TSEA29 Konstruktion med mikrodatorer, Fö6

Anders Nilsson 2021-09-28
Institutionen för systemteknik

LINKÖPINGS UNIVERSITET

1

3

Agenda : Design av inbyggda system

- Hårdvarunära design – Erfarenhet/Utmaningar värda att tänka på
- Avbrottsrutiner och huvudloopar – hantering av gemensamma data
- Verkligt parallella processer – hantering av gemensamma data
- Kopplingsschema – hur och varför
- 10 sätt att lyckas med ett projekt
- Designspecar – bra / dåligt / hur / varför / när
- Handledning

LINKÖPINGS UNIVERSITET

3

Hårdvarunära design

LINKÖPINGS UNIVERSITET

4

Hårdvarunära design

Inbyggt system = ”man ser inget”

LINKÖPINGS UNIVERSITET

5

6

Hårdvarunära design

- Systemet byggs samtidigt som man programmerar det

Osäkerhet : Ligger problemen i hårdvaran eller mjukvaran?

LINKÖPINGS UNIVERSITET

6

7

Hårdvarunära design

- Man behöver ta *SMÅ* utvecklingssteg, justera kompasskursen ofta.

Varför?

- Helheten är svår att överblicka
- Små delar kan testas lättare
- Förutsättningar kanske förändras

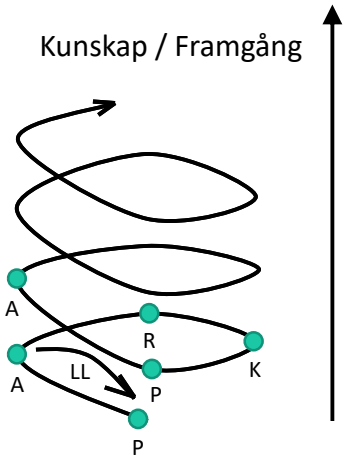
Jämför med återkopplingstiden i reglerloopen

LINKÖPINGS UNIVERSITET

7

8

Hårdvarunära design : P.A.R.K.



- **Planera**
  Samla information, strukturera, upprätta dokumentation
- **Agera**
  Implementera, konstruera
- **Reflektera**
  Varför fungerar det inte?
  Varför fungerar det?
- **Korrigera**
  Rätta till, uppdatera, optimera

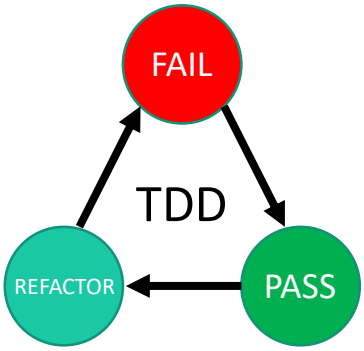
[David Kolb]

LINKÖPINGS UNIVERSITET

8

9

Hårdvarunära design
TDD : Test Driven Development



- FAIL : Skriv ingen kod förrän det tagits fram ett testfall som ännu inte uppfylls
- PASS : Skriv kod som uppfyller testfallet/en, men inte mer än det
- REFACTOR : Hyfsa till, strukturera, optimera koden och kontrollera att den fortfarande uppfyller testfallet/en

Överdrivet? Kanske…
Bra: Leder till iterativ testning, ofta, och undviker big-bang-testning

[Exempel : I2C]

LINKÖPINGS UNIVERSITET

9

10

Hårdvarunära design

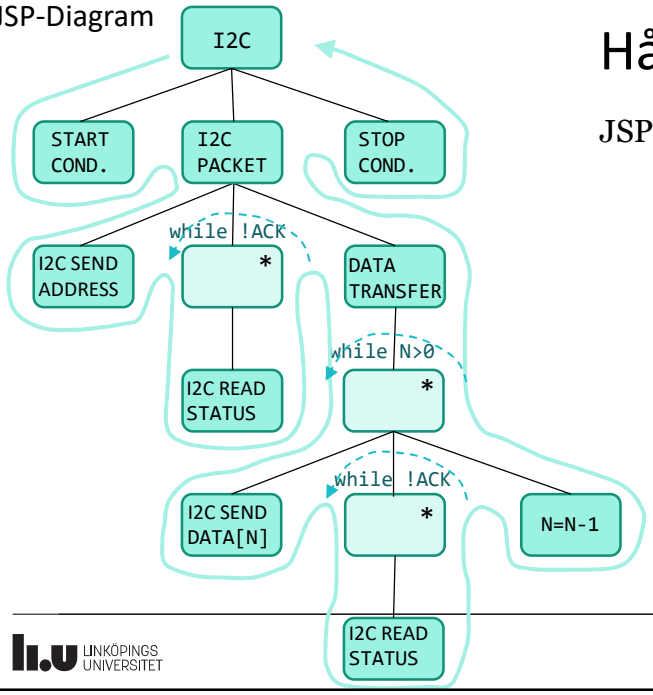
I2C-kommunikation

- ☑ M : Startvillkor
- ☑ M : Adressera slav/w
- ☑ S : Acknowledge
- ☑ M : Skriv data till S
- N
- ☑ S : Acknowledge
- ☑ M : Stoppvillkor

LINKÖPINGS UNIVERSITET

11

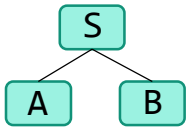
JSP-Diagram



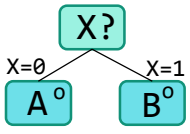
Hårdvarunära design

JSP : Jackson Structured Programming

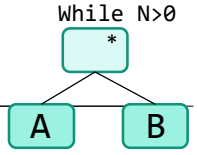
Sekvens:
Först A sen B



Selektion:
A eller B
beroende på X

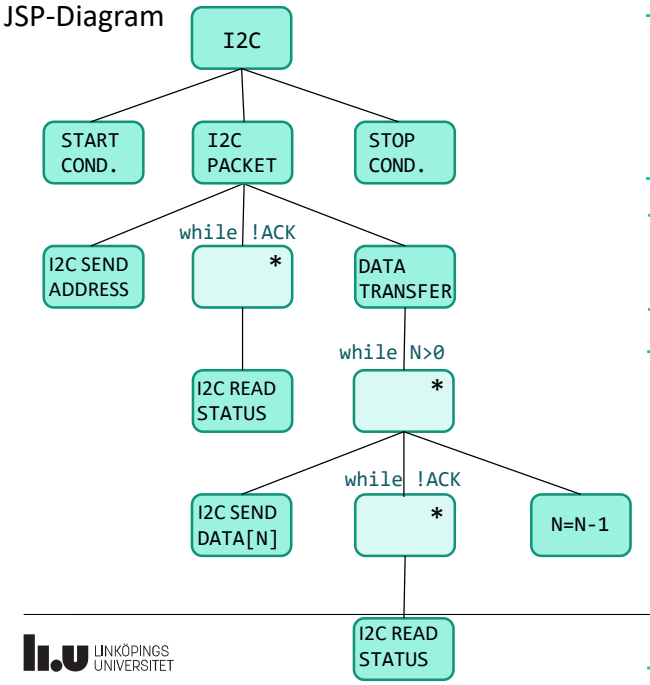


Iteration:
Först A sen B
Så länge N>0



LINKÖPINGS UNIVERSITET

JSP-Diagram



Programkod

```
I2C() {
  I2C_Start_Cond();
  I2C_Packet();
  I2C_Stop_Cond();
}

I2C_Packet() {
  I2C_Send(ADDRESS);
  Wait_For_ACK();
  Data_Transfer();
  }

Data_Transfer() {
  while (N>0) {
    I2C_Send(DATA[N]);
    Wait_For_ACK();
    N--;
  }
}

Wait_For_ACK() {
  while (Get_I2C_Status() != ACK));
}
```

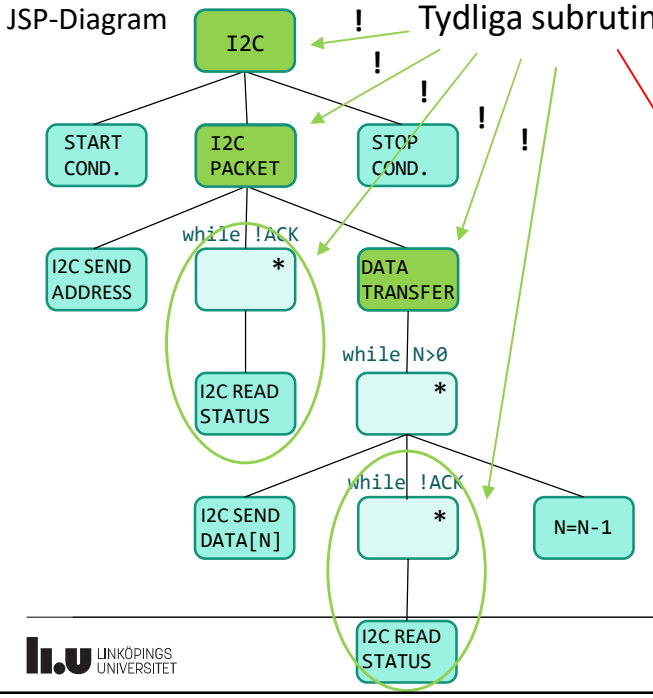
LINKÖPINGS UNIVERSITET

12

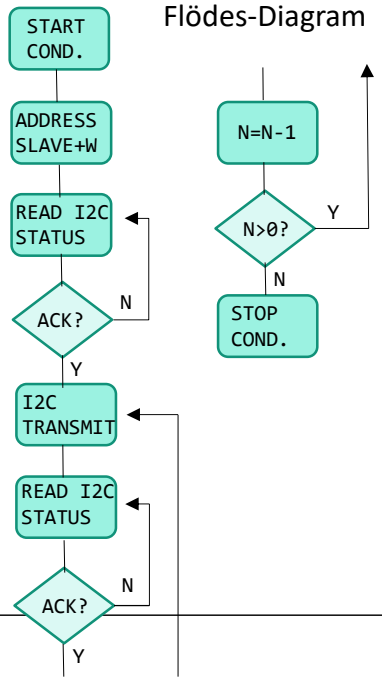
12

JSP-Diagram

Tydliga subrutiner



Flödes-Diagram



LINKÖPINGS UNIVERSITET

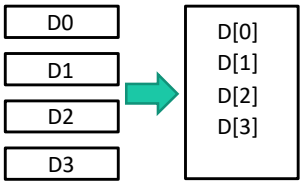
13

13

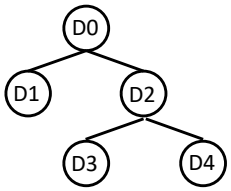
Datastrukturer

14

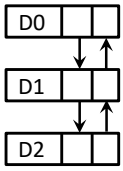
Buffer



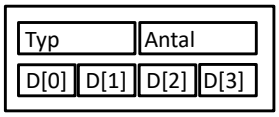
Träd



Länkad lista



Paketering med huvud



Rådata / SI-enheter / Flyttal / Heltal

35	Rådata
11cm	SI-enheter
3.14	Flyttal eller Heltal

LINKÖPINGS UNIVERSITET

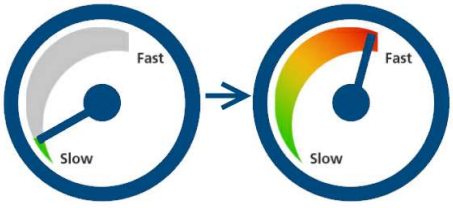
14

15

Hårdvarunära design

- Börja med låga hastigheter – öka vid behov

- Robustare
- Energisnålare



- Störningskänsligt
- Energikrävande

Det kan fungera bra att köra med full fart, **i början**, när det digitala bruset är lågt.
Men med mer och mer funktionalitet blir systemet störningskänsligt.
Plötsligt kan t ex ytterligare en sensor orsaka att I2C-kommunikationen slutar fungera, trots att det inte hör ihop. Problemet är egentligen relaterat till hastigheten.

LINKÖPINGS UNIVERSITET

15

16

Hårdvarunära design

- Spana framåt – undvik hinder
- Är vi på väg åt rätt håll?
- Leder det här till målet?
- Är det här en framkomlig väg?
- Finns det en lättare lösning?

Tänk efter före!

17

Hårdvarunära design

- Se även bakåt – ingen del av systemet är färdigt förrän hela produkten är klar
- Man kan få riva upp och göra om
- Då är man glad om man tagit små steg

18

Hårdvarunära design

- Var paranoid!!

Överallt!

Koll på allt!

Hela tiden!

LINKÖPINGS UNIVERSITET

18

Avbrottsrutiner och huvudloopar

Ett sätt att uppnå realtidsfunktionalitet och virtuell parallellism

LINKÖPINGS UNIVERSITET

19

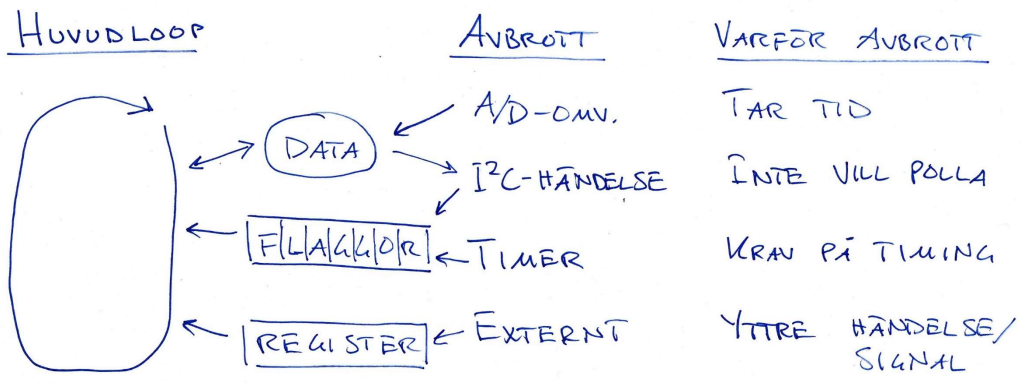
20

Avbrottsrutiner och huvudloopar

- Vilka avbrott ska vi hantera?
- Vad ska göras i avbrottsrutiner?
- Vad ska göras i kod som inte är avbrottsrutiner?
- Vad är lagom mycket för en avbrottsrutin?
- Hur klarar man krav på timing?
- Vilka data delas mellan avbrottsrutiner och övrig kod?
- Hur delas data? Ena läser och andra skriver, båda läser, båda skriver, etc.

21

Avbrottsrutiner och huvudloopar



Avbrottsrutiner och huvudloopar

Avbrott i AVR, förutsättningar:

- Nya avbrott blockeras när avbrottsrutin körs (dvs pågående avbrott avbryts inte)
- Nya avbrott som inkommer när avbrottsrutin körs latchas/memoreras (en flagga per avbrottsvektor)
- Vissa variabler kan läsas och skrivas atomärt (dvs accessen av variabeln avbryts inte), andra är icke atomära

Avbrottsrutiner och huvudloopar

Avbrott i AVR, atomär läsning av en variabel vid upprepade tillfällen

Gemensam variabel mellan ISR (Interrupt Service Routine) och huvudloop

```
int Distance;
```

Variabeln sätt av ISR

Avbrottsrutiner och huvudloopar

Avbrott i AVR, atomär läsning av en variabel vid upprepade tillfällen

Kod i huvudloop

```
if (Distance < 10)
  decrease robot speed
else if (Distance > 20)
  increase robot speed
else
  do not change robot speed
```

Vad händer om ISR förändrar Distance mellan avläsningarna?

Avbrottsrutiner och huvudloopar

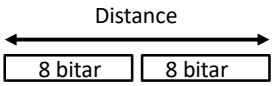
Avbrott i AVR, atomär läsning av en variabel vid ett tillfälle

```
local_distance = Distance;
if (local_distance < 10)
  decrease robot speed
else if (local_distance > 20)
  increase robot speed
else
  do not change robot speed
```

Avbrottsrutiner och huvudloopar

Avbrott i AVR, icke-atomär läsning av en variabel

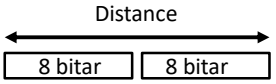
```
DISABLE_INTERRUPTS;
local_distance = Distance;
ENABLE_INTERRUPTS;
if (local_distance < 10)
  decrease robot speed
else if (local_distance > 20)
  increase robot speed
else
  do not change robot speed
```



Avbrottsrutiner och huvudloopar

Avbrott i AVR, läsning av *flyktig* en variabel

```
uint_16t Distance = 0;
...
DISABLE_INTERRUPTS;
local_distance = Distance;
ENABLE_INTERRUPTS;
if (local_distance < 10)
  decrease robot speed
else if (local_distance > 20)
  increase robot speed
else
  do not change robot speed
```




```
ISR {
  ...
  Distance = ...;
  ...
}
```

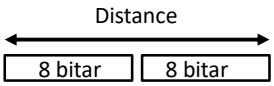
Kompilatorn kanske optimerar bort Distance.

Avbrottsrutiner och huvudloopar

28

Avbrott i AVR, läsning av *flyktig* en variabel : volatile

```
volatile uint_16t Distance = 0;
...
DISABLE_INTERRUPTS;
local_distance = Distance;
ENABLE_INTERRUPTS;
if (local_distance < 10)
  decrease robot speed
else if (local_distance > 20)
  increase robot speed
else
  do not change robot speed
```




```
ISR {
  ...
  Distance = ...;
  ...
}
```

Kompilatorn förstår att Distance är flyktig och kan ändras, t ex av ISR.

LINKÖPINGS UNIVERSITET

28

Avbrottsrutiner och huvudloopar

29

Avbrott, andra typer av scenarier – gemensamma data

- Buffertar – avbrottsrutin skriver/läser, huvudloop skriver/läser
- Gemensamma datablock – huvudloop skriver/läser hela block med avbrott stängda (för att säkerställa konsistens av data inom ett block)
- Dubbelbuffring – huvudloop arbetar med kopia, avbrottsrutin arbetar med en annan - avbrottsrutin gör ”swap” genom att flytta pekare – huvudloop stänger av avbrott vid läsning av pekare om dessa inte kan läsas atomärt

LINKÖPINGS UNIVERSITET

29

Avbrottsrutiner och huvudloopar

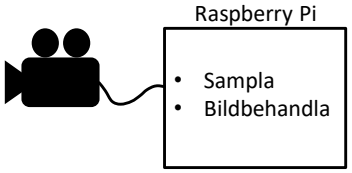
Avbrott - embedded, referensmaterial

- AVR : Tutorial, GCC m m – https://www.mikrocontroller.net/articles/AVR
  Tysk site, men funkar bra med Google translate
- Artikel på vanheden – http://vanheden.isy.liu.se
- En artikelserie från embedded.com – http://bit.ly/SVHrIh

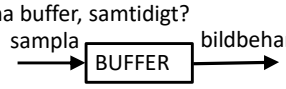
Parallella processer

Verkligt parallella processer i flera kärnor

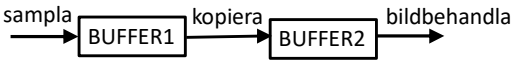
Parallella processer



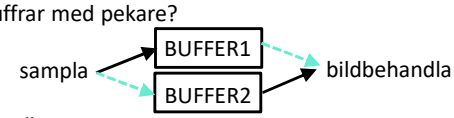
Sampla först -> buffer -> bildbehandla sen?
Hinner man bildbehandla mellan bilderna? 32
Knappast!
Skippa varannan bild?
Dålig framrate!
Kan man sampla och bildbehandla samtidigt?
Ja, med parallella trådar/processer!
På samma buffer, samtidigt?



Vad händer om sampling skriver samtidigt som bildbehandling läser?
Risk för korrupt/inkonsistent data!
Hjälper det att stänga av avbrott?
Nej! Trådarna/processerna är inga avbrott
Kopiera mellan dubbla buffrar?



Växla buffrar med pekare?

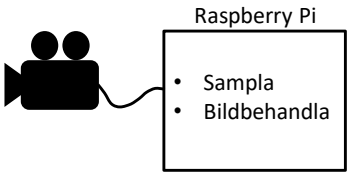


Ja! MEN NÄR???

LINKÖPINGS UNIVERSITET

32

Parallella processer



Trådarna/processerna måste synkroniseras! 33

Trådar : -Har sitt ursprung från samma program
-Kan synkroniseras med semaforer
-Kan kommunicera via gemensamt/delat minne

Process: -Är ett program, skrivet i t ex Python, Java, C
-Kan synkroniseras med meddelanden
-Kan kommunicera via filer, pipes, sockets

I ett operativsystem förekommer många processer, som alla har en viss prioritet.

Vilken prioritet behöver sampling och bildbehandling ha?
I förhållande till varandra?
Sampling högre prio än bildbehandling
I förhållande till andra processer i operativsystemet?
Beror på situationen, dvs finns det fler viktiga tidskritiska saker som operativsystemet måste hantera

LINKÖPINGS UNIVERSITET

33

Parallella processer


```
//thread1 : sampling
while(1) {
  for (i=0; i<count; i++)
    buf[bp][i] = sample();
  bp = 1 - bp;
  sem_post(&Semaphore);
}
```

```
//thread2 : img_processing
while(1) {
  sem_wait(&Semaphore);
  img_process(buf[1-bp]);
}
```

Trådar inom samma program, kan kommunicera via gemensamt minne.

Synkroniseras med semafor.

```
//process1 : sampling
while(1) {
  for (i=0; i<count; i++)
    fputc(sample(), FileP[bp]);
  M.bp = bp;
  bp = 1 - bp;
  message_send(&M, PORT);
}
```

```
//process2 : img_processing
while(1) {
  message_recieve(&M, PORT)
  img_process(M.bp);
}

int img_process(int bp) {
  data = fgetc(FileP[bp]);
  ...
```

Processer i olika program, kan kommunicera via externa filer, lämpligen virtuella i RAM.

Synkroniseras med meddelandehantering.

LINKÖPINGS UNIVERSITET

34

Kopplingsschema

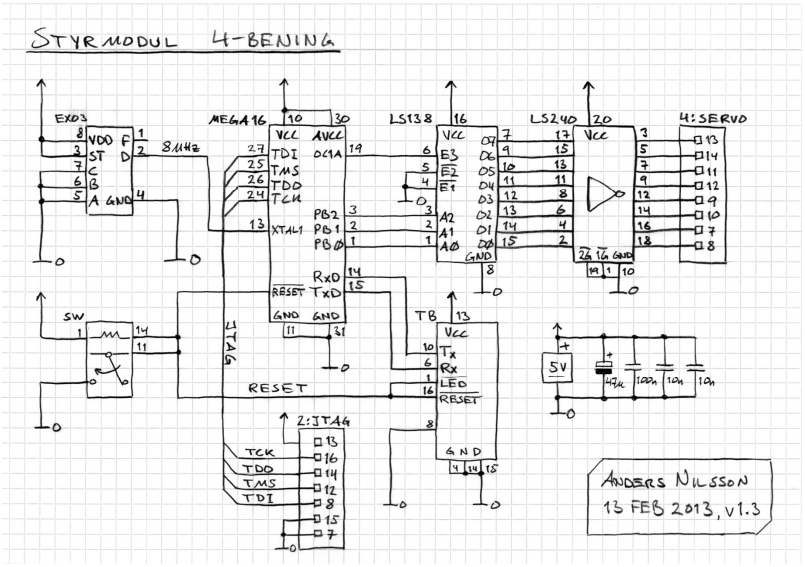
LINKÖPINGS UNIVERSITET

35

Kopplingsschema

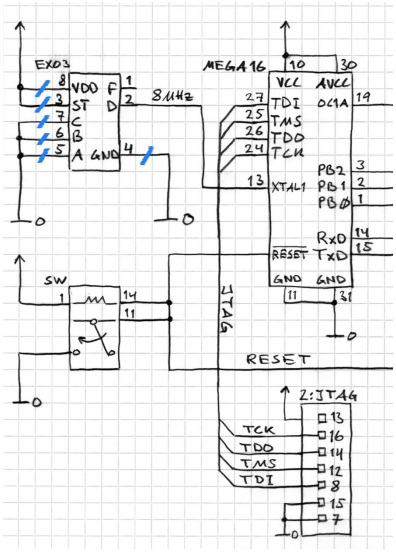
- Hur?
- Varför?

- Renrita!!
- Använd rutat papper!!



Kopplingsschema

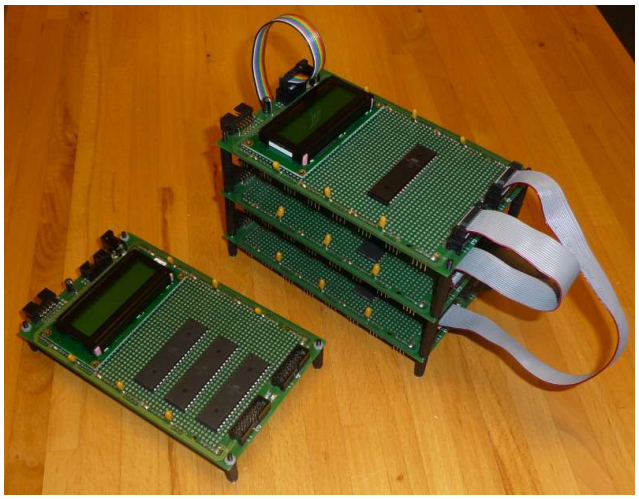
Från schema till bygge
- Vira in en liten del åt gången
- Bocka av virade trådar i schemat



Kopplingsschema

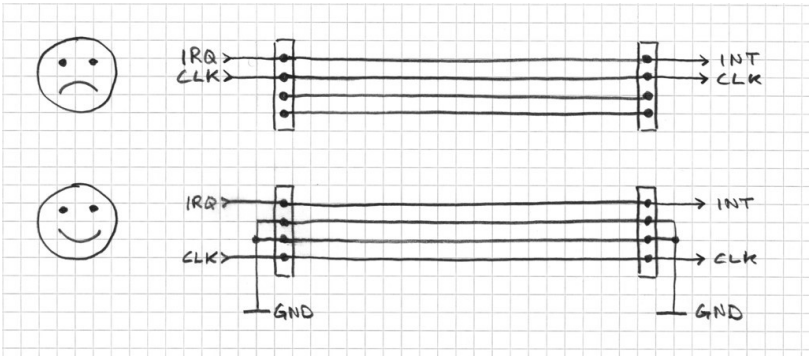
Från schema till bygge
- Ett eller flera virkort

...



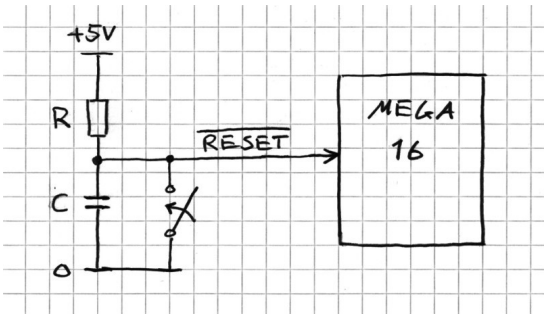
Kopplingsschema

Från schema till bygge
- Parallella ledare ...



Kopplingsschema

Från schema till bygge
- Anslut reset!



Utan reset : svår felsökning!

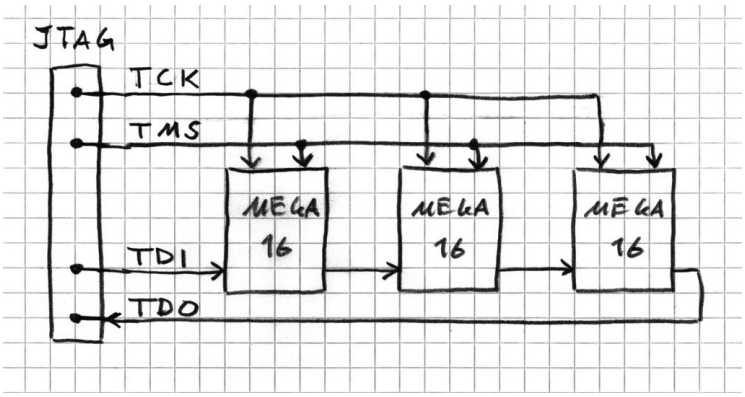
Utan reset : felaktig programmering!

Kopplingsschema

Från schema till bygge
- JTAG används för programmering och debugging

JTAG-kedja …

… eller inte



Schemaritningsprogram

- EAGLE – https://www.autodesk.com/products/eagle/
  - Kommersiell, men gratis för mindre storlekar (80 cm^2) av konstruktioner
- KiCad – www.kicad-pcb.org
  - Open Source via GNU GPL v3

10 sätt att lyckas med ett projekt

Baserat på råd från tidigare års studenter

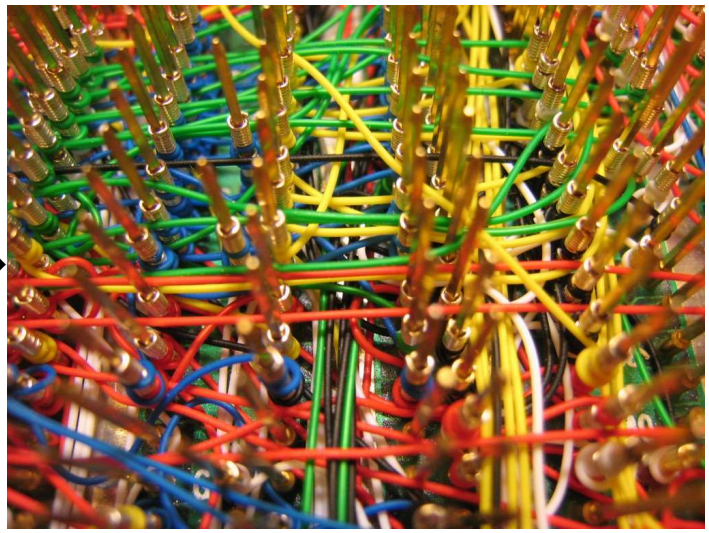
10 sätt att lyckas med ett projekt

1 : Börja i tid

10 sätt att lyckas med ett projekt

2 : Gör ordentliga virningar

www.bigmessofwires.com →



10 sätt att lyckas med ett projekt

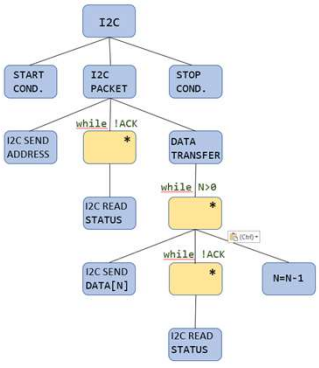
3 : Skriv programkoden själv


```
a= DateTime.Now.Millisecond;
ber = new Random(seed);
i= number.Next(0,9);
(maskedSet[i] ==0)
askedSet[i]=1;
etCount++;
// Mask each set
 seed = DateTime.Now.Millisecond;
number = new Random(seed);
int maskPos = number.Next(minPos,maxPos);
int j=0;
do
{
 seed   = DateTime.Now.Millisecond;
 number = new Random(seed);
 int newPos = number.Next(1,9);
 int x = _setRowPosition[i]+po
 int y=  _setColPosition[i]+
 if(_problemSet[x,y]==0)
 {
 _problemSet[x,y] = num
 j++;
```


10 sätt att lyckas med ett projekt

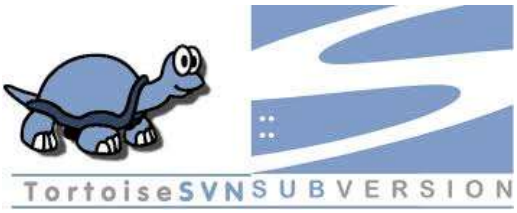
4 : Strukturera programkoden

```
int banan22(int banan13, int banan37)
{
    if (banan37 > banan13) then
        banan13 = banan37*3;
    else
        banan13 = banan37-banan13;
    return banan13;
}
```



10 sätt att lyckas med ett projekt

5 : Använd versionshantering



https://gitlab.liu.se
https://docs.isy.liu.se/bin/view/VanHeden/GiT

Undvik helst GitHub

Håll även kopplingsshemat uppdaterat!!!

10 sätt att lyckas med ett projekt

6 : Läs databladen

1. Översiktligt
2. Detaljerat
3. Verifiering

10 sätt att lyckas med ett projekt

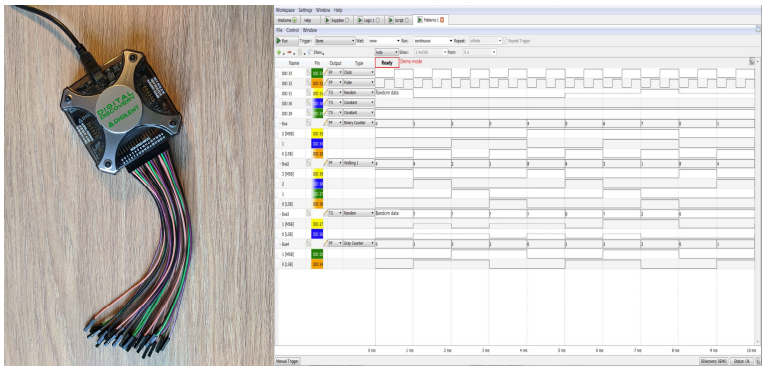
7 : Verifiera komponentens funktion ur databladet

REKLAM

VERKLIGHET

10 sätt att lyckas med ett projekt

8 : MÄT!



Logikanalysatorlab
Flera tillfällen, gå på ett av dom
Anmälan i Lisam

”Zu messen ist zu wissen”

10 sätt att lyckas med ett projekt

9 : Samarbeta

Kommunicera!

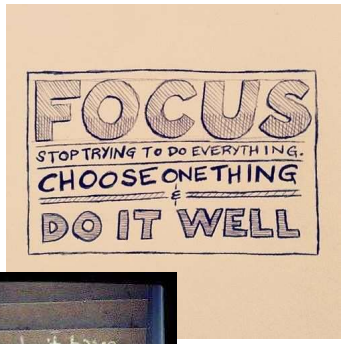
Färdig med veckans arbete i förväg? Tar man helg då?

Sjuk? Berättar men det först när man är tillbaka igen?

Alla har ett gemensamt ansvar!

10 sätt att lyckas med ett projekt

10 : Konstruera och lös problemen steg för steg



Designspecar

LINKÖPINGS UNIVERSITET

55

Designspecar

- Checklista för designspec =>
- Exempel på designspecar =>
  + Fokus på hårdvaran
  + Pseudokod som bilaga
  + Upprepa inte datablad

Inlämningstid

- Senast 7/10 kl 16:00 ska en första version av designspecen vara inlämnad till handledaren
- Senast 14/10 ska designspecen kunna vara godkänd av handledaren

LINKÖPINGS UNIVERSITET

56

Designspecar

57

Designspecifikation : Förslag på arbetsordning

1 : Kopplingsschema
- Så detaljerat som möjligt
- Ta bort ovidkommande information
- Vilken / vilka bussar är lämpliga / krävs?
- Namnge signaler och bussar
- Planera in anslutning för JTAG samt reset
- Behövs nivåkonvertering? 5V <-> 3V

2 : Datastrukturer
- Hur ska data lagras?
- Vad ska skickas?
- På vilken form? Rådata eller SI-enheter?
- Behövs nåt protokoll, paketering av data?

3 : Programflöde / struktur
- Vad ska vara avbrott?
- Vad passar i huvudloopen

4 : Beskrivande information
- Textuell
- Bilder / figurer

Handledning

58

- Två timmar handledning per vecka
- Använd tilldelad handledare i första hand
  Vid enstaka tillfällen då din handledare inte är tillgänglig kan andra handledare vara behjälpliga med mindre insatser, t ex lämna ut en komponent eller svara på en akut fråga.
- Handledarens huvudsakliga uppgift är att förmedla arbetssätt, t ex felsökningsmetodik, snarare än att utföra felsökning
- Handledningsform:
  -Uppsöka vid behov?
  -Boka möten?
  -E-post? Teams?
  Diskutera upplägget med er handledare

Anders Nilsson

www.liu.se

LINKÖPINGS UNIVERSITET

59