Fakultet elektrotehnike i računarstva
Zavod za elektroniku, mikroelektroniku,
računalne i inteligentne sustave

DIGITALNA LOGIKA

UPUTA ZA ČETVRTU LABORATORIJSKU VJEŽBU

dr. sc. Marko Čupić

Zagreb, 2013.

1. Uvod

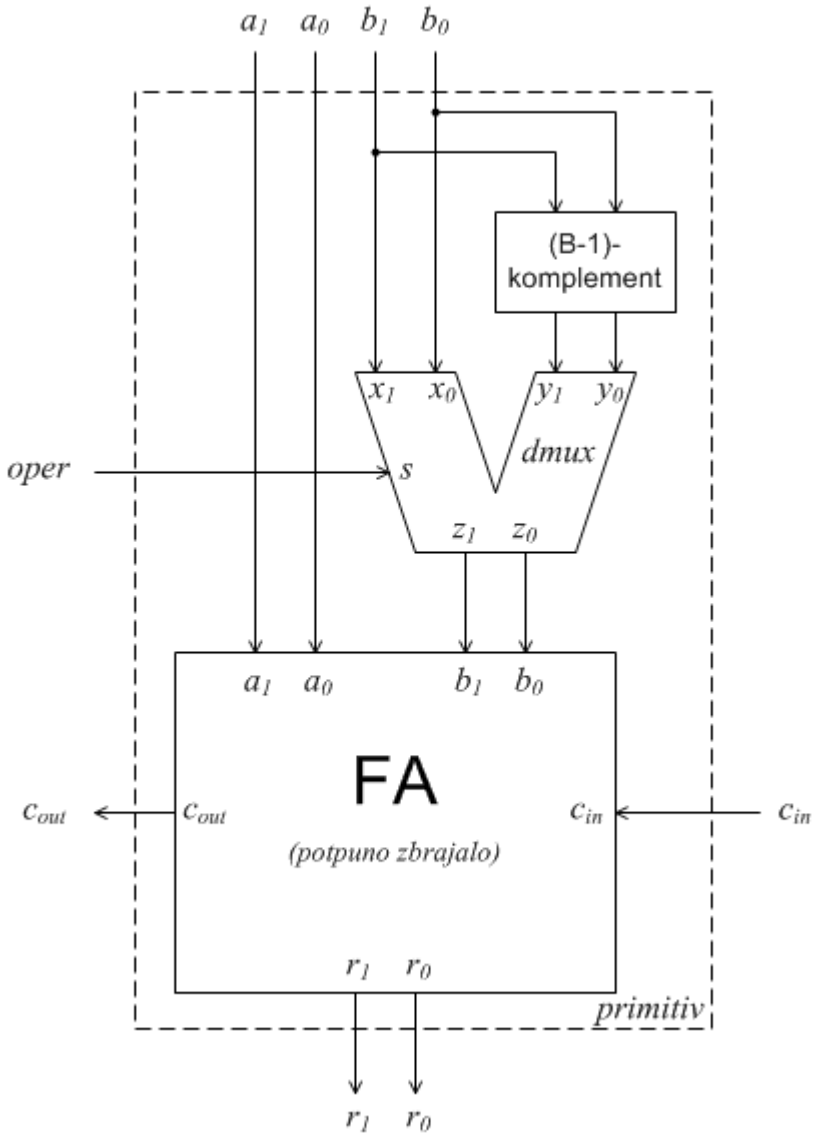
U ovoj laboratorijskoj vježbi obrađuje se digitalna aritmetika. Konkretno, vježba se izrada sklopovlja za zbrajanje/oduzimanje u kodu. Jedan poznati primjer ovakvih sklopa predstavljaju sklopovi za zbrajanje BCD znamenaka, kod kojih se na ulaz dovode dva 4-bitna binarna uzorka (koje tumačimo kao dvije dekadske znamenke zapisane u BCD kodu), a na izlazu se generira novi 4-bitni binarni uzorak koji tumačimo kao znamenku rezultata zapisanu u BCD kodu.

Ako govorimo o osnovama digitalne aritmetike odnosno o zbrajanju binarnih znamenaka, definira se pojam "potpuno zbrajalo" (engl. *Full Adder*), kao sklop koji na ulaz dobiva dva bita koja treba zbrojiti te prijenos s nižeg mjesta, a generira bit rezultata te prijenos za više mjesto (preporuka: pogledati u [1] poglavlje 7.6). Kada razmatramo aritmetiku u kodu, odnosno artmetiku nad znamenkama proizvoljne baze, tada je pojam potpuno zbrajalo potrebno proširiti na sklop koji na ulaz dobiva dvije znamenke te prijenos s nižeg mjesta, a na izlazu generira znamenku rezultata te prijenos za više mjesto. Ulančavanjem ovakvih sklopova moguće je postići zbrajalo višeznamenkastih brojeva (preporuka: pogledati u [2] zadatke 8.7 i 8.8). Jednom kada na raspolaganju imamo sklop koji zna obaviti zbrajanje dviju znamenaka, moguće je konstruirati sklop koji će obavljati oduzimanje tih znamenaka, odnosno sklop koji će obavljati oduzimanje višeznamenkastih brojeva (preporuka: pogledati u [2] zadatke 8.9, 8.10, 8.11). Konačno, pažljivim kombiniranjem takvih sklopova odnosno malom modifikacijom njihove građe moguće je izgraditi sklopove koji će obavljati zbrajanje ili oduzimanje dviju znamenaka ovisno o nekom upravljačkom ulazu, odnosno zbrajanje ili oduzimanje dvaju višeznamenkastih brojeva ovisno o nekom upravljačkom ulazu (preporuka: pogledati u [2] zadatak 8.12).

U ovoj vježbi izgradit ćemo sklop koji obavlja zbrajanje/oduzimanje brojeva u bazi 4. Ako govorimo o bazi 4, postoje 4 znamenke: 0, 1, 2 i 3. Kako imamo više od dvije vrijednosti znamenke, očito je da svaku znamenku trebamo kodirati određenom kombinacijom bitova. Za prikaz 4 različite vrijednosti minimalno su nam potrebna 2 bita. U nastavku je prikazan primjer gdje se svaka znamenka kodira kombinacijom koja odgovara Grayevom kodu.

Znamenka	***Kodna riječ***
0	00
1	01
2	11
3	10

Opća struktura sklopa koji može poslužiti za izgradnju višeznamenkastog zbrajala odnosno oduzimala (ovisno o upravljačkom ulazu), prikazana je u zadatku 8.12 [2]. Što se dobije kada tu strukturu prilagodimo za brojeve kod koji je znamenka kodirana s 2 bita prikazano je u nastavku.



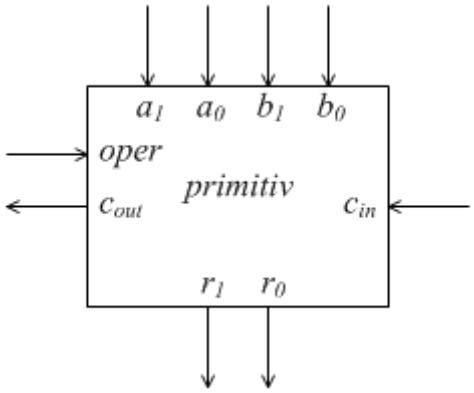
Sklop označen s FA predstavlja potpuno zbrajalo. To je sklop koji na ulaz dobiva dvije znamenke (dva dvobitna binarna niza koji predstavljaju kodirane znamenke), a na izlazu generira znamenku rezultata (dvobitni binarni niz koji predstavlja kodirani rezultat). Prilikom zbrajanja, u obzir se dakako uzima i prijenos s nižeg mjesta, odnosno generira se prijenos za više mjesto. Primjerice, ako u našem slučaju na ulaz sklopa FA dovedemo sljedeće podatke: a=11, b=01, cin=0, rezultat će biti r=10 i cout=0. Naime, kombinacija 11 predstavlja znamenku 2, dok kombinacija 01 predstavlja znamenku 1. Stoga FA zapravo zbraja 2+1+0=3. U našem kodu 3 se prikazuje kombinacijom 10 pa se to pojavljuje na rezultatu. Također, prijenos je 0.

Sklop koji je na slici označen kao "(B-1)-komplement" je sklop koji na izlazu generira 3-komplement znamenke (prisjetimo se – radimo u bazi 4). Primjerice, u našem slučaju, ako bi na ulaz doveli kombinaciju 11, na izlazu bi dobili kombinaciju 01. Naime, kôd 11 odgovara znamenci 2. 3-komplement znamenke 2 je 1, što je kodirano s 01.

Sklop označen s *dmux* (oblika slova V) predstavlja dvostruki multipleksor. To je sklop koji se konceptualno može gledati kao spoj dva multipleksora koji dijele upravljački (selekcijski) ulaz. Konkretno, kada je s=0, na izlaze se propuštaju bitovi označeni s *x* a kada je s=1, na izlaze se propuštaju bitovi označeni s *y*. Ovaj nam sklop treba kako bismo na ulaz potpunog zbrajala ovisno o

ulazu *oper* doveli ili direktno drugi broj, ili njegov 3-komplement (ovisno o tome radimo li zbrajanje ili oduzimanje). Pri tome, ako je oper=0, obavlja se zbrajanje, a ako je oper=1, obavlja se oduzimanje.

Svaki od ovih sklopova može se projektirati kao klasični kombinacijski sklop – koristeći tablicu istinitosti za svaki izlaz i minimizacijom. Jednom kada imamo na raspolaganju ovakav *primitiv*, možemo koristiti nadomjesni simbol prikazan u nastavku.



Povezivanjem ovakvih primitiva jednostavno se može izgraditi višeznamenkasto zbrajalo/oduzimalo. Evo nekoliko primjera za slučaj korištenog koda.

oper	***a***	***b***	***r***	***cout***
0	11100100	01000010	10100110	0
0	11100100	01001010	00000010	1
1	11100100	01000010	01100001	1
1	11100100	01001010	01110101	1

2. Zadatak

Zadatak ove vježbe je izgradnja 4-znamenkastog zbrajala/oduzimala brojeva u bazi 4, ovisno o upravljačkom ulazu *oper*. Ako je *oper*=0, potrebno je obaviti zbrajanje, a ako je *oper*=1, potrebno je obaviti oduzimanje. Da biste mogli riješiti zadatak, potrebno je još definirati način kodiranja znamenaka. Na kraju ovog dokumenta nalazi se tablica s propisanim kodiranjem znamenaka, ovisno o vašem JMBAG-u, kao i naputak kako otkriti koju tablicu kodiranja trebate koristiti.

2.1. Dodijeljeni način kodiranja

U tablicu u nastavku čitko prepišite Vama dodijeljeni kod.

Znamenka	***Kodna riječ***
0	
1	
2	
3	

2.2. Izrada potpunog zbrajala

Projektirajte sklop potpuno zbrajalo. Konkretno, promatrajte izlaze c_{out}, r_1 i r_0 kao funkcije od a_1, a_0, b_1, b_0 te c_{in} (lokalne oznake ulaza sklopa FA). Poslužite se tabličnim prikazom. Kako glasi minimalni oblik tih funkcija? Upišite ih u sljedeću tablicu.

$c_{out}(a_1,a_0,b_1,b_0,c_{in}) =$

$r_1(a_1,a_0,b_1,b_0,c_{in}) =$

$r_0(a_1,a_0,b_1,b_0,c_{in}) =$

U sustavu VHDLLab2 napišite ponašajni VHDL model ovog sklopa. Sklop nazovite *FA*. U sučelju koristite signale `a`, `b` (vektor od 1 do 0), `cin` (skalar), `r` (vektor od 1 do 0) te `cout` (skalar) – upravo tim redoslijedom. Svi izlazi trebaju kasniti 10 ns.

2.3. Izrada sklopa za izračun (B-1)-komplementa

Projektirajte sklop koji računa 3-komplement. Ulaze sklopa označite s x_1 i x_0, izlaze s y_1 i y_0. Izlaze promatrajte kao funkcije od ulaza, i poslužite se tabličnim prikazom. Kako glase minimalni oblici izlaznih funkcija? Upišite ih u tablicu u nastavku.

$y_1(x_1,x_0) =$

$y_0(x_1,x_0) =$

U sustavu VHDLLab2 napišite ponašajni VHDL model ovog sklopa. Sklop nazovite *b1kompl*. U sučelju koristite signal x (vektor od 1 do 0) te y (vektor od 1 do 0) – upravo tim redoslijedom. Svi izlazi trebaju kasniti 10 ns.

2.4. Izrada dvostrukog multipleksora

Projektirajte dvostruki multipleksor. Podatkovne ulaze sklopa označite s x_1, x_0, y_1 te y_0, selekcijski ulaz sa s, a izlaze sa z_1 i z_0. Izlaze promatrajte kao funkcije od ulaza, i poslužite se tabličnim prikazom. Kako glase minimalni oblici izlaznih funkcija? Upišite ih u tablicu u nastavku.

$z_1(x_1,x_0,y_1,y_0,s) =$

$z_0(x_1,x_0,y_1,y_0,s) =$

U sustavu VHDLLab2 napišite ponašajni VHDL model ovog sklopa. Sklop nazovite *dmux*. U sučelju koristite signal x (vektor od 1 do 0), y (vektor od 1 do 0), s (skalar) te z (vektor od 1 do 0) – upravo tim redoslijedom. Svi izlazi trebaju kasniti 10 ns.

2.5. Izrada primitiva

U sustavu VHDLLab2 napišite strukturni VHDL model sklopa *primitiv*. Napišite strukturni model koristeći se prethodno razvijenim sklopovima.

2.6. Izrada 4-znamenkastog zbrajala/oduzimala

Koristeći **pojednostavljeni simbol primitiva** prikazan na početku upute, u prazno polje u nastavku nacrtajte shemu 4-znamenkastog zbrajala/oduzimala (gdje je točna operacija definirana ulazom *oper*). Na shemi jasno označite nazive primjeraka primitiva koje ćete koristiti pri strukturnom modeliranju, kao i nazive internih signala koje ćete koristiti.

U sustavu VHDLLab2 napišite strukturni VHDL model ovog sklopa. Sklop nazovite *zbrajalo*. U sučelju koristite signal a (vektor od 7 do 0), b (vektor od 7 do 0), oper (skalar) te r (vektor od 7 do 0) i cout (skalar) – upravo tim redosljedom. Uočite da ovaj sklop nema ulaz c_{in}. – zašto? Razmislite!

Prije dolaska na laboratorijske vježbe potrebno je riješiti sve prethodno navedene zadatke (projektiranje sklopova te njihov opis u jeziku VHDL), kao i popuniti tablicu u poglavlju 2.8. Na laboratorijskim vježbama provjerite još i sljedeće:

- prilikom zbrajanja brojeva, u kojem slučaju dolazi do najdužeg kašnjenja do pojave ispravnog rezultata (tzv. vrijeme stabilizacije rezultata)?
- koliko iznosi to vrijeme?

2.7. Priprema

Na samu vježbu potrebno je donijeti ovu uputu u kojoj sva polja za unos rješenja trebaju biti popunjena. Modeli traženih sklopova opisanih jezikom VHDL moraju biti uneseni u sustavu VHDLLab2. Sa sobom morate donijeti i papire korištene tijekom projektiranja samih sklopova (gdje su vidljive tražene tablice i sam postupak minimizacije – predlažemo K-tablice umjesto metode QMC). Sva rješenja trebaju biti napisana rukom, običnom ili kemijskom olovkom. Na vrhu svake stranice potrebno je kemijskom olovkom napisati prezime, ime te matični broj studenta.

2.8. Provjera rada sklopa

Za Vama dodijeljeni kod, što će biti na izlazu 4-znamenkastog zbrajala/oduzimala za ulaze iz sljedeće tablice? **Ovo popunite tablicu <u>prije dolaska</u> na vježbu** i to s rezultatima koje očekujete da će sklop generirati – ne s rezultatima koje je dala simulacija rada Vašeg sklopa. Jednom kada ste to popunili i kada imate na papiru napisan postupak dolaska do rješenja, tablicu iskoristite kako biste provjerili radi li Vam sklop dobro.

oper	***a***	***b***	***r***	***cout***
0	11100100	01000010		
0	11100100	01001010		
1	11100100	01000010		
1	11100100	01001010		

2.9. Pitanja za vježbu

1. Kako izgleda sklop *primitiv* ako se radi o zbrajanju/oduzimanju binarnih brojeva (dakle, kada je B=2)? U što tada degenerira multipleksor i sklop za izračun komplementa?

2. Ako razmatramo znamenke baze 10, konkretno, BCD kod, kako tada možemo ostvariti sklop FA? Isplati li se tada sklop projektirati tablično ili imamo bolje rješenje? Kako tada izgleda sklop za izračun 9-komplementa?

3. Ako razmatramo znamenke baze 10, konkretno, Excess-3 kod, kako tada možemo ostvariti sklop FA? Isplati li se tada sklop projektirati tablično ili imamo bolje rješenje? Kako tada izgleda sklop za izračun 9-komplementa?

4. Usporedite rješenja 2. i 3. zadatka. Što daje jednostavnije sklopovlje?

Literatura:

[1] Peruško, Glavinić: *Digitalni sustavi*. Školska knjiga, 2005.

[2] Čupić, *Digitalna elektronika i Digitalna logika. Zbirka riješenih zadataka*. Kigen, 2006.

3. Dodijeljene tablice kodova

Predzadnje dvije znamenke vašeg JMBAG-a modulo 20 određuju koju tablicu kodiranja znamenaka koristite. Primjerice, student s JMBAG-om 0012345**67**8 računa: N = 67 % 20 = 7, te odabire tablicu uz koju stoji N=7.

N=0

Znamenka	Kodna riječ
0	00
1	10
2	01
3	11

N=1

Znamenka	Kodna riječ
0	00
1	10
2	11
3	01

N=2

Znamenka	Kodna riječ
0	00
1	11
2	01
3	10

N=3

Znamenka	Kodna riječ
0	00
1	11
2	10
3	01

N=4

Znamenka	Kodna riječ
0	01
1	00
2	10
3	11

N=5

Znamenka	Kodna riječ
0	01
1	00
2	11
3	10

N=6

Znamenka	Kodna riječ
0	01
1	10
2	00
3	11

N=7

Znamenka	Kodna riječ
0	01
1	10
2	11
3	00

N=8

Znamenka	Kodna riječ
0	01
1	11
2	00
3	10

N=9

Znamenka	Kodna riječ
0	01
1	11
2	10
3	00

N=10

Znamenka	Kodna riječ
0	10
1	00
2	01
3	11

N=11

Znamenka	Kodna riječ
0	10
1	00
2	11
3	01

N=12

Znamenka	Kodna riječ
0	10
1	01
2	00
3	11

N=13

Znamenka	Kodna riječ
0	10
1	01
2	11
3	00

N=14

Znamenka	Kodna riječ
0	10
1	11
2	00
3	01

N=15

Znamenka	Kodna riječ
0	10
1	11
2	01
3	00

N=16

Znamenka	Kodna riječ
0	11
1	00
2	01
3	10

N=17

Znamenka	Kodna riječ
0	11
1	00
2	10
3	01

N=18

Znamenka	Kodna riječ
0	11
1	01
2	00
3	10

N=19

Znamenka	Kodna riječ
0	11
1	01
2	10
3	00