

# **IR4VLS SI4VLS, Projektni zadatak 2013/2014**

## **Verzija dokumenta 1.2**

Pre pristupanja izradi projekta pročitati dati tekst u celini. Sve što nije precizirano u tekstu zadatka ostavlja se studentima da definišu i obrazlože svoj izbor. Ukoliko su postavljeni kontradiktorni zahtevi, od studenata se očekuje da uvedu RAZUMNU pretpostavku, jasno je obrazlože u dokumentaciji i nastave da na njoj izgrađuju preostali deo rešenja. Na kraju teksta data su pravila vezana za izradu projekta.

Stil kodiranja može da utiče na konačan broj poena. Stil kodiranja se odnosi na to koliko je kod čitljiv i stepen moguće reupotrebe. Čitljivost koda se ogleda u formatiranju teksta, deskriptivnim identifikatorima i komentarima. Stepenn moguće reupotrebe se ocenjuje na osnovu dekompozicije koda, korišćenje konstanti i generičkih parametara. Svaki drugi faktor koji utiče na čitljivost koda ili stepenn reupotrebe koda može da utiče na ukupan broj osvojenih poena.

Projektni zadatak se sastoji od dva nezavisna dela. Prvi deo nosi maksimalno 30 poena i radi se u FPGA tehnologiji. Drugi deo nosi maksimalno 10 poena i radi se u Maxeler tehnologiji.

Projekat se radi u grupama od jednog ili dva studenta. Izuzetno, uz saglasnost predmetnog asistenta, projekat može da se radi u grupi od tri studenta, uz dodatne zahteve. Dodatni zahtevi su specifikirani u postavci.

### **Prvi deo (30 poena)**

Grupa od tri studenta realizuje ceo sistem. Manje grupe realizuju samo jedno jezgro i ne treba da realizuju sinhronizaciju (RMW instrukcije) i keš koherenciju.

Potrebno je isprojektovati 32-bitni procesor opšte namene. Procesor se sastoji od dva indentična jezgra. Svako jezgro ima privatnu L1 keš memoriju.

Interfejs procesora prema okolini su magistrala, RESET signal i signal kloka. Prilikom startovanja sistema RESET signal se postavlja na aktivnu vrednost i nakon nekog vremena se vraća na neaktivnu vrednost. Smatrati da je trajanje RESET signala dovoljno da se ceo sistem resetuje. Magistrala se sastoji od 32 adresne linije, 32 linije podataka i nekoliko kontrolnih linija. Smatrati da pristup memoriji traje 13 signala takta (jedan takt za postavljanje zahteva, jedanaest za dohvaćanje podataka i jedan takt za vraćanje rezultata). Upis u memoriju nema fazu za vraćanje rezultata. Memorija može istovremeno da opslužuje maksimalno 12 zahteva. Adresibilna jedinica je reč. Reč je veličine 4 bajta.

Jezgro je RISC arhitektura sa sledećim osobinama:

- registarski fajl

- load/store arhitektura sa RMW instrukcijama
- adresiranje je bazirano na vrednostima u registrima
- uniformna instrukcijska reč, veličine 32 bita

Jezgro sadrži protočnu obradu. Broj stepeni je proizvoljan ali mora biti veći od 2. Sve hazarde je potrebno hardverski razrešiti. Voditi računa o performansama, drugim rečima koristiti tehnike za ubrzanje protočne obrade kao što su prosleđivanje, deljenje keš memorije na instrukcijski deo i deo za podatke, prosleđivanje zahteva memoriji iako već memorija obrađuje neke zahteve. Smatrati da L1 keš memorija odgovara na zahtev u jednom taktu. Veličina bloka u keš memoriji je 4 reči. Upis u keš memoriju je write back (odložen upis). L1 keš memorija ne mora da bude veća od 4KB. Deo keš memorije u koji se smeštaju reči iz memorije mora da bude sintetizovan u memorijskim blokovima na čipu. Mora da važi princip inkluzije u memorijskoj hijerarhiji. Protokol za keš koherenciju je Snoopy (bilo koja varijanta). Sinhronizacija između jezgara treba da bude minimalna, tj. blokiranje jednog jezgra treba da se desi samo kada oba jezgra pristupaju istoj memorijskoj lokaciji i drugo jezgro pristupa nedeljivo.

Jezgro ima 16 registara opšte namene širine 32-bita. Registri su obeleženi sa R0 – R15. Registar R13 je softverski pokazivač na stek (SP). Registar R14 je link registar. Registar R15 je pokazivač na sledeću instrukciju (PC). Registar specijalne namene je statusni registar CSR. Instrukcijski set dat je u sledećem poglavlju.

Statusni registar ima format prikazan na Slika 1. Značenje pojedinih bitova dato je u Tabela 1. Bitove za uslov skoka setuju instrukcije poređenja i druge aritmetičke instrukcije.

U slučaju greške (npr. nepostojeća instrukcija) izvršiti instrukciju stop za jezgro u kom se desila greška.

|     |    |    |    |    |          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
|-----|----|----|----|----|----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
|     | 31 | 30 | 29 | 28 | 27       | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CSR | N  | Z  | C  | V  | Reserved |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |

Slika 1 – CSR registar

| Bit | Značenje                        |
|-----|---------------------------------|
| N   | Rezultat je negativan           |
| Z   | Rezultat je nula                |
| C   | Postoji prenos u sledeći razred |
| V   | Došlo je do prekoračenja        |

Tabela 1 – Bitovi CSR registra

Kao rešenje potrebno je priložiti sledeće:

1. VHDL kod koji opisuje dati procesor
2. Testove i VHDL kod testbench-a
3. Dokumentacija (template dokumentacije se nalazi na sajtu predmeta)

Na odbrani se očekuje sledeće:

1. Da se opis datog procesora može sintetizovati i da sintetizovani model ima sve tražene funkcionalnosti
2. Prikaz simulacije studentskih testova
3. Prikaz simulacije testova koje obezbedi predmetni asistent
4. Ispitivanje u vezi realizacije procesora

Jedan test se sastoji iz dva tekstualna fajla. Jedan fajl inicijalizuje memoriju. Drugi fajl sadrži očekivano stanje memorije podataka nakon izvršavanja programa. Fajl sadrži vrednosti memorijskih lokacija. Format fajla je sledeći: u jednoj liniji nalazi se adresa zapisana kao heksadecimalni broj i vrednost lokacije koja se nalazi na toj adresi zapisana kao binarni broj. Ulazni fajl na početku sadrži dve adrese, zapisane kao heksadecimalni broj, koja će biti smeštena na početku simulacije u PC registre jezgara. Potrebno je dostaviti sledeće tipove testova: test svih instrukcija, test izvršavanja proizvoljnog programa, test koji testira sve moguće kombinacije instrukcija koje izazivaju hazarde. Kada se procesor zaustavi potrebno je uporediti stanje memorije podataka i vrednosti iz fajla koji sadrži očekivano stanje memorije podataka. U slučaju nepodudaranja prijaviti grešku. Za potrebe simuliranja i testiranja napisati kod za memoriju (ne treba da se vrši sinteza memorije).

### Instrukcijski set

Procesor ima 6 tipova instrukcija prikazanih na Slika 2. Tabela 2 prikazuje skraćenice tipova instrukcija. Dužina instrukcije je fiksna i iznosi 32 bita.

|      | 31 | 30 | 29 | 28       | 27 | 26            | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17       | 16        | 15 | 14 | 13 | 12       | 11 | 10 | 9     | 8        | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|----|----|----|----------|----|---------------|----|----|----|----|----|----|----|----|----------|-----------|----|----|----|----------|----|----|-------|----------|---|---|---|---|---|---|---|---|
| DP R | 0  | 0  | 0  | opcode   |    |               |    |    | Rn |    |    | Rd |    |    | S        | Rm        |    |    |    | Rs       |    | F  | shift | Reserved |   |   |   |   |   |   |   |   |
| DP I | 0  | 0  | 1  | opcode   |    |               |    |    | Rn |    |    | Rd |    |    | S        | immediate |    |    |    |          |    |    |       |          |   |   |   |   |   |   |   |   |
| L/S  | 0  | 1  | 0  | L        |    |               |    |    | Rn |    |    | Rd |    |    | Reserved |           |    |    |    |          |    |    |       |          |   |   |   |   |   |   |   |   |
| RMW  | 0  | 1  | 1  | opcode   |    |               |    |    | Rn |    |    | Rd |    |    | S        | Rm        |    |    |    | Reserved |    |    |       |          |   |   |   |   |   |   |   |   |
| B/BL | 1  | 0  | 0  | cond     | L  | 26-bit offset |    |    |    |    |    |    |    |    |          |           |    |    |    |          |    |    |       |          |   |   |   |   |   |   |   |   |
| S    | 1  | 0  | 1  | Reserved |    |               |    |    |    |    |    |    |    |    |          |           |    |    |    |          |    |    |       |          |   |   |   |   |   |   |   |   |

**Slika 2** - Tipovi instrukcija

| Skraćenica tipa instrukcije | Tip instrukcije                            |
|-----------------------------|--|
| DP R                        | Obrada podataka sa registarskim operandima |
| DP I                        | Obrada podataka sa neposrednom vrednošću   |
| L/S R                       | Load/Store                                 |
| RWM                         | Read Modify Write                          |
| B/BL                        | Instrukcije skoka                          |
| S                           | Stop instrukcija                           |

**Tabela 2** – Tabela skraćenica tipova instrukcija

| cond | Skraćenica | Značenje       |
|------|------------|----------------|
| 00   | EQ         | Jednako        |
| 01   | GT         | Označen veći   |
| 10   | HI         | Neoznačen veći |
| 11   | AL         | Bezuslovno     |

**Tabela 3** – Uslovni kodovi

## Instrukcije za obradu podataka

|      | 31 | 30 | 29 | 28     | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15        | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6     | 5        | 4 | 3 | 2 | 1 | 0 |
|------|----|----|----|--------|----|----|----|----|----|----|----|----|----|----|----|----|-----------|----|----|----|----|----|---|---|---|-------|----------|---|---|---|---|---|
| DP R | 0  | 0  | 0  | opcode |    |    |    | Rn |    |    |    | Rd |    |    |    | S  | Rm        |    |    |    | Rs |    |   |   | F | shift | Reserved |   |   |   |   |   |
| DP I | 0  | 0  | 1  | opcode |    |    |    | Rn |    |    |    | Rd |    |    |    | S  | immediate |    |    |    |    |    |   |   |   |       |          |   |   |   |   |   |

Postoje dva tipa instrukcija za obradu podataka. Tip instrukcije određuje šta će biti drugi operand. Instrukcija za obradu podataka postavlja bitove u CSR registru. Tabela 4 prikazuje kodove operacija za različite instrukcije. Instrukcije koje upisuju u PC registar vrše običan skok. Instrukcije za premeštanje podataka imaju samo drugi operand. Ostale instrukcije imaju dva operanda. Rn specificira prvi operand. Rm specificira drugi registarski operand. Immediate specificira drugi neposredni operand. Rd specificira odredište. Instrukcija SWAP postoji samo u prvom tipu instrukcije i menja vrednosti registara Rd i Rm.

| opcode | Skraćenica | Instrukcija                 |
|--------|------------|-----------------------------|
| 0000   | AND        | Logičko I                   |
| 0010   | SUB        | Oduzimanje                  |
| 0100   | ADD        | Sabiranje                   |
| 0101   | ADC        | Sabiranje sa bitom prenosa  |
| 0110   | SBC        | Oduzimanje sa bitom prenosa |
| 1000   | SWAP       | Zamena vrednosti registara  |
| 1010   | CMP        | Upoređivanje                |
| 1101   | MOV        | Premeštanje drugog operanda |
| 1111   | NOT        | Logičko ne                  |

**Tabela 4** – Instrukcije za obradu podataka

U slučaju instrukcije sa registarskim operandima prvi operand je vrednost iz registra Rn pomerena ili rotirana za onoliko mesta kolika je vrednost registra Rs. Tip pomeranja ili rotiranja se određuje na osnovu shift bitova, prikazanih u Tabela 5. Bit F određuje da li se radi o pomeranju (vrednost 0) ili o rotiranju (vrednost 1). U slučaju instrukcije sa neposrednim podatkom drugi operand je immediate vrednost koja se proširuje na 32 bita kao neoznačen broj u slučaju da je vrednost S bita 0 ili kao označen broj u slučaju da je vrednost S bita 1. Bit S određuje da li se aritmetičke operacije rade nad neoznačenim brojevima (vrednost 0) ili nad označenim brojevima (vrednost 1).

| shift | Skraćenica | Tip                                    |
|-------|------------|--|
| 00    | LL         | Logičko pomeranje/rotiranje ulevo      |
| 01    | LSR        | Logičko pomeranje/rotiranje udesno     |
| 10    | ASR        | Aritmetičko pomeranje/rotiranje udesno |
| 11    | -          | Nema pomeranja/rotiranja               |

**Tabela 5 – Pomeračke operacije**

### Load/Store instrukcije

|     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |          |
|-----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|----------|
|     | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0        |
| L/S | 0  | 1  | 0  | L  |    |    |    |    | Rn |    |    |    |    | Rd |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   | Reserved |

Load/Store (L/S) instrukcije mogu da pristupaju rečima u memoriji. Adresa podatka kojem se pristupa je u registru Rn. Prilikom Load instrukcije podatak se smešta u Rd, a prilikom Store instrukcije podatak iz Rd se smešta u memoriju. Bit L određuje da li je instrukcija Load (L = 1) ili Store (L = 0).

### Read Modify Write instrukcije

|     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |          |
|-----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|----------|
|     | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0        |
| RMW | 0  | 1  | 1  |    |    |    |    |    |    |    | Rn |    |    |    |    | Rd | S  |    |    |    |    |    |   |   |   |   |   |   |   |   |   | Reserved |

Read Modify Write (RMW) instrukcije omogućavaju atomičnu izmenu memorijskih lokacija. Operacija učitava podatak iz memorije sa adrese koja je smeštena u registru Rn. Podatak smesti u registar Rd. Izvrši operaciju koja je data poljem opcode (Tabela 4). Prvi operand je registar Rd, drugi operand je registar Rm. Odrediste je memorijska lokacija čija se adresa nalazi u registru Rn. Bit S određuje da li se aritmetičke operacije rade nad neoznačenim brojevima (vrednost 0) ili nad označenim brojevima (vrednost 1).

### Istrukcije skoka

|      |    |    |    |      |    |               |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
|------|----|----|----|------|----|---------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
|      | 31 | 30 | 29 | 28   | 27 | 26            | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| B/BL | 1  | 0  | 0  | cond | L  | 26-bit offset |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |

Postoje dve instrukcije skoka. Bit L određuje da li se radi o običnom skoku Branch (L = 0) ili o skoku sa pamćenjem povratne adrese u registru R14 Branch and Link (L = 1). Adresa skoka se dobija sabiranjem tekuće vrednosti PC registra (adresa instrukcije skoka plus 1) i proširene označene vrednosti 26-bitnog offset-a na 32 bita. Uslov skoka je određen na osnovu bitova cond, prikazanih u Tabela 3.

### Stop instrukcija

|   |    |    |    |          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
|---|----|----|----|----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
|   | 31 | 30 | 29 | 28       | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| S | 1  | 0  | 1  | Reserved |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |

Stop instrukcija zaprljane blokove iz keš memorije jezgra vraća u memoriju i zaustavlja jezgro.

## Drugi deo (10 poena)

Sve grupe realizuju ceo sistem.

Implementirati program u Maxeler tehnologiji koji vrši aproksimaciju funkcije  $y$ .

Funkcija pomoću koje se vrši aproksimacija je:

$$P(x) = c_0 + c_1x + c_1x^2 + \dots + c_{m-1}x^{m-1} + c_mx^m.$$

Funkcija  $y$  se aproksimira u  $n$  tačaka po formuli:

$$y(i) = P(x(i)).$$

Aproksimacija može da se dobije rešavanjem matrične jednačine:

$$\begin{bmatrix} 1 & x_1^1 & x_1^2 & \dots & x_1^m \\ 1 & x_2^1 & x_2^2 & \dots & x_2^m \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_n^1 & x_n^2 & \dots & x_n^m \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_m \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix}.$$

Ulazni test fajl je tekstualni. Sastoji se od proizvoljnog broja aproksimacija. Na početku fajla se nalazi broj aproksimacija. Svaka aproksimacija je specificirana u tri reda. Prvi red sadrži brojeve  $n$  i  $m$ . Drugi red se sastoji od  $m$   $c$  koeficijenata. Treći red sadrži  $n$   $x$  vrednosti.

Maxeler program treba da radi sa proizvoljnim vrednostima  $m$  i  $n$ , bez potrebe za rekompilacijom kernela. Implementacija treba da bude maksimalno efikasna.

Potrebno je dostaviti sledeće:

1. Java Maxeler kod kernela, menadžera, simulatora
2. C kod programa koji učitava podatke iz ulaznog test fajla, pokreće izračunavanje u kernelu i ispisuje rezultate u izlazni tekstualni fajl.
3. Test primere i očekivane rezultate
4. Grafički prikaz kernela u pdf formatu (prikaz ne sme biti automatski generisan)

Na odbrani se očekuje sledeće:

1. Simulacija rešenja
2. Ispitivanje u vezi implementacije programa

| Verzija | Izmene   |
|---------|--|
| 1.1     | Navedeno je koliko traje upis u memoriju.<br>R14 je link registar u delu za instrukcije skoka.<br>Izbačen je zahtev za test prekida. |
| 1.2     | Izmenjen format ulaznog fajla za Maxeler deo.<br>Ispravljena formula za aproksimaciju ( $y_m \rightarrow y_n$ ).                     |

## **Pravila za izradu projekta 2013/2014**

Poeni sa projekta dobijeni pre školske godine 2013/2014 važe do kraja školske godine 2013/2014. Poeni sa projekta dobijeni u toku školske godine 2013/2014 važe do kraja školske godine 2013/2014. Na odbrani projekta moguće je osvojiti od 0 do 40 poena. Prvi deo se vrednuje sa maksimalno 30 poena. Drugi deo se vrednuje sa maksimalno 10 poena. Može da se brani samo jedan deo. Projekat može da se brani samo jednom u toku tekuće školske godine. Projekat se brani pred predmetnim asistentom. Odbrana projektnog zadatka nije uslov za izlazak na ispit. Okvirni termini odbrane projekta su:

1. 5 radnih dana pre januarnskog ispitnog roka ili u toku januarnskog ispitnog roka,
2. U toku letnjeg semestra, ili
3. 5 radnih dana pre septembarskog ispitnog roka ili u toku septembarskog ispitnog roka.

Tačan termin za odbranu projekta pod stavkom 1. i 3. biće objavljeni najkasnije nedelju dana pre odbrane. Tačan termin za odbranu projekta pod stavkom 2. biće određen u dogovoru sa studentskim predstavnicima, da bi se izbegle nedelje u kojima studenti masovno odsustvuju (elektrijada, ekskurzije, itd.). Tačan termin odbrane će biti isključivo u toku nedelje u kojima se održava nastava iz letnjeg semestra. Prijave za odbranu će se primati u roku od tri dana nakon objave tačnog termina. Uz prijavu potrebno je predati ceo projekat, zajedno sa dokumentacijom. Način prijave i predaje biće specificiran zajedno sa objavom tačnog termina.

Projekat se izrađuje u grupama od jednog ili dva studenta ili tri studenta. Svaki student mora da zna sve specifičnosti rešenja. Prijava grupa za izradu projekta se vrši do 15. decembra tekuće školske godine, na način koji će biti objavljen do 15. novembra tekuće školske godine. Ukoliko student ne prijavi grupu za izradu projekta smatra se da je odustao od izrade projekta u toku tekuće školske godine. Zakasnele prijave se neće naknadno prihvatati.

Nedozvoljena saradnja se smatra uzimanjem celokupnog rešenja ili dela rešenja jedne grupe od druge grupe. U slučaju da se utvrdi nedozvoljena saradnja dve ili više grupa, svim studentima, iz tih grupa koji do tog trenutka nisu položili ispit, biće anuliran broj poena osvojenih na projektu za tekuću školsku godinu.