Inspiratron.org - Natural language processing, machine learning and cybersecurity

DLX procesor sa 5 stepena pipeline-a (VLSI sistemi projekat ETF Beograd 2010 deo 1)

by Nikola Miloševi? - Monday, January 18, 2010

https://inspiratron.org/blog/2010/01/18/dlx-procesor-sa-5-stepena-pipeline-a-vlsi-sistemi-projekat-etf-beograd-2010-deo-1/

DLX procesor
sa 5 stepena pipeline-a

Nikola Vidosavljevi?, Nikola Miloševi?

1. Definisanje projekta

1.1 Uvod

Projekat iz VLSI (Very Large Scaled Integration) nam služi da savladamo osnovne elemente jezika VHDL (VLSI Hardware Language) i projektovanjem sopstvenog procesora upoznamo detalje njegove arhitekture koja je vrlo bliska sa arhitekturom procesora koji se danas koriste, ali ipak dovoljno pojednostavljena da ju je mogu?e isprojektovati kao doma?i zadatak. Projekat je logi?an nastavak projekta iz predmeta AOR2 (Arhitektura i Organizacija Ra?unara 2) gde smo imali priliku da projektujemo procesor, ali vrlo apstraktno. Uz pomo? VHDL-a u obzir uzimamo sva kašnjenja koja elementi i provodnici unose, što nije bio slu?aj u projektu iz AOR-a 2. Tako?e, analiza i testiranje su konkretnije i jednostavnije uz pomo? VHDL-a.

1.2 Ciljevi projekta

Osim upoznavanja jezika VHDL i arhitekture procesora, ovaj projekat se može iskoristiti za testiranje izvršavanja instrukcija na opisanom procesoru. To je najoptimalnije iskoristiti u edukacione svrhe za objašnjavanje osnovnih stvari u arhitekturi ra?unara koje se mnogo lakše mogu savladati uz pomo? vizuelizacije problema na koje studenti mogu nai?i.

2. SpecifikacijA

2.1 Uvod

Isprojektovati i modelovati 16-bitni troadresni procesor koji treba da ima slede?e grupe instrukcija:

load i store:

-jedine instrukcije koje pristupaju

memoriji,

-transfer izme?u memorijske lokacije
zadate jednim registrom i drugog zadatog registra;

mov:

-transfer podataka iz registra u
registar,

-mogu?e zadati i rotiranje podatka
pri transferu sa 4-bitnim neozna?enim pomerajem u desno kao neposrednim
podatkom u instrukciji;

aritmeti?ke operacije sa 3 operanda:

-sabiranje,

-oduzimanje;

**logi?ke instrukcije po bitima sa 3
operanda:**

-i,

-ili,

-ekskluzivno ili,

-negacija;

Uslovni skokovi

-u zavisnosti od odnosa dva zadata
registra ska?e se na adresu zadatu tre?im registrom,

-podržati proveru: >, >=,
<, <=, = =, !=;

podrška za potprograme:

-poziv potprograma (adresa se pamti
na dole opisanom steku),

-povratak iz potprograma (adresa se
skida sa dole opisanog steka);

funkcije za rad sa stekom:

-push

-pop

-transfer podataka sa steka
relativno u odnosu na vrh steka (ozna?eni pomjeraj je dio instrukcije, 8 bita)
u zadati registar i obrnuto.

Procesor treba da poseduje
registarski fajl koji se sastoji od 16 registara. Adresni prostor procesora je 216
16-bitnih re?i (adresibilna jedinica re?).

Za potrebe poziva potprograma,
procesor poseduje hardverski stek (potpuno odvojen od operativne memorije)
veli?ine 256 re?i. U slu?aju prepunjavanja steka ili prekora?enja opsega pri
pristupu steku resetovati procesor.

Po resetu, sistem po?inje
izvršavanje od adrese 0.

Kao operand u bilo kojoj instrukciji
(uklju?uju?i i adresu u load i store), može da se pojavi samo registar, osim u
gore navedenim slu?ajevima kada je neki operand zadat kao neposredni podatak.

U sistemu postoje dva odvojena
adresna prostora, jedan za instrukcije i jedan za podatke. U skladu s tim,
postoji dvije 16-bitne adresne magistrale, dvije 16-bitne magistrale podataka,
kao i standardne linije, dve RD i jedna WR (instrukcije i podaci su odvojeni.
Ukoliko se neki od kontrolnih signala postavi do uzlazne ivice signala takta,
smatrati da ?e se odgovaraju?a operacija obaviti pre naredne uzlazne ivice
signala takta.

Ako je to mogu?e, veli?ina svih
instrukcija treba da budu jedna re?.

Sve instrukcije se izvršavaju kroz 5
stepeni pipeline-a (u skladu sa potrebama pojedinih instrukcija):

1. Dohvatanje
instrukcije,

2. Dekodovanje
instrukcije,

3. Dohvatanje
operanada,

4. Izvršavanje
instrukcije,

5. Upis
rezultata.

Neophodno
je detektovati sve hazarde na vreme i zaustaviti odgovaraju?e stepene
pipeline-a potreban broj perioda takta da bi se dobio ispravan rezultat
izvršavanja programa.

2.2 Spoljašnji interfejsi

Ure?aj sadrži 88 pinova, od kojih je 37 ulaznih i
51 izlazni. Ulazni su instrukcijska magistrala, magistrala podataka, takt,
reset procesora i linije koje nam pokazuju da li su eksterne memorije postavile
podatak na magistralu. Izlazni su adresne linije instrukcijske magistrale i
adresne linije memorijske magistrale. Izlazne linije su tako?e linije ctrl_read
i ctrl_write koje služe za zahteve upisa i ?itanja iz memorije. Postoji linija
inst_read koja je aktivna kada treba pro?itati podatak iz instrukcijske
memorije. Program counter postavlja vrednost na adresne linije instrukcijske magistale
i zahtev inst_read, pomo?u kojih se ?ita instrukcija. Tako?e procesor postavlja
adresne linije i liniju zahteva prilikom upisa ili ?itanja iz memorije, nakon
?ega ?eka signal function complete, koji postavlja memorija i tad skida podatak
sa magistale, na ivicu slede?eg takta.

2.3 Specifikacija hardvera

Pri projektovanju
procesora koriš?eni su slede?i elementi: sabira? (sa uslovnim sumama), I, ILI, NE i ekskluzivno ILI logi?ka kola, blok za rotaciju (barel), dekoder, D flip-flop, multiplekser, demultiplekser, registar, registarski file, sabira?, trostati?ki elementi i stek.

Procesor se može
podeliti na 5 stepena proto?ne obrade.

U prvom stepenu se
?ita instrukcija, tako što se pro?ita iz instrukcijske memeorije podatak koji se nalazi na adresi na koju ukazuje PC. PC se nakon toga inkrementira u sabira?u, i nakon toga u?itava. U Instruction fetch jedinici postoji i multiplekser koji ukoliko dodje do skoka, na ulaz PC propušta adresu skoka, umesto inkrementirane vrednosti.

U drugom stepenu se
dohvataju operandi instrukcija iz registarskog file-a, kako se svi operandi svih instrukcija moraju na?i u nekom od registara opšte namene. Registara opšte namene ima 16. U ovoj jedinici može do?i do hazarda podataka ukoliko neka kasnija instrukcija pokuša da ?ita podatak koji treba tek da se upiše i ?ija obrada se nalazi u nekom od stepena proto?ne obrade.

U Execution fazi se
izvršavaju instrukcije ra?unanja, jer se u njoj nalazi ALU jedinica. Tako?e ispred alu se nalaze multiplekseri koji odre?uju koji podatak ?e u?i u ALU jedinicu. U ovom bloku se odre?uje i da li ?e do?i do skoka na osnovu flagova koje daje ALU jedinica.

Postoji aritmeti?ko
logi?ka jedinica (ALU) koja izvršava izra?unavanja. Ulazi u ALU aedovode kao pro?itani registri iz registarskog file-a. Izlaz se vodi u EX/MEM blok. Sastoji se od sabira?a, I, ILI, ekskluzivno ILI i NOT logi?kih kola i bloka za rotaciju podatka. Izlazi ovih blokova su dovedeni na ulaz multipleksera koji pomo?u selekcionih ulaza u ALU blok bira jedan od njih i prosle?uje ga na izlaz. Postoji kombinaciona mreža sastavljena od I, ILI i NE logi?kih elemenata koja izra?unava statusne C, V, Z i N bitove. Kao sabira? je koriš?en sabira? sa uslovnim sumama. On je odabran jer predstavlja optimum izme?u brzine složenosti realizacije i brzine izvršavanja. Može se koristiti sabira? po bitima (Ripple Carry) koji je jednostavniji, ali znatno sporiji ili Carry Select Adder koji je neznatno brži ali koristi multipleksere sa ve?im brojem bitova, tako da mu je struktura složenija.

Nardna faza služi za
upis u memoriju. U ovoj jedinici se generišu signali zahteva ?itanja iz memorije i pisanja u memoriju. Prilikom upisa se postavljaju vrednosti na adresnu magistralu i magistalu podataka, dok se prilikom ?itanja postavi adresa i ?eka se podatak. U ovoj jedinici se nalazi i stek, na koji se upisuje pomo?u instrukcija push ili prilikom jsr instrukcije se upisuje PC na stek. Takodje pop instrukcija ?ita sa steka, ali i rti instrukcija ?ita PC sa steka.

Poslednja Write back
jedinica služi za vra?anje podatka u i upis u registarski file. U ovoj jedinici postoji multiplekser koji odre?uje da li podatak pro?itan iz memorije se upisuje u registarski file ili je to podatak koji je dobijen nekom od operacija u EX jedinici.

2.4 Specifikacija softvera

U sabira?u se koristi algoritam sabiranja sa
uslovnim sumama. Algoritam se sastoji u tome da se sabiranje bitova izvršava
istovremeno, pa se kasnije na osnovu prenosa iz ranijih razreda vrši selekcija
ve? izra?unatih suma. U prvom koraku se vrši sabiranje pojedina?nih razreda, a
u narednim koracima se vrši grupisanje tako što se grupa pove?ava na dvostruko
ve?u od one u prethodnom koraku. Tako dolazimo da se sabiranje u našem slu?aju
sa 16-bitnim sabira?em sabiranje vrši u 6 koraka (jedan za ra?unanje po?etnih
suma i 5 koraka za selekciju uslovnih suma).

Algoritam se može iskazati slede?im formulama:

Rezultat kada je
prenos ulazni prenos 0:
$S_i^0 = A_i \oplus B_i = (A_i + B_i)\,(A_i + B_i) = (A_i + B_i)\,(A_i B_i)$,
$C_i^0 = A_i B_i$

Rezultat kada je
prenos ulazni prenos 1:
$S_i^1 = A_i \oplus B_i \oplus 1 = A_i \oplus B_i = S_i^0$,
$C_i^1 = A_i B_i + (A_i + B_i) = A_i + B_i$.

2.5 Interfejs izme?u softvera i hardvera

3. Unutrašnji blokovi

Subblokovi procesora:

Registarski fajl sa logikom za ?itanje i upis u
registarski fajl.

Hardveski stek, sa logikom koja služi za upis i
?itanje sa steka, kao i logikom za odlu?ivanje koji podatak treba upisati na
stek.

Aritmeti?ko-logi?ki blok sastavlje od ALU jedinice
na ciji ulaz su dovedeni prihvatni registri operanada iz ID/EX jedinice.

IF/ID blok za prenos PC i IR iz Instruction feth
stepena u Instruction decode stepen.

ID/EX blok za prenos 3 regista opšte namene (dva
su operandi, dok se tre?i koristi u instrukcijama store kao operand koji se
smešta u memoriju i IR.

Zero blok koji detektuje na osnovu logike da li je
dožlo do skoka.

EX/MEM blok za prosle?ivanje u MEM jedinicu
rezultata ALU jedinice, 2 registra, kao i bita koji odre?uje da li je došlo do
skoka i IR registra .

MEM/WB blok za prosle?ivanje u Write back blok IR
registra i podataka koji treba da se smeste u registarski fajl.