3.6.3. Prijenos sklopom DMA i sklop CT

Riješen: DA Težina: ★★

Napisati program koji pomoću sklopa DMA treba prenijeti 3000_{10} 32-bitnih podataka počevši od memorijske lokacije 1000_{16} na bezuvjetnu vanjsku jedinicu BVJ. Sklop DMA radi zaustavljanjem procesora. Tijekom prijenosa, sklop CT broji koliko ciklusa signala CLOCK je prijenos trajao (signal CLOCK spojen je na ulaz CT-a). Nakon prijenosa, broj ciklusa je potrebno zapisati na adresu 987654_{16}. Pretpostavite da će prijenos završiti prije nego sklop CT izbroji maksimalan broj ciklusa. Adrese vanjskih jedinica odaberite sami.

Prijedlog rješenja:

U zadatku je potrebno CT-om brojiti cikluse tijekom DMA-prijenosa. Kako sklop DMA radi zaustavljanjem procesora, nisu potrebne posebne provjere kraja prijenosa – nakon završetka prijenosa potrebno je jednostavno zaustaviti rad sklopa CT, čime će se zaustaviti i brojenje impulsa. Iako je tijekom DMA-prijenosa procesor zaustavljen, sklop CT nesmetano obavlja svoju zadaću jer mu sabirnica nije potrebna i radi neovisno o procesoru. Sklop CT radi tako da broji prema dolje – od konstante LR prema ništici. Zato treba zadati konstantu 0 koja će omogućiti najdulje brojenje od 10000_{16} impulsa. Nakon zaustavljanja brojenja, potrebno je pročitati trenutnu vrijednost brojača i oduzeti je od najvećeg broja impulsa koje CT može prebrojiti, tj. od broja 10000_{16}.

Komentar rješenja:

U ovom zadatku CT se ne koristi kao što je uobičajeno: nema određenog ciklusa brojenja čiji bi se kraj dojavio prekidom ili prepoznao uvjetnim ispitivanjem. Koristi se samo brojilo CT-a kako bi se izračunalo koliko ciklusa je odbrojeno. CT-u se ne briše spremnost niti mu se dojavljuje kraj posluživanja, jer do postavljanja spremnosti neće ni doći (uz zadanu pretpostavku da će prijenos završiti prije nego sklop CT izbroji maksimalan broj ciklusa).

Mjerenje je izvedeno tako da se pokrene brojenje na CT-u i neposredno nakon toga se pokrene DMA-prijenos. Naredba iza pokretanja DMA-prijenosa će se izvoditi nakon završenog DMA-prijenosa i tu se zaustavlja rad CT-a. Nije bilo nužno zaustaviti CT, već se moglo odmah pročitati stanje brojila. Mjerenje nije posve precizno, jer određeni broj ciklusa će proteći od pokretanja CT-a do pokretanja DMA-prijenosa, a također i od zaustavljanja DMA-prijenosa do zaustavljanja CT-a.

Oduzimanje se moglo izvesti bez prethodne naredbe **`MOVE 10000,R1`**, tj. izravno naredbom **`SUB R1,10000,R0`**. No, kako u naredbi **`SUB`** neposredno zadani broj može biti samo drugi operand, rezultat bi bio negativan, pa bi ga još trebalo pretvoriti u pozitivan broj.

Rješenje:

```
DMA_SRC     EQU     0FFFF0000          ; adrese vanjskih jedinica
DMA_DEST    EQU     0FFFF0004
DMA_SIZE    EQU     0FFFF0008
DMA_CTRL    EQU     0FFFF000C
DMA_START   EQU     0FFFF0010
DMA_ACK     EQU     0FFFF0014

BVJ         EQU     0FFFF1000

CTCR        EQU     0FFFF2000
CTLR        EQU     0FFFF2004


            ORG     0
GLAVNI      MOVE    1000, R0           ; adresa izvorišnog bloka
            STORE   R0, (DMA_SRC)
            MOVE    BVJ, R0            ; adresa odredišnog bloka
            STORE   R0, (DMA_DEST)
            MOVE    %D 3000, R0        ; veličina bloka
            STORE   R0, (DMA_SIZE)
            MOVE    %B 1000, R0        ; memorija -> VJ, zaustavljanje procesora, ...
            STORE   R0, (DMA_CTRL)     ; ... bez prekida

            MOVE    0, R0              ; vremensku konstantu 0 (10000 u 16-bita)
            STORE   R0, (CTLR)         ; stavljamo u LR
            MOVE    %B 01, R0          ; upravljačka riječ: nema prekida, a ...
            STORE   R0, (CTCR)         ; ... brojilo broji

            STORE   R0, (DMA_START)    ; pokretanje prijenosa

            MOVE    %B 00, R0          ; prijenos je gotov
            STORE   R0, (CTCR)         ; zaustavljanje brojenja CT-a
            STORE   R0, (DMA_ACK)      ; brisanje spremnosti DMA

            LOAD    R0, (CTLR)         ; čitanje trenutne vrijednosti brojača
            MOVE    10000, R1          ; početna vrijednost brojača
            SUB     R1, R0, R0         ; rezultat = početna vrijednost brojača -
                                       ;            trenutačna vrijednost

            LOAD    R1, (ADRESA)       ; dohvat adrese rezultata
            STORE   R0, (R1)           ; spremanje rezultata (tj. broja ciklusa)
            HALT


ADRESA      DW      987654             ; adresa rezultata

            ORG     987654
            DW      0                  ; mjesto za upis rezultata
```

3.6.17. Alarm za bicikl – uvjetna vanjska jedinica, sklopovi PIO i CT (2MI11)

Riješen: DA Težina: ★★★

Alarmni uređaj za bicikl se sastoji od procesora FRISC, uvjetne vanjske jedinice te sklopova PIO i CT. Uvjetna vanjska jedinica je senzor vibracija koji detektira pokušaj krađe bicikla. Senzor vibracija postaje spreman:

- ako nije bilo vibracija pa su počele; vrijednost podatka pročitanog sa senzora bit će 1
- ako je bilo vibracija pa su prestale; vrijednost podatka pročitanog sa senzora bit će 0

Na PIO (koji radi u načinu postavljanja bitova) spojen je zvučnik koji može raditi na sljedeće načine:

- tiši alarm: ako se postave jedinica na nižih 4 bita sklopa PIO
- glasniji alarm: ako se postave jedinice na svih 8 bitova sklopa PIO
- alarm isključen: ako se postave ništice na svih 8 bitova sklopa PIO

Na sklop CT spojen je signal frekvencije 10 kHz. CT radi u prekidnom načinu i spojen je na NMI.

Napišite program koji upravlja alarmom na sljedeći način:

Glavni program kontinuirano ispituje stanje senzora. Kada počnu vibracije, treba uključiti tiši alarm. Tada se pokreće i sklop CT koji odbrojava 3 sekunde od uključivanja tišeg alarma. Ako nakon 3 sekunde vibracije nisu prestale, treba uključiti glasniji alarm. Ako vibracije prestanu (u bilo kojem trenutku), treba isključiti alarm.

Na početku rada, alarm treba biti isključen. Adrese vanjskih jedinica odabrati po volji.

Prijedlog rješenja:

Na početku su i alarm (zvučnik spojen na PIO) i sklop CT isključeni. U glavnom programu se inicijalizira PIO da radi u načinu postavljanja bitova i isključuje se zvučnik, a nakon toga se u (beskonačnoj) glavnoj petlji prvo čeka spremnost senzora (uvjetna vanjska jedinica), a zatim se ispituju senzori te se uključuje tiši alarm kad započnu vibracije, ili se isključuje alarm ako vibracije prestanu. Važno je uočiti kako radi senzor vibracija. Suprotno onome što bi možda bilo intuitivno, spremnost ne znači da vibracije traju, a nespremnost ne znači da vibracija nema. Umjesto toga, spremnost senzora označava da su vibracije ili počele ili završile. Kad senzor postane spreman, pročita se iz njega podatak na temelju kojega se može ustanoviti jesu li vibracije započele ili su završile. Ova dva slučaja obrađuju se na sljedeći način.

Ako su vibracije započele, uključuje se tiši alarm, kojim se upravlja preko sklopa PIO, slanjem odgovarajućeg podatka. Istovremeno s uključivanjem tišeg alarma, inicijalizira se i sklop CT, koji započne mjeriti razdoblje od 3 sekunde. Nakon 3 sekunde dolazi do prekida od CT-a i tada se u prekidnom potprogramu uključi glasniji alarm i isključi se CT (jer se razdoblje od 3 sekunde treba izmjeriti samo jednokratno). CT se isključuje slanjem ništice kao upravljačke riječi, što će zaustaviti brojenje impulsa i zabraniti postavljanje prekida.

Ako su vibracije završile, onda treba isključiti alarm, bez obzira je li trenutno aktivan tiši ili glasniji alarm. Zato se na sklop PIO šalje ništica da se alarm isključi. Ako su vibracije prestale dok je aktivan tiši alarm, onda je sigurno aktivan CT koji mjeri razdoblje od 3 sekunde, pa ga treba isključiti. Ako su pak vibracije prestale dok je aktivan glasniji alarm, onda CT nije aktivan. U ovom rješenju glavni program će i u ovom slučaju poslati ništicu na CT, tj. isključit će CT koji već jeste isključen (što je jednostavnije nego pamtiti je li trenutno aktivan tiši ili glasniji alarm).

Budući da procesor radi na frekvenciji 10 kHz, a potrebno je mjeriti 3 sekunde, vremenska konstanta za sklop CT je: $10 * 10^3$ Hz * 3 s = 30000.

Rješenje:

```
UVJ_DATA    EQU     0FFFF0000          ; adrese vanjskih jedinica
UVJ_TEST    EQU     0FFFF0004

PIO_C       EQU     0FFFF1000
PIO_D       EQU     0FFFF1004
PIO_IACK    EQU     0FFFF1008
PIO_IEND    EQU     0FFFF100C

CT_LR       EQU     0FFFF2000
CT_CR       EQU     0FFFF2004
CT_IACK     EQU     0FFFF2008
CT_IEND     EQU     0FFFF200C

            ORG     0
POCETAK     MOVE    10000, SP          ; stog
            JP      GLAVNI             ; skok na glavni


            ORG     0C                 ; adresa nemaskirajućeg prekida NMI (za CT)
            ; prekidni potprogram – prošle su 3 sekunde
PP          PUSH    R0                 ; spremanje konteksta
            MOVE    SR, R0
            PUSH    R0

            STORE   R0, (CT_IACK)      ; prihvaćanje prekida od CT-a
            MOVE    %B 000, R0         ; isključivanje brojila
            STORE   R0, (CT_CR)
            STORE   R0, (CT_IEND)      ; kraj posluživanja CT-a

GLASNO      MOVE    %B 11111111, R0    ; uključivanje glasnijeg alarma: 11111111
            STORE   R0, (PIO_D)

            POP     R0                 ; obnova konteksta
            MOVE    SR, R0
            POP     R0
            RETN                       ; povratak iz potprograma



GLAVNI      MOVE    %B 0010, R0        ; PIO: postavljanje bitova, bez prekida
            STORE   R0, (PIO_C)
            MOVE    0, R0              ; na početku je alarm isključen
            STORE   R0, (PIO_D)        ; tj. zvučnik ne radi

PETLJA      LOAD    R0, (UVJ_TEST)     ; čekanje spremnosti senzora
            OR      R0, R0, R0
            JR_Z    PETLJA

ISPITAJ     LOAD    R0, (UVJ_DATA)     ; čitanje stanja senzora iz UVJ
            STORE   R0, (UVJ_TEST)     ; brisanje spremnosti
            CMP     R0, 0              ; ako je stanje 0 (vibracije su prestale):
            JR_EQ   ISKLJUCI           ; ... idi na isključivanje

            ; u protivnom stanje je 1 (vibracije su počele): uključi tiši alarm
TIHO        MOVE    %B 1111, R0        ; tiši alarm: slanje 1111 na PIO (zvučnik)
            STORE   R0, (PIO_D)

            ; pokreni mjerenje 3 sekunde pomoću CT-a
            MOVE    %D 30000, R0       ; konstanta, 3 sekunde * 10 kHz
            STORE   R0, (CT_LR)
            MOVE    %B 111, R0         ; brojilo počinje brojati, postavlja prekid NMI
            STORE   R0, (CT_CR)
            JR      PETLJA             ; povratak na ispitivanje spremnosti

ISKLJUCI    MOVE    0, R0              ; isključivanje brojila pa alarma
            STORE   R0, (CT_CR)        ; zaustavljanje brojila u CT-u
            STORE   R0, (PIO_D)        ; isključivanje alarma (zvučnika)
            JR      PETLJA
```

Komentar rješenja:
Prilikom rješavanja važno je shvatiti da sklop CT služi isključivo za mjerenje vremena od uključivanja tišeg do uključivanja glasnijeg alarma (3 sekunde). Kada je uključen glasniji alarm, sklop CT treba biti isključen. Također, nakon svakog isključivanja alarma, sklop CT treba isključiti, i pri sljedećem uključivanju alarma početi brojiti ispočetka – nema „nastavka“ brojenja.

Iako se na prvi pogled može činiti da nije potrebno provjeravati koja je vrijednost na senzoru (već samo provjeriti spremnost senzora), jer se vrijednost uvijek izmjenjuje (0 -> 1 -> 0), ipak je potrebno učitati i provjeriti vrijednost, jer ne znamo u kojem će stanju senzor biti prilikom samog uključenja.

Razmotrimo i rubni slučaj kada nakon početnih vibracija i uključenja tihog alarma dođe do prestanka vibracija u isto vrijeme kad ističu 3 sekunde i kad CT izaziva prekid.

Ako do prekida dođe neznatno prije prestanka vibracija, tada će se u prekidnom potprogramu uključiti glasniji alarm, ali će već povratkom u glavni program biti prepoznata spremnost senzora i nakon što se ispitivanjem ustanovi da su vibracije prestale, isključit će se alarm. Tako će glasni alarm trajati zanemarivo kratko – dok se izvede svega desetak naredaba (što je možda i prekratko da ga se može čuti). Ovo ponašanje može se ocijeniti ispravnim.

Ako senzor dojavi kraj vibracija neznatno prije isteka 3 sekunde, može se dogoditi da glavni program izvodi naredbe na labelama **PETLJA**, **ISPITAJ** i **ISKLJUCI**, a da još nije stigao do naredbe **STORE R0,(CT_CR)** iza labele **ISKLJUCI**. Ako u tom razdoblju dođe do isteka 3 sekunde i postavljanja prekida NMI, izvest će se prekidni potprogram u kojemu će biti uključen glasniji alarm, a povratkom u glavni program će se petlja izvesti do kraja, pri čemu će se zaustaviti već zaustavljeni CT, te će se isključiti zvučnik i ponašanje će opet biti ispravno. Ako se uspije izvesti naredba **STORE R0,(CT_CR)** iza labele **ISKLJUCI**, tada će CT biti zaustavljen i neće moći izazvati prekid te će se nakon toga isključiti zvučnik, što je ponovno ispravno ponašanje.

Što bi se moglo dogoditi kad bi se zamijenio redoslijed dviju naredaba **STORE** iza labele **ISKLJUCI**?

3.6.14. Sklop DMA, disk i sklop CT (2MI09)

Riješen: DA Težina: ★★

Na procesor FRISC spojeni su sklop DMA (adresa FFFF1000), disk (adresa FFFF2000) i sklop CT (spojen na NMI, adresa FFFF3000).

Program treba slijedno pokretati prijenos sklopom DMA (zaustavljanje procesora) svake sekunde pomoću sklopa CT (na ulaz CNT spojen je signal frekvencije 25 kHz). U svakom prijenosu sklopom DMA prenosi se po jedan blok od 20_{10} 32-bitnih podataka iz diska u memoriju, počevši slijedno od memorijske lokacije 1000_{10} na dalje (prvi blok na 1000_{10}, drugi blok na 1080_{10}, itd.). Nakon prijenosa 100_{10} blokova treba zaustaviti procesor. Nakon inicijalizacije vanjskih jedinica, glavni program treba cijelo vrijeme izvoditi praznu petlju.

S diska se podatci čitaju bezuvjetno, a disk u sebi ima međuspremnik za podatke kapaciteta 20_{10} riječi te zauzima 20_{10} 32-bitnih lokacija počevši od adrese FFFF2000.

Prijedlog rješenja:
Glavni program služi samo za inicijalizaciju sklopa CT, a nakon toga izvodi praznu petlju u kojoj se ispituje varijabla **GOTOVO**, čije početno stanje 0 služi kao oznaka da ima još blokova koje sklop DMA mora prenijeti. Kad se ustanovi da je varijabla **GOTOVO** poprimila vrijednost 1, izlazi se iz petlje i zaustavlja se procesor.

Svi se DMA-prijenosi inicijaliziraju, pokreću i odvijaju u prekidnom potprogramu od CT-a koji se periodički izvodi svake sekunde. Za generiranje prekida svake sekunde, uz ulazni signal frekvencije 25 kHz, vremenska konstanta sklopa CT je 25 * 10^3 Hz * 1 s = 25 000. DMA-prijenos odvija se zaustavljanjem procesora pa nije potrebno provjeravati završetak prijenosa. Budući da se prenosi samo 20 podataka, sigurno za vrijeme DMA-prijenosa neće doći do novog prekida od CT-a.

Iako se na prvi pogled može učiniti da sklop DMA s diskom treba komunicirati kao da se radi o vanjskoj jedinici, zadano je da se podatcima na disku pristupa preko međuspremnika, koji zauzima niz od 80 adresa (tj. bajtova). Zato će sklop DMA biti inicijaliziran za prijenos iz memorije u memoriju, zaustavljanjem procesora i bez postavljanja prekida. Početna adresa za prijenos podataka bit će adresa diska (FFFF2000), a uvijek će se prenositi 20 32-bitnih podataka. Adresu početka odredišnog memorijskog bloka (**BLOK**) treba mijenjati prilikom svakog prijenosa, kako stari podatci ne bi bili prebrisani novima. Nakon svakog prijenosa, na kraju prekidnog potprograma, smanjuje se brojač prenesenih blokova podataka (**BROJAC**). Ako je brojač još uvijek različit od ništice, povratak iz prekidnog potprograma se ostvaruje na uobičajen način (na labeli **IMA_JOS**). Kad brojač blokova poprimi vrijednost 0, zna se da je prenesen zadnji, tj. stoti blok. Tada se prije povratka iz prekidnog potprograma izvede još i odsječak na labeli **NEMA_VISE**. Ovaj odsječak će upisati vrijednost 1 u varijablu **GOTOVO** koju ispituje glavni program kako bi ustanovio jesu li svi blokovi prenesni. Također se zaustavlja rad CT-a kako ne bi postavljao daljnje prekide.

Rješenje:

```
DMASRC     EQU      0FFFF1000         ; adrese vanjskih jedinica
DMADEST    EQU      0FFFF1004
DMACNT     EQU      0FFFF1008
DMACTRL    EQU      0FFFF100C
DMASTART   EQU      0FFFF1010
DMASTAT    EQU      0FFFF1014

DISK       EQU      0FFFF2000

CTLR       EQU      0FFFF3000
CTCR       EQU      0FFFF3004
CTIACK     EQU      0FFFF3008
CTIEND     EQU      0FFFF300C

(nastavak na sljedećoj stranici)
```

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

          ORG     0C              ; adresa nemaskirajućeg prekidnog potprograma
          JP      NMI             ; skok u prekidni potprogram


GLAVNI    ; inicijalizacija CT-a
          MOVE    %D 25000, R0    ; vremenska konstanta
          STORE   R0, (CTLR)      ; postavi CTLR
          MOVE    %B 111, R0      ; brojilo broji, postavlja prekide, NMI
          STORE   R0, (CTCR)      ; pokreni CT
          ; CT je spojen na NMI – ne treba omogućavati prekide

PETLJA    LOAD    R0, (GOTOVO)    ; prazna petlja s
          CMP     R0, 0           ; ... ispitivanjem uvjeta za kraj
          JR_EQ   PETLJA

KRAJ      HALT                    ; zaustavljanje procesora


          ; prekidni potprogram za NMI
NMI       PUSH    R0              ; spremanje konteksta
          MOVE    SR, R0          ;
          PUSH    R0              ;
          STORE   R0, (CTIACK)    ; dojava prihvata prekida CT-u

          MOVE    DISK, R0        ; početna adresa diska
          STORE   R0, (DMASRC)    ; postavljanje početne adrese kao izvorišne

          LOAD    R0, (BLOK)      ; učitavanje sljedeće adrese bloka
          STORE   R0, (DMADEST)   ; postavljanje odredišne adrese

          ADD     R0, %D 80, R0   ; pomicanje adrese sljedećeg bloka
          STORE   R0, (BLOK)      ; ... i spremanje u memoriju

          MOVE    R0, %D 20       ; 20 – veličina bloka
          STORE   R0, (DMACNT)    ; brojač podataka za DMA prijenos

          MOVE    %B 0000, R0     ; MEM->MEM, zaustavljanje proc., bez prekida
          STORE   R0, (DMACTRL)   ; slanje kontrolne riječi

          STORE   R0, (DMASTART)  ; pokretanje DMA

          ; DMA-prijenos
          STORE   R0, (DMASTAT)   ; brisanje statusa od DMA (iako se ne koristi)

          LOAD    R0, (BROJAC)    ; smanjivanje brojača blokova
          SUB     R0, 1, R0
          STORE   R0, (BROJAC)    ; spremanje brojača
          JP_NZ   IMA_JOS         ; ako ima još blokova, nastaviti s radom

NEMA_VISE MOVE    1, R0           ; upis oznake da su svi blokovi preneseni
          STORE   R0, (GOTOVO)    ; ... u varijablu GOTOVO
          MOVE    0, R0           ; zaustaviti rad CT-a
          STORE   R0, (CTCR)

IMA_JOS   STORE   R0, (CTIEND)    ; dojava kraja posluživanja
          POP     R0              ; obnova konteksta
          MOVE    R0, SR
          POP     R0
          RETN                    ; povratak iz potprograma


GOTOVO    DW      0               ; oznaka kraja, početno stanje je 0

BLOK      DW      %D 1000         ; adresa trenutnog bloka
BROJAC    DW      %D 100          ; brojač blokova

          ORG     %D 1000         ; blok podataka
          SPACE   %D 8000         ; rezerviranje prostora za podatke
```

Komentar rješenja:
Prekidni potprogram za NMI počinje na adresi $0C_{16}$ i mogao je biti u cijelosti napisati od te lokacije na dalje, a glavni program je mogao biti napisan iza prekidnog potprograma. U ovom rješenju je prekidni potprogram napisan iza glavnog programa, a iz lokacije $0C_{16}$ se u prekidni potprogram skače naredbom **`JP NMI`**.

U prekidnom potprogramu se dio inicijalizacija sklopa DMA (izvorišna adresa i kontrolna riječ), ponavljaju svaki puta kad se obrađuje prekid iako bi ih bilo dovoljno izvesti samo jednom. Ovakvo rješenje je napravljeno zbog veće jednostavnosti.

Prije prvog prijenosa, varijabla **`BLOK`** pokazuje na adresu 1000_{16}, no prilikom svakog DMA prijenosa, potrebno je varijablu **`BLOK`** uvećati za 80_{10} (jer je u bloku 20_{10} podataka širine 32-bita). To se radi u prekidnom potprogramu.