Završni ispit iz Arhitekture računala 1 — 30. lipnja 2016.

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. Završni ispit traje 135 minuta i nosi 40 bodova.

1.a (1,5 bod) 4-bitna ALU oduzima binarne brojeve 0101-1000. Rezultat oduzimanja je _____. Napišite stanja zastavica poslije oduzimanja: Prijenos=____, Posudba=____, Preljev=____, Ništica=____, Predznak=____. **Mora se vidjeti postupak.**

1.b (1 bod) 3-bitni podatak 101_2 u formatu NBC predstavlja broj _____, a u formatu 2'k predstavlja broj _____**.** 4-bitni podatak 0111_2 u formatu NBC predstavlja broj _____, a u formatu 2'k predstavlja broj _____. **Mora se vidjeti postupak.**

1.c (1,5 boda) Osim priključaka za spajanje na procesor FRISC, sklop FRISC-CT ima ulazni priključak _____ te izlazni priključak _____ . Kada se na ulaznom priključku pojavi impuls, onda se _____ (što se događa), ali samo uz preduvjet da je u upravljačkom registru na bitu _____ (ime bita) upisana vrijednost _____. Kad brojilo dođe do nule, što se događa na izlaznom priključku: _____

1.d (2,5 bod) Na desnoj strani na prazne crte napišite korake koje ARM automatski izvodi prilikom prihvaćanja prekida FIQ (može opisno ili navesti stanja registara).

Iz potprograma za FIQ povratak se ostvaruje naredbom _____ koja obnavlja sadržaj registara _____ i _____.

_____

_____

_____

_____

_____

_____

_____

1.e (2 boda): Izvođenje ovog odsječka na procesoru ARM7 traje:_____ ciklusa. Kraj svake naredbe napišite koliko puta se izvodi i koliko traje pojedino izvođenje (npr. 3 x 1c + 1 x 2c znači da naredba tri puta traje 1 ciklus i jednom traje 2 ciklusa).

```
      MOV R4,#4
      MOV R1,#10<8
LOOP  LDRB R0,[R1]
      ADD R0,R0,#1
      STRB R0,[R1],#1
      SUBS R4,R4,#1
      BNE LOOP
```

_____

_____

_____

_____

_____

_____

_____

1.f (2,5 boda) Memorijske i UI-sabirnice mogu biti spojene na dva načina. Navedite imena od ova dva načina spajanja i nadocrtajte sabirnice i način spajanja na obje sheme.

ime: _____ ime: _____

1.g (2 boda): U registru R0 nalazi se 32-bitni podatak u formatu s bitom za predznak. Napišite programski odsječak koji računa apsolutnu vrijednost tog broja. Ne smiju se koristiti dodatni registri ni memorijske lokacije. Odsječak napišite i za FRISC i za ARM. Na svaku crtu smijete upisati najviše jednu naredbu.

frisc: _____ arm: _____

_____ _____

_____

2. (10 bodova) Na **FRISC** su spojeni GPIO, CT, DMA i bezuvjetna vanjska jedinica BVJ (adrese im odaberite sami).

Glavni program treba u beskonačnoj petlji primati 8-bitne NBC podatke sa sklopa GPIO (na uvjetni način). U 16-bitnoj lokaciji ZBROJ treba računati zbroj svih podataka do tada primljenih sa sklopa GPIO. Ako dođe do prekoračenja 16-bitnog NBC opsega, u lokaciju GRESKA treba upisati 1 i normalno nastaviti sa primanjem i zbrajanjem podataka. Ako nikada nije bilo prekoračenja, u lokaciji GRESKA treba biti vrijednost 0.

Svake stotinke sekunde treba pomoću DMA-prijenosa prenijeti 100_{16} podataka iz BVJ1 u memorijski blok na adresi 7000_{16}. DMA treba raditi krađom ciklusa i javljati kraj prekidom INT. Mjerenje stotinke treba ostvariti sklopom CT na čiji ulaz je spojena frekvencija 2 MHz, a CT treba davati prekid na priključku INT.

Kada DMA-prijenos završi, treba pozvati potprogram OBRADI koji će obraditi podatke u bloku (potprogram OBRADI ne treba pisati). Pretpostavite da DMA-prijenos skupa s potprogramom OBRADI traju kraće od jedne stotinke.

Kada se primi 3333_{16} prekida sa CT-a, treba odmah zaustaviti rad CT-a i ne pokretati nove DMA-prijenose, a također treba zaustaviti i glavni program ali ne odmah, nego tek nakon što se dovrši trenutno primanje podatka sa sklopa GPIO i računanje novog zbroja.

3. (7 bodova) Za **ARM** napišite program koji obrađuje svaki 16-bitni podatak u formatu 2'k koji se nalazi u bloku memorije na adresi 1000_{16}, a 32-bitne rezultate obrade sprema u memorijski blok na adresi 2000_{16}. Početni blok sadrži 80_{16} podataka. Obradu pojedinog podatka treba raditi potprogramom POTP.

Napišite potprogram POTP koji prima 32-bitni 2'k parametar pomoću stoga i vraća 32-bitni rezultat registrom R0. POTP izračunava formulu:

rezultat = (parametar+7)^3 * konstanta + 128_{16}

Kubiranje se mora izvesti pomoću potprograma KUB. "Konstanta" je 8-bitni 2'k broj spremljen na adresi 7008_{16} (na tu adresu stavite broj -36_{16}). Pretpostavite da ni jedan izračun neće prekoračiti 32-bitni opseg 2'k brojeva.

Potprogram KUB prima parametar (32-bitni 2'k) pomoću lokacije iza naredbe BL, a 32-bitni rezultat vraća pomoću registra R0. KUB treba samo kubirati primljeni parametar.

4. (10 bodova) Računalni sustav za grijanje bazena 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. Adrese sklopova odaberite sami.

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

- bitovi 0-5 – ulazni: temperatura vode u bazenu u rasponu od 0 do 63 stupnja celzija
- bit 6 – ulazni bit: temperatura je valjana
- bit 7 – izlazni bit: temperatura je pročitana

Na vrata B sklopa GPIO spojeni su prekidač i dva grijača (bitovi 3-7 se ne koriste):

- bit 0 – ulazni bit: prekidač za uključivanje (1) / isključivanje (0) sustava grijanja
- bit 1 – izlazni bit: uključivanje (1) / isključivanje (0) malog grijača
- bit 2 – izlazni bit: uključivanje (1) / isključivanje (0) velikog grijača

Napišite program koji beskonačno kontrolira temperaturu u bazenu na sljedeći način:

(1) Svakih 10 sekundi provjeri se je li sustav isključen. Ako je sustav isključen, grijače treba isključiti. Također potprogramu SUSTAV treba poslati parametar 0. Potprogram SUSTAV ne treba pisati, a parametar mu se šalje registrom R2.

(2) Ako je sustav uključen treba potprogramu SUSTAV poslati parametar 1. Zatim treba očitati trenutačnu temperaturu vode T. Ako je T>=24, treba isključiti grijače. Ako je T <= 20, treba držati uključena oba grijača. Ako je 21 <= T <= 23 stupnja, treba držati uključen samo mali grijač.

Dok ne protekne prvih 10 sekundi, grijači trebaju biti isključeni. Glavni program za vrijeme regulacije temperature vrti beskonačnu praznu petlju.

RJEŠENJA

1.a (1,5 bod) 4-bitna ALU oduzima binarne brojeve 0101-1000. Rezultat oduzimanja je **1101**. Napišite stanja zastavica poslije oduzimanja: Prijenos= **0**, Posudba= **1**, Preljev= **1**, Ništica= **0**, Predznak= **1**. **Mora se vidjeti postupak.**

1.b (1 bod) 3-bitni podatak 101_2 u formatu NBC predstavlja broj **5**, a u formatu 2'k predstavlja broj **-3**. 4-bitni podatak 0111_2 u formatu NBC predstavlja broj **7**, a u formatu 2'k predstavlja broj **7**. **Mora se vidjeti postupak.**

1.c (1,5 boda) Osim priključaka za spajanje na procesor FRISC, sklop FRISC-CT ima ulazni priključak **CNT** te izlazni priključak **ZC** . Kada se na ulaznom priključku pojavi impuls, onda se **smanjuje brojilo (DC) za jedan** (što se događa), ali samo uz preduvjet da je u upravljačkom registru na bitu **STOP/START** (ime bita) upisana vrijednost **1**. Kad brojilo dođe do nule, što se događa na izlaznom priključku: **generira se impuls**

1.d (2,5 bod) Na desnoj strani na prazne crte napišite korake koje ARM automatski izvodi prilikom prihvaćanja prekida FIQ (može opisno ili navesti stanja registara).

Iz potprograma za FIQ povratak se ostvaruje naredbom **SUBS PC, LR, #4** koja obnavlja sadržaj registara **PC** i **CPSR**.

R14_fiq = adresa sljedeće naredbe + 4

SPSR_fiq = CPSR

CPSR[4:0] = %B10001 ili ulazak u način rada fiq

CPSR[5] = 0 ili ulazak u ARM način (izlazak iz Thumb načina)

CPSR[6] = 1 ili zabrani prekide FIQ

CPSR[7] = 1 ili zabrani prekide IRQ

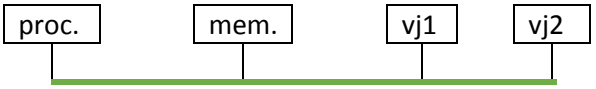
PC = 1C ili skok na adresu 1C

1.e (2 boda): Izvođenje ovog odsječka na procesoru ARM7 traje: **42** ciklusa. Kraj svake naredbe napišite koliko puta se izvodi i koliko traje pojedino izvođenje (npr. 3 x 1c + 1 x 2c znači da naredba tri puta traje 1 ciklus i jednom traje 2 ciklusa).

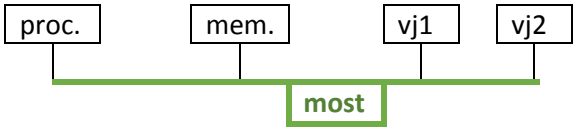
```
      MOV R4,#4          1 + 2 punjenje
      MOV R1,#10<8       1
LOOP  LDRB R0,[R1]       4x3
      ADD R0,R0,#1       4x1
      STRB R0,[R1],#1    4x2
      SUBS R4,R4,#1      4x1
      BNE LOOP           3x3 + 1x1
```

1.f (2,5 boda) Memorijske i UI-sabirnice mogu biti spojene na dva načina. Navedite imena od ova dva načina spajanja i nadocrtajte sabirnice i način spajanja na obje sheme.

ime: **zajednička (memorijska i UI) sabirnica ili backplane**



ime: **neizravno spajanje ili spajanje mostom (međusklopom)**



1.g (2 boda): U registru R0 nalazi se 32-bitni podatak u formatu s bitom za predznak. Napišite programski odsječak koji računa apsolutnu vrijednost tog broja. Ne smiju se koristiti dodatni registri ni memorijske lokacije. Odsječak napišite i za FRISC i za ARM. Na svaku crtu smijete upisati najviše jednu naredbu.

frisc: **ROTL R0, 1, R0** **ili kraće:**

AND R0, -2, R0 **SHL/ROTL R0,1,R0**

ROTR R0, 1, R0 **SHR R0,1,R0**

arm: **MOV R0, R0, LSL #1**

MOV R0, R0, LSR #1

ili kraće: BIC R0,R0,#80<24 //#8<28 ili #2<30...

2. (10 bodova) Na **FRISC** su spojeni GPIO, CT, DMA i bezuvjetna vanjska jedinica BVJ (adrese im odaberite sami).

Glavni program treba u beskonačnoj petlji primati 8-bitne NBC podatke sa sklopa GPIO (na uvjetni način). U 16-bitnoj lokaciji ZBROJ treba računati zbroj svih podataka do tada primljenih sa sklopa GPIO. Ako dođe do prekoračenja 16-bitnog NBC opsega, u lokaciju GRESKA treba upisati 1 i normalno nastaviti sa primanjem i zbrajanjem podataka. Ako nikada nije bilo prekoračenja, u lokaciji GRESKA treba biti vrijednost 0.

Svake stotinke sekunde treba pomoću DMA-prijenosa prenijeti 100_{16} podataka iz BVJ1 u memorijski blok na adresi 7000_{16}. DMA treba raditi krađom ciklusa i javljati kraj prekidom INT. Mjerenje stotinke treba ostvariti sklopom CT na čiji ulaz je spojena frekvencija 2 MHz, a CT treba davati prekid na priključku INT.

Kada DMA-prijenos završi, treba pozvati potprogram OBRADI koji će obraditi podatke u bloku (potprogram OBRADI ne treba pisati). Pretpostavite da DMA-prijenos skupa s potprogramom OBRADI traju kraće od jedne stotinke.

Kada se primi 3333_{16} prekida sa CT-a, treba odmah zaustaviti rad CT-a i ne pokretati nove DMA-prijenose, a također treba zaustaviti i glavni program ali ne odmah, nego tek nakon što se dovrši trenutno primanje podatka sa sklopa GPIO i računanje novog zbroja.

```
PIOD        EQU    0FFFF0000              ; adrese vanjskih jedinica
PIOCR       EQU    0FFFF0004
PIOACK      EQU    0FFFF0008
PIOEND      EQU    0FFFF000C

CTLR        EQU    0FFFF1000
CTCR        EQU    0FFFF1004
CTIACK      EQU    0FFFF1008
CTIEND      EQU    0FFFF100C

DMASRC      EQU    0FFFF2000
DMADEST     EQU    0FFFF2004
DMACNT      EQU    0FFFF2008
DMACR       EQU    0FFFF200C
DMASTART    EQU    0FFFF2010
DMAIACK     EQU    0FFFF2014

BVJ         EQU    0FFFF3000


            ORG    0                      ; početna adresa izvođenja
            MOVE   10000, SP              ; inicijaliziraj pokazivač stoga
            JP     GLAVNI                 ; skok u glavni

            ORG    8                      ; adresa vektora
            DW     200                    ; vektor


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

GLAVNI      MOVE   %D 20000, R0           ; vremenska konstanta za CT
            STORE  R0, (CTLR)
            MOVE   %B 011, R0             ; upravljačka riječ za CT
            STORE  R0, (CTCR)

            MOVE   %B 001, R0             ; bez prekida, ulazni način
            STORE  R0, (PIOCR)            ; upravljačka riječ za GPIO

            MOVE   %B 10000, SR           ; dozvoli prekid INT

CEKAJ_PIO   LOAD   R0, (PIOACK)           ; čekaj spremnost GPIO-a
            CMP    R0, 0
            JR_EQ  CEKAJ_PIO

            STORE  R0, (PIOACK)           ; dojava početka posluživanja
            LOAD   R0, (PIOD)             ; učitavanje podatka sa GPIO-a
            STORE  R0, (PIOEND)           ; dojava kraja posluživanja
```

```
          LOADH R1, (ZBROJ)              ; dohvat 16-bitnog zbroja i uvećanje
          ADD   R1, R0, R1
          STOREH R1, (ZBROJ)

          AND   R1, 10000, R1            ; ispitivanje 16-bitnog prijenosa
          JR_Z  OK_ZBROJ                 ; tj. ispitivanje prekoračenja opsega

GRESKA    MOVE  1, R2                    ; postavljanje oznake greške
          STORE R2, (GRESKA)

OK_ZBROJ  LOAD  R0, (BROJAC)             ; ispitivanje uvjeta zaustavljanja
          CMP   R0, 3333
          JR_GE CEKAJ_PIO                ; skok na posluživanje GPIO-a
                                         ; Može i EQ, ali točnije je GE jer
                                         ; dok se čeka spremnost PIO-a, može
                                         ; doći još koji prekid od CT-a.

          HALT                           ; zaustavljanje programa

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

BROJAC    DW    0   ; brojač prekida sa CT-a

GRESKA    DW    0   ; oznaka greške
ZBROJ     DH    0   ; zadani zbroj

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

          ORG   200                      ; adresa prekidnog potprograma

          PUSH  R0                       ; spremanje ispravnih registara
          MOVE  SR, R0
          PUSH  R0

          ; otkrivanje uzročnika prekida: CT ili DMA  (NE ČEKATI U PETLJI)
          LOAD  R0, (CTIACK)
          CMP   R0, 1          ; mogla se ispitati i spremnost DMA-sklopa
          JR_EQ POSLUZI_CT     ; ako je CT tražio prekid, posluži ga

POSLUZI_DMA      ; u suprotnom je prekid došao od DMA pa posluži DMA
          STORE R0, (DMAIACK)            ; dojavi prihvat prekida, nema dojave
                                         ; kraja posluživanja
          CALL  OBRADI                   ; poziv zadanog potprograma
          JR    VAN_2                    ; izlazak

POSLUZI_CT       ; posluživanje CT-a svake stotinke
          STORE R0, (CTIACK)             ; dojavi prihvat prekida
          LOAD  R0, (BROJAC)             ; povećaj brojač u memoriji
          ADD   R0, 1, R0
          STORE R0, (BROJAC)

          CMP   R0, 3333                 ; provjera brojača - je li kraj?
          JR_NE INIT_DMA

          ; ako je kraj, zaustavi rad CT-a
STOP_CT   MOVE  %B 000, R0               ; zaustavi brojanje CT-a
          STORE R0, (CTCR)
          JR    VAN                      ; izlazak

          ; Ako nije kraj, pokreni DMA (inicijaliziraj ga).
          ; Ovdje treba minimalno inicijalizirati adresu odredišta (memorije)
          ; i broj podataka jer se oni mijenjaju svakim DMA-prijenosom. Adresa
          ; izvora (VJ) i kontrolna riječ mogu se inicijalizirati samo jednom
          ; u glavnom programu.
INIT_DMA  MOVE  BVJ1, R0
          STORE R0, (DMASRC)             ; adresa izvora
          MOVE  7000, R0
          STORE R0, (DMADEST)            ; adresa odredišta
          MOVE  100, R0
          STORE R0, (DMACNT)             ; broj podataka
```

```
          MOVE   %B 0111, R0              ; vj->mem, krađa, INT
          STORE  R0, (DMACR)              ; upravljačka riječ

          STORE  R0, (DMASTART)           ; pokreni DMA-prijenos

VAN       STORE  R0, (CTIEND)             ; dojavi kraj posluživanja CT-u

VAN_2     POP    R0                       ; obnavljanje spremljenih registara
          MOVE   R0, SR
          POP    R0
          RETI                            ; izlazak iz običnog prekida
```

3. (7 bodova) Za **ARM** napišite program koji obrađuje svaki 16-bitni podatak u formatu 2'k koji se nalazi u bloku memorije na adresi 1000_{16}, a 32-bitne rezultate obrade sprema u memorijski blok na adresi 2000_{16}. Početni blok sadrži 80_{16} podataka. Obradu pojedinog podatka treba raditi potprogramom POTP.

Napišite potprogram POTP koji prima 32-bitni 2'k parametar pomoću stoga i vraća 32-bitni rezultat registrom R0. POTP izračunava formulu:

rezultat = (parametar+7)^3 * konstanta + 128_{16}

Kubiranje se mora izvesti pomoću potprograma KUB. "Konstanta" je 8-bitni 2'k broj spremljen na adresi 7008_{16} (na tu adresu stavite broj -36_{16}). Pretpostavite da ni jedan izračun neće prekoračiti 32-bitni opseg 2'k brojeva.

Potprogram KUB prima parametar (32-bitni 2'k) pomoću lokacije iza naredbe BL, a 32-bitni rezultat vraća pomoću registra R0. KUB treba samo kubirati primljeni parametar.

```
            ORG    0
GLAVNI      MOV    SP, #10<12                ; inicijalizacija pokazivača stoga

            MOV    R1, #10<8                 ; adrese izvorišnog i ...
            MOV    R2, #20<8                 ; ...odredišnog bloka
            MOV    R3, #80                   ; duljina bloka - brojač za petlju
LOOP        LDRSH  R0, [R1], #2              ; učitaj 16-bitni 2'k (s proširenjem)
            STMFD  SP!, {R0}                 ; stavi parametar na stog
            BL     POTP                      ; pozovi potprogram POTP
            ADD    SP, SP, #4                ; ukloni parametar sa stoga
            STR    R0, [R2], #4              ; pohrani 32-bitni rezultat
            SUBS   R3, R3, #1                ; brojač za petlju i petlja
            BNE    LOOP
            SWI 123456                       ; zaustavljanje programa

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

POTP        STMFD  SP!, {R1, LR}             ; pohrana konteksta sa LR-om
            ; učitaj parametar sa stoga
            LDR R0,[SP,#8]
            ; gornja naredba može se zamijeniti sa:
            ; ADD  R1, SP, #8
            ; LDR  R0, [R1]       ; umjesto ove naredbe može i LDMFD R1,{R0}

            ADD    R0, R0, #7                ; uvećanje za 7 (1. dio izračuna)
            STR    R0, PARAM                 ; postavljanje parametra iza BL
            BL     KUB                       ; poziv potprograma (2. dio izračuna)
PARAM       DW     0                         ; mjesto za parametar
            LDR    R1, ADRESA                ; dohvat adrese konstante
            LDRSB  R1, [R1]                  ; dohvat same konstante + proširenje
            MUL    R0, R0, R1                ; množenje (3. dio izračuna)
            ; 128 = 0001 0010 1000 = 1 0010 10 < 2 = 4A<2 ili LDR iz memorije
            ADD    R0, R0, #4A<2             ; zbrajanje (4. dio izračuna) +
                                             ; broj 128 treba ispravno napisati
            LDMFD  SP!, {R1, LR}             ; obnova konteksta
            MOV    PC, LR                    ; povratak
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

KUB         STMFD  SP!, {R1}                ; pohrana konteksta
            LDR    R1, [LR], #4             ; dohvat parametra i povećanje LR-a
            MUL    R0, R1, R1               ; kubiranje
            MUL    R0, R0, R1
            LDMFD  SP!, {R1}                ; obnova konteksta
            MOV    PC, LR                   ; povratak

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

ADRESA      DW     7008                 ; adresa konstante

            ORG    7008                 ; postavljanje konstante na zadanu adresu
            DB     -36                  ; zadana vrijednost konstante
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

4. (10 bodova) Računalni sustav za grijanje bazena 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. Adrese sklopova odaberite sami.

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

- bitovi 0-5 – ulazni: temperatura vode u bazenu u rasponu od 0 do 63 stupnja celzija
- bit 6 – ulazni bit: temperatura je valjana
- bit 7 – izlazni bit: temperatura je pročitana

Na vrata B sklopa GPIO spojeni su prekidač i dva grijača (bitovi 3-7 se ne koriste):

- bit 0 – ulazni bit: prekidač za uključivanje (1) / isključivanje (0) sustava grijanja
- bit 1 – izlazni bit: uključivanje (1) / isključivanje (0) malog grijača
- bit 2 – izlazni bit: uključivanje (1) / isključivanje (0) velikog grijača

Napišite program koji beskonačno kontrolira temperaturu u bazenu na sljedeći način:

(1) Svakih 10 sekundi provjeri se je li sustav isključen. Ako je sustav isključen, grijače treba isključiti. Također potprogramu SUSTAV treba poslati parametar 0. Potprogram SUSTAV ne treba pisati, a parametar mu se šalje registrom R2.

(2) Ako je sustav uključen treba potprogramu SUSTAV poslati parametar 1. Zatim treba očitati trenutačnu temperaturu vode T. Ako je T>=24, treba isključiti grijače. Ako je T <= 20, treba držati uključena oba grijača. Ako je 21 <= T <= 23 stupnja, treba držati uključen samo mali grijač.

Dok ne protekne prvih 10 sekundi, grijači trebaju biti isključeni. Glavni program za vrijeme regulacije temperature vrti beskonačnu praznu petlju.

```
            ORG    0                    ; početak izvođenja
            B      GLAVNI               ; skok u glavni

            ORG    18                   ; adresa prekidnog
            B PREKIDNI                  ; skok u prekidni, a može i odmah prekidni

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

GLAVNI      MOV    SP, #10<12                 ; inicijalizacija stoga

            ; inicijaliziraj smjerove sklopa GPIO
            LDR    R0, GPIO                   ; dohvat adrese GPIO-a
            MOV    %B 10000000, R1            ; smjer porta A
            STR    R1, [R0, #8]
            MOV    %B 00000001, R1            ; smjer porta B
            STR    R1, [R0, #0C]

            MOV    R1, %B 000                 ; isključi grijače - neobavezno jer...
            STR    R1, [R0, #4]               ; 0 je po defaultu u registru podataka

            ; inicijaliziraj sklop RTC
            LDR    R0, RTC                    ; dohvat adrese RTC-a
            LDR    R1, KONST                  ; konstanta brojenja
            STR    R1, [R0, #4]
            MOV    R1, #0
            STR    R1, [R0, #0C]              ; neobavezno: brisanje brojila
            MOV    R1, #1                     ; kontrolna riječ: dozvoli prekide
            STR    R1, [R0, #10]

            ; dozvoli prekide IRQ
            MRS    R0, CPSR
            BIC    R0, R0, #80
            MSR    CPSR_c, R0

LOOP        B      LOOP                       ; beskonačna petlja

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


GPIO        DW     0FFFF0000            ; adrese GPIO-a i RTC-a
RTC         DW     0FFFF1000
KONST       DW     %D 10000             ; konstanta za RTC: 1kHz --> 10 sek
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

```
PREKIDNI   STMFD  SP!, {R0, R1, R2, LR}    ; spremanje konteksta i LR-a

           ; reinicijaliziraj RTC
           LDR    R0, RTC                  ; dohvat adrese RTC-a
           MOV    R1, #0                   ; brisanje brojila
           STR    R1, [R0, #0C]
           STR    R1, [R0, #8]             ; dojava prihvata prekida

REGULACIJA ; početak regulacije temperature
           LDR    R0, GPIO

           LDR    R1, [R0, #4]             ; ispitivanje uključenosti
           ANDS   R1, R1, #%B 00000001
           BNE    UKLJUCEN

ISKLJUCEN  ; sustav grijanja je isključen

           MOV    R2, #0                   ; slanje 0 potprogramu SUSTAV
           BL     SUSTAV

           MOV    R1, #%B 00000000         ; isključivanje oba grijača

           B      GRIJACI_I_VAN            ; slanje na grijače i izlazak

UKLJUCEN   ; sustav grijanja je uključen => treba ispitati temperaturu

           MOV    R2, #1                   ; slanje 1 potprogramu SUSTAV
           BL     SUSTAV

           ; čitaj temperaturu
CEKAJ      LDR    R1, [R0, #0]             ; čekanje spremnosti temperature
           ANDS   R1, R1, #%B 01000000
           BEQ    CEKAJ

           ; očitaj temperaturu i dojavi to impulsom na bitu 7
           LDR    R1, [R0, #0]
           ORR    R1, R1, #%B 10000000     ; postavi bit 7 u jedan
           STR    R1, [R0, #0]
           AND    R1, R1, #%B 01111111     ; postavi bit 7 u nulu
           STR    R1, [R0, #0]

           ; ispitivanje tri temperaturna područja
           AND    R1, R1, #3F              ; brisanje sinkronizacijskih bitova
           CMP    R1, #%D 24               ; test najviše granice
VRUCE      MOVHS  R1, #%B 00000000         ; isključi oba grijača
           BHS    GRIJACI_I_VAN            ; postavi grijače i izađi

           CMP    R1, #%D 20               ; test najniže granice
HLADNO     MOVLS  R1, #%B 00000110         ; uključi oba grijača
           BLS    GRIJACI_I_VAN            ; postavi grijače i izađi

SREDNJE    ; inače je srednja temperatura (21 do 23)
           MOV    R1, #%B 00000010         ; uključi samo mali grijač

GRIJACI_I_VAN    ; upravljanje grijačima i izlazak i prekidnog potprograma
           STR    R1, [R0, #4]

           LDMFD  SP!, {R0, R1, R2, LR}          ; obnovi spremljeni kontekst
           SUBS   PC, LR, #4                     ; izlazak iz prekida
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```