24/9/2013



Manual de Inicio | Miguel Ángel Lozano Pérez

Contenido

1.1 GUÍA DE SOFTWARE	3
1.1.1 Primer paso: crear un diseño nuevo	3
1.1.2 Siguiente paso: añadir un fichero en VHDL.	4
1.1.3 Siguiente paso: Programación	5
1.1.4 Siguiente paso: SIMULACIÓN.	7
Problema resuelto 1.1-1	0
Problema resuelto 1.1-2	4

1.1 GUÍA DE SOFTWARE

Mediante esta guía se pretende obtener un aprendizaje rápido, para el diseño de dispositivos, mediante la programación del diseño de tres formas diferentes:

- Programación en lenguaje de descripción hardware, VHDL.
- Programación mediante la realización de máquinas de estado finitos.
- Mediante bloques.

La implementación de los diseños también se puede desarrollar mediante la combinación de cualquiera de

1.1.1 Primer paso: crear un diseño nuevo.

Desde el menú de inicio o mediante un acceso directo se abre

File->Nuevo->Diseño



Mediante el asistente, se elige Create un Empty Desing with Desing Flow.

a being	
How	rould you like to create Design Resources?
	Create an Empty Design
	Create an Empty Design with Design Row
	C Add existing Resource Files
	C Import a Design from Active CAD
	This option creates an empty design and enables Design Flow Monager You can select a vendor of your synthesis or implementation tool, Inchningly, Mantes, and ageody the default HDL language of your new design entry sources.
-	Create New Workspace C Add Design to Current Washipsace
	CAttilis Siguente > Cancelar

Se crea a partir de un diseño vacio. Pulsar siguiente y poner un nombre al diseño, por ejemplo puerta_and.

ofy additional information about the new design.	Specify basic information about the new design.
(Type the design name:
C-Synthesis tool:	Pueta_and
(none>	Select the location of the design folder:
Synthesis tool	c:\aldec\active hd student edition\mis diseños
(none)	
Burnet Burkerstein	BONS
Physical Synthesis tool:	A SALE OF A
	The name of the default working library of the design:
Implementation tool	Pueta_and
(none)	The name specified here will be used as the file name for the library files and as the logical name of the library. You can
Default Family (stories)	change the logical name later on.
0.000	
How Settings	
Block Diagram Configuration: Default HDL Language	
Default HDL Language VHDL -	
· · · · ·	
< /erás Siguiente > Cano	Atrás Siguente > Ca



r , besyn directory . c. laidec'active-hd nudert editon'mis diseños	Design name: Puerta_and		*
/ Design directory c.'valdec'vactive-hd student edition'imis diseños			Ŧ
	Design directory c \aldec\active-hd student e	dtion'mis diseños	



Se crea el navegador del diseño.

1.1.2 Siguiente paso: añadir un fichero en VHDL.

Se elige desde el navegador y añadimos un fichero vhdl como fuente.

Eile Edit Search View	Worksp	ace	Design	2	mula	tion	I	ools	W	indov
🐻 • 🗭 🗟 🕺 🕷	● ¥		ρ 🔛	8	Π,		T	Ŷ	15	-
Design Browser	+ x									
Top-Level selection	•									
O Unsorted										
Paerta and				•	ŇΨ	HDL	Sou	rce		1
P & Add Fi	sign		1	N V	erilo	g 50	urce			
Comp				* 9	ate l	Diag	ram		1	
City Country	Marris E.a.I.	ALC: Y			No 101	in the	Dia.			

Aparece el asistente. Se pulsa siguiente



Se dispone un nombre para el fichero y de forma opcional el de la entidad y la arquitectura. Si éstos últimos no se pone se realiza por defecto.



Se pulsa siguiente y se van añadiendo los puertos de entrada y salida que posteriormente se integrarán en la entidad.



Se pulsa finalizar y acaba el asistente con la obtención de un fichero vhdl, preparado para ser confeccionado.

ine hadt bagen gew Wingspace Being Smutation Loos Quidow Beng → e B X B S F J D S X S B M F C S S S B S F C S S S S F F F F F F 1 D ns S H H E d IP S (1 quid or Level selection → 2] Overented → Author : Miguel → Author : Miguel → Author : Miguel → - Author : Miguel → - Author : Miguel →	No simulation - 동료 🕫 🕸 🗠		>>) >>
Image: Control of the state of th	No simulation] 윤 윤 👔 🕸 🗠	ବ୍ର୍ଷ୍	33
scipp Browner ★ X 10:00	88 1 000	ୟସ୍ଷ୍	30
p-fevel selection Author : Miguel Unsoled Unsoled			1
Unsorted 6 Company : UBU			
Washington Threads an			1
Protection and I File : Puerta AND.vhd			
Statistics and 9 Generated : Fri Nar 9 11:26:41 2012			
A wad New Field 10 From : interface description file			
- By : Itf2Vhd1 ver. 1.22			
Add New Library D2			
1 Puerta and library			
is Description :			
17			
18 (/ Section below this comment is automatically maintained			
19 and may be overwritten			
20(entity (Puerta_AND2) architecture (Funcionamiento))			
23			
22 library IEEE;			
use IEEE.STD_LOGIC_1164.all;			
17 Design Business 2000 (a			
enti : in STD LOGIC:			
20 ent2 : in STD LOGIC;			
29 sal: out STD_LOGIC			
12			
31 end Puerta_AND2;			
32			
33 architecture Funcionamiento of Puerta AND2 is			
51 Degin			
97			
55 end Funcionamiento;			
39			
40			- 21
		_	
Files / 2 Stru E puerta_and			
spiles selected file	Ln 28, Col 54	NUM INS	5

1.1.3 Siguiente paso: Programación.

A partir de esta plantilla se cumplimenta la arquitectura, En esta caso se elige la descripción de flujo de datos o RTL.

entity Puerta AND2 is
port(ent1 : in STD_LOGIC;
ent2 : in STD_LOGIC; sal : out STD_LOGIC):
end Puerta_AND2;
architecture Funcionamiento of Puerta_AND2 is
sal<= ent1 AND ent2 AFTER 4ns;
end Funcionamiento;

Se programa la arquitectura.

Se compila para determinar errores.

/	Compilar	
<u>File E</u> dit Sea <u>r</u> ch <u>V</u> iew W <u>o</u> rkspace <u>D</u> esign <u>S</u> imulation <u>T</u> ools <u>W</u> indow <u>H</u> elp		
🖉 ▼ 🖉 🖉 🖉 🖉 📲 📓 🌾 🗒 🖓 🧶 🕊 🌾 💭 🔛 🌾 🕲 🕷	▶ ▶I 🛓 100 ns 🕂 📢 🔳 :	≝ ⊪ ¶≣ ⊊≣ ⊊≣
Design Browser 🔥 🔺 🛛 雲 镡 🗑 🦉 🖉 🤴 🍀 🕇 🕮 蒜	X № 🖻 🗠 🗠 🗛	

En la ventana de consola se visualiza si hay o no errores.

* Compile success 0 Errors 0 Warnings Analysis time : 2.0 [s]	*
<pre>= # ELBREAD: Elaboration process. > * ELBREAD: Elaboration time 0.4 [s].</pre>	
G S Console /	
	Ln 28, Col 54 NUM INS

Despues de la compilación desaparece una interrogación para indicar que ha sido realizada.

Aparece un bloque en rojo indicando entidad y arquitectura compilada.

		10
Puerta_AND2 (Fi	uncionamiento)	B
O Unsorted		
Workspace Workspace Puerta_and Add No	e 'Puerta_and': 1 design(: d ew File	s)
1 🖻 🖬 J Puerta,	AND.vhd	
	rrta_AND2 (Funcionamics	nto)
- Add Ne	ew Library	
* 🕤 Puerta	_and library	
•E	: <u>.</u>	
Files 1	Structure ta Resou	irces

1.1.4 Siguiente paso: SIMULACIÓN.

Del menú simulación, se elige inicializar la simulación.

De esta forma aparecen nuevas opciones para realizar la simulación.



Estas mismas opciones aparecen en la barra de menú.

Eile Edit Search View Workspace Design	Simulation Tools Window Help
🐻 * 🚔 📓 🗶 📕 🚳 ¥ 📕 🖓 * 🐻	Initialize Simulation
Design Browser	Initialize Bost Simulation Debug
Puerta_AND2 (Funcionamiento)	Initialize C Code Debug
0 Unsorted	4 Restart Simulation
Workspace 'Puerta_and': 1 design(s) Superta_and	► Bun Alt+F5 ►I Ryn Until
Add New File	E Rup For F5
- 🐔 Add New Library	 IP Go to current simulation time II Pause
Puerta_and library	Sa Irace Into F7
	[Ⅲ Trgce Over F8
	call Trage Out F10
	Run in Aldec Simulator Run in Aldec ALINT
	Toggle Coxerage <u>X</u> trace
	Breakpoints Clear All Breakpoints
	👌 Toggle Breakpoint 59



Se introduce una nueva ventana para visualizar las formas de onda, dentro del menú elegir: file -> new-> waveform

Aparece una nueva ventana para realizar la simulación

Diseño en Electrónica Digital

d) - untitle	arawe.																					_		
nulation	Waveform	Look	Window	w Help					171				- 1										¢ »	×
	C V ID		9 4 9	000	> •	M	1	.00 ns		** =	4	110-		- 4				10	Uns	•	0			
	🖸 🐜 🖬		1.1.1	5.6	馬馬		* #	- 60	11 1	4.56	74.	76	2 0	Te.	llt.	2	1 11 14	č LA	60	<i>.</i>		_		
Signal	same		Value		-	1		- 4	90		•	•	800	1		•	£200				1600	1	•	44
					Ofs																			^
Cursor	1				65																	2	• •	
	Curson	Curser 1	Currer 1	Currer 1	Currer 1	Cuer 1	Cover 1	Cover 1 Cover 1 Color Cover 1 Color Cover 1 Co	Cover 1 ED Cover	Curren 1 ES	Currer 1 ELS	Currer 1 EL	Curren 1 Curren	Currer 1 Currer	Currer 1 Currer	Cover 1 EX	Other Markon	Correct 1 Table Correct 1 Table Table </td <td>Concr 1 ED Concr 1 ED Concr</td>	Concr 1 ED Concr					

Sobre esta ventana hacer clic con el botón derecho del ratón y elegir la opción Stimulators.

Stimulators: Se eligen las señales que se desean estimular y como.



Aparece una ventana para poder definir la excitación de las señales de entrada.



Para ello, dentro del navegador (abajo), se elige la ficha estructura, de esta forma aparecen las señales de entrada y salida

entre otras si las hubiere de nuestro diseño.

Se definen mediante dos teclas, (type: elegir dibujo de teclado) que al pulsalas, cambian de estado de '0' a '1' y viceversa.

- std.TEXTIO		Stimulators		8 23
	uge_tree	Signals Hoticeys Predefined	14	
		Set : ASDB Stimulators	New Remove	
		Signals:	Type:	
		Name Type	Select signals Dook and stimulator type	
e 👘	· · ·		f(t)	
Name	Value		10000	
P enti	0		118	
- sal	U U		Value	
		Display paths Save	Acoly Strength:	¥.
				Close
		L		
		ursor 1	•	
KI (II.			A	
H Files \$ S	Stru/ Seso/	puerta_and with untitled awc		

En el stimulator se han elegido las teclas A y B para excitar a las entradas ent1 y ent2 respectivamente, tal y como se aprecia en la figura.

Cada vez que se define una señal se ha de aplicar y elegir una nueva señal del navegador para definir el tipo de estimulo. Una vez creadas todas se cierra la ventada stimulus.

Posteriormente arrastrar las señales que se desean visualizar hacia la ventana de las formas de onda (waveform) desde el navegador.



Al variar las diferentes señales (con las teclas A y B) se consiguen variar los valores de las señales *ent1* y *ent2* y ver el cambio producido en la salida *sal*, tal y como se aprecia en la figura.



GUÍA PRÁCTICA DEL SOFTWARE de ACTIVE-HDL. Diseño en Electrónica Digital

PROBLEMA RESUELTO 1.1-1



Diseñar y simular con el programa Active-HDL Student Edition una máquina de estados.

Objetivos: Conocer la metodología de diseño SECUENCIAL en un dispositivo PAL.

Conocer los procedimientos de programación de estos dispositivos.

Diseño del fichero VHDL con Active HDL

Diseño del dispositivo en VHDL con arquitectura serie o algorítmica.

Diseñar una máquina de estados (con ACTIVE-FSM), con las señales indicadas.

Se trata de disponer un número cualquiera num de valor entre 1 y 6. (ej 6D "110"B). Si éste coincide con el valor o estado de un dado (contador de 1 a 6) que evoluciona con la señal de reloj, activa la señal de salida led='1' (coincidencia).



El funcionamiento es el siguiente con la señal p='0' el contador evoluciona de 1 a 6.

Se pulsa p, y el contador se para en ese estado, si con el flanco de subida de ésta señal coincide el número dispuesto con el estado del contador se activa la señal de salida led='1'

Se pide realizar el diseño como máquina de estado y simularlo.

Diseño en Electrónica Digital



library IEEE; use IEEE.std_logic_1164.all;

entity pra3 is
port (clk: in STD_LOGIC;
 num: in STD_LOGIC_VECTOR (2 downto 0);
 pulsa: in STD_LOGIC;
 rst: in STD_LOGIC;
 led: out STD_LOGIC;
 sal: out STD_LOGIC_VECTOR (2 downto 0));
end;

architecture pra3_arch of pra3 is

--diagram signal declarations signal aux_sal: STD_LOGIC_VECTOR (2 downto 0);

-- SYMBOLIC ENCODED state machine: Sreg0 type Sreg0_type is (S1, S2, S3, S4, S5, S6); signal Sreg0: Sreg0_type;

-- SYMBOLIC ENCODED state machine: Sreg1 type Sreg1_type is (X1, X2, X3); signal Sreg1: Sreg1_type;

Diseño en Electrónica Digital

Manual de Inicio. 3º GIEA

begin
--concurrent signal assignments
--diagram ACTIONS
aux_sal<=sal;</pre>

Sreg0_machine: process (clk)

begin

```
if clk'event and clk = '1' then
        if rst='1' then
                 Sreg0 <= S1;
                 sal<="001";
        else
        case Sreg0 is
                 when S1 =>
                          sal<="001";
                          if pulsa='0' then
                                   Sreg0 <= S2;
                          elsif pulsa='1' then
                                   Sreg0 <= S1;
                          end if;
                 when S2 =>
                          sal<="010";
                          if pulsa='0' then
                                   Sreg0 <= S3;
                          elsif pulsa='1' then
                                   Sreg0 <= S2;
                          end if;
                 when S3 =>
                          sal<="011";
                          if pulsa='0' then
                                   Sreg0 <= S4;
                          elsif pulsa='1' then
                                   Sreg0 \le S3;
                          end if;
                 when S4 =>
                          sal<="100";
                          if pulsa='0' then
                                   Sreg0 <= S5;
                          elsif pulsa='1' then
                                   Sreg0 <= S4;
                          end if;
                 when S5 =>
                          sal<="101";
                          if pulsa='0' then
                                   Sreg0 <= S6;
                          elsif pulsa='1' then
                                   Sreg0 <= S5;
                          end if;
                 when S6 =>
                          sal<="110";
                          if pulsa='0' then
                                   Sreg0 <= S1;
                          elsif pulsa='1' then
                                   Sreg0 <= S6;
                          end if;
                 when others =>
                          null;
```

Diseño en Electrónica Digital

end case; end if; end if; end process;

```
Sreg1_machine: process (clk)
```

begin

```
if clk'event and clk = '1' then
        case Sreg1 is
                 when X1 =>
                          led<='0';
                          if pulsa='1' then
                                  Sreg1 <= X2;
                          end if;
                 when X2 =>
                          if aux_sal /= num then
                                  Sreg1 <= X1;
                          elsif aux_sal=num then
                                  Sreg1 <= X3;
                          end if;
                 when X3 =>
                          led<='1';
                          if pulsa='0' then
                                  Sreg1 <= X1;
                          end if;
                 when others =>
                          null;
        end case;
end if;
end process;
```

end pra3_arch;



-

PROBLEMA RESUELTO 1.1-2



Diseñar y simular con el programa Active-*HDL Student Edition* una puerta lógica and y un contador bcd..

Realizar y simular un sistema digital, que dispone de una señal de entrada, denominada *GATE*, y un vector de salida de 4 dígitos en BCD, para visualizar el tiempo que ha estado activa a '1' la señal de entrada *GATE*.

Resolver mediante vhdl.

Pasos:

Diseñar una puerta and y un contador BCD.

Integrar mediante bloques. Utilizando los diseños anteriores, realizar el esquema de bloques necesario.

Solución:

Se diseña en vhdl la puerta and y un contador bcd.



Se crea un nuevo disgrama de bloques. Mediante un asistente se pone nombre al diseño y se generan las entradas y salidas al diseño, al igual que la entidad en vhdl.

Al nuevo diseño se le añaden nuevos bloques a partir de entidades realizadas en vhdl.

Las conexiones se realizan de forma sencilla, como en cualquier captura de esquemas y con ayuda del ratón.



GUÍA PRÁCTICA DEL SOFTWARE de ACTIVE-HDL. Diseño en Electrónica Digital

Al final se obtiene el disgrama de bloques deseado en un fichero .bde.



ANEXO A EDICCIÓN POR BLOQUES.

Diagrama de flujo para el diseño.

Creating diagram	Graphical editing of a diagram with Block Diagram Editor. Tell me about Design Rule Check - optional verification of formal correctness of the diagram. Tell me about
Generating HDL code	Automatic translation of the graphical description into HDL code.
Compilation	Compilation of the generated HDL code.
Simulation	Simulation of the HDL code with displaying the results directly on the diagram. I Tell me about

Ventana del editor.

