**1. (Od roka do roka se jedino menja to što ne pita uvek za sve registre, ali dole je dat odgovor za svaki)**

**У процесору одређене архитектуре и организације од регистара постоје и регистри опште намене (GPR), програмска статусна реч (PSW), указивач на врх стека (SP), помоћни регистри B1 и B2, меморијски адресни регистар (MAR) и меморијски регистар податка (MDR). Навести и објаснити за сваки од ових регистара да ли припада архитектури или организацији процесора.**

Arhitekturi procesora pripadaju oni registri koji su namenjeni da se u njih nekom od instrukcija upiše neka vrednost, pa da se ta vrednost nekim kasnijim instrukcijama čita.

To su registri: PC, SP, ACC, GPR, IVTP, PSW

Organizaciji računara pripadaju oni registri procesora koje projektant procesora ubacuje da bi prema nekom svom pristupu projektovanja procesora obezbedio čuvanje neophodnih sadržaja prilikom prolaska kroz sve korake iz kojih se sastoji izvršavanje jedne instrukcije. Upisivanje vrednosti u ove registre i čitanje vrednosti iz ovih registara je nemoguće specificirati instrukcijama. To je određeno usvojenim algoritmima izvršavanja svake instrukcije posebno.

To su registri: MAR, MBR, IR, Pomocni (B1 i B2)

 **2. У процесору постоји инструкција вишеструког грањања CASE.**

**a. Дати формат инструкције и навести шта се сваким од операнада у формату инструкције специфицира.**

**b. Објаснити како се извршава инструкција у свим ситуацијама које могу да настану.**

a)

CASE selector, base, limit, displ[0], displ[1],... displ[limit-base]

Instrukcija CASE realizuje relativni skok sa pomerajem displ[selector-base] u odnosu na tekuću vrednost programskog brojača PC pod uslovom da je vrednosti parametara selector veća ili jednaka vrednosti parametra base i manja ili jednaka vrednosti parametra limit. U suprotnom slučaju se prelazi na sledeću instrukciju.

b)

Izvršavanje instrukcije CASE se može predstaviti na sledeći način:

if ((selector geq base ) and (selector leq limit ))

then PC <= PC + displ[selector-base]

else PC <= PC + (limit-base+1)

**3. (Može da se desi da bude više ulaznih/izlaznih uređaja, ali se na istu foru radi)**

**У посматраном рачунарском систему постоји само једна линија по којој улазно/излазни уређаји могу процесору да шаљу захтеве за прекид, док линија за слање сигнала потврде не постоји. У датом систему постоје 4 улазно/излазна уређаја UI3, UI2, UI1 и UI0, при чему приоритет ових уређаја опада од UI3, који има највиши приоритет, преко UI2 и UI1 до UI0, који има најнижи приоритет. Нацртати како та 4 улазно/излазна уређаја треба повезати помоћу те линије на процесор ради слања захтева за прекид и објаснити како се реализује скок на одговарајућу прекидну рутину сваког од 4 улазно/излазна уређаја. Скок на одговарајућу прекидну рутину се реализује делом хардверски, а делом софтверски. Објаснити шта се и како ради хардверски, а шта и како софтверски. За онај део који се ради софтверски написати одговарајући програм. Узети да се адресе прекидних рутина уређаја UI3, UI2, UI1 и UI0 налазе у улазима 0, 1, 2 и 3, респективно, посебно формиране табеле, чија се почетна адреса налази у меморијској локацији означеној са *tabadresa*.**

Jedna linija za zahtev, kombinovano softversko i hardversko utvrđivanje adrese prekidne rutine -> poliranje prekida.



Hardverski se radi:

- stavljanje registara PC i PSW na stek

- stavljanje ostalih programski dostupnih registara na stek ako je njihov broj nije velik, inače se čuvaju softverski (u ovom slučaju je uzeto da se čuvaju softverski)

- određivanje adrese zajedničke prekidne rutine i skok na tu prekidnu rutinu

Softverski se radi:

- ako je broj ostalih programski dostupnih registara veliki, registri čije se vrednosti menjaju u prekidnoj rutini se čuvaju na početku te prekidne rutine

- u okviru zajedničke prekidne rutine se određuje koji je uređaj poslao zahtev za prekidom (idući od uređaja najvišeg ka uređaju najnižeg prioriteta)

- izračunava se adresa prekidne rutine uređaja najvišeg prioriteta koji je poslao zahtev za prekidom i skače se u njegovu prekidnu rutinu

- nakon povratka iz prekidne rutine, restauiraju se sačuvane vrednosti ostalih programski dostupnih registara čija se vrednost menjala u zajedničkoj prekidnoj rutini

- vrši se povratak iz zajedničke prekidne rutine

Zajednička prekidna rutina

PUSH R1

PUSH R2

MOV R1, #0 // u R1 se čuva broj ulaza u IVT

PER3: IN R2, Statper3 // učitava se statusni registar

TST R2, #maskaper3 // provera da li je setovan ready bit

JZ PER2 // ako nije, skačemo na proveru za sledeću PER

JMP TBL // ako jeste, skačemo na prekidnu rutinu

PER2: INC R1

IN R2, Statper2

TST R2, #maskaper2

JZ PER1

JMP TBL

PER1: INC R1

IN R2, Statper1

TST R2, #maskaper1

JZ PER0

JMP TBL

PER0: INC R1

TBL: SHL R1 // dve memorijske reči za adresu

ADD R1, tabadresa // nalazi se odgovarajući ulaz

MOV R2, [R1] // dohvata se adresa prekidne rutine

JSR [R2] // skok na prekidnu rutinu

POP R2

POP R1

RTI

**4. У процесору одређене архитектуре од регистара постоје и регистри опште намене (GPR), који могу да се користе на три начина. Навести и објаснити за сваки од ова три начина како се ови регистри користе и чиме је начин коришћења ових регистара одређен.**

Registri opšte namene GPR se koriste na isti način kao registri podataka, adresni registri, bazni registri i indeksni registri:

1) kao DR:

- kod registarskog direktnog adresiranja

- njima se pristupa programskim putem instrukcijama prenosa, aritmetičkim, logičkim i pomeračkim instrukcijama

2) kao AR:

- kod registarskog indirektnog adresiranja i autoinkrement i autodekrement načina adresiranja

- njima se pristupa programskim putem instrukcijama prenosa i aritmetičkim, logičkim i pomeračkim instrukcijama, kada se sadržaj registra koristi kao adresa memorijske lokacije sa koje se čita ili u koju se upisuje podatak

- pristupa im se i instrukcijom skoka JMPIND kada se sadržaj registra koristi kao adresa skoka koju treba upisati u programski brojač PC

3) kao BR:

- kod baznih i bazno-indeksnih adresiranja

- zbir sadržaja specificiranog baznog registra i pomeraja kod baznih adresiranja, odnosno baznog registra, indeksnog registra i pomeraja kod bazno-indeksnih adresiranja, predstavlja adresu memorijske lokacije na kojoj se nalazi izvorišni ili odredišni operand

4) kao XR:

- kod indeksnih i bazno-indeksnih adresiranja, kao i kod operacija nad nizovima alfanumeričkih znakova

- zbir sadržaja specificiranog indeksnog registra i pomeraja kod indeksnih adresiranja, odnosno baznog registra, indeksnog registra i pomeraja kod bazno-indeksnih adresiranja, predstavlja adresu memorijske lokacije na kojoj se nalazi izvorišni ili odredišni operand.

**5. У процесору постоји инструкција контроле петље ACB (Add Compare and Branch – додај упореди и скачи).**

**а) Дати формат инструкције и навести шта се сваким од операнада у формату инструкције специфицира.**

**б) Објаснити како се извршава инструкција.**

a)

ACB limit, step, index, displ

Instrukcija ACB (Add Compare and Branch) realizuje relativni skok sa pomerajem displ u odnosu na tekuću vrednost programskog brojača PC pod uslovom da je suma vrednosti parametara index i step manja ili jednaka vrednosti parametra limit. Ukoliko je uslov za skok ispunjen ažurira se vrednost parametra index sumom vrednosti parametara index i step.

b)

Izvršavanje instrukcije ACB se može predstaviti na sledeći način:

if ((index + step) leq limit) then (PC <= PC + displ),

index <= index + step

**6. (Izraz se naravno menja od roka do roka ali se radi na istu foru)**

**Написати секвенцу инструкција неопходних за срачунавање израза за: F = (A - B ) / ((C + D) \* (E - 5))**

**а) процесор код кога аритметичке инструкције имају формат: OC reg, mem/reg/imm где је OC код операције, први операнд мора бити у регистру (reg), док други може бити у меморији, регистру или дат непосредно (mem/reg/imm). Инструкција LOAD има формат: LOAD reg, mem/reg/imm где је првим операндом дат одредишни регистар (reg), а другим извориште. Инструкција STORE има формат: STORE reg, mem/reg где је првим операндом дат изворишни регистар (reg), а другим одредиште.**

**б) процесор код кога аритметичке инструкције имају формат: OC reg1, reg2, reg3 где је OC код операције, и сва три операнда су дата регистрским директним начином адресирања. Инструкција LOAD и STORE имају исти формат као у тачки а).**

**A, B, C, D, E i F su simboličke oznake adresa memorijskih lokacija u kojima se nalaze operandi. Sadržaj memorijskih lokacija označenih adresama A, B, C, D i E treba da ostane nepromenjen.**

a)

b)

**7. У процесору постоји стринг инструкција CMPC (упоређивање карактера два стринга ради откривања једнакости) која дозвољава да стрингови над којима се она извршава буду различите дужине.**

**а) Дати формат инструкције и навести шта се сваким од операнада у формату инструкције специфицира.**

**б) Објаснити како се извршава инструкција у свим ситуацијама које могу да настану као последица могућих специфицираних дужина стрингова.**

a)

CMPC src1len, src1addr, fill, src2len, src2addr

Instrukcija CMPC (COMPARE CHARACTERS) upoređuje karaktere dva izvorišna stringa radi utvrđivanja da li su dva stringa identična ili ne. Upoređivanje karaktera dva izvorišna stringa se realizuje na dužini dužeg izvorišnog stringa ukoliko postoji jednakost karaktera koji se upoređuju ili se završava ranije i to kada se prvi put otkrije nejednakost karaktera koji se upoređuju.

Početna adresa prvog izvorišnog stringa je src1addr, dužina prvog izvorišnog stringa je src1len, početna adresa drugog izvorišnog stringa je src2addr, dužina drugog izvorišnog stringa je dst2len i karakter FILL je specificiran operandom fill.

b)

Indikator Z ukazuje da li su dva izvorišna stringa identična ili ne, dok indikator C, u slučaju da se otkrije da neki par karaktera dva izvorišna stringa nije identičan, ukazuje koji od ta dva karaktera ima veću vrednost.

Ukoliko su stringovi identični indikator Z bit će biti postavljen na vrednost jedan (Z=1), a indikator C na vrednost nula (C=0).

Ukoliko stringovi nisu identični indikator Z će biti postavljen na vrednost nula (Z=0), a vrednost indikatora C zavisiće od vrednosti karaktera stringova za koje je utvrđeno da nisu jednaki.

Ako su vrednosti takve da je vrednost karaktera prvog izvorišnog stringa manja od vrednosti karaktera drugog izvorišnog stringa indikator C će biti postavljen na vrednost 1 (C=1). Ako su vrednosti takve da je vrednost karaktera prvog izvorišnog stringa veća od vrednosti karaktera drugog izvorišnog stringa indikator C će biti postavljen na vrednost 0 (C=0).

**8. У посматраном рачунарском систему постоји само једна линија по којој улазно/излазни уређаји могу процесору да шаљу захтеве за прекид, док линија за слање сигнала потврде не постоји. У датом систему постоје 4 улазно/излазна уређаја UI3, UI2, UI1 и UI0, при чему приоритет ових уређаја опада од UI3, који има највиши приоритет, преко UI2 и UI1 до UI0, који има најнижи приоритет. Нацртати како та 4 улазно/излазна уређаја треба повезати помоћу те линије на процесор ради слања захтева за прекид и објаснити како се реализује скок на одговарајућу прекидну рутину сваког од 4 улазно/излазна уређаја. Скок на одговарајућу прекидну рутину се реализује делом хардверски, а делом софтверски. Објаснити шта се и како ради хардверски, а шта и како софтверски. За онај део који се ради софтверски написати одговарајући програм коришћењем табеле са адресама прекидних рутина уређаја UI3, UI2, UI1 и UI0. Адресе прекидних рутина уређаја UI3, UI2, UI1 и UI0 су 33333333h, 22222222h, 11111111h и 00000000h, респективно, се налазе у улазима 0, 1, 2 и 3, респективно, посебно формиране табеле, чија се почетна адреса налази у меморијској локацији означеној са tabadresa. Попунити табелу са адресама прекидних рутина уређаја UI3, UI2, UI1 и UI0. Ширина меморијске речи и адресе меморијских локација су 32 бита.**

Radi se na istu foru ko 3. Tabela izgleda ovako:



Primetiti poslednju rečenicu iz teksta zadatka. Na osnovu toga, jedina razlika u odnosu na 3) je ta što u delu gde se nalazi se odgovarajući ulaz umesto

ADD R1, tabadresa

prvo treba pomnožiti R1 sa 2, pa tek onda dodati tabadresa, jer je adresa 64b, a memorijska lokacija 32b.

**9. У процесору постоји инструкција SOB (Subtract One and Branch – Одузми један и скочи).**

**а) Дати формат инструкције и навести шта се сваким од операнада у формату инструкције специфицира.**

**б) Објаснити како би се ефекти ове инструкције реализују код процесора који не поседује ову инструкцију. Као коментар узети формат инструкција процесор по жељи.**

a)

SOB index, displ

Instrukcija SOB (Subtract One and Branch) realizuje relativni skok sa pomerajem displ u odnosu na tekuću vrednost programskog brojača PC pod uslovom da je vrednost parametra index umanjena za 1 veća ili jednaka 0. Ukoliko je uslov za skok ispunjen, ažurira se vrednost parametra index vrednošću parametara index umanjenom za 1.

b)

Izvršavanje instrukcije SOB se može predstaviti na sledeći način:

if ((index - 1) geq 0) then (PC <= PC + displ),

index <= index – 1

**10. (Može da se menja broj parova linija i broj ulaznih/izlaznih uređaja, ali se na istu foru radi)**

**У посматраном рачунарском систему процесор има 4 пара линија по којима улазно/излазни уређаји могу процесору да шаљу захтеве за прекид и од процесора добијају сигнале потврда. У датом систему има 12 улазно/излазних уређаја које треба некако повезати помоћу та 4 пара линија на процесор и омогућити за сваки улазно/излазни уређај скок на одговарајућу прекидну рутину векторисаним механизмом прекида.**

**а) Нацртати како тих 12 улазно/излазних уређаја треба повезати помоћу та 4 пара линија на процесор.**

**б) Објаснити како тих 12 улазно/излазних уређаја шаљу захтеве за прекид и добијају дозволе, и како се реализује скок на одговарајућу прекидну рутину.**

b)

K-ti uredjaj povezan na I-ti par linija za slanje zahteva za prekid i dobijanje potvrde šalje procesoru zahtev za prekid generisanjem signala intrI,K. Uređaji od procesora dobijaju signal potvrde inta pri čemu se serijskom arbitracijom proverava koji je od uređaja poslao zahtev. Uređaji na jednom paru linija su vezani u lanac. Signal na I-toj liniji preko intI stize na intainI,0 i daje prvom uredjaju u lancu (ujedno i prvom po prioritetu) dozvolu za prekid. Vrsi se provera da li je taj uređaj trazio dozvolu i ako nije, onda se generišu signali intaoutI,0 i intainI,1 i vrši se provera sledećeg uređaja u lancu. Ako jeste, onda se startuje čitanje sadržaja registra broja ulaza u tom uređaju. Po završenom čitanju, uređaj šalje očitani sadržaj na linije podataka DBUS i signalom na upravljačkoj liniji FCBUS procesoru signalizira da je broj ulaza raspoloživ na osnovu kojeg procesor određuje adresu prekidne rutine na koju skače.

a)

****

**11. (Od roka do roka se menja broj argumenata i registara opšte namene, ali se radi na istu foru)**

 **Дат је програм PRIMER током чијег извршавања се скаче на процедуру PRIM. У процедури се врши упис у регистре опште намене R0, R1, R2 и програмску статусну реч PSW. По повратку из процедуре вредности ових регистара треба да буду исте као и пре скока на процедуру.**

**а) Навести које се инструкције генеришу приликом позива процедуре, при уласку у процедуру и при изласку из процедуре и објаснити зашто се то ради.**

**б) Нацртати ситуације на стеку по извршавању генерисаних инструкција приликом позива процедуре, при уласку у процедуру и при изласку из процедуре и објаснити зашто се то ради. Стек расте према вишим локацијама и указује на последњу заузету локацију.**

**в) Навести како се генеришу адресе за приступ x, y, z, e и f.**

**PROGRAM PRIMER;**

 **begin**

 **integer a,b,c,d;**

 **...**

 **procedure PRIM (integer x,y,z);**

 **begin**

 **integer e,f;**

 **...**

 **begin**

 **...**

 **end;**

 **end;**

 **...**

 **begin**

 **...**

 **CALL PRIM (a,b,c);**

 **...**

 **end;**

 **end.**

a)

Poziv procedure:

PUSH a ;prvi argument na stek

PUSH b ;drugi argument na stek

PUSH c ;treći argument na stek

JSR PRIM ;poziv potprograma PRIM

SUB SP, 3 ;skidanje argumenata sa steka

Ulazak u proceduru:

PUSH BP ;stara vrednost BPa se stavlja na stek

MOVE SP, BP ;BP se postavlja da pokazuje na lokaciju sa starom vrednošću BPa

PUSH R0 ;u registre R0..R2 i PSW se vrši upis u proceduri PRIM,

PUSH R1 ;a pošto je rečeno da pri povratku iz procedure vrednosti ovih

PUSH R2 ;registara treba da ostanu nepromenjene, moramo ih pre toga

PUSH PSW ;sačuvati na steku (pre njihove izmene), a onda pri povratku

 ;iz procedure treba da restauiramo njihove sačuvane vrednosti

ADD SP, 2 ;odvajamo prostor za lokalne promenljive

Izlazak iz procedure:

SUB SP, 2 ;skidamo sa steka lokalne promenljive

POP PSW ;restauiramo vrednosti registara R0...R2 i PSW

POP R2 ;prethodno sačuvanih na steku pri ulasku u proceduru

POP R1 ;...

POP R0 ;...

POP BP ;restauiramo sačuvanu vrednost BPa skidanjem sa steka

RTS ;povratak iz potprograma

b)



напомене:

● ниже адресе (лево) <-----> више адресе (десно)

● позиција SP-а је обележена црвеном бојом

● позиција BP-а је обележена плавом бојом

v)

Adrese za pristup parametrima i lokalnim promenljivim potprograma se generišu u odnosu na trenutnu vrednost registra BP kao i poziciju u listi parametara, odnosno u listi lokalnih promenljivih, i date su u sledećoj tabeli:



напомена: у изразу BP+X, константа X је померај у односу на вредност регистра BP

**12. У посматраном рачунарском систему процесор има једну линију по којој може да прими захтев за прекид и једну линију по којој може да пошаље сигнал потврде. У датом систему има 5 улазно/излазних уређаја које треба некако повезати помоћу тог пара линија на процесор и омогућити за сваки улазно/излазни уређај скок на одговарајућу прекидну рутину векторисаним механизмом прекида.**

**а) Нацртати како тих 5 улазно/излазних уређаја треба повезати помоћу тог пара линија на процесор.**

**б) Објаснити како тих 5 улазно/излазних уређаја шаљу захтеве за прекид и добијају дозволе, и како се реализује скок на одговарајућу прекидну рутину.**

a)

Jedna linija po kojoj se prima i jedna linija po kojoj se šalje zahtev -> promenljivi ulazi sa serijskim slanjem zahteva i potvrda.



b)

Svaka periferija ima svoj signal slanja zahteva za prekid. Kada je bar jedan od tih signala aktivan, generiše se signal intr. Procesor daje dozvolu aktiviranjem signala inta, koji ulazi u prvu periferiju PER0 (koja ima najviši prioritet). Ukoliko ona nije uputila zahtev za prekid, aktivna vrednost inta se prosleđuje sledećoj periferiji preko intaout. Isti postupak se ponavlja dok se ne dođe do periferije koja je uputila zahtev za prekid. Ona tada šalje svoj broj ulaza u IV tabelu na DBUS i signalom na upravljačkoj liniji FCBUS procesoru signalizira da je broj ulaza raspoloživ.

**12. Програмским путем се помоћу контролера улазно периферије са директним приступом меморији (DMA контролер) уноси блок од 100 бајтова из улазне периферије у неки део меморије. Навести колико пута DMA контролер генерише (1) захтев за коришћење магистрале и (2) захтев за прекид у ситуацијама када је DMA програмиран да пренос реализује у**

**a) појединачном режиму преноса и**

**b) пакетском режиму преноса и објаснити за сваку од ситуација зашто се генерише наведени број захтева.**

a)

-100 puta zahtev za magistralom jer za svaki podatak koji treba kontroler da prenese iz registra podatka u memorijsku lokaciju kontroler mora da traži dozvolu za koriščenje magistrale i da tek po dobijanju dozvole realizuje ciklus na magistrali.

-1 zahtev za prekid jer se prekidna rutina prilikom koriscejna dma kontrolera poziva samo kada on zavrsi sa prenosom podataka(tad se i gasi dma kontroler).

b)

-1 put zahtev za magistralom jer u ovom režimu rada kontroler, po dobijanju dozvole korišćenja magistrale, drži magistralu zauzetu sve vreme dok ne prenese ceo blok podataka i tek po završetku prenosa ukida zahtev za korišćenje magistrale.

-1 zahtev za prekid jer se prekidna rutina prilikom koriscejna dma kontrolera poziva samo kada on zavrsi sa prenosom podataka(tad se i gasi dma kontroler).

**13. У процесору са једноадресним форматом инструкције и бајтовским адресирањем подржан је рад са целобројним величинама дужине један, два и четири бајта, које у меморији заузимају једну, две и четири суседне меморијске локације. Адресним делом инструкције специфицира се адреса само првог бајта захтеване величине без обзира на то да ли је она дужине један, два или четири бајта. Објаснити на основу чега се приликом извршавања инструкција датог процесора утврђује да ли треба из меморије читати само један бајт са специфициране адресе, или два бајта са специфициране адресе и прве суседне адресе, или четири бајта са специфициране адресе и три следеће суседне адресе.**

Za svaki tip podatka (podaci su razlicitih duzina) je definisana posebna instrukcija (npr postoji mnozenje brojeva duzine 1, 2 i 3 bajta). Broj bajtova koje treba pročitati utvrđuje se na osnovu koda operacije.

**14. Написати секвенцу инструкција неопходних за срачунавање израза: G = (A + B) / ((E – F) \* (C + D)) за процесоре са троадресним, двоадресним, једноаресним и нулаадресним форматом инструкција. A, B, C, D, E, F и G су симболичке ознаке адреса меморијских локација у којима се налазе операнди. Садржај меморијских локација означених адресама A, B, C, D, E и F треба да остане непромењен.**

****

****

****

****

**15. У процесору постоји стринг инструкција MOVC (Move with translation – пребацивање са превођењем) која дозвољава да стрингови над којима се она извршава буду различите дужине.**

**а) Дати формат инструкције и навести шта се сваким од операнада у формату инструкције специфицира.**

**б) Објаснити како се извршава инструкција у свим ситуацијама које могу да настану као последица могућих специфицираних дужина стрингова.**

a)

MOVC srclen, srcaddr, fill, dstlen, dstaddr

Instrukcija MOVC realizuje kopiranje karaktera izvorišnog stringa u karaktere odredišnog stringa. Početna adresa izvorišnog stringa je specificirana operandom srcaddr, dužina izvorišnog stringa je specificirana operandom srclen, početna adresa odredišnog stringa je specificirana operandom dstaddr, dužina odredišnog stringa je specificirana operandom dstlen i karakter FILL je specificiran operandom fill.

b)

Indikator Z ukazuje da li su dužine izvorišnog I odredišnog stringa jednake ili ne, dok indikator C, u slučaju da dužine izvorišnog i odredišnog stringa nisu jednake, ukazuje da li je odredišni string duži od izvorišnog ili ne.

Ako su izvorišni i odredišni string jednake dužine (srclen = dstlen) karakteri izvorišnog stringa će u celosti biti kopirani u karaktere odredišnog stringa. Indikator Z će biti postavljen na jedan (Z=1), a indikator C na nulu (C=0).

Ako je izvorišni string duži od odredišnog stringa (srclen > dstlen) biće preneto samo prvih dstlen karaktera izvorišnog stringa u odredišni string, a preostali karakteri izvorišnog stringa neće biti preneti. U ovom slučaju indikatori Z i C će biti postavljeni na nulu (Z=0, C=0).

Ako je izvorišni string kraći od odredišnog stringa (srclen < dstlen) svi karakteri izvorišnog stringa će biti preneti u prvih srclen karaktera odredišnog stringa, a preostali karakteri odredišnog stringa do dužine odredišnog stringa će biti popunjeni karakterom FILL. Indikator Z će biti postavljen na nulu (Z=0), a indikator C na jedan (C=1).

**16. У процесору постоји стринг инструкција MOVT (пренос са превођењем) која дозвољава да стрингови над којима се она извршава буду различите дужине.**

**а) Дати формат инструкције и навести шта се сваким од операнада у формату инструкције специфицира.**

**б) Објаснити како се извршава инструкција у свим ситуацијама које могу да настану као последица могућих специфицираних дужина стрингова.**

a)

MOVT srclen, srcaddr, fill, tbladdr, dstlen, dstaddr

Instrukcija MOVT realizuje prevodjenje karaktera izvorišnog stringa i njihovo smeštanje u odgovarajuće karaktere odredišnog stringa. Za prevodjenje se koristi translaciona tabela koja ima 256 ulaza. Element odredišnog stringa popunjava se karakterom koji se nalazi unutar translacione tabele na adresi koja je određena vrednošću karaktera izvorišnog stringa koji se prevodi.

Početna adresa izvorišnog stringa je srcaddr, dužina izvorišnog stringa je srclen, početna adresa odredišnog stringa je dstaddr, dužina odredišnog stringa je dstlen, adresa translacione tabele je tbladdr i karakter FILL je specificiran operandom fill.

b)

Ako su izvorišni i odredišni string jednake dužine (srclen = dstlen) svaki karakter izvorišnog stringa će biti preveden i smešten na odgovarajuće mesto unutar odredišnog stringa. Indikator Z će biti postavljen na jedan (Z=1), a indikator C na nulu (C=0).

Ako je izvorišni string duži od odredišnog stringa (srclen > dstlen) biće prevedeno samo prvih dstlen karaktera izvorišnog stringa i smešteno na

odgovarajuće mesto unutat odredišnog stringa, a preostali karakteri izvorišnog stringa neće biti prevedeni. U ovom slučaju indikatori Z i C će biti postavljeni na nulu (Z=0, C=0).

Ako je izvorišni string kraći od odredišnog stringa (srclen < dstlen) svi karakteri izvorišnog stringa će biti prevedeni i prevedene vrednosti smeštene u prvih srclen karaktera odredišnog stringa, a preostali karakteri odredišnog stringa do dužine odredišnog stringa će biti popunjeni karakterom FILL. Indikator Z će biti postavljen na nulu (Z=0), a indikator C na jedan (C=1).

**17. У процесору постоји стринг инструкција MATCH (налажење подстринга у стрингу) код које стрингови над којима се она извршава могу да буду различите дужине.**

**a) Дати формат инструкције и навести шта се сваким од операнада у формату инструкције специфицира.**

**b) Објаснити како се извршава инструкција.**

a)

MATCH src1len, src1addr, src2len, src2addr

Početna adresa izvorišnog stringa src1addr, dužina izvorišnog stringa je src1len, početna adresa podstringa je src2addr i dužina podstringa je src2len.

b)

Instrukcija MATCH (MATCH CHARACTERS) izvršava operaciju poredjenja izmedju karaktera izvorišnog stringa i karaktera podstringa. Operacija se realizuje ili na dužini izvorišnog stringa ukoliko je rezultat operacije nejednakost ili se završava ranije kada se prvi put otkrije da je rezultat operacije jednakost.

Rezultat izvršavanja instrukcije MATCH je utvrdjivanje da li se u nekom delu izvorišnog stringa nalazi niz karaktera identičan sa karakterima podstringa. Rezultat izvršavanja instrukcije MATCH se daje postavljanjem indikatora Z na vrednost jedan ili nula, dok indikator C uvek postavlja na vrednost nula (C=0). Ukoliko podstring nije pronadjen izvorišnom stringu indikator Z bit će biti postavljen na vrednost jedan (Z=1). Ukoliko je podstring pronadjen u izvorišnom indikator Z bit će biti postavljen na vrednost nula (Z=0).

**18. У процесору су од типова податак подржане 8 битне и 16 битне величине без знака и са знаком и 32 битне и 64 битне величине у покретном зарезу. Навести колико кодова операција мора да постоји за операцију множења у датом процесору и објаснити шта се сваким од тих кодова операција специфицира.**

Za svaki tip podataka koji je podržan od strane procesora mora da postoji po jedan kod operacije množenja. Broj tipova podataka podržanih od strane procesora ima 6, pa toliko ima i kodova za

operaciju množenja, a kojima se specificira množenje sledećih veličina:

1. 8-bitnih veličina bez znaka

2. 8-bitnih veličina sa znakom

3. 16-bitnih veličina bez znaka

4. 16-bitnih veličina sa znakom

5. 32-bitnih veličina u pokretnom zarezu

6. 64-bitnih veličina u pokretnom zarezu

**19. У процесору су од типова податак подржане 32 битне и 64 битне целобројне величине са знаком и без знака, и 32 битне и 64 битне величине у покретном зарезу. Навести колико кодова операција мора да постоји за операцију множења у датом процесору и објаснити шта се сваким од тих кодова операција специфицира.**

Za svaki tip podataka koji je podržan od strane procesora mora da postoji po jedan kod operacije množenja. Broj tipova podataka podržanih od strane procesora ima 6, pa toliko ima i kodova za

operaciju množenja, a kojima se specificira množenje sledećih veličina:

1. 32-bitnih veličina bez znaka

2. 32-bitnih veličina sa znakom

3. 64-bitnih veličina bez znaka

4. 64-bitnih veličina sa znakom

5. 32-bitnih veličina u pokretnom zarezu

6. 64-bitnih veličina u pokretnom zarezu