abra la plantilla principal y nos dirigimos al final del programa y allí pegamos la instanciación copiada previamente, como se aprecia en la siguiente imagen.



A continuación realizamos el procedimiento de verificación sintáctica del programa como lo hemos ido realizando en las prácticas anteriores, el cual una vez completado exitosamente lo podemos generar y cargar en la FPGA.

Image: No Processes Running 109 110 109 110 110 Processes TOP_PONG 110 111 Ejercicionuxi8 instance_name (112 112 Image: New Command Line Log View HDL Instantiation Te User Constraints 112 .sw(SW(2:01)), 113 .v(ED[7:0]) Image: Non-Section Section Sectio	
≽ Start 📲 Design 🚺 Files 🖺 Libraries 😰 Design Summary (Synthesized) 🗶 📄 mux2_1.v 🛛 🚔 mux_1.v 💟 🚔 Ejercidomuxi8.v 💟 🚔 Ejercidomuxi8.ti 💟 🚔 Plantia_top.v 😫	2
Console	⇔⊡∄×
Minimum input arrival time before Glock: No path found Maximum output required time after clock: No path found Maximum combinational path delay: 7.288ns	^
Process "Synthesize - XSI" completed successfully	v
Console O Frrors A Warnings A Find in Files Results	
	Ln 115 Col 1 Verilog

Por seguridad es recomendable realizar la simulación antes de conectar la FPGA, para revisar que el programa esté funcionando correctamente y en caso contrario realizar una revisión, determinar el error y proceder a realizar el debido ajuste. Para ello activamos el botón de "Simulation", le damos "click" en el módulo creado y luego en "Behavioral Check Syntax" para que verifique que esté escrito bien la sintaxis del programa, para luego darle doble "click" en "Simulate Behavioral Model" para que genere la simulación.



Luego nos aparecerá un programa encargado de hacer la simulación, donde aparecerá los buses de entrar y salida que creamos. Para comprobar que nos delui resultado que esperamos, necesitamos forzar las entrada que serán los "switches", como ya lo hemos hecho en las prácticas anteriores, le damos "click" derecho en los "switches" y escogemos la opción de "force constant", y como estamos trabajando en el sistema binario la dejamos ahí e introducimos una entrada, oprimimos en "Apply" y "Ok".

								1,000,000 ps
Name	Value	1999,99	5 ps	999,996 ps	999,997 ps	999,998 ps	1999,999 ps	1,000,000 ps
Sw [2:0]	ZZZ				ZZZ			
🕨 📑 y[7:0]	XXXXXXX				XXXXXXXXXXXX			
755	Force Selected	d Signal	? ×					
Enter parameter value. Assignme previously applie overridden.	rs below to force ents made from v ed constant or clo	the signal to vithin HDL co ock force will	a constant de or any be					
Signal Name:		/Ejerciciomu	ixi8/sw					
Value Radix		Binary	~					
Force to Value:		000						
Starting at Time Of	fset:	0						
Cancel after Time C	Offset:							
OK	Cancel	Analu	Hala	1				
UK	Cancel	Арріу	neip					>

Escogemos la opción de ejecutar en un determinado tiempo, como se observa a continuación, y observamos que nos dio la respuesta esperada, en cierto intervalo de tiempo.



Una vez realizada la simulación podemos proceder a comprobar realmente el programa, conectando la FPGA y descargándole el programa.