POKAZNA VEŽBA 3

VHDL opis kombinacionih mreža

Potrebno predznanje

- Urađena pokazna vežba 2
- Standardne kombinacione mreže

Šta će biti naučeno tokom izrade vežbe?

Nakon urađene vežbe, bićete u mogućnosti da:

- Opišete proizvoljnu kombinacionu mrežu u VHDL-u koristeći opis na nivou logičkih kola
- Opišete proizvoljnu kombinacionu mrežu u VHDL-u koristeći uslovne dodele
- Opišete proizvoljnu kombinacionu mrežu u VHDL-u koristeći kombinacione procese
- Sintetišete i implementirate vaš sistem za FPGA integrisano kolo

Apstrakt i motivacija

Opis digitalnih sistema na nivou logičkih kola, bilo da ga radite crtajući električnu šemu ili opisujući VHDL jezikom na nivou logičkih kola, zahteva da poznajete Bulove jednačine sistema koji želite da opišete. Da biste došli do Bulovih jednačina, neophodno je da obavite proces izvođenja jednačine koji podrazumeva izvođenje SDNF iz istinitosne tablice željenog ponašanja sistema i minimizaciju te jednačine analitičkim pristupom ili nekom metodom minimizacije, npr. Karnoovim mapama. Postupak izvođenja i minimizacije jednačine je šablonski i lako ga je automatizovati, a ne doprinosi kreativnosti u dizajnu samog digitalnog sistema. Zbog toga VHDL jezik podržava konstrukcije koje će Vama, kao autoru digitalnog sistema, omogućiti da sistem opišete koncentrišući se na logiku ponašanja sistema, tj. šta želite da sistem ima na izlazu i pod kojim uslovima, a šablonski postupak izvođenja i minimizacije jednačina ostavite računaru. Na kraju vežbe imaćete priliku da implementirate svoj prvi složeni kombinacioni sistem koji veoma liči na aritmetičko-logičku jedinicu nekog procesora.

TEORIJSKE OSNOVE

1. VHDL opis kombinacionih mreža

VHDL jezik omogućuje nekoliko načina za opis kombinacionih mreža, koji su kraći i čitljiviji od opisa na nivou logičkih kola. Ovi opisi podsećaju na opis programske podrške pošto koriste iste ili slične konstrukte. Kombinacione mreže se u VHDL jeziku mogu opisati sledećim načinima:

- Dodelom vrednosti signalu na nivou logičkih kola (način koji smo koristili u prethodnim vežbama),
- Dodelom vrednosti signalu pomoću ostalih VHDL operatora,
- Uslovnom dodelom vrednosti signalu,
- Pomoću kombinacionih procesa.

1.1. Dodela vrednosti

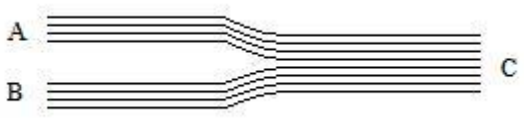
Signalu se u VHDL jeziku vrednost može dodeliti pomoću operatora **<= .**

Na nivou logičkih kola vrednost signala se definiše navodeći logički izraz sa desne strane, koristeći VHDL logičke operatore **not**, **and**, **or**, **xor**, **nand**, **nor**, **xnor**, kao što smo radili u prethodnim vežbama.

Prilikom definisanja (dodele) vrednosti nekom signalu, mogu se koristiti i ostali VHDL operatori. Trenutno je za nas najzanimljiviji operator **konkatenacije (&).** Operator konkatenacije omogućuje da signalu dodelimo vrednost kombinacije nekih drugih signala. Na primer, ako pretpostavimo da je signal C 8-bitni, a signali A i B 4-bitni, tada ako napišemo sledeće:

```
C <= A & B;
```

signal C će dobiti vrednost koja se dobije spajanjem signala A i B. Viših 4 bita signala C će dobiti vrednost signala A, a nižih 4 bita signala C će dobiti vrednost signala B. Konkatenacija se može posmatrati i kao davanje novog imena skupu žica. Posmatrajući Sliku 1-1, ako signal A predstavlja gornje 4 žice, a signal B predstavlja donje 4 žice (svaka žica prenosi 1 bit, signali su 4-bitni), tada gornja konkatenacija znači da ovih 8 žica posmatramo kao jedinstveni signal C. Konkatenacija ne definiše nove komponente u digitalnom sistemu, već samo postojećim "žicama" iz drugih signala daje novo ime i posmatra kao novu celinu.



Slika 1-1. Konkatenacija signala A i B u signal C

Konkatenacija se može vršiti i nad delovima signala koristeći operator indeksiranja, kao i nad konstantama. Na primer, ako napišemo sledeće:

```
C <= '1' & A(1 downto 0) & B(3 downto 2) & "00";
```

definišemo signal C širine 7 bita, i to jednog bita vrednosti konstante '1', dva bita iz signala A, dva bita iz signala B i dva bita vrednosti konstante '0'.

Za kompletan spisak VHDL **aritmetičkih operatora** preporučujemo da konsultujete udžbenik i naredne vežbe, jer oni izlaze izvan opsega ove vežbe.

1.2. Uslovna dodela vrednosti

Uslovna dodela se koristi da bi se opisala kombinaciona logika tipa "IF-THEN-ELSE". Kako bi se skratio VHDL opis, VHDL ne zahteva da se ovakva logika opisuje na nivou logičkih kola, već za to postoji posebna konstrukcija prilikom dodele signala. Sintaksa uslovne dodele je data u Listingu 1-1.

Listing 1-1. Sintaksa uslovne dodele u VHDL-u

```
<signalName> <= <expression1> when <condition1> else
                <expression2> when <condition2> else
                ...
                <expressionN> when <conditionN> else
                <expressionDefault>;
```

Signal `<signalName>` dobija vrednost `<expression1>` ukoliko je ispunjen uslov `<condition1>`, u suprotnom dobija vrednost `<expression2>` ukoliko je ispunjen uslov `<condition2>`, itd. Ukoliko nijedan od uslova nije ispunjen, signal dobija vrednost `<expressionDefault>`.

Vrednosti sa desne strane se mogu definisati pomoću VHDL operatora kao u prethodnoj sekciji, a za definisanje uslova mogu se koristiti VHDL **relacioni operatori**:

=	jednakost	<	manje od	<=	manje od ili jednako
/=	nejednakost	>	veče od	>=	veće od ili jednako

1.3. Kombinacioni procesi

Sem pomoću operatora dodele vrednosti, kojim se definiše vrednost izlaza kombinacione mreže na osnovu vrednosti ulaza, kombinaciona mreža se može opisati i pomoću **kombinacionog procesa**. Listing 1-2 prikazuje sintaksu za VHDL opis kombinacione mreže pomoću procesa.

Listing 1-2. Sintaksa VHDL kombinacionog procesa

```
process (<sensitivityList>) begin
    <processBody>
end process;
```

Unutar zagrada u prvoj liniji procesa definiše se **lista osetljivosti** procesa. Lista osetljivosti služi alatu za simulaciju kako bi znao kada komponenta (kombinaciona mreža) koja se opisuje procesom može da promeni izlaznu vrednost. Prema osobinama kombinacione mreže, promena izlazne vrednosti se može desiti prilikom promene bilo kog od ulaza, pošto izlaz uvek predstavlja funkciju trenutnih vrednosti ulaza. Dakle, u listi osetljivosti kombinacionog procesa treba da budu navedeni **svi ulazi kombinacione mreže koja se procesom opisuje**, odvojeni zarezom.

Telo procesa sadrži opis kombinacione mreže. Svim izlazima kombinacione mreže treba da se dodeli vrednost. Za tu svrhu se može koristiti operator dodele, ali i VHDL konstrukcije koje se mogu koristiti samo u procesima, a najčešće korištene su uslovne konstrukcije IF i CASE. Listing 1-3 i Listing 1-4 prikazuju sintaksu ovih konstrukcija.

Listing 1-3. Sintaksa IF strukture

```
if (<condition1>) then
    <statements>
elsif (<condition2>) then
    <statements>
else
    <statements>
end if;
```

Listing 1-4. Sintaksa CASE strukture

```
case (<signalName>) is
    when <value1> => <statements>
    when <value2> => <statements>
    ...
    when <valueN> => <statements>
    when others => <statements>
end case;
```

IF i CASE struktura imaju istu logiku kao i u programskim jezicima. Samo iskazi koji su u delu strukture za koju je uslov zadovoljen će definisati ponašanje izlaza kombinacione mreže.

Prilikom definisanja kombinacionih mreža pomoću procesa koji sadrže IF i CASE, neophodno je definisati svaki izlaz procesa u svakoj grani IF i CASE strukture. U suprotnom kombinaciona mreža je nepotpuno definisana i rezultovaće lošim digitalnim sistemom. Takođe treba izbegavati paralelne IF i CASE strukture pošto svaki izlaz definiše **jedna** logička funkcija, a svaka IF i CASE struktura predstavlja različitu logičku funkciju.

1.4. Koja je razlika ranijih načina opisivanja?

Naveli smo dva pristupa opisivanju kombinacionih mreža – pomoću operatora dodele (što se još naziva i konkurentnim iskazom) i pomoću kombinacionih procesa. Između navedenih načina opisivanja kombinacionih mreža **nema razlike**, odn. oba načina rezultuju istom kombinacionom mrežom. Procesi imaju više podržanih konstrukcija, a dodele su u najvećem broju slučajeva kraće i ne zahtevaju listu osetljivosti. Da li ćete koristiti dodele ili procese ostaje na vama.

ZADACI

2. Multiplekser

Multiplekser je kombinaciona mreža koja koja implementira uslovnu logiku – vrednost izlaza multipleksera je jednaka vrednosti jednog od njegovih ulaza. Koji ulaz se prosleđuje na izlaz zavisi od stanja selekcionih bita.

Vaš zadatak je da implementirate 4x1 multiplekser (4 ulaza opšte namene, 2 selekciona ulaza i 1 izlaz) na sledeće načine:

- definisanjem tabele istinitosti, Bulove jednačine izlaza i crtanjem logičke šeme,
- opisom u VHDL-u na nivou logičkih kola,
- opisom u VHDL-u koristeći uslovnu dodelu,
- opisom u VHDL-u koristeći kombinacioni proces.

3. Primena multipleksera u složenijem sistemu

Vaš naredni zadatak je da implementirate složeniji digitalni sistem, sa dva 4-bit ulaza (iA i iB), jednim selekcionim ulazom od 2 bita (iSEL) i jednim 4-bit izlazom (oC), tako da se sistem ponaša prema sledećoj jednačini:

$$oC = \begin{cases} iA & if\ iSEL = \text{"00"} \\ iB & if\ iSEL = \text{"01"} \\ iA + iB & if\ iSEL = \text{"10"} \\ iA + 1 & if\ iSEL = \text{"11"} \end{cases}$$

Primetite da ovaj sistem sadrži 10 ulaza i 4 izlaza, što znači da on računa 4 Bulove funkcije od 10 promenljivih. Tradicionalni pristup projektovanju digitalnih sistema kroz istinitosne tablice i crtanjem logičke šeme bi trajao veoma dugo, a VHDL nam omogućuje opis istog sistema u samo nekoliko linija!

Ali nikada ne treba zaboraviti da ono što mi opisujemo VHDL-om jesu logička kola i jesu Bulove funkcije izvedene iz istinitosne tablice! VHDL je samo drugi (jednostavniji) način da se opiše isti sistem.

Prilikom implementacije sistema povežite ulaze iA i iB na prekidače, selekcione ulaze na tastere, a izlaz na LED-ove po izboru.

ZAKLJUČAK

Koristeći uslovne dodele i kombinacione procese moguće je opisati veoma složenu Bulovu funkciju na kraći i razumljiviji način. Nakon kompletiranja ove vežbe, trebali bi biti u stanju da opišete proizvoljno složenu kombinacionu mrežu koristeći VHDL konstrukcije i tradicionalno, pomoću Bulovih funkcija.

DODATAK. Složeni kombinacioni sistem (priprema za prvi kolokvijum)

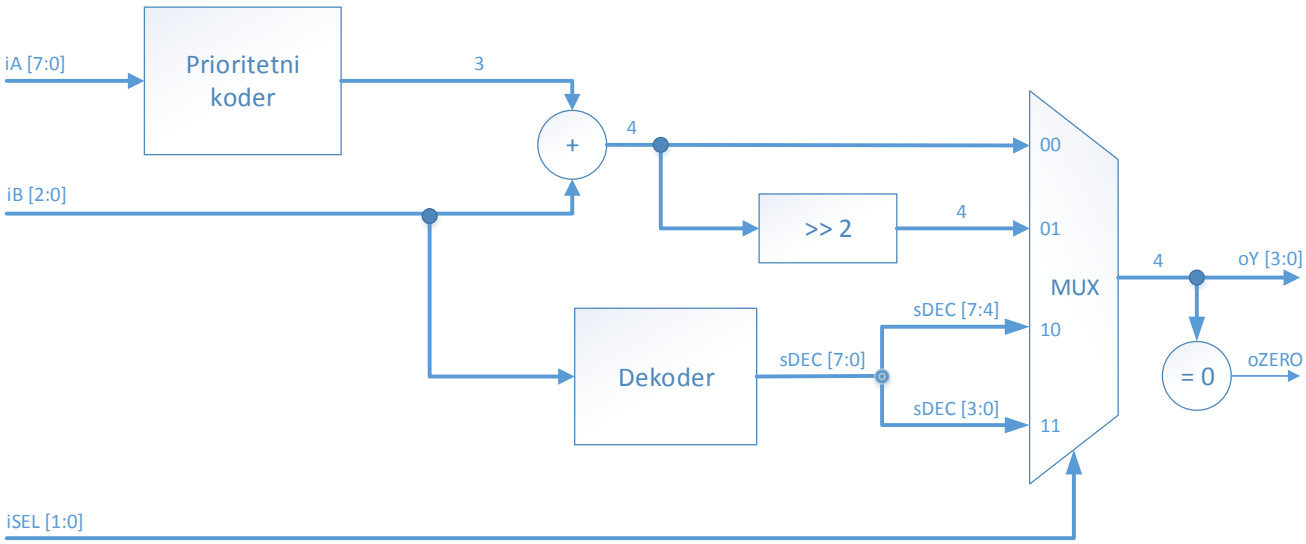
U VHDL jeziku za opis hardvera opisati i simulirati digitalni sistem prikazan na slici koji predstavlja jednostavnu aritmetičko-logičku jedinicu.

Ulazi digitalnog sistema:

- **iA [7:0]** – prvi ulazni operand koji se preuzima „dekodovan“, tj. kao indeks jednog od 8 bita,
- **iB [2:0]** – drugi ulazni operand koji se preuzima „kodovan“ u binarnoj predstavi,
- **iSEL [1:0]** – izbor operacije.

Izlazi digitalnog sistema:

- **oY [3:0] –** rezultat operacije,
- **oZERO** – oznaka da li je rezultat operacije jednak nuli.



Slika 1. Aritmetičko-logička jedinica

Operacije koje aritmetičko-logička jedinica podržava su:

- zbir dva ulazna operanda,
- zbir dva ulazna operanda podeljen sa 4, tj. aritmetički pomeren u desno za 2 mesta,
- gornja 4 bita dekodovane vrednosti drugog operanda,
- donja 4 bita dekodovane vrednosti drugog operanda.

Sistem simulirati izvršavajući svaku od operacija bar jednom, sa bar 2 različite vrednosti na svakom operandu. Bar jedan od slučajeva treba da rezultuje nulom i bar jedan od slučajeva brojem različitim od nule.