# Εισαγωγή στη Verilog με το ISE

Πατάμε new project – Δίνουμε όνομα και κατάλογο όπου θα αποθηκευτεί το project.

Create Nev	v Project	
Specify project	location and type.	
Enter a name	, locations, and comment for the project	
Name:	7segtest	
Location:	C:\Users\stav.ZEUS64\xilinx\7segtest	
Select the typ Top-level sou	be of top-level source for the project arce type:	
HDL		

Next όπου επιλέγουμε chip και preferred language βάζουμε Verilog

Device Properties		
•		
Select the device and design flow for the	project	
Property Name	Value	
Product Category	All	
Family	XC9500 CPLDs	
Device	XC95108	
Package	PC84	
Speed	-15	
Top-Level Source Type	HDL	
Synthesis Tool	XST (VHDL/Verilog)	
Simulator	ISim (VHDL/Verilog)	
Preferred Language	Verilog	
Manual Compile Order		
Enable Enhanced Design Summary		
Enable Message Filtering		
Display Incremental Messages		

Next και στο Create new source το παραλείπουμε πατώντας και πάλι next.

# Προγραμματίζοντας σε Verilog με το ISE της XILINX

🗾 New Proje	ect Wizard		
Croate New S	2017.20		
create new St	Jurce		
You may optionally o additional sources w	reate one source at this time. You can add ith the "Project->New Source" command.	existing sources on the next page	, and later create
Create a new sourc	2 2 2		
	Source File	Туре	New Source
1			Remove

Και πάλι next και μετά finish.

ISE I	SE Project	Navig	ator - C:\	Users\sta	v.ZEUS64\	kilinx\7se	gtest\7s
					× 10	(24   KA	. De
Desig	gn			÷	□ ₽ ×		
	Sources fo	or: Imp	olementatio	n	•		
1	Hierarch	/					
		segtes	t 3-15PC84		New Sour Add Sour Add Copy Manual C File Name Display Fr	ce ce y of Sourc compile C es ull Paths	:e Irder
	< No singl	e desi Desi	III gn modul gn Utilitie	e is select	Design Pr	operties	

Πατάμε verilog module και δίνουμε και ένα όνομα.

Mew Source Wizard		
Select Source Type Select source type, file name and its location. Schematic User Document Verling Test Fixture VHDL Module VHDL Library VHDL Package VHDL Test Bench	File name: 7segl Location: C:\Users\stav.ZEUS64\xilmx\7segtest Xdd to project	
More Info		Next Cancel

Next και στο define module πάλι δίνουμε Next και μετά finish. Τώρα στο κυρίως παράθυρο ανοίγει ο editor με τα αρχικά για το module 7seg.v. Εκεί γράφουμε τον κώδικα και μετά ελέγχουμε την ορθότητα.

	File Edit View Project Source Process	Tools	Window	Help
	∾ ~ X û û X   § ¶ <b>6 1</b>	A	₽₽¥	X 🔎 🗟 🔁 🖻
Desi	gn ↔ □ ♂ ⊃ Sources for: Implementation Hierarchy		1 2 3 4	<pre>`timescale 1ns / ///////////////////////////////////</pre>
	☐ 7segtest xc95108-15PC84 ☑ xc95108-15PC84 ☑ v u v v v v v v v v v v v v v v v v v	× * * * >  112  001  012	5 6 7 8 9 10 11 12 13 14 15	<pre>// Create Date: // Create Date: // Module Name: // Project Name: // Target Devices // Tool versions: // Description: // // Dependencies: // Col versions</pre>
<b>弾</b> 戦			16 17 18 19 20 21 22 23 24	<pre>// Revision: // Revision 0.01 // Additional Com // module ss_seg(ss_ //input a1;</pre>
	Configure Target Device     Configure Target Device     Optional Implementation Tools	Rel Rel Sto Vie For Op De	n Run run All p w Text Repo rce Process en Without sign Goals & press Prope	y ss_c s_out Up-to-Date Updating & Strategies

Αν όλα πάνε καλά δημιουργείται ένα tab με πληροφορίες για το project και τη χρήση πόρων.

Ē

	_		
<u> </u>		<b>A</b> .'	
$\rightarrow \tau \alpha \square$	$nnc \rightarrow$		

<b>XILINX</b>	CPLD Reports XC9500								
	Fitter Report   Timing	Report							
ter Report			Su	mmary					
	Design Na	ime	ss_se	ss_seg					
	Fitting Sta	Fitting Status Software Version Device Used							
	Software				L.33				
	Device Us				XC95108-15-PC84				
	Date	3-26	3-26-2010, 0:23AM						
			RESOUR	CES SUMMARY					
ation	Macrocells Used	Pterms Us	ed Regis	sters Used	Pins Used	Function Block			
olay Style	7/108 (7%)	(7%) 0/540 (0%)		08 (0%)	7/69 (11%)	0/216 (0%)			
DL 🗾	PIN RESOURCES								
	Signal Type	Required	Mapped	Pin Ty	oe Used	Total			
	Input	0	0	I/O	7	64			
	Output	7	7	GCK/IO	0	3			
	Bidirectional	0	0	GTS/IO	0	2			
	GCK	0	0	GSR/IO	0	1			
	GTS	0	0						
	GSR	0	0						
			GLOBA	RESOURCES					
	Global clock net(s) us	ed		0					

Στο Inputs βλέπουμε την χρήση εισόδων και στο logic τη χρήση εξόδων και pin στο chip.

ame Tota Pterm	Total	Function			1				
	s Inputs	Block	Macrocell	Power Mode	Slew Rate	<u>Pin</u> Number	Pin Type	Pin Use	Reg Init State
0	0	FB1	MC9	STD	FAST	6	I/O	0	
0	0	FB6	MC2	STD	FAST	45	I/O	0	
0	0	FB2	MC2	STD	FAST	71	I/O	0	
0	0	FB3	MC2	STD	FAST	14	I/O	0	
0	0	FB4	MC2	STD	FAST	57	I/O	0	
0	0	FB5	MC2	STD	FAST	32	I/O	0	
0	0	FB1	MC2	STD	FAST	1	I/O	0	
			0 0 FB6 0 0 FB2 0 0 FB3 0 0 FB4 0 0 FB4 0 0 FB5 0 0 FB1	0         0         FB6         MC2           0         0         FB2         MC2           0         0         FB3         MC2           0         0         FB4         MC2           0         0         FB5         MC2           0         0         FB5         MC2           0         0         FB1         MC2	0         0         FB6         MC2         STD           0         0         FB2         MC2         STD           0         0         FB3         MC2         STD           0         0         FB4         MC2         STD           0         0         FB4         MC2         STD           0         0         FB5         MC2         STD           0         0         FB1         MC2         STD	0         0         FB6         MC2         STD         FAST           0         0         FB2         MC2         STD         FAST           0         0         FB3         MC2         STD         FAST           0         0         FB4         MC2         STD         FAST           0         0         FB4         MC2         STD         FAST           0         0         FB5         MC2         STD         FAST           0         0         FB1         MC2         STD         FAST	0         0         FB6         MC2         STD         FAST         45           0         0         FB2         MC2         STD         FAST         71           0         0         FB3         MC2         STD         FAST         14           0         0         FB4         MC2         STD         FAST         57           0         0         FB5         MC2         STD         FAST         32           0         0         FB1         MC2         STD         FAST         1	0         0         FB6         MC2         STD         FAST         45         I/O           0         0         FB2         MC2         STD         FAST         71         I/O           0         0         FB3         MC2         STD         FAST         14         I/O           0         0         FB4         MC2         STD         FAST         57         I/O           0         0         FB5         MC2         STD         FAST         32         I/O           0         0         FB1         MC2         STD         FAST         1         I/O	0         0         FB6         MC2         STD         FAST         45         I/O         O           0         0         FB2         MC2         STD         FAST         71         I/O         O           0         0         FB3         MC2         STD         FAST         71         I/O         O           0         0         FB3         MC2         STD         FAST         14         I/O         O           0         0         FB4         MC2         STD         FAST         57         I/O         O           0         0         FB5         MC2         STD         FAST         32         I/O         O           0         0         FB1         MC2         STD         FAST         1         I/O         O

Επειδή ορίζει με δική του πρωτοβουλία θα φτιάξουμε αρχείο ucf που θα λέμε ποια pin να συνδεθούν στο κάθε σήμα. Πάμε στο Design στο αρχείο του module και new source.



Πατάμε Implementation Constraints File και δίνουμε το όνομά του. Μετά Next και Finish.

	61 (			↔ □ ₽ ×
1	Sources	for: Implement	tation	-
	Hierarc	hy		
	- ē	7segtest	104	
		V ss sea	.84 (7sea.v)	
2112		ss_sec	ucf (ss_seg.ucf	)
-				
3				
	4		11	- +
	Process		И.	•
	Process	ies: ss_seg.ucf	II.	×
	<ul> <li>✓</li> <li>Process</li> <li>✓</li> </ul>	ies: ss_seg.ucf User Cons ■ Edit Co	II traints	•
	۲ Process	res: ss_seg.ucf User Cons <sup>®</sup> Edit Co	n traints onstraints (Text)	•

Διπλό κλικ στο Edit Constraints (Text) και ανοίγει κενό tab όπου γράφουμε τα παρακάτω :

NET "ss_out<0>"	LOC = "P14";
NET "ss_out<1>"	LOC = "P15" ;
NET "ss_out<2>"	LOC = "P17";
NET "ss_out<3>"	LOC = "P18" ;
NET "ss_out<4>"	LOC = "P19" ;
NET "ss_out<5>"	LOC = "P20" ;
NET "ss_out<6>"	LOC = "P21";

Αφού το αποθηκεύσουμε πατάμε και πάλι Implement Design. Αν όλα πάνε καλά βλέπουμε το εξής :



# Εγγραφή στο chip

Πατάμε Configure Target Device και ανοίγει το Impact.

₽t	Processes: ss_seg		
	Design Summary/Reports Design Utilities Create Schematic Symbol View Command Line Log File View HDL Instantiation Template User Constraints Floorplan IO - Pre-Synthesis Floorplan IO - Pre-Synthesis View RTL Schematic View RTL Schematic View Technology Schematic Check Syntax Check Syntax Fit Configure Target Device Manage Configuration Project (iN Manage Configuration Project (iN Optional Implementation Tools		Design Properties Design Properties Display Incremen Enable Message I Optional Design Summa Show Clock Repo Show Failing Show Earings Show Errors Run ReRun
			Rerun All
		맔	Stop
De	sign Files Libraries		View Text Report
Cons	ole		Force Process Up-to-Date
I	aunching : "Configure Target Device		Open Without Updating
I	Process "Configure Target Device" co	<b>9</b> 2	Design Goals & Strategies Process Properties

File - New Project. Στην ερώτηση Yes

### Προγραμματίζοντας σε Verilog με το ISE της XILINX

Welcome to iMPACT	aller fas halen		×
Configure devices using	n the list below g Boundary-Scan (JTAG	5)	
Automatically	connect to a cable and	identify Boundary-Scan chain 🖉	
Prepare a PROM File			
Prepare a System ACE	File		
Prepare a Boundary-S	can File		
	SVF		
Configure devices			
	using Slave Serial mo	ode 👻	
_			
	OK	Cancel	

# Πατάμε ΟΚ

SE iMPACT - D:\Xilinx\auto_project.ipf - [Boundary Scan]	COLUMN AND AND AND A	
🚱 File Edit View Operations Output Debug Wind	ow Help	- 5 ×
🗋 🤌 🗐 🔓 📰 💥 🌐 🗟 🖬 🥬 😢		
MPACT Flows	Right click device to select operations	
IMPACT Processes Available Operations are:	Auto Assign Configuration Files Query Dialog  Do you want to continue and assign configuration files(s)?  Don't show this message again, save the setting in preference.  Yes No	
Console PROCEPTSS_ENDEnd_Oneration	Boundary Scan	+ □ # ×
PROGRESS_END - End Operation. Elapsed time = 0 sec. // *** BATCH CMD : identifyMPM (	Configuration Para	*

Configuration Par Πατάμε Yes και ψάχνουμε για το αρχείο .jed που βρίσκεται στον φάκελο του project.

#### Σταύρος Σ. Φώτογλου

ISE iMPACT - D:\Xilinx\auto_project.ipf - [Bc	oundary Scan]	@ [] * #	BOBUTH V.		
File Edit View Operations Output	Debug Window Help				- 5 ×
▲ 🖬 🗙 🖬 🖾 🗙 📰 💥	: : : : : : : : : : : : : : : : : : : :	PN?			
iMPACT Flows	↔□₽×				
Boundary Scan     Boundary Scan     SlaveSerial     SystemACE     PROM File Formatter		TDI X SS TDO	c95108 _segjed		
MPACT Processes Available Operations are: Frase Blank Check Readback Get Device ID Get Device Checksum Get Device Signature/Usercode	Device Programming Pro Category     Boundary-Scan     Device 1 (CPLD xc9)	si08)	Programming Properties Property Name Verify General CPLD And PROM Properties Erase Before Programming Read Protect CPLD Specific Properties Write Protect Functional Test	Value V U U U U U U U U U U U U U U U U U U	
Console J) INFO: iMPACT: 501 - '1': Addec		•	< m	Cancel Apply Help	
Console Errors Warnings				Configuration Parallel III 200	• • • • • • • • • • • • • • • • • • •

### Πατάμε ΟΚ.



Μετά δεξί click πάνω στο chip και Program. Αν όλα πάνε καλά θα βγάλει :

#### Προγραμματίζοντας σε Verilog με το ISE της XILINX

SE iMPACT - D:\Xilinx\auto_project.ipf - [Boundary Scan]	
🛞 File Edit View Operations Output Debug Window Help	- 8 ×
	J# 167
MPACI Hows ↔ □ P ∧	
Boundary Scan	6a #4200000000
Direct SPI Configuration	
SystemACE	TDI 2 XILINX"
PROM File Formatter	<b>******</b>
	xc95108
	ss_seg.jed
	тро
MPACT Processes ↔ □ & ×	
Available Operations are:	
Program	
Verify	
Erase	D
Blank Check	Program Succeeded
Readback	
Get Device ID	
Get Device Checksum	Boundary Scan
A Got Davico Signaturo/Ilcorcado	
Console	↔□∄×
'1': Programming completed successfully.	A.
PROGRESS END - End Operation.	
Liapseu cime - 12 sec.	
<	
Console Errors Warnings	
Console Linois Warnings	
	Configuration Parallel III 200 KHz   LP11

# Μερικά βασικά για την verilog

Για συνδυαστικά κυκλώματα μπορώ να χρησιμοποιώ δομές assign ή always. Για ακολουθιακά μόνο always. Καλό να αποφεύγουμε τις always αν αυτό υλοποιείται και με assign. Ο παρακάτω κώδικας εμφανίζει το γράμμα a ή b σε ένα 7 segment display ανάλογα με το bit a1. To bit a2 απλώς ανάβει ή όχι το decimal point.

```
module ss_seg(a1, a2, ss_out, ss_dp);
input a1, a2;
output [6:0] ss out;
output ss_dp;
wire al, a2, ss_dp;
reg [6:0] ss_out;
assign ss dp = a2;
always @ (a1)
begin
   //ss out = 0;
   if (a1 == 1'b0)
       ss_out = 7'b1111101; //a char
   else
       ss_out = 7'b1011110; //b char
end
endmodule
```

```
Aκολουθεί το ίδιο κύκλωμα φτιαγμένο μόνο με εντολές assign.
module ss_seg(a1, a2, ss_out, ss_dp);
input a1, a2;
output [6:0] ss_out;
output ss_dp;
wire a1, a2, ss_dp;
```

```
assign ss_dp = a2;
assign ss out = (a1 == 1'b1) ? 7'b1111101 : 7'b1011110;
```

#### endmodule

Ακολουθεί κώδικας για binary to 7 segment decoder. Με τα dip switches 1-4 δίνω είσοδο στο δυαδικό και βλέπω στο display τα ψηφία 0-9. Αν η είσοδος είναι > 9 τότε εμφανίζει το μείον (-). module ss\_seg(inp, dp, ss\_out, ss\_dp);

```
input [3:0] inp;
input dp;
output [6:0] ss_out;
output ss_dp;
wire [3:0] inp;
wire dp, ss_dp;
reg [6:0] ss_out;
assign ss dp = dp;
always @ (inp)
begin
   ss_out = 0;
   case (inp)
              4'b0000: ss_out = 7'b0111111; //0
              4'b0001: ss_out = 7'b0110000; //1
               4'b0010: ss_out = 7'b1101101; //2
              4'b0011: ss_out = 7'b1111001; //3
              4'b0100: ss_out = 7'b1110010; //4
               4'b0101: ss out = 7'b1011011; //5
               4'b0110: ss out = 7'b1011111; //6
               4'b0111: ss_out = 7'b0110001; //7
               4'b1000: ss_out = 7'b111111; //8
              4'b1001: ss_out = 7'b1111011; //9
              default: ss_out = 7'b1000000; //-
    endcase
end
```

endmodule

#### Και το αρχείο ucf.

	1 70	
NET	"ss_out<0>"	LOC = "P14";
NET	"ss out<1>"	LOC = "P15";
NET	"ss out<2>"	LOC = "P17";
NET	"ss out<3>"	LOC = "P18";
NET	"ss out<4>"	LOC = "P19";
NET	"ss out<5>"	LOC = "P20";
NET	"ss out<6>"	LOC = "P21";
NET	"ss dp"	LOC = "P23";
NET	"inp<0>"	LOC = "P1";
NET	"inp<1>"	LOC = "P2";
NET	"inp<2>"	LOC = "P3";
NET	"inp<3>"	LOC = "P4";
NET	"dp"	LOC = "P5";

Αν θέλω δεκαδικό ή δεκαεξαδικό σύστημα το τροποποιώ ως εξής:

case (inp)
 4'd0: ss\_out = 7'h3f; //0
 4'd1: ss\_out = 7'b0110000; //1
 4'd2: ss\_out = 7'b1011011; //2
 4'd3: ss\_out = 7'b1110010; //3
 4'd4: ss\_out = 7'b1110010; //4
 4'd5: ss\_out = 7'b1011011; //5
 4'd6: ss\_out = 7'b1011011; //6
 4'd7: ss\_out = 7'b0110001; //7
 4'd8: ss\_out = 7'b111011; //8
 4'd9: ss\_out = 7'b111011; //9
 default: ss\_out = 7'b1000000; //-

Το ίδιο αλλά μόνο με assign:

module ss\_seg(inp, dp, ss\_out, ss\_dp);

input [3:0] inp; input dp;

## Διαιρέτης / απαριθμητής

Δέχεται στην είσοδο ρολόι συχνότητας 244,140625 Hz από το HC4060 και βγάζει στην έξοδο παλμό 0,953 Hz. **Προσοχή** το = μέσα σε δομή always είναι blocking εντολή, πράγμα που σημαίνει ότι οι εντολές εκτελούνται με τη σειρά. Το <= είναι no blocking που αφήνει να εκτελεστούν παράλληλα.

```
module counter(clk, tc);
input clk;
output tc;
wire clk;
reg [7:0] cnt;
reg tc;
always @ (posedge clk)
begin
 tc = 0;
     if (cnt == 8'hff)
        begin
          tc = 1'b1;
          cnt = 8'h00; //ή cnt <= 8'h00;
        end
     else
        cnt = cnt + 1; //ή cnt <= cnt + 1;
end
endmodule
              ======= ucf ==
NET "clk"
                                    LOC = "P9";
NET "tc"
                                    LOC = "P71";
```

Μετρητής 12 bit που μόνο τα 4 σημαντικά ψηφία συνδέονται σε pin εξόδου module counter(clk, out);

```
input clk;
output out;
wire clk;
reg [11:0] cnt;
reg [3:0] out;
always @ (posedge clk)
begin
    out = 0;
        if (cnt == 12'h9ff)
            cnt = 12'h9ff)
            cnt = 12'h000;
        else
            cnt = cnt + 1;
        out = {cnt[11],cnt[10],cnt[9],cnt[8]};
end
endmodule
```

NET	"clk"	LOC	=	"P9";
NET	"out<0>"	LOC	=	"P14";
NET	"out<1>"	LOC	=	"P15";
NET	"out<2>"	LOC	=	"P17";
NET	"out<3>"	LOC	=	"P18":

### Δουλεύοντας με submodules

Καλό είναι να σπάμε το project σε κομμάτια τα οποία επικοινωνούν μεταξύ τους. Θα δούμε πως θα βάλουμε τον προηγούμενο counter να οδηγεί το bcd to 7 segment και να μετράει από 0 έως 9. Στο design βλέπουμε το συνολικό module main και τα δύο submodules.



#### Kαι ο κώδικας είναι ο εξής : Submodule counter

```
module counter(clk, out);
input clk;
output out;
wire clk;
reg [11:0] cnt;
reg [3:0] out;
always @ (posedge clk)
begin
    out = 0;
        if (cnt == 12'h9ff)
            cnt = 12'h9ff)
            cnt = 12'h000;
        else
            cnt = cnt + 1;
        out = {cnt[11],cnt[10],cnt[9],cnt[8]};
end
```

```
endmodule
```

#### Submodule bcd2ss

(inp == 4	'd3) ?	7'b1111001	:	//3			
(inp == 4	'd4) ?	7'b1110010	:	//4			
(inp == 4	'd5) ?	7'b1011011	:	//5			
(inp == 4	'd6) ?	7'b1011111	:	//6			
(inp == 4	'd7) ?	7'b0110001	:	//7			
(inp == 4	'd8) ?	7'b1111111	:	//8			
(inp == 4	'd9) ?	7'b1111011	:	7'b1000000;	//9	:	default

endmodule

#### Κύριο module

<pre>input clk; output [6:0] ss_out;</pre>			
wire [3:0] out;			
<pre>counter counter_instance (clk, out); bcd2ss bcd2ss_instance (out, ss_out); endmodule</pre>	;		
Αρχείο ucf			
NET "clk"	LOC	=	"P9";
NET "ss out<0>"	LOC	=	"P14";
NET "ss out<1>"	LOC	=	"P15";
NET "ss out<2>"	LOC	=	"P17";
NET "ss out<3>"	LOC	=	"P18";

### Simulation

NET "ss out<4>"

NET "ss\_out<5>"

NET "ss\_out<6>"

Στο Design πατάμε το listbox sources for και επιλέγουμε Behavioral Simulation. Πριν που γράφαμε τον κώδικα ήταν Implementation.

LOC = "P19";

LOC = "P20";

LOC = "P21";

Desi	gn	↔□₽×	0
口样	Sources for:	Implementation	-
E Hierarchy		Implementation	C
	<ul> <li>☐ fc</li> <li>☐ xc9!</li> <li>☐ · ♥:</li> </ul>	Behavioral Simulation Post-Fit Simulation 5108-15P-084 decade_counter (decade_counter.v) decade.ucf (decade.ucf)	<b>U</b> <b>U</b> 3

Στο module που θα γίνει εξομοίωση πατάμε δεξί κλικ και New Source.

Desig	jn	⇔□₽×	👝 🗉 Desi
*	Sources for: Behavioral Simulation	•	
6	Hierarchy		
	🖳 🔄 fc		Error
	🖻 🗂 xc95108-15PC84		0
00	🔤 🛛 🗤 decade_counter (decade_counter.v)		<b>.</b>
E		Mew Source	2
-		🚛 Add Source	
		Add Copy o	of Source

Από το παράθυρο που ανοίγει επιλέγουμε Verilog Test Fixture και δίνουμε ένα όνομα και Next. Μετά ανοίγει ένα νέο παράθυρο που προτείνει Associate Source και έχει επιλεγμένο το decade\_counter. Πατάμε Next και στο επόμενο finish.

#### Προγραμματίζοντας σε Verilog με το ISE της XILINX

New Course Million	1. 1	Carat Recept (N.S.	Strage Strategy	Server server	×
Select Source Type Select source type, file name and its location. Schematic Schematic Verilog Module Verilog Module Verilog Module Verilog Module VHDL Module VHDL Module VHDL Module VHDL Dackage VHDL Test Bench		File name: fctest Location: C:\Users\v	stav.ZEUS64\xilinx\fc		
More Info				Next	Cancel

Δημιουργεί αυτόματα κάτι που μοιάζει με το ακόλουθο.

Design 🗰 🕂 🗖 i	🗗 🗙 🚛 🛛 20	) // Revision 0.01 - File Created
Sources for: Behavioral Simulation	21	I // Additional Comments:
El Historehu	22	2 //
	23	3 /////////////////////////////////////
	<u>n</u> 24	
🖃 🖾 xc95108-15PC84	25	5 module fctest;
□ V fctest (fctest.v)	= 26	5
uut - decade_counter (decade_counter	er.v) 27	// Inputs
	- 28	reg clk;
0	1 29	) reg res;
D)	30	
	*24 31	1 // Outputs
	32	2 wire [3:0] out;
	: 🎋 33	3 wire ov;
	34	
	35	// Instantiate the Unit Under Test (UUT)
	36	decade_counter uut (
	37	/ .clk(clk),
	38	.res(res),
	39	) .out(out),
<	+ 40	) .00 (00)
W Deserves fatest	41	);
Processes: rctest	42	
🚛 🖃 🎬 🔤 ISim Simulator	43	initial pegin
Behavioral Check Syntax	44	I // Initialize inputs
Simulate Behavioral Model	43	CLK = 0;
-	- 40	res = 0;
11	4	// Whit 100 ms for slobal report to finis
	40	7/ wait 100 hs for global reset to finis tion.
	49	3 #100;
	50	// ldd stimulus here
	51	// Add Stillards Here
	54	and and
	53	
	5	: endmodule
	0.00	J CHEMOLOGIC

Αν τρέξουμε το Simulate Behavioral Model βγάζει διάφορα λάθη. Γι' αυτό πριν το module fctest; βάζουμε `include "decade\_counter.v" και τώρα γίνεται όπως παρακάτω.



```
Πριν το τέλος του module προσθέτουμε :
// Clock generator
always begin
#5 clk = ~clk; // Toggle clock every 5 ticks
end
```

Για να δουλέψει σωστά το simulation χωρίς λάθη, στο hierarchy επιλέγω το fctest (fctest.v) και μετά κάτω στο Processes τρέχω το Simulate Behavioral Model.

Μετά προσθέτω λίγο κώδικα στο initial begin για να δώσω αρχικές τιμές και να κάνω reset. Ο τελικός κώδικας είναι ο παρακάτω :

```
`include "decade_counter.v
module fctest;
       // Inputs
       reg clk;
       reg res;
       // Outputs
       wire [3:0] out;
       wire ov;
       // Instantiate the Unit Under Test (UUT)
       decade counter uut (
               .clk(clk),
               .res(res),
               .out(out),
               .ov(ov)
       );
       initial begin
              // Initialize Inputs
               clk = 0;
               res = 0;
               // Wait 100 ns for global reset to finish
               #100;
               // Add stimulus here
               #7 res = 1; // Assert the reset
               #6 res = 0; // De-assert the reset
```



					115 (	000 ps					
Name	Value	80	0 000 ps	100 000 ps		120 000 ps	140 000 p	s	160 000 ps	180 000 ps	200 000 ps
🔚 clk	1										
🐻 res	0										
▶ 📷 out[3:0]	0001		XXXX	0000	00	01 <u>X 0010 X</u>	0011 🗙 01	00 🗙 01	01 X 0110 X 0	111 X 1000 X	1001 X 0000 >
U <sub>o</sub> ov	0										

Εδώ χρησιμοποίησα το ISIM που τρέχει μόνο στην 32 bit έκδοση. Αν έχω και το ModelSim το δηλώνω στο design properties και χρησιμοποιώ αυτό αντί του ISIM.

# Χρήση module



#### Το κύριο module

`include "decade\_counter.v"
module full\_counter(mclk, mrst, out\_cnt, mov);
input mclk, mrst;
output [7:0] out\_cnt;
output mov;
//wire [3:0] out1;
//wire [3:0] out2;
wire ov1;
decade\_counter counter1 (mclk, mrst, out\_cnt[3:0], ov1);
decade\_counter counter2 (ov1, mrst, out\_cnt[7:4], mov);
//assign out\_cnt = {out2, out1};
endmodule

#### To submodule

```
module decade_counter(clk, res, out, ov);
input clk, res;
output [3:0] out;
output ov;
wire clk, res;
reg [3:0] out;
reg [3:0] cnt;
reg ov;
always @ ( posedge clk or posedge res)
begin
  out = 0;
  ov = 0;
  if (res)
    begin
           out = 4'b0000;
                ov = 1'b0;
                cnt = 4'b0000;
         end
  else
    begin
      cnt = cnt + 1;
               ov = 1'b0;
                 if (cnt == 4'd10)
         begin
           cnt = 4'd0;
                ov = 1'b1;
        end
      out = cnt;
    end
end
endmodule
```

To ucf ανάβει τα τμήματα ενός 7 segment ως εξής : μονάδες 1-a, 2-b, 4-g, 8-f και δεκάδες 1-c, 2-d, 4-e, 8-decimal point και mov – ένα led.

NET	"mclk"	LOC	=	"P10";
NET	"mrst"	LOC	=	"P74";
NET	"out_cnt<0>"	LOC	=	"P14";
NET	"out_cnt<1>"	LOC	=	"P15";
NET	"out_cnt<2>"	LOC	=	"P21";
NET	"out_cnt<3>"	LOC	=	"P20";
NET	"out_cnt<4>"	LOC	=	"P17";
NET	"out_cnt<5>"	LOC	=	"P18";
NET	"out_cnt<6>"	LOC	=	"P19";
NET	"out_cnt<7>"	LOC	=	"P23";
יתיידוא	"mott"	TOC	_	"D71".

#### και για simulation



module tst;

```
// Inputs
       reg mclk;
       reg mrst;
       // Outputs
       wire [7:0] out_cnt;
       wire mov;
        // Instantiate the Unit Under Test (UUT)
        full_counter uut (
               .mclk(mclk),
.mrst(mrst),
                .out cnt(out cnt),
                .mov(mov)
        );
        initial begin
               // Initialize Inputs
                mclk = 0;
mrst = 0;
                // Wait 100 ns for global reset to finish
                #10;
                // Add stimulus here
      #7 mrst = 1; // Assert the reset
#6 mrst = 0; // De-assert the reset
       end
       always begin
   #5 mclk = ~mclk; // Toggle clock every 5 ticks
end
```

#### endmodule

000 ps
X 00

# Σύνθεση με χρήση σχημάτων

Ένας πολύ ωραίος τρόπος για σύνθεση με χρήση πολλών modules είναι τα σχέδια. Αρχικά εισάγω τον κάθε module ξεχωριστά σε πηγαίο κώδικα verilog. Μετά το κάνω σχήμα από Processes – Design Utilities – Create Schematic Symbol.

Sources for: Implementation		-	X								
Hierarchy			-1	divide							
🧧 fc			abo	cik in c							
😑 🛄 xc95108-15TQ100	- <u>-</u>										
📄 🔁 👬 counter4	- 10										
🛛 💟 divder1 - divider	⊢	: L									
🛛 💟 cnt1 - decade_counter	-										
🛛 💟 cnt2 - decade_counter		-									
🔤 💟 ltch1 - latch			Ð								
🔤 💟 ltch2 - latch											
mux1 - mux			۵								
bcd_to_7seg1 - bcd2ss			~								
🔤 😋 general.ucf		;	Ľ								
			0								
			/								
		- F									
Processes: counter4	_										
Design Summary/Reports			A								
Design Summary/Reports			2								
Create Schematic Symbol			-17%								
View Command Line Log File	71	Run									
Check Design Rules	Check Design Rules										
View HDL Functional Model		Rerun All									
📄 🛛 View HDL Instantiation Template	Contra 1	Neruti All									
🖮 🎾 User Constraints	Ϋ́	Stop									
Create Timing Constraints		View Text Report									
Floorplan IO - Pre-Synthesis	Force Proce	cess Up-to-Date									
Implement Design		Onen With	out Undating								
By Synthesize - XST	open min										
		Design Goals & Strategies									
sign Files Libraries Symbols	70	Process Properties									
ingen integ clorence bymbols											

Μετά την δημιουργία όλων των σχημάτων, στο design properties – Top level source type βάζω schematic. Δημιουργώ νέο source – schematic όπου εισάγω και συνδέω τα modules. Το κάνω top module και τελικά δημιουργώ το ucf. Αν θέλω το ucf να γίνει με γραφικό τρόπο πάω Processes – User constraints – Floorplan IO – Pre – Synthesis, όπου βλέπω το package του chip και επιλέγω pins για κάθε σήμα.



Πατάω implement design και αν όλα πάνε καλά μπορώ να γράψω την λογική στο πυρίτιο.

# Εξομοίωση και μέτρηση χρόνου

Για δοκιμή θα φτιάξω ένα διαιρέτη που θα δέχεται ρολόι 31250 Hz και θα βγάζει παλμό 1 Hz. Αρχικά βάζω `timescale lus / lns δηλαδή το τικ είναι 1μsec και έχω ακρίβεια στην εξομοίωση lns. Στο τέλος για το ρολόι γράφω : always begin #16 \_clkip = ~clkip: // Toggle clock every 16 ticks -> 16+16=32us -> 31250 Hz

```
#16 clkin = ~clkin; // Toggle clock every 16 ticks -> 16+16=32us -> 31250 Hz
end
```

Δηλαδή η ημιπερίοδος του παλμού θα διαρκεί 16 μsec.

									~							
1 5	5 🗆 🗖 🗗 🖄 🍂 📢	) 🗩 🗩 🗶 丿	•   🕻	e 🗄 🗧	🛨   🚦	1 m	<b>G b</b>	10s	चे ५व							
					8 524	368.000.00	0 ns							9 524 368	000 000	ns
~					0.021		0 00							5 52 1 566		20
P	News	Mala			19 500 00	0.000.000 -	20				000 000 0			. 10	500.000	000.000 pc
$\times$	Name	value	<u> </u>		8 200 00		يىبىنا		1	90000000	000 000 p	<u> </u>	1	· · · · · ľ	000000	000 000 ps
۲	Lie cikout1	0														
	lo cikout2	1														
œ	🗓 dig0	0														
Θ	🖓 dig1	0	100	00000	nrinnni	honon								nnnnnn	innnn	
14	🕼 clkin	1														
-	1 res	0								~~~~~		~~~~~~				
2																
5																
সি		-	;													
5,14																
					•						_					
			X	l: 9 52	4 368	000 000	) ps 🛛 X	2:85	24 368	000 000	) ps [ 🛛	X: 1 (	000 000	000 000	l ps	
							· ·									

Θέτω τον χρόνο εξομοίωσης 1sec και επιλέγω την κυματομορφή που θέλω να μετρήσω. Αυτή γίνεται έντονη. Αν κάνω αριστερό κλικ ο κέρσορας θα πάει αυτόματα στην πλησιέστερη αλλαγή κατάστασης. Μετά πάω λίγο δεξιά και πατάω κλικ και σέρνω μέχρι το σημείο που θέλω να μετρήσω. Τότε εμφανίζεται και δεύτερος κέρσορας και κάτω βλέπω την διαφορά ΔΧ, όπου στο παράδειγμά μας είναι 1 sec.

#### Post Fit Simulation

Εδώ βλέπω και τους χρόνους καθυστέρησης (propagation delay) για το συγκεκριμένο chip.

Σταύρος Φώτογλου