Međuispit iz Arhitekture računala 1 **26. travanj 2019.**

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

Izjavljujem da tijekom ispita 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 ovog ispita. Potpis: _____.

Dozvoljeno je koristiti isključivo službene šalabahtere (popis naredaba FRISC-a i ARM-a). Programe treba pisati uredno i komentirati pojedine cjeline programa. Rješenja teorijskih zadatka treba napisati na ovaj papir. Ispit traje 100 minuta i nosi 30 bodova.

1.a (1,5 bod) Dekadski broj 251,375 pretvorite u broj u binarnoj bazi **1111 1011,011** , a zatim binarni broj prikažite u heksadekadskoj bazi **FB,6** .

OBAVEZNO SE MORA VIDJETI POSTUPAK:

251 / 2 = 125 i ostatak 1 (niža)
125/2 = 62 i ostatak 1
62/2 = 31 i ostatak 0
31/2 =15 i ostatak 1
15/2=7 i ostatak 1 **može i neki drugi postupak**
7/2 =3 i ostatak 1
3/2 =1 i ostatak 1
1/2 = 0 i ostatak 1 (viša)

0,375*2 = 0,75 (viša)
0,75* 2 = 1,5
0,5 * 2 = 1,0 (niža)

1.b (1,5 bod) Heksadekadski broj -A,2 prikažite u binarnoj bazi **-1010,0010** . Binarni broj zatim pretvorite u dekadski **-10,125** .

OBAVEZNO SE MORA VIDJETI POSTUPAK:

$1*2^3 + 0*2^2 + 1*2^1 + 0*2^0 = 8 + 2 = 10$

$0*2^{-1} + 0*2^{-2} + 1*2^{-3} + 0*2^{-4} = 0,125$

1.c (1 bod) Ulazna uvjetna vanjska jedinica primila je podatak od vanjskog uređaja, ali je FRISC još nije poslužio. U tom trenutku vrijedi: status bistabil je u stanju **1 (spremnosti)**, priključak READY je u stanju **0 (nespremnosti/neaktivan)**. Ako postoji više uvjetnih vanjskih jedinica, i ako su one nezavisne, onda ih poslužujemo postupkom koji se naziva **prozivanje (pooling)**. U uvjetnoj vanjskoj jedinici postoji jedan sklopovski dio koji ne postoji u bezuvjetnoj - koji je to dio: **bistabil stanja (status bistabil)**.

1.d (2,5 boda) Za FRISC sa brzom memorijom odredite **trajanje** izvođenja sljedećeg **programskog odsječka**. Na prvoj crti napišite **koliko puta** se naredba izvodi i **koliko** ciklusa traje izvođenje jedne naredbe (npr. 6 x 2c ili 45 x 2c + 1 x 1c). Ako neka naredba izaziva **hazard**, napišite njegovo ime na drugoj crti. Cijeli odsječak traje **17** ciklusa.

		Trajanje	Hazard
	`MOVE 3, R3`	1x1c +1 punjenje	
`L1`	`STORE R3,(R3+1000)`	3x2c	strukturni
	`SUB R3, 1, R3`	3x1c	
	`JR_NZ L1`	2x2c + 1x1c	upravljački
	`MOVE 0, R2`	1x1c	

1.e (2 boda) Na prazne crte upišite korake koje FRISC obavlja prilikom izvođenja naredbe **XOR R1,58,R0**. **Ne morate popuniti sve crte.**

Razina dohvata:

Rastući brid CLOCK-a:

PC -> AR

Padajući brid CLOCK-a:

(AR) -> IR
dekodiranje
operandi->ALU (ili R1, ext 58->ALU)
izbor i pokretanje operacije (XOR)
PC + 4 -> PC

Razina izvođenja:

Rastući brid CLOCK-a:

ALU završava operaciju, rezultat->R0
stanje zastavica -> SR

Padajući brid CLOCK-a:

1.f (1 bod) Zaokružite točne odgovore za FRISC.

Ulazno-izlazna i memorijska sabirnica kod FRISC-a su:

zajednička sabirnica **točan** — neizravno spojene (pomoću međusklopa)

Za UI-adresiranje FRISC koristi:

memorijsko UI preslikavanje **točan** — izdvojeno UI adresiranje

Sabirnički protokoli kod FRISC-a su:

asinkroni — sinkroni **točan**

Koliko taktova signala clock traje čitanje iz brze memorije:

jedan takt **točan** — jedan takt + jedan takt čekanja

2. FRISC (9 bodova) Napišite potprogram BZP_2K koji **16-bitni format s bitom za predznak** pretvara u **32-bitni format 2'k**. Parametar se prenosi stogom, a rezultat se vraća registrom R0.

U memoriji se na adresi BLOK nalazi blok **16-bitnih brojeva u formatu s bitom za predznak**. Blok ima 100_{16} podataka i mora biti na adresi 1000_{16}.

Glavni program treba za svaki *podatak* iz bloka BLOK izračunati formulu: *podatak**4 - *podatak*/8+*podatak*. Dobiveni rezultat treba spremiti **kao 32-bitni broj u formatu 2'k** u memorijski blok na adresi REZ. Blok REZ mora se nalaziti neposredno iza bloka BLOK.

Prilikom izračuna formule koristite format 2'k jer je on podržan s aritmetičko-logičkom jedinicom procesora. Za pretvorbu brojeva koristite potprogram BZP_2K. Blokove podataka morate definirati (naravno, umjesto podataka u prvom bloku napišite ...)

```
GLAVNI      MOVE   10000, SP      ; inicijalizacija stoga
            MOVE   BLOK, R1           ; inicijalizacija pokazivača i brojača
            MOVE   REZ, R2
            MOVE   100, R4
LOOP        LOADH  R5, (R1)        ; čitaj prvi
            PUSH   R5                 ; pretvori prvi u 2'k (poziv potporgrama)
            CALL   BZP_2K
            ADD    SP, 4, SP
FORMULA     SHL    R0, 2, R6            ; računaj formulu
            ASHR   R0, 3, R5
            SUB    R5, R6, R5
            ADD    R0, R5, R0
            STORE  R0, (R2)        ; spremi rezultat
            ADD    R1, 2, R1         ; pomak pokazivača
            ADD    R2, 4, R2
            SUB    R4, 1, R4        ; kraj petlje
            JR_NZ  LOOP
            HALT                   ; kraj
-------
BZP_2K      PUSH   R1                ; spremi kontekst
            LOAD   R0, (SP+8)        ; dohvat parametra sa stoga
            ROTL   R0, %D 17, R1      ; provjera predznaka na 16. bitu
            JR_NC  VAN                ; ako je pozitivan, ne treba raditi ništa
            ; provjera predznaka može i ROTL za 16 bita pa ispitivanje JR_P
            ; a može i maskiranjem:
            ; OR     R0, 00008000, R1
            ; JR_NZ  VAN
NEGAT       ; pretvorba negativnog broja
            AND    R0, 0FFFF7FFF, R0          ; brisanje bita za predznak (16. bit)
                                              ; dovoljna je maska 7FFF (gore su ionako 0)
            XOR    R0, -1, R0                 ; dvojno komplementiranje
            ADD    R0, 1, R0
VAN         POP    R1                 ; obnovi kontekst i povratak
            RET
------
            ORG    1000            ; adresa
BLOK        DW     ...             ; labela + podatci
REZ         DS     400              ; labela + veličina bloka
```

3. FRISC (11,5 bodova) Na FRISC su spojene bezuvjetne jedinice BVJ1 i BVJ2 i uvjetne jedinice UVJ1 i UVJ2. Također su spojene prekidne jedinice PVJ1 i PVJ2 i to obje na priključak INT. Adrese jedinica odaberite sami.

FRISC treba beskonačno prenositi podatke sa BVJ1 na UVJ1 te se BVJ2 na UVJ2. Pri tome se prebraja koliko je podataka preneseno na UVJ1. Vrijednost ovog brojača šalje se na PVJ1 kad ona postavi prekid. Vrijednost brojača se vraća u 0 kada prekid postavi PVJ2 (sa PVJ2 se ne prenosi nikakav podatak).

PVJ1 je prioritetnija od PVJ2, ali se prekidi ne mogu gnijezditi.

```
B1          EQU       0FFFF0000       ; adrese vanjskih jedinica

B2          EQU       0FFFF0100

U1_D        EQU       0FFFF0200
U1_B        EQU       0FFFF0204

U2_D        EQU       0FFFF0300
U2_B        EQU       0FFFF0304

P1_D        EQU       0FFFF0400
P1_B        EQU       0FFFF0404
P1_E        EQU       0FFFF0408

P2_D        EQU       0FFFF0500
P2_B        EQU       0FFFF0504
P2_E        EQU       0FFFF0508

            ORG       0             ; početak izvođenja na adresi 0
            MOVE      10000, SP         ; inicijalizacija stoga i skok u glavni
            JR        GLAVNI

            ORG       8                 ; vektor na adresi 8
            DW        100
------

GLAVNI      MOVE      %B 10000, SR      ; dozvoli prekide

            ; između uvjetnih jedinica nema ovisnosti pa se moraju prozivati (pooling)

POOL        LOAD      R0, (U1_B)        ; ispitaj spremnost U1
            OR        R0, R0, R0
            CALL_NZ   PU1               ; posluži U1 ako je spremna, inače nastavi

            LOAD      R0, (U2_B)        ; ispitaj spremnost U2
            OR        R0, R0, R0
            CALL_NZ   PU2               ; posluži U2 ako je spremna, inače nastavi

            JR        POOL            ; povratak na početak prozivanja


; potprogram za posluživanje U1
; (može i obični odsječak, ali je lakše CALL/RET nego jumpanje)

PU1         PUSH      R0              ; spremanje konteksta

            LOAD      R0, (B1)        ; čitaj bezuvjetnu B1
            STORE     R0, (U1_D)      ; šalji na uvjetnu U1
            STORE     R0, (U1_B)      ; briši status od U1

            LOAD      R0, (BROJAC)      ; povećaj brojač poslanih podataka
            ADD       R0, 1, R0
            STORE     R0, (BROJAC)

            POP       R0              ; obnova konteksta i povratak
            RET


BROJAC      DW        0               ; brojač prenesenih podataka na U1
```

```
; potprogram za posluživanje U2

PU2         PUSH     R0              ; spremanje konteksta

            LOAD     R0, (B2)        ; čitaj bezuvjetnu B2
            STORE    R0, (U2_D)      ; šalji na uvjetnu U2
            STORE    R0, (U2_B)      ; briši status od U2

            POP      R0              ; obnova konteksta i povratak
            RET
-------



            ORG      100             ; prekidni potprogram na adresi zadanoj vektorom

PREKIDNI    PUSH     R0                ; spremanje konteksta
            MOVE     SR, R0
            PUSH     R0

            LOAD     R0, (P1_B)        ; ispitivanje tko je izazvao prekid
            OR       R0, R0, R0
            JR_NZ    PP1             ; prvo se ispituje P1 jer je prioritetnija

            ; prekid je došao sa P2
PP2         STORE    R0, (P2_B)      ; dojavi prihvat prekida na P2

            MOVE     0, R0           ; brisanje brojača
            STORE    R0, (BROJAC)

            STORE    R0, (P2_E)      ; dojavi kraj posluživanja na P2

            JR       VAN             ; izlazak iz prekidnog potprograma

            ; prekid je došao sa P1
PP1         STORE    R0, (P1_B)      ; dojavi prihvat prekida na P1

            LOAD     R0, (BROJAC)    ; čitanje brojača
            STORE    R0, (P1_D)      ; slanje vrijednosti brojača na P1

            STORE    R0, (P1_E)      ; dojavi kraj posluživanja na P1

VAN         POP      R0                ; obnova konteksta
            MOVE     R0, SR
            POP      R0
            RETI                     ; povratak iz maskirajućeg prekida
--------
```