

Seminarski rad

# Alati za projektovanje

- upotreba alata za projektovanje na konkretnim primerima -



Goran Mišić, 12103

Niš, 24. 03. 2008.

# 1 Uvod

Savremena elektronska kola su dostigla stepen kompleksnosti koji prevazilazi mogućnosti njihovog efikasnog projektovanja bez upotrebe adekvatnog softvera i računara. Šta više, neka kola su proizvedena tako da se upotreba softvera za projektovanje podrazumeva pri njihovom "programiranju" odnosno formiranju veza u njima.

Programi koji se koriste za projektovanje nazivaju se "alatima". Često se ne radi o jednom programu već o skupu programa - programskom paketu. Takodje, pri projektovanju složenog elektronskog sistema, može se zahtevati i upotreba više različitih alata za isti projekat. Ovo ukazuje na značaj kompatibilnosti alata za projektovanje.

U ovom radu je na praktičnim primerima prikazana upotreba nekih od alata za projektovanje. Detaljno je opisan svaki postupak koji je preduzet u toku procesa projektovanja.

U poglavlju 2 je uradjena implementacija 32-bitnog generatora slučajnih brojeva u FPGA kolo. Najpre je opisan algoritam koji je korišćen za generisanje slučajnih brojeva. Algoritam je zatim preveden u VHDL opis kola koje obavlja funkciju generisanja slučajnih brojeva (RNG). Zatim je pomoću alata Active-HDL uradjena simulacija rada kola. Na kraju je, pomoću alata Xilinx ISE, uradjena implementacija VHDL opisa kola u FPGA čip.

U poglavlju 3 je ukratko opisan postupak generisanja lejauta ASIC kola iz VHDL opisa. Korišćeni alat je ModelSim, Leonardo Spectrum, DesignArchitect i ICStation.

U poglavlju 4 je uradjeno projektovanje lejauta ASIC kola stereo kodera. Najpre je dat funkcionalni opis stereo kodera. Zatim je kolo kodera opisano na nivou šeme i uradjena je simulacija rada kola. Korišćen je softverski paket PSpice. Zatim je dat skup MOSIS pravila koja su korišćena pri projektovanju lejauta. Na kraju je pomoću alata LASI7 izvršeno projektovanje lejauta čipa stereo kodera. Simulacija podkola čipa je radjena pomoću alata PSpice AD.

# 2 Implementacija 32-bitnog MWC RNG u FPGA

Algoritam MWC RNG (Multiply-With-Carry Random Number Generator) za generisanje 32-bitnih pseudoslučajnih brojeva je veoma jednostavan. Realizacija MWC generatora u digitalnom kolu zahteva veliki broj gejtova. Zato je u ovom slučaju odabrano FPGA kolo iz familije Spartan3E firme Xilinx. Implementacija je izvedena do trenutka kada je generisan fajl kojim se može programirati FPGA.

## 2.1 MWC algoritam

MWC algoritam se zasniva na sledećoj rekurziji [1]:

$$x_n = \left(a \cdot x_{n-1} + c_{n-1}\right) \mod b$$
$$c_n = \operatorname{int}\left\{\frac{a \cdot x_{n-1} + c_{n-1}}{b}\right\}$$

odnosno:

- novi x = ostatak od (a·x+c)/b
- novo c = celobrojni deo od (a·x+c)/b .

Za bilo koju inicijalnu vrednost x iz opsega 0 < x < (b-1) i inicijalnu vrednost za c iz opsega 0 < c < (a-1) rekurzija generiše niz x-ova: x<sub>0</sub>, x<sub>1</sub>, x<sub>2</sub>, ... koji je striktno periodičan. Dužina periode, a nama je potrebno da bude što veća, zavisi od izabranih vrednosti a i b. Ako je m = a·b-1 prost broj, tada perioda ima dužinu m.

Vrednost za b se može izabrati tako da bude stepen broja 2 što može značajno olakšati primenu MWC algoritma u nekom binarnom okruženju. Neka je b =  $2^{32}$ . Vrednosti za a, takve da je m = a·b-1 prost broj, možemo uzeti iz sledeće tabele [2]:

а												
1075433238	1447497129	1517746329	1554115554	1557985959								
1606218150	1631296680	1655692410	1675393560	1683268614								
1781943330	1791398085	1873196400	1893513180	1929682203								
1965537969	1967773755	2051013963	2083801278	2131995753								

Neka je a = 2083801278. Tada je dužina periode m =  $a \cdot b - 1 = 2083801278 \cdot 2^{32} = 8949858340373004287 !!! Uzmemo digitron (malo bolji digitron - TI-89) i proverimo da li je m prost broj:$ 

F1+ F2	v F3v F4v	FS F6	Ϋ́υ)
Tools A19el	braCa1cOtherPi	r9ml0C1ear	
∎ facto	r(2083801)	278 - 2 <sup>32</sup>	<sup>2</sup> - 1)
	89498583	403730	04287
factor	(208380127	78*2^32	(=1)
MMN	Rad Auto	FUNC	1/30

Kao što se vidi, faktorizacijom broja m dobijamo opet m, što znači da je m prost broj.

<sup>[1]</sup> http://www.stat.fsu.edu/pub/diehard/

<sup>[2]</sup> http://www.rkrupinski.ps.pl/langEn/random.php

Neka je w binarna reč dužine 64 bita takva da je prva polovina (bitovi 63 do 31) načinjena od bitova binarne reprezentacije broja c, a druga polovina (bitovi 31 do 0) načinjena od binarne reprezentacije broja x. Sada možemo umesto gornje rekurzije koristiti jednostavniju:

$$w_n = a * (w_{n-1} & (2^{32} - 1)) + (w_{n-1} >> 32)$$

pri čemu se u svakoj novoj iteraciji dobija novi 32-bitni pseudo-slučajni broj:

$$W_{n-1}$$
 &  $(2^{32}-1)$ 

Program napisan u C-u koji generiše prvih 1000 pseudo-slučajnih brojeva koristeći ovakav oblik rekurzije MWC algoritma je:

```
#include <stdio.h>
void main()
{    int i;
    long long w=1;
    for(i=0;i<1000;i++)
    {       w=2083801278*(w&4294967295)+(w>>32); //a=2083801278
            printf("%lld\n",w&4294967295);
    }
}
```

Može se koristiti Microsoft Visual C++. Generisani niz brojeva je:

```
2083801278
2983947524
144095773
4100253040
2723449940
3923414890
...
2674864222
```

Ako taj niz kopiramo u Microsoft Office Excel i nacrtamo grafik, izgledaće veoma "slučajno":



## 2.2 VHDL opis MWC RNG

Implementacija MWC RNG u FPGA kolo podrazumeva njegov opis u jeziku za opis hardvera (Hardware Description Language - HDL). Koristićemo jedan od mogućih jezika - VHDL (Very high speed Hardware Description Language). Korišćeni alat je Active-HDL firme Aldec.

## 2.2.1 Kratak opis alata Active-HDL

Active-HDL je integrisano okruženje za rad sa VHDL, Verilog, EDIF i kombinovanim VHDL-Verilog-EDIF projektima. Čini ga nekoliko alata za:

- unos projekta
- VHDL i Verilog kompajliranje
- simulaciju
- debagiranje
- grafički i tekstualni pregled rezultata simulacije
- pomoć u menadžmentu fajlova i biblioteka projekta:
  - Block Diagram Editor
  - o Code2Graphics converter
  - $\circ \quad \text{Console} \quad$
  - Design Browser
  - Design Flow Manager
  - $\circ \quad \text{HDL Editor} \quad$
  - o Language Assistant
  - o Library Manager
  - Memory View
  - o State Diagram Editor
  - o Waveform Viewer/Editor
  - Workspace/Design Explorer
  - o ...



## 2.2.2 Priprema radnog prostora

Nakon pokretanja programa Active-HDL otvara se sledeći prozor:

Getting	Started		?×
	O Open existing workspace		
			More
	1		
	Create new workspace		
□ Alw	ays open last workspace		
		Ū. OK	Cancel

Kliknemo na OK. Otvoriće se novi prozor u kome definišemo ime radnog prostora (ovde je to Goran) i kliknemo na OK:



Označimo Create an empty Design i kliknemo na Next >:



U novootvorenom prozoru samo kliknemo na Next >:

New Design Wizard		×
	Specify additional information about the new design.  Synthesis tool:  Implementation tool:  Default Family:  Default Family:  Default HDL Language VHDL V	
	< <u>B</u> ack <u>N</u> ext > Cancel	

Sada ukucamo ime projekta (u ovom slučaju je to MWC) i kliknemo na Next >:



#### Kliknemo na Finish:

New Design Wizard		×
	The new design will have the following specifications:          Design name: MWC         Image: mage:	
	< <u>B</u> ack Finish Can	cel

Ovim je priprema radnog prostora završena.

## 2.2.3 Unos VHDL koda

Nakon završene pripreme radnog prostora možemo uneti VHDL kod . Duplim klikom na Add New File unutar Design Browsera:



otvoriće se sledeći prozor:

Add New File		? ×
Empty Files   Wizards		
VHDL Source Block Diagram S Code	State Diagram System( Source Co	C Verilog Source ode Code
New Empty File:		
Name:		
		Add Existing File
	ОК	Cancel

Ukucamo ime novog fajla (ovde je to MWC), selektujemo VHDL Source Code i kliknemo na OK.

Otvoriće se prozor editora. U njemu pišemo nas VHDL kod, proveravamo njegovu sintaksnu ispravnost, editujemo ga ako je potrebno, i konačno, ako je kod ispravan, kompilujemo ga.

VHDL kod MWC generatora je veoma jednostavan:

```
library ieee;
use ieee.std logic 1164.all;
use ieee.std logic unsigned.all;
entity mwc is
port( clk : in std_logic;
    rst : in std logic;
    mwc out : out std logic vector(31 downto 0)
   );
end mwc;
architecture goran mwc of mwc is
signal w : std logic vector(63 downto 0);
begin
process (clk, rst)
begin
if rst='1' then
    elsif clk'event and clk='1' then
    w <= "0111110000110100010010010111110" * w(31 downto 0) +
       ("000000000000000000000000000000" & w(63 downto 32));
end if;
end process;
end goran mwc;
```

Napomena: onaj egzotični broj u VHDL kodu ( "0111110000110100010010010111110") je binarna reprezentacija našeg broja a = 2083801278.

Da bi smo proverili sintaksnu ispravnost koda i istovremeno izvršili njegovu kompilaciju kliknućemo na ikonu za kompilaciju:



U donjem delu prozora (Console) dobićemo izveštaj:

```
# File: c:\My_Designs\Goran\MWC\src\MWC.vhd
# Compile Entity "mwc"
# Compile Architecture "goran_mwc" of Entity "mwc"
# Top-Level unit detected
# Entity => mwc
# Compile success 0 Errors 0 Warnings Analysis time : 0.6 [s]
```

Ako postoje greške u kodu, ispravimo ih i ponovimo poslednju tačku.

## 2.2.4 Simulacija VHDL koda

Funkcionalnu simulaciju VHDL koda MWC RNG izvršićemo posmatranjem talasnih oblika signala koje koristimo u kodu.

Da bi smo dobili talasne oblike najpre kliknemo na ikonu New Waveform:



Zatim kliknemo na Structures:



Kliknemo na entitet mwc (goran\_mwc):



Pojaviće se signali koje koristimo u VHDL kodu. One signale čiji nas talasni oblici zanimaju prebacićemo u Waveform Editor tako što ih jednostavno "uhvatimo i prevučemo" u Waveform Editor:

🄷	ctive-	HDL 6.	3 (Gora	n ,MWI	C) - \	Vave	eforn	n Edi	itor	1 *						
Eile	<u>E</u> dit	Sea <u>r</u> ch	⊻iew	W <u>o</u> rksj	pace	Des	sign	Simu	Ilatio	n Y	<u>N</u> av	eform	n <u>T</u> o	ols	<u>W</u> inc	low <u>H</u> e
	1 - 1	ê 🔛	<b>张</b> 88	I	ß	Ş		ρ	C++ DBG	HEE DBG	ŝ	Ű	-	Ŷ	<b>5</b>	8
De	sign	Brov	vser	•	×		X 🖣	ð í		⊯2	S≥l		3	. <b></b>	• 🖟	Q
	mwc (g	oran_m	wc)		•	Na	me		V	/alue	9	Sti	- 1	- 2	0 · i	- 4 <u>0</u> -
F	🕂 mv	vc (gor	an_mwo	:)	-11		► clk		$\geq$							
	P std	l.stand	ard	~ .			► rst		$\geq$							
	lee Piee	e.sta_i e.std_i	ogic_11 ogic_ar	.64 ith 🖌		1	• mw	ic_ou	ıt 🔪							
	P iee	e.std_l	ogic_un	signe		<u>/</u>										
				//	Χ	L .										
⊫			$-\!\!/$			L .										
<u>  </u>					┙║	L .										
	ne E clk	$\prec$	$\square$	value Upavail	abl	L .										
5	rst	$\prec$		Unavail	abl	L										
12	• mwc	_out		Unavail	abl	L										
	<b>v</b> w			Unavail	abl	L										

Sada definišemo pobudu za ulazne signale, a to su clk i rst. Dvoklikom na Stimulator iza clk definišemo talasni oblik signala za clk, a dvoklikom na Stimulator iza rst - oblik signala za rst:

eform Edito	r 1 *							
sign <u>S</u> imulati	ion <u>W</u> a <sup>,</sup>	veform	<u>T</u> ools	<u>W</u> indo	w <u>H</u> e	lp		
📃 🔎 🔛		° 🛍	<b>7</b>		8	\$ 🗇	٢	&  ]
X 🖻 🕄	<b>K2</b> ©	¥ 💫	ч, Д	+ 🕅	•	ର୍ ବ୍	<del>C</del>	<b>°</b> W '
ame	Value	Sti	1 • 2	(0 · i	40	ı 60 -		80
► clk								
► rst								
• mwc_out								
	eform Editor sign Simulat Simulat E Clk C Clk rst mwc_out	eform Editor 1 * sign Simulation Wa Sign Sign Simulation Wa Sign Sign Sign Sign Sign Sign Sign Sign	eform Editor 1 * sign Simulation Waveform P Clk rst * mwc_out	eform Editor 1 * sign Simulation Waveform Tools	eform Editor 1 * sign Simulation Waveform Tools Windo	eform Editor 1 * ssign Simulation Waveform Tools Window He Sign Simulation Sim	eform Editor 1 * sign Simulation Waveform Tools Window Help ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓	eform Editor 1 * ssign Simulation Waveform Tools Window Help

Nakon dvoklika na Stimulator za clk otvoriće se prozor:

NI Stimulators		<u>?×</u>
Signals Hotkeys Predefined		
Signals: Name Type	Type: Clock	Forces a clock pulse of a specific frequency and duty cycle
Display paths	010 110 Valui 🗸	Apply Strength: Override Close

Kliknemo na ikonu Clock, podesimo parametre signala kao na slici i kliknemo na Apply pa na Close.

Analogno, nakon dvoklika na Stimulator za rst otvoriće se prozor:

Stimulators		<u>?×</u>
Signals Hotkeys Predefined		
Signals:	Туре:	Forces a waveform defined by a textual formula.
Name Type		value: 1 time offset: 0 ns
	Clock	value: 0 time offset: 10 ns
	fa	value: time offset:
	$\int J^{(1)}$	repeat above sequence every:
	Formula	Enter formula: Accept
	010	10 ns format.
Display paths	Valu	Apply Strength: Override 💌
		Close

Kliknemo na ikonu Formula, unesemo vrednosti kao na slici i kliknemo na Apply pa na Close.

Sada podesimo vreme trajanja simulacije (ovde je to 800 ns):

<u>W</u> ind	ow <u>H</u>	lelp																	
<b>S</b> .	8		Þ	۲	٩		•	1 E	80	0 ns		( 🔳	⊴	⊪	Ŧ	⊧Ç≣	φΞ		
- La	Q	Q	Q	<del>Q</del>	থ্য	<u>N</u>	AR	<b>6</b>	IL <del>L</del>	<b>±</b> 11	<b>#</b>	<b>1</b> 2	1	%	%	殇			
L . I	4,0		60		8 <u>0</u> -	1.0	100 -	1.1	120	1 - 1	40 i	- 160	I	· 180	1	200	I	2 <u>2</u> 0	T

Konačno, podesimo osobine signala mwc\_out, tako što desnim tasterom miša kliknemo na signal mwc\_out a zatim kliknemo na Properties. U novootvorenom prozoru (mwc\_out Properties) označimo Decimal i Unsigned i kliknemo na OK:



п	nwc_out Properties	? ×
	General Display	
	Signal name: mwc_out Hierarchy: Declaration: port ( mwc_out : out std_logic_vector(31 downto (	)))
	Values	-
	<u>R</u> adix: <u>N</u> otation:	
	O Binary O Unsigned	
	C Octal C Signed 2's Complement	
	O Signed 1's Complement	
	🔿 Hexadecimal 🛛 🔿 Signed Magnitude	
	Reverse <u>O</u> rder	
	Use <u>A</u> lias <u>E</u> dit Aliases	
	OK Cancel Appl	y

Sada možemo pokrenuti simulaciju klikom na ikonu Run For:

<u>W</u> indow	<u>H</u> elp																		
<b>4</b> 8	)	Þ	٩	ا ≪		•	( ⊵	во	) ns		( 1	₫	⊪	Ŧ	⊧ Ç≡	⊊≡			
<b>N</b> (	Q Q	0	<del>Q</del>	<del>ئ</del> س	M	AR	<b>₽</b>	Ut-	±₩	44	85	٨	%	%	<b>%</b>				
4	0 - 1	6,0 -	1.0	80.7	1.1	100 -	1.0	120	i - 1	40 i	· 160	I.	· 180	I	200	I.	2 <u>2</u> 0	1	1

Nakon završetka simulacije, kliknemo na ikonu End simulation, a zatim na ikonu Zoom To Fit:

Tools Window Help
🏆 🖓 🙀 🥴 😓 🖉 🔌 🕨 🕨 🕨 🗞 800 ns 🕀 📢 🔳 🖉 🕩 🖓 📮
ı 460 i 480 i 500 i 520 i 540 i 560 i 580 i 600 i 620 i 640 i 660 i
X2723449940 X3923414890 X3503709169

Simulacija pokazuje sledeće:

Name	٧	Sti	т. • 50 • т. • 100 • т. • 150 • т. • 200 • т. • 250 • т. • 300 • т. • 350 • т. • 400 • т. • 450 • т. •	5
<b>⊳</b> clk	0	Clo		=
► rst	0	For		
🛨 🔹 mwc_out	59		(1 X2083801278 X2983947524 X144095773 X4100253040 X27234499	40

Signal mwc\_out je std\_logic\_vector (31 downto 0), a njegove vrednosti prevedene u decimalne brojeve se vide na prethodnoj slici. Uporedjivanjem vrednosti koje smo dobili simulacijom VHDL koda, sa vrednostima koje je generisao program u C-u, konstatovaćemo da su identične, ©.

Oblik signala mwc\_out možemo prikazati i na drugaciji - analogni način. Podesimo vreme trajanja simulacije na 100 us. Zatim podesimo osobine signala mwc\_out, tako što desnim tasterom miša kliknemo na signal mwc\_out a zatim kliknemo na Properties. U novootvorenom prozoru kliknemo na zaglavlje Display i podesimo Height 128, Shape Analog from 0 to 4294967295 i kliknemo na OK.

mwc_out Properties	<u>? ×</u>
General Display	
Display	
Height: 128	ΛΛΛΛΛ
<u>C</u> olor: <u>B</u> old	7 7 7 7 7 7
<u>S</u> hape	
O Literal	
C Logic	to 4294967295
	10 4204001200
ОК С	Cancel <u>Apply</u>

Ponovo pokrećemo simulaciju klikom na ikonu Run For. Nakon završetka simulacije, kliknemo na ikonu End simulation, a zatim na ikonu Zoom To Fit.

Ovoga puta simulacija pokazuje analogni oblik signala mwc\_out:



Ovim je proverena funkcionalna ispravnost VHDL opisa MWC RNG i sada se može preći na implementaciju koda u FPGA kolo.

## 2.3 Implementacija VHDL opisa MWC RNG u FPGA

lako je VHDL opis MWC generatora pseudo-slučajnih brojeva veoma jednostavan, njegova implementacija u digitalno kolo zahteva veliki broj gejtova. Zato je u ovom slučaju odabrano FPGA kolo iz familije Spartan3E firme Xilinx, u koje je VHDL opis MWC generatora "stao" bez ikakvih problema. Ciljna tačka u ovom poglavlju je stići do trenutka kada se generiše fajl kojim se može programirati FPGA kolo. Korišćeni alat je Xilinx ISE.

## 2.3.1 Kratak opis alata Xilinx ISE

Ime alata potiče od Integrated Software Environment (ISE). Alat predstavlja potpuno okruženje za projektovanje programabilnih kola firme Xilinx, počev od unosa HDL opisa kola pa do verifikacije već programiranog i ugradjenog kola. Tok projektovanja je dat na sledećoj slici:



Project Navigator je program za upravljanje i procesiranje toka projektovanja kroz sledeće korake:

- Unošenje projekta
  - prvi korak u ISE toku projektovanja 0
  - kreiraju se izvorni fajlovi projekta po principu top-down korišćenjem jezika za opis hardvera 0 (VHDL, Verilog, ABEL) ili korišćeniem šematskog unosa
- Sinteza
  - pokreće se nakon unosa i eventualne simulacije projekta 0
  - iz izvornih fajlova projekta kreiraju fajlovi netlisti neophodnih za implementaciju 0
- Implementacija
  - pokreće se nakon sinteze 0
  - projekat opisan na RTL nivou konvertuje se u fajl za programiranje izabranog logičkog kola 0
  - proces implementacije zavisi od tipa izabranog logičkog kola (FPGA ili CPLD) 0
- Verifikacija
  - projekat se može više puta proveravati u toku procesa projektovanja 0
  - verifikuje se funkcionalnost i tajming projekta ili njegovog dela 0
- Konfiguracija kola
  - nakon konvertovanja RTL opisa u fail za programiranje, kolo se može programirati 0
  - generišu se konfiguracioni fajlovi i upišuju se pomoću kompjutera u izabrano Xilinx kolo 0

Pomoćni alati iz Xilinx ISE paketa su:

- Architecture Wizard
  - pomoć u kreiranju novog projekta
- Constraints Editor
  - o Constraints su instrukcije ograničenja plasirane pri unosu opisa projekta
  - specifikuju razmeštaj pinova, stanja memorije flip-flopova i lečeva, globalni i tajming grupa portova, imena i smerove signala
  - o pišu se u kotisničkim fajlovima ograničenja (User Constraints File UCF)
- CORE Generator
  - o generisanje fajlova korova koji štite intelektualnu svojinu (Intellectual Property IP)
  - FIFO i memorije, Reed-Solomon Decoder i Encoder, FIR filtri, brza Furijerova transformacija (FFT), standardni bus interfejsi (PCI i PCI-X), konekcije i mreže (Ethernet, SPI-4.2, RapidIO, CAN i PCI Express)
- Floorplanner
  - o planiranje detaljnog razmeštaja pinova
  - o pregled i editovanje ograničenja razmeštaja
  - o pronalaženje logičkih komponenti ili mreža na čipu
  - automatski razmeštaj portova
- FPGA Editor
  - o prikaz i konfigurisanje FPGA
  - zahteva Native Circuit Description (.ncd) fajl koji sadrži logiku projekta mapiranu u odgovarajuće blokove kao što su CLB i IOB
  - o razmeštaj i povezivanje kritičnih komponenti pre automatskog razmeštanja i povezivanja
  - završni razmeštaj i povezivanje ukoliko program za rutiranje nije sam kompletirao razmeštaj
  - o pomoć pri analizi stanja signala i debagiranju kola
  - o analiza tajminga
- iMPACT
  - o pomoć pri konfigurisanju i realizaciji procesa programiranja kola
- PACE
  - o pregled i editovanje ograničenja za lokacije I/O i globalne logike
  - o kreiranje ograničenja za površinu koju zauzima logika
  - o utvrdjivanje zahteva za resursima izabranog kola koje zahteva projekat
- StateCAD
  - o grafički način unošenja projekta pomoću dijagrama stanja
- Timing Analyzer
  - o analiza statičkog tajminga FPGA i CPLD projekata
  - o izveštaj o kašnjenju na putanjama i pomoć u analizi kritičnih putanja u kolu
  - o set up i hold provere tajminga
  - analiza clock signala kod sinhronih sistema
  - o kreiranje izveštaja o tajmingu
- XPower
  - o analiza potrošnje FPGA i CPLD kola
  - o analiza temperature na spojevima
  - o provera pravila projektovanja (DRC)

## 2.3.2 Priprema projekta



Nakon pokretanja Project Navigator-a kliknemo na File pa na New Project:

Time se pokreće New Project Wizard. U polju Project Name upišemo ime projekta (ovde je to MWC\_RNG) i kliknemo na Next >:

🚾 New Project Wizard - Create New Project		
Enter a Name and Location for the Project		
Project Name:	Project Location	
MWC_RNG	C:\/ilinx91i\MWC_RNG	
Select the Tupe of Tep Louid Service for the Project		
Top-Level Source Tupe:		
HDL		<b>_</b>
More Info	< <u>B</u> ack <u>N</u> ext >	Cancel

Sada treba definisati parametre izabranog FPGA kola kao na slici pa kliknuti na Next >:

Property Name	/	Value			
Product Category		All	<u> </u>		
Family		Spartan3E	<u> </u>		
Device		XC3S500E	<u> </u>		
Package		FG320	<b>•</b>		
Speed		-4	<u> </u>		
Top-Level Source Type		HDL			
Synthesis Tool		XST (VHDL/Verilog)	<u> </u>		
Simulator		ISE Simulator (VHDL/Verilog)	<b>_</b>		
Preferred Language		VHDL	<b>•</b>		
Enable Enhanced Desig	n Summary	<b>v</b>			
Enable Message Filtering	,				
Display Incremental Messages					

#### Samo kliknuti na Next >:

🚾 New Project Wizard - Create New Source							
Create a New Source							
Source File	Туре	New <u>S</u> ource					
Creating a new source to add to the project is option. Additional sources can be created and added	al. Only one new source can be created with I to the project by using the "Project->New So	the New Project Wizard. urce'' command.					
Existing sources	Existing sources can be added on the next page.						
More Info	< <u>B</u> ack <u>N</u> ext>	Cancel					

Kliknuti na Next >:

Sew F	roject Wizard - Add Existing Sources			
1	Source File	Сор	by to Project	Add <u>S</u> ource
Adding	wisting sources is optional. Additional sources o	an he added after the project is	created using th	e "Project->Add
Source"	or "Project->Add Copy of Source" commands.	A Back	Nevt \	Cancel

Kliknuti na Finish:

🚾 New Project Wizard - Project Summary	
Project Navigator will create a new project with the following specifications:	
Project: Project Name: MWC_RNG Project Path: C:\Xilinx91i\MWC_RNG Top Level Source Type: HDL	-
Device: Device Family: Spartan3E Device: xc3s500e Package: fg320 Speed: -4	
Synthesis Tool: XST (VHDL/Verilog) Simulator: ISE Simulator (VHDL/Verilog) Preferred Language: VHDL	
Enhanded Design Summary: enabled Message Filtering: disabled	▼ Cancel

Kliknemo dva puta na Create New Source:

		×	
Sources for: Sy	nthesis/Imp	olementat 💌	
😇 MWC_R	NG		
<sup>i</sup> 🛄 xc3s500e	e-4fg320		
📲 Sources	Snapshot:	🜔 Libraries	
			-
		<u>^</u>	
Processes for: x	c3s500e-4	fg320	
🗖 🛛 Add E	xisting Sou	urce	
- Creat	e New Sou	irce 💙	
🗄 🥍 Desig	n Utilities		

Upišemo ime fajla (ovde je to mwc), zatim selektujemo VHDL Module i kliknemo na Next >:

🚾 New Source Wizard - Select Source Type	
<ul> <li>IP (Coregen &amp; Architecture Wizard)</li> <li>Schematic</li> <li>State Diagram</li> <li>Test Bench WaveForm</li> <li>User Document</li> <li>Verilog Module</li> <li>Verilog Test Fixture</li> <li>VHDL Module</li> <li>VHDL Library</li> <li>VHDL Package</li> <li>VHDL Test Bench</li> </ul>	File name:   mwc   Logation:   C:\Xilinx91i\MWC_RNG
More Info	< Back Next > Cancel

Samo kliknemo na Next >:

15	New Source Wizard	- Define Module							×	
	Entity Name mwc									
	Architecture Name Behavioral									
	Port Name	Direction		Bus	MSB		LSB		•	
		in	-							
		lin	-							
		lin	-							
		lin	-							
		in	-							
		in	-							
		in	-							
		lin	-							
		in	-							
		in	-						<b>-</b>	
_	, 									
	<u>M</u> ore Info				< <u>B</u> ack		<u>N</u> ext >	Cancel		

#### Kliknemo na Finish:

🚾 New Source Wizard - Summary							
Project Navigator will create a new skeleton source with the following specifications:							
Add to Project: Yes Source Directory: C:\Xilinx91i\MWC_RNG Source Type: VHDL Module Source Name: mwc.vhd Entity Name: mwc Architecture Name: Behavioral Port Definitions:							
<	ack <u>F</u> inish Cancel						

Ovim je priprema projekta završena.

## 2.3.3 Unos VHDL koda

Nakon završene pripreme projekta potrebno je da unesemo VHDL kod . Najpre kliknemo na zaglavlje editora, mwc.vhd:



Sada u editoru selektujemo i izbrišemo (Delete na tastaturi) kod koji je program automatski kreirao. Zatim unesemo naš kod za MWC generator. Uneti VHDL kod treba sačuvati. Kliknemo na File pa na Save:



## 2.3.4 Unos ograničenja

Unećemo samo ograničenja koja se tiču fizičkog razmeštaja pinova izabranog FPGA kola.

Najpre, kliknemo na + ispred User Constraints, zatim kliknemo dva puta na Assign Package Pins:



#### Kliknemo na Yes:

🚾 Proje	ct Navigator
?	This process requires that an Implementation Constraint File (UCF) be added to the project and associated with the selected design module. Would you like Project Navigator to automatically create a UCF and add it to the project at this time? If you select "No" you will need to create or add an existing UCF to the project before running this process.

Pokrenuće se jedan od pomoćnih alata - Xilinx PACE. U donjem delu prozora treba kliknuti na zaglavlje Package View:





#### Pojaviće se XC3S500E FPGA sa svih svojih 320 (!) pinova:

Da bi smo se malo snašli u tom polju pinova kliknemo na View, pa na Toolbars i zatim na Legend:



Package	Pin Legend 🛛 🛛 🔟
Symbol	Pin Type
0	User IO
0	Input Only
	User Prohibit
	GND
	VCCINT
	VCCAUX
	vcco
	CONFIG
	JTAG
0	GCLK / GCK
0	Left Hand Clock
0	Right Hand Clock
0	IRDY / TRDY / LHCLK / RHCLK
$\bigcirc$	Power Management
$\Box$	Not Connected
*	Bank0
	Bank1
	Bank2
	Bank3

Sada, pošto nam je legenda pomogla oko snalaženja u polju pinova, ☺, možemo izvršiti njihov fizički razmeštaj u smislu dodele signala pojedinim pinovima.

Kliknemo na polje Loc za neki od signala i upišemo lokaciju pina na kojoj će biti taj signal. Svaka dodela signala nekom pinu odmah bude označena plavom bojom u prozoru koji pokazuje njihov razmeštaj:

間	Design Ohjer	t List - T/O Pin	s 🗖	Ini xi								Top۱	/iew								
		LO Disation				12	3	4 !	56	67	8	9.	0 1	1 1	2 13	14	15	5 16	17	18	
L		I/U Direction	DO	DANK	A		$\mathbf{O}$	$\Box$		$\mathbb{D}$				5		C	)(	$) \cap$			А
	cin mula cut/05	niput Outout	00	DANK			$\sim$		ž.				-			17	82	3F			
	mwc_out<1>	Output		DANK									4				4		( <u></u>	$\smile$	D
	mwc_out	Output	117	DANK		QC	<u>IO</u>				ЭO				<u> </u>		X			$\bigcirc$	С
	mwc_out<2>	Output	110	DANK	D		10	A			$\cap$				$\cap C$	$\mathcal{T}$	T			$\bigcirc$	D
	mwc_out<4>	Output	D10	BANK			1×							-	×P		17		Ħ	$\widetilde{\frown}$	F
	mwc_out<5>	Output	<u>ц</u> р	BANK				$\Delta$									14	ЦĊ		$\smile$	E
	mwc_out/6>	Output	H5	BANK	F	OC	2	$ \bigcirc $	$\supset$				DIC	Х			х	2	$\bigcirc$	$\bigcirc$	F
	mwc_out<7>	Output	НЛ	BANK	G	$\bigcirc$							$\square$			T	10			$\bigcirc$	G
	mwc_out/8>	Output	H15	BANK	L						. =							10		õ	ц
	mwc_out<9>	Output	Н6	BANK			4 🔤					<b>;</b> ;					4			$\leq$	
	mwc_out<10	Output	48	BANK	J	OC		$\cup$	$\mathcal{I}$	$\mathcal{I}($	2				ЭC	)C	ЛÇ	JÇ			J
	mwc_out<11>	Output	F9	BANK	к		$) \cap$	$\bigcap$	ົງເ	$\mathcal{T}$						רו			$\bigcirc$	$\bigcirc$	К
	mwc_out<12>	Output	F10	BANK		$\overline{a}$	iň	Ň	Ň						Ē	٦ř	V.		iñ	Õ	ī
	mwc_out<13>	Output	A11	BANK	L			X	석	4.			~			<b>4</b> ≻	⇔	�⊨		$\mathbf{H}$	
	mwc_out<14>	Output	B10	BANK	м	$\bigcirc$	$\Box$	$\Box$				$\cup$				JL.	1			$\cup$	M
	mwc_out<15>	Output	J12	BANK	N	OC								$\mathbf{D}_{\mathbf{Z}}$	$\Delta$		MΖ		$\mathbf{O}$	$\bigcirc$	N
	mwc_out<16>	Output	D9	BANK	Р		ΥA.		1			O	n		ΛT	1	i?				Р
	mwc_out<17>	Output	G9	BANK					-			X	X	4			✐	1-	K	$\bowtie$	
	mwc_out<18>	Output	H14	BANK	ĸ	$\square$					20	$\square$	<u> </u>				4		$\sim$		н
	mwc_out<19>	Output	H17	BANK	Т	OC	JO	$\bigcirc$			$\mathcal{O}$		$\mathcal{I}($	X			X	JC.		$\bigcirc$	Т
	mwc_out<20>	Output	D11	BANK	υ	$\bigcirc$							$\gamma$			)(	X	70			U
	mwc_out<21>	Output	B11	BANK	V .		ŇŎ			Ā 17		$\widetilde{\frown}$		1		17	1	10			
	mwc_out<22>	Output	J13	BANK	v		$\mathbb{N}$	$\nabla_{\mu}$	ΔjZ	_j2	$\Delta \mathcal{U}$	P		4		<u> </u>	4	$\mathcal{A}$			v
	mwc_out<23>	Output	C9	BANK		12	3	4 !	56	67	8	9.	0 1	1 1	2 13	14	15	5 16	17	18	
	mwc_out<24>	Output	G15	BANK																	
	mwc_out<25>	Output	E9	BANK																	
	mwc_out<26>	Output	C11	BANK																	
	mwc_out<27>	Output	J15	BANK																	
	mwc_out<28>	Output	J14	BANK																	
	mwc_out<29>	Output	K15	BANK																	
	mwc_out<30>	Output	A10	BANK																	
	mwc_out<31>	Output	H16	BANK																	
	rst	Input	F10	BANK																	

۲

Nakon izvršenog dodeljivanja signala pinovima, potrebno je to i sačuvati. Kliknemo na File, zatim na Save:



Pojaviće se Dialog Box u kome samo treba kliknuti na OK:

Bı	us Delimiter	x
	Select IO Bus Delimiter	
	◯ X <u>S</u> T Optional {}	
	◯ Synplify <u>V</u> erilog Default: [ ]	
	© Synplify V <u>H</u> DL / Exemplar Default: ( )	
	Don't show this dialog again (can be set through preferences dialog)	
	OK Cancel <u>H</u> elp	

Dodeljivanje signala pinovima koje smo upravo uradili predstavlja ograničenje za softver, kog se on mora pridržavati u daljem toku implementacije. Fajl koji sadrži ograničenja je User Constraints File - UCF. Prethodnom naredbom (Save), softver je upravo generisao UCF fajl i dodao ga našem projektu. Vratimo se nazad na naš projekat, kliknemo na + ispred mwc - goran\_mwc i proverimo: mwc.ucf fajl je zaista dodat našem projektu:

🚾 Xilinx - ISE - C:\Xilinx91i\MWC_RN	IG\MWC_R	NG.ise - [mwc.vhd]						
🔽 Eile Edit View Project Source Process Window Help								
] 🗋 🤌 🗟 🕼 🖉 🗳 🗍 🔏 🖨	🗙 🖾	@ 🛛 🖉 🖉 🖉						
I 🖷 🗏 🗏 🖻 🖆 🔺 🛪 🦄 🖑 🕱 🛛 🏵								
×	1	library ieee;						
Sources for: Synthesis/Implementat 🔫	2	use ieee.std_lo						
	3	use ieee.std_lo						
	4	entity mwc is						
⊡- 🛄 xc3s500e-4fa320	5	port( clk : in						
🖃 🔚 🏭 mwc - goran_mwc (mwc.	6	rst : in std						
🖳 🚾 mwc.ucf (mwc.ucf)	7	mwc out : ou						
<	8	);						
It Sources	9	end mwc;						
	10	architecture go						

## 2.3.5 Implementacija

Xilinx ISE je veoma automatizovani softwer. Nakon unošenja VHDL opisa i ograničenja, potrebno je još par puta kliknuti mišem da bi se stiglo do krajnje tačke imlementacije, koju smo (u ovom projektu) definisali kao generisanje fajla za programiranje FPGA kola. Sve medjufaze u toku projektovanja softver može obaviti samostalno i po automatizmu. Nećemo postavljati nikakva ograničenja vezana za tajming. Funkcionalnu simulaciju koda smo već obavili u programu Active-HDL i sada to nećemo ponavljati.

Najpre treba izvršiti nekoliko podešavanja. Kliknemo na zaglavlje Libraries a zatim na + ispred biblioteke work:



Desnim tasterom kliknemo na fajl mwc.vhd a zatim kliknemo na Properties ...:



U prozoru Source Properties kliknemo na Synthesis/Implementation Only, a zatim na OK:

15	Sou	rce Properties	×	۱
	Use th	iis dialog to change the Design Unit	source views that the design unit appears in. Association	
		ዤ mwc goran_mwc	Simulation Only	
			None	
			Synthesis/Implementation Only	
			Simulation Univ	
	1		Synthesis/Imp + Simulation	
		C C	OK Cancel Help	

Sada se vratimo nazad klikom na zaglavlje Sources, zatim kliknemo dva puta na Generate Programming File i odemo da gledamo našu omiljenu seriju o Džordžu, dok softver sam dovršava projekat, ③.



## 2.3.6 Izveštaji implementacije

Nakon komande Generate Programming File, softver je od VHDL opisa MWC generatora prošao sam kroz celokupan tok projektovanja sve do generisanja fajla kojim možemo programirati izabrano FPGA kolo. Usput je, o svakom od preduzetih koraka generisao

iscrpan izveštaj. Rezime izveštaja implementacije dobijamo klikom na zaglavlje Design Summary:



MWC_RNG Project Status						
Project File:	MWC_RNG.ise	Current State:	Programming File Generated			
Module Name:	mwc	• Errors:	No Errors			
Target Device:	xc3s500e-4fg320	<ul> <li>Warnings:</li> </ul>	<u>1 Warning</u>			
Product Version:	ISE 9.1.01i	<ul> <li>Updated:</li> </ul>	Sat 9. Feb 01:40:51 2008			

#### MWC\_RNG Partition Summary

No partition information was found.

Device Utilization Summary						
Logic Utilization	Used	Available	Utilization	Note(s)		
Number of Slice Flip Flops	64	9,312	1%			
Number of 4 input LUTs	153	9,312	1%			
Logic Distribution						
Number of occupied Slices	101	4,656	2%			
Number of Slices containing only related logic	101	101	100%			
Number of Slices containing unrelated logic	0	101	0%			
Total Number of 4 input LUTs	201	9,312	2%			
Number used as logic	153					
Number used as a route-thru	48					
Number of bonded <u>IOBs</u>	34	232	14%			
Number of GCLKs	1	24	4%			
Number of MULT18×18SIOs	4	20	20%			
Total equivalent gate count for design	2,426					
Additional JTAG gate count for IOBs	1,632					

Performance Summary							
Final Timing Score:	0	Pinout Data:	Pinout Report				
Routing Results:	All Signals Completely Routed	Clock Data:	Clock Report				
Timing Constraints:	All Constraints Met						

Detailed Reports							
Report Name	Status	Generated	Errors	Warnings	Infos		
Synthesis Report	Current	Sat 9. Feb 01:39:53 2008	0	<u>1 Warning</u>	0		
Translation Report	Current	Sat 9. Feb 01:40:01 2008	0	0	0		
Map Report	Current	Sat 9. Feb 01:40:12 2008	0	0	<u>3 Infos</u>		
Place and Route Report	Current	Sat 9. Feb 01:40:34 2008	0	0	<u>2 Infos</u>		
Static Timing Report	Current	Sat 9. Feb 01:40:40 2008	0	0	<u>3 Infos</u>		
Bitgen Report	Current	Sat 9. Feb 01:40:51 2008	0	0	0		

Odmah se može videti da je procenat korišćenja resursa (Utilization) FPGA kola XC3S500E izrazito mali, što znači da je 32-bitni MWC RNG bez problema stao u FPGA kolo. Ovo se sigurno ne bi dogodilo da je na početku projektovanja bilo izabrano bilo koje od Xilinx CPLD kola. Naime, autor teksta je probao da implementira 16-to bitnu verziju MWC generatora u CPLD kola serije XC9500 i nije mu uspelo. Softver je redovno prijavljivao da izabrana CPLD kola nemaju dovoljno resursa za realizaciju VHDL opisa. Ono što je uzimalo najviše resursa je množač.

Ostalim izveštajima implementacije jednostavno se pristupa klikom na ime izveštaja u Design Summary. Iz tih izveštaja se na "početnu stranu" vraćamo klikom na ikonu ().

<u>Synthesys Report</u> je tekstualni fajl generisan u toku same implementacije. Podeljen je na više sekcija. Jedna od njih se odnosi na to kako je softver video naš VHDL kod MWC generatora. Naime, softver je u kodu "prepoznao":

# Multipliers	:	1
32x32-bit multiplier	:	1
<pre># Adders/Subtractors</pre>	:	1
64-bit adder	:	1
# Registers	:	64
Flip-Flops	:	64

A evo sa čime je VHDL kod realizovao:

Cell Usage		
# IOs	:	34
# BELS	:	533
# GND	:	1
# LUT1	:	47
# LUT2	:	151
# LUT3	:	1
# LUT4	:	1
# MUXCY	:	166
# VCC	:	1
# XORCY	:	165
<pre># FlipFlops/Latches</pre>	:	64
# FDC	:	63
# FDP	:	1
<pre># Clock Buffers</pre>	:	1
# BUFGP	:	1
# IO Buffers	:	33
# IBUF	:	1
# OBUF	:	32
# MULTs	:	4
# MULT18X18SIO	:	4

#### Korišćenje resursa:

Number of Slices:	103	out of	4656	2%
Number of Slice Flip Flops:	64	out of	9312	0%
Number of 4 input LUTs:	200	out of	9312	2%
Number of bonded IOBs:	34	out of	232	14%
Number of MULT18X18SIOs:	4	out of	20	20%
Number of GCLKs:	1	out of	24	4%

#### Tajming:

Minimum	period:	16.376ns (14.225ns logic, 2.151ns route) (86.9% logic, 13.1% route)	
Maximum	Frequenc	cy: 61.065MHz	
Maximum	output a	required time after clock: 5.597ns	3
Maximum	combinat	cional path delay: 7.071ns	

<u>Pinout Report</u> je tabelarni prikaz pinova, njihovih imena, signala koji su im dodeljeni, smera (O/I) i niza drugih parametara.

Static Timing Report je izveštaj koji sadrži statičke tajminge kola:

- Clock to Pad
- Clock to Setup on destination
- Pad to Pad

<u>Clock Report</u> sadrži izveštaj o tome koji bafer je upotrebljen za clock signal, koliki je fanout, kosina (skew) i maksimalno kašnjenje.

Izveštaji implementacije su veoma opširni. Njih projektant može koristiti u proceni performansi i funkcionalnosti projektovanog sistema. Analizom izveštaja projektant je u mogućnosti da uoči kritična mesta, eventualne nedostatke ili greške, ili da koriguje svoj projekat u smislu poboljšanja performansi.

## 2.3.7 RTL i tehnološka šema

Interesantno je pogledati kako je softver prepoznao VHDL kod MWC RNG i konvertovao ga u šeme na RTL i tehnološkom nivou.

Za RTL šemu treba kliknuti na + ispred Synthesize i dva puta kliknuti na View RTL Schematic:



Dobiće se top-level dijagram MWC generatora. Vide se ulazi clk i rst i 32-bitni izlazni bus mwc\_out:



Da bi se u dijagramu spustili jedan nivo ispod, treba kliknuti na ikonu Push:



Vide se 32x32-bitni množač neoznačenih brojeva, 32-bitni potpuni sabirač neoznačenih brojeva i 64-bitni shift registar:



Za tehnološku šemu treba kliknuti na View Technology Schematic:



Dobija se šema koju je softver generisao za konkretno FPGA kolo i njegove resurse. Ovde je dat samo deo jedne od 11 stranica koliko sadrži šema. Vide se upotrebljeni LUT blokovi, množači, multiplekseri, gejtovi.



## 2.3.8 Pregled rasporeda i veza

Za grafički prikaz razmeštanja i povezivanja komponenti i blokova u FPGA kolu i eventualno editovanje, ukoliko softver nije mogao sam to obaviti, koristi se FPGA Editor.

Da bi pokrenuli FPGA Editor, treba kliknuti na + ispred Implement Design, zatim na + ispred Place & Route i onda dva puta kliknuti na View/Edit Routed Design (FPGA Editor):



Ovim je pokrenut FPGA Editor i možemo videti kako je softver planirao razmeštaj komponenti i veza u izabranom FPGA kolu (da vidimo veze porebno je kliknuti na ikonu Routes 🔀 u meniju):

 ì

sector sect	
	u <mark>s</mark> tatatatatatatatatata
e e e e e e e e e e e e e e e e e e e	
*********************************	x 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
a a a a a a a a a a a a a a a a a a a	
	9
e entrerereretorerere encourte	
	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
e severences a severe se	- <u> </u>
ga analana analan analan analan analan a	
	2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2010年1月1日の日本では、1月1日には1月1日には、1月1日、1月1日、1月1日、1月1日、1月1日、1月1日、1月1日、1月1	1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
	1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
	i > >



Programiranje FPGA se obavlja pomoću softverskog alata iMPACT iz paketa Xilinx ISE. Potreban je fajl za programiranje, PC, JTAG kabal i sistem sa ugradjenim FPGA kolom, i ceo proces traje par sekundi. Medjutim, pošto je naša ciljna tačka projekta već ostvarena tokom implementacije (generisanjem fajla za programiranje izabranog FPGA kola), ovde se projektovanje 32-bitnog MWC RNG završava.