Laboration 5

HENRIK BÄCK
850611-6253

Karlstads Universitet
2005-02-21

Handledare: Hans Hedbom
Nils Dåverhög

trafikljus.s

```
#include<iregdef.h>
#include<idtcpu.h>
#include<excepthdr.h>
#define IO 0xBFA00000
#define KNAPP2 0x00000001
#define KNAPP1 0x00000002
#define TIMER 0x0003

.data

     tick: .word 0
     huvud: .word 1              #talar om signal på huvudled
     kors: .word 0               #talar om signal på korsväg
     prev: .word 0               #0=huvudväg   1=korväg
     ljusbyt: .word 4            #0=grön huvud, 1=gul mot huvud, 2=gul mot kors,
     3=grön kors, 4=inget

.set noreorder
.text

.globl int_sub
.ent int_sub
int_sub:

     j int_routine               #hoppa till den riktiga interrupt rutinen
     nop

.end int_sub

.globl int_routine
.ent int_routine
int_routine:

     #spara undan på stacken
     subu sp,sp,56
     sw ra,48(sp)
     sw v0,44(sp)
     sw v1,40(sp)
     sw t0,36(sp)
     sw t1,32(sp)
     sw t2,28(sp)
     sw t3,24(sp)
     sw t4,20(sp)
     sw t5,16(sp)
     sw t6,12(sp)
     sw t7,8(sp)
     sw t8,4(sp)
     sw t9,0(sp)

     mfc0 k0,C0_CAUSE            #hämta cause register
     nop
     andi t0,k0,EXCMASK          #skilj ut våra bitar
     nop
     bne t0,zero,restore         #vi vill inte ta hand om annat än våra avbrott
     nop

     la t9,IO
     nop
```

```
     lb t0,0(t9)
     nop

     andi t1,t0,KNAPP1          #skilj ur slinga 1
     nop
     bne t1,KNAPP1,knp2         #hoppa ifall inte slinga 1 är aktiv
     nop
     la t9,huvud                #ladda in huvudvägens variabeladress
     addi t3,zero,1             #lägg in 1 i t3
     nop
     sw t3,0(t9)                #spara 1 i f.g variabel för att symbolisera grön
     b restore                  #klart
     nop


knp2:
     andi t1,t0,KNAPP2          #skilj ur slinga 2
     nop
     bne t1,KNAPP2,timer        #hoppa ifall inte slinga 2 är aktiv
     nop
     la t9,kors                 #ladda in huvudvägens variabeladress
     addi t3,zero,1             #lägg in 1 i t3
     nop
     sw t3,0(t9)                #spara 1 i f.g variabel för att symbolisera grön
     b restore                  #timer ska inte köras om den inte gjort avbrott
     nop

timer:
     la t3,tick
     nop
     lw t2,0(t3)                #läsa in tick
     nop
     addi t2,t2,1               #addera tick med ett
     nop
     sw t2,0(t3)                #spara tick
     la t4,huvud
     la t5,kors
     la t6,prev
     lw t4,0(t4)
     lw t5,0(t5)
     lw t6,0(t6)

     bne t4,0,hit               #om huvudvägen är grön hoppa
     nop
     bne t5,0,dit               #om korsvägen är grön hoppa
     nop

hit:
     bne t5,1,timer_ljus        #om korsvägen inte är grön hoppa
     nop

     fall båda skall vara gröna görs följande
     blt t2,10,timer_ljus       #hoppa om tick<10, vi får inte göra ngt
     nop

     la t8,prev
     nop
     lw t8,0(t8)
     nop

     bne t8,0,hgron             #om huvud=1&kors=1 och huvud var 1 senast byt sign.
```

```
     nop
     addi t6,zero,1             #sätt nuvarande sign. till föreg. sign.
    la t8,prev
     nop
     sw t6,0(t8)                #spara ner prev igen
     addi t4,zero,0             #sätt huvudväg röd
     la t8,huvud
     nop
     sw t4,0(t8)                #spara huvudvägsign.

     addi t2,zero,0             #nollställt tick
     nop
     sw t2,0(t3)                #spara tick

     #fixa ljussignal
     addi t8,zero,2             #gult mot kors
     nop
     la t7,ljusbyt
     nop
     sw t8,0(t7)

     b timer_ljus
     nop


dit:
     blt t2,10,timer_ljus       #hoppa om tick<10, vi får inte göra ngt
     nop

hgron:
     addi t6,zero,0             #sätt nuvarande sign. till föreg. sign.
    la t8,prev
     nop
     sw t6,0(t8)                #spara ner prev igen
     addi t5,zero,0             #sätt korsvag röd
     la t8,kors
     nop
     sw t5,0(t8)                #spara korsvägsign.

     addi t4,zero,1
     la t8,huvud
     nop
     sw t4,0(t8)                 #spara huvudsignal.

     addi t2,zero,0              #nollställt tick
     nop
     sw t2,0(t3)                 #spara tick

     #fixa ljussignal
     addi t8,zero,1              #gult mot huvudled
     nop
     la t7,ljusbyt
     nop
     sw t8,0(t7)

timer_ljus:
     #gör ett ljusbyte
     la t9, ljusbyt
     nop
     lw t9,0(t9)                 #läs in ljusstatus
     nop
```

```
     beq t9,4,restore            #gör inget om vi inte ska byta
     nop
     bne t9,3,next1              #om ej grön kors hoppa till next1
     nop

     #sätt ljussignaler grönt i korset, rött i huvudled
     addi t8,zero,129
     nop
     la t7,0xBF900000
     nop
     sb t8,0(t7)

     la t8,ljusbyt
     addi t7,zero,4              #sätt nästa byte till inget
     nop
     sw t7,0(t8)                 #lagra byte

     b restore
     nop

next1:
     bne t9,2,next2              #om ej gul mot kors
     nop

     #sätt ljusen gula
     addi t8,zero,70
     nop
     la t7,0xBF900000
     nop
     sb t8,0(t7)

     la t8,ljusbyt
     addi t7,zero,3              #sätt nästa byte till grön kors
     nop
     sw t7,0(t8)                 #lagra byte

     b restore
     nop


next2:
     bne t9,1,next3              #om ej gul mot hvud
     nop

     #sätt ljusen gula
     addi t8,zero, 194
     nop
     la t7,0xBF900000
     nop
     sb t8,0(t7)

     la t8,ljusbyt
     addi t7,zero,0              #sätt nästa byte till grön huvud
     nop
     sw t7,0(t8)                 #lagra byte

     b restore
     nop

next3:
     #sätt huvud grön
```

```
     addi t8,zero,36
     nop
     la t7,0xBF900000
     nop
     sb t8,0(t7)

     la t8,ljusbyt
     addi t7,zero,4              #sätt nästa byte till inget
     nop
     sw t7,0(t8)                 #lagra byte


restore:



     la t9,IO
     nop
     sb t0,0(t9)                           kvtittera avbrottet


     #lägg tillbaka från stacken
     lw ra,48(sp)
     lw v0,44(sp)
     lw v1,40(sp)
     lw t0,36(sp)
     lw t1,32(sp)
     lw t2,28(sp)
     lw t3,24(sp)
     lw t4,20(sp)
     lw t5,16(sp)
     lw t6,12(sp)
     lw t7,8(sp)
     lw t8,4(sp)
     lw t9,0(sp)
     addu sp,sp,56

     mfc0 k1,C0_EPC
     nop
     jr k1
     rfe


.end int_routine


.globl start
.ent start
start:

     #lägg in avbrottrutinen på 0x80000080
     la t0,int_sub               #ladda adress för hopprutinen
     la t1,0x80000080            #läladda adressen vi vill lägga detta på
     lw t2,0(t0)                 #läs in hoppinstr.
     lw t3,4(t0)                 #läs in nop i hopplucka
     sw t2,0(t1)                 #spara hoppinstr. på rätt adress
     sw t3,4(t1)                 #spara nop på rätt adress

     #uppdatera statusregistret för anv. def. avbrott.
     mfc0 t0,C0_SR
     nop
```

```
     ori t0,t0,1
     nop

     #tillåt EXT_INT3 som ett avbrott
     ori t0,t0,EXT_INT3
     nop

     #uppdatera statusregistret
     mtc0 t0,C0_SR

     #sätt huvud grön
     addi t8,zero,36
     nop
     la t7,0xBF900000
     nop
     sb t8,0(t7)

loop:

     #här kan man göra något annat om man vill medan vi väntar på att slingan skall
     #uppdateras.

     b loop
     nop

.end start
```