Ј. ЂОРЂЕВИЋ, З. РАДИВОЈЕВИЋ, М. ПУНТ,
Б. НИКОЛИЋ, Д. МИЛИЋЕВ, Ј. ПРОТИЋ,
А. МИЛЕНКОВИЋ

АРХИТЕКТУРА И ОРГАНИЗАЦИЈА РАЧУНАРА

ПРЕКИДИ, МАГИСТРАЛА И УЛАЗ/ИЗЛАЗ

ЗБИРКА РЕШЕНИХ ЗАДАТАКА

Београд 2013.

САДРЖАЈ

1 УЛАЗ/ИЗЛАЗ

1.1 ЗАДАТАК

Дат је рачунарски систем који чине процесор, меморија, контролер периферије **KPER0** са периферијом **PER0** и контролер периферије **KPER1** са периферијом **PER1** повезани магистралом.

Процесор је са једноадресним форматом инструкција. Од програмски доступних регистара постоји 16 битни акумулатор ACC, указивач на врх стека SP, указивач на табелу (IV табела) са адресама прекидних рутина IVTP и програмска статусна реч PSW. Типови података који се користе су 16 битне целобројне величине са знаком и без знака. Приликом извршавања инструкције преноса у акумулатор и аритметичких, логичких и померачких инструкција постављају се индикатори N, Z, C и V програмске статусне речи PSW. При прекиду хардверски се на стеку чувају програмски бројач PC и програмска статусна реч PSW. Меморијски и улазно-излазни адресни простори су раздвојени.

Меморијски адресни простор је величине 2^{16} меморијских локација, при чему је ширина меморијских локација 16 битова.

Улазно-излазни адресни простор је величине 2^{16} регистара, при чему је ширина регистара 16 битова. Регистри контролера периферија **KPER0** и **KPER1** се налазе у улазно-излазном адресном простору.

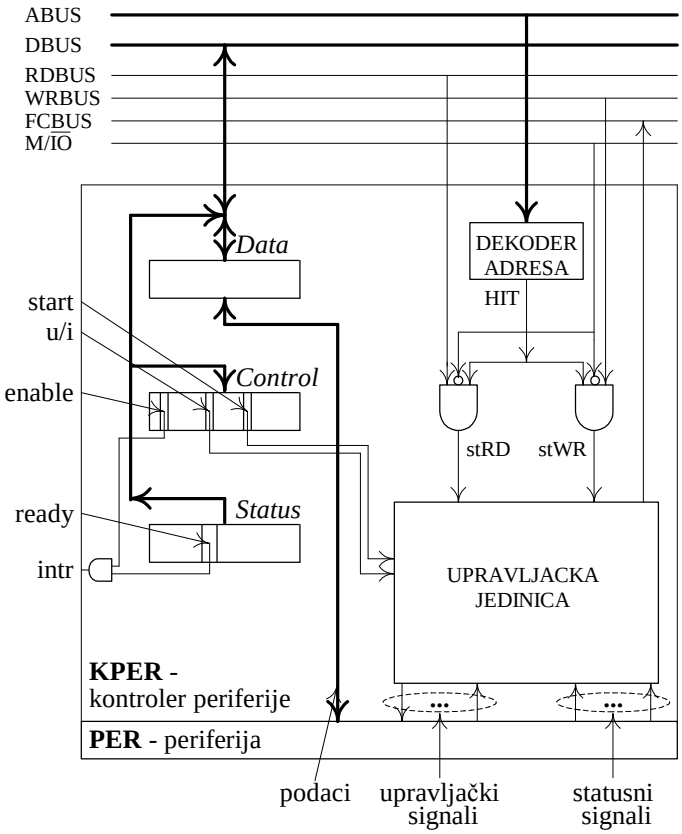
Контролери периферија **KPER0** и **KPER1** имају управљачке регистре (*Control*), статусне регистре (*Status*) и регистре података (*Data*) и то редом на адресама FF00h, FF01h и FF02h за контролер периферије **KPER0** и FF10h, FF11h и FF12h за контролер периферије **KPER1**. У регистрима контролера периферија **KPER0** и **KPER1** највиши бит је означен са 15, а најнижи са 0. У управљачким регистрима *Control* бит 15 је бит *start* којим се покреће извршавање операције (0—заустављен, 1—стартован), бит 1 је бит *u/i* којим се задаје смер преноса са периферијом (0—улаз, 1—излаз) и бит 0 је бит *enable* којим се дозвољава прекид (0—маскиран, 1—дозвољен). У статусним регистрима *Status* бит 0 је бит спремности *ready* (0—није спреман, 1—спреман).

Адресне линије и линије података магистрале рачунара су широке по 16 битова.

Написати програм којим се блок од 200h података учитава из **PER0**, смешта у меморију од локације F000h, обрађује процедуром `Obrada` и резултат шаље у **PER1**. Улаз из периферије **PER0** реализовати са контролером периферије **KPER0**, а излаз у периферију **PER1** реализовати са контролером периферије **KPER1** и то техником програмираног улаза и излаза са испитивањем спремности, респективно. Процедуру `Obrada` не треба реализовати, већ је само позвати на одговарајућем месту у програму помоћу инструкције `JSR Obrada`. Процедура `Obrada` не мења место и дужину блока података.

Решење:

Периферије **PER0** и **PER1** се повезују на магистралу рачунарског система са контролерима периферија **KPER0** и **KPER1** на начин приказан на слици 1.



Слика 1 Повезивање периферије **PER** са контролером периферије **KPER**

Адресе и структура регистара контролера **KPER0** и **KPER1** су дати на сликама 2 и 3, респективно.

KPER0 - Kontroler periferije **PER0**

Registar	Control	Stataus	Data
Adresa	FF00h	FF01h	FF02h

KPER1 - Kontroler periferije **PER1**

Registar	Control	Stataus	Data
Adresa	FF10h	FF11h	FF12h

Слика 2 Адресе регистара контролера

Слика 3 Структура регистара контролера

Тражени програм је приказан на слици 4.

```
;glavni program
;unos bloka podataka iz periferije PER0 u memoriju sa KPER0 sa ispitivanjem
;bita spremnosti ready
  ;inicijalizacija prenosa iz periferije PER0 u memoriju sa KPER0
    ;veličina bloka podataka
      LOAD   #200h     ;upis veličine bloka podataka za PER0 preko ACC u
      STORE  MemCnt0   ;mem. lok. na adr. MemCnt0
    ;početna adresa bloka podataka
      LOAD   #F000h    ;upis adr. bloka podataka za PER0 preko ACC u
      STORE  MemAdr0   ;mem. lok. na adr. MemAdr0
  ;startovanje kontrolera periferije KPER0
      LOAD   #8000h    ;upis režima rada i startovanja (start=1,
      OUT    FF00h     ;u/i=0,enable=0)preko ACC u reg. Control KPER0
  ;unos bloka podataka
    ;provera da li podatak može da se prenese iz registra Data KPER0
Loop0: IN     FF01h     ;upis sadrž. reg. Status KPER0 u ACC
       AND    #0001h    ;ispitivanje bita ready
       JZ     Loop0     ;povratak na Loop0 ukoliko je ready 0
    ;prenos podatka iz registra Data KPER0 u memorijsku lokaciju
      IN     FF02h     ;upis sadrž. reg. Data KPER0 preko ACC u mem. lok.
      STORE  (MemAdr0) ;na adresi datoj sadrž. mem. lok. na adr. MemAdr0
    ;inkrementiranje sadržaja memorijske lokacije MemAdr0
      LOAD   MemAdr0   ;upis sadrž. mem.lok. sa adr. MemAdr0 u ACC
      INC              ;inkrementiranje sadrž. ACC
      STORE  MemAdr0   ;upis sadrž. ACC u mem.lok. na adr. MemAdr0
    ;dekrementiranje sadržaja memorijske lokacije MemCnt0
      LOAD   MemCnt0   ;upis sadrž. mem.lok. sa adr. MemCnt0 u ACC
      DEC              ;dekrementiranje sadrž. ACC
      STORE  MemCnt0   ;upis sadrž. ACC u mem.lok. na adr. MemCnt0
    ;provera da li je unet poslednji podatak
      JNZ    Loop0     ;povratak na Loop0 ukoliko nije poslednji podatak
  ;zaustavljanje kontrolera periferije KPER0
      LOAD   #0        ;upis režima zaustavljanja (start=0, u/i=0,
      OUT    FF00h     ;enable=0)preko ACC u reg. Control KPER0
;obrada
       JSR    Obrada    ;obrada bloka od 200h podataka u memorijskim
                        ;lokacijama od adrese F000h do adrese F1FFh

;slanje bloka podataka iz memorije u periferiju PER1 sa KPER1 sa
;ispitivanjem bita spremnosti ready
  ;inicijalizacija prenosa iz memorije u periferiju PER1 sa KPER1
    ;veličina bloka podataka
      LOAD   #200h     ;upis veličine bloka podataka za PER1 preko ACC u
      STORE  MemCnt1   ;mem. lok. na adr. MemCnt1
    ;početna adresa bloka podataka
      LOAD   #F000h    ;upis adr. bloka podataka za PER1 preko ACC u
      STORE  MemAdr1   ;mem. lok. na adr. MemAdr1
  ;startovanje kontrolera periferije KPER1
      LOAD   #8002h    ;upis režima rada i startovanja (start=1,
      OUT    FF10h     ;u/i=1,enable=0) preko ACC u reg. Control KPER1
  ;slanje bloka podataka
    ;provera da li podatak može da se prenese u registar Data KPER1
Loop1: IN     FF11h     ;upis sadrž. reg. Status KPER1 u ACC
       AND    #1        ;ispitivanje bita ready
       JZ     Loop1     ;povratak na Loop1 ukoliko je ready 0
    ;prenos podatka iz memorijske lokacije u registar Data KPER1
      LOAD   (MemAdr1) ;upis sadrž. mem. lok. sa adr. date sadrž. mem.
      OUT    FF12h     ;lok. sa adr. MemAdr1 preko ACC u reg. Data KPER1
```

```
    ;inkrementiranje sadržaja memorijske lokacije MemAdr1
      LOAD   MemAdr1  ;upis sadrž. mem.lok. sa adr. MemAdr1 u ACC
      INC             ;inkrementiranje sadrž. ACC
      STORE  MemAdr1  ;upis sadrž. ACC u mem.lok. na adr. MemAdr1
    ;dekrementiranje sadržaja memorijske lokacije MemCnt1
      LOAD   MemCnt1  ;upis sadrž. mem.lok. sa adr. MemCnt1 u ACC
      DEC             ;dekrementiranje sadrž. ACC
      STORE  MemCnt1  ;upis sadrž. ACC u mem.lok. na adr. MemCnt1
    ;provera da li je prenet poslednji podatak
      JNZ    Loop1    ;povratak na Loop1 ukoliko nije poslednji podatak
  ;zaustavljanje kontrolera periferije KPER1
      LOAD   #0       ;upis režima zaustavljanja (start=0, u/i=0,
      OUT    FF10h    ;enable=0)preko ACC u reg. Control KPER1
. . .
```

Слика 4 Програм

Приликом уноса блока података из периферије **PER0** у меморију са контролером периферије **KPER0** бит спремности *ready* статусног регистра *Status* **KPER0** се поставља на вредности 0 и 1 на начин приказан у даљем тексту. Унос блока података из периферије **PER0** у меморију започиње тако што се извршавањем одговарајућег програма у управљачки регистар *Control* **KPER0** упише садржај којим се контролер стартује (*start*=1) за рад у режиму улаза (*u/i*=0) и без генерисања прекида (*enable*=0). Том приликом контролер **KPER0** поставља на вредност 0 бит спремности *ready* статусног регистра *Status* **KPER0** и започиње читање податка из периферије **PER0** и пренос у регистар података *Data* **KPER0**. Контролер **KPER0**, приликом уписа података из периферије **PER0** у регистар *Data* **KPER0**, поставља бит *ready* на вредност 1. Када се, у оквиру извршавања одговарајућег програма којим се преноси садржај регистра *Data* **KPER0** у меморијску локацију, чита регистар *Data* **KPER0**, контролер **KPER0** поставља на вредност 0 бит *ready* и започиње читање следећег података из периферије **PER0** и пренос у регистар података *Data* **KPER0**.

Приликом слања блока података из меморије у периферију **PER1** бит спремности *ready* статусног регистра *Status* **KPER1** се поставља на вредности 1 и 0 на начин приказан у даљем тексту. Слање блока података из меморију у периферију **PER1** започиње тако што се извршавањем одговарајућег програма у управљачки регистар *Control* **KPER1** упише садржај којим се контролер **KPER1** стартује (*start*=1) за рад у режиму излаза (*u/i*=1) и без генерисања прекида (*enable*=0). Том приликом контролер **KPER1** поставља на вредност 1 бит спремности *ready* статусног регистра *Status* **KPER1**. Када се, у оквиру извршавања одговарајућег програма којим се преноси садржај меморијске локације у регистар *Data* **KPER1**, врши упис у регистар *Data* **KPER1**, контролер **KPER1** поставља на вредност 0 бит *ready* и започиње пренос података из регистра података *Data* **KPER1** у периферију **PER1**. Контролер **KPER1**, када заврши пренос података из регистра података *Data* **KPER1** у периферију **PER1**, поставља бит *ready* на вредност 1.

Програм се састоји из три дела. У првом делу се из периферије **PER0** техником програмираног улаза са испитивањем бита спремности *ready* статусног регистра *Status* **KPER0** уноси блок од 200h података и смешта у меморијске локације од адресе F000h до адресе F1FFh. У другом делу се позива процесура *Obrada* која врши одређену обраду над унетим подацима у блоку података у меморијским локацијама од адресе F000h до адресе F1FFh и резултат смешта у исте меморијске локације од адресе F000h до адресе F1FFh. У трећем делу се у периферију **PER1** техником програмираног излаза са испитивањем бита спремности *ready* статусног регистра *Status* **KPER1** шаље блок од 200h података прочитаних из меморијских локација од адресе F000h до адресе F1FFh.

У првом делу се уноси блок података са периферије **PER0** у меморију тако што се најпре се врши иницијализација преноса из периферије **PER0** и стартовање контролера периферије **KPER0**, затим сам унос блока података и на крају заустављање контролера периферије **KPER0**.

У оквиру иницијализације преноса се најпре у меморијску локацију чија је адреса симболички означена са MemCnt0 уписује вредност 200h која представља величину блока података који треба пренети, а затим се у меморијску локацију чија је адреса симболички означена са MemAdr0 уписује вредност F000h која представља почетну адресу дела меморије у који треба пренети блок података. У оквиру стартовања контролера периферије **KPER0** у управљачки регистар *Control* **KPER0**, који се налази на адреси FF00h у улазно/излазном адресном простору, се уписује вредност 8000h чиме се контролер стартује (*start*=1) за рад у режиму улаза (*u/i*=0) и без генерисања прекида (*enable*=0).

У оквиру уноса блока података се најпре у унутрашњој петљи са лабелом Loop0 проверава да ли податак може да се пренесе из регистра *Data* **KPER0** у меморијску локацију, а затим се у спољашњој петљи са лабелом Loop0 најпре реализује пренос податка из регистра *Data* **KPER0** у меморијску локацију, потом инкрементира садржај меморијске локације MemAdr0 и декрементира садржај меморијске локације MemCnt0 и на крају на основу провере да ли је пренет последњи податак или остаје у петљи и продужава са преносом података или излази из петље и завршава са преносом података. Провера да ли податак може да се пренесе из регистра *Data* **KPER0** у меморијску локацију се реализује тако што се у унутрашњој петљи најпре садржај статусног регистра *Status* **KPER0**, који се налази на адреси FF01h у улазно/излазном адресном простору, преноси у акумулатор ACC и затим реализује логичка И операција са непосредном величином 0001h која на биту који одговара биту *ready* статусног регистра *Status* **KPER0** има вредност 1 а на осталим битовима има вредност 0. Уколико бит *ready* има вредност 0, резултат И операције ће бити уписивање вредности 0000h у акумулатор ACC и постављање индикатора Z програмске статусне речи PSW на вредност 1, док у случају да бит *ready* има вредност 1, резултат И операције ће бити уписивање вредности 0001h у акумулатор ACC и постављање индикатора Z програмске статусне речи PSW на вредност 0. На крају унутрашње петље се реализује условни скок на лабелу Loop0 и остаје у унутрашњој петљи уколико индикатор Z има вредност 1 и излази из унутрашње петље уколико индикатор Z има вредност 0. Пренос податка из регистра *Data* **KPER0** у меморијску локацију се реализује тако што се најпре садржај регистра податка *Data* **KPER0**, који се налази на адреси FF02h у улазно/излазном адресном простору, преноси у акумулатор ACC и затим из акумулатора ACC уписује у меморијску локацију на адреси одређеној садржајем меморијске локације MemAdr0. Садржај меморијске локације MemAdr0 представља показивач на меморијску локацију у коју треба пренети следећи податак, па се зато њен садржај инкрементира после преноса сваког податка. Како је усвојено да је могуће инкрементирати само садржај акумулатора ACC, најпре се садржај меморијске локације MemAdr0 пребацује у акумулатор ACC и затим се инкрементирани садржај акумулатора ACC враћа у меморијску локацију MemAdr0. Садржај меморијске локације MemCnt0 представља преостали број података из блока података које треба пренети, па се зато њен садржај декрементира после преноса сваког податка. Како је усвојено да је могуће декрементирати само садржај акумулатора ACC, најпре се садржај меморијске локације MemCnt0 пребацује у акумулатор ACC и затим се декрементирани садржај акумулатора ACC враћа у меморијску локацију MemCnt0. Уколико је као резултат декрементирања садржај акумулатор ACC постао 0, индикатор Z програмске статусне речи PSW ће се поставити на вредност 1, док ће се у супротном случају поставити на вредност 0.

Инструкција STORE којом се декрементирани садржај акумулатора ACC враћа у меморијску локацију MemCnt0 не мења вредност индикатора Z постављену на основу резултата декрементирања. На крају спољашње петље се реализује условни скок на лабелу Loop0 и остаје у спољашњој петљи уколико индикатор Z има вредност 0 и излази из спољашње петље уколико индикатор Z има вредност 1.

У оквиру заустављања контролера периферије **KPER0** у управљачки регистар *Control* **KPER0** се уписује вредност 0000h, што има за последицу да се контролер **KPER0** зауставља (*start*=0). При томе уписивање вредности 0 у битове којима се задаје режим рада (*u/i*=0 и *enable*=0) нема никаквог значаја.

У другом делу се позива процесура *Obrada* која врши одређену обраду над унетим подацима у блоку података у меморијским локацијама од адресе F000h до адресе F1FFh и резултат смешта у исте меморијске локације од адресе F000h до адресе F1FFh.

У трећем делу се шаље блок података из меморије у периферију **PER1** и то на сличан начин као што се уноси блок података из периферије **PER0** у меморију. Разлика је у самом преносу јер се сада садржај меморијске локације са адресе одређенс садржајем меморијске локације MemAdr1 преноси у акумулатор ACC и затим из акумулатора ACC уписује у регистар података *Data* **KPER1**, који се налази на адреси FF12h у улазно/излазном адресном простору.

Дискусија:

1. Процесор је једноадресни, па се сви преноси података, који могу да се нађу у меморијским локацијама, регистрима контролера периферија или непосредно у самој инструкцији, реализују преко акумулатора ACC. Поред тога, меморијски и улазно-излазни адресни простори су раздвојени, па се регистрима у контролерима периферија **KPER0** и **KPER1** приступа искључиво инструкцијама IN и OUT. У складу са тим уписивање вредности 200h у меморијску локацију на адреси MemCnt0 се реализује инструкцијама LOAD #200h и STORE MemCnt0, а уписивање вредности #8000h у регистар *Control* **KPER0** на адреси FF00h инструкцијама LOAD #8000h и OUT FF00h.

2. Усвојено је да су инструкције INC и DEC безадресне и да се њима инкрементира и декрементира садржај регистра акумулатора ACC. Због тога, када се јави потреба да се садржај неке локације инкрементира или декрементира, садржај дате локације се најпре пребацује у акумулатор ACC, затим се садржај акумулатора инкрементира или декрементира и на крају добијена вредност враћа у локацију. У складу са тим инкрементирање садржаја меморијске локације на адреси MemAdr0 се реализује инструкцијама LOAD MemAdr0, INC и STORE MemAdr0. Међутим, инструкције INC и DEC могу да имају и једноадресни формат, па би се тада инкрементирање садржаја меморијских локација MemAdr0 и MemAdr1 и декрементирање садржаја меморијских локација MemCnt0 и MemCnt1 реализовало једном инструкцијом (INC MemAdr0, INC MemAdr1, DEC MemCnt0 и DEC MemCnt1).

3. Провера вредности бита *ready* статусног регистра *Status* контролера периферије мора да се реализује логичком инструкцијама AND или TST (логичко поређење), а не аритметичким инструкцијама SUB или CMP (аритметичко поређење). Инструкција логичког поређења TST, као и инструкција AND, реализује логичку И операцију над садржајем акумулатора ACC и садржајем операнда специфицираним адресним делом инструкције, али, за разлику од инструкције AND, резултат не уписује у акумулатор ACC већ само, као и инструкција AND, на основу резултата операције поставља индикаторе N, Z, C и V програмске статусне речи PSW. Слична је ситуација и са инструкцијом аритметичког поређења CMP која, као и инструкција SUB, реализује аритметичку операцију одузимања над садржајем акумулатора ACC и садржајем операнда специфицираним адресним делом инструкције, али, за разлику од инструкције

SUB, резултат не уписује у акумулатор ACC већ само, као и инструкција SUB, на основу резултата операције поставља индикаторе N, Z, C и V програмске статусне речи PSW.

4. Иницијализација и стартовање контролера периферије **KPER1** су реализовани на почетку трећег дела јер се на извршавање дела програма којим се реализује сам пренос података из меморије у периферију **PER1** сме прећи тек после извршавања програма којим се обавља комплетан пренос података из периферије **PER0** у меморију и реализује обрада. Програм се, међутим, може и тако написати да се одмах на почетку иницијализује и стартује не само контролер периферије **KPER0** већ и контролер периферије **KPER1**. Тада би део програма са слике4

```
;inicijalizacija prenosa iz memorije u periferiju PER1 sa KPER1
  ;veličina bloka podataka
    LOAD   #200h     ;upis veličine bloka podataka za PER1 preko ACC u
    STORE  MemCnt1   ;mem. lok. na adr. MemCnt1
  ;početna adresa bloka podataka
    LOAD   #F000h    ;upis adr. bloka podataka za PER1 preko ACC u
    STORE  MemAdr1   ;mem. lok. na adr. MemAdr1
;startovanje kontrolera periferije KPER1
    LOAD   #8002h    ;upis režima rada i startovanja (start=1,
    OUT    FF10h     ;u/i=1,enable=0) preko ACC u reg. Control KPER1
```

којим се иницијализује и стартује контролер периферије **KPER1** требало преместити и ставити између дела програма којим се иницијализује и стартује контролер периферије **KPER0** и дела програма којим се у петљи Loop0 реализује унос блока података из периферије **PER0** у меморију.

Међутим тиме се не добија ништа, јер се на извршавање дела програма којим се реализује сам пренос података из меморије у периферију **PER1** сме прећи тек после извршавања програма којим се обавља комплетан пренос података из периферије **PER0** у меморију и реализује обрада. Контролер периферије **KPER1** иницијализован и стартован за рад у режиму програмираног излаза са испитивањем спремности иако би био стартован одмах на почетку не би кренуо са самим слањем података из меморије у периферију **PER1** све док се не дође на део програма у коме се у петљи Loop1 реализује само слање блока података из меморије у периферију **PER1**.

1.2 ЗАДАТАК

Дат је рачунарски систем из задатка 1.1.

Написати програм којим се блок од 200h података учитава из **PER0**, смешта у меморију од локације F000h, обрађује процедуром `Obrada` и резултат шаље у **PER1**. Улаз из периферије **PER0** реализовати са контролером периферије **KPER0** техником програмираног улаза са испитивањем спремности, а излаз у периферију **PER1** реализовати са контролером периферије **KPER1** техником програмираног излаза са прекидом. Процедуру `Obrada` не треба реализовати, већ је само позвати на одговарајућем месту у програму помоћу инструкције `JSR Obrada`. Процедура `Obrada` не мења место и дужину блока података.

Решење:

Тражени програм је приказан на слици 5.

```
;glavni program
;unos bloka podataka iz periferije PER0 u memoriju sa KPER0 sa ispitivanjem
;bita spremnosti ready
  ;inicijalizacija prenosa iz periferije PER0 u memoriju sa KPER0
    ;veličina bloka podataka
      LOAD   #200h     ;upis veličine bloka podataka za PER0 preko ACC u
      STORE  MemCnt0   ;mem. lok. na adr. MemCnt0
    ;početna adresa bloka podataka
      LOAD   #F000h    ;upis adr. bloka podataka za PER0 preko ACC u
      STORE  MemAdr0   ;mem. lok. na adr. MemAdr0
  ;startovanje kontrolera periferije KPER0
      LOAD   #8000h    ;upis režima rada i startovanja (start=1,
      OUT    FF00h     ;u/i=0,enable=0)preko ACC u reg. Control KPER0
  ;unos bloka podataka
    ;provera da li podatak može da se prenese iz registra Data KPER0
Loop0: IN     FF01h     ;upis sadrž. reg. Status KPER0 u ACC
       AND    #1        ;ispitivanje bita ready
       JZ     Loop0     ;povratak na Loop0 ukoliko je ready 0
    ;prenos podatka iz registra Data KPER0 u memorijsku lokaciju
      IN     FF02h     ;upis sadrž. reg. Data KPER0 preko ACC u mem. lok.
      STORE  (MemAdr0) ;na adresi datoj sadrž. mem. lok. na adr. MemAdr0
    ;inkrementiranje sadržaja memorijske lokacije MemAdr0
      LOAD   MemAdr0   ;upis sadrž. mem.lok. sa adr. MemAdr0 u ACC
      INC              ;inkrementiranje sadrž. ACC
      STORE  MemAdr0   ;upis sadrž. ACC u mem.lok. na adr. MemAdr0
    ;dekrementiranje sadržaja memorijske lokacije MemCnt
      LOAD   MemCnt0   ;upis sadrž. mem.lok. sa adr. MemCnt0 u ACC
      DEC              ;dekrementiranje sadrž. ACC
      STORE  MemCnt0   ;upis sadrž. ACC u mem.lok. na adr. MemCnt0
    ;provera da li je unet poslednji podatak
      JNZ    Loop0     ;povratak na Loop0 ukoliko nije poslednji podatak
  ;zaustavljanje kontrolera periferije KPER0
      LOAD   #0        ;upis režima zaustavljanja (start=0, u/i=0,
      OUT    FF00h     ;enable=0)preko ACC u reg. Control KPER0
;obrada
       JSR    Obrada    ;obrada bloka od 200h podataka u memorijskim
                        ;lokacijama od adrese F000h do adrese F1FFh
;slanje bloka podataka iz memorije u periferiju PER1 sa KPER1 sa prekidom
  ;inicijalizacija prenosa iz memorije u periferiju PER1 sa KPER1
    ;veličina bloka podataka
      LOAD   #200h     ;upis veličine bloka podataka za PER1 preko ACC u
      STORE  MemCnt1   ;mem. lok. na adr. MemCnt1
```

```
   ;početna adresa bloka podataka
        LOAD   #F000h    ;upis adr. bloka podataka za PER1 preko ACC u
        STORE  MemAdr1   ;mem. lok. na adr. MemAdr1
   ;postavljanje "semafor"-a za PER1 na 1
        LOAD   #1h       ;upis vrednosti 1 preko ACC u
        STORE  MemSem1   ;mem. lok. na adr. MemSem1
   ;startovanje kontrolera periferije KPER1
        LOAD   #8003h    ;upis režima rada i startovanja (start=1,
        OUT    FF10h     ;u/i=1,enable=1)preko ACC u reg. Control KPER1
   ;čekanje na završetak slanja bloka podataka iz memorije u periferiju
   ;PER1 proverom vrednosti "semafor"-a za PER1
Wait1: LOAD   MemSem1   ;upis sadrž. mem.lok. sa adr. MemSem1 u ACC
        CMP    #0       ;ispitivanje da li je "semafor" postao 0
        JNZ    Wait1    ;povratak na Wait1 ukoliko "semafor" nije 0
. . .

;prekidna rutina periferije PER1
;slanje bloka podataka iz memorije u periferiju PER1 sa KPER1
PER1:  PUSHA            ;akumulator ACC na stek
   ;prenos podatka iz memorijske lokacije u registar Data KPER1
        LOAD   (MemAdr1) ;upis sadrž. mem. lok. sa adr. date sadrž. mem.
        OUT    FF12H    ;lok. sa adr. MemAdr1 preko ACC u reg. Data KPER1
   ;inkrementiranje sadržaja memorijske lokacije MemAdr1
        LOAD   MemAdr1   ;upis sadrž. mem.lok. sa adr. MemAdr1 u ACC
        INC              ;inkrementiranje sadrž. ACC
        STORE  MemAdr1   ;upis sadrž. ACC u mem.lok. na adr. MemAdr1
   ;dekrementiranje sadržaja memorijske lokacije MemCnt
        LOAD   MemCnt1   ;upis sadrž. mem.lok. sa adr. MemCnt1 u ACC
        DEC              ;dekrementiranje sadrž. ACC
        STORE  MemCnt1   ;upis sadrž. ACC u mem.lok. na adr. MemCnt1
   ;provera da li je prenet poslednji podatak
        JNZ    Back1     ;prelaz na Back1 ukoliko nije poslednji podatak
   ;zaustavljanje kontrolera periferije KPER1
        LOAD   #0        ;upis režima zaustavljanja (start=0, u/i=0,
        STORE FF10H      ;enable=0)preko ACC u reg. Control KPER1
   ;postavljanje "semafor"-a za PER1 na 0
                         ;upis vrednosti 0 preko ACC u
        STORE  MemSem1   ;mem. lok. na adr. MemSem1
   ;izlazak iz prekidne rutine
Back1: POPA              ;restauracija akumulatora ACC sa steka
        RTI              ;povratak iz prekidne rutine
. . .
```

Слика 5 Програм

Програм се састоји из три дела. У првом делу се из периферије **PER0** техником програмираног улаза са испитивањем бита спремности *ready* статусног регистра *Status* **KPER0** уноси блок од 200h података и смешта у меморијске локације од адресе F000h до адресе F1FFh. У другом делу се позива процесура *Obrada* која врши одређену обраду над унетим подацима у блоку података у меморијским локацијама од адресе F000h до адресе F1FFh и резултат смешта у исте меморијске локације од адресе F000h до адресе F1FFh. У трећем делу се у периферију **PER1** техником програмираног излаза са прекидом шаље блок од 200h података прочитаних из меморијских локација од адресе F000h до адресе F1FFh. Прва два дела програма су идентична као прва два дела програма у задатку 1.1. У трећем делу постоји разлика јер се за слање блока података из меморије у периферију **PER1** уместо технике програмираног излаза са испитивањем бита спремности *ready* статусног регистра *Status* **KPER1** користи техника програмираног излаза са прекидом.

Део програма којим се шаље блок података из меморије у периферију **PER1** има два дела и то један део који се извршава у главном програму и други део који се извршава у прекидној рутини периферије **PER1**. У оквиру дела програма који се извршава у главном програму врши се иницијализација преноса у периферију **PER1**, постављање "semafor"-а за **PER1** на 1, стартовање контролера периферије **KPER1** и чекање на завршетак слања блока података из меморије у периферију **PER1**. У оквиру дела програма који се извршава у прекидној рутини периферије **PER1** врши се пренос податка из меморијске локације у регистар *Data* **KPER1**, инкрементирање садржаја меморијске локације MemAdr1 и декрементирање садржаја меморијске локације MemCnt1 и, уколико је пренет последњи податак, заустављање контролера периферије **KPER1** и постављање "semafor"-а за **PER1** на 0.

У оквиру иницијализације преноса у периферију **PER1** се најпре у меморијску локацију чија је адреса симболички означена са MemCnt1 уписује вредност 200h која представља величину блока података који треба пренети, а затим се у меморијску локацију чија је адреса симболички означена са MemAdr1 уписује вредност F000h која представља почетну адресу дела меморије из кога треба пренети блок података. У оквиру постављања "semafor"-а за **PER1** на 1, у меморијску локацију MemSem1 се уписује вредност 1 која служи као индикација да је пренос података из меморије у периферију **PER1** у току и да се не сме продужити са извршавањем главног програма док пренос не буде завршен. У оквиру стартовања контролера периферије **KPER1** у управљачки регистар *Control* **KPER1**, који се налази на адреси FF10h у улазно/излазном адресном простору, се уписује вредност 8003h чиме се контролер стартује (*start*=1) за рад у режиму излаза (*u/i*=1) са генерисањем прекида (*enable*=1).

Од овог тренутка процесор и контролер периферије **KPER1** почињу да раде паралелно. Процесор чека завршетак преноса блока података из меморије у периферију **PER1** у петљи са лабелом Wait1. Контролер периферије **KPER1**, најпре, пошто је стартован (*start*=1) за рад у режиму излаза (*u/i*=1), поставља на вредност 1 бит спремности *ready* статусног регистра *Status* **KPER1**, а потом, пошто је стартован (*start*=1) за рад са генерисањем прекида (*enable*=1), генерише прекид и чека да се податак пренесе из меморијске локације у регистар *Data* **KPER1**.

На прекид генерисан од стране контролера периферије **KPER1**, процесор излази из петље са лабелом Wait1 и прелази на извршавање инструкција прекидне рутине периферије **PER1** на чијем почетку преноси податак из меморијске локације у регистар *Data* **KPER1**. Од овог тренутка процесор и контролер периферије **KPER1** настављају да раде паралелно. Процесор наставља са извршавањем инструкција прекидне рутине тако што инкрементира садржај меморијске локадине MemAdr1, декрементира садржај меморијске локације MemCnt1 и, уколико није пренет последњи податак, враћа се у петљу са лабелом Wait1 прекинутог главног програма. Контролер периферије **KPER1** поставља на вредност 0 бит *ready* статусног регистра *Status* **KPER1** и започиње пренос податка из регистра података *Data* **KPER1** у периферију **PER1** и по његовом завршетку, поставља бит *ready* статусног регистра *Status* **KPER1** на вредност 1 и поново генерише прекид.

Рад процесора и контролера приказан у претходном параграфу се наставља до преласка процесора на прекидну рутину периферије **PER1** ради преноса последњег податка. Када у прекидној рутини периферије **PER1** садржај меморијске локације MemCnt1 декрементирањем постане 0, процесор најпре изврши инструкције прекидне рутине којима се поставља "semafor" за **PER1** на 0 и зауставља контролер периферије **KPER1**, па се затим враћа се у петљу са лабелом Wait1 прекинутог главног програма.

Током наставка извршавања програма у петљи са лабелом Wait1 утврдиће се да "semafor" за **PER1** сада има вредност 0. Ова вредност служи као индикација да је

пренос података завршен, па се излази из петље са лабелом Wait1 и продужава извршавање главног програма.

Првом инструкцијом прекидне рутине PUSHA на стеку се чува садржај акумулатора ACC прекинутог главног програма, док се предзадњом инструкцијом прекидне рутине POPA садржај акумулатора ACC прекинутог главног програма рестаурира вредношћу са стека. Ово је неопходно учинити јер се у главном програму користи садржај акумулатора ACC, а у прекидној рутини се садржај акумулатора ACC мења. Садржаји регистара PC и PSW прекинутог главног програма се чувају на стеку хардверски у оквиру фазе опслуживање прекида инструкције главног програма из које се прелази у прекидну рутину, док се задњом инструкцијом прекидне рутине RTI садржаји регистара PSW и PC прекинутог главног програма рестаурирају вредностима са стека.

Дискусија:

1. Иницијализација и стартовање контролера периферије **KPER1** су као и у задатку 1.1 реализовани на почетку трећег дела јер се на извршавање дела програма којим се реализује сам пренос података из меморије у периферију **PER1** сме прећи тек после извршавања програма којим се обавља комплетан пренос података из периферије **PER0** у меморију и реализује обрада. У задатку 1.1 је показано да се програм може тако написати да се одмах на почетку иницијализује и стартује не само контролер **KPER0** већ и контролер периферије **KPER1**. Показано је да се тиме није добило ништа, јер се на извршавање дела програма којим се реализује сам пренос података из меморије у периферију **PER1** техником програмираног излаза са испитивањем спремности прелазило тек после извршавања програма којим је обављен комплетан пренос података из периферије **PER0** у меморију и реализована обрада. Уколико би се по аналогији са задатком 1.1 усвојило да се одмах на почетку иницијализује и стартује не само контролер периферије **KPER0** већ и контролер периферије **KPER1,** тада би део програма са слике 5

```
;inicijalizacija prenosa iz memorije u periferiju PER1 sa KPER1
  ;veličina bloka podataka
    LOAD   #200h     ;upis veličine bloka podataka za PER1 preko ACC u
    STORE  MemCnt1   ;mem. lok. na adr. MemCnt1
  ;početna adresa bloka podataka
    LOAD   #F000h    ;upis adr. bloka podataka za PER1 preko ACC u
    STORE  MemAdr1   ;mem. lok. na adr. MemAdr1
;postavljanje "semafor"-a za PER1 na 1
    LOAD   #1h       ;upis vrednosti 1 preko ACC u
    STORE  MemSem1   ;mem. lok. na adr. MemSem1
;startovanje kontrolera periferije KPER1
    LOAD   #8003h    ;upis režima rada i startovanja (start=1,
    OUT    FF10h     ;u/i=1,enable=1)preko ACC u reg. Control KPER1
```

којим се иницијализује и стартује контролер периферије **KPER1** требало преместити и ставити између дела програма којим се иницијализује и стартује контролер периферије **KPER0** и дела програма којим се у петљи Loop0 реализује унос блока података из периферије **PER0** у меморију. Међутим, с обзиром да се у овом задатку сам пренос података из меморије у периферију **PER1** реализује техником програмираног излаза са прекидом, одмах по стартовању контролер периферије **KPER1** би кренуо са генерисањем прекида, па би се преласцима на прекидну рутину периферије **PER1** кренуло са слањем података у периферију иако извршавања програма којим се обавља пренос података из периферије **PER0** у меморију и реализује обрада није завршен. Ово се не сме дозволити, па се може избећи тако што би се стартовање контролера периферије **KPER1** уместо инструкцијама

```
;startovanje kontrolera periferije KPER1
     LOAD   #8003h    ;upis režima rada i startovanja (start=1,
     OUT    FF10h     ;u/i=1,enable=1)preko ACC u reg. Control KPER1
```

реализовало инструкцијама

```
;startovanje kontrolera periferije KPER1
     LOAD   #8002h    ;upis režima rada i startovanja (start=1,
     OUT    FF10h     ;u/i=1,enable=0)preko ACC u reg. Control KPER1
```

којима би се у бит *enable* управљачког регистра *Control* **KPER1** уместо вредности 1 уписивала вредност 0. Због вредности 0 бита *enable* контролер **KPER1** неће генерисати прекиде, па неће бити прелазака на прекидну рутину периферије **PER1** и слања података у периферију **PER1** све док се у бит *enable* управљачког регистра *Control* **KPER1** не упише вредност 1 и тиме дозволи генерисања прекида. То се сме учинити тек када пренос података из периферије **PER0** у меморију и обрада буду завршени и то инструкцијама

```
      IN     FF10h     ;prebacivanje Control KPER1 u ACC, izvršavanje
      OR     #0001h    ;operacije ILI sa neposred vel koja na pozic bita
      OUT    FF10h     ;enable ima 1 i vraćanje u reg. Control KPER1
```

које треба ставити пре петље `Wait1`.

Међутим тиме се не добија ништа, јер се на извршавање дела програма којим се реализује сам пренос података из меморије у периферију **PER1** прелази тек после извршавања програма којим се обавља комплетан пренос података из периферије **PER0** у меморију и реализује обрада. Контролер периферије **KPER1**, иако би био иницијализован и стартован одмах на почетку, али у режиму без генерисања прекида, не би кренуо са самим слањем података из меморије у периферију **PER1** све док се, непосредно пре дела програма са петљом `Wait1`, режим рада контролера **KPER1** не модификује тако што се дозволи генерисање прекида.

1.3 ЗАДАТАК

Дат је рачунарски систем из задатка 1.1.

Написати програм којим се блок од 200h података учитава из **PER0**, смешта у меморију од локације F000h, обрађује процедуром *Obrada* и резултат шаље у **PER1**. Улаз из периферије **PER0** реализовати са контролером периферије **KPER0** техником програмираног улаза са прекидом, а излаз у периферију **PER1** реализовати са контролером периферије **KPER1** техником програмираног излаза са испитивањем спремности. Процедуру `Obrada` не треба реализовати, већ је само позвати на одговарајућем месту у програму помоћу инструкције `JSR Obrada`. Процедура `Obrada` не мења место и дужину блока података.

Решење:

Тражени програм је приказан на слици 6.

```
;glavni program
;unos bloka podat iz periferije PER0 u memoriju sa KPER0 sa prekidom
  ;inicijalizacija prenosa iz periferije PER0 u memoriju sa KPER0
    ;veličina bloka podataka
      LOAD   #200h     ;upis veličine bloka podataka za PER0 preko ACC u
      STORE  MemCnt0   ;mem. lok. na adr. MemCnt0
    ;početna adresa bloka podataka
      LOAD   #F000h    ;upis adr. bloka podataka za PER0 preko ACC u
      STORE  MemAdr0   ;mem. lok. na adr. MemAdr0
  ;postavljanje "semafor"-a za PER0 na 1
      LOAD   #1h       ;upis vrednosti 1 preko ACC u
      STORE  MemSem0   ;mem. lok. na adr. MemSem0
  ;startovanje kontrolera periferije KPER0
      LOAD   #8001h    ;upis režima rada i startovanja (start=1,
      OUT    FF00h     ;u/i=0,enable=1)preko ACC u reg. Control KPER0
  ;čekanje na završetak unosa bloka podataka iz periferiju PER0 u memoriju
  ;proverom vrednosti "semafor"-a za PER0
Wait0: LOAD   MemSem0   ;upis sadrž. mem.lok. sa adr. MemSem0 u ACC
       CMP    #0        ;ispitivanje da li je "semafor" postao 0
       JNZ    Wait0     ;povratak na Wait0 ukoliko "semafor" nije 0
;obrada
       JSR    Obrada    ;obrada bloka od 200h podataka u memorijskim
                        ;lokacijama od adrese F000h do adrese F1FFh
;slanje bloka podataka iz memorije u periferiju PER1 sa KPER1 sa
;ispitivanjem bita spremnosti ready
  ;inicijalizacija prenosa iz memorije u periferiju PER1 sa KPER1
    ;veličina bloka podataka
      LOAD   #200h     ;upis veličine bloka podataka za PER1 preko ACC u
      STORE  MemCnt1   ;mem. lok. na adr. MemCnt1
    ;početna adresa bloka podataka
      LOAD   #F000h    ;upis adr. bloka podataka za PER1 preko ACC u
      STORE  MemAdr1   ;mem. lok. na adr. MemAdr1
  ;startovanje kontrolera periferije KPER1
      LOAD   #8002h    ;upis režima rada i startovanja PER1 (start=1,
      OUT    FF10h     ;u/i=1,enable=0)preko ACC u reg. Control PER1
  ;slanje bloka podataka
    ; provera da li podatak može da se prenese u registar Data KPER1
Loop1: IN     FF11h     ;upis sadrž. reg. Status KPER1 u ACC
       AND    #1        ;ispitivanje bita ready
       JZ     Loop1     ;povratak na Loop1 ukoliko je ready 0
    ;prenos podatka iz memorijske lokacije u registar Data KPER1
      LOAD   (MemAdr1) ;upis sadrž. mem. lok. sa adr. date sadrž. mem.
      OUT    FF12h     ;lok. sa adr. MemAdr1 preko ACC u reg. Data KPER1
```

```
     ;inkrementiranje sadržaja memorijske lokacije MemAdr1
       LOAD   MemAdr1  ;upis sadrž. mem.lok. sa adr. MemAdr1 u ACC
       INC             ;inkrementiranje sadrž. ACC
       STORE  MemAdr1  ;upis sadrž. ACC u mem.lok. na adr. MemAdr1
     ;dekrementiranje sadržaja memorijske lokacije MemCnt
       LOAD   MemCnt1  ;upis sadrž. mem.lok. sa adr. MemCnt1 u ACC
       DEC             ;dekrementiranje sadrž. ACC
       STORE  MemCnt1  ;upis sadrž. ACC u mem.lok. na adr. MemCnt1
     ;provera da li je prenet poslednji podatak
       JNZ    Loop1    ;povratak na Loop1 ukoliko nije poslednji podatak
   ;zaustavljanje kontrolera periferije KPER1
       LOAD   #0       ;upis režima zaustavljanja (start=0, u/i=0,
       OUT    FF10h    ;enable=0)preko ACC u reg. Control KPER1
. . .
;prekidna rutina periferije PER0
;unos bloka podataka iz periferiju PER0 u memoriju sa KPER0
PER0:  PUSHA           ;akumulator ACC na stek
     ;prenos podatka iz registra Data KPER0 u memorijsku lokaciju
       IN     FF02h    ;upis sadrž. reg. Data KPER0 preko ACC u mem. lok.
       STORE  (MemAdr0) ;na adresi datoj sadrž. mem. lok. na adr. MemAdr0
     ;inkrementiranje sadržaja memorijske lokacije MemAdr0
       LOAD   MemAdr0  ;upis sadrž. mem.lok. sa adr. MemAdr0 u ACC
       INC             ;inkrementiranje sadrž. ACC
       STORE  MemAdr0  ;upis sadrž. ACC u mem.lok. na adr. MemAdr0
     ;dekrementiranje sadržaja memorijske lokacije MemCnt
       LOAD   MemCnt0  ;upis sadrž. mem.lok. sa adr. MemCnt0 u ACC
       DEC             ;dekrementiranje sadrž. ACC
       STORE  MemCnt0  ;upis sadrž. ACC u mem.lok. na adr. MemCnt0
   ;provera da li je prenet poslednji podatak
       JNZ    Back0    ;prelaz na Back0 ukoliko nije poslednji podatak
   ;zaustavljanje kontrolera periferije KPER0
       LOAD   #0       ;upis režima zaustavljanja (start=0, u/i=0,
       OUT    FF00H    ;enable=0)preko ACC u reg. Control KPER0
   ;postavljanje "semafor"-a za PER0 na 0
                       ;upis vrednosti 0 preko ACC u
       STORE  MemSem0  ;mem. lok. na adr. MemSem0
   ;izlazak iz prekidne rutine
Back0: POPA            ;restauracija akumulatora ACC sa steka
       RTI             ;povratak iz prekidne rutine
. . .
```

Слика 6 Програм

Програм се састоји из три дела. У првом делу се из периферије **PER0** техником програмираног улаза са прекидом уноси блок од 200h података и смешта у меморијске локације од адресе F000h до адресе F1FFh. У другом делу се позива процесура *Obrada* која врши одређену обраду над унетим подацима у блоку података у меморијским локацијама од адресе F000h до адресе F1FFh и резултат смешта у исте меморијске локације од адресе F000h до адресе F1FFh. У трећем делу се у периферију **PER1** техником програмираног излаза са испитивањем бита спремности *ready* статусног регистра *Status* **KPER1** шаље блок од 200h података прочитаних из меморијских локација од адресе F000h до адресе F1FFh. Други и трећи део су идентични као други и трећи део у задатку 1.1. У првом делу постоји разлика јер се за унос блока података из периферије **PER0** у меморију уместо технике програмираног излаза са испитивањем бита спремности *ready* статусног регистра *Status* **KPER0** користи техника програмираног улаза са прекидом.

Део програма којим се уноси блок података из периферије **PER0** у меморију има два дела и то један део који се извршава у главном програму и други део који се извршава у

прекидној рутини периферије **PER0**. У оквиру дела програма који се извршава у главном програму врши се иницијализација преноса из периферију **PER0**, постављање "semafor"-а за **PER0** на 1, стартовање контролера периферије **KPER0** и чекање на завршетак уноса блока података из периферије **PER0** у меморију. У оквиру дела програма који се извршава у прекидној рутини периферије **PER0** врши се пренос података из регистра *Data* **KPER0** у меморијску локацију, инкрементирање садржаја меморијске локације MemAdr0 и декрементирање садржаја меморијске локације MemCnt0 и, уколико је пренет последњи податак, заустављање контролера периферије **KPER0** и постављање "semafor" за **PER0** на 0.

У оквиру иницијализације преноса у периферију **PER0** се најпре у меморијску локацију чија је адреса симболички означена са MemCnt0 уписује вредност 200h која представља величину блока података који треба пренети, а затим се у меморијску локацију чија је адреса симболички означена са MemAdr0 уписује вредност F000h која представља почетну адресу дела меморије у који треба пренети блок података. У оквиру постављања "semafor"-а за **PER0** на 1, у меморијску локацију MemSem0 се уписује вредност 1 која служи као индикација да је пренос података из периферије **PER0** у меморију у току и да се не сме продужити са извршавањем главног програма док пренос не буде завршен. У оквиру стартовања контролера периферије **KPER0** у управљачки регистар *Control* **KPER0**, који се налази на адреси FF00h у улазно/излазном адресном простору, се уписује вредност 8001h чиме се контролер стартује (*start*=1) за рад у режиму улаза (*u/i*=0) са генерисањем прекида (*enable*=1).

Од овог тренутка процесор и контролер периферије **KPER0** почињу да раде паралелно. Процесор чека завршетак преноса блока података из периферију **PER0** у меморију у петљи са лабелом Wait0. Контролер периферије **KPER0**, најпре, пошто је стартован (*start*=1) за рад у режиму улаза (*u/i*=0), поставља на вредност 0 бит спремности *ready* статусног регистра *Status* **KPER0**, затим, започиње пренос податка из периферије **PER0** у регистар податка *Data* **KPER0**, потом, по упису податка у регистар *Data* **KPER0**, поставља бит *ready* статусног регистра *Status* **KPER0** на вредност 1 и, пошто је стартован (*start*=1) за рад са генерисањем прекида (*enable*=1), генерише прекид и на крају чека да се податак пренесе из регистра *Data* **KPER0** у меморијску локацију.

На прекид генерисан од стране контролера периферије **KPER0**, процесор излази из петље са лабелом Wait0 и прелази на извршавање инструкција прекидне рутине периферије **PER0** на чијем почетку преноси податак из регистра *Data* **KPER0** у меморијску локације. Од овог тренутка процесор и контролер периферије **KPER0** настављају да раде паралелно. Процесор наставља са извршавањем инструкција прекидне рутине тако што инкрементира садржај меморијске локадине MemAdr0, декрементира садржај меморијске локације MemCnt0 и, уколико није пренет последњи податак, враћа се у петљу са лабелом Wait0 прекинутог главног програма. Контролер периферије **KPER0**, најпре, поставља на вредност 0 бит спремности *ready* статусног регистра *Status* **KPER0**, затим, започиње пренос податка из периферије **PER0** у регистар податка *Data* **KPER0**, потом, по упису податка у регистар *Data* **KPER0**, поставља бит *ready* статусног регистра *Status* **KPER0** на вредност 1 и генерише прекид и на крају чека да се податак пренесе из регистра *Data* **KPER0** у меморијску локацију.

Рад процесора и контролера приказан у претходном параграфу се наставља до преласка процесора на прекидну рутину периферије **PER0** ради преноса последњег податка. Када садржај меморијске локације MemCnt0 декрементирањем постане 0, процесор најпре изврши инструкције прекидне рутине којима се поставља "semafor" за **PER0** на 0 и зауставља контролер периферије **KPER0**, па се затим враћа у петљу са лабелом Wait0 прекинутог главног програма.

Током наставка извршавања програма у петљи са лабелом Wait0 утврдиће се да "semafor" за **PER0** сада има вредност 0. Ова вредност служи као индикација да је пренос података завршен, па се излази из петље са лабелом Wait0 и продужава извршавање главног програма.

Првом инструкцијом прекидне рутине PUSHA на стеку се чува садржај акумулатора ACC прекинутог главног програма, док се предзадњом инструкцијом прекидне рутине POPA садржај акумулатора ACC прекинутог главног програма рестаурира вредношћу са стека. Ово је неопходно учинити јер се у главном програму користи садржај акумулатора ACC, а у прекидној рутини се садржај акумулатора ACC мења. Садржаји регистара PC и PSW прекинутог главног програма се чувају на стеку хардверски у оквиру фазе опслуживање прекида инструкције главног програма из које се прелази у прекидну рутину, док се задњом инструкцијом прекидне рутине RTI садржаји регистара PSW и PC прекинутог главног програма рестаурирају вредностима са стека.

Дискусија:

1. Разматрања везана за иницијализацију и стартовање контролера периферије **KPER1** у овом задатку су идентична као и разматрања везана за иницијализацију и стартовање контролера периферије **KPER1** у задатку 1.1 и дата су у задатку 1.1 у одељку **Дискусија** под **4**.

1.4 ЗАДАТАК

Дат је рачунарски систем који чине процесор, меморија и контролери периферија **KPER0**, **KPER1** и **KPER2** са периферијама **PER0**, **PER1** и **PER2**, респективно, повезани магистралом.

Процесор је са једноадресним форматом инструкција. Од програмски доступних регистара постоји 16 битни акумулатор ACC, указивач на врх стека SP, указивач на табелу (IV табела) са адресама прекидних рутина IVTP и програмска статусна реч PSW. Типови података који се користе су 8 битне и 16 битне целобројне величине са знаком и без знака. Инструкције преноса са акумулатором, аритметичке, логичке и померачке инструкција се извршавају и над 8 битним и над 16 битним целобројним величинама. Приликом извршавања инструкција над 8 битним целобројним величинама користи се само нижих 8 битова акумулатора, а приликом извршавања инструкција над 16 битним бинарним речима свих 16 битова. Приликом извршавања инструкције преноса у акумулатор и аритметичких, логичких и померачких инструкција постављају се индикатори N, Z, C и V програмске статусне речи PSW. При прекиду хардверски се на стеку чувају програмски бројач PC и програмска статусна реч PSW. Улазно-излазни адресни простор је меморијски пресликан.

Меморијски адресни простор је величине 2^{16} меморијских локација, при чему део меморијских локација чине регистри контролера периферија **KPER0**, **KPER1** и **KPER2**. Ширина меморијских локација и регистара контролера периферија **KPER0**, **KPER1** и **KPER2** је 8 битова.

Контролери периферија **KPER0**, **KPER1** и **KPER2** имају управљачке регистре (*Control*), статусне регистре (*Status*) и регистре података (*Data*) и то редом на адресама FF00h, FF01h и FF02h за контролер периферије **KPER0**, FF10h, FF11h и FF12h за контролер периферије **KPER1** и FF20h, FF21h и FF22h за контролер периферије **KPER2**. У регистрима контролера периферија **KPER0**, **KPER1** и **KPER2** највиши бит је означен са 7, а најнижи са 0. У управљачким регистрима *Control* бит 7 је бит *start* којим се покреће извршавање операције (0—заустављен, 1—стартован), бит 3 је бит *u/i* којим се задаје смер преноса са периферијом (0—улаз, 1—излаз) и бит 0 је бит *enable* којим се дозвољава прекид (0—маскиран, 1—дозвољен). У статусним регистрима *Status* бит 7 је бит спремности *ready* (0—није спреман, 1—спреман).

Адресне линије и линије података магистрале рачунара су широке 16 битова и 8 битова, респективно.

Написати главни програм и одговарајуће прекидне рутине којима се истовремено из **PER0** и **PER1** учитавају блокови од по 100h бајтова и смештају у меморију од адресе 1000h до адресе 10FFh за **PER0** и од адресе 1100h до адресе 11FFh за **PER1**, затим блок од 200h унетих бајтова од адресе 1000h до адресе 11FFh обрађује процесуром `Obrada` и на крају врши пренос обрађеног блока од 200h бајтова из меморије у **PER2**. Улаз са периферија **PER0** и **PER1** реализовати са контролерима периферија **KPER0** и **KPER1** техником програмираног улаза са прекидом, а излаз у периферију **PER2** са контролером периферије **KPER2** техником програмираног излаза са испитивањем спремности. Процедуру `Obrada` не треба реализовати, већ је само позвати на одговарајућем месту у програму помоћу инструкције `JSR Obrada`. Процедура `Obrada` не мења место и дужину блока података

Решење:

Периферије **PER0**, **PER1** и **PER2** се повезују на магистралу рачунарског система са контролерима периферија **KPER0**, **KPER1** и **KPER2** на начин приказан на слици 1.

Адресе и структура регистара контролера периферија **KPER0**, **KPER1** и **KPER2** су дати на сликама 7 и 8, респективно.

KPER0 - Kontroler periferije **PER0**

Registar	Control	Stataus	Data
Adresa	FF00h	FF01h	FF02h

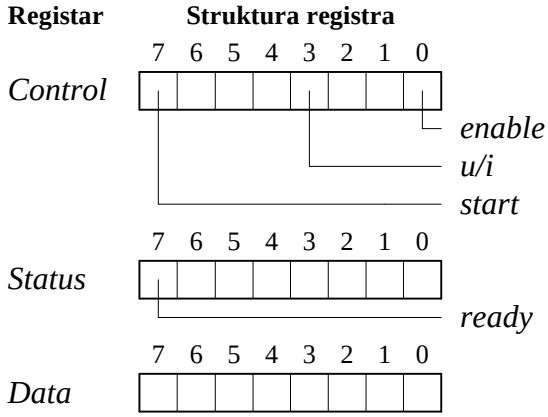
KPER1 - Kontroler periferije **PER1**

Registar	Control	Stataus	Data
Adresa	FF10h	FF11h	FF12h

KPER2 - Kontroler periferije **PER2**

Registar	Control	Stataus	Data
Adresa	FF20h	FF21h	FF22h

Слика 7 Адресе регистара контролера



Слика 8 Структура регистара контролера

Тражени програм је приказан на слици 9

```
;glavni program
;unos blokova podataka iz periferije PER0 u memoriju sa KPER0 sa prekidom i
;iz periferije PER1 u memoriju sa KPER1 sa prekidom
  ;inicijalizacija prenosa iz periferija PER0 u memoriju sa KPER0 i
  ;iz periferija PER1 u memoriju sa KPER1
    ;veličine blokova podataka
      LOADW  #100h     ;upis veličine blokova podataka preko ACC u
      STOREW MemCnt0   ;mem. lok. na adr. MemCnt0 za PER0
      STOREW MemCnt1   ;mem. lok. na adr. MemCnt1 za PER1
    ;početne adrese blokova podataka
      LOADW  #1000h    ;upis adr. bloka podataka za PER0 preko ACC u
      STOREW MemAdr0   ;mem. lok. na adr. MemAdr0
      LOADW  #1100h    ;upis adr. bloka podataka za PER1 preko ACC u
      STOREW MemAdr1   ;mem. lok. na adr. MemAdr1
  ;postavljanje "semafor"-a za periferije PER0 i PER1 na 1
       LOADB  #1       ;upis vrednosti 1 preko ACC u
       STOREB MemSem0  ;mem. lok. na adr. MemSem0 za PER0
       STOREB MemSem1  ;mem. lok. na adr. MemSem1 za PER1
  ;startovanje kontrolera periferija KPER0 i KPER1
```

```
        LOADB  #81h      ;upis režima rada i startovanja
                         ;(start=1,u/i=0,enable=1)preko ACC
        STOREB FF00h     ;u reg. Control KPER0
        STOREB FF10h     ;u reg. Control KPER1
   ;čekanje na završetak unosa bloka podataka iz periferije PER0 u memoriju
   ;proverom vrednosti "semafor"-a za PER0
Wait0: LOADB  MemSem0   ;upis sadrž. mem.lok. sa adr. MemSem0 u ACC
        CMPB   #0        ;provera da li je "semafor" za PER0 postao 0
        JNZ    Wait0     ;povratak na Wait0 ukoliko "semafor" nije 0
   ;čekanje na završetak unosa bloka podataka iz periferije PER1 u memoriju
   ;proverom vrednosti "semafor"-a za PER1
Wait1: LOADB  MemSem1   ;upis sadrž. mem.lok. sa adr. MemSem1 u ACC
        CMPB   #0        ;provera da li je "semafor" za PER1 postao 0
        JNZ    Wait1     ;povratak na Wait1 ukoliko "semafor" nije 0
;obrada
        JSR    Obrada    ;obrada bloka od 200h podataka u memorijskim
                         ;lokacijama od adrese F000h do adrese F1FFh
;slanje bloka podataka iz memorije u periferiju PER2 sa KPER2 sa
;ispitivanjem bita spremnosti ready
   ;inicijalizacija prenosa iz memorije u periferiju PER2 sa KPER2
     ;veličina bloka podataka
        LOADW  #200h     ;upis veličine bloka podataka za PER2 preko ACC u
        STOREW MemCnt2   ;mem. lok. na adr. MemCnt2
     ;početne adrese bloka podataka
        LOADW  #1000h    ;upis adr. bloka podataka za PER2 preko ACC u
        STOREW MemAdr2   ;mem. lok. na adr. MemAdr2
   ;startovanje kontrolera periferije KPER2
        LOADB  #88h      ;upis režima rada i startovanja (start=1,
        STOREB FF20h     ;u/i=1,enable=0)preko ACC u reg. Control KPER2
   ;slanje bloka podataka
     ;provera da li podatak može da se prenese u registar Data KPER2
Loop2: LOADB  FF21h     ;upis sadrž. reg. Status KPER2 u ACC
        ANDB   #80h      ;ispitivanje bita ready
        JZ     Loop2     ;povratak na Loop2 ukoliko je ready 0
     ;prenos podatka iz memorijske lokacije u registar Data KPER2
        LOADB  (MemAdr2) ;upis sadrž. mem. lok. sa adr. date sadrž. mem.
        STOREB FF22h     ;lok. sa adr. MemAdr2 preko ACC u reg. Data KPER2
     ;inkrementiranje sadržaja memorijske lokacije MemAdr2
        LOADW  MemAdr2   ;upis sadrž. mem.lok. sa adr. MemAdr2 u ACC
        INCW             ;inkrementiranje sadrž. ACC
        STOREW MemAdr2   ;upis sadrž. ACC u mem.lok. na adr. MemAdr2
     ;dekrementiranje sadržaja memorijske lokacije MemCnt2
        LOADW  MemCnt2   ;upis sadrž. mem.lok. sa adr. MemCnt2 u ACC
        DECW             ;dekrementiranje sadrž. ACC
        STOREW MemCnt2   ;upis sadrž. ACC u mem.lok. na adr. MemCnt2
     ;provera da li je prenet poslednji podatak
        JNZ    Loop2     ;povratak na Loop2 ukoliko nije poslednji podatak
   ;zaustavljanje kontrolera periferije KPER2
        LOADB  #0        ;upis režima zaustavljanja (start=0, u/i=0,
        STOREB FF20h     ;enable=0)preko ACC u reg. Control KPER2

. . .

;prekidna rutina periferije PER0
; unos bloka podataka iz periferiju PER0 u memoriju sa KPER0
PER0:  PUSHA             ;akumulator ACC na stek
     ;prenos podatka iz registra Data KPER0 u memorijsku lokaciju
        LOADB  FF02h     ;upis sadrž. reg. Data KPER0 preko ACC u mem. lok.
        STOREB (MemAdr0) ;na adresi datoj sadrž. mem. lok. na adr. MemAdr0
     ;inkrementiranje sadržaja memorijske lokacije MemAdr0
        LOADW  MemAdr0   ;upis sadrž. mem.lok. sa adr. MemAdr0 u ACC
        INCW             ;inkrementiranje sadrž. ACC
```

```
        STOREW MemAdr0   ;upis sadrž. ACC u mem.lok. na adr. MemAdr0
      ;dekrementiranje sadržaja memorijske lokacije MemCnt
        LOADW  MemCnt0   ;upis sadrž. mem.lok. sa adr. MemCnt0 u ACC
        DECW             ;dekrementiranje sadrž. ACC
        STOREW MemCnt0   ;upis sadrž. ACC u mem.lok. na adr. MemCnt0
    ;provera da li je unet poslednji podatak
        JNZ    Back0     ;prelaz na Back0 ukoliko nije poslednji podatak
    ;zaustavljanje kontrolera periferije KPER0
        LOADB  #0        ;upis režima zaustavljanja(start=0, u/i=0,
        STOREB FF00h     ;enable=0)preko ACC u reg. Control KPER0
    ;postavljanje "semafor"-a na 0 za PER0
        LOADB  #0        ;upis vrednosti 0 preko ACC u
        STOREB MemSem0   ;mem. lok. na adr. MemSem0
    ;izlazak iz prekidne rutine
Back0: POPA              ;restauracija akumulatora ACC sa steka
        RTI              ;povratak iz prekidne rutine

. . .

;prekidna rutina periferije PER1
;unos bloka podataka iz periferiju PER1 u memoriju sa KPER1
PER1:  PUSHA             ;akumulator ACC na stek
      ;prenos podatka iz registra Data KPER1 u memorijsku lokaciju
        LOADB  FF12h     ;upis sadrž. reg. Data KPER1 preko ACC u mem. lok.
        STOREB (MemAdr1) ;na adresi datoj sadrž. mem. lok. na adr. MemAdr1
      ;inkrementiranje sadržaja memorijske lokacije MemAdr1
        LOADW  MemAdr1   ;upis sadrž. mem.lok. sa adr. MemAdr1 u ACC
        INCW             ;inkrementiranje sadrž. ACC
        STOREW MemAdr1   ;upis sadrž. ACC u mem.lok. na adr. MemAdr1
      ;dekrementiranje sadržaja memorijske lokacije MemCnt1
        LOADW  MemCnt1   ;upis sadrž. mem.lok. sa adr. MemCnt1 u ACC
        DECW             ;dekrementiranje sadrž. ACC
        STOREW MemCnt1   ;upis sadrž. ACC u mem.lok. na adr. MemCnt1
    ;provera da li je unet poslednji podatak
        JNZ    Back1     ;prelaz na Back1 ukoliko nije poslednji podatak
    ;zaustavljanje kontrolera periferije KPER1
        LOADB  #0        ;upis režima zaustavljanja(start=0, u/i=0,
        STOREB FF10h     ;enable=0)preko ACC u reg. Control KPER1
    ;postavljanje "semafor"-a na 0 za PER1
        LOADB  #0        ;upis vrednosti 0 preko ACC u
        STOREB MemSem1   ;mem. lok. na adr. MemSem1
    ;izlazak iz prekidne rutine
Back1: POPA              ;restauracija akumulatora ACC sa stek
        RTI              ;povratak iz prekidne rutine
. . .
```

Слика 9 Програм

Програм се састоји из три дела. У првом делу се из периферија **PER0** и **PER1** техником програмираног улаза са прекидом уносе два блока од по 100h података и смештају у меморијске локације од адресе 1000h до адресе 10FFh за периферију **PER0** и од адресе 1100h до адресе 11FFh за периферију **PER1**. У другом делу се позива процесура `Obrada` која врши одређену обраду над унетим подацима у блоковима од по 100h података у меморијским локацијама од адресе 1000h до адресе 10FFh и од адресе 1100h до адресе 11FFh и резултат блок од 200h података смешта у исте меморијске локације од адресе 1000h до адресе 11FFh. У трећем делу се у периферију **PER2** техником програмираног излаза са испитивањем бита спремности *ready* статусног регистра *Status* KPER2 шаље блок од 200h података прочитаних из меморијских локација од адресе 1000h до адресе 11FFh.

Уноси података из периферија **PER0** и **PER1** у меморију се реализује упоредо техником програмираног улаза са прекидом на начин објашњен у задатку 1.3. Због тога се у првом делу главног програма иницијализују преноси из обе периферије, постављају на 1 "semafor"-и за обе периферије и стартују контролери обе периферије. Прелазак на други део програма у коме се реализује обрада унетих блокова података не сме да се дозволи док се не унесу блокови података из обе периферије. Чекање на завршетак уноса блокова од по 100h података из обе периферије се реализује најпре у петљи са лабелом Wait0 за периферију **PER0** а затим и у петљи са лабелом Wait1 за периферију **PER1**, а сами уноси података из периферија **PER0** и **PER1** у меморију се реализују у прекидним рутинама периферија **PER0** и **PER1** на које се прелази када контролери периферија **KPER0** и **KPER1** генеришу прекиде. Том приликом могу да се јаве два случаја у зависности од брзине периферија **PER0** и **PER1**. Први случај се јавља када је периферија **PER0** бржа од периферије **PER1**, па ће се унос података из периферије **PER0** завршити пре од уноса података из периферије **PER1** и прво "semafor" периферије **PER0** поставити на вредност 0 а затим и "semafor" периферије **PER1** поставити на вредност 0. Због тога ће се из петље са лабелом Wait0 изаћи чим унос података из периферије **PER0** буде завршен и "semafor" периферије **PER0** буде постављен на вредност 0, али ће се затим у петљи са лабелом Wait1 чекати да буде завршен и унос података из периферије **PER1** и да "semafor" периферије **PER1** буде постављен на вредност 0 и тек онда прећи на други део програма у коме се реализује обрада унетих блокова података. Други случај се јавља када је периферија **PER0** спорија од периферије **PER1**, па ће се унос података из периферије **PER1** завршити пре од уноса података из периферије **PER0** и прво "semafor" периферије **PER1** поставити на вредност 0 а затим и "semafor" периферије **PER0** поставити на вредност 0. Због тога ће се из петље са лабелом Wait0 изаћи чим унос података из периферије **PER0** буде завршен и "semafor" периферије **PER0** буде постављен на вредност 0, али се у петљи са лабелом Wait1 неће чекати, јер ће унос података из периферије **PER1** бити завршен и "semafor" периферије **PER1** постављен на вредност 0, па ће се одмах прећи на други део програма у коме се реализује обрада унетих блокова података.

Други део у коме се врши обрада и трећи део у коме се у периферију **PER2** техником програмираног излаза са испитивањем бита спремности *ready* статусног регистра *Status* **KPER2** шаље блок података се реализује на начин објашњен у задатку 1.3

Дискусија:

1. У решењу задатка су на почетку покренуте улазне операције и са периферије **PER0** и са периферије **PER1**, па се затим најпре у петљама са лабелама Wait0 и Wait1 чека на завршетак улазних операција са обе периферије па тек после тога прелази на извршавање процедуре `Obrada`. Решење по коме би се само покренула улазна операција са једне периферије и тек по њеном завршетку покренула улазна операција са друге периферије је исправно али не и ефикасно, јер би се тиме улазне операције са периферија **PER0** и **PER1** непотребно секвенцијално извршавале. Битан ефекат технике програмираног улаза са прекидом је управо у томе што контролери обе периферије могу паралелно да извршавају уносе података из периферија у регистре *Data* контролера периферија. Ипак, процесор не може истовремено преносити саме податке из регистара *Data* оба контролера у меморију, већ се то обавља секвенцијално преласком процесора на извршавање прекидних рутина периферија **PER0** и **PER1** онако како прекиди од периферија буду генерисани. Добитак на времену је у томе што саме периферије и њихови контролери раде паралелно и независно.

2. Улазно-излазни адресни простор је меморијски пресликан, процесор је са једноадресним форматом инструкција, а регистри контролера периферија су дужине 8

битова. Због тога се за приступ регистрима контролера периферија користе инструкције LOADB и STOREB.

3. Инструкције за рад са 8-битним величинама се користе за преношење 8-битних података у/из меморије и регистара контролера периферија и када су 8-битне вредности довољне за представљање одређених величина ("semafor"). Инструкције за рад са 16-битним величинама се користе за рад са 16-битним адресама и када 8-битни подаци нису довољни за представљање одређених величина већ морају да се користе 16-битни подаци (величина блока података).

4. Разматрања везана за иницијализацију и стартовање контролера периферије **KPER2** у овом задатку су идентична као и разматрања везана за иницијализацију и стартовање контролера периферије **KPER1** у задатку 1.1 и дата су у задатку 1.1 у одељку **Дискусија** под **4**.

1.5 ЗАДАТАК

Дат је рачунарски систем из задатка 1.4.

Написати главни програм и одговарајуће прекидне рутине којима се блок од 200h података учитава из **PER0**, смешта у меморију почев од адресе 2000h до адресе 21FFh, обрађује процедуром `Obrada` и резултат шаље у **PER1** и **PER2**. Улаз из периферије **PER0** реализовати са контролером периферије **KPER0** и то техником програмираног улаза са испитивањем спремности, а излаз на периферије **PER1** и **PER2** реализовати упоредо са контролерима периферије **KPER1** и **KPER2** техником програмираног излаза са прекидом. Процедуру `Obrada` не треба реализовати, већ је само позвати на одговарајућем месту у програму помоћу инструкције `JSR Obrada`. Процедура `Obrada` не мења место и дужину блока података

Решење:

Периферије **PER0**, **PER1** и **PER2** се повезују на магистралу рачунарског система са контролерима периферија **KPER0**, **KPER1** и **KPER2** на начин приказан на слици 1.

Тражени програм је приказан на слици 10.

```
;glavni program
;unos bloka podataka iz periferija PER0 u memoriju sa KPER0 sa ispitivanjem
;bita spremnosti ready
  ;inicijalizacija prenosa iz periferije PER0 u memoriju sa KPER0
    ;veličina bloka podataka
      LOADW  #200h     ;upis veličine bloka podataka preko ACC u
      STOREW MemCnt0   ;mem. lok. na adr. MemCnt0 za PER0
    ;početna adresa bloka podataka
      LOADW  #2000h    ;upis adr. bloka podataka za PER0 preko ACC u
      STOREW MemAdr0   ;mem. lok. na adr. MemAdr0
  ;startovanje kontrolera periferije KPER0
      LOADB  #80h      ;upis režima rada i startovanja
                       ;(start=1,u/i=0,enable=0)preko ACC
      STOREB FF00h     ;u reg. Control KPER0
  ;unos bloka podataka
    ;provera da li podatak može da se prenese iz registra Data KPER0
Loop0: LOADB  FF01h    ;upis sadrž. reg. Status KPER0 u ACC
       ANDB   #80h     ;ispitivanje bita ready
       JZ     Loop0    ;povratak na Loop0 ukoliko je ready 0
    ;prenos podatka iz registra Data KPER0 u memorijsku lokaciju
      LOADB  FF02h     ;upis sadrž. reg. Data KPER0 preko ACC u mem. lok.
      STORE  (MemAdr0) ;na adresi datoj sadrž. mem. lok. na adr. MemAdr0
    ;inkrementiranje sadržaja memorijske lokacije MemAdr0
      LOADW  MemAdr0   ;upis sadrž. mem.lok. sa adr. MemAdr0 u ACC
      INCW             ;inkrementiranje sadrž. ACC
      STOREW MemAdr0   ;upis sadrž. ACC u mem.lok. na adr. MemAdr0
    ;dekrementiranje sadržaja memorijske lokacije MemCnt0
      LOADW  MemCnt0   ;upis sadrž. mem.lok. sa adr. MemCnt0 u ACC
      DECW             ;dekrementiranje sadrž. ACC
      STOREW MemCnt0   ;upis sadrž. ACC u mem.lok. na adr. MemCnt0
    ;provera da li je unet poslednji podatak
      JNZ    Loop0     ;povratak na Loop0 ukoliko nije poslednji podatak
  ;zaustavljanje kontrolera periferije KPER0
      LOADB  #0        ;upis režima zaustavljanja (start=0, u/i=0,
      STOREB FF00h     ;enable=0)preko ACC u reg. Control KPER0
;obrada
       JSR    Obrada   ;obrada bloka od 200h podataka u memorijskim
                       ;lokacijama od adrese 2000h do adrese 20FFh
```

```
;slanje bloka podataka iz memorije u periferiju PER1 sa KPER1 sa prekidom i
;iz memorije u periferiju PER2 sa KPER2 sa prekidom
  ;inicijalizacija prenosa iz mem u perif PER1 sa KPER1 i u PER2 sa KPER2
    ;veličine blokova podataka
      LOADW  #200h     ;upis veličine bloka podataka preko ACC u
      STOREW MemCnt1   ;mem. lok. na adr. MemCnt1 za PER1 i
      STOREW MemCnt2   ;mem. lok. na adr. MemCnt2 za PER2
    ;početne adrese blokova podataka
      LOADW  #2000h    ;upis adr. bloka podataka za PER1 i PER2
      STOREW MemAdr1   ;preko ACC u mem. lok. na adr. MemAdr1
      STOREW MemAdr2   ;i mem. lok. na adr. MemAdr2
  ;postavljanje "semafor"-a za periferije PER1 i PER2 na 1
      LOADB  #1        ;upis vrednosti 1 preko ACC u
      STOREB MemSem1   ;mem. lok. na adr. MemSem1 za PER1
      STOREB MemSem2   ;mem. lok. na adr. MemSem2 za PER2
  ;startovanje kontrolera periferija KPER1 i KPER2
      LOADB  #89h      ;upis režima rada i startovanja
                       ;(start=1,u/i=1,enable=1)preko ACC
      STOREB FF10h     ;u reg. Control KPER1
      STOREB FF20h     ;u reg. Control KPER2
  ;čekanje na završetak slanja bloka podataka iz memoriju u periferije PER1
  ;proverom vrednosti "semafor"-a za PER1
Wait1: LOADB  MemSem1  ;upis sadrž. mem.lok. sa adr. MemSem1 u ACC
       CMPB   #0       ;provera da li je "semafor" za PER1 postao 0
       JNZ    Wait1    ;povratak na Wait1 ukoliko "semafor" nije 0
  ;čekanje na završetak slanja bloka podataka iz memoriju u periferije PER2
  ;proverom vrednosti "semafor"-a za PER2
Wait2: LOADB  MemSem2  ;upis sadrž. mem.lok. sa adr. MemSem2 u ACC
       CMPB   #0       ;provera da li je "semafor" za PER2 postao 0
       JNZ    Wait2    ;povratak na Wait2 ukoliko "semafor" nije 0
. . .

;prekidna rutina periferije PER1
;slanje bloka podataka iz memorije u periferiju PER1 sa KPER1
PER1:  PUSHA            ;akumulator ACC na stek
  ;prenos podatka iz memorijske lokacije u registar Data KPER1
       LOADB  (MemAdr1) ;upis sadrž. mem. lok. sa adrese date sadrž. mem.
       STOREB FF12h     ;lok. na adr. MemAdr1 preko ACC u reg. Data KPER1
  ;inkrementiranje sadržaja memorijske lokacije MemAdr1
       LOADW  MemAdr1   ;upis sadrž. mem.lok. sa adr. MemAdr1 u ACC
       INCW             ;inkrementiranje sadrž. ACC
       STOREW MemAdr1   ;upis sadrž. ACC u mem.lok. na adr. MemAdr1
  ;dekrementiranje sadržaja memorijske lokacije MemCnt1
       LOADW  MemCnt1   ;upis sadrž. mem.lok. sa adr. MemCnt1 u ACC
       DECW             ;dekrementiranje sadrž. ACC
       STOREW MemCnt1   ;upis sadrž. ACC u mem.lok. na adr. MemCnt1
  ;provera da li je prenet poslednji podatak
       JNZ    Back1     ;prelaz na Back1 ukoliko nije poslednji podatak
  ;zaustavljanje kontrolera periferije KPER1
       LOADB  #0        ;upis režima zaustavljanja (start=0, u/i=0,
       STOREB FF10h     ;enable=0)preko ACC u reg. Control KPER1
  ;postavljanje "semafor"-a na 0 za PER1
                        ;upis vrednosti 0 preko ACC u
       STOREB MemSem1   ;mem. lok. na adr. MemSem1
  ;izlazak iz prekidne rutine
Back1: POPA             ;restauracija akumulatora ACC sa steka
       RTI              ;povratak iz prekidne rutine
  . . .
```

```
;prekidna rutina periferije PER2
;slanje bloka podataka iz memorije u periferiju PER2 sa KPER2
PER2:  PUSHA            ;akumulator ACC na stek
  ;prenos podatka iz memorijske lokacije u registar Data KPER2
       LOADB  (MemAdr2) ;upis sadrž. mem. lok. sa adrese date sadrž. mem.
       STOREB FF22h     ;lok. na adr. MemAdr2 preko ACC u reg. Data PER2
  ;inkrementiranje sadržaja memorijske lokacije MemAdr2
       LOADW  MemAdr2   ;upis sadrž. mem.lok. sa adr. MemAdr2 u ACC
       INCW             ;inkrementiranje sadrž. ACC
       STOREW MemAdr2   ;upis sadrž. ACC u mem.lok. na adr. MemAdr2
  ;dekrementiranje sadržaja memorijske lokacije MemCnt2
       LOADW  MemCnt2   ;upis sadrž. mem.lok. sa adr. MemCnt2 u ACC
       DECW             ;dekrementiranje sadrž. ACC
       STOREW MemCnt2   ;upis sadrž. ACC u mem.lok. na adr. MemCnt2
  ;provera da li je prenet poslednji podatak
       JNZ    Back2     ;prelaz na Back2 ukoliko nije poslednji podatak
  ;zaustavljanje kontrolera periferije KPER2
       LOADB  #0        ;upis režima zaustavljanja (start=0, u/i=0,
       STOREB FF20h     ;enable=0)preko ACC u reg. Control KPER2
  ;postavljanje "semafor"-a na 0 za PER2
                        ;upis vrednosti 0 preko ACC u
       STOREB MemSem2   ;mem. lok. na adr. MemSem2
  ;izlazak iz prekidne rutine
Back2: POPA             ;restauracija akumulatora ACC sa steka
       RTI              ;povratak iz prekidne rutine
  . . .
```

Слика 10 Програм

Програм се састоји из три дела. У првом делу се из периферије **PER0** техником програмираног улаза са испитивањем бита спремности *ready* статусног регистра *Status* **KPER0** уноси блок од 200h података и смешта у меморијске локације од адресе 2000h до адресе 21FFh. У другом делу се позива процесура `Obrada` која врши одређену обраду над унетим подацима у блоку података у меморијским локацијама од адресе 2000h до адресе 21FFh и резултат смешта у исте меморијске локације од адресе 2000h до адресе 21FFh. У трећем делу се у периферије **PER1** и **PER2** техником програмираног излаза са прекидом шаље блок од 200h података прочитаних из меморијских локација од адресе 2000h до адресе 21FFh. Програм је сличан програму из задатка 1.2. Разлика је само у трећем делу у коме се, за разлику од задатка 1.2 у коме се техником програмираног излаза са прекидом блок података шаље из меморије само у периферију **PER1**, техником програмираног излаза са прекидом блок података шаље из меморије упоредо у периферије **PER1** и **PER2**. Овај део програма је реализован на сличан начин као и први део програма у задатку 1.4

На почетку трећег дела главног програма иницијализују се преноси у обе периферије, постављају на 1 "semafor"-и за обе периферије и стартују контролери обе периферије. Прелазак на следећи део програма не сме да се дозволи док се не заврши слање блокова података у обе периферије. Чекање на завршетак слања блокова од по 100h података у обе периферије се реализује најпре у петљи са лабелом Wait1 за периферију **PER1** а затим и у петљи са лабелом Wait2 за периферију **PER2**, а само слање података из меморије у периферије **PER0** и **PER1** се реализују у прекидним рутинама периферија **PER1** и **PER2** на које се прелази када контролери периферија **KPER1** и **KPER2** генеришу прекиде. И овде као и у првом делу задатка 1.4 могу да се јаве два случаја у зависности од брзине периферија **PER1** и **PER2**. Први случај се јавља када је периферија **PER1** бржа од периферије **PER2**, па ће се слање података у периферију **PER1** завршити пре слања података у периферију **PER2** и прво "semafor" периферије **PER1** поставити на вредност 0 а затим и "semafor" периферије **PER2**

поставити на вредност 0. Због тога ће се из петље са лабелом Wait1 изаћи чим слање података у периферију **PER1** буде завршено и "semafor" периферије **PER1** буде постављен на вредност 0, али ће се затим у петљи са лабелом Wait2 чекати да буде завршено и слање података у периферију **PER2** и да "semafor" периферије **PER2** буде постављен на вредност 0, па затим прећи на следећи део програма. Други случај се јавља када је периферија **PER1** спорија од периферије **PER2**, па ће слање података у периферију **PER2** да буде завршено пре слања података у периферију **PER1** и прво "semafor" периферије **PER2** поставити на вредност 0 а затим и "semafor" периферије **PER1** поставити на вредност 0. Због тога ће се из петље са лабелом Wait1 изаћи чим слање података у периферију **PER1** буде завршено и "semafor" периферије **PER0** буде постављен на вредност 0, али се у петљи са лабелом Wait1 неће чекати, јер ће слање података у периферију **PER2** бити завршено и "semafor" периферије **PER2** постављен на вредност 0, па ће се одмах прећи на следећи део програма.

Дискусија:

1. Разматрања везана за иницијализацију и стартовање контролера периферија **KPER2** и **KPER2** у овом задатку су идентична као и разматрања везана за иницијализацију и стартовање контролера периферије **KPER1** у задатку 1.2 и дата су у задатку 1.2 у одељку **Дискусија** под **1**.

1.6 ЗАДАТАК

Дат је рачунарски систем који чине процесор, меморија и контролер периферије **KPER0** и DMA контролер **DMA0** са периферијом **PER0**, контролер периферије **KPER1** са периферијом **PER1** и контролер периферије **KPER2** са периферијом **PER2** повезани магистралом.

Процесор је са једноадресним форматом инструкција. Од програмски доступних регистара постоје 16 битни акумулатор ACC, указивач на врх стека SP, указивач на табелу (IV табела) са адресама прекидних рутина IVTP и програмска статусна реч PSW. Типови података који се користе су 16 битне целобројне величине са знаком и без знака. Приликом извршавања инструкције преноса у акумулатор и аритметичких, логичких и померачких инструкција постављају се индикатори N, Z, C и V програмске статусне речи PSW. Механизам прекида је векторисан. При прекиду хардверски се на стеку чувају програмски бројач PC и програмска статусна реч PSW. Бројеви улаза у IV табелу су фиксни за сваку периферију. Меморијски и улазно-излазни адресни простори су раздвојени.

Меморијски адресни простор је величине 2^{16} меморијских локација, при чему је ширина меморијских локација 16 битова.

Улазно-излазни адресни простор је величине 2^{16} регистара, при чему је ширина регистара 16 битова. Регистри контролера периферија **KPER0**, **KPER1** и **KPER2** и регистри DMA контролера **DMA0** се налазе у улазно-излазном адресном простору.

Контролери периферија **KPER0**, **KPER1** и **KPER2** имају управљачке регистре (*Control*), статусне регистре (*Status*) и регистре података (*Data*) и то редом на адресама FF00h, FF01h и FF02h за контролер периферије **KPER0**, FF10h, FF11h и FF12h за контролер периферије **KPER1** и FF20h, FF21h и FF22h за контролер периферије **KPER2**. У регистрима контролера периферија **KPER0**, **KPER1** и **KPER2** највиши бит је означен са 15, а најнижи са 0. У управљачким регистрима *Control* бит 15 је бит *u/i* којим се задаје смер преноса са периферијом (0—улаз, 1—излаз), бит 4 је бит *start* којим се покреће извршавање операције (0—заустављен, 1—стартован) и бит 0 је бит *enable* којим се дозвољава прекид (0—маскиран, 1—дозвољен). У статусним регистрима *Status* бит 15 је бит спремности *ready* (0—није спреман, 1—спреман).

DMA контролер **DMA0** има управљачки регистар (*Control*), статусни регистар (*Status*), регистар податка (*Data*), регистар величине блока података (*Count*), изворишни адресни регистар (*AdrSrc*) и одредишни адресни регистар (*AdrDst*) и то редом на адресама FF08h, FF09h, FF0Ah, FF0Bh, FF0Ch и FF0Dh, респективно. У регистрима DMA контролера **DMA0** највиши бит је означен са 15, а најнижи са 0. У управљачком регистру *Control* бит 15 је бит *u/i* којим се задаје смер преноса са периферијом (0—улаз, 1—излаз), бит 14 је бит *mem/per* којим се задаје пренос између делова меморије или између периферије и меморије (0—периферија/меморија, 1—меморија/меморија), бит 4 је бит *start* којим се покреће извршавање операције (0—заустављен, 1—стартован), бит 1 је бит *burst* којим се задаје режим преноса (0—циклус по циклус, 1—цео блок) и бит 0 је бит *enable* којим се дозвољава прекид (0—маскиран, 1—дозвољен), при чему је вредност бита 15 битна само уколико је вредношћу 0 бита 14 задат пренос између периферије и меморије. У статусном регистру *Status* бит 15 је бит завршетка преноса *end* (0—пренос у току, 1—пренос завршен).

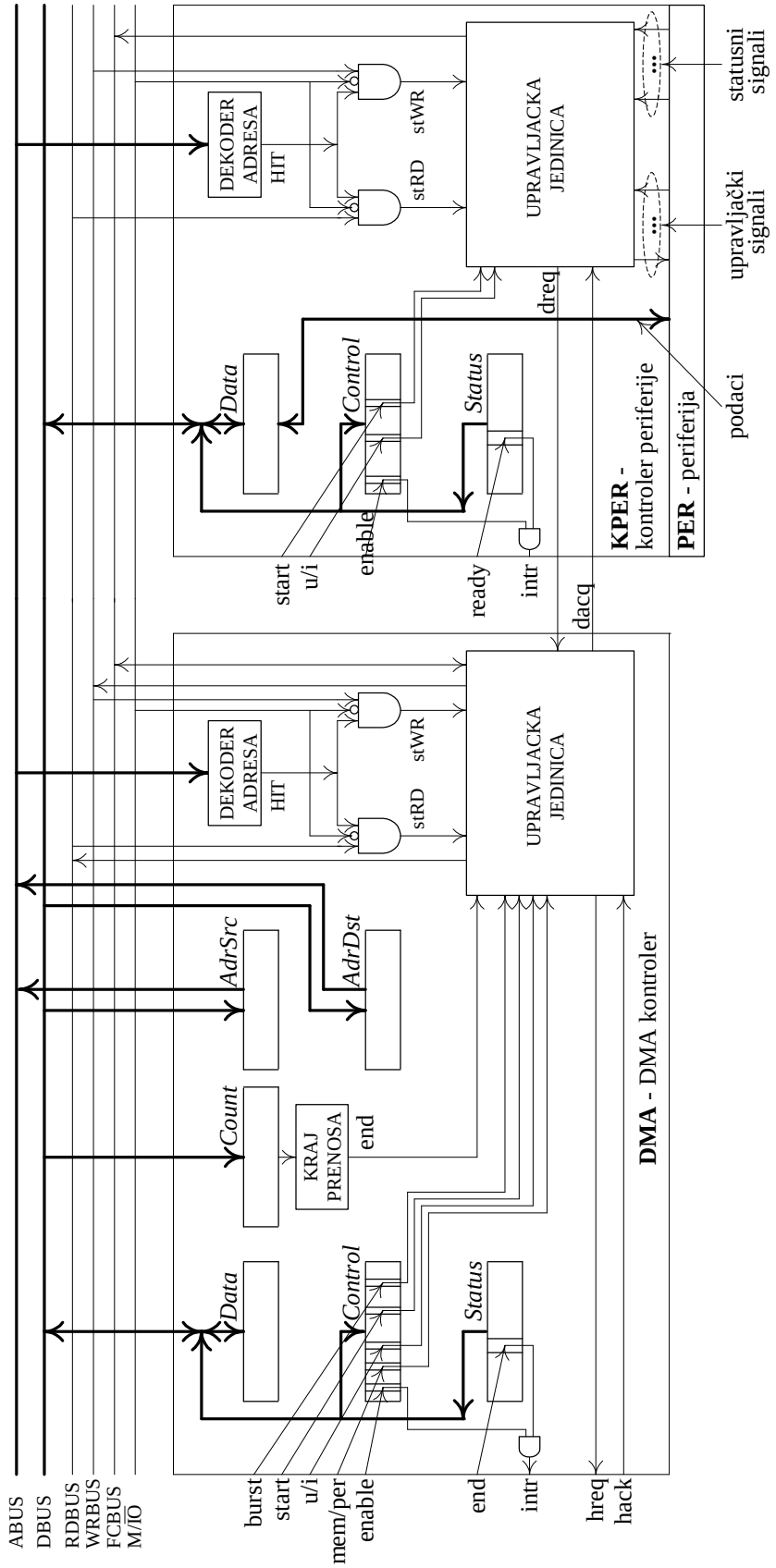
Адресне линије и линије података магистрале рачунара су широке по 16 битова.

Написати главни програм и одговарајућу прекидну рутину којима се упоредо учитавају низ A(i), где је i = 0, …, FFh, из **PER0** у меморијски блок који почиње од адресе A000h и низ B(i), где је i = 0, …, FFh, из **PER2** у меморијски блок који почиње од адресе B000h, затим формира низ C(i) = A(i) + B(i), где је i = 0, …, FFh, у меморијском

блоку који почиње од адресе C000h и резултујући низ C(i), где је i = 0, …, FFh, из меморијског блока који почиње од адресе C000h шаље у **PER1**. Улаз из периферије **PER0** реализовати са контролером периферије **KPER0** и DMA контролером **DMA0** у режиму преноса циклус по циклус, улаз из периферије **PER2** реализовати са контролером периферије **KPER2** техником програмираног улаза са испитивањем спремности, а излаз у периферију **PER1** реализовати са контролером периферије **KPER1** техником програмираног излаза са прекидом.

Решење:

Периферија **PER0** се повезује на магистралу рачунарског система са контролером периферије **KPER0** и DMA контролера **DMA0** на начин приказан на слици 11, док се периферије **PER1** и **PER2** повезују на магистралу рачунарског система са контролерима периферија **KPER1** и **KPER2** на начин приказан на слици 1.



Слика 11 Повезивање периферије **PER** са контролером периферије **KPER** и DMA контролером **DMA**

Адресе и структура регистара контролера **KPER0**, **DMA0**, **KPER1** и **KPER2** су дати на сликама 12 и 13, респективно.

KPER0 - Kontroler periferije **PER0**

Registar	Control	Status	Data
Adresa	FF00h	FF01h	FF02h

DMA0 –DMA kontroler

Registar	Control	Status	Data	Count	AdrSrc	AdrDst
Adresa	FF08h	FF09h	FF0Ah	FF0Bh	FF0Ch	FF0Dh

KPER1 - Kontroler periferije **PER1**

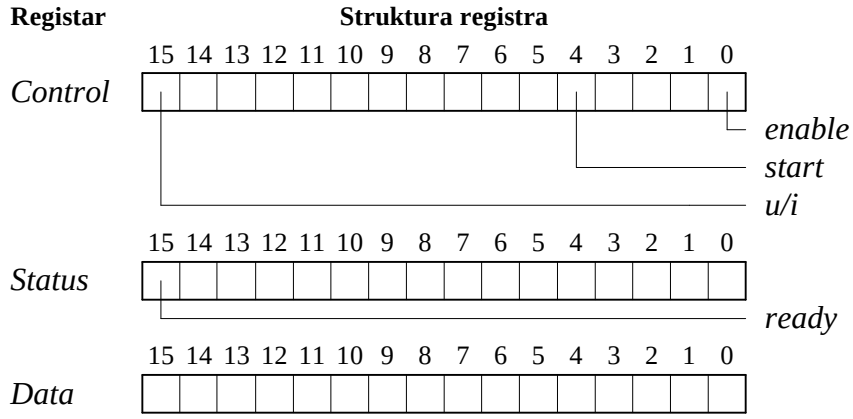
Registar	Control	Status	Data
Adresa	FF10h	FF11h	FF12h

KPER2 - Kontroler periferije **PER2**

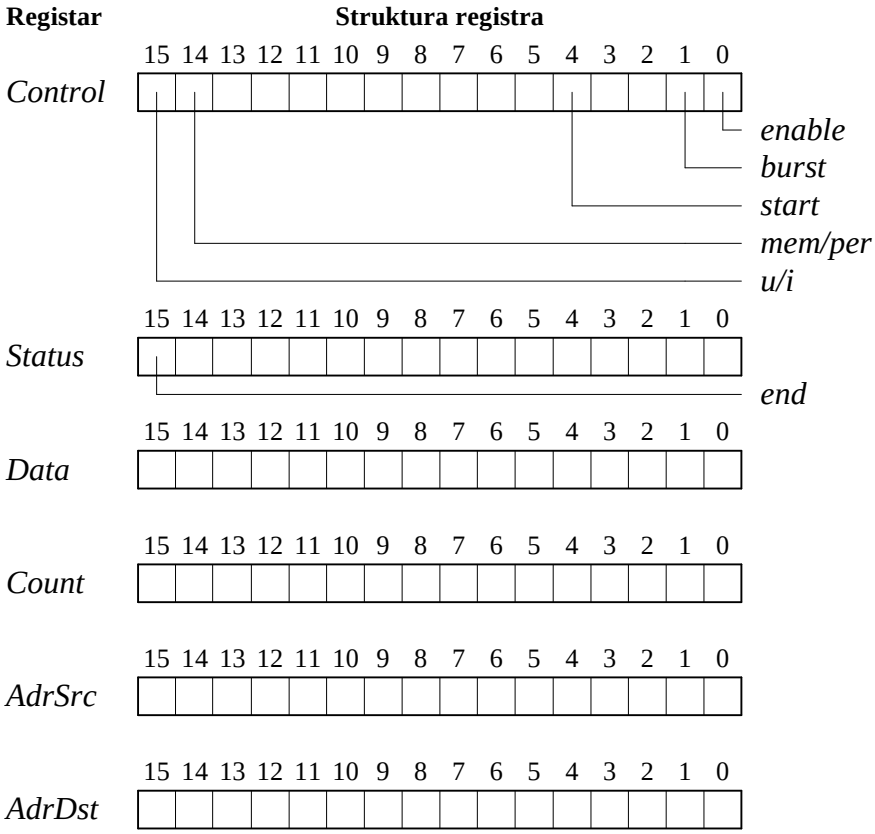
Registar	Control	Status	Data
Adresa	FF20h	FF21h	FF22h

Слика 12 Адресе регистара контролера

Registri kontrolera periferija **KPER**



Registri DMA kontrolera **DMA**



Слика 13 Структура регистара контролера

Тражени програм је приказан на слици 14.

```
;glavni program
;unos blokova podataka iz periferije PER0 u memoriju sa KPER0 i DMA0 i
;iz periferije PER2 u memoriju sa KPER2 ispitivanjem bita spremnosti ready
  ;inicijalizacija prenosa u mem iz PER0 sa KPER0 i DMA0 i iz PER2 sa KPER2
    ;veličine blokova podataka
      LOAD   #100h     ;upis vel bloka podat za PER0 i PER2 preko ACC u
      OUT    FF0Bh     ;reg Count DMA0 i
      STORE  MemCnt2   ;mem. lok. na adr. MemCnt2
    ;početne adrese blokova podataka
      LOAD   #A000h    ;upis adr. bloka podataka za PER0 preko ACC u
      OUT    FF0Dh     ;reg AdrDst DMA0
      LOAD   #B000h    ;upis adr. bloka podataka za PER2 preko ACC u
      STORE  MemAdr2   ;mem. lok. na adr. MemAdr2
  ;postavljanje "semafor"-a za DMA0 na 1
      LOAD   #1h       ;upis vrednosti 1 preko ACC u
      STORE  MemSem0   ;mem. lok. na adr. MemSem0
  ;startovanje kontrolera periferije KPER0
      LOAD   #0010h    ;upis režima rada i startovanja (u/i=0,
      OUT    FF00h     ;start=1,enable=0)preko ACC u reg. Control KPER0
  ;startovanje kontrolera DMA0
      LOAD   #0011h    ;upis režima rada i start (u/i=0,mem/per=0,start=1,
      OUT    FF08h     ;burst=0,enable=1)preko ACC u reg. Control DMA0
  ;startovanje kontrolera periferija KPER2
      LOAD   #0010h    ;upis režima rada i startovanja (u/i=0,
      OUT    FF20h     ;start=1,enable=0)preko ACC u reg. Control KPER2
  ;unos blokova podataka
    ;provera da li podatak može da se prenese iz registra Data KPER2
Loop2: IN      FF21h     ;upis sadrž. reg. Status KPER2 u ACC
       AND     #8000h    ;ispitivanje bita ready
       JZ      Loop2     ;povratak na Loop2 ukoliko je ready 0
    ;prenos podatka iz registra Data KPER2 u memorijsku lokaciju
      IN      FF22h     ;upis sadrž. reg. Data KPER2 preko ACC u mem. lok.
      STORE   (MemAdr2) ;na adresi datoj sadrž. mem. lok. na adr. MemAdr2
    ;inkrementiranje sadržaja memorijske lokacije MemAdr2
      INC     MemAdr2   ;inkrement sadrž. mem.lok. sa adr. MemAdr2
    ;dekrementiranje sadržaja memorijske lokacije MemCnt2
      DEC     MemCnt2   ;dekrement sadrž. mem.lok. sa adr. MemCnt2
    ;provera da li je unet poslednji podatak iz PER2
      JNZ     Loop2     ;povratak na Loop2 ukoliko nije poslednji podatak
  ;zaustavljanje kontrolera periferije KPER2
      LOAD   #0        ;upis režima zaustavljanja (u/i=0,
      OUT    FF20h     ;start=0, enable=0) preko ACC u reg. Control KPER2
  ;čekanje na završetak unosa bloka podataka iz periferije PER0 u memoriju
  ;sa KPER0 i DMA0 proverom vrednosti "semafor"-a za DMA0
Wait0: LOAD   MemSem0   ;upis sadrž. mem.lok. sa adr. MemSem0 u ACC
       CMP    #0        ;ispitivanje da li je "semafor" postao 0
       JNZ    Wait0     ;povratak na Wait0 ukoliko "semafor" nije 0
;obrada
       JSR    Obrada    ;obrada blokova od po 100h podataka u memorijskim
  ;lokacijama od adrese A000h do adrese A0FFh i od adrese B000h do adrese
  ;B0FFh i upis rezultata u memorijske lokacije od adrese C000h do adrese
  ;C0FFh
```

```
;slanje bloka podataka iz memorije u periferiju PER1 sa KPER1 sa
;generisanjem prekida
  ;inicijalizacija prenosa iz memorije u periferiju PER1 sa KPER1
    ;veličina bloka podataka
      LOAD   #100h     ;upis veličine bloka podataka za PER1 preko ACC u
      STORE  MemCnt1   ;mem. lok. na adr. MemCnt1
    ;početna adresa bloka podataka
      LOAD   #C000h    ;upis adr. bloka podataka za PER1 preko ACC u
      STORE  MemAdr1   ;mem. lok. na adr. MemAdr1
  ;postavljanje "semafor"-a za PER1 na 1
      LOAD   #1h       ;upis vrednosti 1 preko ACC u
      STORE  MemSem1   ;mem. lok. na adr. MemSem1
  ;startovanje kontrolera periferije KPER1
      LOAD   #8011h    ;upis režima rada i startovanja (u/i=1,
      OUT    FF10h     ;start=1,enable=1)preko ACC u reg. Control KPER1
  ;čekanje na završetak slanja bloka podataka iz memorije u periferiju
  ;PER1 proverom vrednosti "semafor"-a za PER1
Wait1: LOAD   MemSem1   ;upis sadrž. mem.lok. sa adr. MemSem1 u ACC
       CMP    #0        ;ispitivanje da li je "semafor" postao 0
       JNZ    Wait1     ;povratak na Wait1 ukoliko "semafor" nije 0
. . .

;prekidna rutina za DMA0
DMA0:  PUSHA            ;akumulator ACC na stek
  ;zaustavljanje kontrolera periferije KPER0
       LOAD   #0        ;upis režima zaustavljanja (u/i=0, start=0,
       OUT FF00h        ;enable=0)preko ACC u reg. Control KPER0
  ;zaustavljanje kontrolera DMA0
                        ;upis režima zaustavljanja(u/i=0,mem/per=0,start=0,
       OUT FF08h        ;burst=0,enable=0)preko ACC u reg. Control DMA0
  ;postavljanje "semafor"-a za DMA0 na 0
                        ;upis vrednosti 0 preko ACC u
       STORE  MemSem0   ;mem. lok. na adr. MemSem0
  ;izlazak iz prekidne rutine
       POPA             ;restauracija akumulatora ACC sa steka
       RTI              ;povratak iz prekidne rutine
```

```
;Procedura Obrada u kojoj se izračunava C(i)=A(i)+B(i), gde je i=0,…,FFh
;Elementi niza A(i), gde je i=0,…,FFh, su na adresama A000h do A0FFh,
;elementi niza B(i), gde je i=0,…,FFh, su na adresama B000h do B0FFh, a
;elementi niza sume C(i), gde je i=0,…,FFh, se smešt na adr C000h do C0FFh
Obrada:  LOAD   #100h     ;upis vel. blok. podat. za A(i), B(i) i C(i)
         STORE  MemCnt1   ;u ACC pa u mem. lok. na adr. MemCnt1
         LOAD   #A000h    ;upis adr. bloka podataka A(i)preko ACC u
         STORE  MemAdr0   ;mem. lok. na adr. MemAdr0
         LOAD   #B000h    ;upis adr. bloka podataka B(i)preko ACC u
         STORE  MemAdr2   ;mem. lok. na adr. MemAdr2
         LOAD   #C000h    ;upis adr. bloka podataka C(i)preko ACC u
         STORE  MemAdr1   ;mem. lok. na adr. MemAdr1

Loop:    LOAD   (MemAdr0) ;upis A(i) iz mem lok sa adrese date sadr
                          ;mem lok sa adrese MemAdr0 u ACC
         ADD    (MemAdr2) ;suma ACC i B(i) iz mem lok sa adrese date sadr
                          ;mem lok sa adrese MemAdr2 u ACC
         STORE  (MemAdr1) ;sadržaj ACC u C(i) u mem lok na adr datoj sadr
                          ;mem lok na adresi MemAdr1
         INC    MemAdr0   ;inkrementiranje sadr mem lok MemAdr0
         INC    MemAdr2   ;inkrementiranje sadr mem lok MemAdr2
         INC    MemAdr1   ;inkrementiranje sadr mem lok MemAdr1
         DEC    MemCnt1   ;dekrementiranje sadr mem lok MemCnt1
  ;provera da li je poslednja suma
         JNZ    Loop      ;povratak na Loop ukoliko nije poslednja suma
         RTS              ;povratak u glavni program

;prekidna rutina periferije PER1
;slanje bloka podataka iz memorije u periferiju PER1 sa KPER1
PER1:  PUSHA             ;akumulator ACC na stek
  ;prenos podatka iz memorijske lokacije u registar Data KPER1
       LOAD   (MemAdr1) ;upis sadrž. mem. lok. sa adr. date sadrž. mem.
       OUT    FF12h     ;lok. sa adr. MemAdr1 preko ACC u reg. Data KPER1
  ;inkrementiranje sadržaja memorijske lokacije MemAdr1
       INC    MemAdr1   ;inkr sadrž. mem.lok. sa adr. MemAdr1
  ;dekrementiranje sadržaja memorijske lokacije MemCnt1
       DEC    MemCnt    ;dekr sadrž. mem.lok. sa adr. MemCnt1
  ;provera da li je prenet poslednji podatak
       JNZ    Back1     ;prelaz na Back1 ukoliko nije poslednji podatak
  ;zaustavljanje kontrolera periferije PER1
       LOAD   #0        ;upis režima zaustavljanja (start=0, u/i=0,
       OUT    FF10h     ;enable=0)preko ACC u reg. Control KPER1
  ;postavljanje "semafor"-a za PER1 na 0
                         ;upis vrednosti 0 preko ACC u
       STORE  MemSem1   ;mem. lok. na adr. MemSem1
  ;izlazak iz prekidne rutine
Back1: POPA              ;restauracija akumulatora ACC sa steka
       RTI               ;povratak iz prekidne rutine
. . .
```

Слика 14 Програм

За унос података из периферије **PER0** у меморију се у овом задатку користи не само контролер периферије **KPER0**, већ и DMA контролер **DMA0**. Контролер периферије **KPER0** се у овом задатку иницијализује и стартује за унос блока података из периферије у меморију на сличан начин као и у задацима 1.1 и 1.3 и у овом задатку функционише на сличан начин као и у задацима 1.1 и 1.3. Унос блока података из периферије **PER0** у меморију започиње тако што се извршавањем одговарајућег програма у управљачки регистар *Control* **KPER0** упише садржај којим се контролер **KPER0** стартује (*start*=1) за рад у режиму улаза (*u/i*=0) и без генерисања прекида (*enable*=0). Том приликом контролер **KPER0** поставља на вредност 0 бит спремности *ready* статусног регистра *Status* **KPER0** и започиње читање податка из периферије **PER0** и пренос у регистар података *Data* **KPER0**. Контролер **KPER0**, приликом уписа података из периферије **PER0** у регистар *Data* **KPER0**, поставља на вредност 1 бит *ready* статусног регистра *Status* **KPER0** и вредношћу 1 сигнала dreq који иде из контролера периферије **KPER0** у DMA контролер **DMA0** тражи од DMA контролера **DMA0** да податак пренесе из регистра *Data* **KPER0** у меморијске локацију. Сам пренос података из регистра *Data* **KPER0** у меморијску локацију се за разлику од задатака 1.1 и 1.3 не реализује програмским путем неком од техника програмираног улаза са испитивањем спремности (задатак 1.1) или са прекидом (задатак 1.3), већ се реализује хардверски са DMA контролером **DMA0** сигналом dack који иде из DMA контролера **DMA0** у контролер периферије **KPER0**. Том приликом контролер **KPER0** поставља на вредност 0 бит *ready* статусног регистра *Status* **KPER0** и започиње читање следећег податка из периферије **PER0** и пренос у регистар података *Data* **KPER0**. Контролер **KPER0**, када заврши пренос следећег податка из **PER0** у регистар података *Data* **KPER0**, поново поставља на вредност 1 бит спремности *ready* статусног регистра *Status* **KPER2** и сигнал dreq.

DMA контролер **DMA0** хардверски реализује пренос података из регистра *Data* **KPER0** у меморијску локацију, генерише адресе одредишних меморијских локација у које треба да уписује један за другим податке које контролер периферије **KPER0** чита из периферије **PER0** и смешта у регистар *Data* **KPER0** и води евиденцију колико још података треба пренети из периферије **PER0** у меморију. Због тога DMA контролер **DMA0** има одредишни адресни регистар *AdrDst* и регистар величине блока података *Count*. За разлику од задатака 1.1 и 1.3 код којих се, у оквиру иницијализације преноса блока података из периферије **PER0** у меморију, почетна адреса меморијске локације од које треба унети блок података и величина блока података уписују у меморијске локације MemAdr0 и MemCnt0, у овом задатку се те вредности уписују у регистре *AdrDst* и *Count* DMA контролера **DMA0**. DMA контролер **DMA0** мора после сваког пренетог податка из регистра *Data* **KPER0** у меморијску локацију да изврши инкрементирање садржаја регистра *AdrDst* **DMA0** и декрементирање садржаја регистра *Count* **DMA0** и изврши проверу да ли је садржај регистра *Count* **DMA0** декрементирањем постао 0. Уколико садржај регистра *Count* **DMA0** није нула, DMA контролер **DMA0** треба да продужи са преносом података из периферије **PER0** у меморију, док у супротном случају треба да завршити са преносом података из периферије **PER0** у меморију и у бит *end* статусног регистра *Status* **DMA0** упише вредност 1.

Унос блока података из периферије **PER0** у меморију са **KPER0** и **DMA0** започиње тако што се извршавањем одговарајућег програма најпре "semafor" за **DMA0** постави на 1, затим контролер периферије **KPER0** стартује на начин објашњен у претходном параграфу, и на крају у управљачки регистар *Control* **DMA0** упише садржај којим се DMA контролер **DMA0** стартује (*start*=1) за рад у режиму улаза (*u/i*=0, *mem/per*=0) и то циклус по циклус (*burst*=0) са генерисања прекида (*enable*=1) на начин објашњен у

даљем тексту. Унос блока података из периферије **PER0** у меморију одвија се комплетно хардверски тако што сваки податак из блока података најпре контролер периферије **KPER0** преноси из периферије **PER0** у регистар података *Data* **KPER0** а затим DMA контролер **DMA0** преноси податак из регистра података *Data* **KPER0** у меморијску локацију. Ово се понавља све док садржај регистра *Count* **DMA0** декрементирањем не постане 0, када DMA контролер **DMA0** уписује вредност 1 у бит *end* статусног регистра *Status* **DMA0** и генерише прекид. У прекидној рутини DMA контролера **DMA0** се извршавањем одговарајућег програма најпре "semafor" за **DMA0** постави на 0 а затим у управљачке регистре *Control* **KPER0** и **DMA0** упише садржај којим се контролер периферије **KPER0** и DMA контролер **DMA0** зауставаљају (*start*=0).

Пренос једног податка из периферије **PER0** у меморијску локацију контролери **KPER0** и **DMA0** реализују на начин објашњен у даљем тексту. По стартовању контролера **KPER0** и **DMA0**, контролер периферије **KPER0** креће са читањем податка из периферије **PER0**, док DMA контролер **DMA0** чека да контролер периферије **KPER0** пребаци податак из периферије **PER0** у регистар података *Data* **KPER0** и постављањем сигнала dreq на вредност 1 затражи од **DMA0** да податак пренесе из регистра података *Data* **KPER0** у меморијску локацију. Када сигнал dreq постане 1, DMA контролер **DMA0** креће са пребацивањем податка из регистра података *Data* **KPER0** у меморијску локацију, док контролер периферије **KPER0** чека да пребацивање података буде реализовано. DMA контролер **DMA0** најпре поставља на 1 сигнал hreq којим, у зависности од тога како је реализована арбитрација, од процесора или арбитратора магистрале тражи дозволу за коришћење магистрале. Тек када вредношћу 1 сигнала hack добије дозволу за коришћење магистрале, DMA контролер **DMA0** креће са реализацијом циклуса уписа. Том приликом **DMA0** најпре отвара бафере са три стања и на адресне линије магистрале ABUS пушта садржај одредишног адресног регистра *AdrDst* **DMA0**, затим вредношћу 1 сигнала dack, који иде из DMA контролера **DMA0** у контролер периферије **KPER0**, отвара у контролеру периферије **KPER0** бафере са три стања и на линије података магистрале DBUS пушта садржај регистра података *Data* **KPER0** и на крају стартује операцију уписа постављањем управљачког сигнала уписа WRBUS на вредност 1. Пошто је узето да је магистрала асинхрона, DMA контролер **DMA0** и контролер периферије **KPER0** држе садржај на адресним линијама магистрале ABUS и линијама података магистрале DBUS и управљачки сигнал уписа WRBUS на вредности 1 све време док упис у меморију траје. По завршетку уписа најпре меморија поставља управљачки сигнал завршетка циклуса на магистрали FCBUS на вредност 1, па затим DMA контролер **DMA0** поставља сигнал WRBUS на вредност 0 и на крају меморија поставља сигнал FCBUS на вредност 0. Тиме је циклус уписа завршен, па **DMA0** затвара бафере са три стања и адресне линије магистрале ABUS ставља у стање високе импедансе и вредношћу 0 сигнала dack у **KPER0** затвара бафере са три стања и линије података магистрале DBUS ставља у стање високе импедансе. Вредност 0 сигнала dack је индикација контролеру периферије **KPER0** да је завршен пренос садржаја регистра *Data* **KPER0** у меморијску локацију, па контролер **KPER0** поставља сигнал dreq на вредност 0. Поред тога контролер периферије **KPER0** поставља на вредност 0 бит *ready* статусног регистра *Status* **KPER0** и започиње читање следећег податка из периферије **PER0** и пренос у регистар података *Data* **KPER0**. Паралелно са тим DMA контролер **DMA0** постављањем на сигнала hreq на вредност 0 укида захтев за коришћење магистрале и као одговор процесор или арбитратор магистрале му вредношћу 0 сигнала hack укидају дозволу за коришћење магистрале. Поред тога, DMA контролер **DMA0** инкрементира садржај регистра *AdrDst* **DMA0** и декрементира садржај регистра *Count* **DMA0** и врши проверу да ли је садржај регистра *Count* **DMA0** декрементирањем постао 0. Уколико садржај регистра *Count* **DMA0** није нула, DMA

контролер **DMA0** прелази на чекање да контролер периферије **KPER0** упише следећи податак из периферије **PER0** у регистар *Data* **KPER0** и постави на вредност 1 бит *ready* статусног регистра *Status* **KPER0** и вредношћу 1 сигнала dreq затражи од DMA контролера **DMA0** да податак пренесе из регистра *Data* **KPER0 у** меморијске локацију. Међутим, уколико је садржај регистра *Count* **DMA0** нула, DMA контролер **DMA0** уписује вредност 1 у бит *end* статусног регистра *Status* **DMA0** и генерише прекид. У прекидној рутини DMA контролера **DMA0** се извршавањем одговарајућег програма најпре "semafor" за **DMA0** постави на 0 а затим у управљачке регистре *Control* **KPER0** и **DMA0** упише садржај којим се контролер периферије **KPER0** и DMA контролер **DMA0** заустављају (*start*=0).

Програм се састоји из три дела. У првом делу се из периферија **PER0** и **PER2** упоредо уносе два блока од по 100h података и смештају у меморијске локације од адресе A000h до адресе A0FFh за периферију **PER0** и од адресе B000h до адресе B0FFh за периферију **PER2**. Унос из периферије **PER0** се реализује са контролером периферије **KPER0** и DMA контролером **DMA0** у режиму преноса циклус по циклус, док се унос из периферије **PER2** реализује са контролером периферије **KPER2** техником програмираног улаза са испитивањем спремности. У другом делу се позива процесура `Obrada` која врши одређену обраду над унетим подацима у блоковима од по 100h података у меморијским локацијама од адресе A000h до адресе A0FFh и од адресе B000h до адресе B0FFh и резултат који представља блок од 100h података смешта у меморијске локације од адресе C000h до адресе C0FFh. У трећем делу се у периферију **PER1** техником програмираног излаза са прекидом шаље блок од 100h података прочитаних из меморијских локација од адресе C000h до адресе C0FFh.

Први део програма у коме се из периферија **PER0** и **PER2** у меморију упоредо уносе два блока података има два дела и то један део који се извршава у главном програму и други део који се извршава у прекидној рутини за **DMA0**. У оквиру дела програма који се извршава у главном програму врши се иницијализација преноса из периферија **PER0** и **PER2**, постављање "semafor"-а за **DMA0** на 1, стартовање контролера периферије **KPER0**, DMA контролера **DMA0** и контролера периферије **KPER2**, унос блока података из периферије **PER2**, заустављање контролера периферије **KPER2** и чекање на завршетак уноса блока података из периферије **PER0** у меморију. У оквиру дела програма који се извршава у прекидној рутини за **DMA0** врши се заустављање контролера периферије **KPER0** и DMA контролера **DMA0** и постављање "semafor"-а за **DMA0** на 0.

У оквиру иницијализације преноса из периферија **PER0** и **PER2** најпре се у регистар *Count* **DMA0** и меморијску локацију MemCnt2 уписује вредност 100h која представља величину блокова података које треба пренети, а затим у регистар *AdrDst* **DMA0** и меморијску локацију MemAdr2 уписују вредности A000h и B000h које представљају почетне адресе делова меморије у које треба пренети блокове података из периферија **PER0** и **PER2**, респективно. У оквиру постављања "semafor"-а за **DMA0** на 1, у меморијску локацију MemSem0 се уписује вредност 1 која служи као индикација да је унос података из периферије **PER0** у току и да се не сме прећи на извршавање програма у коме се користе подаци из блока меморије у који се уносе подаци из периферије **PER0**. У оквиру стартовања контролера периферије **KPER0,** DMA контролера **DMA0** и контролера периферије **KPER2** у управљачки регистар *Control* **KPER0** се уписује вредност 0010h чиме се контролер периферије **KPER0** стартује (*start*=1) за рад у режиму улаза (*u/i*=0) без генерисањем прекида (*enable*=0), у управљачки регистар *Control* **DMA0** се уписује вредност 0011h чиме се DMA контролер **DMA0** стартује (*start*=1) за рад у режиму улаза (*u/i*=0, *mem/per*=0) и то циклус по циклус (*burst*=0) са генерисањем прекида (*enable*=1) и у управљачки регистар *Control* **KPER2** се уписује

вредност 0010h чиме се контролер периферије **KPER2** стартује (*start*=1) за рад у режиму улаза (*u/i*=0) без генерисањем прекида (*enable*=0). Од овог тренутка процесор и контролер периферије **KPER0,** DMA контролер **DMA0** и контролер периферије **KPER2** почињу да раде паралелно. Процесор извршава програм којим најпре у петљи Loop2 реализује унос блока података из периферије **PER2** са контролером периферије **KPER2** техником програмираног улаза са испитивањем спремности на начин објашњен у задатку 1.1 и затим уписивањем вредности 0000h у управљачки регистар *Control* **KPER2** зауставља (*start*=0) контролер периферије **KPER2**, док DMA контролер **DMA0** и контролер периферије **KPER0** хардверски реализују унос блока података из периферије **PER0**.

Прелазак на други део програма у коме се реализује обрада унетих блокова података не сме да се дозволи док се не унесу блокови података из обе периферије. Због тога се по завршетку уноса блока података из периферије **PER2** у петљи са лабелом Wait0 чека завршетак уноса блока података из периферије **PER0.** Том приликом могу да се јаве два случаја у зависности од брзине периферија **PER0** и **PER2**. Први случај се јавља када је периферија **PER2** бржа од периферије **PER0**, па ће се у главном програму завршити унос података из периферије **PER2** пре завршетка уноса података из периферије **PER0**. Тада ће се у главном програму доћи на петљу са лабелом Wait0 и у њој утврдити да "semafor" за **DMA0** још увек има вредност 1. Због тога ће се најпре у петљи са лабелом Wait0 чекати да **DMA0** по завршетку уноса блока података из периферије **PER0** генерише прекид и изазове прелазак на прекидну рутину за **DMA0** у којој ће у "semafor" за **DMA0** бити уписана вредност 0, а затим ће се по повратку из прекидне рутине на петљу са лабелом Wait0 утврдити да "semafor" за **DMA0** има вредност 0, па ће се изаћи из петље са лабелом Wait0 и прећи на други део програма у коме се реализује обрада унетих блокова података. Други случај се јавља када је периферија **PER2** спорија од периферије **PER0**, па ће се унос података из периферије **PER0** завршити пре од уноса података из периферије **PER2** и прво преласком у прекидну рутину за **DMA0** "semafor" за **DMA0** поставити на вредност 0, па тек онда у главном програму доћи на петљу са лабелом Wait0. Због тога ће се по завршетку уноса блока података из периферије **PER2** и преласком на петљу са лабелом Wait0 утврдити да "semafor" за **DMA0** има вредност 0, па се неће чекати у петљи, већ ће се одмах прећи на други део програма у коме се реализује обрада унетих блокова података.

Други део у коме се врши обрада се реализује позивом на процесуру `Obrada`. У процедури `Obrada` се најпре меморијске локације MemCnt1, MemAdr0, MemAdr2 и MemAdr1 иницијализују вредностима 100h, A000h, B000h и C000h које представљају величину блокова података над којима треба извршити обраду и почетне адресе два блока података над којима треба извршити обраду и блока у који треба уписати резултат, респективно. Потом се у петљи Loop из меморије читају парови података из меморијских локација од адресе A000h до адресе A0FFh и од адресе B000h до адресе B0FFh и сума уписује у меморијске локације од адресе C000h до адресе C0FFh, инкрементирају садржаји меморијских локација MemAdr0, MemAdr2 и MemAdr1 и декрементира садржај меморијске локације MemCnt1 и на крају у зависности од тога да ли је садржај меморијске локације MemCnt1 после декрементирања још увек различит од нуле или је постао нула или остаје у петљи Loop или излази из петље Loop, респективно. На крају се враћа у главни програм.

Трећи део у коме се у периферију **PER1** техником програмираног излаза са прекидом шаље блок података се реализује на сличан начин као у задатку 1.2

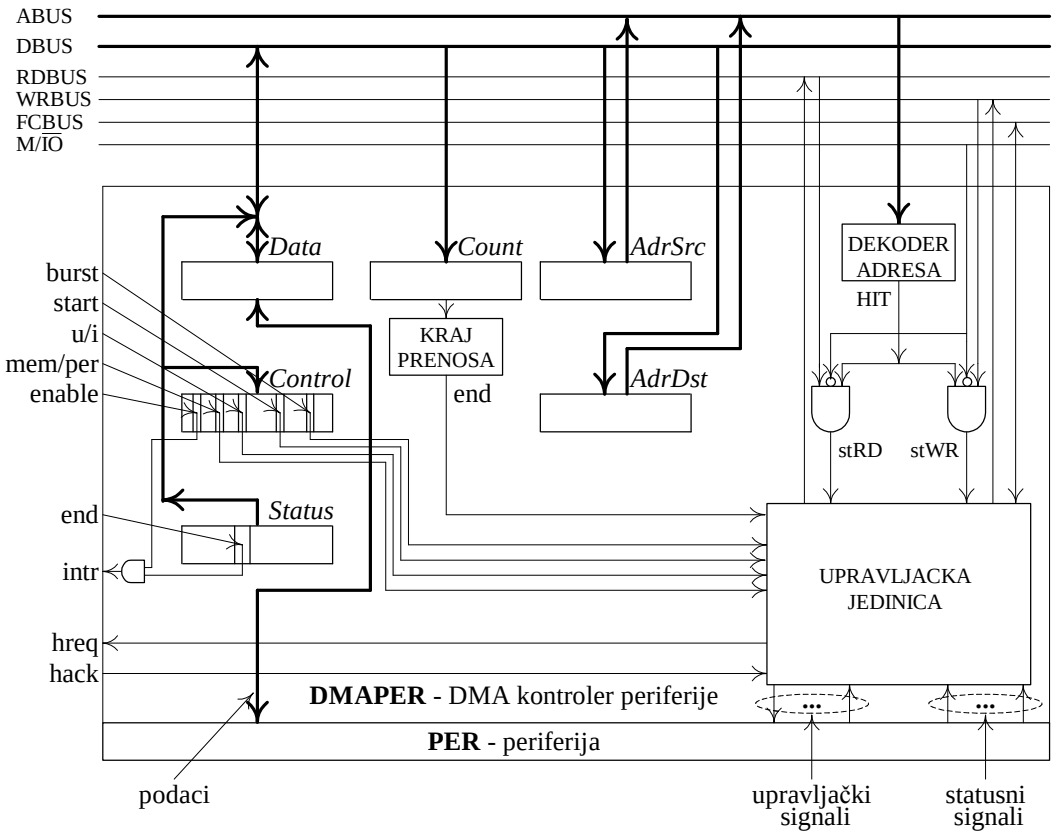
Дискусија:

1. DMA контролер **DMA0** може да ради и у режиму преноса цео блок. Разлика у односу на објашњени режим преноса циклус по циклус је само у томе да се сигнал

захтева за коришћење магистрале hreq држи на вредности 1 све време док траје пренос блока података, па стога и сигнал дозволе коришћења магистрале hack има вредност 1 све време док траје пренос блока података. Сигнал захтева за коришћење магистрале hreq и сигнал дозволе коришћења магистрале hack се постављају на вредности 0 тек када се пренесе задњи податак из блока података. Ефекат режима преноса цео блок на рачунарски систем је у томе да нико не може да користи магистралу док DMA контролер **DMA0** не заврши пренос целог блока података. Структура програма је, међутим, иста без обзира на то да ли DMA контролер **DMA0** ради у режиму циклус по циклус или у режиму цео блок.

2. Функционисање контролера периферије **KPER0** и DMA контролера **DMA0** као и коришћење изворишног адресног регистра *AdrSrc* при преносу блока података из меморије у периферију објашњено је у задатку 1.7, док је функционисање DMA контролера **DMA0** као и коришћење изворишног и одредишног адресног регистра *AdrSrc* и *AdrDst* при преносу блока података из меморије у меморију објашњено у задатку 1.8.

3. Рачунарски систем се може и тако конфигурисати да се периферија **PER0** повеже на магистралу рачунарског система са DMA контролером периферије **DMAPER0** (слика 15), а да се за периферије **PER1** и **PER2** задржи повезивање на магистралу рачунарског система са контролерима периферија **KPER1** и **KPER2** (слика 1).



Слика 15 Повезивање периферије **PER** преко DMA контролера периферије **DMAPER**

Тражени програм је приказан на слици 16. Овај програм је веома сличан програму са слике 14. Добијен је тако што су у програму са слике 14 избачене инструкције главног програма

```
;startovanje kontrolera periferije KPER0
     LOAD   #0010h    ;upis režima rada i startovanja (u/i=0,
     OUT    FF00h     ;start=1,enable=0)preko ACC u reg. Control KPER0
```

које се односе на стартовање контролера периферије **KPER0** и инструкције прекидне рутине **DMA0**

```
  ;zaustavljanje kontrolera periferije KPER0
       LOAD   #0        ;upis režima zaustavljanja (u/i=0,
       OUT FF00h        ;start=0,enable=0)preko ACC u reg. Control KPER0
```

које се односе на заустављање контролера периферије **KPER0** и уместо ознака DMA контролера **DMA0** стављене су ознаке DMA контролера периферије **DMAPER0**. Ради једноставнијег писања програма узето је да регистри DMA контролера периферије **DMAPER0** (слика 15) имају исте адресе и структуру као и регистри DMA контролера **DMA0** (слика 11). Адресе и структура регистара су дати на сликама 12 и 13.

```
 ;glavni program
;unos blokova podataka iz periferije PER0 u memoriju sa DMAPER0 i
;iz periferije PER2 u memoriju sa KPER2 ispitivanjem bita spremnosti ready
  ;inicijalizacija prenosa u mem iz PER0 sa DMAPER0 i iz PER2 sa KPER2
    ;veličine blokova podataka
      LOAD   #100h     ;upis vel bloka podat za PER0 i PER2 preko ACC u
      OUT    FF0Bh     ;reg Count DMAPER0 i
      STORE  MemCnt2   ;mem. lok. na adr. MemCnt2
    ;početne adrese blokova podataka
      LOAD   #A000h    ;upis adr. bloka podataka za PER0 preko ACC u
      OUT    FF0Dh     ;reg AdrDst DMAPER0
      LOAD   #B000h    ;upis adr. bloka podataka za PER2 preko ACC u
      STORE  MemAdr2   ;mem. lok. na adr. MemAdr2
  ;postavljanje "semafor"-a za DMAPER0 na 1
       LOAD   #1h      ;upis vrednosti 1 preko ACC u
       STORE  MemSem0  ;mem. lok. na adr. MemSem0
  ;startovanje kontrolera DMAPER0
       LOAD   #0013h   ;upis režima rada i startovanja (u/i=0, start=1,
       OUT    FF08h    ;burst=1,enable=1)preko ACC u reg. Control DMAPER0
  ;startovanje kontrolera periferija KPER2
       LOAD   #0010h   ;upis režima rada i startovanja (u/i=0,
       OUT    FF20h    ;start=1,enable=0)preko ACC u reg. Control KPER2
  ;unos blokova podataka
    ;provera da li podatak može da se prenese iz registra Data KPER2
Loop2: IN     FF21h    ;upis sadrž. reg. Status KPER2 u ACC
       AND    #8000h   ;ispitivanje bita ready
       JZ     Loop2    ;povratak na Loop2 ukoliko je ready 0
    ;prenos podatka iz registra Data KPER2 u memorijsku lokaciju
      IN     FF22h    ;upis sadrž. reg. Data KPER2 preko ACC u mem. lok.
      STORE  (MemAdr2);na adresi datoj sadrž. mem. lok. na adr. MemAdr2
    ;inkrementiranje sadržaja memorijske lokacije MemAdr2
      INC    MemAdr2  ;inkrement sadrž. mem.lok. sa adr. MemAdr2
    ;dekrementiranje sadržaja memorijske lokacije MemCnt2
      DEC    MemCnt2  ;dekrement sadrž. mem.lok. sa adr. MemCnt2
    ;provera da li je unet poslednji podatak iz PER2
      JNZ    Loop2    ;povratak na Loop2 ukoliko nije poslednji podatak
  ;zaustavljanje kontrolera periferije KPER2
       LOAD   #0       ;upis režima zaustavljanja (u/i=0,
       OUT    FF20h    ;start=0, enable=0) preko ACC u reg. Control KPER2
  ;čekanje na završetak unosa bloka podataka iz periferije PER0 u memoriju
  ;sa DMAPER0 proverom vrednosti "semafor"-a za DMAPER0
Wait0: LOAD   MemSem0  ;upis sadrž. mem.lok. sa adr. MemSem0 u ACC
       CMP    #0       ;ispitivanje da li je "semafor" postao 0
       JNZ    Wait0    ;povratak na Wait0 ukoliko "semafor" nije 0
;obrada
       JSR    Obrada   ;obrada blokova od po 100h podat u mem lokacijama
  ;od adrese A000h do adrese A0FFh i od adrese B000h do adrese B0FFh
  ;i upis rezultata u mem lokacije od adrese C000h do adrese C0FFh
```

```
;slanje bloka podat iz mem u periferiju PER1 sa KPER1 generisanjem prekida
  ;inicijalizacija prenosa iz mem u periferiju PER1 sa KPER1
    ;veličina bloka podataka
      LOAD   #100h     ;upis veličine bloka podataka za PER1 preko ACC u
      STORE  MemCnt1   ;mem. lok. na adr. MemCnt1
    ;početna adresa bloka podataka
      LOAD   #C000h    ;upis adr. bloka podataka za PER1 preko ACC u
      STORE  MemAdr1   ;mem. lok. na adr. MemAdr1
  ;postavljanje "semafor"-a za PER1 na 1
      LOAD   #1h       ;upis vrednosti 1 preko ACC u
      STORE  MemSem1   ;mem. lok. na adr. MemSem1
  ;startovanje kontrolera periferije KPER1
      LOAD   #8011h    ;upis režima rada i startovanja (u/i=1,
      OUT    FF10h     ;start=1,enable=1)preko ACC u reg. Control KPER1
  ;čekanje na završetak slanja bloka podataka iz memorije u periferiju
  ;PER1 proverom vrednosti "semafor"-a za PER1
Wait1: LOAD   MemSem1   ;upis sadrž. mem.lok. sa adr. MemSem1 u ACC
       CMP    #0        ;ispitivanje da li je "semafor" postao 0
       JNZ    Wait1     ;povratak na Wait1 ukoliko "semafor" nije 0
. . .

;prekidna rutina za DMAPER0
DMAPER0:  PUSHA              ;akumulator ACC na stek
  ;zaustavljanje kontrolera DMAPER0
                             ;upis režima zaustav(u/i=0,mem/per=0, start=0,
          OUT FF08h          ;burst=0,enable=0)preko ACC u reg. Control DMA0
  ;postavljanje "semafor"-a za DMAPER0 na 0
                             ;upis vrednosti 0 preko ACC u
          STORE    MemSem0   ;mem. lok. na adr. MemSem0
  ;izlazak iz prekidne rutine
          POPA               ;restauracija akumulatora ACC sa steka
          RTI                ;povratak iz prekidne rutine

;Procedura Obrada u kojoj se izračunava C(i)=A(i)+B(i), gde je i=0,…,FFh
;Elementi niza A(i), gde je i=0,…,FFh, su na adresama A000h do A0FFh,
;elementi niza B(i), gde je i=0,…,FFh, su na adresama B000h do B0FFh, a
;elementi niza sume C(i), gde je i=0,…,FFh, se smešt na adr C000h do C0FFh
Obrada:   LOAD   #100h     ;upis vel. blok. podat. za A(i), B(i) i C(i)
          STORE  MemCnt1   ;u ACC pa u mem. lok. na adr. MemCnt1
          LOAD   #A000h    ;upis adr. bloka podataka A(i)preko ACC u
          STORE  MemAdr0   ;mem. lok. na adr. MemAdr0
          LOAD   #B000h    ;upis adr. bloka podataka B(i)preko ACC u
          STORE  MemAdr2   ;mem. lok. na adr. MemAdr2
          LOAD   #C000h    ;upis adr. bloka podataka C(i)preko ACC u
          STORE  MemAdr1   ;mem. lok. na adr. MemAdr1
Loop:     LOAD   (MemAdr0) ;upis A(i) iz mem lok sa adrese date sadr
                           ;mem lok sa adrese MemAdr0 u ACC
          ADD    (MemAdr2) ;suma ACC i B(i) iz mem lok sa adrese date sadr
                           ;mem lok sa adrese MemAdr2 u ACC
          STORE  (MemAdr1) ;sadržaj ACC u C(i) u mem lok na adr datoj sadr
                           ;mem lok na adresi MemAdr1
          INC    MemAdr0   ;inkrementiranje sadr mem lok MemAdr0
          INC    MemAdr2   ;inkrementiranje sadr mem lok MemAdr2
          INC    MemAdr1   ;inkrementiranje sadr mem lok MemAdr1
          DEC    MemCnt1   ;dekrementiranje sadr mem lok MemCnt1
  ;provera da li je poslednja suma
          JNZ    Loop      ;povratak na Loop ukoliko nije poslednja suma
          RTS              ;povratak u glavni program
```

```
;prekidna rutina periferije PER1
;slanje bloka podataka iz memorije u periferiju PER1 sa KPER1
PER1:  PUSHA           ;akumulator ACC na stek
  ;prenos podatka iz memorijske lokacije u registar Data KPER1
       LOAD   (MemAdr1);upis sadrž. mem. lok. sa adr. date sadrž. mem.
       OUT    FF12h    ;lok. sa adr. MemAdr1 preko ACC u reg. Data KPER1
  ;inkrementiranje sadržaja memorijske lokacije MemAdr1
       INC    MemAdr1  ;inkr sadrž. mem.lok. sa adr. MemAdr1
  ;dekrementiranje sadržaja memorijske lokacije MemCnt1
       DEC    MemCnt   ;dekr sadrž. mem.lok. sa adr. MemCnt1
  ;provera da li je prenet poslednji podatak
       JNZ    Back1    ;prelaz na Back1 ukoliko nije poslednji podatak
  ;zaustavljanje kontrolera periferije PER1
       LOAD   #0       ;upis režima zaustavljanja (start=0, u/i=0,
       OUT    FF10h    ;enable=0)preko ACC u reg. Control KPER1
  ;postavljanje "semafor"-a za PER1 na 0
                       ;upis vrednosti 0 preko ACC u
       STORE  MemSem1  ;mem. lok. na adr. MemSem1
  ;izlazak iz prekidne rutine
Back1: POPA            ;restauracija akumulatora ACC sa steka
       RTI             ;povratak iz prekidne rutine
. . .
```

Слика 16 Програм

За унос података из периферије **PER0** у меморију се у овом случају користи DMA контролер периферије **DMAPER0** у коме су интегрисане функције контролера периферије **KPER0** и DMA контролера **DMA0**. Унос блока података из периферије **PER0** у меморију започиње тако што се извршавањем одговарајућег програма у управљачки регистар *Control* **DMAPER0** упише садржај којим се DMA контролер периферије **DMAPER0** стартује (*start*=1) за рад у режиму улаза (*u/i*=0, *mem/per*=0) и то циклус по циклус (*burst*=0) са генерисања прекида (*enable*=1). Контролер **DMAPER0** на исти начин као и контролер **KPER0** чита податак из периферије **PER0** и преноси у регистар податка *Data* **DMAPER0** и на сличан начин као и контролер **DMA0** реализује пренос податка из регистра *Data* **DMAPER0** у меморијску локацију, генерише адресе одредишних меморијских локација у које треба да уписује један за другим податке које чита из периферије **PER0** и води евиденцију колико још података треба пренети из периферије **PER0** у меморију. Контролер **DMAPER0** као и контролер **DMA0** има одредишни адресни регистар *AdrDst* и регистар величине блока података *Count*. Стога је пре старовања контролера **DMAPER0** потребно у оквиру иницијализације преноса блока података из периферије **PER0** у меморију, почетну адресу меморијске локације од које треба унети блок података и величину блока података уписати у регистре *AdrDst* и *Count* **DMAPER0**. Такође је потребно као индикацију да је пренос података у току у "semafor" за **DMAPER0** уписати вредност 1. Контролер **DMAPER0** мора после сваког пренетог податка из регистра *Data* **DMAPER0** у меморијску локацију да изврши инкрементирање садржаја регистра *AdrDst* **DMAPER0** и декрементирање садржаја регистра *Count* **DMAPER0** и изврши проверу да ли је садржај регистра *Count* **DMAPER0** декрементирањем постао 0. Уколико садржај регистра *Count* **DMAPER0** није нула, контролер **DMAPER0** продужава са преносом података из периферије **PER0** у меморију тако што чита следећи податак из периферије **PER0** и преноси у регистар податка *Data* **DMAPER0** и из њега у меморијску локацију, док у супротном случају завршава са преносом података из периферије **PER0** у меморију и у бит *end* статусног регистра *Status* **DMAPER0** уписује вредност 1 и генерише прекид. У прекидној рутини контролера **DMAPER0** се извршавањем одговарајућег програма најпре "semafor" за

DMAPER0 поставља на 0 а затим у управљачки регистар *Control* **DMAPER0** уписује садржај којим се контролер **DMAPER0** зауставља (*start*=0).

Пребацивање податка из регистра података *Data* **DMAPER0** у меморијску локацију се незнатно разликује од случаја када се користе контролери **KPER0** и **DMA0**. Контролер **DMAPER0** најпре поставља на 1 сигнал hreq којим, у зависности од тога како је реализована арбитрација, од процесора или арбитратора магистрале тражи дозволу за коришћење магистрале. Тек када вредношћу 1 сигнала hack добије дозволу за коришћење магистрале, контролер **DMAPER0** креће са реализацијом циклуса уписа. Том приликом контролер **DMAPER0** најпре отвара бафере са три стања и на адресне линије магистрале ABUS и линије података магистрале DBUS пушта садржај одредишног адресног регистра *AdrDst* **DMAPER0** и регистра података *Data* **DMAPER0**, респективно, и затим стартује операцију уписа постављањем управљачког сигнала уписа WRBUS на вредност 1. Пошто је узето да је магистрала асинхрона, контролер **DMAPER0** држи садржај на адресним линијама магистрале ABUS и линијама података магистрале DBUS и управљачки сигнал уписа WRBUS на вредности 1 све време док упис у меморију траје. По завршетку уписа најпре меморија поставља управљачки сигнал завршетка циклуса на магистрали FCBUS на вредност 1, па затим контролер **DMAPER0** поставља сигнал WRBUS на вредност 0 и на крају меморија поставља сигнал FCBUS на вредност 0. Тиме је циклус уписа завршен, па контролер **DMAPER0** затвара бафере са три стања и адресне линије магистрале ABUS и линије података магистрале DBUS ставља у стање високе импедансе. Затим контролер **DMAPER0** постављањем сигнала hreq на вредност 0 укида захтев за коришћење магистрале и као одговор процесор или арбитратор магистрале му вредношћу 0 сигнала hack укидају дозволу за коришћење магистрале. Поред тога, контролер **DMAPER0** инкрементира садржај регистра *AdrDst* **DMA0** и декрементира садржај регистра *Count* **DMAPER0** и врши проверу да ли је садржај регистра *Count* **DMAPER0** декрементирањем постао 0. Уколико садржај регистра *Count* **DMAPER0** није нула, контролер **DMAPER0** започиње читање следећег податка из периферије **PER0** и пренос у регистар података *Data* **DMAPER0**. Међутим, уколико је садржај регистра *Count* **DMAPER0** нула, контролер **DMAPER0** уписује вредност 1 у бит *end* статусног регистра *Status* **DMAPER0** и генерише прекид. У прекидној рутини контролера **DMAPER0** се извршавањем одговарајућег програма најпре "semafor" за **DMAPER0** постави на 0 а затим у управљачки регистар *Control* **DMAPER0** упише садржај којим се контролер **DMAPER0** зауставља (*start*=0).

DMA контролер периферије **DMAPER0** може да ради и у режиму преноса цео блок и то на исти начин као и DMA контролер **DMA** (**Дискусија** под **1**).

Функционисање DMA контролера периферије **DMAPER0** као и коришћење изворишног адресног регистра *AdrSrc* при преносу блока података из меморије у периферију објашњено је у задатку 1.7, док је функционисање DMA контролера периферије **DMAPER0** као и коришћење изворишног и одредишног адресног регистра *AdrSrc* и *AdrDst* при преносу блока података из меморије у меморију објашњено у задатку 1.8.

1.7 ЗАДАТАК

Дат је рачунарски систем који чине процесор, меморија и контролер периферије **KPER0** са периферијом **PER0**, контролер периферије **KPER1** са периферијом **PER1** и контролер периферије **KPER2** и DMA контролер **DMA2** са периферијом **PER2** повезани магистралом.

Процесор је са једноадресним форматом инструкција. Од програмски доступних регистара постоје 16 битни акумулатор ACC, указивач на врх стека SP, указивач на табелу (IV табела) са адресама прекидних рутина IVTP и програмска статусна реч PSW. Типови података који се користе су 16 битне целобројне величине са знаком и без знака. Приликом извршавања инструкције преноса у акумулатор и аритметичких, логичких и померачких инструкција постављају се индикатори N, Z, C и V програмске статусне речи PSW. Механизам прекида је векторисан. При прекиду хардверски се на стеку чувају програмски бројач PC и програмска статусна реч PSW. Бројеви улаза у IV табелу су фиксни за сваку периферију. Меморијски и улазно-излазни адресни простори су раздвојени.

Меморијски адресни простор је величине 2^{16} меморијских локација, при чему је ширина меморијских локација 16 битова.

Улазно-излазни адресни простор је величине 2^{16} регистара, при чему је ширина регистара 16 битова. Регистри контролера периферија **KPER0**, **KPER1** и **KPER2** и регистри DMA контролера **DMA2** се налазе у улазно-излазном адресном простору.

Контролери периферија **KPER0**, **KPER1** и **KPER2** имају управљачке регистре (*Control*), статусне регистре (*Status*) и регистре података (*Data*) и то редом на адресама FF00h, FF01h и FF02h за контролер периферије **KPER0**, FF10h, FF11h и FF12h за контролер периферије **KPER1** и FF20h, FF21h и FF22h за контролер периферије **KPER2**. У регистрима контролера периферија **KPER0**, **KPER1** и **KPER2** највиши бит је означен са 15, а најнижи са 0. У управљачким регистрима *Control* бит 15 је бит *u/i* којим се задаје смер преноса са периферијом (0—улаз, 1—излаз), бит 4 је бит *start* којим се покреће извршавање операције (0—заустављен, 1—стартован) и бит 0 је бит *enable* којим се дозвољава прекид (0—маскиран, 1—дозвољен). У статусним регистрима *Status* бит 15 је бит спремности *ready* (0—није спреман, 1—спреман).

DMA контролер **DMA2** има управљачки регистар (*Control*), статусни регистар (*Status*), регистар податка (*Data*), регистар величине блока података (*Count*), изворишни адресни регистар (*AdrSrc*) и одредишни адресни регистар (*AdrDst*) и то редом на адресама FF28h, FF29h, FF2Ah, FF2Bh, FF2Ch и FF2Dh, респективно. У регистрима DMA контролера **DMA2** највиши бит је означен са 15, а најнижи са 0. У управљачком регистру *Control* бит 15 је бит *u/i* којим се задаје смер преноса са периферијом (0—улаз, 1—излаз), бит 14 је бит *mem/per* којим се задаје пренос између делова меморије или између периферије и меморије (0—периферија/меморија, 1—меморија/меморија), бит 4 је бит *start* којим се покреће извршавање операције (0—заустављен, 1—стартован), бит 1 је бит *burst* којим се задаје режим преноса (0—циклус по циклус, 1—цео блок) и бит 0 је бит *enable* којим се дозвољава прекид (0—маскиран, 1—дозвољен), при чему је вредност бита 15 битна само уколико је вредношћу 0 бита 14 задат пренос између периферије и меморије. У статусном регистру *Status* бит 15 је бит завршетка преноса *end* (0—пренос у току, 1—пренос завршен).

Адресне линије и линије података магистрале рачунара су широке по 16 битова.

Написати главни програм и одговарајућу прекидну рутину којима се упоредо учитавају низ A(i), где је i = 0, …, FFh, из **PER0** у меморијски блок који почиње од адресе A000h и низ B(i), где је i = 0, …, FFh, из **PER1** у меморијски блок који почиње од адресе B000h, затим формира низ C(i) = A(i) + B(i), где је i = 0, …, FFh, у меморијском блоку који почиње од адресе C000h и резултујући низ C(i), где је i = 0, …, FFh, из меморијског блока који почиње од адресе C000h шаље у **PER2**. Улаз из периферије **PER0** реализовати са контролером периферије **KPER0** техником програмираног улаза са прекидом, улаз из периферије **PER1** реализовати са контролером периферије **KPER1** техником програмираног улаза са испитивањем спремности, а излаз у периферију **PER2** реализовати са контролером периферије **KPER2** и DMA контролером **DMA2** у режиму преноса циклус по циклус.

Решење:

Адресе и структура регистара контролера **KPER0**, **KPER1, KPER2** и **DMA2** су дати на сликама 17 и 18, респективно.

KPER0 - Kontroler periferije **PER0**

Registar	Control	Status	Data
Adresa	FF00h	FF01h	FF02h

KPER1 - Kontroler periferije **PER1**

Registar	Control	Status	Data
Adresa	FF10h	FF11h	FF12h

KPER2 - Kontroler periferije **PER2**

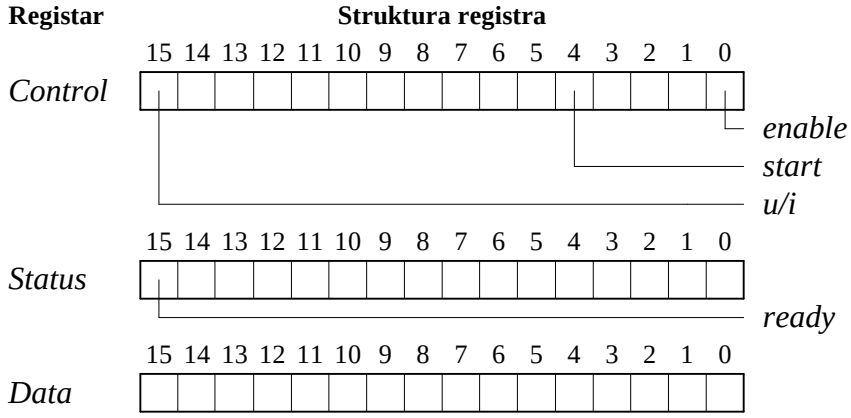
Registar	Control	Status	Data
Adresa	FF20h	FF21h	FF22h

DMA2 –DMA kontroler

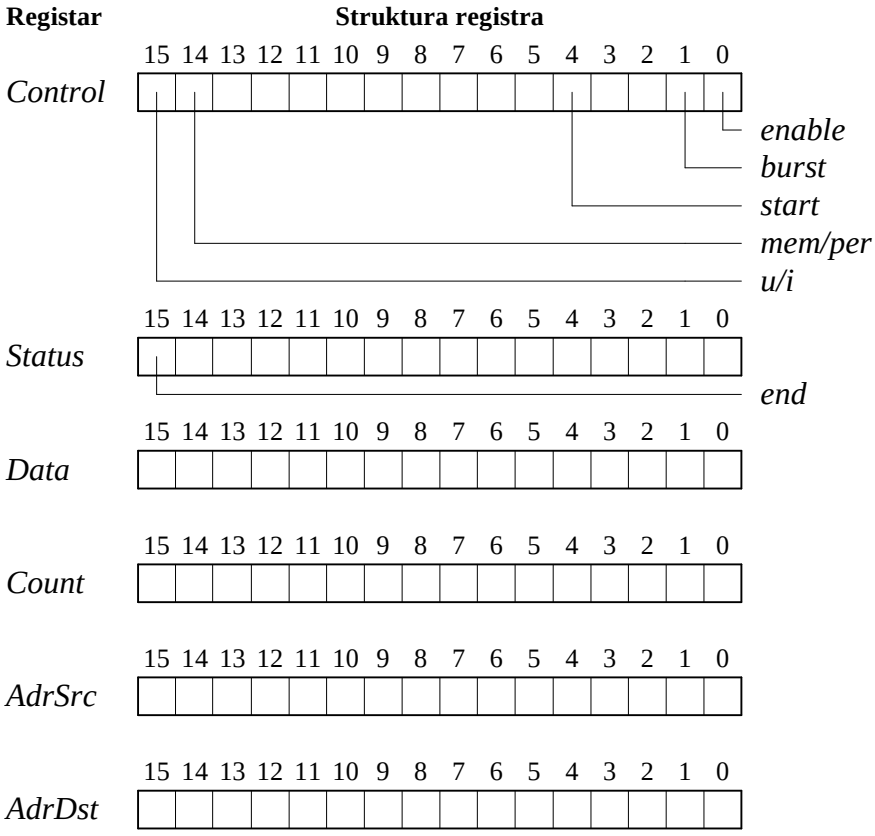
Registar	Control	Status	Data	Count	AdrSrc	AdrDst
Adresa	FF28h	FF29h	FF2Ah	FF2Bh	FF2Ch	FF2Dh

Слика 17 Адресе регистара контролера

Registri kontrolera periferija **KPER**



Registri DMA kontrolera **DMA**



Слика 18 Структура регистара контролера

Периферија **PER0** и **PER1** повезују на магистралу рачунарског система са контролерима периферија **KPER0** и **KPER1** на начин приказан на слици 1, док се периферија **PER2** повезује на магистралу рачунарског система са контролером периферије **KPER2** и DMA контролера **DMA2** на начин приказан на слици 11.

Тражени програм је приказан на слици 19.

```
;glavni program
;unos blokova podataka iz periferije PER0 u mem sa KPER0 sa prekidom i
;iz periferije PER1 u mem sa KPER1 sa ispitivanjem bita spremnosti ready
  ;inicijalizacija prenosa iz PER0 u mem sa KPER0 i iz PER1 u mem sa KPER1
    ;veličina blokova podataka
      LOAD   #100h     ;upis vel bloka podat za PER0 i PER1 preko ACC u
      STORE  MemCnt0   ;mem. lok. na adr. MemCnt0 i
      STORE  MemCnt1   ;mem. lok. na adr. MemCnt1
    ;početne adrese blokova podataka
      LOAD   #A000h    ;upis adr. bloka podataka za PER0 preko ACC u
      STORE  MemAdr0   ;mem. lok. na adr. MemAdr0
      LOAD   #B000h    ;upis adr. bloka podataka za PER1 preko ACC u
      STORE  MemAdr1   ;mem. lok. na adr. MemAdr1
  ;postavljanje "semafor"-a za PER0 na 1
      LOAD   #1h       ;upis vrednosti 1 preko ACC u
      STORE  MemSem0   ;mem. lok. na adr. MemSem0
  ;startovanje kontrolera periferije KPER0
      LOAD   #0011h    ;upis režima rada i startovanja (u/i=0,
      OUT    FF00h     ;start=1,enable=1)preko ACC u reg. Control KPER0
  ;startovanje kontrolera periferije KPER1
      LOAD   #0010h    ;upis režima rada i startovanja (u/i=0,
      OUT    FF10h     ;start=1,enable=0)preko ACC u reg. Control KPER1
  ;unos blokova podataka
    ;provera da li podatak može da se prenese iz registra Data KPER1
Loop1: IN     FF11h     ;upis sadrž. reg. Status KPER1 u ACC
       AND    #8000h    ;ispitivanje bita ready
       JZ     Loop1     ;povratak na Loop1 ukoliko je ready 0
    ;prenos podatka iz registra Data KPER1 u memorijsku lokaciju
      IN     FF12h     ;upis sadrž. reg. Data KPER1 preko ACC u mem. lok.
      STORE  (MemAdr1) ;na adresi datoj sadrž. mem. lok. na adr. MemAdr1
    ;inkrementiranje sadržaja memorijske lokacije MemAdr1
      INC    MemAdr1   ;inkrement sadrž. mem.lok. sa adr. MemAdr1
    ;dekrementiranje sadržaja memorijske lokacije MemCnt1
      DEC    MemCnt1   ;dekrement sadrž. mem.lok. sa adr. MemCnt1
    ;provera da li je unet poslednji podatak iz PER1
      JNZ    Loop1     ;povratak na Loop1 ukoliko nije poslednji podatak
  ;zaustavljanje kontrolera periferije KPER1
      LOAD   #0        ;upis režima zaustavljanja (u/i=0,
      OUT    FF10h     ;start=0, enable=0) preko ACC u reg. Control KPER1
  ;čekanje na završetak unosa bloka podataka iz periferiju PER0 u memoriju
  ;proverom vrednosti "semafor"-a za PER0
Wait0: LOAD   MemSem0   ;upis sadrž. mem.lok. sa adr. MemSem0 u ACC
       CMP    #0        ;ispitivanje da li je "semafor" postao 0
       JNZ    Wait0     ;povratak na Wait0 ukoliko "semafor" nije 0
;obrada
       JSR    Obrada    ;obrada blokova od po 100h podat u mem lokacijama
  ;od adrese A000h do adrese A0FFh i od adrese B000h do adrese B0FFh
  ;i upis rezultata u mem lokacije od adrese C000h do adrese C0FFh
```

```
;slanje bloka podataka iz memorije u periferiju PER2 sa KPER2 i DMA2
  ;inicijalizacija prenosa iz memorije u periferiju PER2 sa KPER2 i DMA2
    ;veličina bloka podataka
      LOAD   #100h     ;upis veličine bloka podataka za PER2 preko ACC u
      OUT    FF2Bh     ;u reg. Count DMA2
    ;početna adresa bloka podataka
      LOAD   #C000h    ;upis adr. bloka podataka za PER2 preko ACC u
      STORE  FF2Ch     ;u reg. AdrSrc DMA2
  ;postavljanje "semafor"-a za DMA2 na 1
       LOAD   #1h      ;upis vrednosti 1 preko ACC u
       STORE  MemSem2  ;mem. lok. na adr. MemSem2
  ;startovanje kontrolera periferije KPER2
       LOAD   #8010h   ;upis režima rada i startovanja (u/i=1,
       OUT    FF20h    ;start=1,enable=0)preko ACC u reg. Control KPER2
  ;startovanje kontrolera DMA2
       LOAD   #8011h   ;upis režima rada i start (u/i=1,mem/per=0,start=1,
       OUT    FF28h    ;burst=0,enable=1)preko ACC u reg. Control DMA2
  ;čekanje na završetak slanja bloka podataka iz memorije u periferiju
  ;PER2 sa KPER2 i DMA2 proverom vrednosti "semafor"-a za DMA2
Wait2: LOAD   MemSem2  ;upis sadrž. mem.lok. sa adr. MemSem2 u ACC
       CMP    #0       ;ispitivanje da li je "semafor" postao 0
       JNZ    Wait2    ;povratak na Wait2 ukoliko "semafor" nije 0
. . .
;prekidna rutina periferije PER0
;unos bloka podataka iz periferiju PER0 u memoriju sa KPER0
PER0:  PUSHA            ;akumulator ACC na stek
  ;unos podatka iz registra Data KPER0 u memorijsku lokaciju
       IN FF02h         ;upis sadrž. reg. Data KPER0 preko ACC u mem. lok.
       STORE  (MemAdr0) ;na adr. date sadrž. mem. lok. na adr. MemAdr0
  ;inkrementiranje sadržaja memorijske lokacije MemAdr0
       INC    MemAdr0   ;inkr sadrž. mem.lok. sa adr. MemAdr0
  ;dekrementiranje sadržaja memorijske lokacije MemCnt0
       DEC    MemCnt    ;dekr sadrž. mem.lok. sa adr. MemCnt0
  ;provera da li je prenet poslednji podatak
       JNZ    Back0     ;prelaz na Back0 ukoliko nije poslednji podatak
  ;zaustavljanje kontrolera periferije KPER0
       LOAD   #0        ;upis režima zaustavljanja (start=0, u/i=0,
       OUT FF10h        ;enable=0)preko ACC u reg. Control KPER0
  ;postavljanje "semafor"-a za PER0 na 0
                        ;upis vrednosti 0 preko ACC u
       STORE  MemSem0   ;mem. lok. na adr. MemSem0
  ;izlazak iz prekidne rutine
Back0: POPA             ;restauracija akumulatora ACC sa steka
       RTI              ;povratak iz prekidne rutine
```

```
;Procedura Obrada u kojoj se izračunava C(i)=A(i)+B(i), gde je i=0,…,FFh
;Elementi niza A(i), gde je i=0,…,FFh, su na adresama A000h do A0FFh,
;elementi niza B(i), gde je i=0,…,FFh, su na adresama B000h do B0FFh, a
;elementi niza sume C(i), gde je i=0,…,FFh, se smešt na adr C000h do C0FFh
Obrada:  LOAD   #100h     ;upis vel. blok. podat. za A(i), B(i) i C(i)
         STORE  MemCnt2   ;u ACC pa u mem. lok. na adr. MemCnt2
         LOAD   #A000h    ;upis adr. bloka podataka A(i)preko ACC u
         STORE  MemAdr0   ;mem. lok. na adr. MemAdr0
         LOAD   #B000h    ;upis adr. bloka podataka B(i)preko ACC u
         STORE  MemAdr1   ;mem. lok. na adr. MemAdr1
         LOAD   #C000h    ;upis adr. bloka podataka C(i)preko ACC u
         STORE  MemAdr1   ;mem. lok. na adr. MemAdr2
Loop:    LOAD   (MemAdr0) ;upis A(i) iz mem lok sa adrese date sadr
                          ;mem lok sa adrese MemAdr0 u ACC
         ADD    (MemAdr1) ;suma ACC i B(i) iz mem lok sa adrese date sadr
                          ;mem lok sa adrese MemAdr1 u ACC
         STORE  (MemAdr2) ;sadržaj ACC u C(i) u mem lok na adr datoj sadr
                          ;mem lok na adresi MemAdr2
         INC    MemAdr0   ;inkrementiranje sadr mem lok MemAdr0
         INC    MemAdr1   ;inkrementiranje sadr mem lok MemAdr1
         INC    MemAdr2   ;inkrementiranje sadr mem lok MemAdr2
         DEC    MemCnt2   ;dekrementiranje sadr mem lok MemCnt2
  ;provera da li je poslednja suma
         JNZ    Loop      ;povratak na Loop ukoliko nije poslednja suma
         RTS              ;povratak u glavni program

;prekidna rutina za DMA2
DMA2:  PUSHA              ;akumulator ACC na stek
  ;zaustavljanje kontrolera periferije KPER2
       LOAD   #0          ;upis režima zaustavljanja (u/i=0,
       OUT FF20h          ;start=0,enable=0)preko ACC u reg. Control KPER2
  ;zaustavljanje kontrolera DMA2
                          ;upis režima zaustavjanja (u/i=0,mem/per=0,start=0,
       OUT FF28h          ;burst=0,enable=0)preko ACC u reg. Control DMA2
  ;postavljanje "semafor"-a za DMA2 na 0
                          ;upis vrednosti 0 preko ACC u
       STORE  MemSem2     ;mem. lok. na adr. MemSem2
  ;izlazak iz prekidne rutine
       POPA               ;restauracija akumulatora ACC sa steka
       RTI                ;povratak iz prekidne rutine
. . .
```

Слика 19 Програм

За слање података из меморије у периферију **PER2** се у овом задатку користи не само контролер периферије **KPER2**, већ и DMA контролер **DMA2**. Контролер периферије **KPER2** се у овом задатку иницијализује и стартује за слање блока података из меморије у периферију **PER2** на сличан начин као и у задацима 1.1 и 1.2 и у овом задатку функционише на сличан начин као и у задацима 1.1 и 1.2. Слање блока података из меморије у периферију **PER2** започиње тако што се извршавањем одговарајућег програма у управљачки регистар *Control* **KPER2** упише садржај којим се контролер **KPER2** стартује (*start*=1) за рад у режиму излаза (*u/i*=1) и без генерисања прекида (*enable*=0). Том приликом контролер **KPER2** поставља на вредност 1 бит спремности *ready* статусног регистра *Status* **KPER2** и вредношћу 1 сигнала dreq који иде из контролера периферије **KPER2** у DMA контролер **DMA2** тражи од DMA контролера **DMA2** да податак пренесе из меморијске локације у регистар *Data* **KPER2**. У овом задатку се сам пренос податка из меморијске локације у регистар *Data* **KPER2** за разлику од задатака 1.1 и 1.2 не реализује програмским путем неком од техника програмираног излаза са испитивањем спремности (задатак 1.1) или са прекидом (задатак 1.2), већ се реализује хардверски са DMA контролером **DMA2** сигналом dack који иде из DMA контролера **DMA2** у контролер периферије **KPER2**. Том приликом контролер **KPER2** поставља на вредност 0 бит *ready* статусног регистра *Status* **KPER2** и започиње пренос податка из регистра података *Data* **KPER2** у периферију **PER2**. Контролер **KPER2**, када заврши пренос податка из регистра података *Data* **KPER2** у периферију **PER2**, поново поставља на вредност 1 бит спремности *ready* статусног регистра *Status* **KPER2** и сигнал dreq.

DMA контролер **DMA2** хардверски генерише адресе изворишних меморијских локација из којих треба пренети податаке у периферију **PER2**, реализује читање података из изворишних меморијских локација и упис у регистар *Data* **KPER2** и води евиденцију колико још података треба пренети из меморије у периферију **PER2**. Због тога DMA контролер **DMA0** има изворишни адресни регистар *AdrSrc* и регистар величине блока података *Count*. За разлику од задатака 1.1 и 1.2 код којих се, у оквиру иницијализације преноса блока података из меморије у периферију **PER2** почетна адреса меморијске локације од које треба пренети блок података и величина блока података уписују у меморијске локациме MemAdr0 и MemCnt0, у овом задатку се те вредности уписују у регистре *AdrSrc* и *Count* DMA контролера **DMA2**. DMA контролер **DMA2** мора после сваког пренетог податка из меморијске локације у регистар *Data* **KPER2** да изврши инкрементирање садржаја регистра *AdrSrc* **DMA2** и декрементирање садржаја регистра *Count* **DMA2** и изврши проверу да ли је садржај регистра *Count* **DMA2** декрементирањем постао 0. Уколико садржај регистра *Count* **DMA2** није нула, DMA контролер **DMA2** треба да продужи са преносом података из меморије у периферију **PER2**, док у супротном случају треба да завршити са преносом података из меморије у периферију **PER2** и у бит *end* статусног регистра *Status* **DMA2** упише вредност 1.

Слање блока података из меморије у периферију **PER2** са **KPER2** и **DMA2** започиње тако што се извршавањем одговарајућег програма најпре "semafor" за **DMA2** постави на 1, затим контролер периферије **KPER2** стартује на начин објашњен у претходном параграфу, и на крају у управљачки регистар *Control* **DMA2** упише садржај којим се DMA контролер **DMA2** стартује (*start*=1) за рад у режиму излаза (*u/i*=1, *mem/per*=0) и то циклус по циклус (*burst*=0) са генерисања прекида (*enable*=1) на начин објашњен у даљем тексту. Слање блока података из меморије у периферију **PER2** одвија се комплетно хардверски тако што сваки податак из блока података најпре DMA контролер **DMA2** преноси из меморијске локације у регистар података *Data* **KPER2** а затим контролер периферије **KPER2** преноси податак из регистра података *Data* **KPER2** у

периферију **PER2**. Ово се понавља све док садржај регистра *Count* **DMA2** декрементирањем не постане 0, када DMA контролер **DMA2** уписује вредност 1 у бит *end* статусног регистра *Status* **DMA2** и генерише прекид. У прекидној рутини DMA контролера **DMA2** се извршавањем одговарајућег програма најпре "semafor" за **DMA2** поставља на 0 а затим у управљачке регистре *Control* **KPER2** и **DMA2** уписује садржај којим се контролер периферије **KPER2** и DMA контролер **DMA2** зауставаљају (*start*=0).

Пренос једног податка из меморијске локације у периферију **PER2** контролери **KPER2** и **DMA2** реализују на начин објашњен у даљем тексту. По стартовању контролера **KPER2** и **DMA2**, контролер периферије **KPER2** поставља на вредност 1 бит спремности *ready* статусног регистра *Status* **KPER2** и вредношћу 1 сигнала dreq тражи од DMA контролер **DMA2** да податак пренесе из меморијске локације у регистар *Data* **KPER2**, док DMA контролер **DMA2** чека да контролер периферије **KPER2** постави сигнал dreq на 1. Када сигнал dreq постане 1, DMA контролер **DMA2** креће са пребацивањем податка из меморијске локације у регистар података *Data* **KPER2**, док контролер периферије **KPER2** чека да **DMA2** пребаци податак из меморијске локације у регистар података *Data* **KPER2** па да тек онда крене са пребацивањем података из регистра *Data* **KPER2** у периферију **PER2.** DMA контролер **DMA2** најпре поставља на 1 сигнал hreq којим, у зависности од тога како је реализована арбитрација, од процесора или арбитратора магистрале тражи дозволу за коришћење магистрале. Тек када вредношћу 1 сигнала hack добије дозволу за коришћење магистрале, DMA контролер **DMA2** креће са реализацијом циклуса читања. Том приликом **DMA2** најпре отвара бафере са три стања и на адресне линије магистрале ABUS пушта садржај изворишног адресног регистра *AdrSrc* **DMA2** и затим стартује операцију читања постављањем управљачког сигнала читања RDBUS на вредност 1. Пошто је узето да је магистрала асинхрона, DMA контролер **DMA2** држи садржај на адресним линијама магистрале ABUS и управљачки сигнал читања RDBUS на вредности 1 све време док читање из меморије траје. Када меморија по завршетку читања отвори своје бафере са три стања и на линије података магистрале DBUS пусти прочитани садржај и постави управљачки сигнал завршетка циклуса на магистрали FCBUS на вредност 1, DMA контролер **DMA2** постави на вредност 1 сигнал dack, који иде из DMA контролера **DMA2** у контролер периферије **KPER2**, и тиме омогући контролеру периферије **KPER2** да садржај са линија података магистрале DBUS упише у регистар података *Data* **KPER2.** По завршетку уписа података у регистар *Data* **KPER2** најпре контролер периферије **KPER2** постављањем сигнал dreq на вредност 0 шаље индикацију DMA контролеру **DMA2** да му садржај на линијама података магистрале DBUS није више потребан, а затим DMA контролер **DMA2** постављањем сигнала RDBUS на вредност 0 шаље индикацију меморији да нема потреба да меморија више држи садржај на линијама података магистрале DBUS. На вредност 0 сигнала RDBUS меморија затвара бафере са три стања и линије података магистрале DBUS ставља у стање високе импедансе, а сигнал FCBUS поставља на вредност 0. На вредност 0 сигнала FCBUS DMA контролер **DMA2** затвара бафере са три стања и адресне линије магистрале ABUS и управљачку линију читања RDBUS ставља у стање високе импедансе и вредношћу 0 сигнала dack шаље индикацију контролеру периферије **KPER2** да је циклус читања на магистрали завршен**.** На вредност 0 сигнала сигнала dack контролер периферије **KPER0** поставља на вредност 0 бит *ready* статусног регистра *Status* **KPER0** и започиње пренос података из регистра података *Data* **KPER2** у периферију **PER2**. Паралелно са тим DMA контролер **DMA2** постављањем сигнала hreq на вредност 0 укида захтев за коришћење магистрале и као одговор процесор или арбитратор магистрале му вредношћу 0 сигнала hack укидају дозволу за коришћење магистрале. Поред тога, DMA контролер **DMA2** инкрементира садржај регистра *AdrSrc* **DMA2** и декрементира садржај регистра *Count*

DMA2 и врши проверу да ли је садржај регистра *Count* **DMA2** декрементирањем постао 0. Уколико садржај регистра *Count* **DMA2** није нула, DMA контролер **DMA2** прелази на чекање да контролер периферије **KPER2** заврши са преносом податка из регистра *Data* **KPER2** у периферију **PER2** и постави на вредност 1 бит *ready* статусног регистра *Status* **KPER0** и вредношћу 1 сигнала dreq тражи од DMA контролер **DMA2** да податак пренесе из меморијске локације у регистар *Data* **KPER2**. Међутим, уколико је садржај регистра *Count* **DMA2** нула, DMA контролер **DMA2** уписује вредност 1 у бит *end* статусног регистра *Status* **DMA2** и генерише прекид. У прекидној рутини DMA контролера **DMA2** се извршавањем одговарајућег програма најпре "semafor" за **DMA2** постави на 0 а затим у управљачке регистре *Control* **KPER2** и **DMA2** упише садржај којим се контролер периферије **KPER2** и DMA контролер **DMA2** зауставаљају (*start*=0).

Програм се састоји из три дела. У првом делу се из периферија **PER0** и **PER1** упоредо уносе два блока од по 100h података и смештају у меморијске локације од адресе A000h до адресе A0FFh за периферију **PER0** и од адресе B000h до адресе B0FFh за периферију **PER1**. Унос из периферије **PER0** се реализује са контролером периферије **KPER0** техником програмираног улаза са прекидом, док се унос из периферије **PER1** реализује са контролером периферије **KPER2** техником програмираног улаза са испитивањем спремности. У другом делу се позива процесура `Obrada` која врши одређену обраду над унетим подацима у блоковима од по 100h података у меморијским локацијама од адресе A000h до адресе A0FFh и од адресе B000h до адресе B0FFh и резултат који представља блок од 100h података смешта у меморијске локације од адресе C000h до адресе C0FFh. У трећем делу се у периферију **PER2** са контролером периферије **KPER2** и DMA контролером **DMA2** у режиму преноса циклус по циклус шаље блок од 100h података прочитаних из меморијских локација од адресе C000h до адресе C0FFh.

Први део програма у коме се из периферија **PER0** и **PER1** у меморију упоредо уносе два блока података има два дела и то један део који се извршава у главном програму и други део који се извршава у прекидној рутини за **PER0**. У оквиру дела програма који се извршава у главном програму врши се иницијализација преноса из периферија **PER0** и **PER1**, постављање "semafor"-а за **PER0** на 1, стартовање контролера периферија **KPER0** и **KPER1**, унос блока података из периферије **PER1**, заустављање контролера периферије **KPER1** и чекање на завршетак уноса блока података из периферије **PER0** у меморију. У оквиру дела програма који се извршава у прекидној рутини периферије **PER0** врши се пренос податка из регистра *Data* **KPER0** у меморијску локацију, инкрементирање садржаја меморијске локације MemAdr0 и декрементирање садржаја меморијске локације MemCnt0 и, уколико је пренет последњи податак, заустављање контролера периферије **KPER0** и постављање "semafor" за **PER0** на 0.

У оквиру иницијализације преноса из периферија **PER0** и **PER1** најпре се у меморијске локације MemCnt0 и MemCnt1 уписује вредност 100h која представља величину блокова података које треба унети, а затим у меморијске локације MemAdr0 и MemAdr1 уписују вредности A000h и B000h које представљају почетне адресе делова меморије у које треба пренети блокове података из периферија **PER0** и **PER1**, респективно. У оквиру постављања "semafor"-а за **PER0** на 1, у меморијску локацију MemSem0 се уписује вредност 1 која служи као индикација да је унос података из периферије **PER0** у току и да се не сме прећи на извршавање програма у коме се користе подаци из блока меморије у који се уносе подаци из периферије **PER0**. У оквиру стартовања контролера периферија **KPER0** и **KPER1** у управљачки регистар *Control* **KPER0** се уписује вредност 0011h чиме се контролер периферије **KPER0** стартује (*start*=1) за рад у режиму улаза (*u/i*=0) са генерисањем прекида (*enable*=1) и у управљачки регистар *Control* **KPER1** се уписује вредност 0010h чиме се контролер

периферије **KPER1** стартује (*start*=1) за рад у режиму улаза (*u/i*=0) без генерисањем прекида (*enable*=0). Од овог тренутка процесор и контролери периферија **KPER0** и **KPER1** почињу да раде паралелно.

Процесор извршава део главног програм којим најпре у петљи Loop1 реализује унос блока података из периферије **PER1** са контролером периферије **KPER1** техником програмираног улаза са испитивањем спремности на сличан начин као у задатку 1.1, затим уписивањем вредности 0000h у управљачки регистар *Control* **KPER1** зауставља (*start*=0) контролер периферије **KPER1** и на крају у петљи са лабелом Wait0 чека завршетак уноса блока података из периферију **PER0** у меморију.

Паралелно са извршавањем овог дела главног програма од стране процесора, контролер периферије **KPER0**, најпре, пошто је стартован (*start*=1) за рад у режиму улаза (*u/i*=0), поставља на вредност 0 бит спремности *ready* статусног регистра *Status* **KPER0**, затим, започиње пренос податка из периферије **PER0** у регистар података *Data* **KPER0**, потом, по упису података у регистар *Data* **KPER0**, поставља бит *ready* статусног регистра *Status* **KPER0** на вредност 1 и, пошто је стартован (*start*=1) за рад са генерисањем прекида (*enable*=1), генерише прекид и на крају чека да се податак пренесе из регистра *Data* **KPER0** у меморијску локацију.

На прекид генерисан од стране контролера периферије **KPER0**, процесор прекида извршавање главног програма и прелази на извршавање инструкција прекидне рутине периферије **PER0** на чијем почетку преноси податак из регистра *Data* **KPER0** у меморијску локацију. Од овог тренутка процесор и контролер периферије **KPER0** настављају да раде паралелно. Процесор наставља са извршавањем инструкција прекидне рутине тако што инкрементира садржај меморијске локадине MemAdr0, декрементира садржај меморијске локације MemCnt0 и, уколико није пренет последњи податак, враћа се у прекинути главни програм. Контролер периферије **KPER0**, најпре, поставља на вредност 0 бит спремности *ready* статусног регистра *Status* **KPER0**, затим, започиње пренос податка из периферије **PER0** у регистар података *Data* **KPER0**, потом, по упису података у регистар *Data* **KPER0**, поставља бит *ready* статусног регистра *Status* **KPER0** на вредност 1 и генерише прекид и на крају чека да се податак пренесе из регистра *Data* **KPER0** у меморијску локацију.

Рад процесора и контролера приказан у претходним параграфима се наставља до преласка процесора на прекидну рутину ради преноса последњег података. Када садржај меморијске локације MemCnt0 декрементирањем постане 0, процесор најпре изврши инструкције прекидне рутине којима се поставља "semafor" за **PER0** на 0 и зауставља контролер периферије **KPER0**, па се затим враћа у прекинути главни програм.

Прелазак на други део програма у коме се реализује обрада унетих блокова података не сме да се дозволи док се не унесу блокови података из обе периферије. Због тога се по завршетку уноса блока података из периферије **PER1** у петљи са лабелом Wait0 чека завршетак уноса блока података из периферије **PER0.** Том приликом могу да се јаве два случаја у зависности од брзине периферија **PER0** и **PER1**. Први случај се јавља када је периферија **PER1** бржа од периферије **PER0**, па ће се у главном програму завршити унос података из периферије **PER1** пре завршетка уноса података из периферије **PER0**. Тада ће се у главном програму доћи на петљу са лабелом Wait0 и у њој утврдити да "semafor" за **PER0** још увек има вредност 1. Због тога ће се најпре у петљи са лабелом Wait0 чекати да **KPER0** генерише прекид и изазове прелазак на прекидну рутину за **PER0** ради уноса последњег података из периферије **PER0**. Том приликом ће у "semafor" за **PER0** бити уписана вредност 0, па ће се по повратку из прекидне рутине на петљу са лабелом Wait0 утврдити да "semafor" за **PER0** има вредност 0. Због тога ће се сада изаћи из петље са лабелом Wait0 и прећи на други део програма у коме се реализује обрада унетих блокова података. Други случај се јавља када је периферија

PER1 спорија од периферије **PER0**, па ће се унос података из периферије **PER0** завршити пре од уноса података из периферије **PER1** и прво преласком у прекидну рутину за **PER0** ради уноса последњег податка "semafor" за **PER0** поставити на вредност 0, па тек онда у главном програму доћи на петљу са лабелом Wait0. Због тога ће се по завршетку уноса блока података из периферије **PER1** и преласком на петљу са лабелом Wait0 утврдити да "semafor" за **PER0** има вредност 0, па се неће чекати у петљи, већ ће се одмах прећи на други део програма у коме се реализује обрада унетих блокова података.

Други део у коме се врши обрада се реализује позивом на процесуру `Obrada`. У процедури `Obrada` се најпре меморијске локације MemCnt2, MemAdr0, MemAdr1 и MemAdr2 иницијализују вредностима 100h, A000h, B000h и C000h које представљају величину блокова података над којима треба извршити обраду и почетне адресе два блока података над којима треба извршити обраду и блока у који треба уписати резултат, респективно. Потом се у петљи Loop из меморије читају парови података из меморијских локација од адресе A000h до адресе A0FFh и од адресе B000h до адресе B0FFh и сума уписује у меморијске локације од адресе C000h до адресе C0FFh, инкрементирају садржаји меморијских локација MemAdr0, MemAdr1 и MemAdr2 и декрементира садржај меморијске локације MemCnt2 и на крају у зависности од тога да ли је садржај меморијске локације MemCnt2 после декрементирања још увек различит од нуле или је постао нула или остаје у петљи Loop или излази из петље Loop, респективно. На крају се враћа у главни програм.

Трећи део програма у коме се са контролером периферије **KPER2** и DMA контролером **DMA2** у режиму преноса циклус по циклус из меморије у периферију **PER2** шаље блок података има два дела и то један део који се извршава у главном програму и други део који се извршава у прекидној рутини за **DMA2**. У оквиру дела програма који се извршава у главном програму врши се иницијализација преноса из меморије у периферију, постављање "semafor"-а за **DMA2** на 1, стартовање контролера периферије **KPER2** и DMA контролера **DMA2** и чекање у петљи са лабелом Wait2 на завршетак слања блока података из меморије у периферију **PER2**. У оквиру дела програма који се извршава у прекидној рутини за **DMA2** врши се заустављање контролера периферије **KPER2** и DMA контролера **DMA2** и постављање "semafor"-а за **DMA2** на 0.

У оквиру иницијализације преноса из меморије у периферију **PER2** најпре се у регистар *Count* **DMA2** уписује вредност 100h која представља величину блокова података које треба пренети, а затим у регистар *AdrSrc* **DMA2** уписују вредности C000h која представља почетну адресу дела меморије из кога треба пренети блок података у периферију **PER2**. У оквиру постављања "semafor"-а за **DMA2** на 1, у меморијску локацију MemSem2 се уписује вредност 1. У оквиру стартовања контролера периферије **KPER2** и DMA контролера **DMA2** у управљачки регистар *Control* **KPER2** се уписује вредност 8010h чиме се контролер периферије **KPER2** стартује (*start*=1) за рад у режиму излаза (*u/i*=1) без генерисања прекида (*enable*=0) и у управљачки регистар *Control* **DMA2** се уписује вредност 8011h чиме се DMA контролер **DMA2** стартује (*start*=1) за рад у режиму излаза (*u/i*=1, *mem/per*=0) и то циклус по циклус (*burst*=0) са генерисањем прекида (*enable*=1).

Од овог тренутка процесор и контролер периферије **KPER2** и DMA контролер **DMA2** почињу да раде паралелно. Док DMA контролер **DMA2** и контролер периферије **KPER2** хардверски реализује слање блока података из меморије у периферију **PER2**, процесор у главном програму долази на петљу са лабелом Wait2 и у њој утврђује да "semafor" за **DMA2** још увек има вредност 1. Због тога ће се најпре у петљи са лабелом Wait2 чекати да **DMA2** по завршетку слање блока података из меморије у периферију

PER2 генерише прекид и изазове прелазак на прекидну рутину за **DMA2** у којој ће у "semafor" за **DMA2** бити уписана вредност 0, а затим ће се по повратку из прекидне рутине на петљу са лабелом Wait0 утврдити да "semafor" за **DMA2** има вредност 0, па ће се изаћи из петље са лабелом Wait2 и продужити са извршавањем главног програма.

Дискусија:

1. DMA контролер **DMA2** може да ради и у режиму преноса цео блок. Разлика у односу на објашњени режим преноса циклус по циклус је само у томе да се сигнал захтева за коришћење магистрале hreq држи на вредности 1 све време док траје пренос блока података, па стога и сигнал дозволе коришћења магистрале hack има вредност 1 све време док траје пренос блока података. Сигнал захтева за коришћење магистрале hreq и сигнал дозволе коришћења магистрале hack се постављају на вредности 0 тек када се пренесе задњи податак из блока података. Ефекат режима преноса цео блок на рачунарски систем је у томе да нико не може да користи магистралу док DMA контролер **DMA2** не заврши пренос целог блока података. Структура програма је, међутим, иста без обзира на то да ли DMA контролер **DMA2** ради у режиму циклус п циклус или у режиму цео блок.

2. Функционисање контролера периферије **KPER2** и DMA контролера **DMA2** као и коришћење аодредишног дресног регистра одредишта *AdrDst* при преносу блока података из периферије у меморију објашњено је у задатку 1.6, док је функционисање DMA контролера **DMA2** као и коришћење изворишног и одредишног адресног регистра *AdrSrc* и *AdrDst* при преносу блока података из меморије у меморију објашњено у задатку 1.8.

3. Рачунарски систем се може и тако конфигурисати да се периферија **PER2** повеже на магистралу рачунарског система са DMA контролером периферије **DMAPER2** (слика 15), а да се за периферије **PER0** и **PER1** задржи повезивање на магистралу рачунарског система са контролерима периферија **KPER0** и **KPER1** (слика 1).

Тражени програм је приказан на слици 20. Овај програм је веома сличан програму са слике 19. Добијен је тако што су у програму са слике 19 избачене инструкције главног програма

```
;startovanje kontrolera periferije KPER2
     LOAD   #8010h    ;upis režima rada i startovanja (u/i=1,
     OUT    FF20h     ;start=1,enable=0)preko ACC u reg. Control KPER2
```

које се односе на стартовање контролера периферије **KPER2** и инструкције прекидне рутине **DMA2**

```
;zaustavljanje kontrolera periferije KPER2
     LOAD   #0        ;upis režima zaustavljanja (u/i=0,
     OUT FF20h        ;start=0,enable=0)preko ACC u reg. Control KPER2
```

које се односе на заустављање контролера периферије **KPER2** и уместо ознака DMA контролера **DMA2** стављене су ознаке DMA контролера периферије **DMAPER2**. Ради једноставнијег писања програма узето је да регистри DMA контролера периферије **DMAPER2** (слика 15) имају исте адресе и структуру као и регистри DMA контролера **DMA2** (слика 11). Адресе и структура регистара су дати на сликама 12 и 13.

```
;glavni program
;unos blokova podataka iz periferije PER0 u mem sa KPER0 sa prekidom i
;iz periferije PER1 u mem sa KPER1 sa ispitivanjem bita spremnosti ready
  ;inicijalizacija prenosa iz PER0 u mem sa KPER0 i iz PER1 u mem sa KPER1
    ;veličina blokova podataka
      LOAD   #100h     ;upis vel bloka podat za PER0 i PER1 preko ACC u
      STORE  MemCnt0   ;mem. lok. na adr. MemCnt0 i
      STORE  MemCnt1   ;mem. lok. na adr. MemCnt1
    ;početne adrese blokova podataka
      LOAD   #A000h    ;upis adr. bloka podataka za PER0 preko ACC u
      STORE  MemAdr0   ;mem. lok. na adr. MemAdr0
      LOAD   #B000h    ;upis adr. bloka podataka za PER1 preko ACC u
      STORE  MemAdr1   ;mem. lok. na adr. MemAdr1
  ;postavljanje "semafor"-a za PER0 na 1
      LOAD   #1h       ;upis vrednosti 1 preko ACC u
      STORE  MemSem0   ;mem. lok. na adr. MemSem0
  ;startovanje kontrolera periferije KPER0
      LOAD   #0011h    ;upis režima rada i startovanja (u/i=0,
      OUT    FF00h     ;start=1,enable=1)preko ACC u reg. Control KPER0
  ;startovanje kontrolera periferije KPER1
      LOAD   #0010h    ;upis režima rada i startovanja (u/i=0,
      OUT    FF10h     ;start=1,enable=0)preko ACC u reg. Control KPER1
  ;unos blokova podataka
    ;provera da li podatak može da se prenese iz registra Data KPER1
Loop1: IN     FF11h     ;upis sadrž. reg. Status KPER1 u ACC
       AND    #8000h    ;ispitivanje bita ready
       JZ     Loop1     ;povratak na Loop1 ukoliko je ready 0
    ;prenos podatka iz registra Data KPER1 u memorijsku lokaciju
       IN     FF12h     ;upis sadrž. reg. Data KPER1 preko ACC u mem. lok.
       STORE  (MemAdr1) ;na adresi datoj sadrž. mem. lok. na adr. MemAdr1
    ;inkrementiranje sadržaja memorijske lokacije MemAdr1
       INC    MemAdr1   ;inkrement sadrž. mem.lok. sa adr. MemAdr1
    ;dekrementiranje sadržaja memorijske lokacije MemCnt1
       DEC    MemCnt1   ;dekrement sadrž. mem.lok. sa adr. MemCnt1
    ;provera da li je unet poslednji podatak iz PER1
       JNZ    Loop1     ;povratak na Loop1 ukoliko nije poslednji podatak
  ;zaustavljanje kontrolera periferije KPER1
      LOAD   #0        ;upis režima zaustavljanja (u/i=0,
      OUT    FF10h     ;start=0, enable=0) preko ACC u reg. Control KPER1
  ;čekanje na završetak unosa bloka podataka iz periferiju PER0 u memoriju
  ;proverom vrednosti "semafor"-a za PER0
Wait0: LOAD   MemSem0   ;upis sadrž. mem.lok. sa adr. MemSem0 u ACC
       CMP    #0        ;ispitivanje da li je "semafor" postao 0
       JNZ    Wait0     ;povratak na Wait0 ukoliko "semafor" nije 0
;obrada
       JSR    Obrada    ;obrada blokova od po 100h podat u mem lokacijama
  ;od adrese A000h do adrese A0FFh i od adrese B000h do adrese B0FFh
  ;i upis rezultata u mem lokacije od adrese C000h do adrese C0FFh
```

```
;slanje bloka podataka iz memorije u periferiju PER2 sa DMAPER2
  ;inicijalizacija prenosa iz memorije u periferiju PER2 sa DMAPER2
    ;veličina bloka podataka
      LOAD   #100h     ;upis veličine bloka podataka za PER2 preko ACC u
      OUT    FF2Bh     ;u reg. Count DMAPER2
    ;početna adresa bloka podataka
      LOAD   #C000h    ;upis adr. bloka podataka za PER2 preko ACC u
      STORE  FF2Ch     ;u reg. AdrSrc DMAPER2
  ;postavljanje "semafor"-a za DMAPER2 na 1
       LOAD   #1h      ;upis vrednosti 1 preko ACC u
       STORE  MemSem2  ;mem. lok. na adr. MemSem2
  ;startovanje kontrolera DMAPER2
       LOAD   #8011h   ;upis režima rada i start (u/i=1,mem/per=0,start=1,
       OUT    FF28h    ;burst=0,enable=1)preko ACC u reg. Control DMAPER2
  ;čekanje na završetak slanja bloka podataka iz memorije u periferiju
  ;PER2 sa DMAPER2 proverom vrednosti "semafor"-a za DMAPER2
Wait2: LOAD   MemSem2  ;upis sadrž. mem.lok. sa adr. MemSem2 u ACC
       CMP    #0       ;ispitivanje da li je "semafor" postao 0
       JNZ    Wait2    ;povratak na Wait2 ukoliko "semafor" nije 0
. . .
;prekidna rutina periferije PER0
;unos bloka podataka iz periferiju PER0 u memoriju sa KPER0
PER0:  PUSHA            ;akumulator ACC na stek
  ;unos podatka iz registra Data KPER0 u memorijsku lokaciju
       IN FF02h         ;upis sadrž. reg. Data KPER0 preko ACC u mem. lok.
       STORE  (MemAdr0) ;na adr. date sadrž. mem. lok. na adr. MemAdr0
  ;inkrementiranje sadržaja memorijske lokacije MemAdr0
       INC    MemAdr0   ;inkr sadrž. mem.lok. sa adr. MemAdr0
  ;dekrementiranje sadržaja memorijske lokacije MemCnt0
       DEC    MemCnt    ;dekr sadrž. mem.lok. sa adr. MemCnt0
  ;provera da li je prenet poslednji podatak
       JNZ    Back0     ;prelaz na Back0 ukoliko nije poslednji podatak
  ;zaustavljanje kontrolera periferije KPER0
       LOAD   #0        ;upis režima zaustavljanja (start=0, u/i=0,
       OUT FF10h        ;enable=0)preko ACC u reg. Control KPER0
  ;postavljanje "semafor"-a za PER0 na 0
                        ;upis vrednosti 0 preko ACC u
       STORE  MemSem0   ;mem. lok. na adr. MemSem0
  ;izlazak iz prekidne rutine
Back0: POPA             ;restauracija akumulatora ACC sa steka
       RTI              ;povratak iz prekidne rutine
```

```
;Procedura Obrada u kojoj se izračunava C(i)=A(i)+B(i), gde je i=0,…,FFh
;Elementi niza A(i), gde je i=0,…,FFh, su na adresama A000h do A0FFh,
;elementi niza B(i), gde je i=0,…,FFh, su na adresama B000h do B0FFh, a
;elementi niza sume C(i), gde je i=0,…,FFh, se smešt na adr C000h do C0FFh
Obrada:  LOAD   #100h     ;upis vel. blok. podat. za A(i), B(i) i C(i)
         STORE  MemCnt2   ;u ACC pa u mem. lok. na adr. MemCnt2
         LOAD   #A000h    ;upis adr. bloka podataka A(i)preko ACC u
         STORE  MemAdr0   ;mem. lok. na adr. MemAdr0
         LOAD   #B000h    ;upis adr. bloka podataka B(i)preko ACC u
         STORE  MemAdr1   ;mem. lok. na adr. MemAdr1
         LOAD   #C000h    ;upis adr. bloka podataka C(i)preko ACC u
         STORE  MemAdr1   ;mem. lok. na adr. MemAdr2
Loop:    LOAD   (MemAdr0) ;upis A(i) iz mem lok sa adrese date sadr
                          ;mem lok sa adrese MemAdr0 u ACC
         ADD    (MemAdr1) ;suma ACC i B(i) iz mem lok sa adrese date sadr
                          ;mem lok sa adrese MemAdr1 u ACC
         STORE  (MemAdr2) ;sadržaj ACC u C(i) u mem lok na adr datoj sadr
                          ;mem lok na adresi MemAdr2
         INC    MemAdr0   ;inkrementiranje sadr mem lok MemAdr0
         INC    MemAdr1   ;inkrementiranje sadr mem lok MemAdr1
         INC    MemAdr2   ;inkrementiranje sadr mem lok MemAdr2
         DEC    MemCnt2   ;dekrementiranje sadr mem lok MemCnt2
  ;provera da li je poslednja suma
         JNZ    Loop      ;povratak na Loop ukoliko nije poslednja suma
         RTS              ;povratak u glavni program

;prekidna rutina za DMAPER2
DMAPER2: PUSHA             ;akumulator ACC na stek
  ;zaustavljanje kontrolera DMAPER2
                           ;upis režima zaustavjanja (u/i=0,mem/per=0,start=0,
         OUT FF28h         ;burst=0,enable=0)preko ACC u reg. Control DMAPER2
  ;postavljanje "semafor"-a za DMAPER2 na 0
                           ;upis vrednosti 0 preko ACC u
         STORE  MemSem2    ;mem. lok. na adr. MemSem2
  ;izlazak iz prekidne rutine
         POPA              ;restauracija akumulatora ACC sa steka
         RTI               ;povratak iz prekidne rutine
. . .
```

Слика 20 Програм

За слање података из меморије у периферију **PER2** се у овом случају користи DMA контролер периферије **DMAPER2** у коме су интегрисане функције контролера периферије **KPER2** и DMA контролера **DMA2**. Слање блока података из из меморије у периферију **PER2** започиње тако што се извршавањем одговарајућег програма у управљачки регистар *Control* **DMAPER2** упише садржај којим се DMA контролер периферије **DMAPER2** стартује (*start*=1) за рад у режиму излаза (*u/i*=1, *mem/per*=0) и то циклус по циклус (*burst*=0) са генерисања прекида (*enable*=1). Контролер **DMAPER2** на сличан начин као и контролер **DMA2** хардверски генерише адресе изворишних меморијских локација из којих треба пренети податаке у периферију **PER2**, реализује читање података из изворишних меморијских локација и упис у регистар *Data* **DMAPER2** и води евиденцију колико још података треба пренети из меморије у периферију **PER2** и на исти начин као и контролер **KPER2** преноси податак из регистра података *Data* **DMAPER2** у периферију **PER2**. Контролер **DMAPER2** као и контролер **DMA2** има изворишни адресни регистар *AdrSrc* и регистар величине блока података *Count*. Стога је пре старовања контролера **DMAPER2** потребно у оквиру иницијализације преноса блока података из меморије у периферије **PER2**, почетну адресу меморијске локације од које треба пренети блок података и величину блока података уписати у регистре *AdrSrc* и *Count* **DMAPER2**. Такође је потребно као индикацију да је пренос података у току у "semafor" за **DMAPER2** уписати вредност 1. Контролер **DMAPER2** мора после сваког пренетог податка из меморијске локације у регистар *Data* **DMAPER2** да изврши инкрементирање садржаја регистра *AdrSrc* **DMAPER2** и декрементирање садржаја регистра *Count* **DMAPER2** и изврши проверу да ли је садржај регистра *Count* **DMAPER2** декрементирањем постао 0. Уколико садржај регистра *Count* **DMAPER2** није нула, контролер **DMAPER2** продужава са преносом података из меморије у периферију **PER2** тако што чита следећи податак из меморијске локације и преноси у регистар података *Data* **DMAPER2** и из њега у периферију **PER2**, док у супротном случају завршава са преносом података из меморије у периферију **PER2** и у бит *end* статусног регистра *Status* **DMAPER2** уписује вредност 1 и генерише прекид. У прекидној рутини контролера **DMAPER2** се извршавањем одговарајућег програма најпре "semafor" за **DMAPER2** поставља на 0 а затим у управљачки регистар *Control* **DMAPER2** уписује садржај којим се контролер **DMAPER2** зауставља (*start*=0).

Пребацивање податка из меморијске локације у регистар података *Data* **DMAPER2** се незнатно разликује од случаја када се користе контролери **KPER0** и **DMA0**. Контролер **DMAPER2** најпре поставља на 1 сигнал hreq којим, у зависности од тога како је реализована арбитрација, од процесора или арбитратора магистрале тражи дозволу за коришћење магистрале. Тек када вредношћу 1 сигнала hack добије дозволу за коришћење магистрале, контролер **DMAPER2** креће са реализацијом циклуса читања. Том приликом контролер **DMAPER2** најпре отвара бафере са три стања и на адресне линије магистрале ABUS пушта садржај изворишног адресног регистра *AdrSrc* **DMAPER2** и затим стартује операцију читања постављањем управљачког сигнала читања RDBUS на вредност 1. Пошто је узето да је магистрала асинхрона, контролер **DMAPER2** држи садржај на адресним линијама магистрале ABUS и управљачки сигнал читања RDBUS на вредности 1 све време док читања из меморије траје. Када меморија по завршетку читања отвори своје бафере са три стања и на линије података магистрале DBUS пусти прочитани садржај и постави управљачки сигнал завршетка циклуса на магистрали FCBUS на вредност 1, DMA контролер **DMAPER2** садржај са линија података магистрале DBUS уписује у регистар података *Data* **DMAPER2.** По завршетку уписа података у регистар *Data* **DMAPER2** DMA контролер **DMAPER2** постављањем сигнала RDBUS на вредност 0 шаље индикацију меморији да нема

потреба да меморија више држи садржај на линијама података магистрале DBUS. На вредност 0 сигнала RDBUS меморија затвара бафере са три стања и линије података магистрале DBUS ставља у стање високе импедансе, а сигнал FCBUS поставља на вредност 0. На вредност 0 сигнала FCBUS DMA контролер **DMAPER2** затвара бафере са три стања и адресне линије магистрале ABUS и управљачку линију читања RDBUS ставља у стање високе импедансе и започиње пренос података из регистра података *Data* **DMAPER2** у периферију **PER2**. Паралелно са преносом података у периферију **PER2** контролер **DMAPER2** постављањем на сигнала hreq на вредност 0 укида захтев за коришћење магистрале и као одговор процесор или арбитратор магистрале му вредношћу 0 сигнала hack укидају дозволу за коришћење магистрале. Поред тога, контролер **DMAPER2** инкрементира садржај регистра *AdrSrc* **DMAPER2** и декрементира садржај регистра *Count* **DMAPER2** и врши проверу да ли је садржај регистра *Count* **DMAPER2** декрементирањем постао 0. Уколико садржај регистра *Count* **DMAPER2** није нула, контролер **DMAPER2** најпре сачека да се заврши слање текућег података из регистра података *Data* **DMAPER2** у периферију **PER2** и затим започиње читање следећег података из меморије и пренос у регистар података *Data* **DMAPER2**. Међутим, уколико је садржај регистра *Count* **DMAPER2** нула, контролер **DMAPER2** уписује вредност 1 у бит *end* статусног регистра *Status* **DMAPER2** и генерише прекид. У прекидној рутини контролера **DMAPER2** се извршавањем одговарајућег програма најпре "semafor" за **DMAPER2** постави на 0 а затим у управљачки регистар *Control* **DMAPER2** упише садржај којим се контролер **DMAPER2** зауставља (*start*=0).

DMA контролер периферије **DMAPER2** може да ради и у режиму преноса цео блок и то на исти начин као и DMA контролер **DMA** (**Дискусија** под **1**).

Функционисање DMA контролера периферије **DMAPER2** као и коришћење одредишног адресног регистра *AdrDst* при преносу блока података из периферије у меморију објашњено је у задатку 1.6, док је функционисање DMA контролера периферије **DMAPER2** као и коришћење изворишног и одредишног адресног регистра *AdrSrc* и *AdrDst* при преносу блока података из меморије у меморију објашњено у задатку 1.8.

1.8 ЗАДАТАК

Дат је рачунарски систем који чине процесор, меморија и контролер периферије **KPER0** са периферијом **PER0**, контролер периферије **KPER1** и DMA контролер **DMA1** са периферијом **PER1**, контролер периферије **KPER2** и DMA контролер **DMA2** са периферијом **PER2** и контролер периферије **KPER3** са периферијом **PER3** повезани магистралом.

Процесор је са једноадресним форматом инструкција. Од програмски доступних регистара постоје 16 битни акумулатор ACC, указивач на врх стека SP, указивач на табелу (IV табела) са адресама прекидних рутина IVTP и програмска статусна реч PSW. Типови података који се користе су 16 битне целобројне величине са знаком и без знака. Приликом извршавања инструкције преноса у акумулатор и аритметичких, логичких и померачких инструкција постављају се индикатори N, Z, C и V програмске статусне речи PSW. Механизам прекида је векторисан. При прекиду хардверски се на стеку чувају програмски бројач PC и програмска статусна реч PSW. Бројеви улаза у IV табелу су фиксни за сваку периферију. Меморијски и улазно-излазни адресни простори су раздвојени.

Меморијски адресни простор је величине 2^{16} меморијских локација, при чему је ширина меморијских локација 16 битова.

Улазно-излазни адресни простор је величине 2^{16} регистара, при чему је ширина регистара 16 битова. Регистри контролера периферија **KPER0**, **KPER1**, **KPER2** и **KPER3** и регистри DMA контролера **DMA1** и **DMA2** се налазе у улазно-излазном адресном простору.

Контролери периферија **KPER0**, **KPER1**, **KPER2** и **KPER3** имају управљачке регистре (*Control*), статусне регистре (*Status*) и регистре података (*Data*) и то редом на адресама FF00h, FF01h и FF02h за контролер периферије **KPER0**, FF10h, FF11h и FF12h за контролер периферије **KPER1**, FF20h, FF21h и FF22h за контролер периферије **KPER2** и FF30h, FF31h и FF32h за контролер периферије **KPER3**. У регистрима контролера периферија **KPER0**, **KPER1**, **KPER2** и **KPER3** највиши бит је означен са 15, а најнижи са 0. У управљачким регистрима *Control* бит 15 је бит *u/i* којим се задаје смер преноса са периферијом (0—улаз, 1—излаз), бит 4 је бит *start* којим се покреће извршавање операције (0—заустављен, 1—стартован) и бит 0 је бит *enable* којим се дозвољава прекид (0—маскиран, 1—дозвољен). У статусним регистрима *Status* бит 15 је бит спремности *ready* (0—није спреман, 1—спреман).

DMA контролери **DMA1** и **DMA2** имају управљачки регистар (*Control*), статусни регистар (*Status*), регистар податка (*Data*), регистар величине блока података (*Count*), изворишни адресни регистар (*AdrSrc*) и одредишни адресни регистар (*AdrDst*) и то редом на адресама FF18h, FF19h, FF1Ah, FF1Bh, FF1Ch и FF1Dh за **DMA1** и адресама FF28h, FF29h, FF2Ah, FF2Bh, FF2Ch и FF2Dh за **DMA2**, респективно. У регистрима DMA контролера **DMA1** и **DMA2** највиши бит је означен са 15, а најнижи са 0. У управљачком регистру *Control* бит 15 је бит *u/i* којим се задаје смер преноса са периферијом (0—улаз, 1—излаз), бит 14 је бит *mem/per* којим се задаје пренос између делова меморије или између периферије и меморије (0—периферија/меморија, 1—меморија/меморија), бит 4 је бит *start* којим се покреће извршавање операције (0—заустављен, 1—стартован), бит 1 је бит *burst* којим се задаје режим преноса (0—циклус по циклус, 1—цео блок) и бит 0 је бит *enable* којим се дозвољава прекид (0—маскиран, 1—дозвољен), при чему је вредност бита 15 битна само уколико је вредношћу 0 бита 14 задат пренос између периферије и меморије. У статусном регистру *Status* бит 15 је бит завршетка преноса *end* (0—пренос у току, 1—пренос завршен).

Адресне линије и линије података магистрале рачунара су широке по 16 битова.

Написати главни програм и одговарајуће прекидне рутине којима се учитава низ A(i), где је i = 0, …, FFh, из **PER0** у меморијски блок који почиње од адресе A000h, затим копира низ A(i), где је i = 0, …, FFh, из меморијског блока који почиње од адресе A000h у низ B(i), где је i = 0, …, FFh, у меморијски блок који почиње од адресе B000h и на крају упоредо шаљу низ A(i), где је i = 0, …, FFh, из меморијског блока који почиње од адресе A000h у **PER2** и низ B(i), где је i = 0, …, FFh, из меморијски блок који почиње од адресе B000h у **PER3**. Улаз из периферије **PER0** реализовати са контролером периферије **KPER0** техником програмираног улаза са испитивањем спремности, копирање низа A(i) у низ B(i), где је i = 0, …, FFh, реализовати са DMA контролером **DMA1** у режиму преноса цикилус по циклус, излаз у периферију **PER2** реализовати са контролером периферије **KPER2** и DMA контролером **DMA2** у режиму преноса циклус по циклус и излаз у периферију **PER3** реализовати са контролером периферије **KPER3** техником програмираног излаза са прекидом.

Решење:

Адресе и структура регистара контролера **KPER0**, **KPER1, DMA1**, **KPER2**, **DMA2** и **KPER3** су дати на сликама 21 и 22, респективно.

KPER0 - Kontroler periferije **PER0**

Registar	Control	Status	Data
Adresa	FF00h	FF01h	FF02h

KPER1 - Kontroler periferije **PER1**

Registar	Control	Status	Data
Adresa	FF10h	FF11h	FF12h

DMA1 –DMA kontroler

Registar	Control	Status	Data	Count	AdrSrc	AdrDst
Adresa	FF18h	FF19h	FF1Ah	FF1Bh	FF1Ch	FF1Dh

KPER2 - Kontroler periferije **PER2**

Registar	Control	Status	Data
Adresa	FF20h	FF21h	FF22h

DMA2 –DMA kontroler

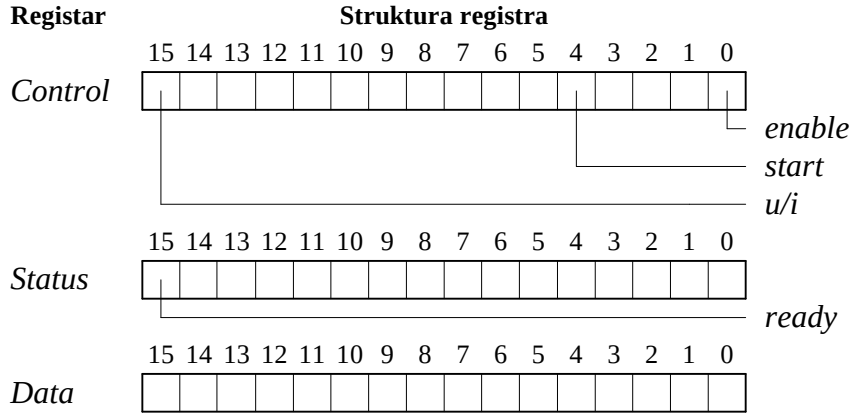
Registar	Control	Status	Data	Count	AdrSrc	AdrDst
Adresa	FF28h	FF29h	FF2Ah	FF2Bh	FF2Ch	FF2Dh

KPER3 - Kontroler periferije **PER3**

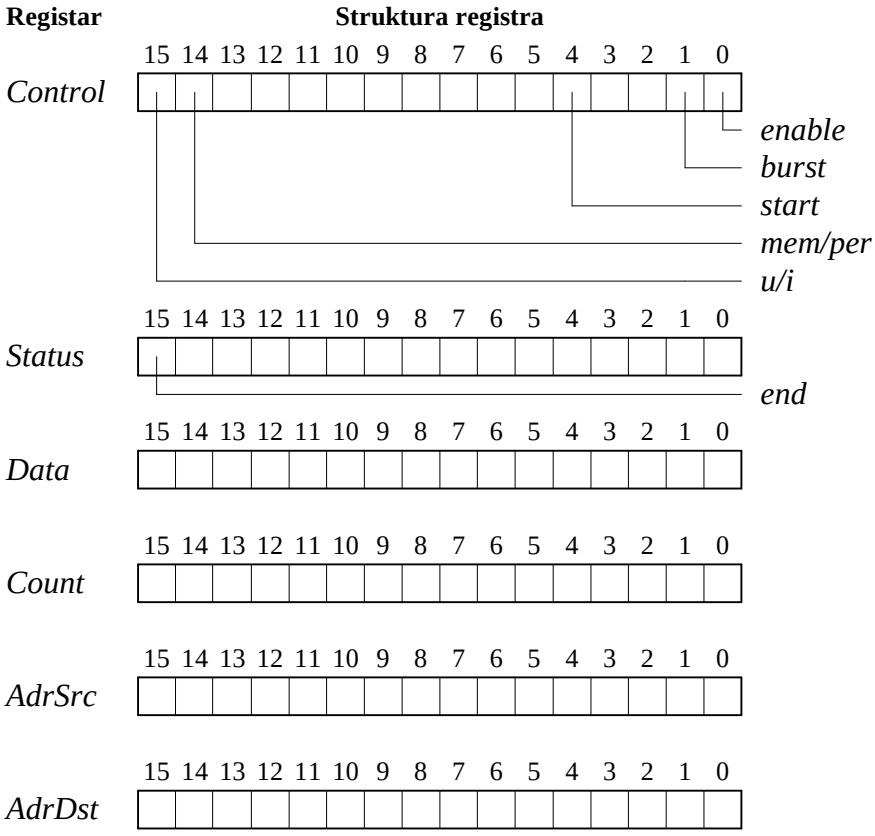
Registar	Control	Status	Data
Adresa	FF30h	FF31h	FF32h

Слика 21 Адресе регистара контролера

Registri kontrolera periferija **KPER**



Registri DMA kontrolera **DMA**



Слика 22 Структура регистара контролера

Периферије **PER0** и **PER3** се повезују на магистралу рачунарског система са контролерима периферија **KPER0** и **KPER3** на начин приказан на слици 1, док се периферије **PER1** и **PER2** повезује на магистралу рачунарског система са контролерима периферија **KPER1** и **KPER2** и DMA контролерима **DMA1** и **DMA 2** на начин приказан на слици 11.

Тражени програм је приказан на слици 23.

```
;glavni program
;unos bloka podataka iz periferije PER0 u memoriju sa KPER0  sa
;ispitivanjem bita spremnosti ready
  ;inicijalizacija prenosa iz PER0 u memoriju sa KPER0
    ;veličina bloka podataka
      LOAD   #100h     ;upis vel bloka podat za PER0 preko ACC u
      STORE  MemCnt0   ;mem. lok. na adr. MemCnt0
    ;početna adresa bloka podataka
      LOAD   #A000h    ;upis adr. bloka podataka za PER0 preko ACC u
      STORE  MemAdr0   ;mem. lok. na adr. MemAdr0
  ;startovanje kontrolera periferija KPER0
      LOAD   #0010h    ;upis režima rada i startovanja (u/i=0,
      OUT    FF00h     ;start=1,enable=0)preko ACC u reg. Control KPER0
  ;unos bloka podataka
    ;provera da li podatak može da se prenese iz registra Data KPER0
Loop0: IN     FF01h     ;upis sadrž. reg. Status KPER0 u ACC
       AND    #8000h    ;ispitivanje bita ready
       JZ     Loop0     ;povratak na Loop0 ukoliko je ready 0
    ;prenos podatka iz registra Data KPER0 u memorijsku lokaciju
      IN     FF02h     ;upis sadrž. reg. Data KPER0 preko ACC u mem. lok.
      STORE  (MemAdr0) ;na adresi datoj sadrž. mem. lok. na adr. MemAdr0
    ;inkrementiranje sadržaja memorijske lokacije MemAdr0
      INC    MemAdr0   ;inkrement sadrž. mem.lok. sa adr. MemAdr0
    ;dekrementiranje sadržaja memorijske lokacije MemCnt0
      DEC    MemCnt0   ;dekrement sadrž. mem.lok. sa adr. MemCnt0
    ;provera da li je unet poslednji podatak iz PER0
      JNZ    Loop0     ;povratak na Loop0 ukoliko nije poslednji podatak
  ;zaustavljanje kontrolera periferije KPER0
      LOAD   #0        ;upis režima zaustavljanja (u/i=0,
      OUT    FF00h     ;start=0, enable=0) preko ACC u reg. Control KPER0
;obrada
       JSR    Obrada    ;kopiranje bloka od 100h podataka iz memorijskih
  ;lokacija od adrese A000h do adrese A0FFh u memorijske lokacije od adrese
  ;B000h do adrese B0FFh
```

```
;slanje blokova podataka iz memorije u periferiju PER2 sa KPER2 i DMA2 i
;iz memorije u periferiju PER3 sa KPER3 sa prekidom
  ;inicijalizacija prenosa iz memorije u PER2 sa KPER2 i DMA2 i
  ;iz memorije u PER3 sa KPER3
    ;veličine blokova podataka
      LOAD   #100h     ;upis vel bloka podat za PER2 i PER3 preko ACC u
      OUT    FF2Bh     ;reg. Count DMA2 i
      STORE  MemCnt3   ;mem. lok. na adr. MemCnt3
    ;početne adrese blokova podataka
      LOAD   #A000h    ;upis adr. bloka podataka za PER2
      OUT    FF2Ch     ;preko ACC u u reg. AdrSrc DMA2
      LOAD   #B000h    ;upis adr. bloka podataka za PER3 preko ACC u
      STORE  MemAdr3   ;mem. lok. na adr. MemAdr3
  ;postavljanje "semafor"-a za DMA2 na 1
      LOAD   #1h       ;upis vrednosti 1 preko ACC u
      STORE  MemSem2   ;mem. lok. na adr. MemSem2
  ;postavljanje "semafor"-a za PER3 na 1
                       ;upis vrednosti 1 preko ACC u
      STORE  MemSem3   ;mem. lok. na adr. MemSem3
  ;startovanje kontrolera periferije KPER2
      LOAD   #8010h    ;upis režima rada i startovanja (u/i=1,
      OUT    FF20h     ;start=1,enable=0)preko ACC u reg. Control KPER2
  ;startovanje kontrolera DMA2
      LOAD   #8011h    ;upis rež rada i start (u/i=1, mem/per=0, start=1,
      OUT    FF28h     ;burst=0,enable=1)preko ACC u reg. Control DMA2
  ;startovanje kontrolera periferije KPER3
      LOAD   #8011h    ;upis režima rada i startovanja (u/i=1,
      OUT    FF30h     ;start=1,enable=1)preko ACC u reg. Control KPER3
  ;čekanje na završetak slanja bloka podataka iz memorije u periferiju
  ;PER2 sa KPER2 i DMA2 proverom vrednosti "semafor"-a za DMA2
Wait2: LOAD   MemSem2   ;upis sadrž. mem.lok. sa adr. MemSem2 u ACC
       CMP    #0        ;ispitivanje da li je "semafor" postao 0
       JNZ    Wait2     ;povratak na Wait2 ukoliko "semafor" nije 0
  ;čekanje na završetak slanja bloka podataka iz memorije u periferiju
  ;PER3 sa KPER3 proverom vrednosti "semafor"-a za PER3
Wait3: LOAD   MemSem3   ;upis sadrž. mem.lok. sa adr. MemSem3 u ACC
       CMP    #0        ;ispitivanje da li je "semafor" postao 0
       JNZ    Wait3     ;povratak na Wait3 ukoliko "semafor" nije 0
. . .
```

```
;Procedura Obrada koja kopira niz A(i), gde je i = 0, …, FFh, iz
;memorijskog bloka koji počinje od adrese A000h u niz B(i), gde je
;i = 0, …, FFh, u memorijski blok koji počinje od adrese B000h
;korišćenjem DMA1
  ;inicijalizacija prenosa iz memorije u memoriju preko DMA1
    ;veličina bloka podataka
Obrada:LOAD   #100h     ;upis veličine bloka podataka za DMA1
       OUT    FF1Bh     ;preko ACC u u reg. Count DMA1
    ;početna adresa izvorišnog bloka podataka
       LOAD   #A000h    ;upis izvorišne adr. bloka podataka za DMA1
       OUT    FF1Ch     ;preko ACC u reg. AdrSrc DMA1
    ;početna adresa odredišnog bloka podataka
       LOAD   #B000h    ;upis odredišne adr. bloka podataka za DMA1
       OUT    FF1Dh     ;preko ACC u reg. AdrDst DMA1
  ;postavljanje "semafor"-a za DMA1 na 1
       LOAD   #1h       ;upis vrednosti 1 preko ACC u
       STORE  MemSem1   ;mem. lok. na adr. MemSem1
  ;startovanje kontrolera DMA1
       LOAD   #4011h    ;upis rež rada i start (u/i=0, mem/per=1, start=1,
       OUT    FF18h     ;burst=0,enable=1)preko ACC u reg. Control DMA1
  ;čekanje na završetak slanja bloka podataka iz memorije u memoriju
  ;preko DMA1 proverom vrednosti "semafor"-a za DMA1
Wait1: LOAD   MemSem1   ;upis sadrž. mem.lok. sa adr. MemSem1 u ACC
       CMP    #0        ;ispitivanje da li je "semafor" postao 0
       JNZ    Wait1     ;povratak na Wait1 ukoliko "semafor" nije 0
  ;izlazak iz procedure
       RTS              ;povratak u glavni program

;prekidna rutina za DMA1
DMA1:  PUSHA            ;akumulator ACC na stek
  ;zaustavljanje kontrolera DMA1
       LOAD   #0        ;upis režima zaust (u/i=0, mem/per=0, start=0,
       OUT FF18h        ;burst=0,enable=0)preko ACC u reg. Control DMA1
  ;postavljanje "semafor"-a za DMA1 na 0
                        ;upis vrednosti 0 preko ACC u
       STORE  MemSem1   ;mem. lok. na adr. MemSem1
  ;izlazak iz prekidne rutine
       POPA             ;restauracija akumulatora ACC sa steka
       RTI              ;povratak iz prekidne rutine
```

```
;prekidna rutina za DMA2
DMA2:  PUSHA           ;akumulator ACC na stek
  ;zaustavljanje kontrolera periferije KPER2
       LOAD   #0       ;upis režima zaustavljanja (u/i=0,
       OUT FF20h       ;start=0,enable=0)preko ACC u reg. Control KPER2
  ;zaustavljanje kontrolera DMA2
                       ;upis režima zaust (u/i=0, mem/per=0, start=0,
       OUT FF28h       ;burst=0,enable=0)preko ACC u reg. Control DMA2
  ;postavljanje "semafor"-a za DMA2 na 0
                       ;upis vrednosti 0 preko ACC u
       STORE  MemSem2  ;mem. lok. na adr. MemSem2
  ;izlazak iz prekidne rutine
       POPA            ;restauracija akumulatora ACC sa steka
       RTI             ;povratak iz prekidne rutine

;prekidna rutina periferije PER3
;slanje bloka podataka iz memorije u periferiju PER3 sa KPER3
PER3:  PUSHA           ;akumulator ACC na stek
  ;prenos podatka iz memorijske lokacije u registar Data KPER3
       LOAD   (MemAdr3);upis sadrž. mem. lok. sa adr. date sadrž. mem.
       OUT    FF32h    ;lok. sa adr. MemAdr3 preko ACC u reg. Data KPER3
  ;inkrementiranje sadržaja memorijske lokacije MemAdr3
       INC    MemAdr3  ;inkr sadrž. mem.lok. sa adr. MemAdr3
  ;dekrementiranje sadržaja memorijske lokacije MemCnt3
       DEC    MemCnt3  ;dekr sadrž. mem.lok. sa adr. MemCnt3
  ;provera da li je prenet poslednji podatak
       JNZ    Back3    ;prelaz na Back3 ukoliko nije poslednji podatak
  ;zaustavljanje kontrolera periferije KPER3
       LOAD   #0       ;upis režima zaustavljanja (start=0, u/i=0,
       OUT    FF30h    ;enable=0)preko ACC u reg. Control KPER3
  ;postavljanje "semafor"-a za PER3 na 0
                       ;upis vrednosti 0 preko ACC u
       STORE  MemSem3  ;mem. lok. na adr. MemSem3
  ;izlazak iz prekidne rutine
Back3: POPA            ;restauracija akumulatora ACC sa steka
       RTI             ;povratak iz prekidne rutine
. . .
```

Слика 23 Програм

За слање података из меморије у меморију се користи само DMA контролер **DMA1**. Пренос блока података из меморије у меморију започиње тако што се извршавањем одговарајућег програма у управљачки регистар *Control* **DMA1** упише садржај којим се DMA контролер **DMA1** стартује (*start*=1) за рад у режиму меморија меморија (*u/i*=0, *mem/per*=1) и то циклус по циклус (*burst*=0) са генерисања прекида (*enable*=1). Приликом преноса блока података из меморије у меморију DMA контролер **DMA1** хардверски генерише адресе изворишних меморијских локација из којих треба читати податаке, реализује читање података из изворишних меморијских локација и упис у регистар *Data* **DMA1**, генерише адресе одредишних меморијских локација у које треба уписивати податаке, реализује уписивање података из регистра *Data* **DMA1** у одредишне меморијске локације и води евиденцију колико још података треба пренети из меморије у меморију. DMA контролер **DMA1** има изворишни адресни регистар *AdrSrc*, одредишни адресни регистар *AdrDst* и регистар величине блока података *Count*. Стога је пре стартовања DMA контролера **DMA1** потребно у оквиру иницијализације преноса блока података из меморије у меморију, почетну адресу меморијске локације од које треба читати блок података, почетну адресу меморијске локације од које треба уписивати блок података и величину блока података уписати у регистре *AdrSrc*, *AdrDst* и *Count* **DMA1**, респективно. Такође је потребно као индикацију да је пренос података у току у "semafor" за **DMA1** уписати вредност 1. DMA контролера **DMA1** мора после сваког пренетог податка из изворишне меморијске локације у одредишну меморијску локацију да изврши инкрементирање садржаја регистара *AdrSrc* **DMA1** и *AdrDst* **DMA1** и декрементирање садржаја регистра *Count* **DMA1** и изврши проверу да ли је садржај регистра *Count* **DMA1** декрементирањем постао 0. Уколико садржај регистра *Count* **DMA1** није нула, DMA контролер **DMA1** продужава са преносом података из меморије у меморију тако што следећи податак преноси најпре из изворишне меморијске локације у регистар податка *Data* **DMA1** а затим из регистра податка *Data* **DMA1** у одредишну меморијску локацију, док у супротном случају завршава са преносом података из меморије у меморију и у бит *end* статусног регистра *Status* **DMA1** уписује вредност 1 и генерише прекид. У прекидној рутини DMA контролера **DMA1** се извршавањем одговарајућег програма најпре "semafor" за **DMA1** поставља на 0 а затим у управљачки регистар *Control* **DMA1** уписује садржај којим се DMA контролер **DMA1** зауставља (*start*=0).

Пребацивање податка из изворишне меморијске локације у регистар податка *Data* **DMA1** се реализује на сличан начин као и када се користи DMA контролер периферије **DMAPER** (задатак 1.7, одељак **Дискусија** под **1**). Контролер **DMA1** најпре поставља на 1 сигнал hreq којим, у зависности од тога како је реализована арбитрација, од процесора или арбитратора магистрале тражи дозволу за коришћење магистрале. Тек када вредношћу 1 сигнала hack добије дозволу за коришћење магистрале, контролер **DMA1** креће са реализацијом циклуса читања. Том приликом контролер **DMA1** најпре отвара бафере са три стања и на адресне линије магистрале ABUS пушта садржај изворишног адресног регистра *AdrSrc* **DMA1** и затим стартује операцију читања постављањем управљачког сигнала читања RDBUS на вредност 1. Пошто је узето да је магистрала асинхрона, контролер **DMA1** држи садржај на адресним линијама магистрале ABUS и управљачки сигнал читања RDBUS на вредности 1 све време док читање из меморије траје. Када меморија по завршетку читања отвори своје бафере са три стања и на линије података магистрале DBUS пусти прочитани садржај и постави управљачки сигнал завршетка циклуса на магистрали FCBUS на вредност 1, контролер **DMA1** уписује садржај са линија података магистрале DBUS у регистар податка *Data* **DMA1.** По завршетку уписа података у регистар *Data* **DMA1,** контролер **DMA1** постављањем сигнала RDBUS на вредност 0 шаље индикацију меморији да нема потреба да меморија

више држи садржај на линијама података магистрале DBUS. На вредност 0 сигнала RDBUS меморија затвара бафере са три стања и линије података магистрале DBUS ставља у стање високе импедансе, а сигнал FCBUS поставља на вредност 0. На вредност 0 сигнала FCBUS контролер **DMA1** затвара бафере са три стања и адресне линије магистрале ABUS и управљачку линију читања RDBUS ставља у стање високе импедансе. Контролер **DMA1** потом постављањем сигнала hreq на вредност 0 укида захтев за коришћење магистрале и као одговор процесор или арбитратор магистрале му вредношћу 0 сигнала hack укидају дозволу за коришћење магистрале.

Пребацивање податка из регистра података *Data* **DMA1** у одредишну меморијску локацију се реализује на сличан начин као и када се користи DMA контролер периферије **DMAPER** (задатак 1.6, одељак **Дискусија** под **1**). Контролер **DMA1** најпре поставља на 1 сигнал hreq којим, у зависности од тога како је реализована арбитрација, од процесора или арбитратора магистрале тражи дозволу за коришћење магистрале. Тек када вредношћу 1 сигнала hack добије дозволу за коришћење магистрале, контролер **DMA1** креће са реализацијом циклуса уписа. Том приликом контролер **DMA1** најпре отвара бафере са три стања и на адресне линије магистрале ABUS и линије података магистрале DBUS пушта садржај одредишног адресног регистра *AdrDst* **DMA1** и регистра података *Data* **DMA1**, респективно, и затим стартује операцију уписа постављањем управљачког сигнала уписа WRBUS на вредност 1. Пошто је узето да је магистрала асинхрона, контролер **DMA1** држи садржај на адресним линијама магистрале ABUS и линијама података магистрале DBUS и управљачки сигнал уписа WRBUS на вредности 1 све време док упис у меморију траје. По завршетку уписа најпре меморија поставља управљачки сигнал завршетка циклуса на магистрали FCBUS на вредност 1, па затим контролер **DMA1** поставља сигнал WRBUS на вредност 0 и на крају меморија поставља сигнал FCBUS на вредност 0. Тиме је циклус уписа завршен, па контролер **DMA1** затвара бафере са три стања и адресне линије магистрале ABUS и линије података магистрале DBUS ставља у стање високе импедансе. Контролер **DMA1** потом постављањем на сигнала hreq на вредност 0 укида захтев за коришћење магистрале и као одговор процесор или арбитратор магистрале му вредношћу 0 сигнала hack укидају дозволу за коришћење магистрале.

Поред тога, контролер **DMA1** инкрементира садржај регистара *AdrSrc* **DMA1** и *AdrDst* **DMA1** и декрементира садржај регистра *Count* **DMA1** и врши проверу да ли је садржај регистра *Count* **DMA1** декрементирањем постао 0. Уколико садржај регистра *Count* **DMA1** није нула, контролер **DMA1** чита следећи податак из изворишне меморијске локације и уписује га у регистар података *Data* **DMA1**, а затим из регистра података *Data* **DMA1** уписује у одредишну меморијску локације. Међутим, уколико је садржај регистра *Count* **DMA1** нула, контролер **DMA1** уписује вредност 1 у бит *end* статусног регистра *Status* **DMA1** и генерише прекид. У прекидној рутини контролера **DMA1** се извршавањем одговарајућег програма најпре "semafor" за **DMA1** постави на 0 а затим у управљачки регистар *Control* **DMA1** упише садржај којим се контролер **DMA1** зауставља (*start*=0).

Програм се састоји из три дела. У првом делу се из периферије **PER0** са контролером периферије **KPER0** техником програмираног улаза са испитивањем бита спремности *ready* статусног регистра *Status* **KPER0** уноси блок од 100h података и смешта у меморијске локације од адресе A000h до адресе A0FFh. У другом делу се позива процесура `Obrada` која са DMA контролером **DMA1** у режиму преноса циклус по циклус копира блок од 100h података из меморијских локација од адресе A000h до адресе A0FFh у меморијске локације од адресе B000h до адресе B0FFh. У трећем делу се упоредо шаљу блок од 100h података прочитаних из меморијских локација од адресе A000h до адресе A0FFh у периферију **PER2** са контролером периферије **KPER2** и DMA

контролером **DMA2** у режиму преноса циклус по циклус и блок од 100h података прочитаних из меморијских локација од адресе B000h до адресе B0FFh и у периферију **PER3** са контролером периферије **KPER3** техником програмираног излаза са прекидом.

Први део програма је сличан као први део програма у задатку 1.1.

Други део програма који се реализује прелазом на процедуру `Obrada` има два дела и то један део који се извршава у процедури `Obrada` и други део који се извршава у прекидној рутини за **DMA1**. У оквиру дела програма који се извршава у процедури `Obrada` врши се иницијализација преноса из једног дела меморије у други део меморије са DMA контролером **DMA1**, постављање "semafor"-а за **DMA1** на 1, стартовање DMA контролера **DMA1** и чекање на завршетак преноса блока података из меморије у меморију. У оквиру дела програма који се извршава у прекидној рутини за **DMA1** врши се заустављање DMA контролера **DMA1** и постављање "semafor"-а за **DMA1** на 0.

У оквиру иницијализације преноса из меморије у меморију најпре се у регистар *Count* **DMA1** уписује вредност 100h која представља величину блока података који треба пренети, затим у регистар *AdrSrc* **DMA1** уписују вредност A000h која представља почетну адресу дела меморије из кога треба читати податке и у регистар *AdrDst* **DMA1** уписују вредност B000h која представља почетну адресу дела меморије у који треба уписивати податке. У оквиру постављања "semafor"-а за **DMA1** на 1, у меморијску локацију MemSem1 се уписује вредност 1 која служи као индикација да је пренос података из меморије у меморију у току и да се не сме из процедуре `Obrada` вратити у главни програм и прећи на извршавање дела главног програма у коме се користе подаци из блока меморије у који се уписују подаци. У оквиру стартовања DMA контролера **DMA1** у управљачки регистар *Control* **DMA1** се уписује вредност 4011h чиме се DMA контролер **DMA1** стартује (*start*=1) за рад у режиму преноса између делова меморије (*u/i*=0, *mem/per*=1) и то циклус по циклус (*burst*=0) са генерисањем прекида (*enable*=1).

Од овог тренутка процесор и DMA контролер **DMA1** почињу да раде паралелно, тако што процесор у процедури `Obrada` прелази на извршавање петље са лабелом Wait1, док DMA контролер **DMA1** хардверски реализује пренос блока података из меморије у меморију. У петљи са лабелом Wait1 ће се утврдити да "semafor" за **DMA1** још увек има вредност 1, па ће се у петљи са лабелом Wait1 чекати да **DMA1** заврши пренос блока података из меморије у меморију, генерише прекид и изазове прелазак на прекидну рутину за **DMA1.** С обзиром на то да ће у прекидној рутини за **DMA1** у "semafor" за **DMA1** бити уписана вредност 0, по повратку из прекидне рутине на петљу са лабелом Wait1 ће се утврдити да "semafor" за **DMA1** има вредност 0, па ће се изаћи из петље са лабелом Wait1 и вратити из процесуре `Obrada` у главни програм.

Трећи део програма у коме се блокови података из меморије упоредо шаљу у периферију **PER2** са контролером периферије **KPER2** и DMA контролером **DMA2** у режиму преноса циклус по циклус и у периферију **PER3** са контролером периферије **KPER3** техником програмираног излаза са прекидом има два дела и то један део који се извршава у главном програму и други део који се извршава у прекидним рутинама за **DMA2** и **PER3**. Део програма за слање блока података из меморије у периферију **PER2** је сличан са трећим делом програма из задатка 1.7 , док је део програма за слање блока података из меморије у периферију **PER3** сличан са трећим делом програма из задатка 1.2.

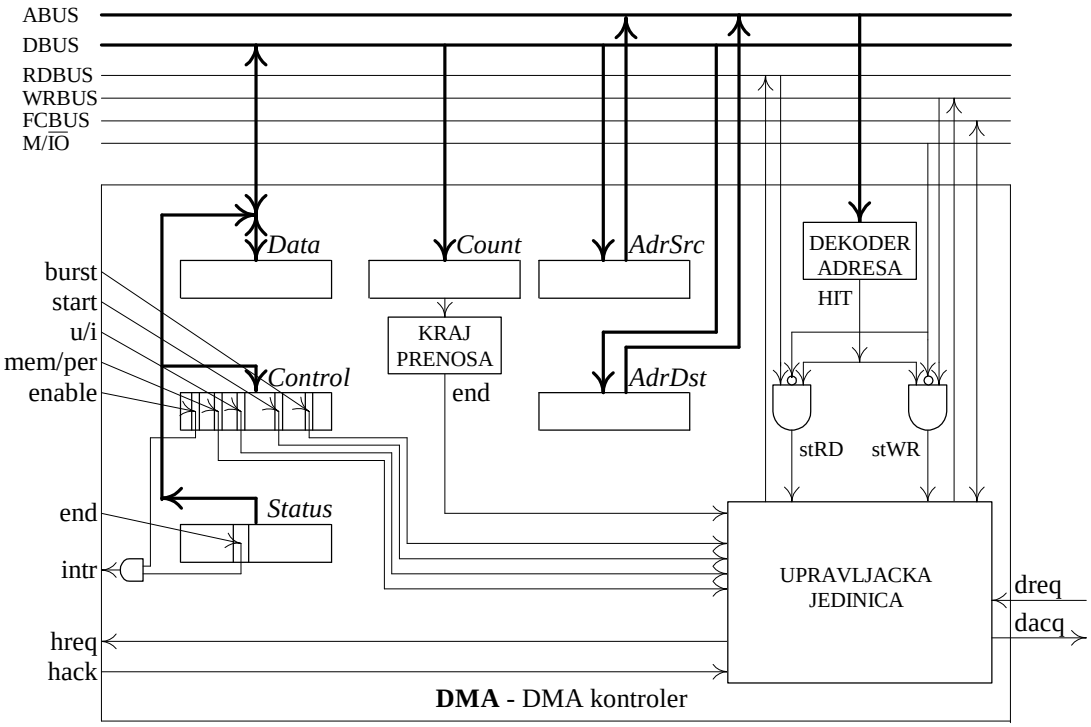
У оквиру дела програма који се извршава у главном програму врши се иницијализација преноса у периферије **PER2** и **PER3**, постављање "semafor"-а за **DMA2** и **PER3** на 1, стартовање контролера периферије **KPER2**, DMA контролера **DMA2** и контролера периферије **KPER3** и чекање на завршетак слања блокова података из меморије у периферије **PER2** и **PER3** у петљама са лабелама Wait2 и Wait3. У оквиру

дела програма који се извршава у прекидној рутини за **DMA2** врши се заустављање контролера периферије **KPER2** и DMA контролера **DMA2** и постављање "semafor"-а за **DMA2** на 0. У оквиру дела програма који се извршава у прекидној рутини периферије **PER3** врши се пренос податка из меморијске локације у регистар *Data* **KPER3**, инкрементирање садржаја меморијске локације MemAdr3 и декрементирање садржаја меморијске локације MemCnt3 и, уколико је пренет последњи податак, заустављање контролера периферије **KPER3** и постављање "semafor"-а за **PER3** на 0.

Прелазак на следећи део главног програма не сме да се дозволи док се не заврши слање блокова података у обе периферије. Чекање на завршетак слања блокова од по 100h података у обе периферије се реализује најпре у петљи са лабелом Wait2 за периферију **PER2** а затим и у петљи са лабелом Wait3 за периферију **PER3**, а само слање података из меморије у периферију **PER2** реализује се хардверски са DMA контролером **DMA2** и контролером периферије **PER2,** а у периферију **PER3** се реализују програмски са контролером периферије **PER3** у прекидној рутини периферије **PER3** на коју се прелази када контролер периферије **KPER3** генерише прекиде. При томе могу да се јаве два случаја у зависности од брзине периферија **PER2** и **PER3**. Први случај се јавља када је периферија **PER2** бржа од периферије **PER3**, па ће се слање података у периферију **PER2** завршити пре слања података у периферију **PER3** и прво "semafor" за **DMA2** поставити на вредност 0 а затим и "semafor" периферије **PER3** поставити на вредност 0. Због тога ће се из петље са лабелом Wait2 изаћи чим слање података у периферију **PER2** буде завршено и "semafor" периферије **PER2** буде постављен на вредност 0, али ће се затим у петљи са лабелом Wait3 чекати да буде завршено и слање података у периферију **PER3** и да "semafor" периферије **PER3** буде постављен на вредност 0, па затим прећи на следећи део главног програма. Други случај се јавља када је периферија **PER2** спорија од периферије **PER3**, па ће слање података у периферију **PER3** да буде завршено пре слања података у периферију **PER2** и прво "semafor" периферије **PER3** поставити на вредност 0 а затим и "semafor" за **DMA2** поставити на вредност 0. Због тога ће се из петље са лабелом Wait2 изаћи чим слање података у периферију **PER2** буде завршено и "semafor" за **DMA2** буде постављен на вредност 0, али се у петљи са лабелом Wait3 неће чекати, јер ће слање података у периферију **PER3** бити завршено и "semafor" периферије **PER3** постављен на вредност 0, па ће се одмах прећи на следећи део главног програма.

Дискусија:

1. За пренос блока података из меморије у меморију се у овом случају користи само DMA контролер **DMA1** док контролер периферије **KPER1** и периферија **PER1** нису потребни и могу се уклонити (слика 24).



Слика 24 DMA контролер **DMA** само за пренос меморија меморија

2. У поставци задатка је дефинисано да се најпре из периферије **PER0** са **KPER0** уноси блок од 100h података и смешта у меморијске локације од адресе A000h до адресе A0FFh, да се затим у процедури `Obrada` са **DMA1** преноси блок од 100h података из меморијских локација од адресе A000h до адресе A0FFh у меморијске локације од адресе B000h до адресе B0FFh и да се на крају упоредо шаљу блок од 100h података прочитаних из меморијских локација од адресе A000h до адресе A0FFh у периферију **PER2** са **KPER2** и **DMA2** и блок од 100h података прочитаних из меморијских локација од адресе B000h до адресе B0FFh у периферију **PER3** са **KPER3**. Задатак је било могуће дефинисати и тако да се подаци шаљу у периферију **PER2** из меморијских локација од адресе A000h до адресе A0FFh одмах по завршетку уноса података из периферије **PER0** у меморијске локације од адресе A000h до адресе A0FFh и то упоредо са преносом података из меморијских локација од адресе A000h до адресе A0FFh у меморијске локације од адресе B000h до адресе B0FFh. Међутим, слање блока података у периферију **PER3** из меморијских локација од адресе B000h до адресе B0FFh је могуће реализовати тек по завршетку преноса блока података из меморијских локација од адресе A000h до адресе A0FFh у меморијске локације од адресе B000h до адресе B0FFh.

3. Слање података из меморије у периферију **PER2** се реализује са контролером периферије **KPER2** и DMA контролером **DMA2** тек по завршетку преноса блока података из меморије у меморију са DMA контролером **DMA1**. При тако дефинисаном задатку могуће је пренос податак из меморије у меморију најпре реализовати са DMA контролером **DMA2**, а затим слање података из меморије у периферију **PER2** са контролером периферије **KPER2** и DMA контролером **DMA2**. У том случају у програму са слике 23 би требало процедуру `Obrada` и прекидну рутину за **DMA1** изменити тако да се уместо адреса регистара *Count* (FF1Bh), *AdrSrc* (FF1Ch), *AdrDst* (FF1Dh) и *Control* (FF18h) DMA контролера **DMA1** ставе адресе регистара *Count* (FF2Bh), *AdrSrc* (FF2Ch), *AdrDst* (FF2Dh) и *Control* (FF28h) DMA контролера **DMA2** и да се уместо меморијске локације MemSem1 "semafor"-а за **DMA1** стави меморијска локација MemSem2 "semafor"-а за **DMA2**.

1.9 ЗАДАТАК

Дат је рачунарски систем који чине процесор, меморија и контролер периферије **KPER0** и DMA контролер **DMA0** са периферијом **PER0**, контролер периферије **KPER1** са периферијом **PER1**, контролер периферије **KPER2** са периферијом **PER2** и контролер периферије **KPER3** са периферијом **PER3** повезани магистралом.

Процесор је са једноадресним форматом инструкција. Од програмски доступних регистара постоје 16 битни акумулатор ACC, указивач на врх стека SP, указивач на табелу (IV табела) са адресама прекидних рутина IVTP и програмска статусна реч PSW. Типови података који се користе су 16 битне целобројне величине са знаком и без знака. Приликом извршавања инструкције преноса у акумулатор и аритметичких, логичких и померачких инструкција постављају се индикатори N, Z, C и V програмске статусне речи PSW. Механизам прекида је векторисан. При прекиду хардверски се на стеку чувају програмски бројач PC и програмска статусна реч PSW. Бројеви улаза у IV табелу су фиксни за сваку периферију. Меморијски и улазно-излазни адресни простори су раздвојени.

Меморијски адресни простор је величине 2^{16} меморијских локација, при чему је ширина меморијских локација 16 битова.

Улазно-излазни адресни простор је величине 2^{16} регистара, при чему је ширина регистара 16 битова. Регистри контролера периферија **KPER0**, **KPER1, KPER2** и **KPER3** и регистри DMA контролера **DMA0** се налазе у улазно-излазном адресном простору.

Контролери периферија **KPER0**, **KPER1, KPER2** и **KPER3** имају управљачке регистре (*Control*), статусне регистре (*Status*) и регистре података (*Data*) и то редом на адресама FF00h, FF01h и FF02h за контролер периферије **KPER0**, FF10h, FF11h и FF12h за контролер периферије **KPER1**, FF20h, FF21h и FF22h за контролер периферије **KPER2** и FF30h, FF31h и FF32h за контролер периферије **KPER3**. У регистрима контролера периферија **KPER0**, **KPER1, KPER2** и **KPER3** највиши бит је означен са 15, а најнижи са 0. У управљачким регистрима *Control* бит 15 је бит *u/i* којим се задаје смер преноса са периферијом (0—улаз, 1—излаз), бит 4 је бит *start* којим се покреће извршавање операције (0—заустављен, 1—стартован) и бит 0 је бит *enable* којим се дозвољава прекид (0—маскиран, 1—дозвољен). У статусним регистрима *Status* бит 15 је бит спремности *ready* (0—није спреман, 1—спреман).

DMA контролер **DMA0** има управљачки регистар (*Control*), статусни регистар (*Status*), регистар податка (*Data*), регистар величине блока података (*Count*), изворишни адресни регистар (*AdrSrc*) и одредишни адресни регистар (*AdrDst*) и то редом на адресама FF08h, FF09h, FF0Ah, FF0Bh, FF0Ch и FF0Dh, респективно. У регистрима DMA контролера **DMA0** највиши бит је означен са 15, а најнижи са 0. У управљачком регистру *Control* бит 15 је бит *u/i* којим се задаје смер преноса са периферијом (0—улаз, 1—излаз), бит 14 је бит *mem/per* којим се задаје пренос између делова меморије или између периферије и меморије (0—периферија/меморија, 1—меморија/меморија), бит 4 је бит *start* којим се покреће извршавање операције (0—заустављен, 1—стартован), бит 1 је бит *burst* којим се задаје режим преноса (0—циклус по циклус, 1—цео блок) и бит 0 је бит *enable* којим се дозвољава прекид (0—маскиран, 1—дозвољен), при чему је вредност бита 15 битна само уколико је вредношћу 0 бита 14 задат пренос између периферије и меморије. У статусном регистру *Status* бит 15 је бит завршетка преноса *end* (0—пренос у току, 1—пренос завршен).

Адресне линије и линије података магистрале рачунара су широке по 16 битова.

Написати главни програм и одговарајућу прекидну рутину којима се упоредо учитавају низ A(i), где је i = 0, …, FFh, из **PER0** у меморијски блок који почиње од адресе A000h, низ B(i), где је i = 0, …, FFh, из **PER1** у меморијски блок који почиње од адресе B000h и низ C(i), где је i = 0, …, FFh, из **PER2** у меморијски блок који почиње од адресе C000h, затим формира низ D(i) = A(i) + B(i)+C(i), где је i = 0, …, FFh, у меморијском блоку који почиње од адресе D000h и резултујући низ D(i), где је i = 0, …, FFh, из меморијског блока који почиње од адресе D000h шаље у **PER3**. Улаз из периферије **PER0** реализовати са контролером периферије **KPER0** и DMA контролером **DMA0** у режиму преноса цео блок, улаз из периферије **PER1** реализовати са контролером периферије **KPER1** техником програмираног улаза са прекидом, улаз из периферије **PER2** реализовати са контролером периферије **KPER2** техником програмираног улаза са испитивањем спремности и излаз у периферију **PER3** реализовати са контролером периферије **KPER3** техником програмираног излаза са испитивањем спремности.

Решење:

Адресе и структура регистара контролера **KPER0**, **DMA0**, **KPER1**, **KPER2** и **KPER3** су дати на сликама 25 и 26, респективно.

KPER0 - Kontroler periferije **PER0**

Registar	Control	Status	Data
Adresa	FF00h	FF01h	FF02h

DMA0 –DMA kontroler

Registar	Control	Status	Data	Count	AdrSrc	AdrDst
Adresa	FF08h	FF09h	FF0Ah	FF0Bh	FF0Ch	FF0Dh

KPER1 - Kontroler periferije **PER1**

Registar	Control	Status	Data
Adresa	FF10h	FF11h	FF12h

KPER2 - Kontroler periferije **PER2**

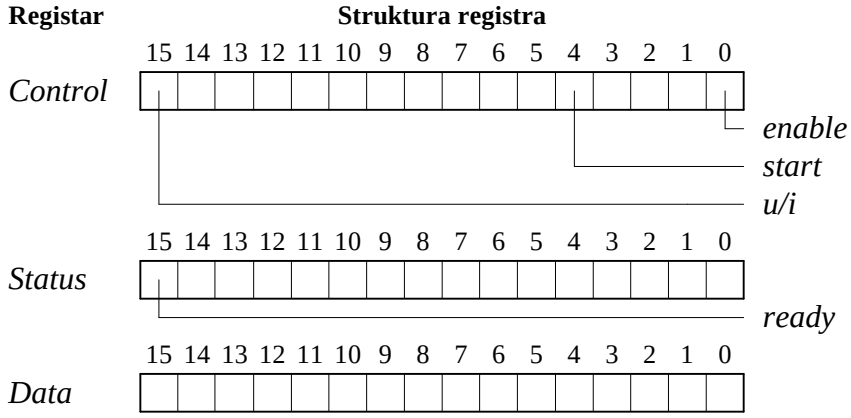
Registar	Control	Status	Data
Adresa	FF20h	FF21h	FF22h

KPER3 - Kontroler periferije **PER3**

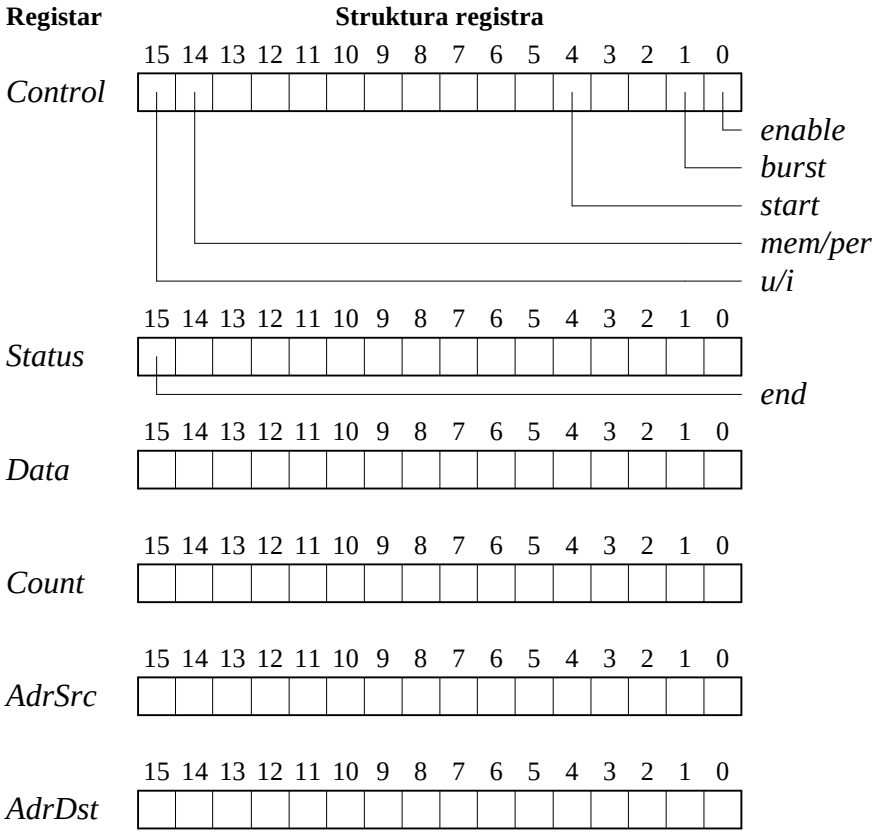
Registar	Control	Status	Data
Adresa	FF30h	FF31h	FF32h

Слика 25 Адресе регистара контролера

Registri kontrolera periferija **KPER**



Registri DMA kontrolera **DMA**



Слика 26 Структура регистара контролера

Периферија **PER0** се повезује на магистралу рачунарског система са контролером периферије **KPER0** и DMA контролера **DMA0** на начин приказан на слици 11, док се периферије **PER1**, **PER2** и **PER3** повезују на магистралу рачунарског система са контролерима периферија **KPER1**, **KPER2** и **KPER3**, респективно, на начин приказан на слици 1.

Тражени програм је приказан на слици 27.

```
;glavni program
;unos blokova podataka iz periferije PER0 u memoriju sa KPER0 i DMA0,
;iz periferije PER1 u memoriju sa KPER1 sa pekidom  i
;iz periferije PER2 u mem sa KPER2 sa ispitivanjem bita spremnosti ready
  ;inicijalizacija prenosa u memoriju iz PER0 sa KPER0 i DMA0,
  ;iz PER1 sa KPER1 i iz PER2 sa KPER2
    ;veličine blokova podataka
      LOAD   #100h    ;upis vel blok podat za PER0, PER1 i PER2 preko ACC
      OUT    FF0Bh    ;u reg Count DMA0 i
      STORE  MemCnt1  ;u mem. lok. na adr. MemCnt1
      STORE  MemCnt2  ;u mem. lok. na adr. MemCnt2
    ;početne adrese blokova podataka
      LOAD   #A000h   ;upis adr. bloka podataka za PER0 preko ACC u
      OUT    FF0Dh    ;reg AdrDst DMA0
      LOAD   #B000h   ;upis adr. bloka podataka za PER1 preko ACC u
      STORE  MemAdr1  ;mem. lok. na adr. MemAdr1
      LOAD   #C000h   ;upis adr. bloka podataka za PER2 preko ACC u
      STORE  MemAdr2  ;mem. lok. na adr. MemAdr2
  ;postavljanje "semafor"-a za DMA0 na 1
      LOAD   #1h      ;upis vrednosti 1 preko ACC u
      STORE  MemSem0  ;mem. lok. na adr. MemSem0
  ;postavljanje "semafor"-a za PER1 na 1
                      ;upis vrednosti 1 preko ACC u
      STORE  MemSem1  ;mem. lok. na adr. MemSem1
  ;startovanje kontrolera periferije KPER0
      LOAD   #0010h   ;upis režima rada i startovanja (u/i=0,
      OUT    FF00h    ;start=1,enable=0)preko ACC u reg. Control KPER0
  ;startovanje kontrolera DMA0
      LOAD   #0013h   ;upis režima rada i start(u/i=0,mem/per=0,start=1,
      OUT    FF08h    ;burst=1,enable=1)preko ACC u reg. Control DMA0
  ;startovanje kontrolera periferije KPER1
      LOAD   #0011h   ;upis režima rada i startovanja (u/i=0,
      OUT    FF10h    ;start=1,enable=1)preko ACC u reg. Control KPER1
  ;startovanje kontrolera periferije KPER2
      LOAD   #0010h   ;upis režima rada i startovanja (u/i=0,
      OUT    FF20h    ;start=1,enable=0)preko ACC u reg. Control KPER2
  ;unos blokova podataka
    ;provera da li podatak može da se prenese iz registra Data KPER2
Loop2: IN    FF21h    ;upis sadrž. reg. Status KPER2 u ACC
      AND    #8000h   ;ispitivanje bita ready
      JZ     Loop2    ;povratak na Loop2 ukoliko je ready 0
    ;prenos podatka iz registra Data KPER2 u memorijsku lokaciju
      IN     FF22h    ;upis sadrž. reg. Data KPER2 preko ACC u mem. lok.
      STORE  (MemAdr2);na adresi datoj sadrž. mem. lok. na adr. MemAdr2
    ;inkrementiranje sadržaja memorijske lokacije MemAdr2
      INC    MemAdr2  ;inkrement sadrž. mem.lok. sa adr. MemAdr2
    ;dekrementiranje sadržaja memorijske lokacije MemCnt2
      DEC    MemCnt2  ;dekrement sadrž. mem.lok. sa adr. MemCnt2
    ;provera da li je unet poslednji podatak iz PER2
      JNZ    Loop2    ;povratak na Loop2 ukoliko nije poslednji podatak
  ;zaustavljanje kontrolera periferije KPER2
      LOAD   #0       ;upis režima zaustavljanja (u/i=0,
      OUT    FF20h    ;start=0, enable=0) preko ACC u reg. Control KPER2
```

```
  ;čekanje na završetak unosa bloka podataka iz periferiju PER0 u memoriju
  ;sa KPER0 i DMA0 proverom vrednosti "semafor"-a za DMA0
Wait0: LOAD   MemSem0  ;upis sadrž. mem.lok. sa adr. MemSem0 u ACC
       CMP    #0       ;ispitivanje da li je "semafor" postao 0
       JNZ    Wait0    ;povratak na Wait0 ukoliko "semafor" nije 0
  ;čekanje na završetak unosa bloka podataka iz periferiju PER1 u memoriju
  ;sa KPER1 proverom vrednosti "semafor"-a za PER1
Wait1: LOAD   MemSem1  ;upis sadrž. mem.lok. sa adr. MemSem1 u ACC
       CMP    #0       ;ispitivanje da li je "semafor" postao 0
       JNZ    Wait1    ;povratak na Wait1 ukoliko "semafor" nije 0
;obrada
       JSR    Obrada   ;obrada blokova od po 100h podataka u memorijskim
  ;lokacijama od adrese A000h do adrese A0FFh, od adrese B000h do adrese
  ;B0FFh i od adrese C000h do adrese C0FFh i upis rezultata u memorijske
  ;lokacije od adrese D000h do adrese D0FFh

;slanje bloka podataka iz memorije u periferiju PER3 sa KPER3 sa
;ispitivanjem bita spremnosti ready
  ;inicijalizacija prenosa iz memorije u periferiju PER3 sa KPER3
    ;veličine bloka podataka
      LOAD   #100h    ;upis veličine bloka podataka za PER3 preko ACC u
      STORE  MemCnt3  ;mem. lok. na adr. MemCnt3
    ;početna adresa bloka podataka
      LOAD   #D000h   ;upis adr. bloka podataka za PER3 preko ACC u
      STORE  MemAdr3  ;mem. lok. na adr. MemAdr3
  ;startovanje kontrolera periferije KPER3
       LOAD   #8040h   ;upis režima rada i start (u/i=1, start=1,
       OUT    FF30h    ;enable=0) preko ACC u reg. Control KPER3
  ;slanje bloka podataka
    ;provera da li podatak može da se prenese iz mem u registar Data KPER3
Loop3: IN     FF31h    ;upis sadrž. reg. Status KPER3 u ACC
       AND    #1       ;ispitivanje bita ready
       JZ     Loop3    ;povratak na Loop3 ukoliko je ready 0
    ;prenos podatka iz memorijske lokacije u registar Data KPER3
      LOAD   (MemAdr3);upis sadrž. mem. lok. sa adr. date sadrž. mem.
      OUT    FF32h    ;lok. sa adr. MemAdr3 preko ACC u reg. Data KPER3
    ;inkrementiranje sadržaja memorijske lokacije MemAdr3
      INC    MemAdr3  ;inkr sadrž. mem.lok. sa adr. MemAdr3
    ;dekrementiranje sadržaja memorijske lokacije MemCnt3
      DEC    MemCnt3  ;dekr sadrž. mem.lok. sa adr. MemCnt3
    ;provera da li je prenet poslednji podatak
      JNZ    Loop3    ;povratak na Loop3 ukoliko nije poslednji podatak
  ;zaustavljanje kontrolera periferije KPER3
       LOAD   #0       ;upis režima zaustavljanja (start=0, u/i=0,
       OUT    FF30h    ;enable=0)preko ACC u reg. Control KPER3

. . .
```

```
;prekidna rutina za DMA0
DMA0:  PUSHA           ;akumulator ACC na stek
  ;zaustavljanje kontrolera periferije KPER0
       LOAD   #0       ;upis režima zaustavljanja (u/i=0,
       OUT FF00h       ;start=0,enable=0)preko ACC u reg. Control KPER0
  ;zaustavljanje kontrolera DMA0
                       ;upis režima zaustav (u/i=0,mem/per=0,start=0,
       OUT FF08h       ;burst=0,enable=0)preko ACC u reg. Control DMA0
  ;postavljanje "semafor"-a za DMA0 na 0
                       ;upis vrednosti 0 preko ACC u
       STORE  MemSem0  ;mem. lok. na adr. MemSem0
  ;izlazak iz prekidne rutine
       POPA            ;restauracija akumulatora ACC sa steka
       RTI             ;povratak iz prekidne rutine


;prekidna rutina periferije PER1
;unos bloka podataka iz periferiju PER1 u memoriju sa KPER1
PER1:  PUSHA           ;akumulator ACC na stek
  ;unos podatka iz registra Data KPER1 u memorijsku lokaciju
       IN FF12h        ;upis sadrž. reg. Data KPER1 preko ACC u mem. lok.
       STORE  (MemAdr1);na adr. date sadrž. mem. lok. na adr. MemAdr1
  ;inkrementiranje sadržaja memorijske lokacije MemAdr1
       INC    MemAdr1  ;inkr sadrž. mem.lok. sa adr. MemAdr1
  ;dekrementiranje sadržaja memorijske lokacije MemCnt0
       DEC    MemCnt1  ;dekr sadrž. mem.lok. sa adr. MemCnt1
  ;provera da li je prenet poslednji podatak
       JNZ    Back1    ;prelaz na Back1 ukoliko nije poslednji podatak
  ;zaustavljanje kontrolera periferije KPER1
       LOAD   #0       ;upis režima zaustavljanja (start=0, u/i=0,
       OUT FF10h       ;enable=0)preko ACC u reg. Control KPER1
  ;postavljanje "semafor"-a za PER1 na 0
                       ;upis vrednosti 0 preko ACC u
       STORE  MemSem1  ;mem. lok. na adr. MemSem1
  ;izlazak iz prekidne rutine
Back1: POPA            ;restauracija akumulatora ACC sa steka
       RTI             ;povratak iz prekidne rutine
```

```
;Procedura Obrada u kojoj se izračunava D(i)=A(i)+B(i)+C(i),
;gde je i=0,…,FFh
;Elementi niza A(i), gde je i=0,…,FFh, su na adresama A000h do A0FFh,
;elementi niza B(i), gde je i=0,…,FFh, su na adresama B000h do B0FFh,
;elementi niza C(i), gde je i=0,…,FFh, su na adresama C000h do C0FFh, a
;elementi niza sume D(i), gde je i=0,…,FFh, se smešt na adr D000h do D0FFh
Obrada:   LOAD   #100h     ;upis vel blok podat za A(i), B(i), C(i) i D(i)
          STORE  MemCnt3   ;u ACC pa u mem. lok. na adr. MemCnt3
          LOAD   #A000h    ;upis adr. bloka podataka A(i)preko ACC u
          STORE  MemAdr0   ;mem. lok. na adr. MemAdr0
          LOAD   #B000h    ;upis adr. bloka podataka B(i)preko ACC u
          STORE  MemAdr1   ;mem. lok. na adr. MemAdr1
          LOAD   #C000h    ;upis adr. bloka podataka C(i)preko ACC u
          STORE  MemAdr2   ;mem. lok. na adr. MemAdr2
          LOAD   #D000h    ;upis adr. bloka podataka D(i)preko ACC u
          STORE  MemAdr3   ;mem. lok. na adr. MemAdr3

Loop:     LOAD   (MemAdr0) ;upis A(i) iz mem lok sa adrese date sadr
                           ;mem lok sa adrese MemAdr0 u ACC
          ADD    (MemAdr1) ;suma ACC i B(i) iz mem lok sa adrese date sadr
                           ;mem lok sa adrese MemAdr1 u ACC
          ADD    (MemAdr2) ;suma ACC i C(i) iz mem lok sa adrese date sadr
                           ;mem lok sa adrese MemAdr2 u ACC
          STORE  (MemAdr3) ;sadržaj ACC u D(i) u mem lok na adr datoj sadr
                           ;mem lok na adresi MemAdr3
          INC    MemAdr0   ;inkrementiranje sadr mem lok MemAdr0
          INC    MemAdr1   ;inkrementiranje sadr mem lok MemAdr1
          INC    MemAdr2   ;inkrementiranje sadr mem lok MemAdr2
          INC    MemAdr3   ;inkrementiranje sadr mem lok MemAdr3
          DEC    MemCnt1   ;dekrementiranje sadr mem lok MemCnt3
  ;provera da li je poslednja suma
          JNZ    Loop      ;povratak na Loop ukoliko nije poslednja suma
          RTS              ;povratak u glavni program
```

Слика 27 Програм

Програм се састоји из три дела. У првом делу се из периферија **PER0**, **PER1** и **PER2** упоредо уносе три блока од по 100h података и смештају у меморијске локације од адресе A000h до адресе A0FFh за периферију **PER0,** од адресе B000h до адресе B0FFh за периферију **PER1** и од адресе C000h до адресе C0FFh за периферију **PER2**. Унос из периферије **PER0** се реализује са контролером периферије **KPER0** и DMA контролером **DMA0** у режиму преноса цео блок, унос из периферије **PER1** се реализује са контролером периферије **KPER1** техником програмираног улаза са прекидом и унос из периферије **PER2** реализује са контролером периферије **KPER2** техником програмираног улаза са испитивањем спремности. У другом делу се позива процесура `Obrada` која врши одређену обраду над унетим подацима у блоковима од по 100h података у меморијским локацијама од адресе A000h до адресе A0FFh, од адресе B000h до адресе B0FFh и од адресе C000h до адресе C0FFh и резултат који представља блок од 100h података смешта у меморијске локације од адресе D000h до адресе D0FFh. У трећем делу се у периферију **PER2** техником програмираног излаза са испитивањем спремности шаље блок од 100h података прочитаних из меморијских локација од адресе D000h до адресе D0FFh.

У првом делу програма је унос података из **PER0**, **PER1** и **PER2** у меморију реализован на сличан начин као унос података у првом делу програма у задатку 1.6, у коме се улаз из периферије **PER0** реализује са контролером периферије **KPER0** и DMA контролером **DMA0** у режиму преноса циклус по циклус и улаз из периферије **PER2** реализује са контролером периферије **KPER2** техником програмираног улаза са испитивањем спремности, затим на сличан начин као унос података у првом делу програма у задатку 1.7, у коме се улаз из периферије **PER0** реализује са контролером периферије **KPER0** техником програмираног улаза са прекидом и улаз из периферије **PER1** реализује са контролером периферије **KPER1** техником програмираног улаза са испитивањем спремности, а чекање на завршетак уноса података из периферија **PER0** и **PER1** у петљама са лабелама `Wait0` и `Wait1` се реализује на исти начин као у задатку 1.4 у коме су улази из периферија **PER0** и **PER1** реализују техником програмираног улаза са прекидом.

Други део се реализује у процедури `Obrada` која је слична са процедурама `Obrada` у задацима 1.6 и 1.7 у којима се израчунава `C(i)=A(i)+B(i)`, док се у овом задатку израчунава `D(i)=A(i)+B(i)+C(i).`

У трећем делу се у периферију **PER2** техником програмираног излаза са испитивањем спремности шаље блок података на сличан начин као у задатку 1.1.

1.10 ЗАДАТАК

Дат је рачунарски систем који чине процесор, меморија и контролер периферије **KPER0** и DMA контролер **DMA0** са периферијом **PER0**, контролер периферије **KPER1** са периферијом **PER1**, контролер периферије **KPER2** и DMA контролер **DMA2** са периферијом **PER2** и контролер периферије **KPER3** са периферијом **PER3** повезани магистралом.

Процесор је са једноадресним форматом инструкција. Од програмски доступних регистара постоје 16 битни акумулатор ACC, указивач на врх стека SP, указивач на табелу (IV табела) са адресама прекидних рутина IVTP и програмска статусна реч PSW. Типови података који се користе су 16 битне целобројне величине са знаком и без знака. Приликом извршавања инструкције преноса у акумулатор и аритметичких, логичких и померачких инструкција постављају се индикатори N, Z, C и V програмске статусне речи PSW. Механизам прекида је векторисан. При прекиду хардверски се на стеку чувају програмски бројач PC и програмска статусна реч PSW. Бројеви улаза у IV табелу су фиксни за сваку периферију. Меморијски и улазно-излазни адресни простори су раздвојени.

Меморијски адресни простор је величине 2^{16} меморијских локација, при чему је ширина меморијских локација 16 битова.

Улазно-излазни адресни простор је величине 2^{16} регистара, при чему је ширина регистара 16 битова. Регистри контролера периферија **KPER0**, **KPER1**, **KPER2** и **KPER3** и регистри DMA контролера **DMA0** и **DMA2** се налазе у улазно-излазном адресном простору.

Контролери периферија **KPER0**, **KPER1**, **KPER2** и **KPER3** имају управљачке регистре (*Control*), статусне регистре (*Status*) и регистре података (*Data*) и то редом на адресама FF00h, FF01h и FF02h за контролер периферије **KPER0**, FF10h, FF11h и FF12h за контролер периферије **KPER1**, FF20h, FF21h и FF22h за контролер периферије **KPER2** и FF30h, FF31h и FF32h за контролер периферије **KPER3**. У регистрима контролера периферија **KPER0**, **KPER1**, **KPER2** и **KPER3** највиши бит је означен са 15, а најнижи са 0. У управљачким регистрима *Control* бит 15 је бит *u/i* којим се задаје смер преноса са периферијом (0—улаз, 1—излаз), бит 4 је бит *start* којим се покреће извршавање операције (0—заустављен, 1—стартован) и бит 0 је бит *enable* којим се дозвољава прекид (0—маскиран, 1—дозвољен). У статусним регистрима *Status* бит 15 је бит спремности *ready* (0—није спреман, 1—спреман).

DMA контролери **DMA0** и **DMA2** имају управљачки регистар (*Control*), статусни регистар (*Status*), регистар податка (*Data*), регистар величине блока података (*Count*), изворишни адресни регистар (*AdrSrc*) и одредишни адресни регистар (*AdrDst*) и то редом на адресама FF08h, FF09h, FF0Ah, FF0Bh, FF0Ch и FF0Dh за DMA контролер **DMA0** и FF28h, FF29h, FF2Ah, FF2Bh, FF2Ch и FF2Dh за DMA контролер **DMA2**. У регистрима DMA контролера **DMA0** и **DMA2** највиши бит је означен са 15, а најнижи са 0. У управљачком регистру *Control* бит 15 је бит *u/i* којим се задаје смер преноса са периферијом (0—улаз, 1—излаз), бит 14 је бит *mem/per* којим се задаје пренос између делова меморије или између периферије и меморије (0—периферија/меморија, 1—меморија/меморија), бит 4 је бит *start* којим се покреће извршавање операције (0—заустављен, 1—стартован), бит 1 је бит *burst* којим се задаје режим преноса (0—циклус по циклус, 1—цео блок) и бит 0 је бит *enable* којим се дозвољава прекид (0—маскиран, 1—дозвољен), при чему је вредност бита 15 битна само уколико је вредношћу 0 бита 14 задат пренос између периферије и меморије. У статусном регистру *Status* бит 15 је бит завршетка преноса *end* (0—пренос у току, 1—пренос завршен).

Адресне линије и линије података магистрале рачунара су широке по 16 битова.

Написати главни програм и одговарајућу прекидну рутину којима се упоредо учитавају низ A(i), где је i = 0, …, FFh, из **PER0** у меморијски блок који почиње од адресе A000h и низ B(i), где је i = 0, …, FFh, из **PER1** у меморијски блок који почиње од адресе B000h, затим формирају низ $C(i) = A(i) + B(i)$, где је i = 0, …, FFh, у меморијском блоку који почиње од адресе C000h и низ $D(i) = A(i) - B(i)$, где је i = 0, …, FFh, у меморијском блоку који почиње од адресе D000h и на крају упоредо шаљу резултујући низ C(i), где је i = 0, …, FFh, из меморијског блока који почиње од адресе C000h у периферију **PER2** и низ D(i), где је i = 0, …, FFh, из меморијског блока који почиње од адресе D000h у периферију **PER3**. Улаз из периферије **PER0** реализовати са контролером периферије **KPER0** и DMA контролером **DMA0** у режиму преноса цео блок, улаз из периферије **PER1** реализовати са контролером периферије **KPER1** техником програмираног улаза са прекидом, излаз у периферију **PER2** реализовати са контролером периферије **KPER2** и DMA контролером **DMA2** у режиму преноса циклус по циклус и излаз у периферију **PER3** реализовати са контролером периферије **KPER3** техником програмираног излаза са испитивањем спремности.

Решење:

Периферије **PER0** и **PER2** се повезује на магистралу рачунарског система са контролерима периферије **KPER0** и **KPER2** и DMA контролерима **DMA0** и **DMA2** на начин приказан на слици 11, док се периферије **PER1** и **PER3** повезују на магистралу рачунарског система са контролерима периферија **KPER1** и **KPER3** на начин приказан на слици 1.

Адресе и структура регистара контролера **KPER0**, **DMA0**, **KPER1**, **KPER2, DMA2** и **KPER3** су дати на сликама 28 и 29, респективно.

KPER0 - Kontroler periferije **PER0**

Registar	Control	Status	Data
Adresa	FF00h	FF01h	FF02h

DMA0 –DMA kontroler

Registar	Control	Status	Data	Count	AdrSrc	AdrDst
Adresa	FF08h	FF09h	FF0Ah	FF0Bh	FF0Ch	FF0Dh

KPER1 - Kontroler periferije **PER1**

Registar	Control	Status	Data
Adresa	FF10h	FF11h	FF12h

KPER2 - Kontroler periferije **PER2**

Registar	Control	Status	Data
Adresa	FF20h	FF21h	FF22h

DMA2 –DMA kontroler

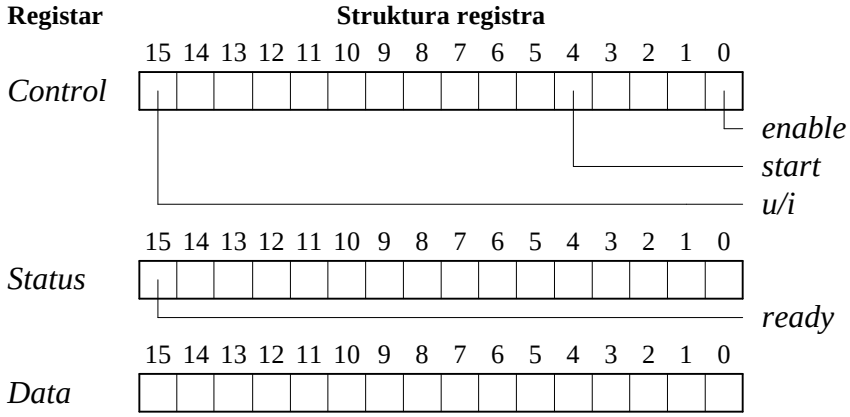
Registar	Control	Status	Data	Count	AdrSrc	AdrDst
Adresa	FF28h	FF29h	FF2Ah	FF2Bh	FF2Ch	FF2Dh

KPER3 - Kontroler periferije **PER3**

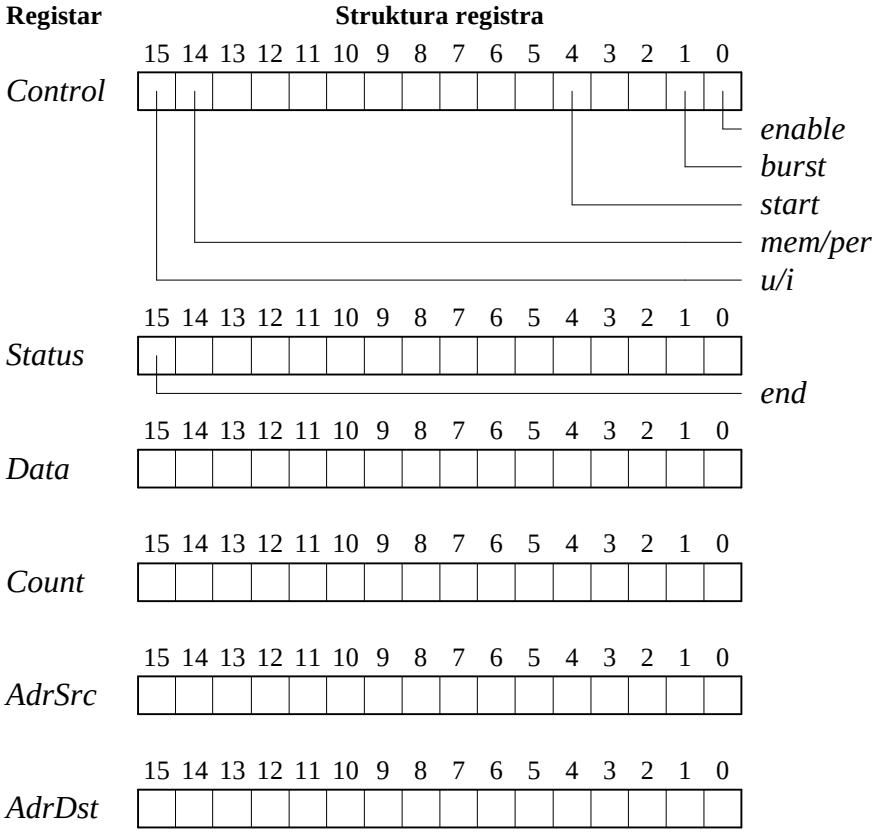
Registar	Control	Status	Data
Adresa	FF30h	FF31h	FF32h

Слика 28 Адресе регистара контролера

Registri kontrolera periferija **KPER**



Registri DMA kontrolera **DMA**



Слика 29 Структура регистара контролера

Тражени програм је приказан на слици 30.

```
;glavni program
;unos blokova podataka iz periferije PER0 u memoriju sa KPER0 i DMA0 i
;iz periferije PER1 u memoriju sa KPER1 sa prekidom
  ;inicijalizacija prenosa u mem iz PER0 sa KPER0 i DMA0 i iz PER1 sa KPER1
    ;veličine blokova podataka
      LOAD   #100h     ;upis vel blok podat za PER0 i PER1 preko ACC
      OUT    FF0Bh     ;u reg Count DMA0 i
      STORE  MemCnt1   ;u mem. lok. na adr. MemCnt1
    ;početne adrese blokova podataka
      LOAD   #A000h    ;upis adr. bloka podataka za PER0 preko ACC u
      OUT    FF0Dh     ;reg AdrDst DMA0
      LOAD   #B000h    ;upis adr. bloka podataka za PER1 preko ACC u
      STORE  MemAdr1   ;mem. lok. na adr. MemAdr1
  ;postavljanje "semafor"-a za DMA0 na 1
      LOAD   #1h       ;upis vrednosti 1 preko ACC u
      STORE  MemSem0   ;mem. lok. na adr. MemSem0
  ;postavljanje "semafor"-a za PER1 na 1
                       ;upis vrednosti 1 preko ACC u
      STORE  MemSem1   ;mem. lok. na adr. MemSem1
  ;startovanje kontrolera periferije KPER0
      LOAD   #0010h    ;upis režima rada i startovanja (u/i=0,
      OUT    FF00h     ;start=1,enable=0)preko ACC u reg. Control KPER0
  ;startovanje kontrolera DMA0
      LOAD   #0013h    ;upis režima rada i start(u/i=0,mem/per=0,start=1,
      OUT    FF08h     ;burst=1,enable=1)preko ACC u reg. Control DMA0
  ;startovanje kontrolera periferije KPER1
      LOAD   #0011h    ;upis režima rada i startovanja (u/i=0,
      OUT    FF10h     ;start=1,enable=1)preko ACC u reg. Control KPER1
  ;unos blokova podataka
  ;čekanje na završetak unosa bloka podataka iz periferiju PER0 u memoriju
  ;sa KPER0 i DMA0 proverom vrednosti "semafor"-a za DMA0
Wait0: LOAD   MemSem0   ;upis sadrž. mem.lok. sa adr. MemSem0 u ACC
       CMP    #0        ;ispitivanje da li je "semafor" postao 0
       JNZ    Wait0     ;povratak na Wait0 ukoliko "semafor" nije 0
  ;čekanje na završetak unosa bloka podataka iz periferiju PER1 u memoriju
  ;sa KPER1 proverom vrednosti "semafor"-a za PER1
Wait1: LOAD   MemSem1   ;upis sadrž. mem.lok. sa adr. MemSem1 u ACC
       CMP    #0        ;ispitivanje da li je "semafor" postao 0
       JNZ    Wait1     ;povratak na Wait1 ukoliko "semafor" nije 0
;obrada
       JSR    Obrada    ;obrada blokova od po 100h podataka u memorijskim
  ;lokacijama od adrese A000h do adrese A0FFh i od adrese B000h do adrese
  ;B0FFh i upis rezultata u memorijske lokacije od adrese C000h do adrese
  ;C0FFh i od adrese D000h do adrese D0FFh
```

```
;slanje bloka podataka iz memorije u periferiju PER2 sa KPER2 i DMA2 i
;iz memorije u periferiju PER3 sa KPER3 ispitivanjem bita spremnosti ready
  ;inicijalizacija prenosa iz mem u PER2 sa KPER2 i DMA2 i u PER3 sa KPER3
    ;veličina bloka podataka
      LOAD   #100h     ;upis velič bloka podat za PER2 i PER3 preko ACC u
      OUT    FF2Bh     ;u reg. Count DMA2 i
      STORE  MemCnt3   ;mem. lok. na adr. MemCnt3
    ;početna adresa bloka podataka
      LOAD   #C000h    ;upis adr. bloka podataka za PER2 preko ACC u
      STORE  FF2Ch     ;u reg. AdrSrc DMA2
      LOAD   #D000h    ;upis adr. bloka podataka za PER3 preko ACC u
      STORE  MemAdr3   ;mem. lok. na adr. MemAdr3
  ;postavljanje "semafor"-a za DMA2 na 1
      LOAD   #1h       ;upis vrednosti 1 preko ACC u
      STORE  MemSem2   ;mem. lok. na adr. MemSem2
  ;startovanje kontrolera periferije KPER2
      LOAD   #8010h    ;upis režima rada i startovanja (u/i=1,
      OUT    FF20h     ;start=1,enable=0)preko ACC u reg. Control KPER2
  ;startovanje kontrolera DMA2
      LOAD   #8011h    ;upis režima rada i start(u/i=1,mem/per=0,start=1,
      OUT    FF28h     ;burst=0,enable=1)preko ACC u reg. Control DMA2
  ;startovanje kontrolera periferije KPER3
      LOAD   #8010h    ;upis režima rada i start PER3 (u/i=1, start=1,
      OUT    FF30h     ;enable=0) preko ACC u reg. Control KPER3
  ;slanje bloka podataka
    ;provera da li podatak može da se prenese u registar Data KPER3
Loop3: IN     FF31h     ;upis sadrž. reg. Status KPER3 u ACC
       AND    #1        ;ispitivanje bita ready
       JZ     Loop3     ;povratak na Loop3 ukoliko je ready 0
    ;prenos podatka iz memorijske lokacije u registar Data KPER3
      LOAD   (MemAdr3) ;upis sadrž. mem. lok. sa adr. date sadrž. mem.
      OUT    FF32h     ;lok. sa adr. MemAdr3 preko ACC u reg. Data KPER3
    ;inkrementiranje sadržaja memorijske lokacije MemAdr3
      INC    MemAdr3   ;inkr sadrž. mem.lok. sa adr. MemAdr3
    ;dekrementiranje sadržaja memorijske lokacije MemCnt3
      DEC    MemCnt3   ;dekr sadrž. mem.lok. sa adr. MemCnt3
    ;provera da li je prenet poslednji podatak
      JNZ    Loop3     ;povratak na Loop3 ukoliko nije poslednji podatak
  ;zaustavljanje kontrolera periferije KPER3
      LOAD   #0        ;upis režima zaustavljanja PER3(start=0, u/i=0,
      OUT    FF30h     ;enable=0)preko ACC u reg. Control KPER3
  ;čekanje na završetak slanja bloka podataka iz memorije u periferiju
  ;PER2 sa KPER2 i DMA2 proverom vrednosti "semafor"-a za DMA2
Wait2: LOAD   MemSem2   ;upis sadrž. mem.lok. sa adr. MemSem2 u ACC
       CMP    #0        ;ispitivanje da li je "semafor" postao 0
       JNZ    Wait2     ;povratak na Wait2 ukoliko "semafor" nije 0
. . .
```

```
;prekidna rutina za DMA0
DMA0:  PUSHA            ;akumulator ACC na stek
  ;zaustavljanje kontrolera periferije KPER0
       LOAD   #0        ;upis režima zaustavljanja (u/i=0,
       OUT FF00h        ;start=0,enable=0)preko ACC u reg. Control KPER0
  ;zaustavljanje kontrolera DMA0
                        ;upis režima zaustavljanja(u/i=0,mem/per =0,start=0,
       OUT FF08h        ;burst=0,enable=0)preko ACC u reg. Control DMA0
  ;postavljanje "semafor"-a za DMA0 na 0
                        ;upis vrednosti 0 preko ACC u
       STORE  MemSem0   ;mem. lok. na adr. MemSem0
  ;izlazak iz prekidne rutine
       POPA             ;restauracija akumulatora ACC sa steka
       RTI              ;povratak iz prekidne rutine

;prekidna rutina periferije PER1
;unos bloka podataka iz periferiju PER1 sa KPER1 u memoriju
PER1:  PUSHA            ;akumulator ACC na stek
  ;unos podatka iz registra Data KPER1 u memorijsku lokaciju
       IN FF12h         ;upis sadrž. reg. Data KPER1 preko ACC u mem. lok.
       STORE  (MemAdr1) ;na adr. date sadrž. mem. lok. na adr. MemAdr1
  ;inkrementiranje sadržaja memorijske lokacije MemAdr1
       INC    MemAdr1   ;inkr sadrž. mem.lok. sa adr. MemAdr1
  ;dekrementiranje sadržaja memorijske lokacije MemCnt0
       DEC    MemCnt1   ;dekr sadrž. mem.lok. sa adr. MemCnt1
  ;provera da li je prenet poslednji podatak
       JNZ    Back1     ;prelaz na Back1 ukoliko nije poslednji podatak
  ;zaustavljanje kontrolera periferije PER1
       LOAD   #0        ;upis režima zaustavljanja (start=0, u/i=0,
       OUT FF10h        ;enable=0)preko ACC u reg. Control KPER1
  ;postavljanje "semafor"-a za PER1 na 0
                        ;upis vrednosti 0 preko ACC u
       STORE  MemSem1   ;mem. lok. na adr. MemSem1
  ;izlazak iz prekidne rutine
Back1: POPA             ;restauracija akumulatora ACC sa steka
       RTI              ;povratak iz prekidne rutine
```

```
;Procedura Obrada u kojoj se izračunava C(i)=A(i)+B(i)i D(i)=A(i)-B(i),
;gde je i=0,…,FFh
;Elementi niza A(i), gde je i=0,…,FFh, su na adresama A000h do A0FFh,
;elementi niza B(i), gde je i=0,…,FFh, su na adresama B000h do B0FFh, a
;elem niza sume C(i), gde je i=0,…,FFh, se smešt na adr C000h do C0FFh i
;elem niza razlike D(i), gde je i=0,…,FFh, se smešt na adr D000h do D0FFh
Obrada:   LOAD   #100h     ;upis vel blok podat za A(i), B(i), C(i) i D(i)
          STORE  MemCnt3   ;u ACC pa u mem. lok. na adr. MemCnt3
          LOAD   #A000h    ;upis adr. bloka podataka A(i)preko ACC u
          STORE  MemAdr0   ;mem. lok. na adr. MemAdr0
          LOAD   #B000h    ;upis adr. bloka podataka B(i)preko ACC u
          STORE  MemAdr1   ;mem. lok. na adr. MemAdr1
          LOAD   #C000h    ;upis adr. bloka podataka C(i)preko ACC u
          STORE  MemAdr2   ;mem. lok. na adr. MemAdr2
          LOAD   #D000h    ;upis adr. bloka podataka D(i)preko ACC u
          STORE  MemAdr3   ;mem. lok. na adr. MemAdr3

Loop:     LOAD   (MemAdr0) ;upis A(i) iz mem lok sa adrese date sadr
                           ;mem lok sa adrese MemAdr0 u ACC
          ADD    (MemAdr1) ;suma ACC i B(i) iz mem lok sa adrese date sadr
                           ;mem lok sa adrese MemAdr1 u ACC
          STORE  (MemAdr2) ;sadržaj ACC u C(i) u mem lok na adr datoj sadr
                           ;mem lok na adresi MemAdr2
          LOAD   (MemAdr0) ;upis A(i) iz mem lok sa adrese date sadr
                           ;mem lok sa adrese MemAdr0 u ACC
          SUB    (MemAdr1) ;razl ACC i B(i) iz mem lok sa adrese date sadr
                           ;mem lok sa adrese MemAdr1 u ACC
          STORE  (MemAdr3) ;sadržaj ACC u D(i) u mem lok na adr datoj sadr
                           ;mem lok na adresi MemAdr3
          INC    MemAdr0   ;inkrementiranje sadr mem lok MemAdr0
          INC    MemAdr1   ;inkrementiranje sadr mem lok MemAdr1
          INC    MemAdr2   ;inkrementiranje sadr mem lok MemAdr2
          INC    MemAdr3   ;inkrementiranje sadr mem lok MemAdr3
          DEC    MemCnt1   ;dekrementiranje sadr mem lok MemCnt3
  ;provera da li je poslednja suma
          JNZ    Loop      ;povratak na Loop ukoliko nije poslednja suma
          RTS              ;povratak u glavni program

;prekidna rutina za DMA2
DMA2:  PUSHA               ;akumulator ACC na stek
  ;zaustavljanje kontrolera periferije KPER2
       LOAD   #0           ;upis režima zaustavljanja (u/i=0,
       OUT FF20h           ;start=0,enable=0)preko ACC u reg. Control KPER2
  ;zaustavljanje kontrolera DMA2
                           ;upis režima zaustavljanja(u/i=0,mem/per =0,start=0,
       OUT FF28h           ;burst=0,enable=0)preko ACC u reg. Control DMA2
  ;postavljanje "semafor"-a za DMA2 na 0
                           ;upis vrednosti 0 preko ACC u
       STORE  MemSem2      ;mem. lok. na adr. MemSem2
  ;izlazak iz prekidne rutine
       POPA                ;restauracija akumulatora ACC sa steka
       RTI                 ;povratak iz prekidne rutine

. . .
```

Слика 30 Програм

Програм се састоји из три дела. У првом делу се из периферија **PER0** и **PER1** упоредо уносе два блока од по 100h података и смештају у меморијске локације од адресе A000h до адресе A0FFh за периферију **PER0** и од адресе B000h до адресе B0FFh за периферију **PER1**. Унос из периферије **PER0** се реализује са контролером периферије **KPER0** и DMA контролером **DMA0** у режиму преноса цео блок и унос из периферије **PER1** реализује са контролером периферије **KPER1** техником програмираног улаза са прекидом. У другом делу се позива процесура `Obrada` која врши одређену обраду над унетим подацима у блоковима од по 100h података у меморијским локацијама од адресе A000h до адресе A0FFh и од адресе B000h до адресе B0FFh и резултат који представља два блока од по 100h података смешта у меморијске локације од адресе C000h до адресе C0FFh и од адресе D000h до адресе D0FFh. У трећем делу се упоредо у периферију **PER2** са контролером периферије **KPER2** и DMA контролером **DMA2** у режиму преноса циклус по циклус шаље блок од 100h података прочитаних из меморијских локација од адресе C000h до адресе C0FFh и у периферију **PER3** са контролером периферије **KPER3** техником програмираног излаза са испитивањем спремности шаље блок од 100h података прочитаних из меморијских локација од адресе D000h до адресе D0FFh.

У првом делу програма је унос података из **PER0** и **PER1** у меморију реализован на сличан начин као унос података у првом делу програма у задатку 1.9, у коме се улаз из периферије **PER0** реализује са контролером периферије **KPER0** и DMA контролером **DMA0** у режиму преноса циклус по циклус и улаз из периферије **KPER1** техником програмираног улаза са прекидом. Разлика у овом задатку је да нема дела програма из задатка 1.9 којим се улаз из периферије **PER2** реализује са контролером периферије **KPER2** техником програмираног улаза са испитивањем спремности

Други део се реализује у процедури `Obrada` која је слична са процедурама `Obrada` у задацима 1.6 и 1.7 у којима се израчунава `C(i)=A(i)+B(i)`, док се у овом задатку израчунава `C(i)=A(i)+B(i)` и `D(i)=A(i)-B(i).`

У трећем делу се упоредо у периферију **PER2** са контролером периферије **KPER2** и DMA контролером **DMA2** у режиму преноса циклус по циклус шаље блок података на сличан начин као и у задатку 1.7 и у периферију **PER3** техником програмираног излаза са испитивањем спремности шаље блок података на сличан начин као у задатку 1.1.

1.11 ЗАДАТАК

Дат је рачунарски систем који чине процесор, меморија, контролер периферије **KPER0** са периферијом **PER0** и контролер периферије **KPER1** са периферијом **PER1** повезани магистралом.

Процесор је са једноадресним форматом инструкција. Од програмски доступних регистара постоје 16 битни акумулатор ACC, указивач на врх стека SP, указивач на табелу (IV табела) са адресама прекидних рутина IVTP и програмска статусна реч PSW. Типови података који се користе су 16 битне целобројне величине са знаком и без знака. Приликом извршавања инструкције преноса у акумулатор и аритметичких, логичких и померачких инструкција постављају се индикатори N, Z, C и V програмске статусне речи PSW. Механизам прекида је векторисан. При прекиду хардверски се на стеку чувају програмски бројач PC и програмска статусна реч PSW. Бројеви улаза у IV табелу су фиксни и то улаз 0 за **PER0** и улаз 1 за **PER1**. Меморијски и улазно-излазни адресни простори су раздвојени.

Меморијски адресни простор је величине 2^{16} меморијских локација, при чему је ширина меморијских локација 16 битова.

Улазно-излазни адресни простор је величине 2^{16} регистара, при чему је ширина регистара 16 битова. Регистри контролера периферија **KPER0** и **KPER1** се налазе у улазно-излазном адресном простору.

Контролери периферија **KPER0** и **KPER1** имају управљачке регистре (*Control*), статусне регистре (*Status*) и регистре података (*Data*) и то редом на адресама FF00h, FF01h и FF02h за контролер периферије **KPER0** и FF10h, FF11h и FF12h за контролер периферије **KPER1**. У регистрима контролера периферија највиши бит је означен са 15, а најнижи са 0. У управљачким регистрима *Control* бит 4 је бит *enable* којим се дозвољава прекид (0—маскиран, 1—дозвољен), бит 1 је бит *u/i* којим се којим се задаје смер преноса са периферијом (0—улаз, 1—излаз) и бит 0 је бит *start* којим се покреће извршавање операције (0—заустављен, 1—стартован). У статусним регистрима *Status* бит 0 је бит спремности *ready* (0—није спреман, 1—спреман). Са свим регистрима контролера периферија **KPER0** и **KPER1** могу да се реализују операције читања и уписа.

Адресне линије и линије података магистрале рачунара су широке по 16 битова.

Написати програм којим се упоредо учитавају низ A(i), где је i = 0, …, FFh, из **PER0** у меморијски блок који почиње од адресе 1000h и низ B(i), где је i = 0, …, FFh, из **PER1** у меморијски блок почев од адресе 1100h, затим сабирају унети низови B(i) = A(i) + B(i), где је i = 0, …, FFh, и резултујући низ B(i), где је i = 0, …, FFh, шаље у периферију **PER0**. Улаз из **PER0** реализовати са контролером периферије **KPER0** техником програмираног улаза са прекидом, улаз из **PER1** са контролером периферије **KPER1** техником програмираног улаза са испитивањем спремности, а излаз у **PER0** са контролером периферије **KPER0** техником програмираног излаза са прекидом.

Решење:

Периферије **PER0** и **PER1** се повезују на магистралу рачунарског система са контролерима периферија **KPER0** и **KPER1** на начин приказан на слици 1.

Адресе и структура регистара контролера **KPER0** и **KPER1** су дати на сликама 31 и 32, респективно.

KPER0 - Kontroler periferije **PER0**

Registar	Control	Stataus	Data
Adresa	FF00h	FF01h	FF02h

KPER1 - Kontroler periferije **PER1**

Registar	Control	Stataus	Data
Adresa	FF10h	FF11h	FF12h

Слика 31 Адресе регистара контролера

Слика 32 Структура регистара контролера

Тражени програм је приказан на слици 33.

```
;glavni program
;unos blokova podat iz periferije PER0 u memoriju sa KPER0 sa prekidom i iz
;periferije PER1 u memoriju sa KPER1 sa ispitivanjem bita spremnosti ready
  ;inicijalizacija prenosa iz periferije PER0 u memoriju sa KPER0 i iz
  ;periferije PER1 u memoriju sa KPER1
    ;veličina bloka podataka
         LOAD   #100h     ;upis vel. bloka podat. za PER0 i PER1 u ACC pa u
         STORE  MemCnt0   ;mem. lok. na adr. MemCnt0 i u
         STORE  MemCnt1   ;mem. lok. na adr. MemCnt1
    ;početne adrese blokova podataka
         LOAD   #1000h    ;upis adr. bloka podataka za PER0 preko ACC u
         STORE  MemAdr0   ;mem. lok. na adr. MemAdr0
         LOAD   #1100h    ;upis adr. bloka podat. za PER1 preko ACC u
         STORE  MemAdr1   ;mem. lok. na adr. MemAdr1
  ;postavljanje "semafor"-a za PER0 na "prenos u toku"
  ;(0—prenos nije u toku, 1—prenos u toku)
         LOAD   #1        ;upis vrednosti 1 preko ACC u
         STORE  MemSem0   ;mem. lok. na adr. MemSem0
  ;postavljanje smera prenosa za prekidnu rutinu za PER0 na "ulaz"
  ;(0–ulaz, 1–izlaz)
         LOAD   #0        ;upis vrednosti 0 preko ACC u
         STORE  MemDir0   ;mem. lok. na adr. MemDir0
  ;startovanje kontrolera periferije KPER0
         LOAD   #0011h    ;upis režima rada i startovanja (enable=1,
         OUT    FF00h     ;u/i=0, start=1)preko ACC u reg. Control KPER0
  ;startovanje kontrolera periferije KPER1
         LOAD   #0001h    ;upis režima rada i startovanja (enable=0,
         OUT    FF10h     ;u/i=0, start=1)preko ACC u reg. Control KPER1
  ;unos blokova podataka iz PER0 i PER1
```

```
  ;unos iz PER1 se realizuje u nastavku glavnog programa u petlji Loop1
  ;unos iz PER0 se realizuje u prekidnoj rutini periferije PER0
  ;na koju se prelazi iz petlje Loop1 na svaki prekid od PER0
    ;provera da li podatak može da se prenese iz registra Data KPER1
Loop1:   IN     FF11h     ;upis sadrž. reg. Status KPER1 u ACC
         AND    #01h      ;ispitivanje bita ready
         JZ     Loop1     ;povratak na Loop1 ukoliko je ready 0
    ;prenos podatka iz registra Data KPER1 u memorijsku lokaciju
         IN     FF12h     ;upis sadrž reg Data KPER1 preko ACC u mem. lok.
         STORE  (MemAdr1) ;na adresi datoj sadrž. mem. lok. na adr. MemAdr1
    ;inkrementiranje sadržaja memorijske lokacije MemAdr1
         INC    MemAdr1
    ;dekrementiranje sadržaja memorijske lokacije MemCnt1
         DEC    MemCnt1
    ;provera da li je prenet poslednji podatak sa PER1
         JNZ    Loop1     ;povratak na Loop1 ukoliko nije poslednji podatak
    ;zaustavljanje kontrolera periferije KPER1
         LOAD   #0        ;upis režima zaustavljanja (enable=0,
         OUT    FF10h     ;u/i=0, start=0)preko ACC u reg. Control KPER1
  ;čekanje u petlji Wait0u proverom vrednosti "semafor"-a za PER0 ukoliko
  ;tokom izvršavanja prethodne petlje Loop1 nije završen unos bloka
  ;podataka iz periferije PER0 u memoriju
  ;unos preostalih podataka sa PER0 u memoriju
  ;se realizuje u prekidnoj rutini periferije PER0
  ;na koju se prelazi iz petlje Wait0u na svaki preostali prekid od PER0
Wait0u:  LOAD   MemSem0   ;upis sadrž. mem.lok. sa adr. MemSem0 u ACC
         CMP    #0        ;ispitivanje da li je "semafor" postao 0
         JNZ    Wait0u    ;povratak na Wait0u ukoliko "semafor" nije 0
;obrada
         JSR    Obrada    ;skok na potprogram Obrada
;slanje bloka podataka iz memorije u periferiju PER0 sa KPER0 sa prekidom
  ;inicijalizacija prenosa iz memorije u periferiju PER0 sa KPER0
    ;veličina bloka podataka
         LOAD   #100h     ;upis veličine bloka podataka za PER0 preko ACC u
         STORE  MemCnt0   ;mem. lok. na adr. MemCnt0
    ;početna adresa bloka podataka
         LOAD   #1100h    ;upis adr. bloka podataka za PER0 preko ACC u
         STORE  MemAdr0   ;mem. lok. na adr. MemAdr0
  ;postavljanje "semafor"-a za PER0 na "prenos u toku"
  ;(0—prenos nije u toku, 1—prenos u toku)
         LOAD   #1        ;upis vrednosti 1 preko ACC u
         STORE  MemSem0   ;mem. lok. na adr. MemSem0
  ;postavljanje smera prenosa za prekidnu rutinu za PER0 na "izlaz"
  ;(0–ulaz, 1–izlaz)
         LOAD   #1        ;upis vrednosti 1 preko ACC u
         STORE  MemDir0   ;mem. lok. na adr. MemDir0
  ;startovanje kontrolera periferije KPER0
         LOAD   #13h      ;upis režima rada i startovanja (enable=1,
         OUT    FF00h     ;u/i=1, start=1)preko ACC u reg. Control KPER0
  ;čekanje u petlji Wait0i proverom vrednosti "semafor"-a za perif PER0
  ;dok prenos podataka iz memorije u periferiju PER0 ne bude završen
  ;slanje iz memorije u PER0 se realizuje u prekidnoj rutini periferije
  ;PER0 na koju se prelazi iz petlje Wait0i na svaki prekid od KPER0
Wait0i:  LOAD   MemSem0   ;upis sadrž. mem.lok. sa adr. MemSem0 u ACC
         CMP    #0        ;ispitivanje da li je "semafor" postao 0
         JNZ    Wait0i    ;povratak na Wait0i ukoliko "semafor" nije 0
         . . .
```

```
;prekidna rutina periferije PER0
  ;unos bloka podataka iz periferije PER0 u memoriju ili
  ;slanje bloka podataka iz memorije u periferiju PER0
  ;u zavisnosti od sadrž. mem.lok. sa adr. MemDir0 sa smerom prenosa za
  ;prekidnu rutinu za PER0 (0—ulaz, 1—izlaz)
        PUSHA           ;akumulator ACC na stek
        LOAD   MemDir0  ;upis sadrž. mem.lok. sa adr. MemDir0 u ACC
        CMP    #1       ;ispitivanje da li je sadržaj 1
        JZ     Out      ;prelazak na labelu Out ukoliko je 1 i
                        ;prelazak na labelu In ukoliko je 0
  ;Ulaz - prenos podatka iz registra Data KPER0 u memorijsku lokaciju
In:     IN     FF02h    ;upis sadrž reg Data KPER0 preko ACC u mem lok
        STORE  (MemAdr0) ;na adresi datoj sadrž mem lok na adr MemAdr0
        JMP    Inc      ;prelazak na labelu Inc
  ;Izlaz - prenos podatka iz memorijske lokacije u registar Data KPER0
Out:    LOAD   (MemAdr0) ;upis sadrž. mem. lok. sa adr. date sadrž. mem.
        OUT    FF02h    ;lok. sa adr. MemAdr0 preko ACC u reg. Data KPER0
  ;inkrementiranje sadržaja memorijske lokacije MemAdr0
Inc:    INC    MemAdr0
  ;dekrementiranje sadržaja memorijske lokacije MemCnt0
        DEC    MemCnt0
  ;provera da li je prenet poslednji podatak
        JNZ    Back0    ;prelaz na Back0 ukoliko nije poslednji podatak
  ;postavljanje "semafor"-a za PER0 na 0
        LOAD   #0       ;upis vrednosti 0 preko ACC u
        STORE  MemSem0  ;mem. lok. na adr. MemSem0
  ;zaustavljanje kontrolera periferije KPER0
        LOAD   #0       ;upis režima zaustavljanja (enable =0,
        OUT    FF00h    ;u/i=0, start =0)preko ACC u reg. Control KPER0
  ;izlazak iz prekidne rutine
Back0:  POPA            ;restauriranje sadržaja ACC vrednošću sa steka
        RTI             ;povratak u glavni program iz prekidne rutine
        . . .
;Procedura Obrada u kojoj se izračunava B(i)=A(i)+B(i), gde je i=0,…, FFh
;Elementi niza A(i), gde je i=0,…, FFh, su na adresama 1000h do 10FFh,
;elementi niza B(i), gde je i=0,…, FFh, su na adresama 1100h do 11FFh, a
;elementi niza sume A(i)+B(i), gde je i=0,…, FFh, se smeštaju u B(i), gde
je ;i=0,…, FFh, na adresama 1100h do 11FFh
Obrada: LOAD   #100h    ;upis vel. blok. podat. za A(i) i B(i) u ACC pa u
        STORE  MemCnt1  ;mem. lok. na adr. MemCnt1
        LOAD   #1000h   ;upis adr. bloka podataka A(i)preko ACC u
        STORE  MemAdr0  ;mem. lok. na adr. MemAdr0
        LOAD   #1100h   ;upis adr. bloka podataka B(i)preko ACC u
        STORE  MemAdr1  ;mem. lok. na adr. MemAdr0
Loop:   LOAD   (MemAdr0) ;upis A(i) iz mem lok sa adrese date sadr
                        ;mem lok sa adrese MemAdr0 u ACC
        ADD    (MemAdr1) ;suma ACC i B(i) iz mem lok sa adrese date sadr
                        ;mem lok sa adrese MemAdr1 u ACC
        STORE  (MemAdr1) ;sadržaj ACC u B(i) u mem lok na adr datoj sadr
                        ;mem lok na adresi MemAdr1
        INC    MemAdr0  ;inkrementiranje sadr mem lok MemAdr0
        INC    MemAdr1  ;inkrementiranje sadr mem lok MemAdr1
        DEC    MemCnt1  ;dekrementiranje sadr mem lok MemCnt1
  ;provera da li je poslednja suma
        JNZ    Loop     ;povratak na Loop ukoliko nije poslednja suma
        RTS             ;povratak u glavni program
. . .
```

Слика 33 Програм

Програм се састоји из три дела.

У првом делу се из периферија **PER0** и **PER1** уносе два блока од по 100h података и смештају у меморијске локације од адресе 1000h до адресе 10FFh за периферију **PER0** и од адресе 1100h до адресе 11FFh за периферију **PER1**, при чему се унос података из периферије **PER0** реализује техником програмираног улаза са прекидом, а из периферије PER1 техником програмираног улаза са испитивањем бита спремности *ready* статусног регистра *Status* контролера периферије **KPER1**. У другом делу се позива процесура `Obrada` која врши обраду над унетим подацима у блоковима од по 100h података у меморијским локацијама од адресе 1000h до адресе 10FFh и од адресе 1100h до адресе 11FFh и резултат блок од 100h података смешта у меморијске локације од адресе 1100h до адресе 11FFh. У трећем делу се у периферију **PER0** техником програмираног излаза са прекидом шаље блок од 100h података прочитаних из меморијских локација од адресе 1100h до адресе 11FFh.

Уноси података из периферија **PER0** и **PER1** у меморију се реализују упоредо. Због тога се на почетку првом дела главног програма иницијализују преноси из периферија **PER0** и **PER1** у меморију, само за периферију **PER0** се постављају "semafor" на "prenos u toku" (MemSem0=1) и смер преноса на "ulaz" (MemDir0=0) и стартују контролери периферија **KPER0** и **KPER1**. Потом се у главном програму прелази на извршавање петље Loop1 у којој се техником програмираног улаза са испитивањем бита спремности *ready* статусног регистра *Status* контролера периферије **KPER1** уноси блок података из периферије **PER1** у меморију на начин објашњен у задатку 1.1, а по завршеном преносу и изласку из петље Loop1 зауставља контролер периферије **KPER1**. Током извршавања овог дела главног програма за сваки податак, који техником програмираног улаза са прекидом треба пренети из периферије **PER0** у меморију, од контролера периферије **KPER0** долази захтев за прекид, па се на сваки захтев за прекид извршавање овог дела програма прекида, прелази на прекидну рутину периферије **PER0** ради преноса једног податка из регистра *Data* **KPER0** у меморију и враћа у прекинути главни програм. Приликом преноса последњег податка у прекидној рутини периферије **PER0** се "semafor" периферије **PER0** поставља на "prenos nije u toku" (MemSem0=0) и зауставља контролер периферије **KPER0**. Унос блока података из периферије **PER0** техником програмираног улаза са прекидом се реализује на начин објашњен у задатку 1.3.

Прелазак на други део програма у коме се позива процедура `Obrada` која врши обраду над унетим подацима не сме да се дозволи док се у меморију не унесу блокови података из обе периферије. Стога се у главном програму процедура `Obrada` позива тек пошто се изађе најпре из петље са лабелом Loop1, а затим и из петље са лабелом Wait0u. Из петље са лабелом Loop1 се излази тек пошто се у њој заврши унос комплетног блока од 100h података из периферије **PER1** у меморију, док је петља са лабелом Wait0u предвиђена само за евентуално чекање завршетка уноса блока од 100h података из периферије **PER0** у меморију који се реализује у прекидној рутини периферије **PER0**. Том приликом могу да се јаве два случаја у зависности од брзине периферија **PER0** и **PER1**. Први случај се јавља када је периферија **PER1** бржа од периферије **PER0**, па ће се унос блока података из периферије **PER1** у петљи са лабелом Loop1 завршити пре завршетка уноса блока података из периферије **PER0**. Стога ће се изаћи из петље са лабелом Loop1 пре него што се стигло до последњег преласка на прекидну рутину периферије **PER0** када се "semafor" периферије **PER0** поставља на вредност 0. Због тога ће се по изласку из петље са лабелом Loop1 најпре у петљи са лабелом Wait0u чекати да се заврши и унос података из периферије PER0 и да се "semafor" периферије PER0 поставити на вредност 0 и затим прећи на други део програма у коме се позива процедура `Obrada`. Други случај се јавља када је периферија **PER1** спорија од периферије **PER0**. Стога ће се унос података из периферије **PER0**

завршити пре завршетка уноса података из периферије **PER1** и прво "semafor" периферије **PER0** поставити на вредност 0 а затим изаћи из петље са лабелом Loop1. Због тога се по изласку из петље са лабелом Loop1 у петљи са лабелом Wait0u неће чекати, јер ће унос података из периферије **PER0** бити завршен и "semafor" периферије **PER0** постављен на вредност 0, па ће се одмах прећи на други део програма у коме се позива процедура `Obrada`.

Други део у коме се врши обрада се реализује позивом на процесуру `Obrada`.

У трећем делу се техником програмираног излаза са прекидом шаље блок од 100h података из меморијских локација од адресе 1100h до адресе 11FFh у периферију **PER0**. Због тога се на почетку трећег дела главног програма иницијализује пренос из меморије у периферију **PER0**, постављају "semafor" на "prenos u toku" (MemSem0=1) и смер преноса на "izlaz" (MemDir0=1) и стартује контролер периферије **KPER0**. Потом се у главном програму прелази на извршавање петље са лабелом Wait0i у којој се чека на завршетак слања блока од 100h података из меморије у периферију **PER0**. Током извршавања петље са лабелом Wait0i за сваки податак, који техником програмираног излаза са прекидом треба пренети из меморије у периферију **PER0**, од контролера периферије **KPER0** долази захтев за прекид, па се на сваки захтев за прекид извршавање петље са лабелом Wait0i прекида, прелази на прекидну рутину периферије **PER0** ради преноса једног податка из меморије у регистар *Data* **KPER0** и враћа у прекинуту петљу са лабелом Wait0i. Приликом преноса последњег податка у прекидној рутини периферије **PER0** се "semafor" периферије **PER0** поставља на "prenos nije u toku" (MemSem0=0) и зауставља контролер периферије **KPER0**. Слање блока података из меморије у периферију PER0 техником програмираног излаза са прекидом се реализује на начин објашњен у задатку 1.2.

Постављањем "semafor"-а периферије **PER0** на "prenos nije u toku" (MemSem0=0) омогућава се излаз из петље са лабелом Wait0i и прелазак на извршавање остатка главног програма.

Прекидна рутина периферије **PER0** је улазно излазна и служи у првом делу главног програма за унос блока података из периферије **PER0** у меморију и у трећем делу главног програма за слање блока података из меморије у периферију **PER0**. Прекидне рутине за унос блока податак из периферије у меморију и слање блока података из меморије у периферију се разликују само у оном делу у коме се реализује сам пренос из регистра *Data* **KPER0** у меморијску локацију чија је адреса дата садржајем меморијске локације на адреси MemAdr0 и пренос садржаја из меморијске локације чија је адреса дата садржајем меморијске локације на адреси MemAdr0 у регистар *Data* **KPER0**, респективно. Због тога се приликом стартовања контролера периферије у зависности од тога да ли се контролер периферије стартује за режим улаза или излаза у меморијску локацију MemDir0 уписује вредност 0 или 1, а на почетку прекидне рутине се проверава садржај меморијске локације MemDir0 и прелази или на лабелу In или на лабелу Out, респективно.

У процедури `Obrada` се најпре меморијске локације MemCnt1, MemAdr0 и MemAdr1 иницијализују вредностима 100h, 1000h и 1100h које представљају величину блокова података над којима треба извршити обраду и почетне адресе два блока података над којима треба извршити обраду, респективно. Потом се у петљи Loop из меморије читају парови података из меморијских локација од адресе 1000h до адресе10FFh и од адресе 1100h до адресе11FFh и сума уписује у меморијске локације од адресе 1100h до адресе11FFh, инкрементирају садржаји меморијских локација MemAdr0 и MemAdr1 и декрементира садржај меморијске локације MemCnt1 и на крају у зависности од тога да ли је садржај меморијске локације MemCnt1 после

декрементирања још увек различит од нуле или је постао нула или остаје у петљи Loop или излази из петље Loop, респективно. На крају се враћа у главни програм.

Дискусија:

Специфичност овог задатка је у томе што је потребно најпре техником програмираног улаза са прекидом из **PER0** у меморију унети блок података, а затим техником програмираног излаза са прекидом из меморије у **PER0** послати блок података. Контролер периферије **PER0** само по једној линији шаље процесору захтев за прекид, без обзира на то да ли је иницијализован за режим улаз и режим излаза, па је приликом прекида, без обзира на то да ли се ради о режиму улаза или режиму излаза, могућ прелазак само на једну прекидну рутину. Стога је неопходно да се у самој прекидној рутини периферије **PER0** софтверски утврди да ли је у току операција улаза или излаза и да се на основу тога врши пренос из периферије у меморију или из меморије у периферију, респективно. Најједноставније решење, које је и реализовано у задатку, је да се двема вредностима променљиве MemDir0 одреди да ли је реч о операцији улаза или излаза. У том случају је потребно да се у главном програму приликом иницијализације контролера периферије у променљиву MemDir0 упише одговарајућа вредност и да се у прекидној рутини испитује променљива MemDir0.

Могуће решење је и да се не користи променљива MemDir0, већ да се у прекидној рутини испитује вредност бита *u/i* (0—улаз, 1—излаз) управљачког регистра *Control* контролера периферије. Ово је могуће урадити јер је у поставци задатка речено да је са регистрима контролера периферија могуће реализовати и читање и упис. Такође је могуће и да се формирају две посебне прекидне рутине за **PER0** и то једна за улаз а друга за излаз. У том случају је потребно, због тога што за периферију **PER0** постоји само једна улаз у IV табелу, да се приликом иницијализације контролера периферије **KPER0** у улаз IV табеле додељен периферији **PER0** упише адреса одговарајуће прекидне рутине.

Друга ствар на коју треба обратити пажњу је да улазне операције са периферија **PER0** и **PER1** треба обављати упоредо. Док је у главном програму у току пренос са **PER1**, контролер периферије **KPER0** генерише прекиде да би се преласцима на прекидну рутину периферије **PER0** обављао пренос једног по једног податка са **PER0**. На овај начин се паралелизује рад две периферије и њихових контролера, док сам пренос из контролера периферија **KPER0** и **KPER1** у меморију врши процесор секвенцијално. Иако нема паралелизације самог преноса из контролера периферија **KPER0** и **KPER1** у меморију, јер то ради процесор, и то за периферију **PER1** у главном програму а за периферију **PER0** у прекидној рутини, постиже се независност рада две периферије и њихових контролера. јер се ничиме не условљава међусобни редослед операција са периферија **PER0** и **PER1**.

Улазно-излазни и меморијски адресни простори су раздвојени, процесор је са једноадресним форматом инструкција, а меморијске локације и регистри контролера периферија су дужине 16 битова. Због тога се све инструкције извршавају над величинама дужине 16 битова, а за приступ регистрима контролера периферија користе се само инструкције IN и OUT.

1.12 ЗАДАТАК

Дат је рачунарски систем који чине процесор, меморија, контролер периферије **KPER0** са периферијом **PER0** и контролер периферије **KPER1** са периферијом **PER1** повезани магистралом.

Процесор је са једноадресним форматом инструкција. Од програмски доступних регистара постоје 16 битни акумулатор ACC, указивач на врх стека SP, указивач на табелу (IV табела) са адресама прекидних рутина IVTP и програмска статусна реч PSW. Типови података који се користе су 16 битне целобројне величине са знаком и без знака. Приликом извршавања инструкције преноса у акумулатор и аритметичких, логичких и померачких инструкција постављају се индикатори N, Z, C и V програмске статусне речи PSW. Механизам прекида је векторисан. При прекиду хардверски се на стеку чувају програмски бројач PC и програмска статусна реч PSW. Меморијски и улазно-излазни адресни простори су раздвојени.

Меморијски адресни простор је величине 2^{16} меморијских локација, при чему је ширина меморијских локација 16 битова.

Улазно-излазни адресни простор је величине 2^{16} регистара, при чему је ширина регистара 16 битова. Регистри контролера периферија **KPER0** и **KPER1** се налазе у улазно-излазном адресном простору.

Контролери периферија **KPER0** и **KPER1** имају управљачке регистре (*Control*), статусне регистре (*Status*) и регистре података (*Data*) и то редом на адресама FF00h, FF01h и FF02h за контролер периферије **KPER0** и FF10h, FF11h и FF12h за контролер периферије **KPER1**. У регистрима контролера периферија највиши бит је означен са 15, а најнижи са 0. У управљачким регистрима *Control* бит 2 је бит *enable* којим се дозвољава прекид (0—маскиран, 1—дозвољен), бит 1 је бит *u/i* којим се којим се задаје смер преноса са периферијом (0—улаз, 1—излаз) и бит 0 је бит *start* којим се покреће извршавање операције (0—заустављен, 1—стартован). У статусним регистрима *Status* бит 4 је бит спремности *ready* (0—није спреман, 1—спреман).

Адресне линије и линије података магистрале рачунара су широке по 16 битова.

Написати програм којим се учитава низ A(i), где је i = 0, …, FFh, из **PER0** у меморијски блок који почиње од адресе 2000h и чим је то могуће срачунава A(i)*A(i), где је i = 0, …, FFh, и шаље у **PER1**. Улаз из **PER0** реализовати са контролером периферије **KPER0** техником програмираног улаза са прекидом, а излаз у **PER1** са контролером периферије **KPER1** техником програмираног излаза са испитивањем спремности.

Решење:

Периферије **PER0** и **PER1** се повезују на магистралу рачунарског система са контролерима периферија **KPER0** и **KPER1** на начин приказан на слици 1.

Адресе и структура регистара контролера **KPER0** и **KPER1** су дати на сликама 34 и 35, респективно.

KPER0 - Kontroler periferije **PER0**

Registar	Control	Stataus	Data
Adresa	FF00h	FF01h	FF02h

KPER1 - Kontroler periferije **PER1**

Registar	Control	Stataus	Data
Adresa	FF10h	FF11h	FF12h

Слика 34 Адресе регистара контролера

Слика 35 Структура регистара контролера

Тражени програм је приказан на слици 36.

```
;glavni program
;unos bloka podat iz periferije PER0 u memoriju sa KPER0 sa prekidom i
;slanje bloka podat iz memorije u periferiju PER1 sa KPER1 sa ispitivanjem
;bita spremnosti ready
  ;inicijalizacija prenosa iz periferije PER0 u memoriju sa KPER0 i
  ;iz memorije u periferiju PER1 sa KPER1
    ;veličina bloka podataka
         LOAD   #100h     ;upis vel. bloka podat. za PER0 i PER1 u ACC pa u
         STORE  MemCnt0   ;mem. lok. na adr. MemCnt0 i u
         STORE  MemCnt1   ;mem. lok. na adr. MemCnt1
    ;početna adresa bloka podataka
         LOAD   #2000h    ;upis adr. bloka podataka preko ACC u
         STORE  MemAdr0   ;mem. lok. na adr. MemAdr0 za PER0 i u
         STORE  MemAdr1   ;mem. lok. na adr. MemAdr1 za PER1
  ;startovanje kontrolera periferije KPER0
         LOAD   #0005h    ;upis režima rada i startovanja (enable=1,
         OUT    FF00h     ;u/i=0, start=1)preko ACC u reg. Control KPER0
  ;startovanje kontrolera periferije PER1
         LOAD   #0003h    ;upis režima rada i startovanja (enable=0,
         OUT    FF10h     ;u/i=1, start=1)preko ACC u reg. Control KPER1
```

```
  ;unos bloka podataka sa PER0, sračunavanje proizvoda i slanje u PER1
  ;unos sa PER0 se realizuje u prekidnoj rutini periferije PER0
  ;na koju se prelazi iz spoljašnje petlje Loop1 na svaki prekid od PER0
  ;slanje na PER1 se realizuje u nastavku glavnog programa u spoljašnjoj
  ;petlji Loop1 ukoliko kontroler KPER1 može da primi podatak u registar
  ;Data KPER1 i ukoliko procesor ima uneti podatak iz PER0 pa može da
  ;sračuna proizvod pošalje ga u registar Data KPER1
    ;provera da li kontroler KPER1 može da primi podat u reg Data KPER1
Loop1:   IN     FF11h    ;upis sadrž. reg. Status KPER1 u ACC
         AND    #0010h   ;ispitivanje bita ready
         JZ     Loop1    ;povratak na Loop1 ukoliko je ready 0
    ;provera da li procesor ima podatak za slanje u registar Data KPER1
Loop0:   LOAD   MemAdr0  ;upis sadrž. mem lok sa adr MemAdr0 u ACC
         SUB    MemAdr1  ;razlika sadrž ACC i mem lok sa adr MemAdr1 u ACC
         JLE    Loop0    ;povratak na Loop0 ukoliko sadr mem lok sa adr
                         ;MemAdr0 nije veći od sadr mem lok sa adr MemAdr1
    ;sračunavanja A(i)*A(i) i slanje u registar  Data KPER1
         LOAD   (MemAdr1);upis sadrž. mem lok sa adr MemAdr1 u ACC
         MUL    (MemAdr1);proizvod sadr ACC i mem lok sa adr MemAdr1 u ACC
         OUT    FF12h    ;sadržaj ACC u registar  Data KPER1
    ;inkrementiranje sadržaja memorijske lokacije MemAdr1
         INC    MemAdr1  ;inkrementiranje sadr mem lok MemAdr1
    ;dekrementiranje sadržaja memorijske lokacije MemCnt1
         DEC    MemCnt1  ;dekrementiranje sadr mem lok MemCnt1
    ;provera da li je poslednji proizvod
         JNZ    Loop1    ;povr na Loop1 ukoliko nije poslednji proizvod
  ;zaustavljanje kontrolera periferije KPER1
         LOAD   #0       ;upis režima zaustavljanja (enable =0,
         OUT    FF10h    ;u/i=0, start =0)preko ACC u reg. Control KPER1
. . .
;prekidna rutina periferije PER0
  ;unos bloka podataka iz periferija PER0 u memoriju
         PUSHA           ;akumulator ACC na stek
  ;Unos podatka iz registra Data KPER0 u memorijsku lokaciju
         IN     FF02h    ;upis sadrž reg Data KPER0 preko ACC u mem lok
         STORE  (MemAdr0);na adresi datoj sadrž mem lok na adr MemAdr0
  ;inkrementiranje sadržaja memorijske lokacije MemAdr0
         INC    MemAdr0
  ;dekrementiranje sadržaja memorijske lokacije MemCnt0
         DEC    MemCnt0
  ;provera da li je prenet poslednji podatak
         JNZ    Back0    ;prelaz na Back0 ukoliko nije poslednji podatak
  ;zaustavljanje kontrolera periferije KPER0
         LOAD   #0       ;upis režima zaustavljanja(enable =0,
         OUT    FF00h    ;u/i=0, start =0)preko ACC u reg. Control KPER0
  ;izlazak iz prekidne rutine
Back0:   POPA            ;restauriranje sadržaja ACC vrednošću sa steka
         RTI             ;povratak u glavni program iz prekidne rutine
. . .
```

Слика 36 Програм

Програм се састоји из главног програма и прекидне рутине периферије **PER0**.

Главни програм се састоји из два дела.

У првом делу главног програма се врши иницијализација уноса података из **PER0** у меморију и слања блока података из меморије у периферију **PER1** и стартовање контролера периферије **KPER0** за унос са прекидом и контролера периферије **KPER1** за слање са испитивањем спремности. У оквиру иницијализације се у меморијске локације `MemCnt0` и `MemCnt1` уписују величине блокова података, а у меморијске локације `MemAdr0` и `MemAdr1` почетне адресе блокова података.

У другом делу главног програма и то у спољашњој петљи са лабелом Loop1 и прекидној рутини периферије **PER0** се упоредо врши унос података из периферије **PER0,** срачунавање производа елемената низа, слање података у периферију **PER1** и заустављање контролера периферије **KPER1**. Унос података из **PER0** са прекидом и слање података у **PER1** са испитивањем спремности се реализују на сличан начин као у задатку 1.3. Том приликом процесор и контролери **KPER0** и **KPER1** раде паралелно**.**

Процесор извршава спољашњу петљу са лабелом Loop1 која има три дела. У првом делу процесор остаје у унутрашњој петљи са лабелом Loop1 све време док контролер периферије **KPER1** не заврши слање текућег податка из регистра *Data* **KPER1** у периферију **PER1** и не постане спреман да прими следећи податак у регистар *Data* **KPER1**. У другом делу процесор остаје у петљи са лабелом Loop0 уколико контролер **KPER0** није пренео податак из периферије **PER0** у регистар *Data* **KPER0** и процесор у прекидној рутини периферије **PER0** из регистра *Data* **KPER0** у меморију за који у трећем делу процесор може да израчуна производ и пошаље у регистар *Data* **KPER1.** У трећем делу процесор за унети податак из **PER0** израчунава производ и шаље у регистар *Data* **KPER1**, инкрементира садржај MemAdr1, декрементира садржај MemCnt1 и у зависности од тога да ли је после декрементирања MemCnt1 различит од 0 или једнак 0 или остаје у спољашњој петљи са лабелом Loop1 и продужава са слањем података у **PER1** или излази из спољашње петље са лабелом Loop1 и заустављањем контролера **KPER1** завршава слање података у **PER1**.

На свако уписивање податка од стране процесора у регистар *Data* **KPER1**, контролер **KPER1** креће са слањем податка из регистра *Data* **KPER1** у периферију **PER1.**

На сваки податак који контролер **KPER0** унесе из периферије **PER0** у регистар *Data* **KPER0**, контролер **KPER0** генерише прекид, па процесор прекида извршавање спољашње петље са лабелом Loop1 и прелази на прекидну рутину периферије **PER0**. У прекидној рутини периферије **PER0** процесор уноси податак из *Data* **KPER0** у меморијску локацију, инкрементира садржај MemAdr0, декрементира садржај MemCnt0 и у зависности од тога да ли је после декрементирања MemCnt0 различит од 0 или једнак 0 или процесор се одмах враћа у прекинути главни програм и контролер **KPER0** започиње пренос следећег податка из **PER0** у регистар *Data* **KPER0** или процесор најпре зауставља контролер **KPER0** и тиме завршава унос података из **PER0** и затим враћа у прекинути главни програм.

2 ЛИТЕРАТУРА

1. B. Lazić, *Logičko projektovanje računara*, Nauka—Elektrotehnički fakultet, Beograd, 1994.

2. D. Živković, M. Popović, *Impulsna и digitalna elektronika*, Nauka—Elektrotehnički fakultet, Beograd, 1992.

3. J. Djordjevic, A. Milenkovic, N. Grbanovic, *"An Integrated Educational Environment for Teaching Computer Architecture and Organisation, "* IEEE MICRO, May 2000.pp. 66-74.

4. J. Djordjevic, M. R. Barbacci, B. Hosler, *A PMS Level Notation for the Description and Simulation of Digital Systems*, The Computer Journal, Vol. 28, No. 4, pp. 357-365, 1985.

5. S. Miladinović, J. Đorđević, A. Milenković, *Programski sistem за grafički opis и simulaciju digitalnih sistema*, Zbornik radova ETRAN 1997, Zlatibor, Jugoslavija, Jun 1997.

6. N. Grbanovic, J. Djordjevic, B. Nikolić, *The Software Package for an Educational Computer System*, International Journal он Electrical Engineering Education, Vol. 40, No. 4, Oct 2003, pp. 270-284.

7. J. Djordjevic, A. Milenkovic, I. Todorovic, D. Marinov, "CALKAS: *A Computer Architecture Learning and Knowledge Assessment System, "* IEEE TC Computer Architecture Newsletter, March 1999.

8. J. Đorđević, *Priručnik из arhitekture računara*, Elektrotehnički fakultet, Beograd, 1997.

9. J. Đorđević, *Priručnik из arhitekture и organizacije računara*, Elektrotehnički fakultet, Beograd, 1997.

10. J. Đorđević, *Arhitektura računara*, *Edukacioni računarski sistem, Arhitektura и organizacija računrskog sistema*, Elektrotehnički fakultet, Beograd, 2002.

11. J. Đorđević, N. Grbanović, B. Nikolić, Z. Radivojević, *Arhitektura računara*, *Edukacioni računarski sistem, Priručnik за simulaciju са zadacima*, Elektrotehnički fakultet, Beograd, 2004.

12. J. Djordjevic, B. Nikolic, A. Milenkovic, "*Flexible Web-based Educational System for Teaching Computer Architecture and Organization, "* IEEE, Transactions он Education, Vol. 48, No. 2, 2005.

13. J. Djordjevic, B. Nikolic, M. Mitrovic, "*A Memory System for Education, "* Computer Journal, (to appear)