



LUND INSTITUTE OF TECHNOLOGY
Lund University

Informationsteknologi
Projektarbete
2000-05-22

IRRCFPC100A

(InfraRed Remote Controller For Personal Computer version 1.00 Alfa)

Fjärrkontrollsmottagare

Johan Ohlsson, e97jol - Daniel Fäldt, e97df

Handledare: Stefan Nyman, Informationsteknologi, LTH

Abstract

The purpose of this project was to get a basic knowledge in product developing using a processor. We chose to use a HC11 for our project. This because it is a single chip computer and contains all the memory and capacity we needed.

We decided to build a remote reciever for a PC. We planned to use a normal remote control from a Philips TV for this. It was going to control certain programs in the computer. For example starting Windows Explorer by a single click on a button.

We gave our invention the name IRRCFPC100A (InfraRed Remote Controler For Personal Computer version 1.00 Alfa).

Innehållsförteckning

1. Inledning	4
2. Komponenter.....	5
2.1 Processor – HC11	5
2.2 IR-dekoder – SAA3049A	5
2.3 Konverterare – Max 232 Maxim.....	6
2.4 IR-diod – IS1U60.....	6
3. Programvara	7
3.1 Simuleringsprogram – IAR Embedded Workbench 2.3	7
3.2 Kretsritningsprogram – Power Logic 3.0.....	7
3.3 Microsoft Visual Basic 5.0	7
3.4 Microsoft Hyper Terminal	7
4. Konstruktion.....	7
4.1 Tillvägagångsätt.....	7
4.2 Signalens väg genom konstruktionen	8
5. Mjuk vara	8
5.1 HC11-program	8
5.1.1 Assembler.....	8
5.1.2 C.....	9
5.2 Visual Basic-program	9
6. Resultat.....	11
7. Referenser	12
Bilaga A	13
Bilaga B.....	18
Bilaga C.....	20
Bilaga D	26

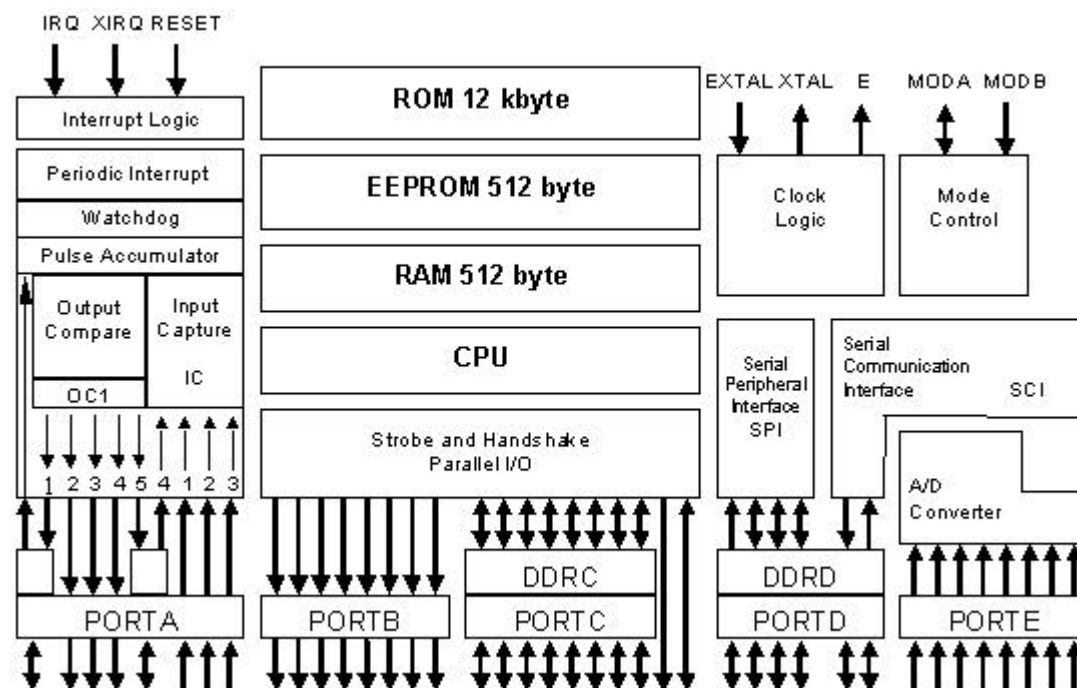
1. Inledning

Vi valde att gå kursen digitala projekt eftersom vi båda aldrig haft intresse/erfarenhet av praktiskt elektronik och ville ha lite grundläggande kunskaper. Vi ville därför bygga något som innehöll hårdvarutillverkning såväl som programmering. Efter övervägande och diskussion med handledare såvalde vi att bygga en fjärrkontrollsmottagare till datorn dvs en krets som mottager en IR-signal från en vanlig fjärrkontroll och sedan skickar en signal till ett datorprogram som i sin tur utför något på skärmen. Pga tidsbegränsning såvalde vi att inte utföra något större kommando i dator utan vikten lade vi vid att konstruera mottagarkretsen.

2. Komponenter

2.1 Processor – HC11

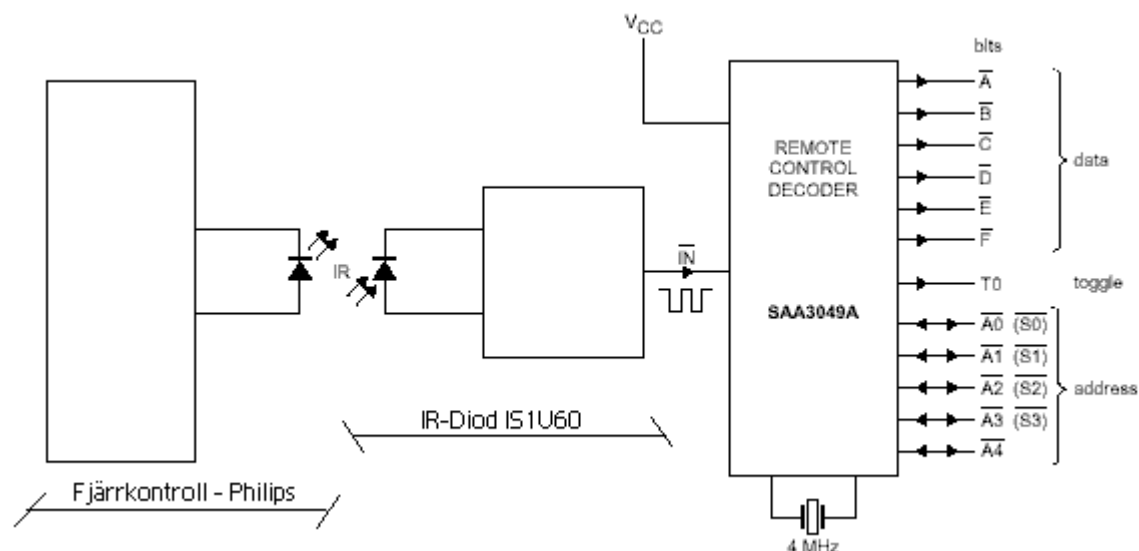
Den processor som användes i projektet var en HC11. Detta är en enchipsdator, dvs med det mesta (minne osv.) inbyggt. Ett krav för processorn var att den skulle klara av interrupts (avbrott), seriell och parallell signalbehandling samt flanktrigging av insignal. Detta var krav som HC11 uppfyllde.



Figur 2.1.1 Översikt bild HC11-funktion

2.2 IR-dekoder – SAA3049A

Denna komponent har till uppgift att detektera en signal från t.ex. en ir-diod och konvertera denna till en parallell signal som kan läsas av HC-11:an. Den har två olika modes. Skillnaden på dessa är att det detekterar olika typer av signaler. Fjärrkontrollen som används är av standarden RC5 dvs Mode = 0V.



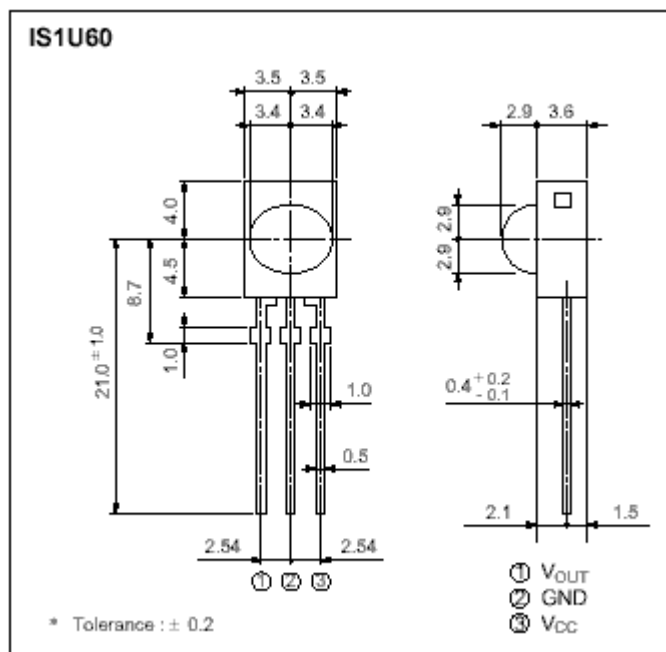
Figur 2.2.1 Ir-dekoder

2.3 Konverterare – Max232 Maxim

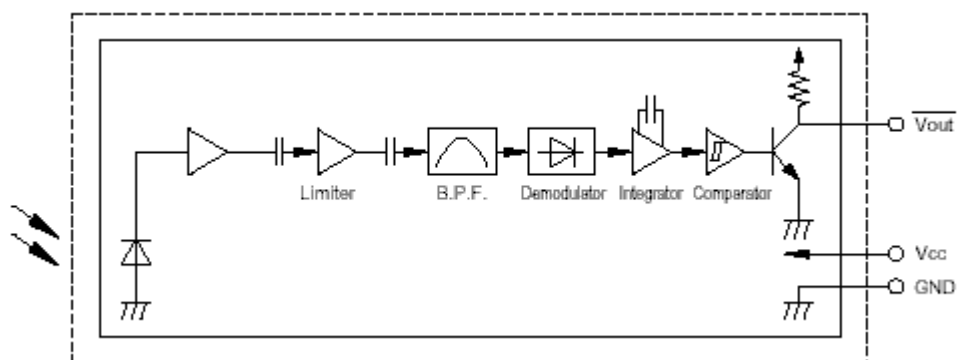
Denna komponents uppgift är att konvertera vanlig 0-5V:s signaler till signaler som kan läsas av en vanlig dator via kommunikationsporten (com-porten) dvs –10 till 10V.

2.4 IR-diod – IS1U60

Denna komponent (se figur 2.4.1) har till uppgift att detektera en ir-signal från en fjärrkontroll och sända denna vidare som 0-5V-signaler. Den har inbyggda filter och förstärkare för att ge en bra utsignal (se figur 2.4.2).



Figur 2.4.1 Ir-diod



Figur 2.4.2 Ir-diodens funktion

3. Programvara

3.1 Simuleringsprogram – IAR Embedded Workbench 2.3

Detta är ett program som används tillsammans med en emulator och simulerar då en riktig HC11.

3.2 Kretsritningsprogram – Power Logic 3.0

Detta är ett program för att på ett snyggt sätt rita upp kretsarna, dess placering och hur de är sammankopplade.

3.3 Microsoft Visual Basic 5.0

Detta är ett utvecklingsprogram för mjukvara. Används i en vanlig pc.

3.4 Microsoft HyperTerminal

Ett enkelt terminalprogram för att detektera signaler på com-portarna.

4. Konstruktion

4.1 Tillvägagångssätt

Vi började med att välja vilken hårdvara som skulle användas. Vi valde processorn HC11 (se 2.1), ir-dekoderna SAA3049A (se 2.2). Som ir-diod valde vi BPW50 vilken långt senare visade sig vara helt fel. De kringkomponenter som skulle behövas för att denna skulle fungera var en bra bit över kunskaps- och tidsnivån som fanns för denna kurs. Dock lade vi ner en ansevärd tid på att försöka förstärka och ta bort brus från signalen innan vi gav upp och började leta en annan komponent. Vi hittade då en annan krets från Sharp vid namn IS1U60 som hade detta inbyggt (se 2.4).

Denna komponent testades nu på en testplatta och fungerade. Dock när komponenten placerades på kortet så fick vi ett antal störningar. Dessa berodde på att vi valt att använda för smala sladdar till strömförsörjning och att inga avlastningskondensatorer använts. Fortfarande fungerade det inte (inga signaler på utgångarna på SAA:n). Efter ytterligare några jobbiga timmar kom vi på att det fanns något som hette ”pull-up-motstånd” vilka gör att signalen kan ses på utgången. Vi åtgärdade detta och fick fortfarande inga signaler på utgångarna på SAA:n. Vi hade tidigare tagit för givet att vår fjärrkontroll skulle fungera men började nu tvivla. Signalen från dioden (fjärrkontrollen) spelades in med ett digitalt oscilloskop. Det visade sig att signalen hade 32 bitar medan vår mottagare endast hade 6-11 bitars utsignal. Detta fick oss att testa med en annan fjärrkontroll och då en av samma märke som vår krets (en Philips). Detta visade sig fungera alldeles utmärkt!

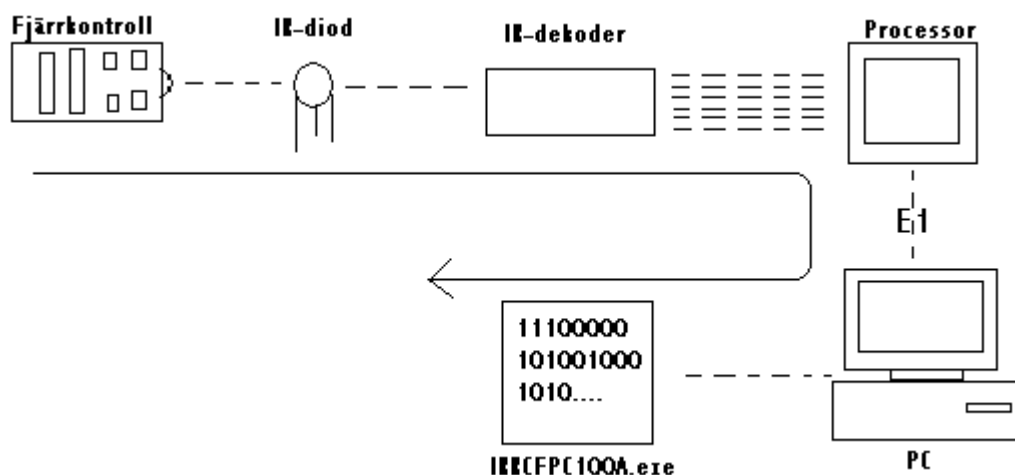
För att snygga till vår konstruktion kopplade vi in en lysdiod som blinkade när en knapp trycktes ner på fjärrkontrollen (ben 19 på SAA:n). Denna visade sig senare dra för mycket ström så vi kopplade bort den (den har ingen praktisk funktion för konstruktionen). Signalen verkade nu nå HC11:an och vi valde att börja programmera denna.

Ett C-program skapades i IEA Embedded Workbench (se 3.1), programmet återfinns i bilaga B nedan. Detta reagerar på avbrott från SAA:an och skickar då en signal till com-porten på datorn. Vi testkörde vårt nya fina program och såg att HC11:an reagerade på avbrotten.

Nästa problem uppkom när allt plötsligt en dag slutade att fungera. Det visade sig att problemet låg i att SAA:an hade gått sönder och låg och kortslöt övriga kretsar. Vi bytte SAA:n och allt fungerade som det skulle igen. I datorn startade vi ett terminalfönster (se 3.4). I detta kunde vi nu se våra signaler, och allt var okej. Ett Visual Basic-program (se 3.3) skapades nu i vilket ett program för att styra händelserna i datorn skapades.

4.2 Signalens väg genom konstruktionen

En vanlig fjärrkontroll av modell Philip MatchLine används för att skicka en 8 bitars lång seriell signal till vår detektor på kortet, en ir-diod (se 2.4). Ir-dioden skickar den kodade signalen vidare till en ir-dekoder (se 2.2 ovan), dvs en krets avkodar signalen. Ir-dekodern omvandlar dessutom insignalen till en 8 bitars parallell utsignal. Denna parallella utsignal skickas sedan vidare till processorn (se 2.1). Processorn i sin tur omvandlar signalen till två ASCII-tecken och skickar dem vidare seriellt via dess seriella utport. Signalen omvandlas sedan av en krets (se 2.3) till signaler som datorns (pc:ns) seriella kommunikationsport kan detektera och förstå. I datorn läses sedan ett Visual Basic-program av ASCII-tecknen och startar sedan ett program (som konfigurerats att användas till just den hexadecimala koden). Hela vägen kan ses i skissen i figur 4.2.1 nedan. För noggrannare kretsschema se bilaga D.



Figur 4.2.1 Signalens väg

5. Mjukvara

5.1 HC11-program

5.1.1 Assembler

Programkoden återfinns i bilaga A. Det är ett standardprogram som endast kopierades rakt av.

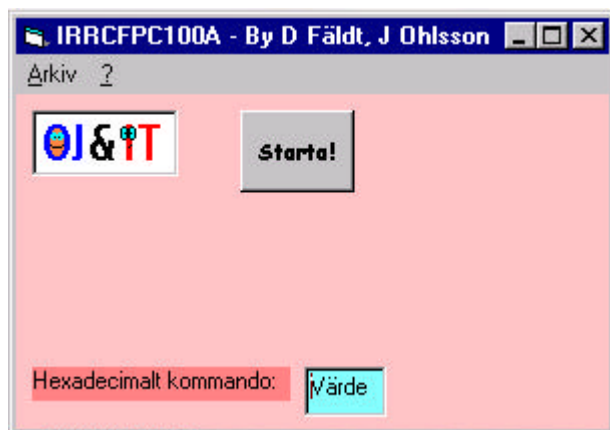
5.1.2 C

I C-programmet började vi med att ställa in vilka portar som skall vara in- resp utsignal, vilken överföringshastighet som skall användas mot datorn (pc:n) samt inställning såatt seriell utsignal används av HC11:an.

Vi löste problemet med att detektera och ta emot data från SAA:n (se 2.2) genom att använda avbrottsshantering (interrupts). Eftersom vi endast skall detektera omslag valde vi att låta avbrottet reagera påbåde positiv och negativ flank. För att underlätta kommunikationen med datorn (pc:n) såkodades den "8-bitiga" binära signalen om till tvåvanliga ascii-tecken. Dessa tvåtecken skickades sedan till datorn (pc:n) via seriella porten påHC11:an.

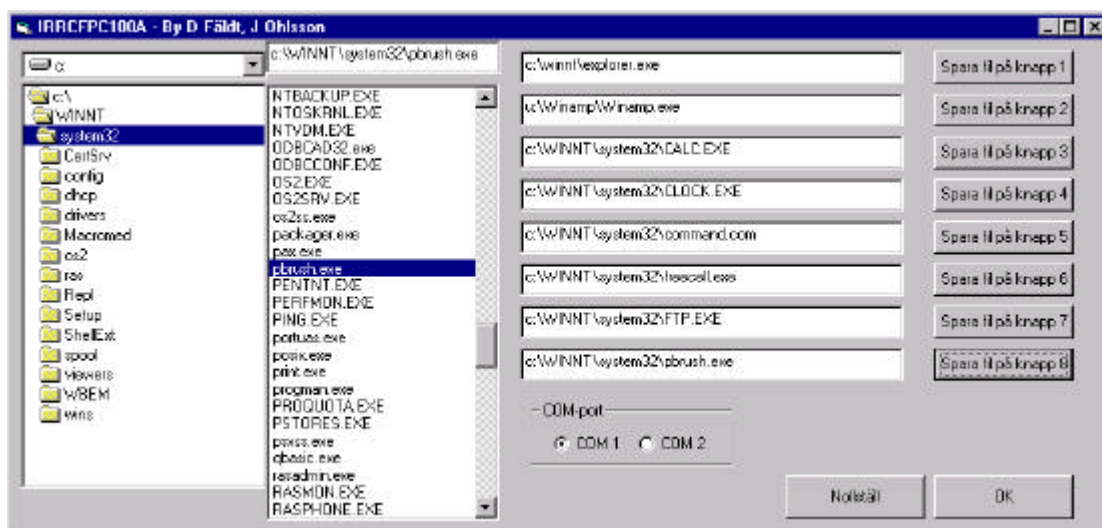
5.2 Visual Basic-program

I Visual Basic tänkte vi göra ett program som reagerar påinformation från den seriella kommunikationsporten (com-porten) och dåutföra vissa kommandon i datorn. Dessa kommandon bestod i att öppna ett antal program, ett för varje knapp. Programmet i startutförande ses i figur 5.2.1 nedan.



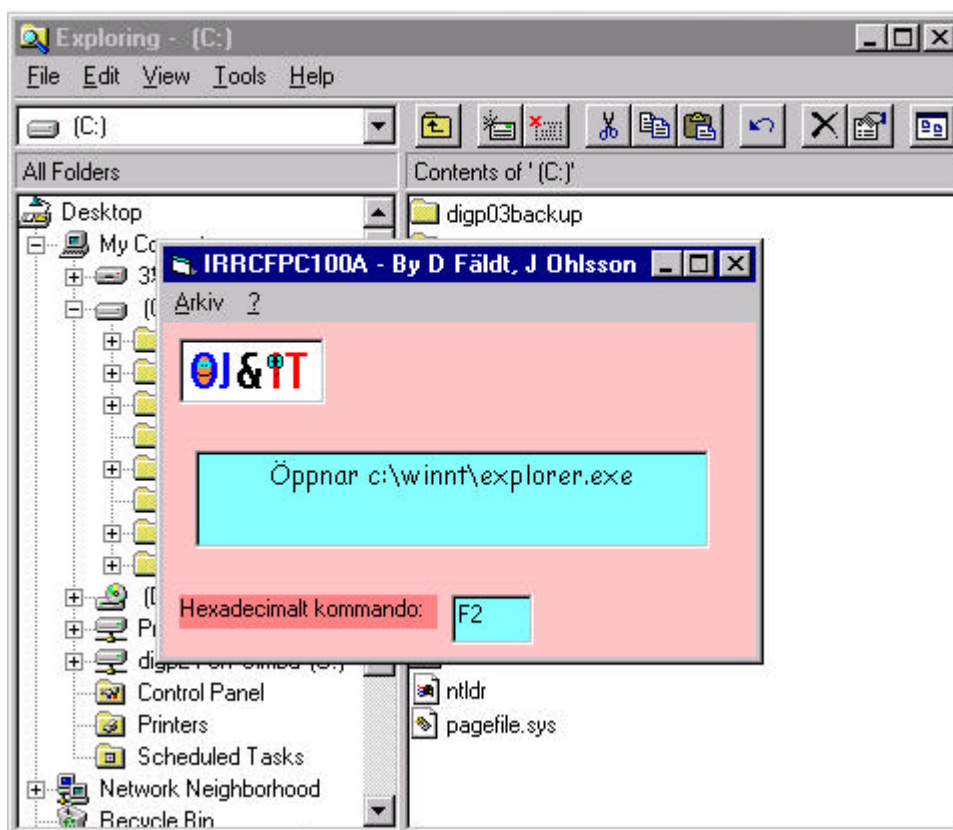
Figur 5.2.1 IRRCFPC100A i startläge

Vi valde nu att istället för att låsa vissa knappar till vissa program att ha ett konfigureringsfönster där man kan välja vilka program som skall aktiveras till varje knapp. Dessutom kan man där välja vilken seriell kommunikationsport (com-port) som skall användas, se figur 5.2.2 nedan.



Figur 5.2.2 IRRCFPC100A i konfigureringsläge.

Som standard ligger `c:\winnt\explorer.exe` på samtliga knappar och den seriella kommunikationsporten 1 (Com 1) som standardport. När ett program startas genom knapptryckning öppnas ett fönster där filens namn visas. Dessutom visas knappens hexadecimala kod i ett annat fönster. Se figur 5.2.3 nedan.



Figur 5.2.3 IRRCFPC100A i aktivt läge

6. Resultat

Vår konstruktion är nu färdig konstruerad och fungerar som den skall och som det var tänkt. Det tog visserligen lite längre tid än vad som var beräknat men vi har på vägen stött på många problem och på så vis införskaffat många nya kunskaper. Våra kunskaper innan denna kurs var i princip lika med noll och det är väl därför det gått lite långsamt. Skulle vi läsa denna kurs en gång till tror vi att vi hade klarat det mycket effektivare och snabbare. Detta pga att vi nu har betydligt större erfarenhet i området. Men trots allt så är vi mycket nöjda med vårt resultat – Vår första egna konstruktion.

7. Referenser

- [1] **Elfakatalogen**, *Elfa*, 1996-97 nr 45
- [2] **HC11 MC68HC711E9 Technical data**, *Motorola*, 1991
- [3] **The C programming language**, Kernichan - Ritchie, *Prentice Hall software series*, 1988 Second edition
- [4] **HC11 MC68HC711E9 Programming reference guide**, *Motorola*, 1991
- [5] **Lättpocket om Visual Basic 5**, Arvidsson - Ek, *Pagina*, 1997
- [6] **Visual Basic programmering**, Banks Sylvian, *Pagina*, 1993
- [7] **Visual Basic 4.0 for Windows**, Mansfield, *Ventana*, 1995
- [8] **Datablad till respektive komponent**, *Tillverkare/handledare*

Bilaga A

Assembler-program (Cstartup.s07)

```
*-----*
*
*          *
*   CSTARTUP.S07          *
*          *
* This module contains the 68HC11 startup routine and      *
* must usually be tailored to suit special hardware needs  *
*          *
* Note: The routine ?SEG_INIT_L07 is now included in CSTARTUP *
* The size of stack is set in the link-file (lnk6???.xcl)   *
* The segment INTVEC is declared COMMON                    *
*          *
* Version:  3.30 [IHAW 11/Feb/92]                          *
* Revised:  3.31 [IHAW 07/Jul/92] Bug in stack init        *
* Revised:  3.32 [IJAR 06/Mar/93] Startup code for K4 added *
* Revised:  4.11 [IMAI 29/Oct/96] Updated for ICC6811 4.11 *
* [SN 19/jan/98] Updated for Digitala projekt             *
*          *
*-----*
NAME  CSTARTUP
EXTERN  ?C_EXIT  ; Where to go when program is done

EXTERN  main    ; Where to begin execution

*-----*
* CSTACK - The C stack segment          *
*          *
* Please look in the link-file ??????.xcl how to increment *
* the stack without having to reassemble cstartup.s07 !    *
*-----*

RSEG  CSTACK
RMB  0  ; A bare minimum !!

*-----*
* Forward declarations of segments used in initialization *
*-----*

RSEG  UDATA0
RSEG  UDATA1
RSEG  IDATA0
RSEG  IDATA1
RSEG  ECSTR
RSEG  WCSTR
RSEG  CDATA0
RSEG  CDATA1
RSEG  CCSTR
RSEG  CONST
```

RSEG CSTR
RSEG TEMP

* RCODE - Where the program actually starts *

RSEG RCODE

init_C:

LDS #\$00FF ;#\$01FF for 68HC11E9
;#\$02FF for 68HC11E20

* If the 68HC11 OPTION register MUST be modified, here is the *
* place to do it in order to meet the 64-cycle requirement. *
* You can also do it in the beginning of main if You don't use *
* seg_init *

* If it is not a requirement that static/global data is set *
* to zero or to some explicit value at startup, the next line *
* of code can be deleted. *

BSR seg_init

* If hardware must be initiated from assembly or if interrupts *
* should be on when reaching main, this is the place to insert *
* such code. *

JSR main main()

* Now when we are ready with our C program (usually 6811 *
* programs are continuous) we must perform a system-dependent *
* action. In this simple case we jump to ?C_EXIT. *

* DO NOT CHANGE NEXT LINE OF CSTARTUP IF YOU WANT TO RUN
YOUR *

* SOFTWARE WITH THE AID OF THE C-SPY HLL DEBUGGER. *

JMP ?C_EXIT

* Copy initialized PROMmed code to shadow RAM and clear *
* uninitialized variables. *

```
*-----*
seg_init:

*-----*
* Zero out UDATA segments *
*-----*

    LDD #.SFE.(UDATA0)
    SUBD #.SFB.(UDATA0)
    BEQ SKIP00
    XGDY
    LDX #.SFB.(UDATA0)
    BSR zero_mem
SKIP00

    LDD #.SFE.(UDATA1)
    SUBD #.SFB.(UDATA1)
    BEQ SKIP01
    XGDY
    LDX #.SFB.(UDATA1)
    BSR zero_mem
SKIP01

*-----*
* Copy CDATA segments into IDATA segments *
*-----*

    LDD #.SFE.(CDATA0)
    SUBD #.SFB.(CDATA0)
    BEQ SKIP02
    LDX #.SFB.(CDATA0)
    LDY #.SFB.(IDATA0)
    BSR copy_mem
SKIP02

    LDD #.SFE.(CDATA1)
    SUBD #.SFB.(CDATA1)
    BEQ SKIP03
    LDX #.SFB.(CDATA1)
    LDY #.SFB.(IDATA1)
    BSR copy_mem
SKIP03

*-----*
* Copy CCSTR into ECSTR *
*-----*

    LDD #.SFE.(CCSTR)
    SUBD #.SFB.(CCSTR)
    BEQ SKIP04
```

```
LDX #.SFB.(CCSTR)
LDY #.SFB.(ECSTR)
BSR copy_mem
SKIP04
RTS          ; End of initialization

*-----*
* Clear memory                               *
*-----*
zero_mem:
CLR 0,X
INX
DEY
BNE zero_mem
RTS

*-----*
* Copy memory                               *
* Copy (X) --> (Y), counter in D           *
*-----*
copy_mem:
PSHA
LDAA 0,X
STAA 0,Y
PULA
INX
INY
SUBD #1
BNE copy_mem
RTS

*-----*
* Interrupt vectors must be inserted by the user. Here we *
* only used RESET.                                     *
*-----*

COMMON INTVEC
        ; Assuming start address = FFD6 for 68HC11
RMB 40

FDB init_C

ENDMOD init_C ; 'init_C' is program entry address

*-----*
* Function/module: exit(int code)             *
*-----*
* When C-SPY is used this code will automatically be replaced *
```

```
* by a 'debug' version of exit().          *
*-----*
MODULE  exit

PUBLIC  exit
PUBLIC  ?C_EXIT

RSEG   RCODE

?C_EXIT:
exit:
*-----*
* The next line could be replaced by user defined code.  *
*-----*
end_loop:
  BRA  end_loop

END
```

Bilaga B

C-program (Ir_remote.c)

```
/**
****
// Infrared Remote Controler for personal computer version 1.0 alfa
/**
****
//          Gjord av Daniel Fäldt, Johan Ohlsson
/**
****

// Konstantdeklarationer:
#include "io6811.h" // Inkludera portdeklarationer
#include "intr6811.h" // Inkludera interrupthanteringsfunktioner
#include "Vectors.h" // Inkludera interruptdeklarationer

void initIR(void) // Startintitering
{
    DDRC = 0x00; // Sätter PC0-PC7 som insignal (0000 0000)
    DDRD = 0xFE; // Alla signaler till utsignaler Port D utom Receiver (1111
1110)
    BAUD = 0x30; // BAUD = 9600 (0011 0000)
    SCCR1 = 0x0; // Start bit, 8 bit, 1 stoppbit
    SCCR2 = 0x08; // Transmitter enable (0000 1000)
    TCTL2 = 0x03; // Ställer in såatt PORTA-IC3 ger interrupt vid flakerna (0000
0011)
    TMSK1 = 0x01; // Slår påinterrupt påIC3 (0000 0001)
    enable_interrupt(); // Starta interrupt
}

interrupt void IC3_interrupt(void) // Interrupthantering av IC3
{
    char low, high;
    TFLG1 = 0x01; // Kvittera interrupt påIC3 (0000 0001)
    low = PORTC & 0xF; // Nedre delen av PORTC (00001111)
    high = PORTC & 0xF0; // Övre delen av PORTC (11110000)
    high = high / 0x10; // Flyttar 11110000 -> 00001111 (00010000)

    // Konvertering till ascii-tecken //
    if (low <= 0x9) // Om talen 0-9
    {
        low = low + 0x30;
    }
    else
    {
        low = low + 0x37;
    }
    if (high <= 0x9)
    {
```

```
    high = high + 0x30;
}
else
{
    high = high + 0x37;
}
// Konvertering slut //

while ((SCSR & 0x80)==0){} // Kvittra sändning
SCDR = low;           // Sänd nedre delen
while ((SCSR & 0x80)==0){} // Kvittra sändning
SCDR = high;          // Sänd övre delen
}

main(void)           // Huvudprogram
{
    initIR();        // Initiera!
    while (0 == 0)   // Gör inget, vänta påinterrupt påIC3
    {
    }
}
```

Bilaga C

Visual Basic-program

Form1

```
Public fil1 As String
Public fil2 As String
Public fil3 As String
Public fil4 As String
Public fil5 As String
Public fil6 As String
Public fil7 As String
Public fil8 As String
Public quit As Boolean
Private Sub Avsluta_Click()
    quit = True                'Stäng program
    Funktion_Click
End Sub
Private Sub copyright_Click()
Form3.Visible = True        'Öppna Copyright-fönster
End Sub
Private Sub Funktion_Click()
Funktion.Visible = False    'Dölj Startknapp
While quit = False
    MSComm1.PortOpen = True    'Öppna com-porten
    Do
        DoEvents                'Lå andra händelser hända samtidigt
    Loop Until MSComm1.InBufferCount >= 2 'Vänta tills data finns
    Data = MSComm1.Input        'Läs data
    MSComm1.PortOpen = False    'Stäng com-porten
    Text.Text = Data
    If Data = "13" Then          'Om knapp 1...
        X = Shell(fil1, vbNormalNoFocus) 'Öppna fil1, ej i fokus
        InfoBox.Visible = True
        InfoBox.Text = "Öppnar " & fil1    'Visa vilken fil i InfoBox
    ElseIf Data = "33" Then      'Om knapp 2...osv
        X = Shell(fil2, vbNormalNoFocus)
        InfoBox.Visible = True
        InfoBox.Text = "Öppnar " & fil2
    ElseIf Data = "23" Then
        X = Shell(fil3, vbNormalNoFocus)
        InfoBox.Visible = True
        InfoBox.Text = "Öppnar " & fil3
    ElseIf Data = "03" Then
        X = Shell(fil4, vbNormalNoFocus)
        InfoBox.Visible = True
        InfoBox.Text = "Öppnar " & fil4
    ElseIf Data = "E1" Then
```

```
X = Shell(fil5, vbNormalNoFocus)
InfoBox.Visible = True
InfoBox.Text = "Öppnar " & fil5
ElseIf Data = "F1" Then

X = Shell(fil6, vbNormalNoFocus)
InfoBox.Visible = True
InfoBox.Text = "Öppnar " & fil6
ElseIf Data = "E2" Then

X = Shell(fil7, vbNormalNoFocus)
InfoBox.Visible = True
InfoBox.Text = "Öppnar " & fil7
ElseIf Data = "F2" Then

X = Shell(fil8, vbNormalNoFocus)
InfoBox.Visible = True
InfoBox.Text = "Öppnar " & fil8
End If

Wend
Unload Form1                'Stäng program
End Sub
Private Sub Form_Load()
    Dim Data As String
    fil1 = "c:\winnt\explorer.exe"    'Initera filerna
    fil2 = "c:\winnt\explorer.exe"
    fil3 = "c:\winnt\explorer.exe"
    fil4 = "c:\winnt\explorer.exe"
    fil5 = "c:\winnt\explorer.exe"
    fil6 = "c:\winnt\explorer.exe"
    fil7 = "c:\winnt\explorer.exe"
    fil8 = "c:\winnt\explorer.exe"
    Form2.Visible = False
    InfoBox.Visible = False
    quit = False
    MSComm1.CommPort = 1            'Initiera med Com 1
    MSComm1.Settings = "9600,N,8,1"    'Baud 9600, 8 bitar...
    MSComm1.InputLen = 2            'Sätt indatalängd=2
End Sub
Private Sub Konfigurera_Click(Index As Integer)
    Form2.Visible = True            'Öppna Konfigureringsfönster
End Sub
```

Form2

```
Public old1 As String
Public old2 As String
Public old3 As String
Public old4 As String
Public old5 As String
```

```
Public old6 As String
Public old7 As String
Public old8 As String
Private Sub Cancel_Click()
    'Återställ
    Form1.fil1 = old1
    Form1.fil2 = old2
    Form1.fil3 = old3
    Form1.fil4 = old4
    Form1.fil5 = old5
    Form1.fil6 = old6
    Form1.fil7 = old7
    Form1.fil8 = old8
    Form_Load
    Unload Form2
End Sub
Private Sub com1_Click()
Dim oppen As Boolean
    oppen = False

    If Form1.MSComm1.PortOpen Then      'Om port öppen...
        oppen = True                    'Kom ihåg att den var öppen
        Form1.MSComm1.PortOpen = False  '...stäng
    End If
    Form1.MSComm1.CommPort = 1          'Sätt com-port 1
    Form1.MSComm1.Settings = "9600,N,8,1"
    Form1.MSComm1.InputLen = 2
    com1.Value = True
    com2.Value = False
    If oppen Then                        'Återställ port öppen/stängd
        Form1.MSComm1.PortOpen = True
    End If
End Sub
Private Sub com2_Click()
Dim oppen As Boolean
    oppen = False

    If Form1.MSComm1.PortOpen Then      'Om port öppen...
        oppen = True                    'Kom ihåg att den var öppen
        Form1.MSComm1.PortOpen = False  '...stäng
    End If
    Form1.MSComm1.CommPort = 2          'Sätt com-port 2
    Form1.MSComm1.Settings = "9600,N,8,1"
    Form1.MSComm1.InputLen = 2
    com1.Value = False
    com2.Value = True
    If oppen Then                        'Återställ port öppen/stängd
        Form1.MSComm1.PortOpen = True
    End If
```

End Sub

Private Sub Dir1_Change()

File1.Path = Dir1.Path
Text1.Text = Dir1.Path

End Sub

Private Sub Drive1_Change()

Dir1.Path = Drive1.Drive
Text1.Text = Dir1.Path

End Sub

Private Sub File1_Click()

Text1.Text = Dir1.Path & "\ " & File1.filename 'Skriv ut hela filnamnet

End Sub

Private Sub Form_Load()

Text2.Text = Form1.fil1
Text3.Text = Form1.fil2
Text4.Text = Form1.fil3
Text5.Text = Form1.fil4
Text6.Text = Form1.fil5
Text7.Text = Form1.fil6
Text8.Text = Form1.fil7
Text9.Text = Form1.fil8
old1 = Form1.fil1 'Spara undan filerna
old2 = Form1.fil2
old3 = Form1.fil3
old4 = Form1.fil4
old5 = Form1.fil5
old6 = Form1.fil6
old7 = Form1.fil7
old8 = Form1.fil8
com1.Value = (Form1.MSComm1.CommPort = 1)
com2.Value = (Form1.MSComm1.CommPort = 2)
File1.Pattern = "*.com;*.exe;*.bat" 'Visa endast exekverbara filer
Drive1.Drive = "c:\ "
Dir1.Path = "c:\winnt"
Text1.Text = Dir1.Path & "\ " & File1.filename

End Sub

Private Sub Ok_Click()

Unload Form2 'Stäng konfigurera

End Sub

Private Sub Knapp1_Click()

If File1.filename = "" Then
Text2.Text = "Markera en fil!"
Else
Form1.fil1 = Dir1.Path & "\ " & File1.filename
Text2.Text = Form1.fil1
End If

End Sub

Private Sub Knapp2_Click()

```
    If File1.filename = "" Then
Text3.Text = "Markera en fil!" ' Om ingen fil markerad, info
    Else
Form1.fil2 = Dir1.Path & "\ " & File1.filename
Text3.Text = Form1.fil2
    End If
End Sub
```

Private Sub Knapp3_Click()

```
    If File1.filename = "" Then
Text4.Text = "Markera en fil!"
    Else
Form1.fil3 = Dir1.Path & "\ " & File1.filename
Text4.Text = Form1.fil3
    End If
End Sub
```

Private Sub Knapp4_Click()

```
    If File1.filename = "" Then
Text5.Text = "Markera en fil!"
    Else
Form1.fil4 = Dir1.Path & "\ " & File1.filename
Text5.Text = Form1.fil4
    End If
End Sub
```

Private Sub Knapp5_Click()

```
    If File1.filename = "" Then
Text6.Text = "Markera en fil!"
    Else
Form1.fil5 = Dir1.Path & "\ " & File1.filename
Text6.Text = Form1.fil5
    End If
End Sub
```

Private Sub Knapp6_Click()

```
    If File1.filename = "" Then
Text7.Text = "Markera en fil!"
    Else
Form1.fil6 = Dir1.Path & "\ " & File1.filename
Text7.Text = Form1.fil6
    End If
End Sub
```

Private Sub Knapp7_Click()

```
    If File1.filename = "" Then
Text8.Text = "Markera en fil!"
    Else
```



```
Form1.fil7 = Dir1.Path & "\ " & File1.filename  
Text8.Text = Form1.fil7  
End If  
End Sub
```

```
Private Sub Knapp8_Click()  
    If File1.filename = "" Then  
Text9.Text = "Markera en fil!"  
    Else  
Form1.fil8 = Dir1.Path & "\ " & File1.filename  
Text9.Text = Form1.fil8  
    End If  
End Sub
```

Form3

```
Private Sub Form_Load()  
  
    'Skriv copyright-text  
Text.Caption = "IRRCFPC100A - InfraRed Remote Controller For Personal  
Computer version 1.00 Alfa" & Chr(13) & "Copyright © Daniel Fäldt, Johan Ohlsson  
-2000"  
  
End Sub  
Private Sub Ok_Click()  
    Unload Form3      'Stäng  
  
End Sub
```

Bilaga D

– Här skall ett kretsschema gjort i Power Logic 3.0 finnas –
SAKNAS I DENNA NÄTVERSION!