Međuispit iz Arhitekture računala 1 **2. svibnja 2013.** **Grupa na predavanjima: _____**

Prezime i ime (tiskanim slovima): _____ JMBAG: _____

Izjavljujem da tijekom izrade ove zadaće neću od drugoga primiti niti drugome pružiti pomoć, te da se neću koristiti nedopuštenim sredstvima. Ove su radnje teška povreda Kodeksa ponašanja te mogu uzrokovati i trajno isključenje s Fakulteta. Izjavljujem da mi zdravstveno stanje dozvoljava pisanje ove zadaće.

Potpis: _____.

Dozvoljeno je koristiti isključivo službeni popis naredaba FRISC-a. Programe treba pisati uredno i komentirati pojedine cjeline programa. Sve teorijske zadatke rješavati na ovaj papir. Međuispit traje 120 minuta.

1 a) (*2 boda*) Sljedeći program treba trajati **točno 6 milisekundi** (uz pretpostavku da FRISC radi na 10 MHz). Kraj svake naredbe napišite koliko puta se naredba izvodi po koliko ciklusa (npr. 6 x 2c ili 1 x 2c + 1 x 1c) i izračunajte koja **vrijednost** mora biti zapisana na lokaciji KONST.

		trajanje u ciklusima
	LOAD R0,(KONST)	_____
PETLJA	LOAD R1,(BROJAC)	_____
	ADD R1,1,R1	_____
	STORE R1,(BROJAC)	_____
	CMP R1,R0	_____
	JR_NE PETLJA	_____
KONST	DW _____	
BROJAC	DW 0	

1 b) (*0,5 boda*) U memoriji FRISC-a zapisan je 16-bitni broj u formatu *big-endian*: na adresi 100_{16} zapisano je 11111110_2, a na adresi 101_{16} zapisano je 11111100_2. Koji je to broj ako ga promatramo kao 16-bitni zapis u formatu dvojnog komplementa _____. (*brojeve prikažite dekadski*)

1 c) (*1 bod*) Broj razina protočne strukture FRISC-a je _____. Naredba se dekodira u razini _____. Dvije vrste hazarda kod FRISC-a su:_____ i _____. Postoji još i _____ hazard, ali do njega ne dolazi kod FRISC-a.

1 d) (*1,5 bod*) Kod FRISC-a postoje dvije vrste prekida:

1) _____ koji dolazi preko priključka _____ i

2) _____ koji dolazi preko _____.

Zastavica GIE nalazi se u registru _____ i ako je u ništici, onda je _____ prekid _____.

1 e) (*1,5 bod*) Za 5-bitne brojeve izvodi se aritmetička operacija. Odredite rezultat operacije i vrijednost prijenosa, preljeva, posudbe ništice i predznaka (**općenito**, NE za FRISC). **Potrebno je napisati postupak rješenja.**

	prijenos	**posudba**	**preljev**	**ništica**	**predznak**
10110+10011 = _____	**___**	**___**	**___**	**___**	**___**
00110-11010 = _____	**___**	**___**	**___**	**___**	**___**

1 f) (*1,5 boda*) Na prazne crte upišite korake koje FRISC obavlja prilikom izvođenja naredbe **CMP R1,35**. Ne moraju se popuniti sve crte.

Razina dohvata:

Prva polovina periode CLOCK-a:

_____

Druga polovina periode CLOCK-a:

_____

_____

_____

_____

Razina izvođenja:

Prva polovina periode CLOCK-a:

_____

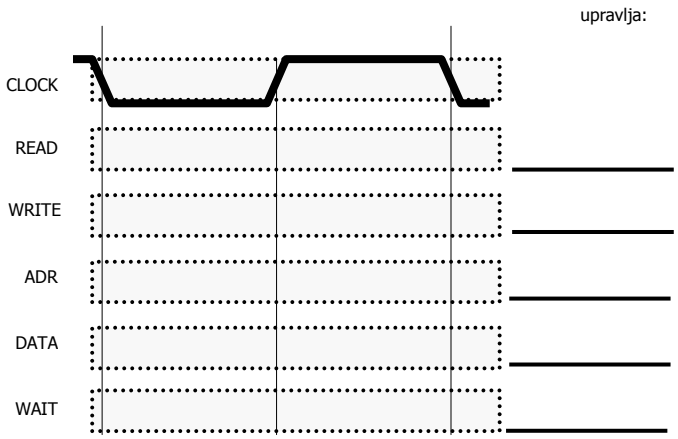
_____

Druga polovina periode CLOCK-a:

_____

_____

1 g) (3 boda) Nacrtajte signale na sabirnicama prilikom čitanja iz brze memorije kod FRISC-a. Napišite (na crte s desna) tko upravlja dotičnom sabirnicom.



2. (*6 bodova*) U memoriji se nalazi blok parova 8-bitnih brojeva u zapisu 1'k. Adresa početka bloka je 1000_{16}, a blok je zaključen parom u kojem je barem jedan od brojeva jednak pozitivnoj nuli: 00_{16} (takav par nije dio bloka).

Napišite program koji pretvara parove brojeva na sljedeći način: ako su brojevi u paru istih predznaka, ne treba učiniti ništa. Ako su brojevi u paru različitih predznaka, brojeve treba pretvoriti u 8-bitne brojeve u zapisu 2'k. Također je potrebno brojati koliko je parova brojeva s različitim predznacima.

Pretvorene je brojeve potrebno pohraniti na iste memorijske lokacije izvornih brojeva. Zaključni par brojeva treba ostaviti nepromijenjen. Broj parova brojeva s različitim predznacima treba pohraniti na adresu RAZLICITI.

3. (*7,5 bodova*) U memoriji se nalazi blok 32-bitnih podataka zapisanih u formatu 2'k. Adresa bloka zapisana je u lokaciji ADRESA, a broj podataka u bloku zapisan je u lokaciji BROJ. U bloku su podatci grupirani u trojke (tri uzastopna 32-bitna podatka).

Napišite glavni program koji pomoću potprograma TROJKA treba obraditi sve trojke u bloku podataka i pomoću potprograma PREDZNACI prebrojiti koliko je negativnih podataka bilo u početnom bloku. Broj negativnih podataka treba zapisati na memorijsku lokaciju NEGATIVNI.

Napišite potprogram PREDZNACI koji preko stoga prima tri 32-bitna parametra. Potprogram treba prebrojiti koliko ima negativnih podataka među parametrima i broj negativnih treba vratiti preko fiksne lokacije NEG_U_3.

Napišite potprogram TROJKA koji preko registra R0 prima adresu trojke podataka. Potprogram TROJKA mora pomoću potprograma PREDZNACI odrediti koliko ima negativnih podataka u trojki, i zatim vratiti taj broj svome pozivatelju pomoću registra R0. Dodatno, potprogram TROJKA treba sve podatke u trojki zamijeniti brojem -1, ali samo ako u trojki ima strogo više od jednog negativnog broja.

4. *(7,5 bodova)* U računalnom sustavu nalazi se procesor FRISC i četiri vanjske jedinice: jedna uvjetna UVJ0, jedna bezuvjetna BVJ1 te dvije prekidne: PVJ2 (spojena na INT2) i PVJ3 (spojena na INT3). Adrese vanjskih jedinica odaberite sami.

Napišite program koji preuzima 32-bitne podatke u zapisu NBC s uvjetne vanjske jedinice UVJ0, i sprema ih u međuspremnik koji se nalazi na adresi BUFFER. Međuspremnik ima kapacitet od 10_{10} podataka i puni se od nižih adresa prema višima. Ako je međuspremnik prepunjen, onda se novopristigli podatci ne zapisuju, ali se u lokaciji PRELJEVI prebraja koliko je podataka izgubljeno, tj. nije zapisano.

Na svaki prekid vanjske jedinice PVJ2, potrebno je učiniti sljedeće:

- ako je između prošle i ove obrade prekida došlo do gubitka podataka, na PVJ2 poslati vrijednost 1, a inače poslati vrijednost 0;
- na bezuvjetnu vanjsku jedinicu BVJ1 poslati sve dotad primljene podatke iz međuspremnika redoslijedom primanja, a sljedeće podatke ponovno puniti od početka međuspremnika.

Na svaki prekid vanjske jedinice PVJ3, potrebno je na PVJ3 poslati broj dosad obrađenih zahtjeva za prekid od jedinice PVJ2.

Glavni program izvodi se beskonačno.

Međuispit iz Arhitekture računala 1 **2. svibnja 2013.** **Grupa na predavanjima: _____**

Prezime i ime (tiskanim slovima): _____ JMBAG: _____

Izjavljujem da tijekom izrade ove zadaće neću od drugoga primiti niti drugome pružiti pomoć, te da se neću koristiti nedopuštenim sredstvima. Ove su radnje teška povreda Kodeksa ponašanja te mogu uzrokovati i trajno isključenje s Fakulteta. Izjavljujem da mi zdravstveno stanje dozvoljava pisanje ove zadaće.

Potpis: _____.

Dozvoljeno je koristiti isključivo službeni šalabahter (popis naredaba FRISC-a). Programe treba pisati uredno i komentirati pojedine cjeline programa. Sve teorijske zadatke rješavati na ovaj papir. Međuispit traje 120 minuta.

1 a) (*2 boda*) Sljedeći program treba trajati **točno 6 milisekundi** (uz pretpostavku da FRISC radi na 10 MHz). Kraj svake naredbe napišite koliko puta se naredba izvodi po koliko ciklusa (npr. 6 x 2c ili 1 x 2c + 1 x 1c) i izračunajte koja **vrijednost** mora biti zapisana na lokaciji KONST.

	LOAD R0,(KONST)	___**1x2c**___
	ADD R0,0,R0	___1x1c___
	LOAD R1,(BROJAC)	___**1x2c**___
P	ADD R1,1,R1	___9999 x 1c____
	STORE R1,(BROJAC)	___9999 x 2c__
	CMP R1,R0	___9999 x 1c____
	JR_NE P	__9998 x 2c + 1x1c__
	HALT	__1x2c____
KONST	DW __**%D 9999**__	
BROJAC	DW 0	

1 b) (*0,5 boda*) U memoriji FRISC-a zapisan je 16-bitni broj u formatu *big-endian*: na adresi 100_{16} zapisano je 11111110_2, a na adresi 101_{16} zapisano je 11111100_2. Koji je to broj ako ga promatramo kao 16-bitni zapis u formatu dvojnog komplementa ___**-260**___. (*brojeve prikažite dekadski*)

1 c) (*1 bod*) Broj razina protočne strukture FRISC-a je ___**2**___ . Naredba se dekodira u razini ____**1**_-**(dohvata - može i samo broj)**_____. Dvije vrste hazarda kod FRISC-a su: _**strukturni**_____ i _**upravljački**_____. Postoji još i _____**podatkovni**_____ hazard, ali do njega ne dolazi kod FRISC-a.

1 d) (*1,5 bod*) Kod FRISC-a postoje dvije vrste prekida:

1) __**maskirajući** _ koji dolazi preko priključka _**INT (može i INT[0])**___ i

2) ___**nemaskirajući**____ koji dolazi preko _____**NMI (može i INT[1])** _____.

Zastavica GIE nalazi se u registru _**SR**_ i ako je u ništici, onda je ___**maskirajući**___ prekid **zabranjeni/onemogućeni/maskirani** _.

1 e) (*1,5 bod*) Za 5-bitne brojeve izvodi se aritmetička operacija. Odredite rezultat operacije i vrijednost prijenosa, preljeva, posudbe ništice i predznaka (**općenito**, NE za FRISC). **Potrebno je napisati postupak rješenja.**

		prijenos	**posudba**	**preljev**	**ništica**	**predznak**
10110+10011 =	**____01001____**	**1___**	**0___**	**1___**	**0___**	**0___**
00110-11010 =	**____01100____**	**0___**	**1___**	**0___**	**0___**	**0___**

1 f) (*1,5 boda*) Na prazne crte upišite korake koje FRISC obavlja prilikom izvođenja naredbe **CMP R1,35**. Ne moraju se popuniti sve crte.

Razina dohvata:

Prva polovina periode CLOCK-a:

_____ **PC → AR**_____

Druga polovina periode CLOCK-a:

_____ **(AR) → IR**_____

_____ **dekodiranje** _____

_____ **R1 i ext 35 → ALU**_____

_____ **ALU: izvodi oduzimanje** _____

_____ **PC+4 → PC**_____

Razina izvođenja:

Prva polovina periode CLOCK-a:

_____**spremanje zastavica u SR-u** _____

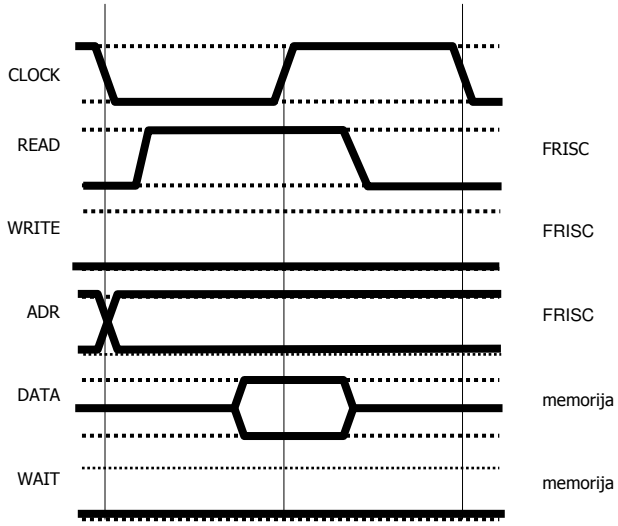
_____

Druga polovina periode CLOCK-a:

_____

_____

1 g) (3 boda) Nacrtajte signale na sabirnicama prilikom čitanja iz brze memorije kod FRISC-a. Napišite (na crte s desna) tko upravlja dotičnom sabirnicom.



2. (*6 bodova*) U memoriji se nalazi blok parova 8-bitnih brojeva u zapisu 1'k. Adresa početka bloka je 1000_{16}, a blok je zaključen parom u kojem je barem jedan od brojeva jednak pozitivnoj nuli: 00_{16} (takav par nije dio bloka).

Napišite program koji pretvara parove brojeva na sljedeći način: ako su brojevi u paru istih predznaka, ne treba učiniti ništa. Ako su brojevi u paru različitih predznaka, brojeve treba pretvoriti u 8-bitne brojeve u zapisu 2'k. Također je potrebno brojati koliko je parova brojeva s različitim predznacima.

Pretvorene je brojeve potrebno pohraniti na iste memorijske lokacije izvornih brojeva. Zaključni par brojeva treba ostaviti nepromijenjen. Broj parova brojeva s različitim predznacima treba pohraniti na adresu RAZLICITI.

```
       ORG 0
       MOVE 0, R6               ; R6 – brojač parova s različitim predznacima
       MOVE 1000, R0            ; R0 – adresa početka bloka

PETLJA
       LOADB R1, (R0)           ; učitavanje prvog broja u paru
       CMP R1, 0                ; provjera kraja bloka
       JR_EQ KRAJ

       LOADB R2, (R0+1)         ; učitavanje drugog broja u paru
       CMP R2, 0                ; provjera kraja bloka
       JR_EQ KRAJ

OK     AND R1, 80, R3           ; izdvajanje predznaka iz brojeva
       AND R2, 80, R4
       CMP R3, R4               ; usporedba predznaka (može i XOR R3, R4, R5)
       JR_EQ DALJE

RAZL   ADD R6, 1, R6            ; različiti su, povećavanje brojača
TST1   ROTL R1, %D 24, R3       ; koji je prvi predznak (može se i testirati R3, ima samo 7.bit)
       JR_NC TST2               ; pozitivne brojeve ne treba mijenjati
NEG1   ADD R1, 1, R1            ; 1'K -> 2'K
TST2   ROTL R2, %D 24, R4
       JR_NC GOTOVO
NEG2   ADD R2, 1, R2            ; 1'K -> 2'K

GOTOVO STOREB R1, (R0)          ; spremanje rezultata
       STOREB R2, (R0+1)
DALJE  ADD R0, 2, R0            ; povećavanje pokazivača za adresu
       JP PETLJA                ; povratak na sljedeći par
KRAJ   STORE R6, (RAZLICITI)    ; spremanje broja različitih parova
       HALT

RAZLICITI DW 0
```

3. (*7,5 bodova*) U memoriji se nalazi blok 32-bitnih podataka zapisanih u formatu 2'k. Adresa bloka zapisana je u lokaciji ADRESA, a broj podataka u bloku zapisan je u lokaciji BROJ. U bloku su podatci grupirani u trojke (tri uzastopna 32-bitna podatka).

Napišite glavni program koji pomoću potprograma TROJKA treba obraditi sve trojke u bloku podataka i pomoću potprograma PREDZNACI prebrojiti koliko je negativnih podataka bilo u početnom bloku. Broj negativnih podataka treba zapisati na memorijsku lokaciju NEGATIVNI.

Napišite potprogram PREDZNACI koji preko stoga prima tri 32-bitna parametra. Potprogram treba prebrojiti koliko ima negativnih podataka među parametrima i broj negativnih treba vratiti preko fiksne lokacije NEG_U_3.

Napišite potprogram TROJKA koji preko registra R0 prima adresu trojke podataka. Potprogram TROJKA mora pomoću potprograma PREDZNACI odrediti koliko ima negativnih podataka u trojki, i zatim vratiti taj broj svome pozivatelju pomoću registra R0. Dodatno, potprogram TROJKA treba sve podatke u trojki zamijeniti brojem -1, ali samo ako u trojki ima strogo više od jednog negativnog broja.

```
       ORG 0
       MOVE 10000, SP            ; inicijalizacija stoga

       LOAD R1, (BROJ)           ; R1 – broj podataka
       LOAD R2, (ADRESA)         ; R2 – adresa bloka
       MOVE 0, R3                ; R3 – brojač negativnih podataka
PETLJA
       MOVE R2, R0               ; adresa trenutne trojke, R0 će biti prebrisan poslije potp.
       CALL TROJKA
       ADD R0, R3, R3            ; povećaj brojač neg.pod
       ADD R2, %D12, R2          ; povećaj adresu trenutne trojke
       SUB R1, 3, R1             ; oduzmi brojač podataka
       JR_NZ PETLJA
       STORE R3, (NEGATIVNI)
       HALT


TROJKA
       PUSH R1                   ; kontekst
       PUSH R2
       PUSH R3

       LOAD R1, (R0)             ; učitavanje podataka
       LOAD R2, (R0+4)
       LOAD R3, (R0+8)

       PUSH R1                   ; parametri za potprogram PREDZNACI
       PUSH R2
       PUSH R3

       CALL PREDZNACI            ; poziv potprograma

       ADD SP, %D 12, SP         ; brisanje parametara
       LOAD R1, (NEG_U_3)        ; učitavanje rezultata

       CMP R1, 1                 ; veće od 1, ili ROTR 1 (ili SUB R0,2,R0)
       JR_ULE KRAJ               ; 0 ili 1 neg. broj
       MOVE -1, R2               ; inače, 2 ili 3, staviti -1
       STORE R2, (R0)            ; spremanje rezultata
       STORE R2, (R0+4)
       STORE R2, (R0+8)
KRAJ   MOVE R1, R0               ; spremi povratnu vrijednost u R0

       POP R3
       POP R2
       POP R1
       RET


PREDZNACI
       PUSH R1
       PUSH R2

       MOVE 0, R2                ; brojač negativnih
BR1    LOAD R1, (SP+%D 12)       ; učitavanje trojke brojeva
       ROTL R1, 1, R1            ; predznak broja 1?
       JR_NC BR2
```

```
        ADD R2, 1, R2               ; povećati brojač negativnih

BR2     LOAD R1, (SP+%D 16)
        ROTL R1, 1, R1              ; predznak broja 2
        JR_NC BR3
        ADD R2, 1, R2

BR3     LOAD R1, (SP+%D 20)
        ROTL R1, 1, R1              ; predznak broja 3
        JR_NC GOTOVO
        ADD R2, 1, R2

GOTOVO STORE R2, (NEG_U_3)          ; spremanje rezultata
        POP R2
        POP R1
        RET

ADRESA DW 1000
BROJ DW 30
NEGATIVNI DW 0
NEG_U_3 DW 0
```

4. *(7,5 bodova)* U računalnom sustavu nalazi se procesor FRISC i četiri vanjske jedinice: jedna uvjetna UVJ0, jedna bezuvjetna BVJ1 te dvije prekidne: PVJ2 (generira maskirajući prekid) i PVJ3 (generira nemaskirajući prekid). Adrese vanjskih jedinica odaberite sami.

Napišite program koji preuzima 32-bitne podatke u zapisu NBC s uvjetne vanjske jedinice UVJ0, i sprema ih u međuspremnik koji se nalazi na adresi BUFFER. Međuspremnik ima kapacitet od 10_{10} podataka i puni se od nižih adresa prema višima. Ako je međuspremnik prepunjen, onda se novopristigli podatci ne zapisuju, ali se u lokaciji PRELJEVI prebraja koliko je podataka izgubljeno, tj. nije zapisano.

Na svaki prekid vanjske jedinice PVJ2, potrebno je učiniti sljedeće:

- ako je između prošle i ove obrade prekida došlo do gubitka podataka, na PVJ2 poslati vrijednost 1, a inače poslati vrijednost 0;
- na bezuvjetnu vanjsku jedinicu BVJ1 poslati sve dotad primljene podatke iz međuspremnika redoslijedom primanja, a sljedeće podatke ponovno puniti od početka međuspremnika.

Na svaki prekid vanjske jedinice PVJ3, potrebno je na PVJ3 poslati broj dosad obrađenih zahtjeva za prekid od jedinice PVJ2.

Glavni program izvodi se beskonačno.

Napomena: Na lokaciji PRELJEVI može se prebrajati ukupan broj izgubljenih podataka, ili broj izgubljenih podataka od zadnjeg prekida INT2. U ovom rješenju se broje izgubljeni podatci između dva prekida INT2.

```
UVJ0_PRIMI   EQU    0FFFF1000    ; adrese vanjskih jedinica
UVJ0_STANJE  EQU    0FFFF1004

BVJ1         EQU    0FFFF2000

PVJ2_POD     EQU    0FFFF3000
PVJ2_IACK    EQU    0FFFF3004
PVJ2_IEND    EQU    0FFFF3008
PVJ2_STOP    EQU    0FFFF300C

PVJ3_POD     EQU    0FFFF4000
PVJ3_IACK    EQU    0FFFF4004
PVJ3_IEND    EQU    0FFFF4008
PVJ3_STOP    EQU    0FFFF400C

        ORG 0
        MOVE 10000, SP              ; inicijalizacija stoga
        JP GLAVNI                   ; skok na glavni program


        `ORG 8                      ; MI
        DW 1000


        `ORG 0C                     ; NMI
```

```
       PUSH R0                  ; kontekst; SR ne treba spremati, jer se ne mijenja

       STORE R0, (PVJ3_IACK)    ; dojava prihvata prekida
       LOAD R0, (BROJACMI)      ; učitavanje broja obrađenih prekida
       STORE R0, (PVJ3_POD)     ; slanje na PVJ3
       STORE R0, (PVJ3_IEND)    ; dojava kraja posluživanja
       POP R0
       RETN                     ; povratak iz potprograma


       ORG 1000
       PUSH R0                  ; kontekst
       PUSH R1
       PUSH R2
       PUSH R3
       PUSH R4
       MOVE SR, R0              ; spremanje SR-a
       PUSH R0

       STORE R0, (PVJ2_IACK)    ; prihvat prekida

       LOAD R0, (BROJACMI)      ; povećavanje brojača prekida
       ADD R0, 1, R0
       STORE R0, (BROJACMI)

       LOAD R0, (PRELJEVI)      ; je li bilo prepunjenja BUFFER-a?
       CMP  R0, 1
       JR_UGE BILO_P
NIJE_P STORE R0, (PVJ2_POD)     ; ako nije bilo, poslati 0 na PVJ2
       JR GOTOVO
BILO_P MOVE 1, R0
       STORE R0, (PVJ2_POD)     ; ako je bilo, poslati 1 na PVJ2

GOTOVO MOVE BUFFER, R0          ; učitavanje adrese međuspremnika
       LOAD R1, (BROJACBUF)     ; pokazivač u međuspremniku - broj trenutno napunjenih
       MOVE 0, R2               ; trenutni pokazivač za slanje podataka

SALJI  ADD R0, R2, R4           ; R4 - adresa pojedinog podataka iz međuspremnika
       LOAD R3, (R4)            ; učitavanje podatka
                                ; ili drugačije: LOAD R3,(R2+BUFFER) pa se onda...
                                ; ... ne moraju koristiti R4 i R0
       STORE R3, (BVJ1)         ; slanje na BVJ1
       ADD R2, 4, R2            ; povećavanje adrese
       CMP R2, R1               ; trenutni pokazivač obradio sve podatke iz međuspremnika?
       JR_NE SALJI
       MOVE 0, R1
       STORE R1, (BROJACBUF)    ; brisanje pokazivača u međuspremniku
       STORE R1, (PRELJEVI)     ; brisanje brojača/zastavice PRELJEVI

       STORE R1, (PVJ2_IEND)    ; dojava kraja posluživanja

       POP R0                   ; kontekst
       MOVE R0, SR
       POP R4
       POP R3
       POP R2
       POP R1
       POP R0
       RETI


GLAVNI MOVE %B 10000, SR        ; omogućavanje prekida INT
       MOVE BUFFER, R1          ; R1 - adresa međuspremnika

CEKAJ  LOAD R0, (UVJ0_STANJE)   ; čekanje UVJ0
       OR R0, R0, R0
       JP_Z CEKAJ

       LOAD R0, (UVJ0_PRIMI)    ; primanje podatka
       STORE R0, (UVJ0_STANJE)  ; brisanje spremnosti

       LOAD R2, (BROJACBUF)     ; provjera pokazivača u međuspremniku - preljev?
       CMP R2, %D 40            ; 10 mjesta (40 bajtova) u međuspremniku
       JR_EQ PREVISE
       ADD R1, R2, R3
       STORE R0, (R3)           ; još nije prepunjeno -> spremanje podatka
```

```
                                ; ili drugačije: STORE R0,(R2+BUFFER) pa se onda...
                                ; ...ne moraju koristiti R1 i R3
       ADD R2, 4, R2            ; povećavanje brojača u međuspremniku (4 BAJTA!)
       STORE R2, (BROJACBUF)    ; spremanje brojača
       JP CEKAJ
PREVISE LOAD R0, (PRELJEVI)     ; brojač preljeva (ujedno i zastavica za preljev)
       ADD R0, 1, R0
       STORE R0, (PRELJEVI)
       JP CEKAJ

BUFFER DS %D 40

BROJACMI DW 0
BROJACBUF DW 0
PRELJEVI DW 0
```