

POKAZNA VEŽBA 1

Osnovi projektovanja digitalnih sistema na nivou logičkih kola

Potrebno predznanje

- Osnovno poznavanje digitalne elektronike
- Bulova (Boolean) algebra

Šta će biti naučeno tokom izrade vežbe?

Nakon urađene vežbe, bićete u mogućnosti da:

- razumete razna digitalna logička kola i njihovu funkciju
- projektujete sistem koji računa proizvoljnu Boolean funkciju pomoću logičkih kola
- opišete sistem sastavljen od digitalnih logičkih kola pomoću logičke šeme
- koristite VHDL test bench za proveru rada sistema sastavljenih od digitalnih logičkih kola
- koristite Xilinx ISE za opis i simulaciju digitalnih sistema.

Sa znanjem dobijenim u ovoj vežbi, bićete spremni da projektujete složenije sisteme sastavljene od digitalnih logičkih kola, tzv. kombinacione mreže, o kojima će biti više reči u narednoj vežbi. Znanje dobijeno u ovoj vežbi je važno za pravljenje mosta između digitalnog matematičkog sveta Bulove algebre i fizičkog sveta digitalne elektronike. Osnovno znanje dobijeno u ovoj vežbi omogućiće vam da razumete složene digitalne sisteme koje ćete projektovati tokom predmeta.

Apstrakt i motivacija

U današnje vreme, mnogi uređaji koji nas okružuju su digitalni. Ovi uređaji čitav svoj rad zasnivaju na operacijama unutar skupa od samo dve vrednosti (nule i jedinice), a operacije koje nad njima vrše su operacije Bulove algebre nad ovim skupom vrednosti. Računanje tokom rada aviona koje vodi računa da putnici sigurno slete na odredište, Facebook chata na vašem mobilnom telefonu ili puta na Mesec se vrši isključivo pomoću operacija Bulove algebre nad skupom od dve vrednosti. Nije teško zaključiti da je poznavanje Bulove algebre i njenih operacija ključno znanje prilikom razumevanja rada uređaja modernih tehnologija.

Ova vežba će vas naučiti osnovama operacija Bulove algebre i projektovanja sistema koji računaju vrednosti Bulovih funkcija, koji su osnovna gradivna komponenta digitalnih sistema. Vežba će vas naučiti i kako da opišete te sisteme koristeći logičke šeme. Tokom projektovanja vašeg prvog digitalnog sistema, koristićete Xilinx ISE alat koji omogućava ne samo da opišete digitalni sistem nego i da ga simulirate, tj. proverite njegov rad, i implementirate na određenoj fizičkoj platformi. Znanje koje dobijete u ovoj vežbi je prvi korak ka konačnom cilju ovog predmeta – projektovanju složenih digitalnih sistema za računanje, tj. procesora.

TEORIJSKE OSNOVE

1. Digitalna logička kola

Digitalna logička kola su elektronska kola koja imaju mogućnost računanja vrednosti logičkih funkcija Bulove algebre. Bulova algebra je definisana nad skupom od 2 vrednosti:

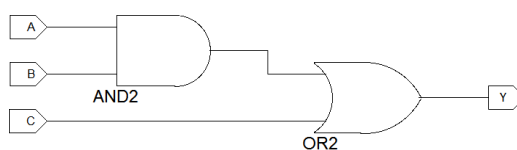
- **Logička nula (0, FALSE)** – predstavljena niskom vrednosti potencijala na žici,
- **Logička jedinica (1, TRUE)** – predstavljena visokom vrednosti potencijala na žici.

Neke od logičkih operacija Bulove algebre su:

- **Negacija (NOT)** – unarna operacija predstavljena znakom komplementa,
- **Konjunkcija (AND)** – binarna operacija predstavljena znakom množenja,
- **Disjunkcija (OR)** – binarna operacija predstavljena znakom sabiranja,
- **Ekskluzivna disjunkcija (XOR)** – binarna operacija predstavljena znakom sabiranja u krugu,
- razne kombinacije ranije navedenih operacija.

Tabela 1.1 na sledećoj strani daje prikaz osnovnih logičkih kola koja se koriste u digitalnoj elektronici, kao i istinitosne tablice koje prikazuju šta će biti izlaz svakog od kola, za svaku kombinaciju ulaza. Svaki od ovih kola se realizuje pomoću elektronskih komponenti – tranzistora. Fizička realizacija logičkih kola je izvan opsega ovog predmeta i neće biti razmatrana. Prilikom projektovanja digitalnih sistema logička kola se koriste kao osnovne komponente, odn. komponente na najnižem nivou hijerarhije.

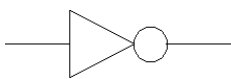
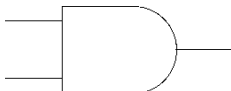
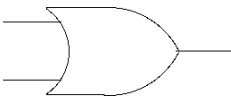
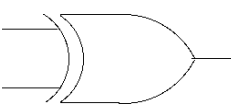
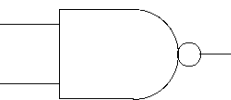
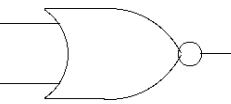
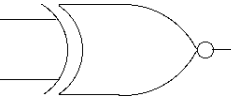
Složene funkcije Bulove algebre se realizuju kombinacijom logičkih kola iz tabele 1-1. Minimalni skup logičkih kola predstavlja skup pomoću kojeg se može realizovati bilo koja složena funkcija Bulove algebre. Može se dokazati da kolo NAND predstavlja minimalni skup, odn. bilo koja funkcija Bulove algebre može da se realizuje koristeći samo NAND kolo. Slično tako, kolo NOR predstavlja minimalni skup. Zanimljivo je da mnogo korišćenije funkcije u matematičkoj logici - AND, OR i NOT samostalno ne predstavljaju minimalni skup, već minimalni skup sadrži sve tri navedene funkcije. Kao primer realizacije složenih funkcija, posmatrajmo funkciju $Y = AB + C$. Slika 1-1 pokazuje kako se navedena složena funkcija može realizovati pomoću logičkih kola.



Slika 1-1. Primer realizacije složene Bulove funkcije pomoću logičkih kola

Iako izgleda vrlo skromno, sa dve vrednosti i nekoliko operacija, prostor Bulove algebre je matematički izuzetno moćan i dovoljan da se izvrše sva matematička računanja - od jednostavnih, kao što su sabiranje, množenje, do veoma složenih kao što su sistemi odlučivanja u složenim uređajima ili numeričko rešavanje parcijalnih diferencijalnih jednačina. Svi digitalni uređaji su zasnovani na računanju unutar prostora Bulove algebre. Računanje prilikom kontrolisanja leta aviona, rada vašeg mobilnog telefona i kućnog računara, automobila, nečeg složenog kao što je superračunar koji kontroliše let svemirske letelice ili nečeg jednostavnog kao što je budilnik koji vodi računa da ne zakasnite na vežbe iz LPRS1 – sve se to vrši računanjem u prostoru Bulove algebre i njene dve vrednosti i minimalno jedna operacija su dovoljne za to.

Tabela 1-1. Digitalna logička kola

Naziv	Oznaka	Funkcija	Istinitosna tablica															
NOT		$Y = \bar{X}$	<table><tr><th>X</th><th>Y</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	X	Y	0	1	1	0									
X	Y																	
0	1																	
1	0																	
AND		$Y = X_1 X_2$	<table><tr><th>X₁</th><th>X₂</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	X ₁	X ₂	Y	0	0	0	0	1	0	1	0	0	1	1	1
X ₁	X ₂	Y																
0	0	0																
0	1	0																
1	0	0																
1	1	1																
OR		$Y = X_1 + X_2$	<table><tr><th>X₁</th><th>X₂</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	X ₁	X ₂	Y	0	0	0	0	1	1	1	0	1	1	1	1
X ₁	X ₂	Y																
0	0	0																
0	1	1																
1	0	1																
1	1	1																
XOR		$Y = X_1 \oplus X_2$	<table><tr><th>X₁</th><th>X₂</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	X ₁	X ₂	Y	0	0	0	0	1	1	1	0	1	1	1	0
X ₁	X ₂	Y																
0	0	0																
0	1	1																
1	0	1																
1	1	0																
NAND		$Y = \overline{X_1 X_2}$	<table><tr><th>X₁</th><th>X₂</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	X ₁	X ₂	Y	0	0	1	0	1	1	1	0	1	1	1	0
X ₁	X ₂	Y																
0	0	1																
0	1	1																
1	0	1																
1	1	0																
NOR		$Y = \overline{X_1 + X_2}$	<table><tr><th>X₁</th><th>X₂</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	X ₁	X ₂	Y	0	0	1	0	1	0	1	0	0	1	1	0
X ₁	X ₂	Y																
0	0	1																
0	1	0																
1	0	0																
1	1	0																
XNOR		$Y = \overline{X_1 \oplus X_2}$	<table><tr><th>X₁</th><th>X₂</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	X ₁	X ₂	Y	0	0	1	0	1	0	1	0	0	1	1	1
X ₁	X ₂	Y																
0	0	1																
0	1	0																
1	0	0																
1	1	1																

Digitalna kola zasnovana na logičkim kolima iz tabele 1-1 se nazivaju **kombinacione mreže**. Svaka kombinaciona mreža zahteva određeno vreme računanja koje mora da protekne da bi se na izlazu pojavio validan rezultat. Ovo vreme postoji zbog parazitne kapacitivnosti u elektronskim komponentima pomoću kojih se realizuju logička kola i konačnog vremena koje mora da protekne da bi signal promenio vrednost od visokog ka niskom potencijalu i obrnuto. Zbog toga se za svako logičko kolo definiše **kašnjenje** logičkog kola, a kašnjenje složenog logičkog kola se računa kao kašnjenje na najdužoj putanji od nekog ulaza do nekog izlaza. Za primer sa slike 1-1, kašnjenje datog kola jednako je zbiru kašnjenja AND i OR kola, pošto je najduža putanja upravo od ulaza A ili B do izlaza Y, koja prolazi kroz oba kola (da bi OR kolo moglo da računa izlaz, prvo mora AND kolo da izračuna vrednost međurezultata). Kašnjenje kola direktno određuje maksimalnu frekvenciju na kojoj digitalni sistem može da funkcioniše, o čemu ćemo više pričati kasnije.

Iako su dovoljna za sva računanja, logička kola iz tabele 1-1 nisu dovoljna da bi se realizovao kompletan digitalni sistem. Da bi računanja imala smisla, rezultat tih računanja negde treba da se zapamti kako bi bio ponovno iskoristiv. Za tu svrhu se koriste elektronska kola koja imaju dva stanja čime se logička nula i jedinica mogu „upamtiti“ u tim kolima. Uvodeći mogućnost memorisanja vrednosti, omogućava se da digitalni sistem prolazi kroz stanja, čime se uvodi sekvencijalnost u rad digitalnog sistema. Kola koja uz komponente iz tabele 1-1 sadrže i elektronska kola za memorisanje nazivaju se **sekvencijalne mreže**. O njima će više biti reči kasnije.

ZADACI

2. Moj prvi digitalni sistem (i Xilinx ISE tutorial)

Projektovanje složenih digitalnih sistema uz pomoć *Xilinx ISE* programskog paketa je moguće na 2 načina: formiranjem logičke šeme digitalnog sistema ili formiranjem HDL (*Hardware Description Language*) opisa digitalnog sistema, tj. opisa pomoću jezika za opis fizičke arhitekture.

Xilinx ISE programski paket podržava opis digitalnih sistema pomoću dva jezika za opis fizičke arhitekture:

1. VHDL (*Very high speed integrated circuit Hardware Description Language*)
2. VERILOG

Postupak projektovanja u Xilinx ISE alatu se uopšteno može podeliti u nekoliko uzastopnih koraka:

1. Formiranje projekta
2. Formiranje datoteke u kojoj će biti opis sistema logičkom šemom ili VHDL jezikom,
3. Opis sistema koristeći logičku šemu ili VHDL jezik,
4. Formiranje datoteke u kojoj će biti opisan test bench sistema,
5. Opis test bencha sistema koristeći VHDL jezik,
6. Provera ispravnosti šeme, sintakse VHDL opisa sistema i VHDL test bencha,
7. Simulacija sistema koristeći napisani test bench u simulatoru,
8. Provera rada sistema posmatrajući simulacioni dijagram.

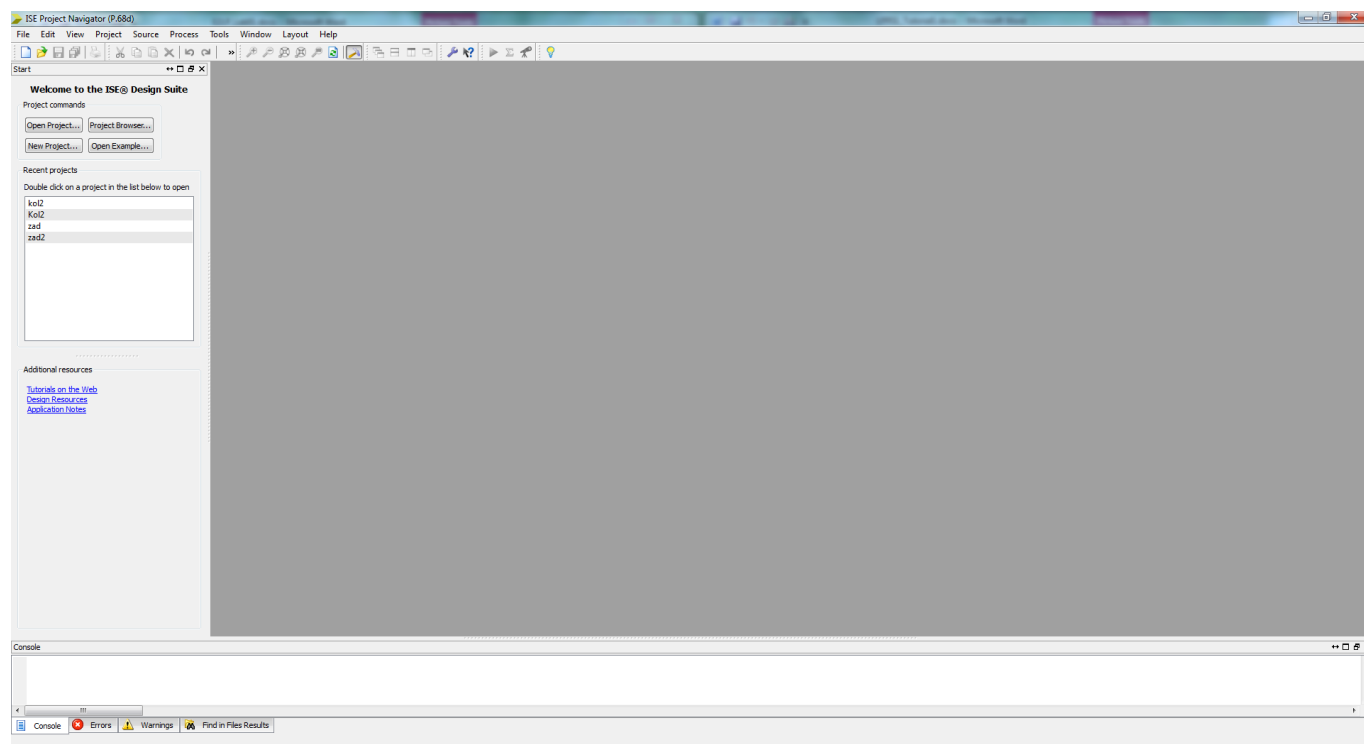
Nakon ovih koraka slede koraci za implementaciju sistema koje ćemo ostaviti za narednu vežbu. Sada ćemo proći kroz sve navedene korake i projektovati naš prvi digitalni sistem.

2.1. Formiranje projekta

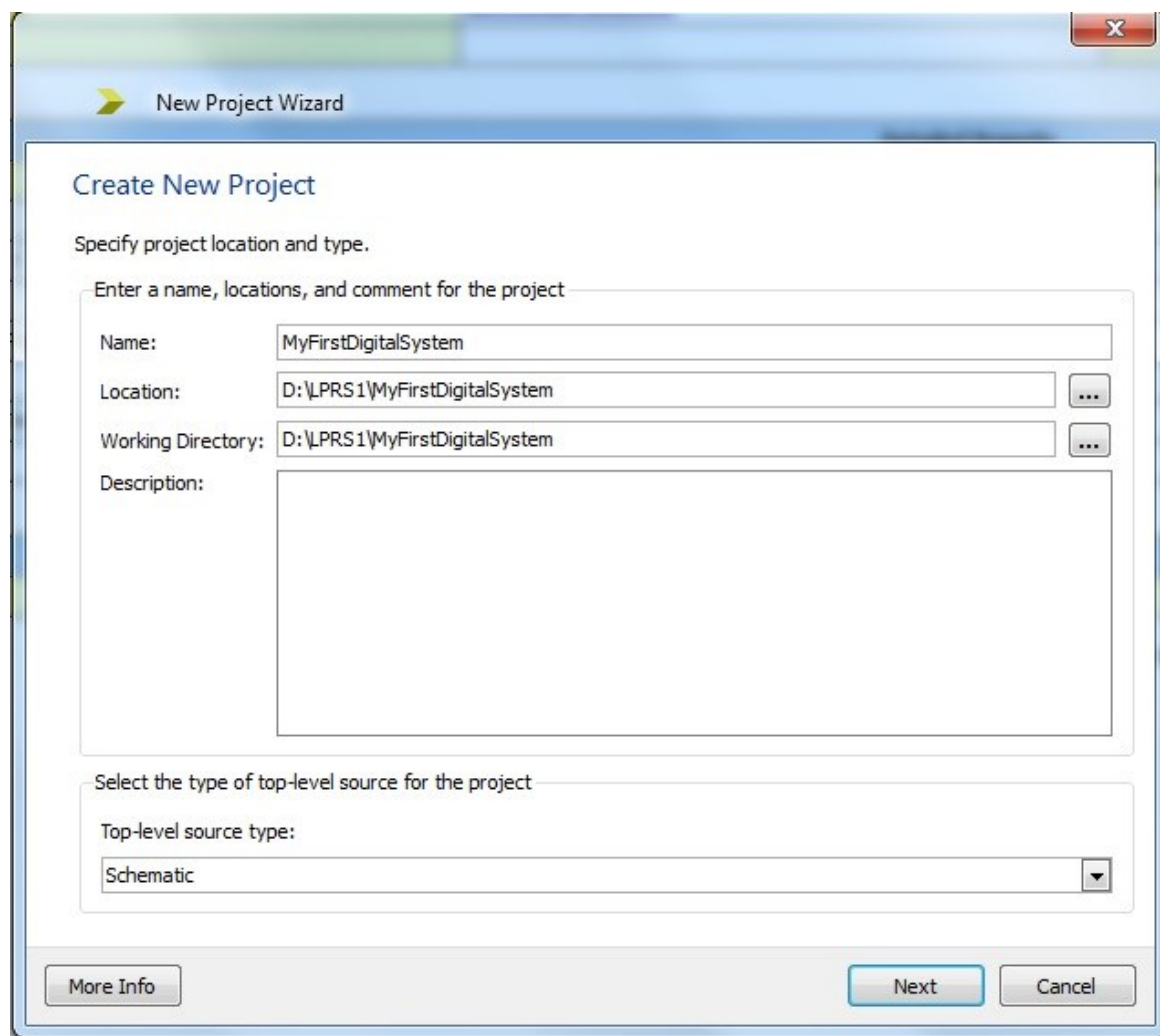
Slika 2-1 prikazuje osnovni prozor *Xilinx ISE* programskog paketa. Vidi se da je on podeljen na tri sastavna dela:

- Početni prozor (**Start**) – levi gornji deo ekrana
U ovom prozoru se mogu pokrenuti neke početne operacije, kao što su formiranje projekta, otvaranje već postojećeg projekta, kao i linkovi ka nekim dodatnim informacijama.
- Izlazni prozor (**Console**) – donji deo ekrana
Prikazuje izlazne poruke pokrenutih procesa
- Radni prozor (**Workspace**) – desni deo ekrana
U ovom prozoru se vrši pisanje i ispravljanje HDL koda ili drugih tekstualnih datoteka, kao i formiranje logičkih šema

Na početku projektovanja digitalnog sistema potrebno je otvoriti projektnu datoteku koja će pamtit sve relevantne podatke o projektu (ime projekta, korišćena komponenta, hijerarhijska struktura, datoteke uključene u projekat). Projektne datoteke se prepoznaju po ekstenziji ***.xise**. Kreiranje projekta se može započeti odabirom komande **File > New Project**, čime se otvara prozor za definisanje informacija o projektu, Slika 2-2.



Slika 2-1. Osnovni prozor Xilinx ISE programskog paketa

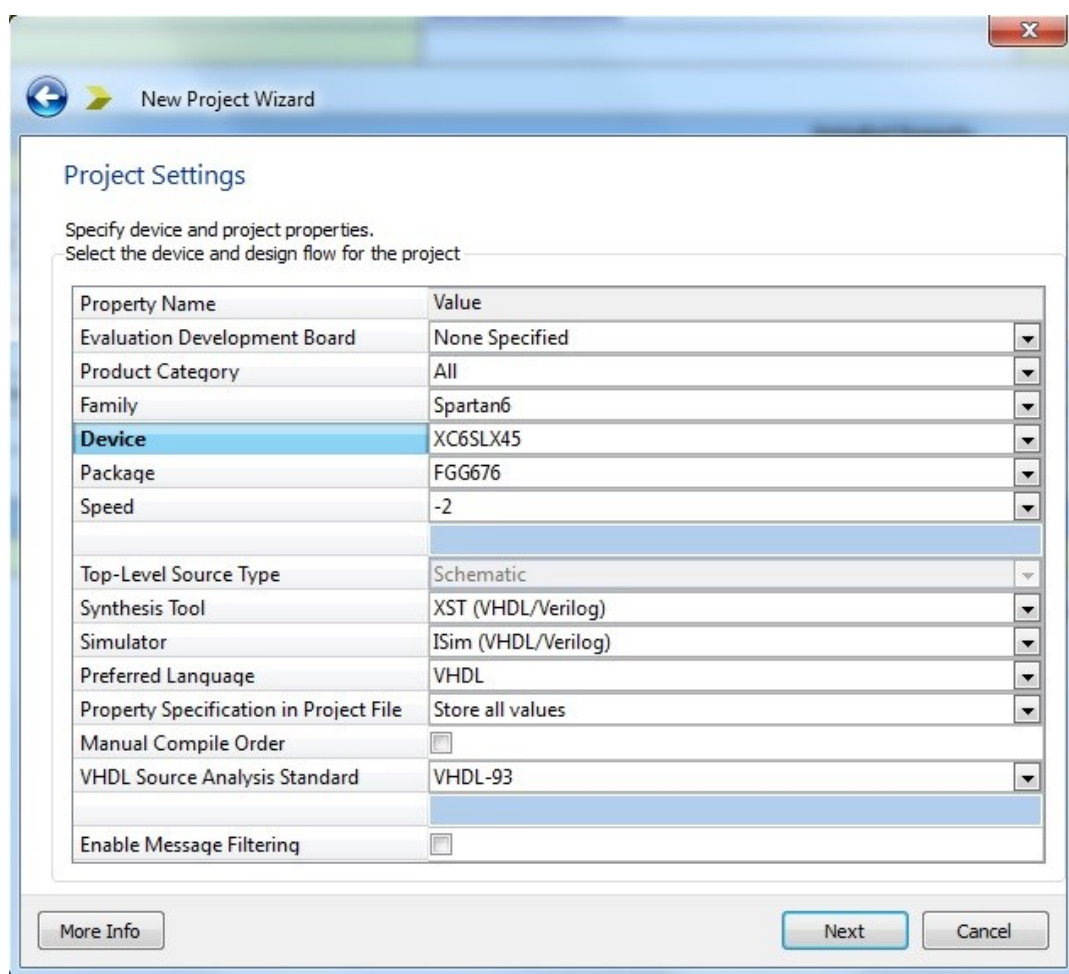


Slika 2-2. Formiranje novog projekta

U polje **Name** treba upisati ime projekta, dok se lokacija projektnog direktorijuma u vidu putanje do određiškog direktorijuma definiše u polju **Location**. U polju **Top-level source type** treba da se izabere tip opisa digitalnog sistema – **HDL** za opis pomoću nekog od jezika za opis fizičke arhitekture ili **Schematic** za opis pomoću formiranja logičke šeme. Na ovoj vežbi ćemo koristiti schematic.

Pritiskom na dugme **Next** prelazi se na sledeći prozor gde se definiše programabilna sekvencijalna mreža za koju se projektuje digitalni sistem, Slika 2-3. U našem slučaju treba odabrati FPGA koja se nalazi na E2LP platformi (više o njoj u narednoj vežbi). Programabilna sekvencijalna mreža je iz familije Xilinx Spartan-6 sa sledećim karakteristikama:

- *Product Category:* **All**
- *Family:* **Spartan6**
- *Device:* **XC6SLX45**
- *Package:* **FGG676**
- *Speed:* **-2**
- *Top-Level Source Type:* **Schematic**
- *Synthesis Tool:* **XST (VHDL/Verilog)**
- *Simulator:* **ISim (VHDL/Verilog)**
- *Preferred Language:* **VHDL**

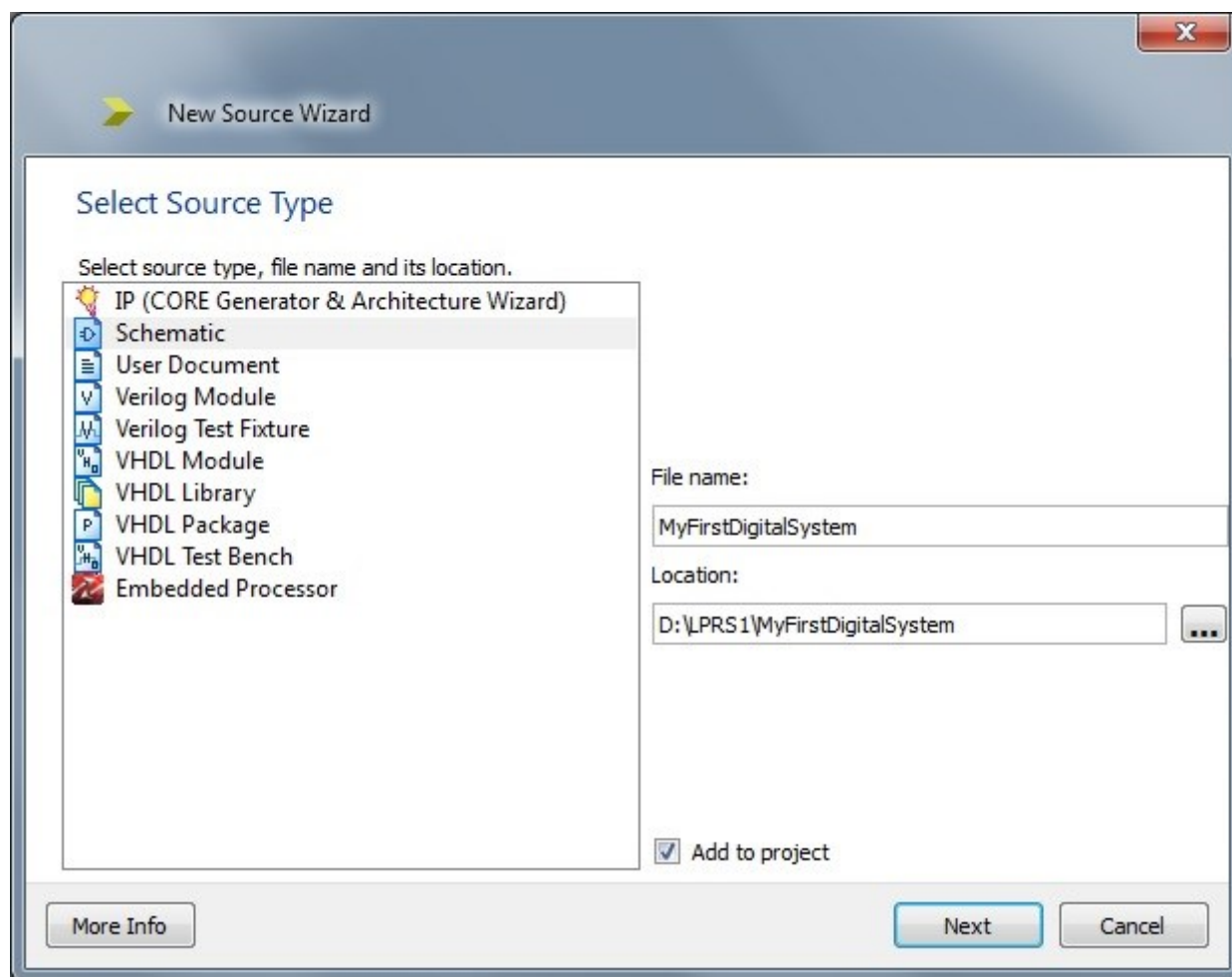


Slika 2-3. Odabir programabilne sekvencijalne mreže

Sledi prozor sa sažetkom odabranih podataka o projektu. Ukoliko se želi promeniti neki parametar, klikom na dugme **Back** se može vratiti unatrag kroz opisane prozore i promeniti željena informacija. Klikom na dugme **Finish** se završava formiranje projekta i dobija se osnovni *Xilinx ISE* prozor.

2.2 Formiranje logičke šeme digitalnog sistema i njeno uključivanje u projekat

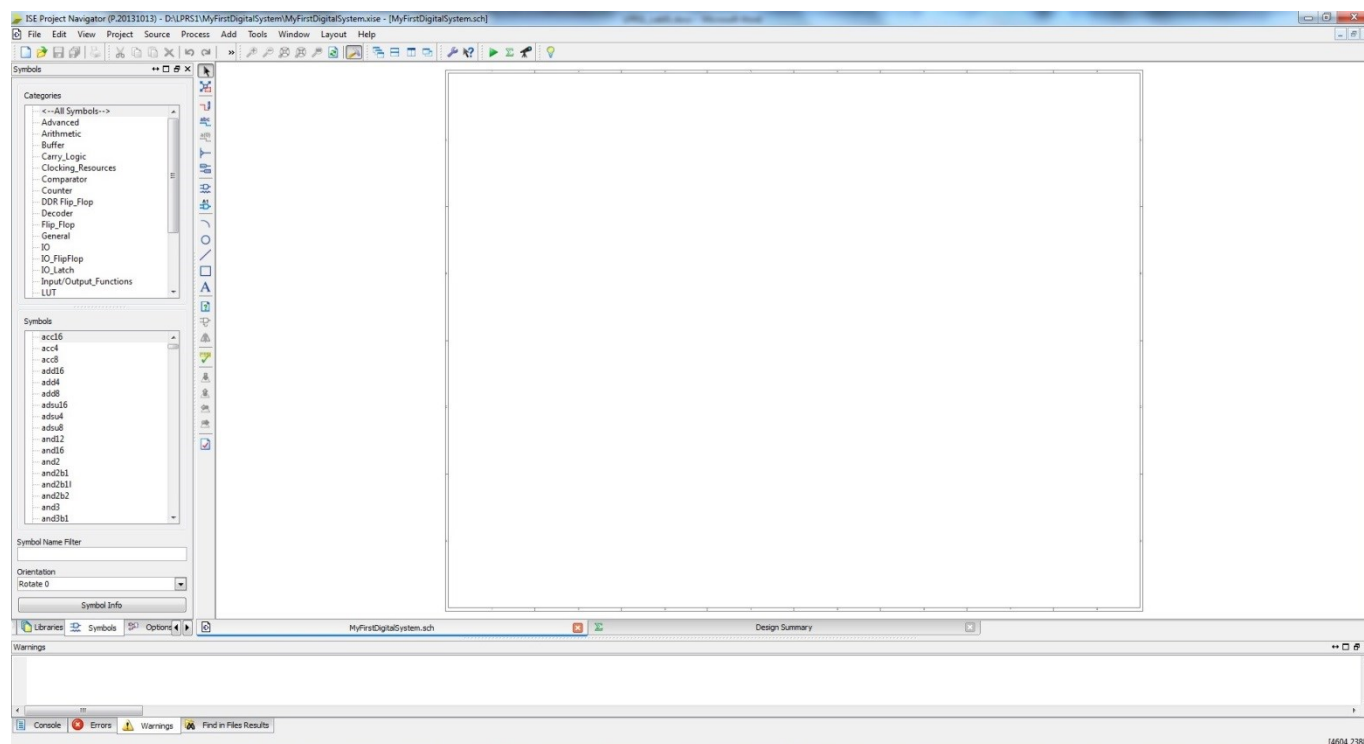
Nova datoteka gde će biti smeštena logička šema digitalnog sistema se formira pomoću komande **New Source** iz **Project** menija. Ova komanda se takođe može pozvati iz menija koji se dobija desnim klikom na oznaku programabilne sekvencijalne mreže (**xc6slx45-2fgg676**) u projektnom prozoru. Slika 2-4 prikazuje prozor koji se dobija pozivom ove komande.



Slika 2-4. Formiranje datoteke sa opisom digitalnog sistema

Sa leve strane treba odabrati **Schematic** a u polje **File name** upisati ime datoteke u koju će se smestiti logička šema. Putanja do direktorijuma gde će biti smeštena datoteka se definiše u polju **Location**. Kontrolno polje **Add to project** mora da je selektovano radi dodavanja novoformirane datoteke u projekat.

Pritiskom na dugme **Next** se dobija kratak sažetak upisanih informacija i pritiskom na dugme **Finish** se novoformirana datoteka otvara u radnom prozoru, Slika 2-5.



Slika 2-5. Radni prozor za formiranje logičke šeme

Logička šema se formira prevlaštenjem simbola logičkih kola, koji se nalaze u prozoru Symbols, na radnu površinu sa desne strane. Simboli koji odgovaraju logičkim kolima iz tabele 1-1 su:

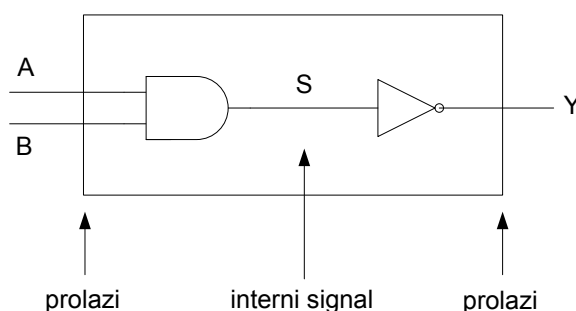
- **inv** – invertor (logičko kolo NOT),
- **and2** – dvoulazno AND kolo,
- **or2** – dvoulazno OR kolo,
- **xor2** – dvoulazno XOR kolo,
- **nand2** – dvoulazno NAND kolo,
- **nor2** – dvoulazno NOR kolo,
- **xnor2** – dvoulazno XNOR kolo.

Veze između kola se definišu spajanjem njihovih ulaza/izlaza pomoću žica:  **Add Wire**.

Ulazi i izlazi sistema se definišu pomoću ulazno/izlaznih markera:  **Add I/O Marker**.

2.3. Opis digitalnog sistema logičkom šemom

Vreme je da opišete vaš prvi digitalni sistem! Nacrtajte logičku šemu sistema sa slike 2-6.



Slika 2-6. Naš prvi digitalni sistem

2.4. Dodavanje datoteke za VHDL test bench

Nakon završetka opisa digitalnog sistema, potrebno je proveriti njegovu funkcionalnost. Funkcionalnost digitalnog sistema ćemo proveriti koristeći VHDL test bench kao generator ulaza u naš sistem. Test bench takođe preuzima vrednosti koje naš sistem generiše na izlazu i pamti ih u internim signalima.

U ovoj vežbi nećemo formirati novi test bench, nego ćemo koristiti već napravljen. Kako bi se dodala datoteka za opis VHDL test bencha, potrebno je izabrati **File --> Add Source**, nakon čega treba izabrati datoteku *MyFirstDigitalSystem_tb.vhd*.

2.5. Opis test bencha za proveru rada digitalnog sistema




Sadržaj test bencha je izvan opsega ove vežbe, pa ćemo ovaj deo preskočiti.

2.6. Provera sintakse






Nakon završenog opisa test bencha, vratite se u *Design* prozor (na glavnom prozoru Xilinx ISE alata). Ovaj prozor je podeljen u dva dela:

- gornji deo sa informacijom o datotekama uključenim u projekat,
- donji deo sa komandama i zadacima koji se mogu izvršiti u projektu.

U prozoru sa raspoloživim procesima je sada moguće po potrebi odabrati izvršenje nekog zadatka. Zadaci su podeljeni u sledeće tri kategorije:

-  **Zadaci (Tasks)**
Pokretanjem ovog procesa ISE programski paket je pokrenut u tzv. "**batch mode**" režimu rada. Tokov izvršenja ovog tipa procesa vrši se odgovarajuće procesiranje ulaznih datoteka i ne otvaraju se nikakve dodatne alatke u radnom prozoru. Izlazni procesi i njihovo stanje se pojavljuje u izlaznom prozoru.
-  **Izveštaji (Reports)**
Izvršenje većine zadataka generiše odgovarajuće izveštaje koji se mogu analizirati odabirom ovog tipa procesa. Pri pokretanju izveštaj se pojavljuje u radnom prozoru.
-  **Dodatne alatke (Tools)**
Pokretanjem ovakvog tipa procesa otvara se povezana alatka kao nezavisna aplikacija ili u radnom prozoru.

Tokom projektovanja digitalnog sistema svaki proces se može naći u nekom od sledećih stanja

-  **Running** – proces je u stanju izvršavanja.
-  **Up-to-date** – Proces je uspešno izvršen bez grešaka ili upozorenja. Ako se izveštaj nađe u ovom stanju znači da je izveštaj ažuran, mada se može desiti da zadaci obuhvaćeni ovim procesom imaju greške ili upozorenja.
-  **Warnings reported** – Proces je uspešno izvršen ali postoje upozorenja.
-  **Errors reported** – Proces je izvršen sa najmanje jednom greškom.
-  **Out-of-Date** – Označava da su u projektu napravljene izmene i da se proces mora ponovo izvršiti.
- **No icon** – U slučaju da pored procesa nema nijedne od prethodnih ikonica proces nije pokrenut.

U gornjem delu *Design* prozora, mogu se izabrati dva pogleda:

- *Implementation* – u kome su prikazane samo datoteke koje služe implementaciji digitalnog sistema, to su sve datoteke sem test bencheva,
- *Simulation* – u kome su prikazane sve datoteke uključujući i datoteke za proveru (test bench).

Dakle, pređimo u pogled za simulaciju i izaberimo test bench u spisku datoteka. Primetite da je opis vašeg sistema “uvučen” u odnosu na test bench, što govori da test bench proverava vaš sistem. Jako je bitno uvek imati izabran test bench prilikom pokretanja simulacije jer su komande u donjem delu prozora zavisne od datoteke koja je izabrana u gornjem delu.

U donjem delu prozora, raširite opciju *ISim Simulator* i izaberite **Behavioral Check Syntax**. Ova opcija proverava VHDL sintaksu selektovane datoteke (u našem slučaju VHDL test bencha) i svih datoteka koje su hijerarhijski ispod ove datoteke, odn. “uvučene” u odnosu na nju (u našem slučaju opis sistema).

Ukoliko se otkrije neka greška u VHDL sintaksi, poruka sa greškom se prikazuje u konzolnom prozoru ispod. U suprotnom, pojavljuje se zelena oznaka ispravnosti VHDL datoteke.

2.7. Pokretanje simulacije sistema

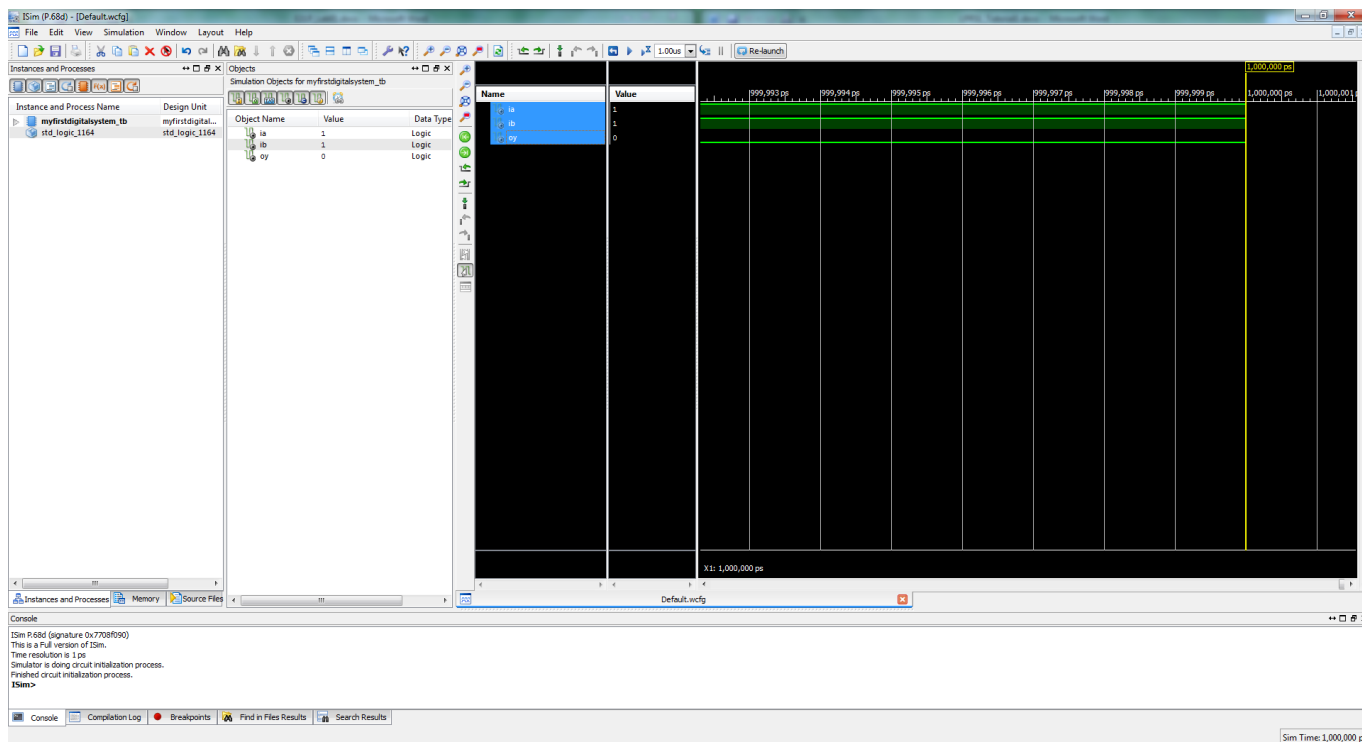
ISim simulator je sastavni deo Xilinx ISE programskog paketa. Uz pomoć ovog programa moguće je izvršiti funkcionalnu proveru ispravnosti digitalnog sistema koji se projektuje. Ovde se navode samo osnovne komande za njegovo korišćenje.

Simulacija HDL koda se izvršava u tri osnovna koraka (važi za svaki HDL simulator):

- Prevođenje izvornog koda (**Compile**) – sadrži ranije navedenu opciju za proveru sintakse
- Pokretanje radnog režima za simulaciju (**Simulate**)
- Pokretanje simulacije (**Run**)


Kada se ISim simulator pozove iz radnog okruženja sva tri koraka se izvrše automatski. U slučaju greške u nekom od koraka, poruka o grešci će biti ispisana u konzoli.

Ukoliko nema greške, otvara se prozor simulatora, slika 2-7.


















Slika 2-7. Primer prozora za simulaciju

2.8. Provera rada sistema posmatranjem simulacionog dijagrama

Grupa ikona  omogućava podešavanje prikaza signala kao i kontrolu toka izvršenja simulacije.

Značenja pojedinih ikona su:

-  **Zoom In** – povećanje nivoa uvećanja.
-  **Zoom Out** – smanjenje nivoa uvećanja.
-  **Zoom to Full View** – podešavanje nivoa uvećanja na nivo koji omogućava pregled kompletnog toka simulacije.
-  **Zoom to Selected** – podešavanje nivoa uvećanja na selektovani deo dijagrama..
-  **Redraw All Windows** – ponovno iscrtavanje svih prozora.
-  **Previous Transition** – skok na prethodnu promenu signala.
-  **Next Transition** – skok na narednu promenu signala.
-  **Add Marker** – postavljanje pojedinačnog markera.
-  **Previous Marker** – skok na prethodni marker.
-  **Next Marker** – skok na sledeći marker.
-  **Restart** – restartovanje simulacije.

-  **Run All** – pokretanje izvršenja simulacije na neodređen vremenski period.
-  **Run for the time specified on the toolbar** – pokretanje izvršenja simulacije na vremenski period definisan u susednom polju.
-  **Step** – izvršenje simulacije korak po korak sa uvidom u progres izvršenja u HDL kodu.
-  **Break** – zaustavljanje izvršenja simulacije.

Da bi simulacioni dijagram bio pregledniji, potrebno je pritisnuti na ikonu **Zoom to Full View**. Na početku, dijagram prikazuje sve signale i vektore u binarnim vrednostima. Radi lakše čitljivosti vektora, moguće je promeniti način ispisivanja njihovih vrednosti u neki drugi brojni sistem, npr. heksadecimalni.

Pored signala sa vrha hijerarhije, moguće je dodati i druge signale u prikaz rezultata simulacije. Da bi neki signal dodali u listu prikazanih potrebno ga je prvo selektovati u Objects potprozoru i nakon desnog klika izabrati opciju Add to Wave Window. Ovim je signal dodan u listu signala čiji se oblik iscrtava. Oblici naknadno uključenih signala biti iscrtani samo u vremeskim trenucima simulacije nakon trenutka uključivanja signala u listu. Ukoliko želimo omogućiti iscrtavanje oblika svih signala od nultog trenutka (i onih naknadno dodatih) neophodno je zatvoriti prozor rezultata simulacije i ponovo pokrenuti simulaciju.

3. Moj (već) drugi digitalni sistem

Čestitamo na vašem prvom uspešno projektovanom, opisanom i proverenom digitalnom sistemu! Sada je vreme da sami implementirate nešto složeniji sistem (kombinacionu mrežu) sa 3 izlaza i 3 ulaza.

Pretpostavimo da je naš novi digitalni sistem opisan sledećom istinitisnom tablicom, tj. sledećim trima Bulovim funkcijama od 3 promenljive:

iX2	iX1	iX0	oY2	oY1	oY0
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	0

Sledeće jednačine predstavljaju Bulove funkcije koje sistem računa:

$$oY2 = iX2 * \overline{iX0} + iX2 * \overline{iX1} + \overline{iX2} * iX1 * iX0$$

$$oY1 = iX1 * \overline{iX0} + \overline{iX1} * iX0$$

$$oY0 = \overline{iX0}$$

Vaš zadatak je da prođete kroz sve korake projektovanja sistema i opišete i simulirate ovaj sistem koristeći Xilinx ISE alat.

I za kraj jedno

Pitanje: Ukoliko ulaz i izlaz sistema posmatramo kao 3-bitne brojeve (gde je indeks bita jednak stepenu mesne vrednosti broja), koju matematičku funkciju ovaj sistem računa?

4. Zaključak

Ova vežba vas je upoznala sa pojmom digitalnih logičkih kola i kombinacionih mreža kao fizičkim realizacijama Bulovih funkcija. Videli ste kako se digitalni sistem projektuje, od momenta ideje na papiru, preko opisa sistema do simulacije rada sistema. Znanje koje ste dobili u ovoj vežbi je dovoljno da ste u stanju samostalno da projektujete bilo koju kombinacionu mrežu zadatu Bulovim funkcijama. U narednoj vežbi videćemo kako nam VHDL jezik značajno olakšava projektovanje kombinacionih mreža jer njihovo projektovanje na nivou logičkih kola, kao u ovoj vežbi, postaje veoma teško i mukotrpno ukoliko se funkcija te kombinacione mreže iole zakomplikuje.