

Fundamentos de Computadores I

VHDL - Introducción curso 2019-20

Dpto. Arquitectura de Computadores y Automática Universidad Complutense de Madrid

Introducción

■ ¿Que es HDL?

- o Lenguaje específicamente creado para el diseño de circuitos:
 - Nivel de puerta (gate level).
 - Nivel de comportamiento (behavioural level).
- La estructura del lenguaje sugiere el diseño hardware.
- Por qué usar HDL?
 - Poder descubrir problemas en el diseño antes de su implementación física.
 - Dado que la complejidad de los sistemas electrónicos crece exponencialmente, son necesarias herramientas que faciliten el trabajo en equipo y la simplificación del diseño.

"Entity" y "Architecture"

entity nombre_entity is

port (lista de puertos de entrada y salida);

end nombre_entity;

architecture circuito of nombre_entity is

-- señales

begin

-- programación

end architecture circuito;





entity nombre_entidad is port (lista puertos de entrada y salida); end nombre_entidad;



Descripción Estructural



- Esta descripción utiliza entidades descritas y compiladas previamente que llamaremos componentes. \rightarrow "COMPONENT"
- de esta manera podemos reutilizar diseños previos, o realizar diseños para que se reutilicen en otros más complicados.
- Se declaran los componentes que se van a utilizar y después se realizan las conexiones entre los puertos.
- Las descripciones estructurales son útiles cuando se trata de diseños jerárquicos → Diseño "bottom-up": empezamos diseñando los componentes más sencillos y los utilizamos como base para crear los más complejos.

Ejemplo trivial



Queremos implementar la siguiente puerta:



Partimos de las definiciones de una puerta "AND" y "OR".

Componentes "AND" y "OR"

 De momento, sólo nos interesa los puertos en la definición de la entidad.

```
1
    entity component and is
 2
      port(
 3
        entrada1, entrada2: in bit;
 4
       salida: out bit
 5
      );
 6
    end entity;
 7
 8
    architecture arch and of component and is
 9
    begin
10
      salida <= '1' when entrada1 = '1' and entrada2 = '1' else '0';</pre>
11
    end architecture:
12
13
14
    entity component or is
15
      port(
16
         entrada1, entrada2: in bit;
17
       salida: out bit
18
      );
19
    end entity;
20
21
    architecture arch_or of component_or is
22
    begin
23
       salida <= '1' when entrada1 = '1' or entrada2 = '1' else '0';</pre>
24
25
    end architecture;
```

Diseño Bottom-Up

- Ahora construimos nuestro módulo "sumprod" (suma de productos) utilizando los componentes anteriores.
- Primero declaramos los puertos en la definición de la entidad:



27	entity sumprod is
28	port(
29	A, B, C, D: in bit;
30	Z: out bit
31);
32	end entity;

Arquitectura



 En la arquitectura necesitamos declarar primero qué componentes vamos a usar y qué puertos tienen cada uno.

```
architecture arch_sumprod of sumprod is
34
    component component and
35
36
    port(
37
      entrada1, entrada2: in bit;
      salida: out bit
38
39
    );
    end component;
40
    component component or
41
42
    port(
      entrada1, entrada2: in bit;
43
      salida: out bit
44
45
    );
    end component;
46
                                               Las salidas de las puertas
47
                                               "and" son señales internas de
      signal sal_and1, sal_and2: bit;
48
- -
                                               esta arquitectura.
```

Conexiones



```
50
    begin
    and1_pm: component_and port map(
51
    entrada1 => A,
52
    entrada2 => B,
53
    salida => sal and1
54
55 );
56
    and2 pm: component and port map(
57
    entrada1 => C.
58
    entrada2 => D,
59
60
    salida => sal and2
61
    );
62
    or pm: component or port map(
63
    entrada1 => sal and1,
64
    entrada2 => sal and2,
65
      salida => Z
66
67
    );
68
    end architecture;
69
```

Código Completo

- 27 entity sumprod is
- 28 port(
- 29 A, B, C, D: in bit;
- 30 Z: out bit
- 31);
- 32 end entity;
- 33
- 34 architecture arch_sumprod of sumprod is
- 35 component component_and
- 36 port(
- 37 entrada1, entrada2: in bit;
- 38 salida: out bit
- 39);
- 40 end component;
- 41 component component_or
- 42 port(
- 43 entrada1, entrada2: in bit;
- 44 salida: out bit
- 45);

47

- 46 end component;
- 48 signal sal_and1, sal_and2: bit;
- 49 50 **begin**
- 51 and1_pm: component_and port map(
- 52 entrada1 => A,
- 53 entrada2 => B,
- 54 salida => sal_and1
- 55); 56
- 57 and2_pm: component_and port map(
- 58 entrada1 => C,
- 59 entrada2 => D,
- 60 salida => sal_and2
- 61);
- 62
- 63 or_pm: component_or port map(
- 64 entrada1 => sal_and1,
- 65 entrada2 => sal_and2,
- 66 salida => Z
- 67);
- 68
- 69 end architecture;



ModelSim

La herramienta de simulación que vamos a utilizar se llama ModelSim:

м						N	1odelSim PE Studen	t Edition 10.4a								- /	נ
File	e Edit View Compile Simulate Add Wave Tools Layout Bookmarks Window Help																
5	• 🛎 🖬 🛸 🎒	<u>X</u> 🖗 🕻	122 🛇 - A 🗄 🛛 🕸 🛱 🛱	11 🛺 🕺 📋 🕇 🏞 🕇 L 🕈	🕋 🌲 📙 Layout NoDesign		ColumnLayout AllColu	ımn s		🛛 🧠 • 🖓 • 🥵 • 🖓	🤹 🛛 I 🗖 1/0	👔 📶 🥔		• 👪 🗈 i 🗗	」≝ ≝ Ҽ ┙	rees.	al ^r
34	1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1																
ti Lit	arary		:±	🖻 🗶 📚 jects 🛲 🗄 🖻 🗶	Wave - Default												<u> </u>
* Nar	ne	△ Type	Path	▼N1☆ ☆♪	* -	Msgs											
• 1	floatfixlib	Library	\$MODEL_TECH//floatfixlib														
	ieee	Library	\$MODEL_TECH//ieee														
1	ieee_env (empty)	Library	\$MODEL_TECH//ieee_env														
• •	infact	Library	\$MODEL_TECH//infact														
1	mc2_lib (empty)	Library	\$MODEL_TECH//mc2_lib														
1	mgc_ams (empty)	Library	\$MODEL_TECH//mgc_ams														
	modelsim_lib	Library	\$MODEL_TECH//modelsim_lib														
	mtiAvm	Library	\$MODEL_TECH//avm														
	mtiOvm	Library	\$MODEL_TECH//ovm-2.1.2														
	mtPA	Library	\$MODEL_TECH//pa_iD														
	mtiknm	Library	\$MODEL_TECH//rnm														
	mtUPF	Library	\$MODEL_TECH//upt_lip														
	mtuvm	Library	\$MODEL_TECH//uvm-1.10														
	osvvm	Library	\$MODEL_TECH//OSVVIII														
197	sto stal developmentia	Library	SMODEL_TECH//stu														
	std_developerant	Library	MODEL_TECH/./Stu_developerson														
	sv_stu	Library	\$MODEL_TECH/ /svppevs														
100	a synopsys	Library	MODEL_TECH/ /verlog														
	verlig verlig	Library	\$MODEL_TECH/ //bdlopt lib														
	vital2000	Library	SMODEL_TECH//vital2000														
1	work (empty)	Library	C:/Modeltech pe edu 10.4a/example														
		,															

ModelSim



м	
File Edit View Compile	Simulate Add Wave Tools Layout Bookmarks Window Help
New	🕨 Folder 🛛 👪 🖭 🛛 🎲 🚟 🌇 🌠 🕺 🍷 🔉 🛊 🗄 🏩 🔊
Open	Source
Load	Project<
Close	Library
Import	Debug Archive
Export	Prant Create Project X
Save Format Ctrl+S	y \$MODEL_TECH//floatfixlib
Save As	y \$MODEL_TECH//ieee Project Name
Benort	y \$MODEL_TECH//ieee_env
Chappen Directory	y \$MODEL_TECH//infact
Charige Directory	y \$MODEL_TECH//mc2_lib
Use Source	y \$MODEL_TECH//mgc_ams Project Location
Source Directory	y \$MODEL_TECH//modelsim_lib
Datasets	\$MODEL_TECH/avm
	y sMODEL_IECH//ovm-2.1.2
Environment	y smoote_itech/pa_iib Default Library Name
Page Setup	v sMODEL_IECH//mm work
Prige Setup	sMODE_TECH/(nmp11d
Printer.	GODEL_TECH/_instruments Goov Settings From
Print Postscript	v sMODEL TECH/./std
Recent Directories	MODEL TECH//std developerskit pe_edu_10.4a/modelsim.ini Browse
Recent Projects	SMODEL TECH//sv std Copy Library Mappings C Reference Library Mappings
Close Window	y \$MODEL_TECH//synopsys
Quit	y \$MODEL_TECH//verilog
	wary \$MODEL_TECH//vhdlopt_lib OK Cancel
⊥ vital2000 Libr	vrary \$MODEL_TECH//vital2000
work (empty) Libr	vrary C:/Modeltech_pe_edu_10.4a/example



Añadimos el fichero con nuestro código



Compilación



Compilamos y empezamos la simulación:

File Edit View Compile Simulate Add Wave Tools Layout Bookmarks Win Image: Selected Compile Compile Compile Image: Selected	1	м										
Image: Compile - Compile Options Image: Compile Options		File	Edit	View	Compile	Simulate	Add	Wave	Tools	Layout	Bookmarks	Win
SystemC Link SystemC Link Instance Compile All Compile Selected Compile Order Text Jand1_pm Jand2_pm Compile Selected Compile Report Compile Summary Text Sumprod Summary Sumary Summary	🛛 🖬 🕶 🖬 🖏		Compile		*	😳 - 🕯	4 ⊕	🦠 🗳) 🛗 🏭 🖹	<u>د</u> [
im - Default Compile All iminitial supervision Instance Compile Selected in unit type Top Category Visibility Image: Supervision of the supervision		N G 🕸 1		System	C Link	_ +	<u>≁</u> ⊾	₹≸	∃ ∃	* - -	Se Se	
Instance Compile Selected In unit type Top Category Visibility	🔊 sim - Default 💳		Compile	All						±∎×		
Image: symprod Compile Order tecture DU Instance +acc= <full> Image: symprod Compile Order tecture DU Instance +acc=<full> Image: symprod Compile Report tecture DU Instance +acc=<full> Image: symprod Compile Summary tecture DU Instance +acc=<full> Image: symprod Compile Summary tecture DU Instance +acc=<full> Image: symprod Standard Package Package +acc=<full></full></full></full></full></full></full>		₹ Insta	ance		Compile	Selected	jn i	unit type	Top C	Category	Visibility	
+ and1_pm Compile Report tecture DU Instance +acc= <full> + and2_pm Compile Report tecture DU Instance +acc=<full> + or _pm Compile Summary tecture DU Instance +acc=<full> standard standard Package Package +acc=<full></full></full></full></full>		EH	sump	orod	Compile	Order	ited	ture	DU In	stance	+acc= <fi< th=""><th>ull></th></fi<>	ull>
j standard standard Package Package +acc= <full></full>			- <mark>]</mark> and 1_pr - <mark>]</mark> and 2_pr -] or pm		Compile Compile	Compile Report Compile Summary		ture ture ture	DU In DU In DU In	stance stance stance	+acc= <fi +acc=<fi +acc=<fi< th=""><th>< L < L < L</th></fi<></fi </fi 	< L < L < L
			stan	dard	standard		Package	2	Packa	ige	+acc= <fi< th=""><th>ull></th></fi<>	ull>

out Bookmarks Wind 3 🗱 🎬 💭 🕱 🛛 🖪
s 🗱 🕮 💭 就 📗 😡
] 3+ - +€ - ≩- Se
ory Visibility
e +acc= <full></full>
e +acc= <full></full>
e +acc= <full></full>
e +acc= <full></full>

Simulación

 Añadimos las señales que queremos ver en la simulación. En este caso, todas.

м						
File Edit View Com	pile Simulate Add Ob	jects Tools Layou	ut Bookmarks V	Window Help)	
🖹 • 📂 🖶 🎲 🏐	🤞 🖻 鶅 🏠 🔔 🛛 📀	- 🛤 🖺 🐇 🗳	1 🕮 🔊 🕅 🗍	🔁 🕇 🖛 🛤) 📑 🛛 100 ns	
N 🛱 🕹 💷 🖭	∎⊳ よさ***	% 3 5 5 5]:	≷⊷ →€ - ऄॖ• S	Search:	*) 🖏 🥐 📙 🤅
🖉 sim - Default 💷 💴			+ @ >	🖌 💊 Objects		_+&X
Instance	Design unit Design unit t	ype Top Category	Visibility	▼ Name	1¢	🗖 Now 🍽 🕨
	sumprod(ar Architecture	DU Instance	+acc= <full></full>	🔺 🔺 🗛	1	Signal
🕂 🗾 and 1_pm	component Architecture	DU Instance	+acc= <full></full>	🥠 В	View Declaration	
🕂 🗾 and2_pm	component Architecture	DU Instance	+acc= <full></full>	🔶 C	View Memory Conter	nts
庄 🗾 or_pm	component Architecture	DU Instance	+acc= <full></full>	🔔 🕁 🗖	Add Wave	CHLIW
🗾 standard	standard Package	Package	+acc= <full></full>	🔷 🕹 Z	Add Wave New	Cultw
				🔷 sa	Add wave new	
				sa 🔷	Add wave to	•
					Add Dataflow	Ctrl+D
					Add to	•
					UPF	×.
					Сору	Ctrl+C
					Find	Ctrl+F
					Insert Breakpoint	
					Toggle Coverage	•
					Modify	+
					Radix	
					Show	•



No forzamos las salidas.

	•••••	··· · · · · · · · · · · · · · · · · ·		 _ ! >>>>
🔷 Objects		- + 7 ×	📰 Wave - Default 🚃	
▼ Name	14 🗖	Now 🍽 🕨	\$ 1•	Msgs
4 📣 📣	1	Signal		1
🔶 В	0	Signal	▲ /sumprod/B	0
e de la c	0	Signal	↓/sumprod/C	0
	0	Signal	./sumprod/D	0
sal and 1	0	Signal	/sumprod/sal_and1	0
sal and2	0	Signal	/sumprod/sal_and2	0
	-	-	/sumprod/Z	ln
				Object Declaration
				Add 🕨
				Edit 🕨
				View 🕨
				UPF •
				Radix 🕨
				Format 🕨
				Cast to
				Combine Signals
				Group
				Ungroup
				Force
				NoForce
				Clock
				Properties

Entradas



 \circ Por ejemplo, A = 1.

💫 Objects 💷	_ ;;;;;;	- + a ×	📰 Wave - Default 🚃		
* Name	14 🗖	Now 🎦 🕨	*	Msgs	Force Selected Signal
A B C D A Z Sal_and1 A Sal_and2	0 0 0 0 0	Signal Signal Signal Signal Signal	/sumprod/A /sumprod/B /sumprod/C /sumprod/D /sumprod/sal_and /sumprod/sal_and /sumprod/z	Object Declaration Add Edit View UPF Radix Format Cast to Combine Signals Group Ungroup	Signal Name: sim:/sumprod/A Value: 1 Kind
				Force NoForce Clock Properties	

Ejecutar la Simulación



Tenemos varias opciones para simular:



Simulación paso a paso

Proceso:

- Forzamos: A = 1, B = 1, C = 0, D = 0.
- o Run,
- o Forzamos el valor de B a 0
- o Run
- o Forzamos el valor de A a 0
- o Run
- o ...

1 🖛 🖦 📑	100 ns 荣		¥ 🕱 🕼 🛯 🐘 🖉 📗	🕈 🏊 🏌 🗼 🗟	k 🏦 🗍	Layout S	imulate		ColumnL	ayout
f 3• • →£ •	🗞 Seard	h:	v (1) (1)	ତ୍ତ୍ର୍୍ୟ 🖉 占	<u>}</u> 🕵 🛛			J.		
💫 Objects 💷 💴		- + • ×	💶 Wave - Default 🚞							
▼Name	14 🗖	Now 🎷 🕨	谷 🗸	Msgs						
🤹 A	0	Signal		0						
🔶 B	0	Signal	↓/sumprod/B	0						
P C C	0	Signal	🍐 /sumprod/C	0						
	0	Signal .	↓/sumprod/D	0						
sal and 1	0	Signal	Isumprod/sal_and \$	0						
sal_and2	0	Signal	Isumprod/sal_and >	0						
· · · · · · · · · · · · · · · · · · ·		- 9.1-1	📥 /sumprod/Z	0						