Završni ispit iz Arhitekture računala 1 **28. lipnja 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 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. Sve teorijske zadatke rješavati na ovaj papir. Završni ispit traje 150 minuta.

1. a. (1 bod) Podatak 000111_2 u 6-bitnom NBC-u predstavlja broj ___, a u 6-bitnom formatu 2'k predstavlja broj ____. Podatak 1100_2 u 4-bitnom NBC-u predstavlja broj _____, a u 4-bitnom formatu 2'k predstavlja _____.

1. b. (1 bod) Sabirnice se prema **namjeni** dijele na: _____, _____ i _____
Prema **načinu komunikacije** sabirnice se dijele na _____ i _____

1. c. (1 bod) Koji **dio vanjske jedinice** postoji unutar uvjetnih i prekidnih, a ne postoji unutar bezuvjetnih jedinica: _____. **Priključci** koji postoje kod uvjetnih i prekidnih vanjskih jedinica (a ne postoje kod bezuvjetnih) nazivaju se _____. Ovi priključci povezuju _____ i _____.

2. a. FRISC (1,5 bod) Prilikom prihvaćanja maksirajućeg prekida FRISC mijenja zastavicu _____ u registru _____ čime se *(postiže što)* _____. Za razliku od obične naredbe RET, naredba RETI **dodatno** *(radi što)* _____, a naredba RETN **dodatno** *(radi što)* _____.
Sve tri naredbe RET, RETI i RETN u registar PC stavljaju _____.

2. b. FRISC (2,5 boda) Napišite **smjerove** sljedećih priključaka procesora FRISC: ADR je _____,
DATA je _____, READ je _____, WRITE je _____,
WAIT je_____, BREQ je _____, BACK je _____, SIZE je _____.
Čemu služi priključak WAIT? _____

2. c. FRISC (1 bod) Koja su 4 načina rada sklopa FRISC-PIO: _____
_____.

2. d. FRISC (1 bod) Kad u sklopu FRISC-CT vrijednost u brojilu postane nula, događa se sljedeće: _____
_____
_____.

3. a. ARM (1,5 bodova) Za procesor ARM7 napišite **trajanja koraka izvođenja** (u ciklusima) sljedećih naredaba:

naredbe za obradu podataka	_____	naredba LDM R13,{R1,R2,R14}	_____
naredba LDR	_____	naredba B s istinitim uvjetom skoka	_____
naredba STR	_____	naredba BL s lažnim uvjetom skoka	_____

3. b. ARM (0,5 bodova) Procesor ARM sa **statičkim predviđanjem grananja** izvodi sljedeći programski odsječak:

```
LABELA1   B  LABELA2
LABELA2   B  LABELA1
```

Zaokružite točan odgovor (a ili b):
Za **prvu** naredbu predvidjet će se da: a) će se grananje dogoditi b) se grananje neće dogoditi
Za **drugu** naredbu predvidjet će se da: a) će se grananje dogoditi b) se grananje neće dogoditi

3. c. ARM (1,5 bodova) ARM ima dva ulazna priključka za prekide: _____ i _____. Za obične prekide adresa prekidnog potprograma je _____, a za brze prekide adresa je _____. Povratak iz prekidnog potprograma izvodi se naredbom SUBS PC,LR,#4, koja obnavlja sadržaje registra (ili više njih): _____

3. d. ARM (1,5 bod) Kada se pojavi impuls na priključku CLK1HZ (ARM-ovog sklopa RTC), što se dogodi s brojilom? _____. Kada vrijednost u brojilu postane jednaka (*čemu*) _____
_____, tada se u RTC-u automatski događa sljedeće: _____
_____.

4. (6 bodova) U računalnom sustavu nalaze se procesor FRISC, 2 CT-a i dvije uvjetne jedinice. Na ulaz CNT sklopa CT1 spojen je signal frekvencije 1 MHz. Sklop CT2 generira prekid na INT3 (CT nema priključak IACK). Adrese vanjskih jedinica odaberite sami. Nacrtajte način spajanja CT-ova.

Procesor treba prenositi podatke s vanjske jedinice UVJ1 na UVJ2. Za sve prijenose podatka s jedne vanjske jedinice na drugu, potrebno je mjeriti vrijeme između dva uzastopna prijenosa podataka (razdoblje do prvog prijenosa se može i ne mora mjeriti). Ta je trajanja – izražena u sekundama – potrebno spremati kao 16-bitne podatke u memoriju od adrese 1000_{16}. Trajanja se izražavaju u cijelim sekundama.

Na ovaj je način potrebno prenijeti 14_{16} podataka i zatim zaustaviti procesor.

5. (5,5 bodova) U računalnom sustavu nalaze se procesor FRISC, DMA i dva sklopa PIO (spojeni su na priključke INT1 i INT2 i postavljaju prekide).

Sklop PIO1 radi u ulaznom načinu rada. Sklop PIO2 radi u načinu ispitivanja bitova: na 4 najviša bita prima se 4-bitni podatak; a pomoću 4 najniža vanjski uređaj dojavljuje sklopu PIO2 da je podatak valjan - tako da postavi sva 4 najniža bita u stanje 1.

Kada dođe do prekida sa PIO1, procesor treba pročitati podatak sa sklopa PIO1 i proširiti ga ništicama. Prošireni podatak pomoću sklopa DMA treba kopirati u 10_{16} 32-bitnih lokacija odredišnog bloka memorije.

Kada dođe do prekida sa PIO2, procesor treba pročitati 4-bitni podatak sa sklopa PIO2 i predznačno ga proširiti. Prošireni podatak pomoću sklopa DMA treba kopirati u 10_{16} 32-bitnih lokacija odredišnog bloka memorije.

Odredišni blok memorije počinje od adrese 1000_{16} i skupine od 10_{16} podataka se redom stavljaju jedna iza druge. DMA radi u načinu zaustavljanja procesora. Glavni program izvodi beskonačnu petlju.

6. (5,5 bodova) Za procesor ARM napišite potprogram PARNULA koji za primljeni 16-bitni podatak ispituje svih 15 parova susjednih bitova unutar podatka, i prebraja parove ništica. Na primjer: broj parova ništica u 8-bitnom broju **00**1**00**1**00** je 3, a u broju 11111**000** je 2. Potprogram PARNULA prima parametar preko fiksne memorijske lokacije TEST, a rezultat (broj parova ništica) vraća preko R0.

Napišite potprogram COMPARE koji prima dva parametra preko stoga. Za svaki parametar treba potprogramom PARNULA izračunati broj parova ništica. Potprogram COMPARE preko registra R0 vraća glavnom programu rezultat koji se računa na sljedeći način:

- 1, ako prvi parametar ima više parova ništica od drugog parametra
- -1, ako prvi parametar ima manje parova ništica od drugog parametra
- 0, ako parametri imaju jednak broj parova ništica

Pod prvim parametrom podrazumijevamo podatak koji se nalazio na lokaciji PRVI (analogno za drugi).

Glavni program treba s memorijskih lokacija PRVI i DRUGI učitati dva 16-bitna podatka, te podatak s većim brojem parova ništica zapisati na lokaciju REZULTAT. Ako podatci imaju jednak broj parova ništica, na lokaciju REZULTAT treba upisati $FFFF_{16}$.

7. (7 bodova) Računalni sustav usisivača sastoji se od procesora ARM, sklopa RTC (radi u prekidnom načinu, spojen na IRQ, na ulaz RTC-a spojen je signal od 1 kHz) i sklopa GPIO.

Na vrata A sklopa GPIO spojen je temperaturni sklop (kao na predavanjima), na sljedeći način:

- bitovi 0-5 – ulazni: temperatura motora u rasponu od 0 do 63
- bit 6 – ulazni bit: temperatura je postavljena
- bit 7 – izlazni bit: temperatura je pročitana

Na vrata B sklopa GPIO spojeno je:

- bit 0 – ulazni bit: tipka za uključivanje (1) / isključivanje (0) usisivača
- bit 1 – izlazni bit: uključivanje (1) / isključivanje (0) motora
- bitovi 2-4 – ulazni: senzor prljavštine
- bitovi 5-7 – izlazni: regulator snage motora

Napisati program koji beskonačno upravlja radom usisivača svakih 0,1 sekundi na sljedeći način. Prvo se provjerava tipka za uključivanje/isključivanje usisivača, te se na temelju stanja tipke uključuje ili isključuje motor. Ako motor treba biti uključen, snaga motora regulira se na temelju vrijednosti senzora prljavštine (npr. ako je vrijednost senzor prljavštine 000_2, snaga motora također treba biti 000_2). Međutim, dodatno treba ispitati temperaturu motora. Ako je ona strogo veća od 60 °C, snagu motora treba smanjiti na minimum (vrijednost 000_2), bez obzira na razinu prljavštine.

Završni ispit iz Arhitekture računala 1 **28. lipnja 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 službene šalabahtere (popis naredaba FRISC-a i ARM-a). Programe treba pisati uredno i komentirati pojedine cjeline programa. Sve teorijske zadatke rješavati na ovaj papir. Završni ispit traje 150 minuta.

1. a. (1 bod) Podatak 000111_2 u 6-bitnom NBC-u predstavlja broj ___**7**___, a u 6-bitnom formatu 2'k predstavlja broj __**7**__. Podatak 1100_2 u 4-bitnom NBC-u predstavlja broj _**12**____, a u 4-bitnom formatu 2'k predstavlja __**-4**___.

1. b. (1 bod) Sabirnice se prema **namjeni** dijele na: _____**adresnu**_____, _____**podatkovnu**____ i ___**upravljačku (kontrolnu)**____. Prema **načinu komunikacije** sabirnice se dijele na __**sinkronu**__ i _ **asinkronu**__.

1.c. (1 bod) Koji **dio vanjske jedinice** postoji unutar uvjetnih i prekidnih, a ne postoji unutar bezuvjetnih jedinica: ___**bistabil stanja (status bistabil)** ____. **Priključci** koji postoje kod uvjetnih i prekidnih vanjskih jedinica (a ne postoje kod bezuvjetnih) nazivaju se ____**priključci za sinkronizaciju (ili handshaking, ili rukovanje, ili READY i STROBE)**. Ovi priključci povezuju ____**vanjsku jedinicu**_____ i _____**vanjski proces (ili vanjski uređaj ili uređaj)** _____.

2. a. FRISC (1,5 bod) Prilikom prihvaćanja maksirajućeg prekida FRISC mijenja zastavicu __**GIE**__ u registru __**SR**___ čime se (postiže što) _____**zabranjuje prihvaćanje maskirajućih prekida**___. Za razliku od obične naredbe RET, naredba RETI **dodatno** (radi što) __**obnavlja zastavicu GIE** __, a naredba RETN **dodatno** (radi što) _**obnavlja (internu) zastavicu IIF**_. Sve tri naredbe RET, RETI i RETN u registar PC stavljaju _**povratnu adresu (sa stoga)**__

2. b. FRISC (2,5 boda) Napišite **smjerove** sljedećih priključaka procesora FRISC: ADR je ___**izlazni**_____, DATA je ___**dvosmjerni**___, READ je ____**izlazni**_____, WRITE je _ **izlazni** _, WAIT je_____ **ulazni** ____, BREQ je __ **ulazni** __, BACK je ____ **izlazni** _____, SIZE je ___ **izlazni** _____. Čemu služi priključak WAIT? __**pomoću njega memorija (ili VJ) dojavljuje da je spora (ili traži umetanje ciklusa čekanja, ili traži od FRISC-a da pričeka itd.)**__

2. c. FRISC (1 bod) Koja su 4 načina rada sklopa FRISC-PIO: **ulazni, izlazni, postavljanje bitova, ispitivanje bitova**

2. d. FRISC (1 bod) Kad u sklopu FRISC-CT vrijednost u brojilu postane nula, događa se sljedeće: ___**brojilo se ponovno postavlja na početnu vrijednost (ili vrijednost limit registra LR kopira se u brojilo), CT postaje spreman, CT može postaviti prekid, generira se impuls na izlaznom priključku ZC**

3. a. ARM (1,5 bodova) Za procesor ARM7 napišite **trajanja koraka izvođenja** (u ciklusima) sljedećih naredaba:

naredbe za obradu podataka	_____**1**_____	naredba LDM R13,{R1,R2,R14}	_____**5**____
naredba LDR	_____**3**_____	naredba B s istinitim uvjetom skoka	_____**3**____
naredba STR	_____**2**_____	naredba BL s lažnim uvjetom skoka	_____**1**____

3. b. ARM (0,5 bodova) Procesor ARM sa **statičkim predviđanjem grananja** izvodi sljedeći programski odsječak:

```
LABELA1     B  LABELA2
LABELA2     B  LABELA1
```

Zaokružite točan odgovor (a ili b):

Za **prvu** naredbu predvidjet će se da: a) će se grananje dogoditi **b) se grananje neće dogoditi**

Za **drugu** naredbu predvidjet će se da: **a) će se grananje dogoditi** b) se grananje neće dogoditi

3. c. ARM (1,5 bodova) ARM ima dva ulazna priključka za prekide: ___**IRQ**___ i ___**FIQ**___. Za obične prekide adresa prekidnog potprograma je __**18_{16}**___, a za brze prekide adresa je ___**$1C_{16}$**___. Povratak iz prekidnog potprograma izvodi se naredbom SUBS PC,LR,#4, koja obnavlja sadržaje registra (ili više njih): __ **PC (ili R15) i CPSR**_____.

3. d. ARM (1,5 bod) Kada se pojavi impuls na priključku CLK1HZ (ARM-ovog sklopa RTC), što se dogodi s brojilom? ____**brojilo se poveća za jedan**__. Kada vrijednost u brojilu postane jednaka (*čemu*) _____ **vrijednosti u registru usporedbe (MR)** _____, tada se u RTC-u automatski događa sljedeće: _____**RTC postaje spreman i RTC može postaviti zahtjev za prekid** ___.

4. (6 bodova) U računalnom sustavu nalaze se procesor FRISC, dva CT-a i dvije uvjetne jedinice. Na ulaz CNT sklopa CT1 spojen je signal frekvencije 1 MHz. Sklop CT2 generira nemaskirajući prekid NMI. Adrese vanjskih jedinica odaberite sami. Nacrtajte način spajanja CT-ova.

Procesor treba prenositi podatke s vanjske jedinice UVJ1 na UVJ2. Za sve prijenose podatka s jedne vanjske jedinice na drugu, potrebno je mjeriti vrijeme između dva uzastopna prijenosa podataka (razdoblje do prvog prijenosa se može i ne mora mjeriti). Ta je trajanja – izražena u sekundama – potrebno spremati kao 16-bitne podatke u memoriju od adrese 1000_{16}. Trajanja se izražavaju u cijelim sekundama.

Na ovaj je način potrebno prenijeti 14_{16} podataka i zatim zaustaviti procesor.

```
UVJ1_PRIMI  EQU    0FFFF1000  ; adrese vanjskih jedinica
UVJ1_STANJE EQU    0FFFF1004

UVJ2_PRIMI  EQU    0FFFF2000
UVJ2_STANJE EQU    0FFFF2004

CT1CR       EQU    0FFFF3000
CT1LR       EQU    0FFFF3004
CT1ACK      EQU    0FFFF3008
CT1END      EQU    0FFFF300C

CT2CR       EQU    0FFFF4000
CT2LR       EQU    0FFFF4004
CT2ACK      EQU    0FFFF4008
CT2END      EQU    0FFFF400C

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

      ORG 0C                  ; prekidni potprogram za NMI
      PUSH  R0                ; spremanje konteksta
      MOVE  SR, R0
      PUSH  R0

      STORE R0,(CT2ACK)       ; dojava prihvata prekida
      LOAD  R0, (SEKUNDE)     ; broj sekundi u ovom trajanju
      ADD   R0, 1, R0         ; učitava se iz memorije
      STORE R0, (SEKUNDE)     ; mijenja i ponovno sprema
      STORE R0, (CT2END)      ; dojava kraja posluživanja

      POP   R0                ; vraćanje konteksta
      MOVE  R0, SR
      POP   R0
      RETN                    ; povratak iz prekidnog potprograma

GLAVNI
      MOVE  %D 1000, R0       ; inicijalizacija vremenskih konstanata
      STORE R0, (CT1LR)
      STORE R0, (CT2LR)

      MOVE  %B01 R0           ; inicijalizacija CT-ova
      STORE R0, (CT1CR)       ; ne postavlja prekid

      MOVE  %B111, R0         ; postavlja prekid
      STORE R0, (CT2CR)

      MOVE  14, R1            ; R1 – brojač prenesenih podataka
      MOVE  1000, R3          ; R3 – adresa liste pojedinih trajanja

UVJ1  LOAD  R0, (UVJ1_STANJE) ; čekanje na spremnost UVJ1
      OR    R0, R0, R0
      JP_Z  UVJ1

      LOAD  R5, (UVJ1)        ; čitanje podatka
      STORE R0, (UVJ1_STANJE) ; brisanje spremnosti
```

```
UVJ2  LOAD  R0, (UVJ2_STANJE) ; čekanje na spremnost UVJ2
      OR    R0, R0, R0
      JP_Z  UVJ2

      STORE R5, (UVJ2)        ; slanje podatka
      STORE R0, (UVJ2_STANJE) ; brisanje spremnosti

      CMP   R1, 14            ; je li ovo prvi prijenos?
      JP_EQ PRVI

      LOAD  R0, (SEKUNDE)     ; trajanje između ova dva prijenosa
      STOREH R0, (R3)         ; spremanje u listu
      ADD   R3, 2, R3         ; povećanje pokazivača na listu
      MOVE  0, R0             ; vraćanje trajanja na 0 za sljedeće
      STORE R0, (SEKUNDE)     ; mjerenje
      JP    DALJE

PRVI  MOVE  0, R0             ; prvo brojanje se ne mjeri
      STORE R0, (SEKUNDE)

DALJE SUB   R1, 1, R1         ; smanjivanje brojača podataka
      JP_NZ UVJ1              ; ima li još podataka? i skok na početak

KRAJ  HALT

SEKUNDE     DW 0              ; brojač sekundi između dva prijenosa

; Alternativno rješenje (nešto preciznije) je da se CT-ovi pokrenu iznova svaki put kad
; se napravi prijenos (tj. kada UVJ2 postane spremna). Sve ostalo bi bilo isto.
```

5. (5,5 bodova) U računalnom sustavu nalaze se procesor FRISC, DMA i dva sklopa PIO koji generiraju maskirajuće prekide INT.

Sklop PIO1 radi u ulaznom načinu rada. Sklop PIO2 radi u načinu ispitivanja bitova: na 4 najviša bita prima se 4-bitni podatak; a pomoću 4 najniža vanjski uređaj dojavljuje sklopu PIO2 da je podatak valjan - tako da postavi sva 4 najniža bita u stanje 1.

Kada dođe do prekida sa PIO1, procesor treba pročitati podatak sa sklopa PIO1 i proširiti ga ništicama. Prošireni podatak pomoću sklopa DMA treba kopirati u 10_{16} 32-bitnih lokacija odredišnog bloka memorije.

Kada dođe do prekida sa PIO2, procesor treba pročitati 4-bitni podatak sa sklopa PIO2 i predznačno ga proširiti. Prošireni podatak pomoću sklopa DMA treba kopirati u 10_{16} 32-bitnih lokacija odredišnog bloka memorije.

Odredišni blok memorije počinje od adrese 1000_{16} i skupine od 10_{16} podataka se redom stavljaju jedna iza druge. DMA radi u načinu zaustavljanja procesora. Glavni program izvodi beskonačnu petlju.

```
PIO1_CR     EQU   FFFF1000
PIO1_DR     EQU   FFFF1004
PIO1_IACK   EQU   FFFF1008
PIO1_IEND   EQU   FFFF100C

PIO2_CR     EQU   FFFF2000
PIO2_DR     EQU   FFFF2004
PIO2_IACK   EQU   FFFF2008
PIO2_IEND   EQU   FFFF200C

DMA_SRC     EQU   FFFF3000
DMA_DST     EQU   FFFF3004
DMA_SIZE    EQU   FFFF3008
DMA_CTRL    EQU   FFFF300C
DMA_START   EQU   FFFF3010
DMA_ACK     EQU   FFFF3014

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

```
      ORG 8             ; prekidni vektor za maskirajući prekid
      DW    500

GLAVNI
      MOVE  %B 0101,R0        ; inicijalizacija PIO1 (INT, prekid, ULAZNI način)
      STORE R0, (PIO1_CR)

      MOVE %B 11110000111100010111, R0    ; aktivna razina najniža 4 bita = 1
      STORE R0, (PIO2_CR)                 ; MASKA = ispituju se najniža 4 bita
                                          ; AND, INT, prekid, ispitivanje bitova

      MOVE  %B 0100, R0       ; DMA bez INT, zaust. procesora, vj->mem
      STORE R0, (DMA_CTRL)

      MOVE  ADRESA, R0
      STORE R0, (DMA_SRC)     ; adresa s koje će se kopirati podatak

      MOVE  %B 10000, SR      ; GIE - omogućavanje prekida

PETLJA JP   PETLJA            ; beskonačna petlja glavnog programa


      ORG   500               ; adresa prekidnog potprograma
      PUSH  R0                ; čuvanje konteksta
      MOVE  SR, R0
      PUSH  R0

      LOAD  R0, (PIO2_CR)     ; tko je tražio prekid: PIO1 ili PIO2?
      CMP   R0, 1             ; (mogu se ispitivati i virtualni bitovi SR-a)
      JP_EQ PIO2_INT

PIO1_INT
      STORE R0, (PIO1_IACK)   ; dojava prihvata prekida
      LOAD  R0, (PIO1_DR)     ; čitanje podatka s PIO1
                              ; proširenje ništicama je "već napravljeno"
      STORE R0, (ADRESA)      ; spremanje u memorijsku lokaciju
      STORE R0, (PIO1_IEND)   ; kraj obrade prekida
      JP    DMA

PIO2_INT
      STORE R0, (PIO2_IACK)   ; dojava prihvata prekida
      LOAD  R0, (PIO2_DR)     ; uzimanje podatka s PIO2
      SHL   R0, %D 24, R0     ; predznačno proširenje
      ASHR  R0, %D 28, R0     ; i kasnije pomak za dodatna 4 bita udesno
      STORE R0, (ADRESA)
      STORE R0, (PIO2_IEND)   ; kraj obrade prekida

DMA   MOVE  10, R0
      STORE R0, (DMA_SIZE)    ; koliko podataka se kopira preko DMA

      STORE R0, (DMA_START)   ; pokretanje DMA prijenosa sa zaustavljanjem procesora

      LOAD  R0, (ADRESA_BLOKA); učitavanje adrese na koju se kopira
      ADD   R0, 40, R0        ; pomicanje pokazivača u ciljnom bloku
      STORE R0, (ADRESA_BLOKA); i njegovo spremanje na mem. lokaciju

      STORE R0, (DMA_DST)     ; postavljanje te adrese u DMA_DST registar
                              ; (može se samo jednom inicijalizirati DMA_DST na
                              ; 1000 u glavnom programu, jer se on automatski
                              ; povećava tijekom DMA-prijenosa za 40)
      STORE R0, (DMA_ACK)     ; brisanje statusa DMA-sklopa

      POP   R0                ; obnova konteksta
      MOVE  R0, SR
      POP   R0
      RETI                    ; povratak iz prekidnog potprograma

ADRESA      DW    0           ; pomoćna lokacija; izvor podataka za DMA
ADRESA_BLOKA DW 1000          ; adresa odredišnog bloka podataka
```

6. (5,5 bodova) Za procesor ARM napišite potprogram PARNULA koji za primljeni 16-bitni podatak ispituje svih 15 parova susjednih bitova unutar podatka, i prebraja parove ništica. Na primjer: broj parova ništica u 8-bitnom broju **00**1**00**1**00** je 3, a u broju 11111**000** je 2. Potprogram PARNULA prima parametar preko fiksne memorijske lokacije TEST, a rezultat (broj parova ništica) vraća preko R0.

Napišite potprogram COMPARE koji prima dva parametra preko stoga. Za svaki parametar treba potprogramom PARNULA izračunati broj parova ništica. Potprogram COMPARE preko registra R0 vraća glavnom programu rezultat koji se računa na sljedeći način:

- 1, ako prvi parametar ima više parova ništica od drugog parametra
- -1, ako prvi parametar ima manje parova ništica od drugog parametra
- 0, ako parametri imaju jednak broj parova ništica

Pod prvim parametrom podrazumijevamo podatak koji se nalazio na lokaciji PRVI (analogno za drugi).

Glavni program treba s memorijskih lokacija PRVI i DRUGI učitati dva 16-bitna podatka, te podatak s većim brojem parova ništica zapisati na lokaciju REZULTAT. Ako podatci imaju jednak broj parova ništica, na lokaciju REZULTAT treba upisati $FFFF_{16}$.

```
      ORG 0
      MOV   R13,  #10<12      ; inicijalizacija stoga

      LDRH  R1, PRVI          ; učitavanje podataka
      LDRH  R2, DRUGI

      STMFD R13!, {R1, R2}    ; stavljanje parametara na stog
      BL    COMPARE           ; poziv potprograma
      ADD   R13, R13, #8      ; čišćenje stoga

      CMP   R0, #0            ; usporedba rezultata s 0
      STRGTH R1, REZULTAT     ; veći – PRVI je rezultat
      STRLTH R2, REZULTAT     ; manji – DRUGI je rezultat
      MVNEQ R0, #0            ; jednaki – stavljamo FFFF = MVN(0)
      STREQH R0, REZULTAT
      SWI 123456

COMPARE
      STMFD R13!, {R1, R2, R14}; spremanje konteksta (i R14!)
      LDR   R1, [R13, #0C]    ; učitavanje prvog parametra (12 bajtova udaljen)
      STRH  R1, TEST          ; spremanje parametra na memorijsku lokaciju TEST
      BL    PARNULA           ; poziv potprograma
      MOV   R1, R0            ; čuvanje rezultata u R1 za usporedbu

      LDR   R2, [R13, #10]    ; učitavanje drugog parametra (16 bajtova udaljen)
      STRH  R2, TEST          ; spremanje parametra na memorijsku lokaciju TEST
      BL    PARNULA           ; poziv potprograma
      MOV   R2, R0            ; čuvanje rezultata u R2 za usporedbu

      CMP   R1, R2            ; usporedba 2 rezultata dobivena od PARNULA
      MOVEQ R0, #0            ; rezultat se vraća preko R0, a postavlja se...
      MOVGT R0, #1            ; ...na temelju zastavica od naredbe CMP
      MVNLT R0, #0            ; MVN #0 daje –1

      LDMFD R13!, {R1, R2, R14}; vraćanje konteksta
      MOV   PC, LR            ; povratak iz potprograma

PARNULA
      STMFD R13!, {R1, R2, R3}; spremanje konteksta
      LDRH  R1, TEST          ; učitavanje parametra
      MOV   R2, #%D 15        ; 15 puta vrtiti petlju (15 parova)
      MOV   R0, #0            ; brojač parova nula

PETLJA      AND R3, R1, #%B11 ; brisanje svih bitova osim zadnja 2
      CMP   R3, #0            ; je li dobiveni podatak = 0?
      ADDEQ R0, R0, #1        ; ako jest -> to je par nula, treba povećati brojač
      MOV   R1, R1, LSR #1    ; pomak udesno za 1 bit
      SUBS  R2, R2, #1        ; smanjivanje brojača pomaka (brojača petlje)
      BNE   PETLJA
```

```
      LDMFD R13!, {R1, R2, R3}; vraćanje konteksta
      MOV   PC, LR            ; povratak iz potprograma

REZULTAT    DW 0
PRVI        DW 10
DRUGI       DW 20
TEST        DW 0
```

7. (7 bodova) Računalni sustav usisivača sastoji se od procesora ARM, sklopa RTC (radi u prekidnom načinu, spojen na IRQ, na ulaz RTC-a spojen je signal od 1 kHz) i sklopa GPIO.

Na vrata A sklopa GPIO spojen je temperaturni sklop (kao na predavanjima), na sljedeći način:

- bitovi 0-5 – ulazni: temperatura motora u rasponu od 0 do 63
- bit 6 – ulazni bit: temperatura je postavljena
- bit 7 – izlazni bit: temperatura je pročitana

Na vrata B sklopa GPIO spojeno je:

- bit 0 – ulazni bit: tipka za uključivanje (1) / isključivanje (0) usisivača
- bit 1 – izlazni bit: uključivanje (1) / isključivanje (0) motora
- bitovi 2-4 – ulazni: senzor prljavštine
- bitovi 5-7 – izlazni: regulator snage motora

Napisati program koji beskonačno upravlja radom usisivača svakih 0,1 sekundi na sljedeći način. Prvo se provjerava tipka za uključivanje/isključivanje usisivača, te se na temelju stanja tipke uključuje ili isključuje motor. Ako motor treba biti uključen, snaga motora regulira se na temelju vrijednosti senzora prljavštine (npr. ako je vrijednost senzor prljavštine 000_2, snaga motora također treba biti 000_2). Međutim, dodatno treba ispitati temperaturu motora. Ako je ona strogo veća od 60 °C, snagu motora treba smanjiti na minimum (vrijednost 000_2), bez obzira na razinu prljavštine.

```
      ORG   0
      B     GLAVNI                  ; skok na glavni program

      ORG   18                      ; prekidni potprogram – IRQ
                                    ; (skok može, ali ne mora jer se FIQ ne koristi)
      STMFD R13!, {R0,R1,R2,R3,R4}  ; spremanje konteksta
      LDR   R0, GPIO                ; učitavanje adresa RTC i GPIO
      LDR   R1, RTC

      ; reinicijalizacija RTC-a
      MOV   R2, #0
      STR   R2, [R1,#0C]            ; resetiranje brojača
      STR   R2, [R1,#8]             ; dojava prihvata prekida

      ; provjera tipke za uključivanje motora
      LDR   R2, [R0,#4]             ; čitanje podataka s vrata B
      ANDS  R3, R2, #1              ; ispitati stanje tipke (najniži bit)

      MOVEQ R3, #0                  ; ako je tipka==0, motor se samo isključi...
      BEQ   SALJI                   ; ...i ne treba daljnja regulacija

      ; Inače motor treba biti uključen, i treba da regulirati.
      ; Kopiraj stanje senzora na bitove za snagu motora (+ uključi motor)
      AND   R3, R2, #%B 00011100    ; bitovi 2–4 su senzor prljavštine
      MOV   R3, R3, LSL #3          ; pomak na mjesto regulatora snage...
      OR    R3, R3, #%B 00000010    ; ...i dodajemo bit za uključivanje motora


      ; čitanje i provjera temperature (na vratima A)
CEKAJ LDR   R2, [R0]                ; čekanje na novu temperaturu
      ANDS  R4, R2, #%B 01000000    ; bit 6 je signal nove temp.
      BEQ   CEKAJ                   ; čekanje dok je signal u niskom stanju

      AND   R4, R2, #%B 00111111    ; bitovi 0–5 su temperatura

      MOV   R2, #%B 10000000        ; generiranje impulsa na bitu 7...
      STR   R2, [R0]                ; (može i naredbama ORR i AND)
```

```
      MOV   R2, #%B 00000000
      STR   R2, [R0]

      CMP   R4, #%D60              ; je li temperatura veća od 60 stupnjeva?
      ANDHI R3, #%B 00011111       ; ako jeste (Higher), spusti snagu na 000

SALJI STR   R3, [R0,#4]            ; slanje snage motora i stanja...
                                   ; uključenosti/isključenosti na vrata B

VAN   LDMFD       R13!, {R0,R1,R2,R3,R4}  ; obnova konteksta
      SUBS  PC, LR, #4             ; povratak


GLAVNI      MOV   R13,#10<12       ; inicijalizacija stoga
      LDR   R0, GPIO               ; bazna adresa sklopa GPIO
      LDR   R1, RTC                ; bazna adresa sklopa RTC

      MOV   R2, #%B10000000        ; bit 7 je izlazni
      STR   R2, [R0,#8]            ; registar smjera vrata A
      MOV   R2, #%B 00011101       ; bitovi 1,5,6,7 izlazni, 0,2,3,4 ulazni
      STR   R2, [R0,#0C]           ; registar smjera vrata B

      MOV   R2, #1                 ; omogućavanje prekida u RTC-u
      STR   R2, [R1,#10]           ; upis u upravljački registar (RTCCR)
      MOV   R2, #%D100             ; konstanta = 0,1 sekunda
      STR   R2, [R1,#4]            ; upis u RTCMR
      MOV   R2, #0                 ; inicijalizacija brojila (nije nužno)
      STR   R2, [R1,#0C]           ; upis u RTCLR

      MRS   R0, CPSR               ; omogućavanje prihvata IRQ-a
      BIC   R0, R0, #80
      MSR   CPSR_c, R0

PETLJA      B     PETLJA           ; prazna petlja


GPIO        DW    0FFFF1000        ; adresa GPIO
RTC         DW    0FFFF2000        ; adresa RTC
```