Fakulteta za računalništvo in informatiko

SEMINARSKA NALOGA
IZ PREDMETA
ARS 2

17. 5. 1999

1. UVOD

Naša naloga je bila napisati program, ki naj bi:
- v najslabšem primeru med dvema ploščicama (računalnikoma) v eno smer prenesel 1 bajt ter uporabljal prekinitve in usklajevalno delovanje na vsaj eni ploščici;
- v najboljšem primeru v obe smeri ves čas prenašal podatke, imel možnost menjave smeri med delovanjem in uporabljal usklajevalno delovanje na obeh ploščicah - to smo naredili mi.

Uporabili smo Motorolin procesor 6802, ki ima že vgrajenih 128 kilobajtov RAM-a, nanj pa je bil priključen še emulator EPROM-a in periferni vmesniški adapter PIA - 6821.

PIA je bila priključena takole:
- na strani A sta bili na nožici PA0 in PA1 priključeni dve stikali, na nožice PA2-PA7 pa so bile priključene podatkovne povezave z druge ploščice, ki je vsebovala iste elemente. Kontrolna signala CA1 in CA2 sta bila vezana tako, da je bil signal CA2 vezan na signal CA1 na nasprotni ploščici in obratno;
- na strani B so bile vse nožice perifernega vmesnika vezane na LED diode, kontrolna signala CB1 in CB2 pa nista bila vezana nikamor.

Delovala je v usklajevalnem načinu:
branje registra DRA na prvi ploščici povzroči aktiviranje signala CA2, ki je priključen na nožico CA1 na drugi ploščici. To na njej sproži prekinitev, med katero se bere register DRA te ploščice, kar povzroči prekinitev na prvi ploščici itd. Tako ni treba nadzorovati, kdaj se podatki preberejo in kdaj se pojavijo novi - ves postopek teče sam od sebe.

2. DELOVANJE RAČUNALNIKA

Testiramo ga takole: emulatorja EPROM-a, ki sta priključena na ploščici s procesorjema, prek tiskalniških vrat povežemo z dvama računalnikoma. Ko ploščici povežemo med sabo in pripravimo za delo, v oba emulatorja naložimo naš program.

Pred zagonom programa moramo poskrbeti, da sta drugi stikali na obeh ploščicah v različnih položajih, saj z njima določimo, katera ploščica je sprejemnik in katera pošiljatelj. Njuni vlogi zamenjamo tako, da najprej na sprejemniku, nato pa še na pošiljatelju preklopimo drugo stikalo. Ob spremembi smeri prenosa se prižge posebna kombinacija lučk. Da program v resnici deluje, vidimo tako, da se lučke na obeh prižigajo v enakem zaporedju. S prvim stikalom pa lahko spremenimo zaporedje njihovega prižiganja in sicer tako, da se lučke namesto z leve proti desni prižigajo ena za drugo z desne proti levi (ali obratno).

3. ZGRADBA PROGRAMA

Uporaba pomnilniških lokacij	
Naslov	*Namen*
$0000	$0F - poslati desno polovico bajta, $F0 - poslati levo polovico bajta
$0001	trenutno stanje lučk
$0002	stanje stikal
$0003	%0000000000 - sprejemnik, %00000010 - pošiljatelj
$0004	$00 - nič naj se ne zgodi, $0F - postane naj sprejemnik, $F0 - postane naj pošiljatelj

Stran A pri PIA (DRA)	
Bit	*Namen*
0	prvo stikalo: 0 - lučke se premikajo v levo, 1 - lučke se premikajo v desno
1	drugo stikalo: 0 - sprejemnik, 1 - pošiljatelj
2	0 - nič naj se ne zgodi, 1 - sprejemnik naj postane pošiljatelj
3	0 - pošilja se leva polovica, 1 - pošilja se desna polovica
4-7	podatki

Opomba: oznake iz programa so kurzivne.

Pri *resetu* se najprej inicializira PIA v neusklajevalnem načinu (*inip*). Prebere se stanje stikal in tako določi smer prenašanja podatkov (ta se zapiše v pomnilnik na lokacijo $0003). Temu primerno se še enkrat inicializira PIA - tokrat v usklajevalnem načinu (*inipp* - pošiljatelj, *inips* - sprejemnik). Če je ploščica pošiljatelj, prvo levo polovico bajta zapiše na izhod PIA, nato pa jo prebere, kar pri sprejemniku sproži prekinitev - *inipos*. Nato se se v *zanki* ugotavlja, če je potrebno spremeniti smer prenašanja (bere se pomnilniška lokacija $0004). Kadar pride prekinitev (*irq*), se najprej z lokacije $0003 prebere, če je ploščica pošiljatelj ali sprejemnik. V prvem primeru se v *irqp* z lokacije $0000 prebere, ali je na vrsti pošiljanje leve ali desne polovice bajta. Če je na na vrsti leva polovica, se z lokacije $0002 naloži stanje stikal, ugotovi položaj prvega stikala in v skladu z njim se lučke premaknejo v levo ali desno (kadar prižgana lučka pride do konca, je treba prižgati lučko na drugi strani, saj ukaza ROL in ROR taga ne naredita sama, ker se carry med prekinitvami zbriše). V obeh primerih se nato stanje lučk shrani nazaj na lokacijo $0001 in zapiše na izhod PIA, nato pa od tam prebere, da se pri sprejemniku sproži prekinitev. Če se pri branju ugotovi, da je bilo preklopljeno drugo stikalo in je potrebno zamenjati smer prenašanja, se napotek za to zapiše v pomnilnik na lokacijo $0004 in se upoštava v *zanki*. Shranita se tudi stanje stikal ($0002) in napotek, katera polovica bajta naj se pošlje naslednjič ($0000). V drugem primeru se v *irqs* prebere poslani podatek. Najprej se ugotovi, katera polovica je bila poslana (bit 3). Če je bila leva, se zgolj shrani v pomnilnik na lokacijo $0001. Če je bila desna, se najprej ugotovi, ali je potrebno zamenjati smer prenašanja (bit 2). Če je, se se napotek za to zapiše v pomnilnik in se upoštava v *zanki*. Sicer pa se poslano združi s shranjeno levo polovico in se zapiše na LED diode. Kadar se v *zanki* ugotovi, da je treba spremeniti smer prenašanja, se najprej iz pomnilnika naloži podatek, v katero smer naj sprememba bo, nato pa se ponovno inicializira PIA in se, če se začenja

pošiljanje, pošlje prva polovica bajta (podobno ko pri *resetu*). Med spreminjanjem smeri so prekinitve onemogočene, da ne pride do kake zmešnjave.

4. KODA PROGRAMA

Opomba: vrstice, komentirane s ;!!! so bile izvirno namenjene le preizkušanju programa, a izkazalo se je, da brez njih ne deluje.

```
****************
*zacetne zadeve*
****************
STCKAD  EQU     $007F

        ORG     $DFF8
        FDB     IRQ         ;rezerviramo spomin za vektor IRQ
        FDB     SWI
        FDB     NMI
        FDB     HWRES       ;rezerviramo spomin za vektor reseta

*************
*V/I naslovi*
*************
PIAADR  EQU     $8004       ;zacetni naslov PIA
DRA     EQU     PIAADR      ;DRA ali DDRA
CRA     EQU     PIAADR+1    ;CRA
DRB     EQU     PIAADR+2    ;DRB ali DDRB
CRB     EQU     PIAADR+3    ;CRB

        ORG     $C000       ;tukaj se zares zacne program

*****************************************
*inicializacija PIA v neusklejevalnem nacinu*
*****************************************
INIP    CLRA
        STAA    CRA         ;dostopamo do DDRA
        STAA    CRB         ;dostopamo do DDRB
        CLRA
        STAA    DRA         ;A je vhod
        LDAA    #$FF
        STAA    DRB         ;B je izhod
        LDAA    #%00000100
        STAA    CRA         ;sprejemamo podatke (ne dostopamo do DDRB)
        STAA    CRB         ;posiljamo podatke (ne dostopamo do DDRA)
        RTS

*******************************************************
*inicializacije PIA v usklajevalnem nacinu za posiljanje*
*******************************************************
INIPP   CLRA
        STAA    CRA         ;dostopamo do DDRA
        STAA    CRB         ;dostopamo do DDRB
        LDAA    #%11111100
        STAA    DRA         ;najnizja bita A sta vhod, ostali izhod
        LDAA    #$FF
        STAA    DRB         ;B je izhod
        LDAA    #%00100101
        STAA    CRA         ;dostopamo do DRA (ne do DDRA), CA2 je izhod,
                            CA1 prozi prekinitve
        LDAA    #%00000100
        STAA    CRB         ;dostopamo do DRB (ne do DDRB)

        RTS
```

```
*********************************************************
*inicializacije PIA v usklajevalnem nacinu za sprejemanje*
*********************************************************
INIPS   CLRA
        STAA    CRA         ;dostopamo do DDRA
        STAA    CRB         ;dostopamo do DDRB
        LDAA    #$00
        STAA    DRA         ;A je vhod
        LDAA    #$FF
        STAA    DRB         ;B je izhod
        LDAA    #%00100101
        STAA    CRA         ;dostopamo do DRA (ne do DDRA), CA2 je izhod,
                            CA1 prozi prekinitve
        LDAA    #%00000100
        STAA    CRB         ;dostopamo do DRB (ne do DDRB)

        RTS

********************
*zacetek posiljanja*
********************
INIPOS  LDAA    #%01010101 ;!!!
        STAA    DRB        ;!!!
        JSR     CAKAJ      ;!!!
        LDAA    #%00000001 ;doloci zacetno stanje luck
        STAA    $0001      ;shrani trenutno stanje luck
        ANDA    #%11110111 ;posilja levo polovico (bit 3 je 0)
        STAA    DRA        ;v DRA poslje vsebino ACCA (uposteva se leva
                           polovica)
        LDAB    DRA        ;bere jo, da sprozi prekinitev in dobi stanje
                           stikal
        STAB    $0002      ;shrani stanje stikal
        LDAA    #$0F
        STAA    $0000      ;shrani napotek, da naslednjic poslje desno
                           polovico

        RTS

*********
*cakanje*
*********
CAKAJ   LDX     #$0000     ;IX na 0
ZCAKAJ  INX                ;poveca IX
        CPX     #$5000     ;pogleda, ce je IX prisel do 20480
        BEQ     KCAKAJ     ;ce je, konca s cakanjem

        JMP     ZCAKAJ     ;sicer pa nadaljuje v zanki

KCAKAJ  RTS

********************
*IRQ za posiljanje*
********************
IRQP    JSR     CAKAJ
        LDAA    $0000      ;nalozi napotek, kaj poslati
        CMPA    #$0F
        BEQ     VMES       ;ce desno polovico, gre na DESNOP (prej na VMES,
                           ker je DESNOP predalec za branch), sicer
                           nadaljuje
        JMP     NAPREJ
```

```
VMES    JMP     DESNOP

NAPREJ  LDAA    $0001       ;nalozi trenutno stanje luck
        LDAB    $0002       ;nalozi stanje stikal
        ANDB    #%00000001  ;ignorira vse razen 1. stikala
        CMPB    #%00000001
        BEQ     LUCKED      ;ce je prizgano, gre na LUCKED, sicer nadaljuje

        CMPA    #%10000000
        BEQ     OKROGL      ;ce je lucka prisla do konca, gre na OKROGL,
                            sicer nadaljuje

        CLC                 ;zbrise carry, da se lucke zagotovo pravilno
                            premaknejo
        ROLA                ;lucke premekne v levo
        JMP     POSLJI

OKROGL  LDAA    #%00000001  ;prizge desno lucko
        JMP     POSLJI

LUCKED  CMPA    #%00000001
        BEQ     OKROGD      ;ce je lucka prisla do konca, gre na OKROGD,
                            sicer nadaljuje

        CLC                 ;zbrise carry, da se lucke zagotovo pravilno
                            premaknejo
        RORA                ;lucke premekne v desno
        JMP     POSLJI

OKROGD  LDAA    #%10000000  ;prizge levo lucko

POSLJI  STAA    $0001       ;shrani trenutno stanje luck
        STAA    DRB         ;!!!
        ANDA    #%11110011  ;posilja levo polovico (bit 3 je 0), ne da
                            napotka, naj drugi racunalnik preklopi na
                            posiljanje (bit 2 je 0)
        STAA    DRA         ;v DRA poslje vsebino ACCA (upostteva se leva
                            polovica)
        LDAB    DRA         ;bere jo, da sprozi prekinitev in dobi stanje
                            stikal
        STAB    $0002       ;shrani stanje stikal
        ANDB    #%00000010  ;ignorira vse razen 2. stikala
        STAB    $0003       ;shrani stanje posiljanja/sprejemanja
        CMPB    #%00000000
        BEQ     SPRP        ;ce je ugasnjeno, gre na SPRP, sicer nadaljuje

        LDAA    #$0F
        STAA    $0000       ;shrani napotek, da naslednjic poslje desno
                            polovico
        JMP     KIRQP

DESNOP  LDAA    $0001      ;nalozi trenutno stanje luck
        ASLA                ;premakne
        ASLA                ;za 4
        ASLA                ;bite
        ASLA                ;v levo
        ORAA    #%00001000  ;posilja desno polovico (bit 3 je 1)
        ANDA    #%11111011  ;ne da napotka, naj drugi racunalnik preklopi na
                            posiljanje (bit 2 je 0)
        STAA    DRA         ;v DRA poslje vsebino ACCA (upostteva se leva
                            polovica)
```

```
        LDAB    DRA         ;bere jo, da sprozi prekinitev in dobi stanje
                            stikal
        STAB    $0002       ;shrani stanje stikal
        ANDB    #%00000010 ;ignorira vse razen 2. stikala
        STAB    $0003       ;shrani stanje posiljanja/sprejemanja
        CMPB    #%00000000
        BEQ     SPRP        ;ce je ugasnjeno, gre na SPRP, sicer nadaljuje

        LDAA    #$F0
        STAA    $0000       ;shrani napotek, da naslednjic poslje levo
                            polovico
        JMP     KIRQP

SPRP    LDAA    #%00001111 ;!!!
        STAA    DRB         ;!!!
        JSR     CAKAJ       ;!!!
        LDAA    #$0F
        STAA    $0004       ;shrani napotek, da je treba preklopiti na
                            sprejemanje
        LDAA    #%00000000
        STAA    $0003       ;shrani, da je zdaj sprejemnik

KIRQP   RTS

********************
*IRQ za sprejemanje*
********************
IRQS    JSR     CAKAJ
        LDAA    DRA         ;nalozi poslano v ACCA in sprozi prekinitev
        TAB                 ;skopira poslano v ACCB
        ANDB    #%00001000 ;ignorira vse razen bita 3
        CMPB    #%00001000
        BEQ     DESNOS      ;ce je bit 3 1, gre na DESNO, sicer nadaljuje

        ANDA    #%11110000 ;v desno polovico da 0
        STAA    $0001       ;shrani levo polovico
        JMP     KIRQS

DESNOS  TAB                 ;skopira poslano v ACCB
        ANDB    #%00000100 ;ignorira vse razen bita 2
        CMPB    #%00000100
        BEQ     SPRS        ;ce je 1, gre na SPRS, sicer nadaljuje

        TAB                 ;skopira poslano v ACCB
        LSRB                ;premakne
        LSRB                ;za 4
        LSRB                ;bite
        LSRB                ;v desno
        LDAA    $0001       ;nalozi levo polovico v ACCA
        ABA                 ;kombinira oba dela v ACCA
        STAA    DRB         ;prizge lucke
        JMP     KIRQS

SPRS    LDAA    #%11110000 ;!!!
        STAA    DRB         ;!!!
        JSR     CAKAJ       ;!!!
        LDAA    #$F0
        STAA    $0004       ;shrani napotek, da je treba preklopiti na
                            posiljanje
        LDAA    #%00000010
        STAA    $0003       ;shrani, da je zdaj posiljatelj
```

```
KIRQS   RTS

*****
*IRQ*
*****
IRQ     LDAA    $0003      ;nalozi stanje posiljanja/sprejemanja
        CMPA    #%00000010
        BEQ     NAIRQP     ;ce posilja, gre na NAIRQP, sicer nadaljuje

        JSR     IRQS
        JMP     KIRQ

NAIRQP  JSR     IRQP

KIRQ    RTI

*****
*SWI*
*****
SWI     RTI

*****
*NMI*
*****
NMI     RTI

*******
*reset*
*******
HWRES   LDS     #STCKAD    ;zacetni naslov sklada
        JSR     INIP       ;inicializacija PIA za sprejemanje
        LDAA    DRA
        ANDA    #%00000010 ;ignorira vse razen 2. stikala
        STAA    $0003      ;shrani stanje posiljanja/sprejemanja
        CMPA    #%00000010
        BEQ     JEPOS      ;ce je prizgano, gre na JEPOS, sicer nadaljuje

        JSR     INIPS
        JMP     KHWRES

JEPOS   JSR     INIPP      ;inicializacija PIA za posiljanje
        JSR     INIPOS     ;zacne posiljanje

KHWRES  CLRA
        STAA    $0004      ;shrani napotek, da ni treba preklopiti na
                           posiljanje/sprejemanje
        CLI                ;omogoci prekinitve

****************
*glavni program*
****************
ZANKA   LDAA    $0004      ;nalozi napotek o preklapljanju
        CMPA    #$00
        BEQ     KZANKA     ;ce je $00, ponovi zanko, sicer nadaljuje

        JSR     CAKAJ
        CMPA    #$0F
        BEQ     NAS        ;ce je $0F, gre na NAS (preklop na sprejemanje),
                           sicer nadaljuje (preklop na oddajanje)
```

```
        SEI             ;onemogoci prekinitve
        JSR     INIPP   ;inicializacija PIA za posiljanje
        JSR     CAKAJ
        CLI             ;omogoci prekinitve
        JSR     INIPOS  ;zacne posiljanje
        CLRA
        STAA    $0004   ;shrani napotek, da ni treba preklopiti na
                        posiljanje/sprejemanje
        JMP     KZANKA

NAS     LDAA    #%00001100
        STAA    DRA     ;poslje napotek, naj drugi racunalnik preklopi
                        na posiljanje
        LDAB    DRA     ;bere ga, da sprozi prekinitev
        JSR     INIPS
        CLRA
        STAA    $0004   ;shrani napotek, da ni treba preklopiti na
                        posiljanje/sprejemanje

KZANKA  JMP     ZANKA

        END
```