

TSEA29 Konstruktion med mikrodatorer, Fö5

Anders Nilsson 2022-09-15

Institutionen för systemteknik

Agenda TSEA29

- Enchipsdatorn AVR
- Kommunikation
- Sensorer
- Reglerteknik
- Avbrott
- Raspberry Pi
- Lab1



www.atmel.com



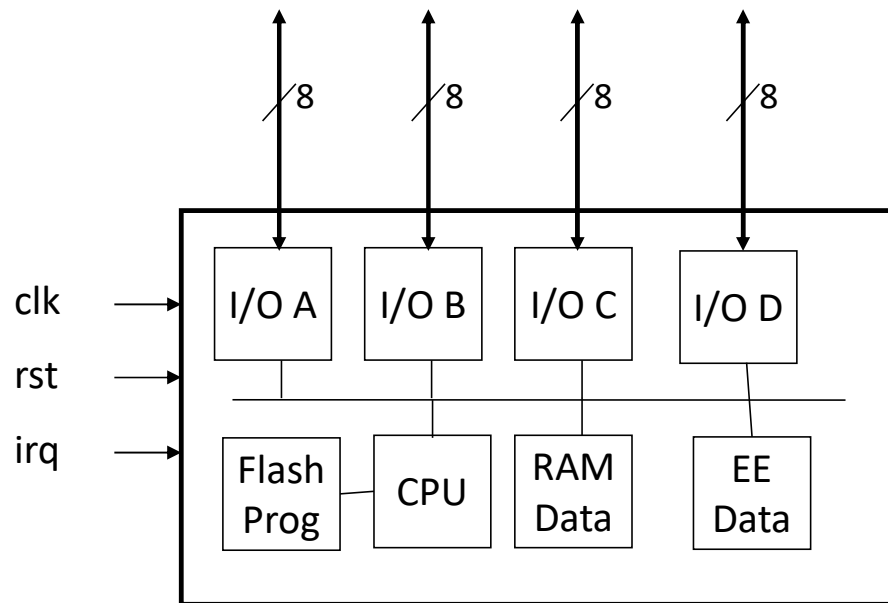
www.avrfreaks.net

Enchipsdatorn AVR

AVR Enchipsdator = allt på ett chip

CPU, RAM, ROM,

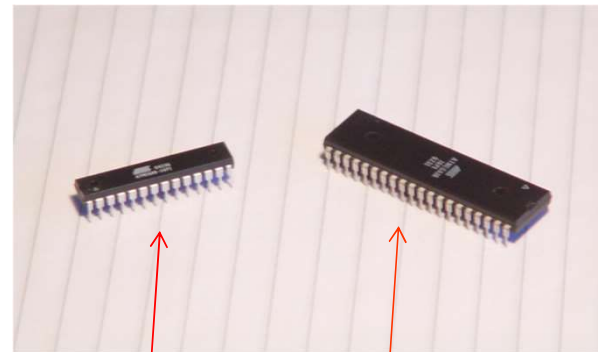
I/O (parallell, seriell, A/D, timer/counter, PWM, I2C, SPI, ...)



AVR-familjen. 8-bits RISC-dator + I/O
AVR = **A**lf **V**egard **R**isc

ATmega1284 <-> ATmega16 <-> ATmega168

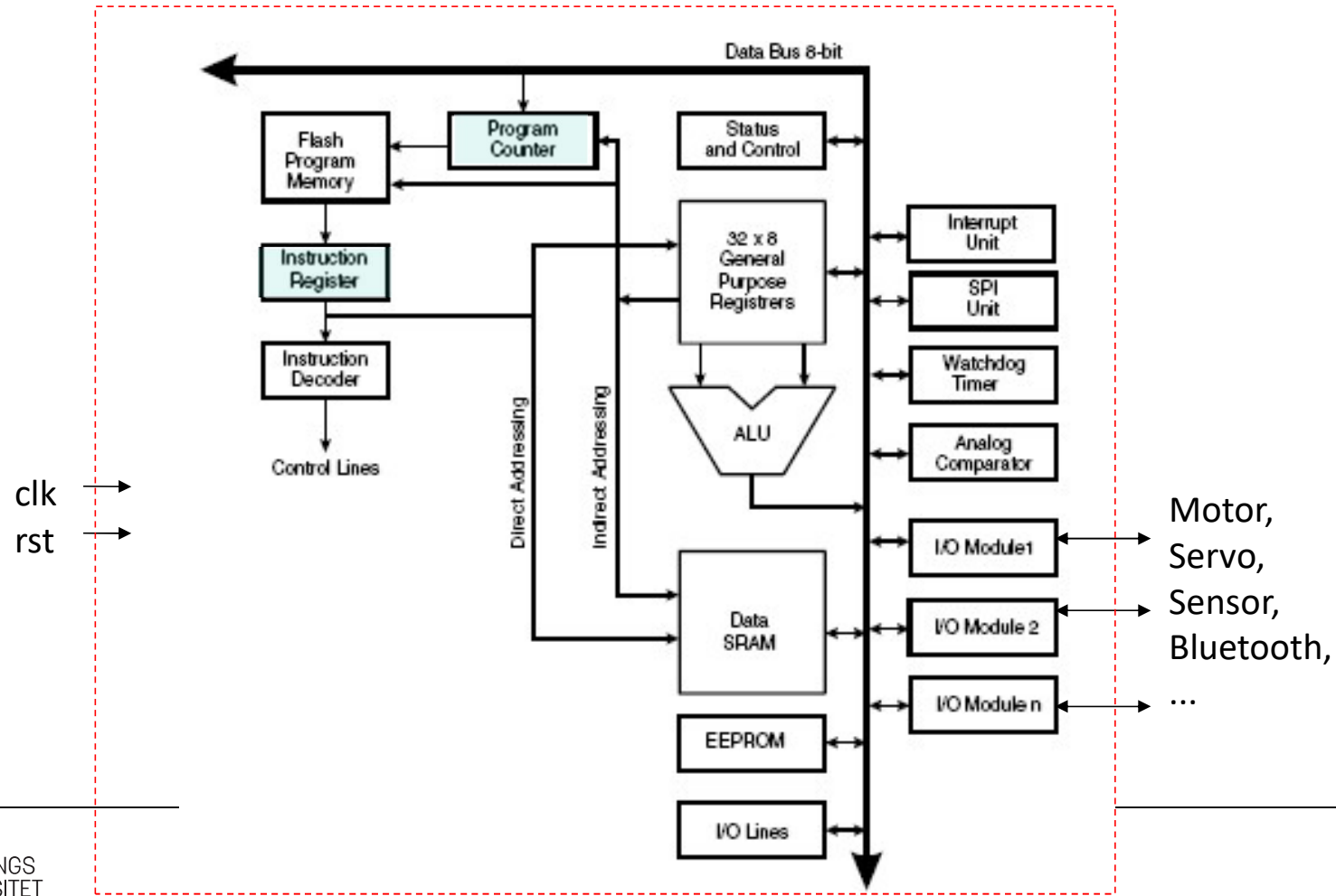
Egenskap	1284	16	168
Instruktioner	131 kraftfulla!		
Register	32 generella 8-bits		
Flash (program, konstanter)	128 kB	16 kB	16 kB
EEPROM (konstanter)	4 kB	512 B	512 B
SRAM (variabler)	16 kB	1 kB	1 kB
I/O (pinnar)	32 (40)	32 (40)	23 (28)
Avbrott (antal)	36	20	26
I ² C	1	1	1
SPI	3	1	1
UART	2	1	1
8-bit timers	2	2	2
16-bit timers	2	1	1
PWM (kanaler)	6	4	6
A/D (kanaler)	8	8	6
klocka	0-20 MHz	0-16 MHz	0-20 MHz



168

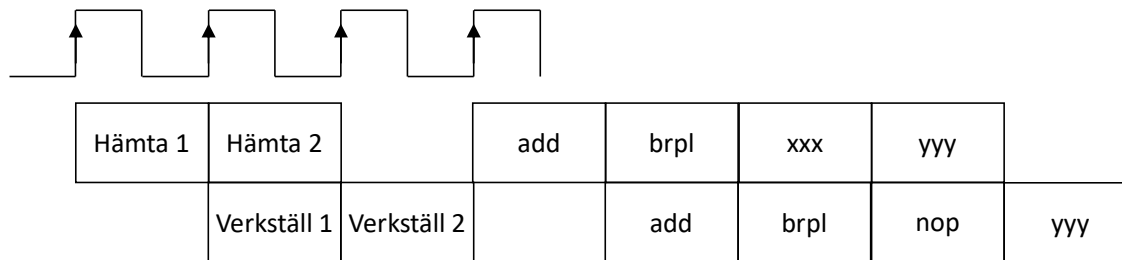
1284, 16

Principschema : 2-stegs pipeline



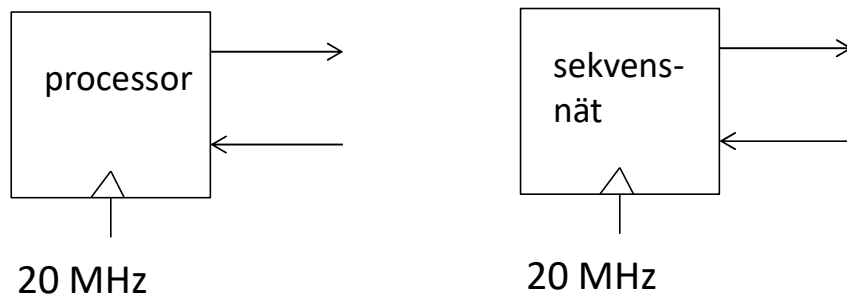
Hur fort går det?

Med 20 MHz klocka gör AVR 20 MIPS (miljoner instruktioner per sekund)



20000 instruktioner per millisekund!

OBS, en FPGA på samma frekvens är klart snabbare!



```

...
add    ...
brpl   skip1
xxx
skip1:
yyy
...

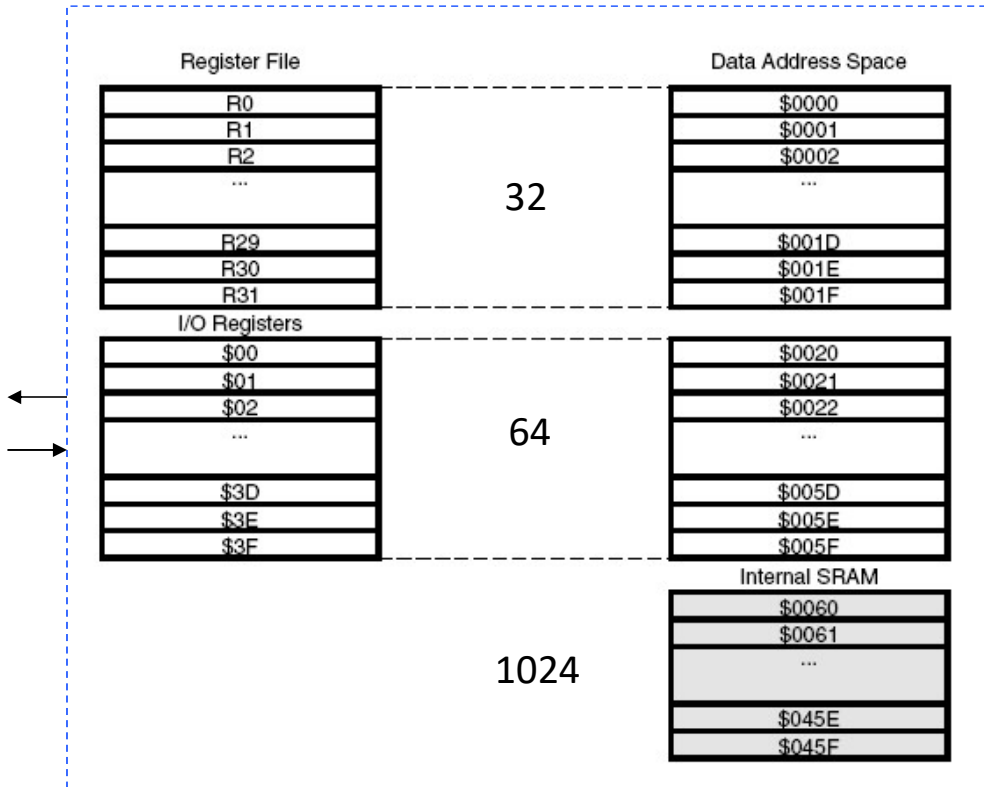
```

Vid brpl-hopp till skip1 fås en extra nop, annars inte.

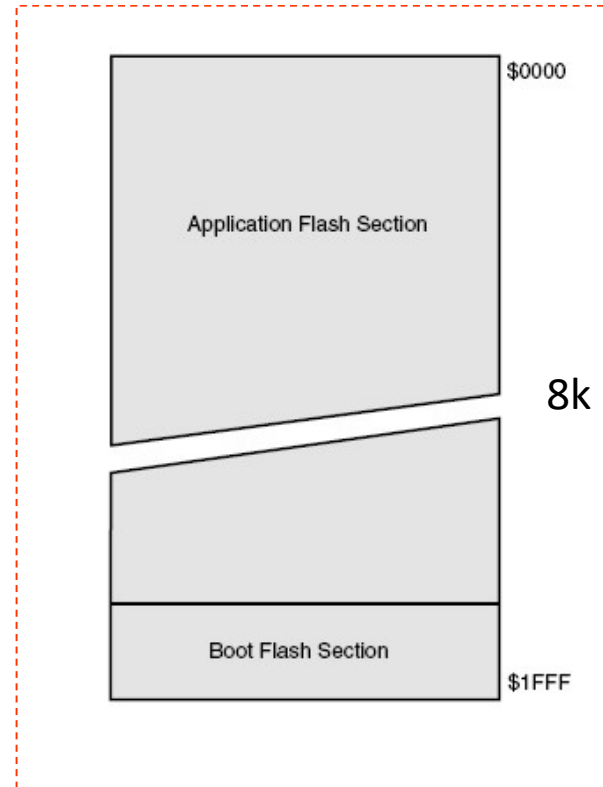
Programmerarmodell

10

Data space

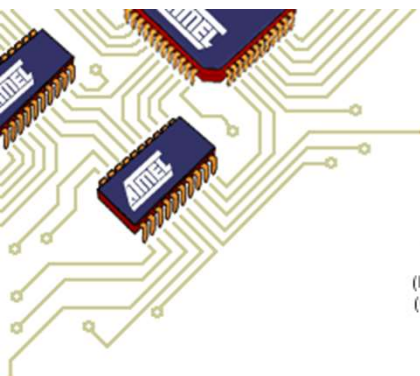


Program space



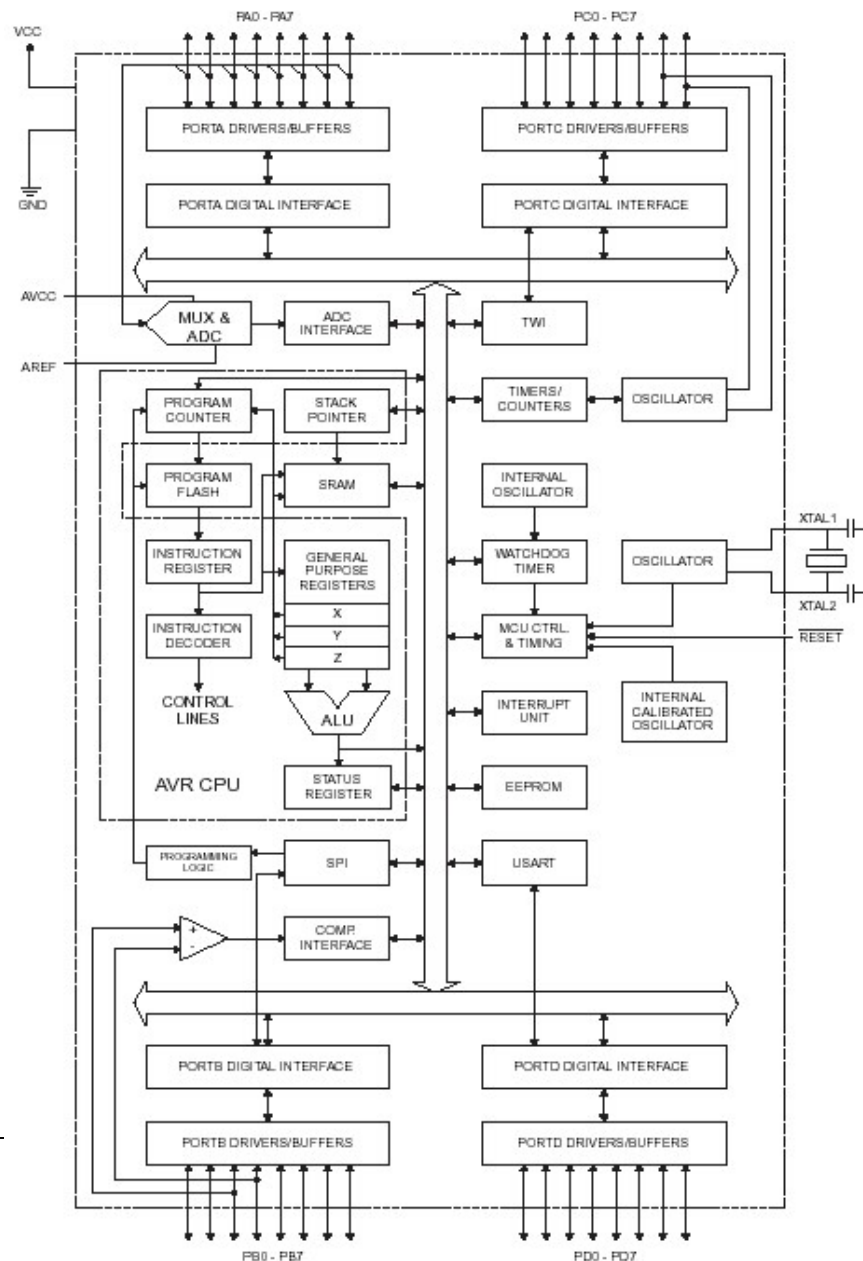
8

16



PDIP			
(XCK/T0) PB0	1	40	PA0 (ADC0)
(T1) PB1	2	39	PA1 (ADC1)
(INT2/AIN0) PB2	3	38	PA2 (ADC2)
(OC0/AIN1) PB3	4	37	PA3 (ADC3)
(SS) PB4	5	36	PA4 (ADC4)
(MOSI) PB5	6	35	PA5 (ADC5)
(MISO) PB6	7	34	PA6 (ADC6)
(SCK) PB7	8	33	PA7 (ADC7)
RESET	9	32	AREF
VCC	10	31	GND
GND	11	30	AVCC
XTAL2	12	29	PC7 (TOSC2)
XTAL1	13	28	PC6 (TOSC1)
(RXD) PD0	14	27	PC5 (TDI)
(TXD) PD1	15	26	PC4 (TDO)
(INT0) PD2	16	25	PC3 (TMS)
(INT1) PD3	17	24	PC2 (TCK)
(OC1B) PD4	18	23	PC1 (SDA)
(OC1A) PD5	19	22	PC0 (SCL)
(ICP) PD6	20	21	PD7 (OC2)

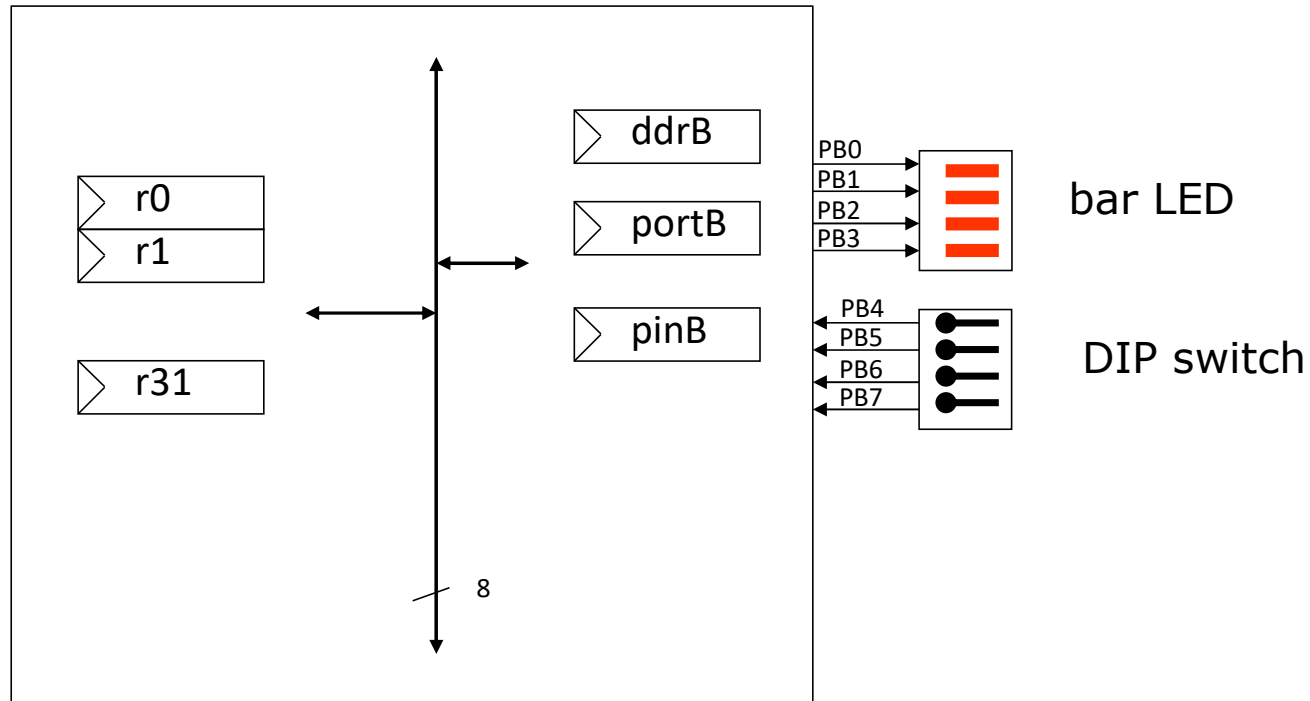
- Stora AVR-en har 4 portar (A,B,C,D) à 8 bitar.
- Varje port kan vara en parallellport eller något annat.
- Till porten hör ett antal I/O-register.

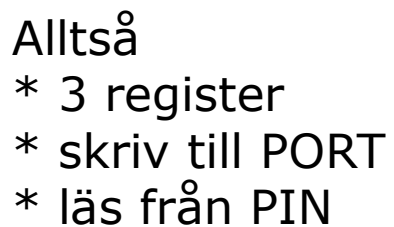


Ett litet exempel

14

AVR





Vanlig digital in/ut-matning - asm

16

```
.include "m16def.inc"
```

```
.org 0
```

```
; Skriv 00001111 i datarikttningsregistret
```

```
ldi r17, (1<<DDB3) | (1<<DDB2) | (1<<DDB1) | (1<<DDB0)
```

```
out DDRB, r17
```

TILLBAKA:

```
; Läs av pinnarna
```

```
in r16, PINB ; kolla tryckknappar 7,6,5,4
```

```
; Skifta ner r16 fyra gånger
```

```
swap r16
```

```
; Skriv ut på LEDarna
```

```
out PORTB, r16
```

```
jmp TILLBAKA
```

Vanlig digital in/ut-matning - C

```
#include <avr/io.h>

int main(void) {
    uint8_t i;
    ...
    /* Define directions for port pins */
    DDRB = (1<<DDB3) | (1<<DDB2) | (1<<DDB1) | (1<<DDB0);

    while (1) {
        /* Läs av DIP-switchen */
        i = PINB; /* kolla tryckknappar */
        /* Skifta 4 steg höger,
        skriv ut på LEDarna */
        PORTB = i>>4;
    }
}
```

Kommentar:

- 1) I/O-programmering likadan som i asm!
- 2) Nytt är variabeln `i`. Var bor den? Hur stor är den?

Assembler <-kontra-> C-programmering

18

- Manualen har exempel för båda

- CPU registren r0-r31 "försvinner"

I/O-programmeringen likadan.
64 I/O-register

Tidskritisk kod?
Mer "känsla" för maskinen ?

- Struktur, if-satser, for-loopar ..., datatyper ...

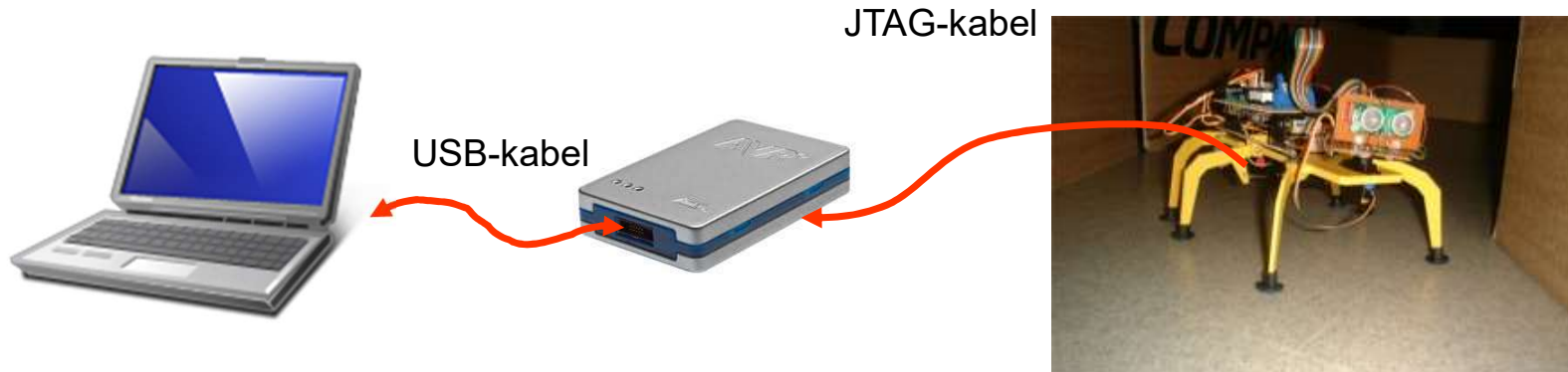
- Kompilatorn fixar kod för avbrottsvektorer

- Det går att blanda, fast gör inte det

Utvecklingsmiljö för Atmel

19

- Windows – Atmel Studio (asm, avr-gcc)



I labbet: Med ICE, exekvering i AVR

- 1) Upload + Programmering
- 2) Läsa/skriva register (vrida på huvudet ...)
- 3) Singlesteppa
- 4) Köra med brytpunkt adress/data

Hemma: Utan ICE, simulering i PCn

- 1) Läsa/skriva register
- 2) Singlesteppa
- 3) Köra med brytpunkt program/data
- 4) Stimuli (mata in 0/1 på pinnarna)

GccApplication1 (Debugging) - AtmelStudio

File Edit View VAssistX ASF Project Build Debug Tools Window Help

Disassembly GccApplication1 GccApplication1.c x

main int main(void) Go

```

/*
 * Created: 2013-09-22 10:57:07
 * Author: olles
 */

#include <avr/io.h>

int8_t a[] = {1,2,3,4,5,6,7,8};
int16_t b=0;

int main(void)
{
    register int16_t mac=0;
    register uint8_t i;

    for (i=0; i<8; i++)
        mac += a[i]*a[i];

    b = mac;
    return(0);
}

```

Processor

Name	Value
Program Counter	0x0000008C
Stack Pointer	0x40F8
X Register	0x010A
Y Register	0x40F8
Z Register	0x0103
Status Register	☐ T ☐ H ☐ S ☐ V ☐ N ☒ Z ☐ C
Cycle Counter	286
Frequency	16,000 MHz
Stop Watch	17,88 µs

Registers

R00	0x00
R01	0x00
R02	0x00
R03	0x00
R04	0x00
R05	0x00
R06	0x00
R07	0x00
R08	0x00

Watch 2

Name	Value	Type
i	3	uint8_t{registers}@R15
mac	30	int16_t{registers}@ R17 R16
b	0	int16_t{data}@0x0108
a	{int8_t[8]{data}@0x0100}	int8_t[8]{data}@0x0100

Memory 1

Memory: prog FLASH

prog 0x000000	45 c0 00 00 60 c0 00 00 5e c0 00 00	EÀ..`À..^À..
prog 0x00000C	5c c0 00 00 5a c0 00 00 58 c0 00 00	\À..ZÀ..XÀ..
prog 0x000018	56 c0 00 00 54 c0 00 00 52 c0 00 00	VÀ..TÀ..RÀ..
prog 0x000024	50 c0 00 00 4e c0 00 00 4c c0 00 00	PÀ..NÀ..LÀ..
prog 0x000030	4a c0 00 00 48 c0 00 00 46 c0 00 00	JÀ..HÀ..FÀ..
prog 0x00003C	44 c0 00 00 42 c0 00 00 40 c0 00 00	DÀ..BÀ..@À..
prog 0x000048	3e c0 00 00 3c c0 00 00 3a c0 00 00	>À..<À..:À..
prog 0x000054	38 c0 00 00 36 c0 00 00 34 c0 00 00	8À..6À..4À..

Autos Locals Watch 1 Watch 2

Stopped

Breakp... Memor... Call Stack Comm... Immedi... Output

GccApplication1 (Debugging) - AtmelStudio

File Edit View VAssistX ASF Project Build Debug Tools Window Help

Disassembly x GccApplication1

Address: main

Viewing Options

```

0000007B SUBI R24,0x00 Subtract immediat
0000007C SBCI R25,0xFF Subtract immediat
0000007D MOVW R30,R24 Copy register pai
0000007E LDD R24,Z+0 Load indirect wit
0000007F MOV R18,R24 Copy register
00000080 CLR R19 Clear Register
00000081 SBRC R18,7 Skip if bit in re
00000082 COM R19 One's complement
00000083 MUL R20,R18 Multiply unsigned
00000084 MOVW R24,R0 Copy register pai
00000085 MUL R20,R19 Multiply unsigned
00000086 ADD R25,R0 Add without carry
00000087 MUL R21,R18 Multiply unsigned
00000088 ADD R25,R0 Add without carry
00000089 CLR R1 Clear Register
0000008A ADD R16,R24 Add without carry
0000008B ADC R17,R25 Add with carry
    for (i=0; i<8; i++)
0000008C INC R15 Increment
    --- No source file ---
0000008D LDI R31,0x07 Load immediate
  
```

Processor

Name	Value
Program Counter	0x0000008C
Stack Pointer	0x40F8
X Register	0x010A
Y Register	0x40F8
Z Register	0x0103
Status Register	THSVNZC
Cycle Counter	286
Frequency	16,000 MHz
Stop Watch	17,88 µs

Registers

Register	Value
R00	0x00
R01	0x00
R02	0x00
R03	0x00
R04	0x00
R05	0x00
R06	0x00

Watch 2

Name	Value	Type
i	3	uint8_t(registers)@R15
mac	30	int16_t(registers)@R17 R16
b	0	int16_t(data)@0x0108
a	{int8_t[8](data)@0x0100}	int8_t[8](data)@0x0100

Memory 1

Memory: data IRAM

Address	Value
data 0x0100	01 02 03 04 05 06 07 08 00 00 00 00
data 0x010C	00 00 00 00 00 00 00 00 00 00 00 00
data 0x0118	00 00 00 00 00 00 00 00 00 00 00 00
data 0x0124	00 00 00 00 00 00 00 00 00 00 00 00
data 0x0130	00 00 00 00 00 00 00 00 00 00 00 00

Autos Locals Watch 1 Watch 2

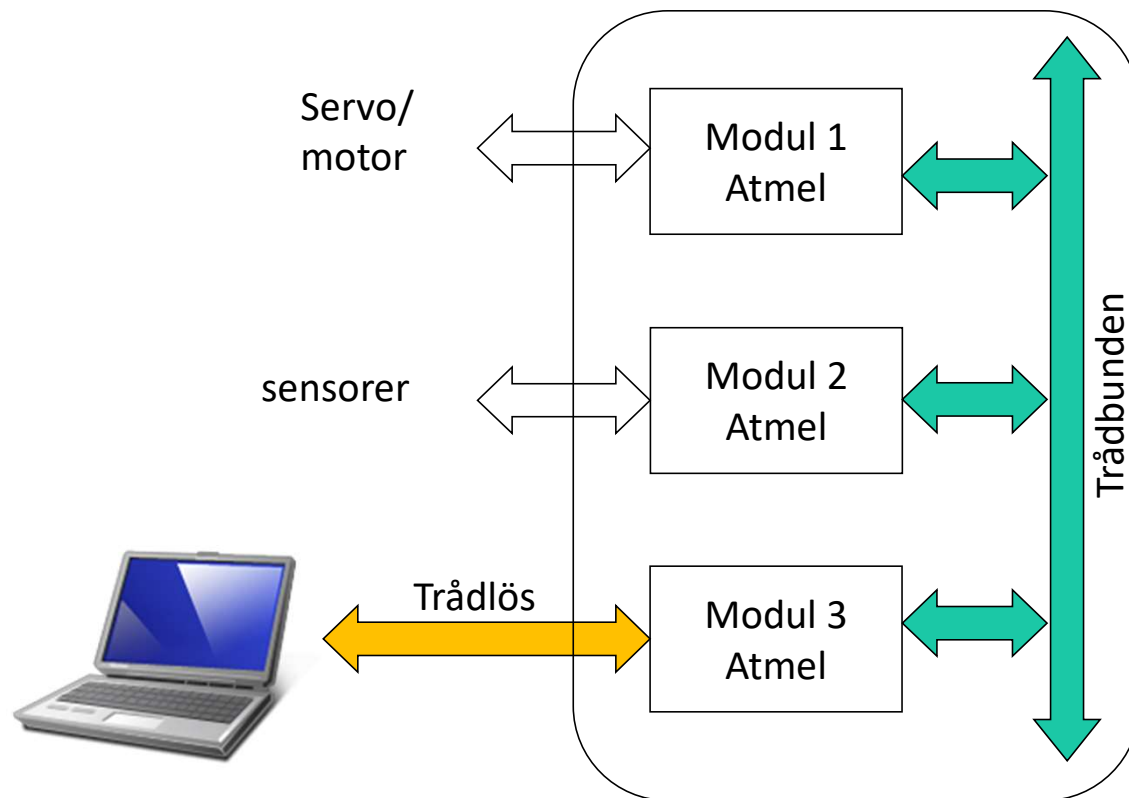
Stopped

Kommunikation (på olika "språk")

- UART
- I2C
- SPI
- Parallell

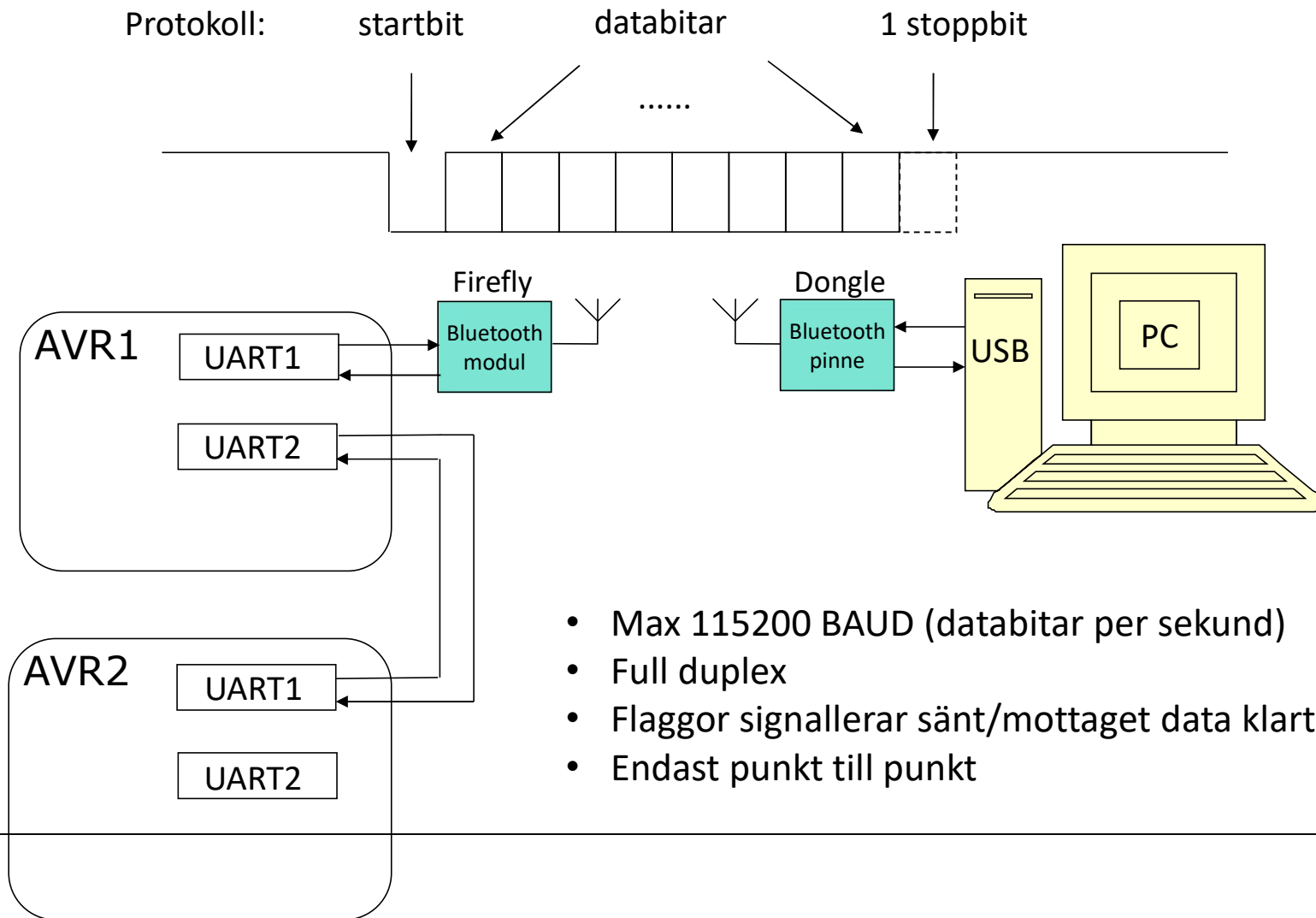
Hur kommunicerar processorerna med varandra och med omvärlden?

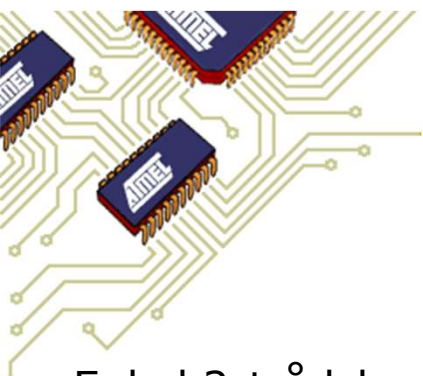
23



1) UART (Universal Asynchronous Receiver/Transmitter)

24



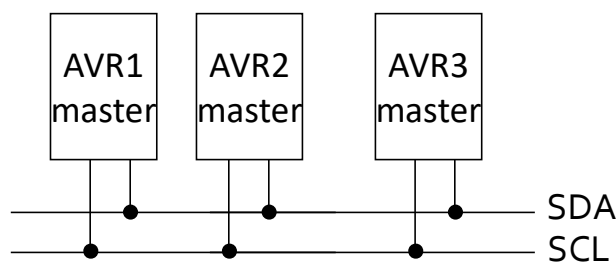
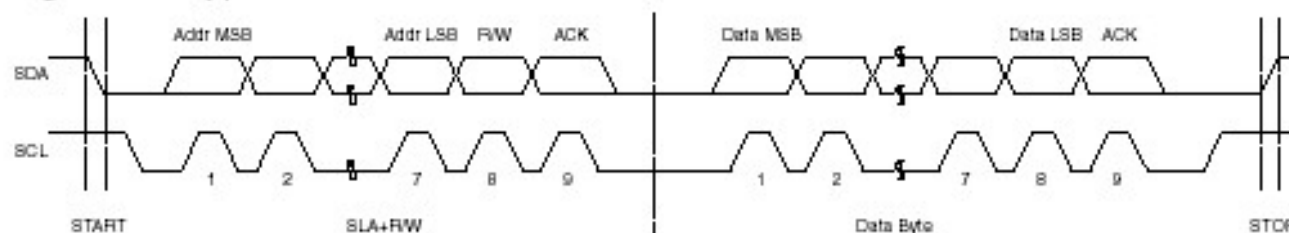


2) I2C - Inter Integrated Circuit TWI - Two Wire Interface

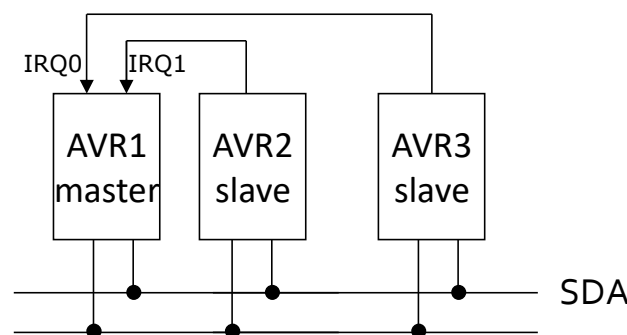
25

- Enkel 2-trådsbuss (+ jord)
- AVR kan vara Master/Slave
- AVR kan vara Sändare/Mottagare
- 7-bits adress => 128 slavar
- Multimaster arbitration supported
- Upp till 400 kHz (200 kHz i praktiken)

Figure 81. Typical Data Transmission



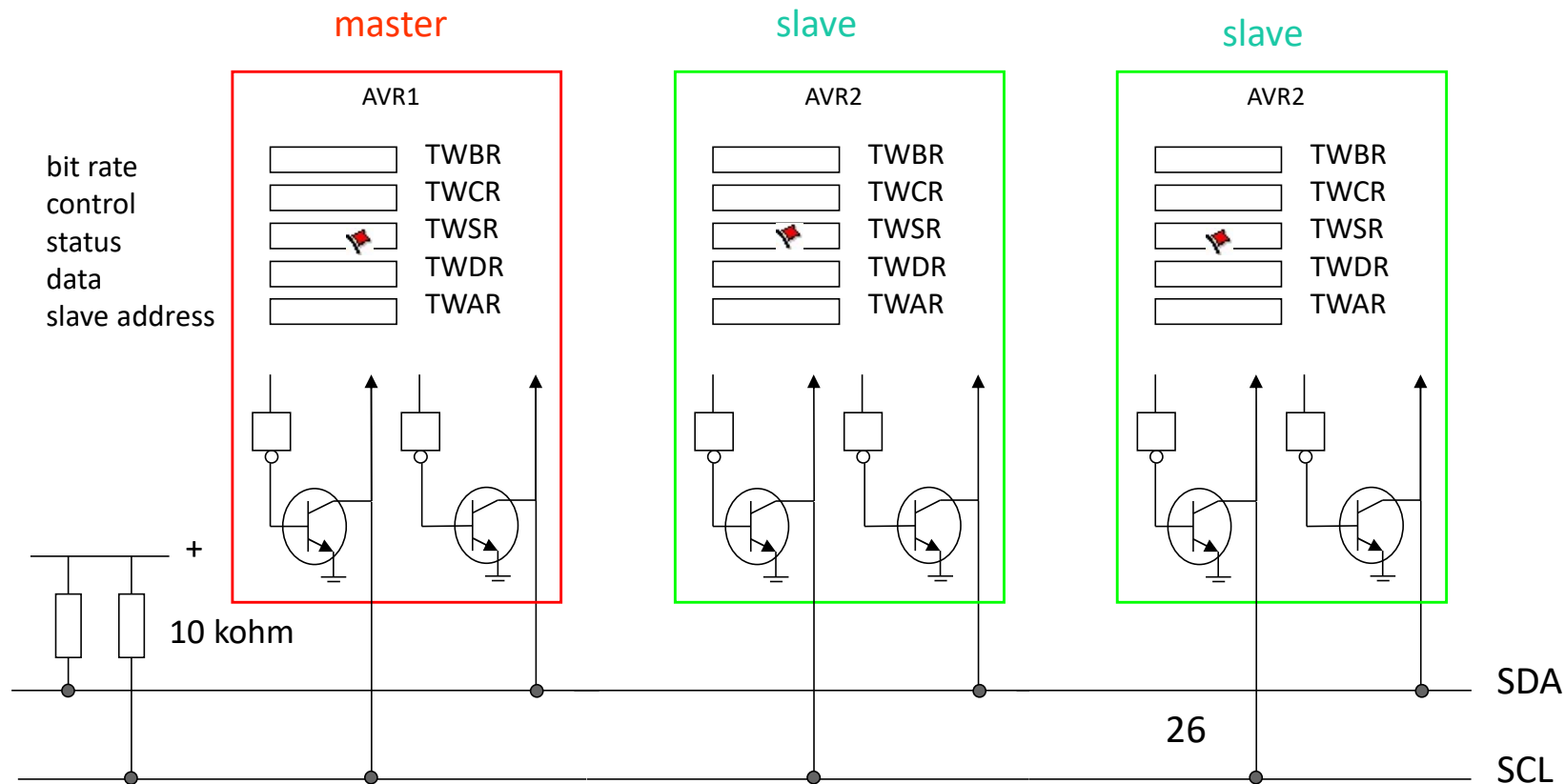
Multimaster
Coolt men svårare!



Master / Slaves
Vanligare, lättare!

2) I2C - Two wire interface Inter-integrated-circuit

26



3) SPI

27

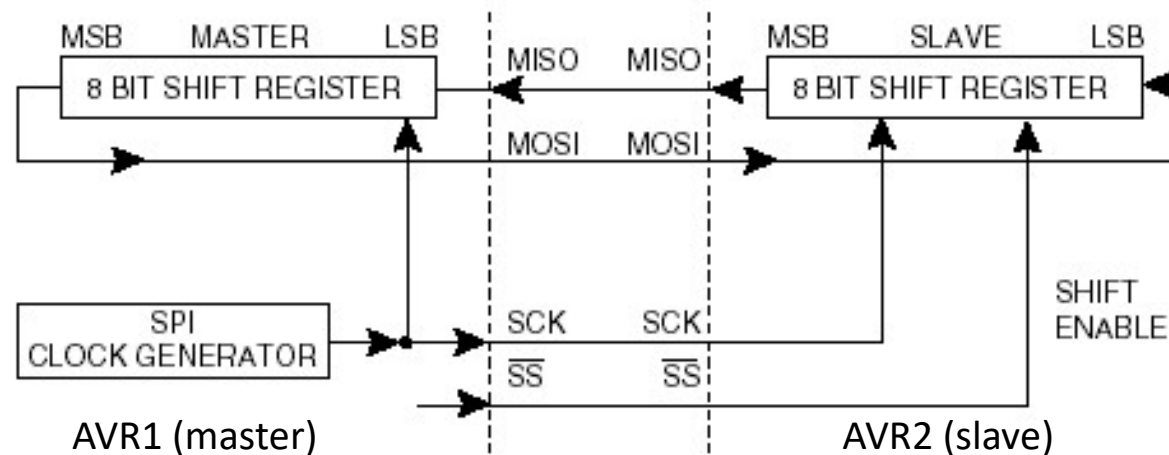
Serial Peripheral Interface – SPI

The Serial Peripheral Interface (SPI) allows high-speed synchronous data transfer between the ATmega16 and peripheral devices or between several AVR devices. The ATmega16 SPI includes the following features:

- Full-duplex, Three-wire Synchronous Data Transfer
- Master or Slave Operation
- LSB First or MSB First Data Transfer
- Seven Programmable Bit Rates
- End of Transmission Interrupt Flag
- Write Collision Flag Protection
- Wake-up from Idle Mode
- Double Speed (CK/2) Master SPI Mode

$$f_{\max} = 8/4 \text{ MHz}$$

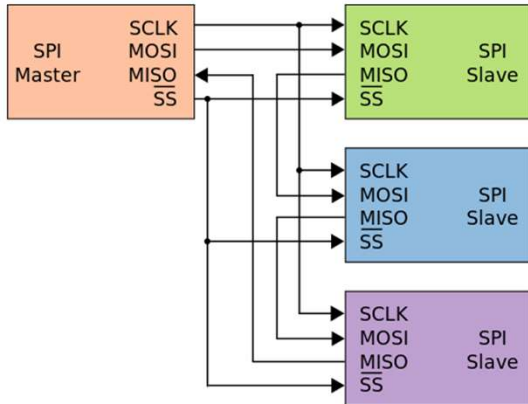
Figure 66. SPI Master-slave Interconnection



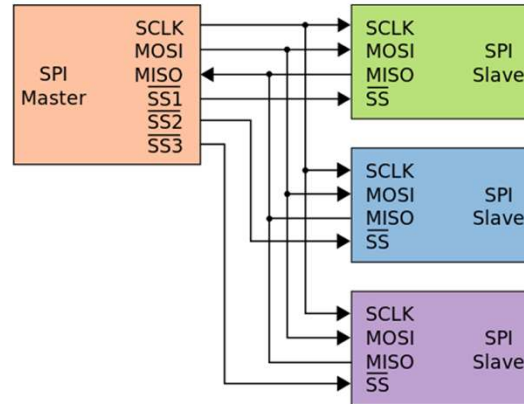
3) SPI

28

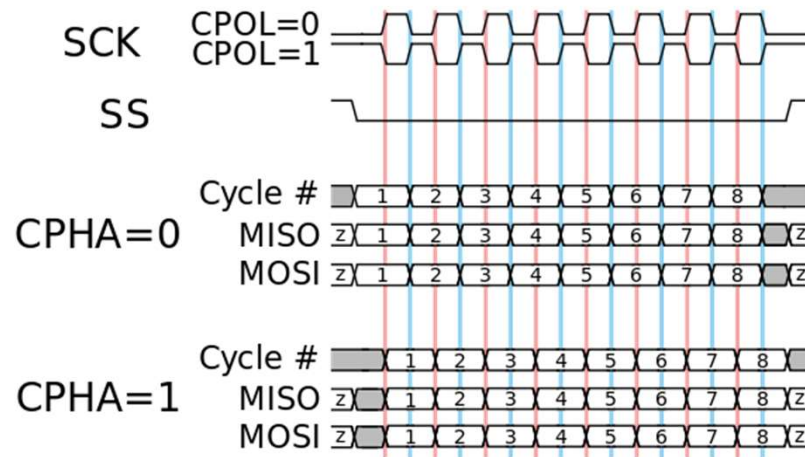
Slavarna kan anslutas seriellt:



Eller parallellt:



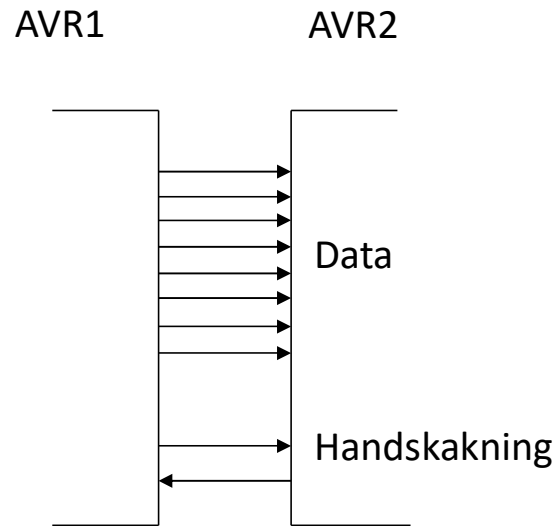
Protokollet:



Bilder från Wikipedia

4) Parallelport

29



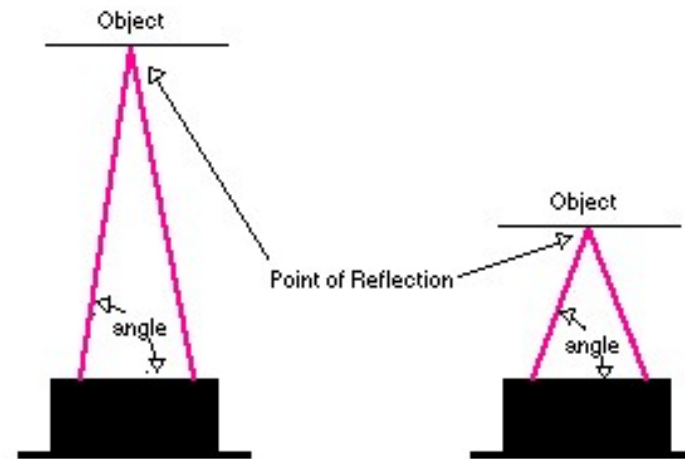
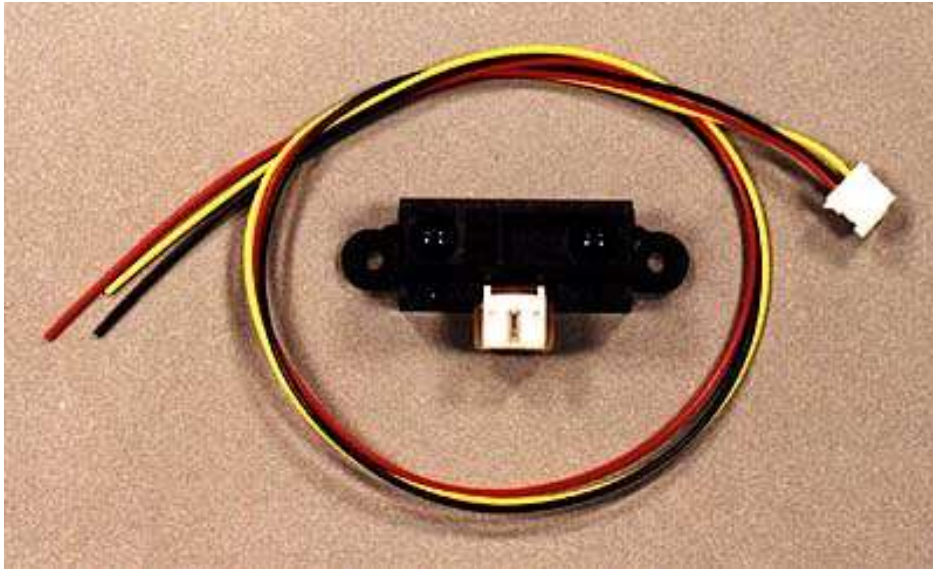
- + Enkelt (?)
- Många pinnar
- Eget protokoll
- Dubbelriktad (?)
- Egen handskakning

Sensorer

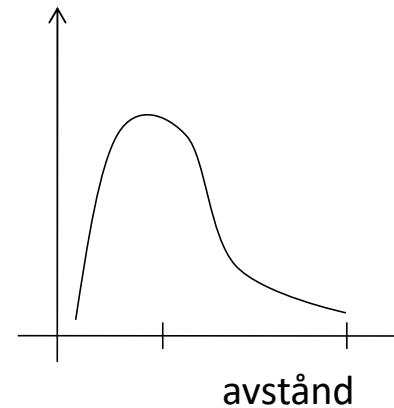
- A/D-omvandling
- Reglerteknik
- Telemetri
- Tidmätning
- Avbrott

Sensor 1: IR-avståndsmätare

31



spänning



Features

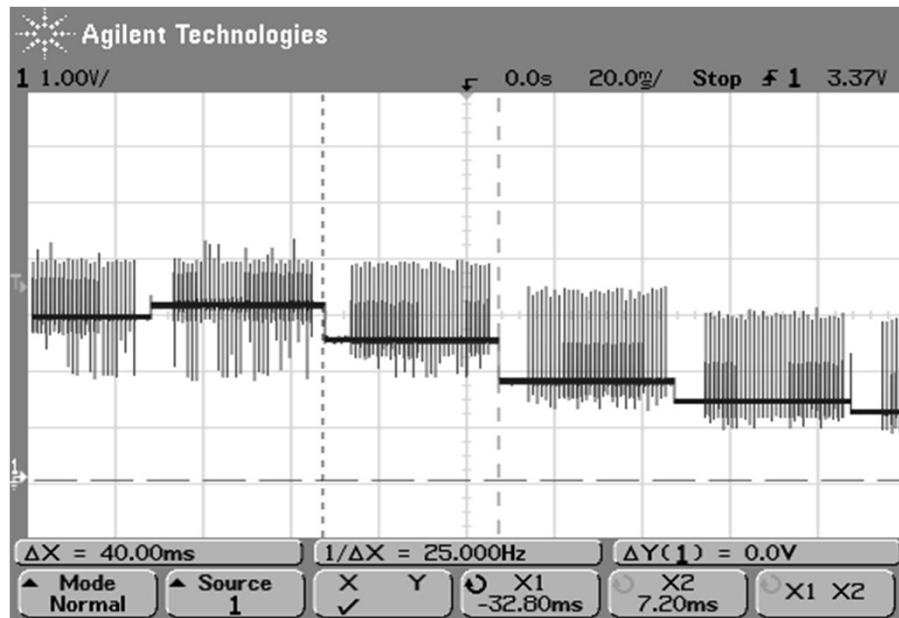
- Less influence on the color of reflected objects, reflectivity
- Analog voltage corresponding to distance
- Detecting distance: 4 to 30 cm, 10 to 80 cm, 20 to 150 cm
- External control circuit unnecessary
- Low cost

Sensordata

32

Ett exempel ur verkligheten:

IR-avståndsmätare ger 25 mätvärden/s i form av analog spänning 0-3V. Vi kopplar den direkt till en AVR, som har A/D-omvandlare.



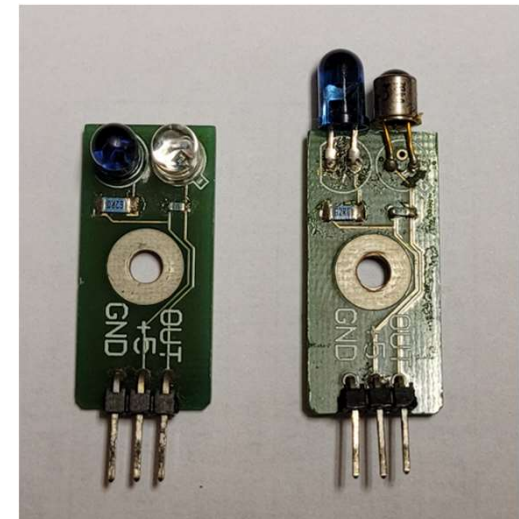
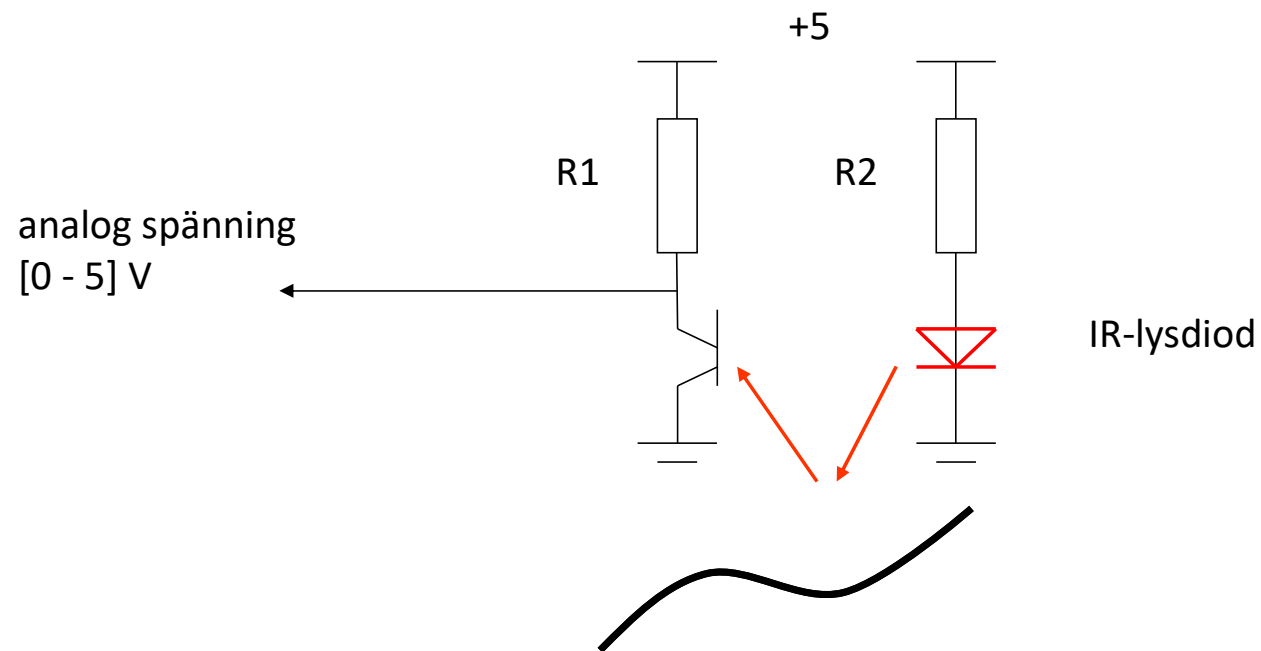
Vid närmare studium av sensordatas kvalitet på oscilloskop upptäcks störningar på mätvärdena!

Bäst är att bli av med dessa, med ett lågpassfilter, innan signalen går in i A/D-omvandlaren.

Sensor 2: Tejpsensorn

mäter tejpens läge

33



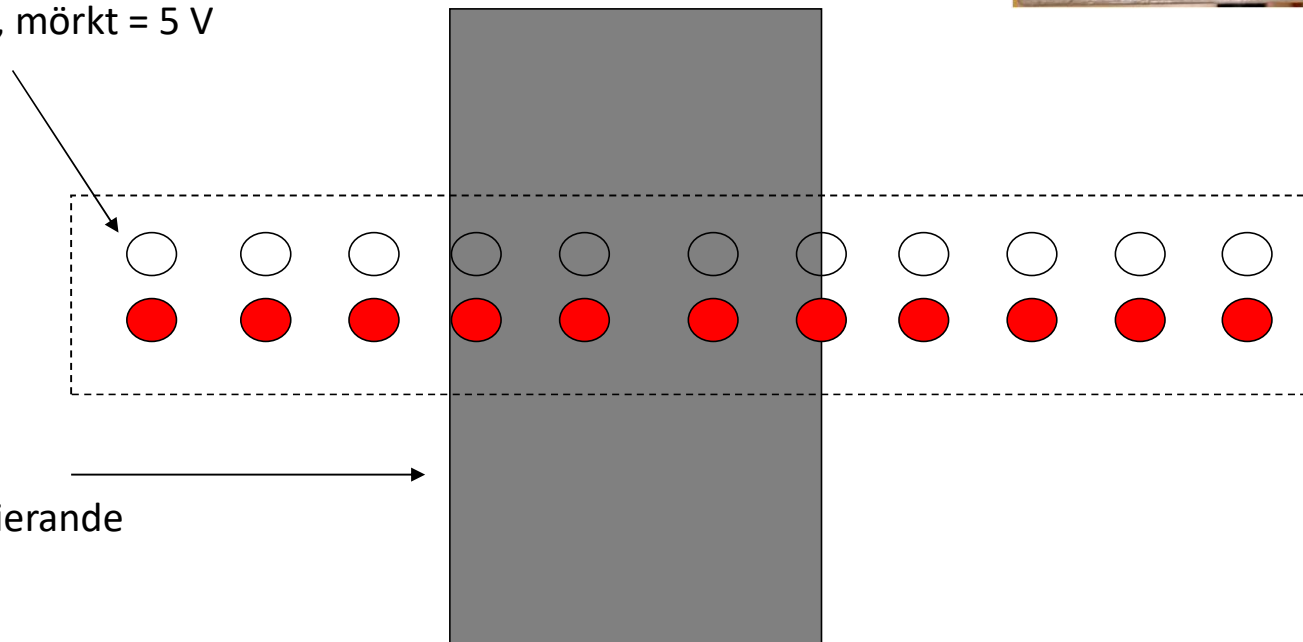
Sensor 3: Linjesensor

34

Vårt sensorpaket,

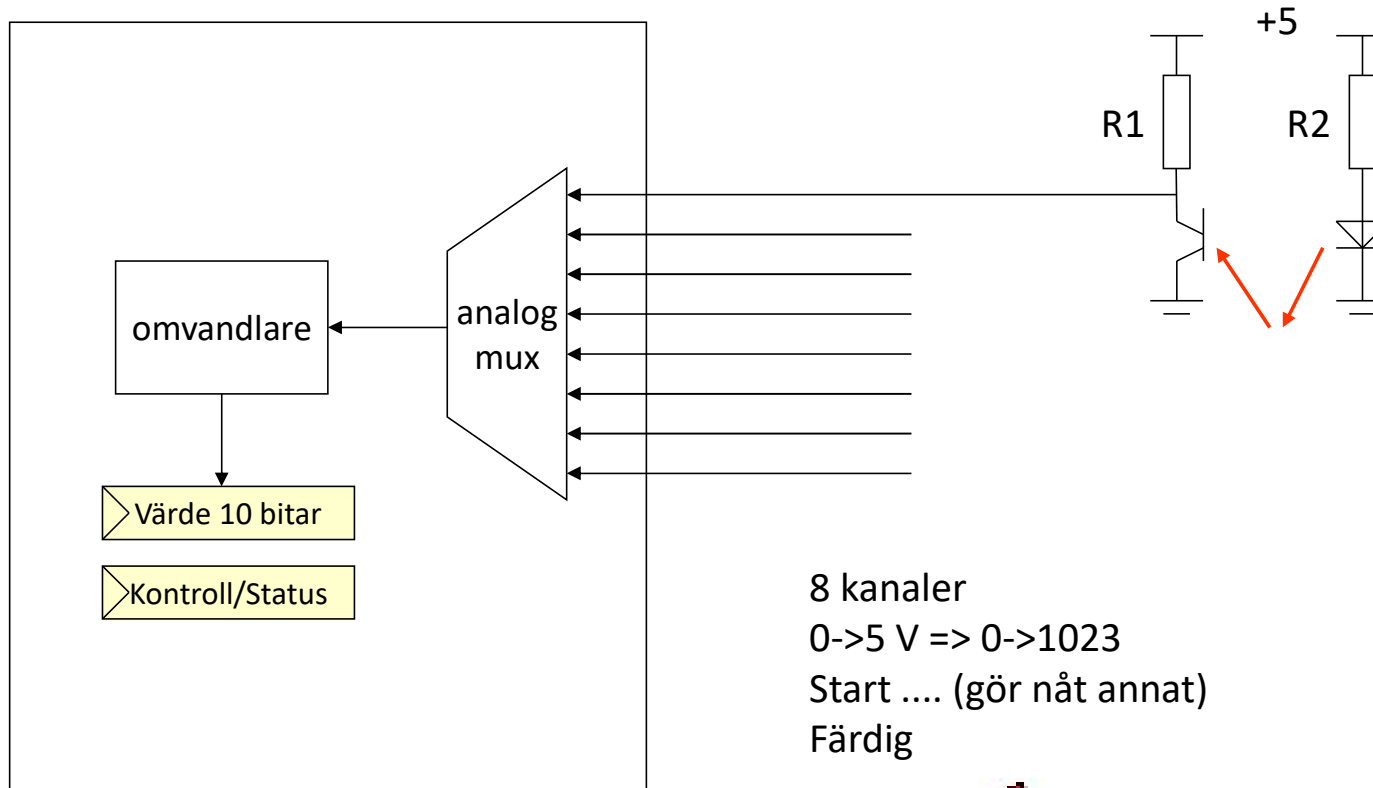
11 sensorer (LED + fototransistor),

- Läses av med A/D
- Varje sensor kontrollerad
 $I_{\text{ljus}} = 0 \text{ V}$, mörkt = 5 V



Att mäta analog spänning: A/D-omvandling

35

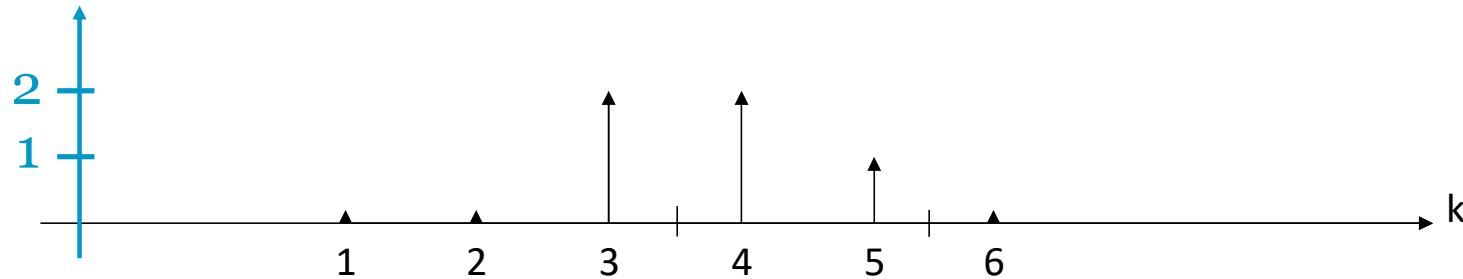


8 kanaler
0-5 V => 0-1023
Start (gör nåt annat)
Färdig



Sensordata

36



Enkel tyngdpunktsberäkning:
$$k_T = \frac{\sum_k m_k k}{\sum_k m_k} = \frac{(2*3 + 2*4 + 1*5)}{(2+2+1)} = \frac{19}{5} = 3,8$$

Felet:
$$e = 3,5 - k_T = 3,5 - 3,8 = -0,3$$

Lite reglerteknik

37

Hur gör man egentligen för att

1) gå mitt i en korridor?

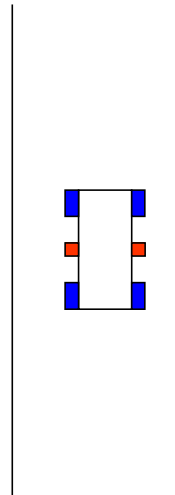
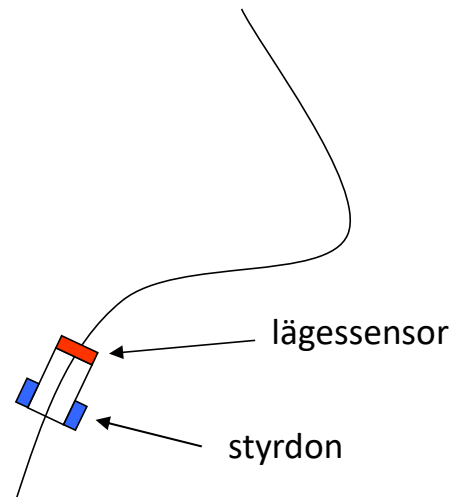
2) följa en linje?

trots att

1) 4-beningen haltar lite grann

2) linjen böjer av

3) golvet lutar



Lite reglerteknik

38

Vi måste konstruera en enkel linjär tidsdiskret regulator.

- 1) Vi bestämmer oss för ett lämpligt ΔT , dvs tiden mellan två sensoravläsningar.
- 2) För roboten i korridoren bildar vi felet $e[n] = x_h[n] - x_v[n]$
För linjeföljaren får vi felet direkt ur sensorn.
- 3) En regulator bildar en lämpligt styrsignal $u[n]$ mha av felet $e[n]$.



Vi antar att $u[n] = 0$ rakt fram
 $u[n] > 0$ sväng höger
 $u[n] < 0$ sväng vänster

och att $e[n] = 0$ mitt på linjen
 $e[n] > 0$ vänster om linjen
 $e[n] < 0$ höger om linjen

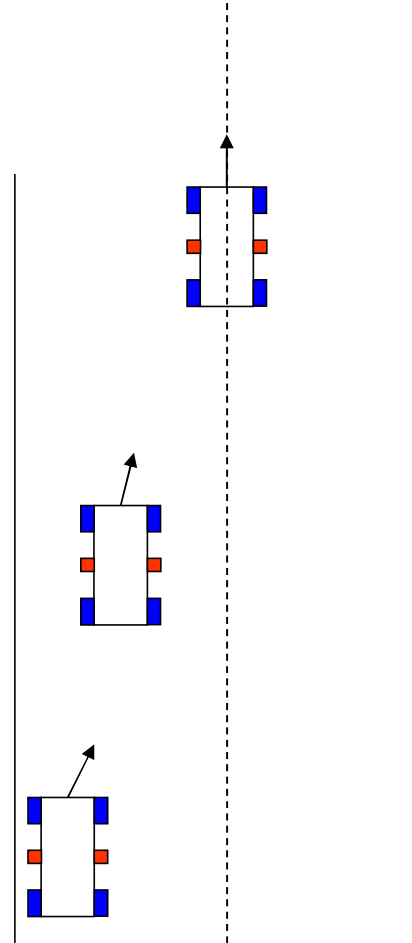
Lite reglerteknik

39

Enklaste regulatorn: P-reglering

$$u[n] = K_p * e[n]$$

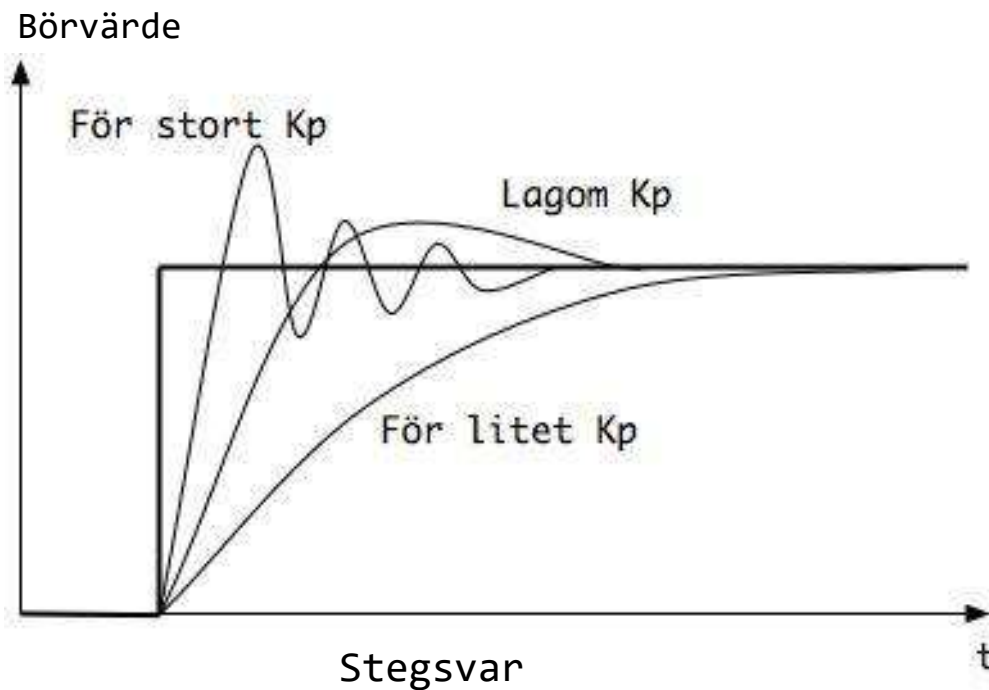
Rattutslaget proportionellt mot felet, dvs olika rattutslag (+/-) för olika storlekar på felet



Lite reglerteknik

40

Vad händer för olika storlekar på K_p ?



För stort K_p :
risk för självsvängning.

För litet K_p :
insvängning tar för lång
tid.

Lagom K_p :
har en liten översläng, som kan
reduceras med en deriverande del (D-delen)

Lite reglerteknik

41

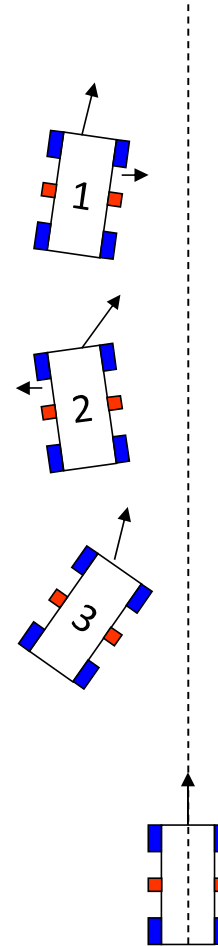
Näst enklaste regulatorn: PD-reglering

$$u[n] = K_P * e[n] + K_D * (e[n] - e[n-1])$$

Alla bilarna har samma $e[n]$
=> samma positiva P-del

- 1) Neg D-del => minskat rattutslag
- 2) Pos D-del => ökat rattutslag
- 3) Mycket neg D-del =>
styr åt andra hållet

I-del behövs ej, kvarstående reglerfel
finns ej här! $e=0 \Rightarrow u=0$



Lite reglerteknik

42

OBS:

1) Det är viktigt att känna till ΔT .

PD-formeln ska ju egentligen se ut så här:

$$\begin{aligned} u[n] &= K_P \cdot e[n] + K_D \frac{e[n] - e[n-1]}{\Delta T} \\ &= K_P \cdot e[n] + \frac{K_D}{\Delta T} (e[n] - e[n-1]) \end{aligned}$$

Det är ju felderivatan vi ska reagera på!!!

Styregenskaperna ska inte ändras om ΔT ändras!

Variant: om vinkeln kan mätas

$$u[n] = K_P \cdot e[n] + K_2 \cdot \phi$$

Var försiktig
med talområdet!
Använd C,
`int16_t u, e, ...;`

Styrstopp?

```
if u > MAX
    u = max; ...
```

Lite reglerteknik

43

Hela PID-regulatorn, inklusive I-del:

$$u[n] = K_p * e[n] + K_I * (e[n] + \dots + e[0]) + K_D * (e[n] - e[n-1])$$

När behövs I-delen?

När P- eller PD-regulatorn inte helt kan ta bort felet.

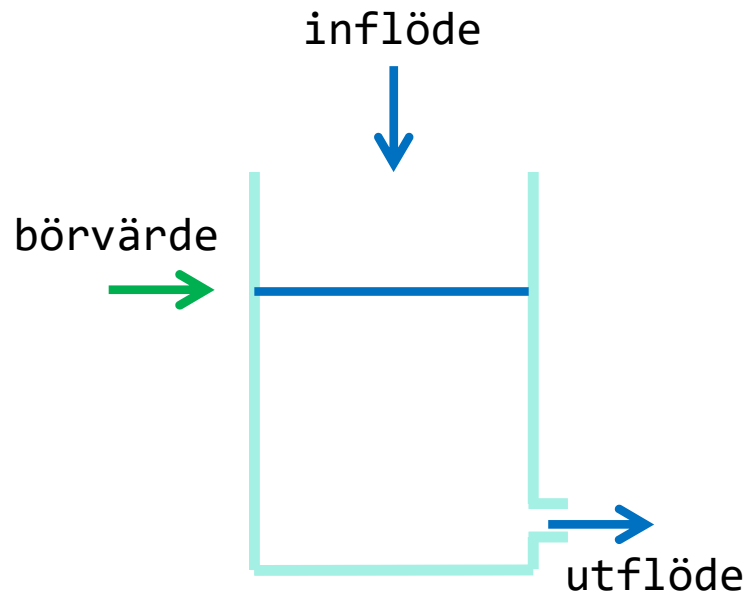
Typiskt när omgivande förutsättningar förändras.

T ex vid reglering av nivån i en vattentank, där vatten ständigt konsumeras.

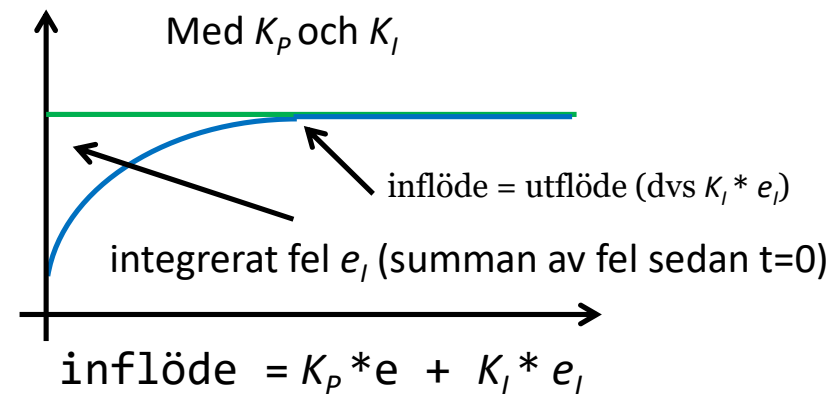
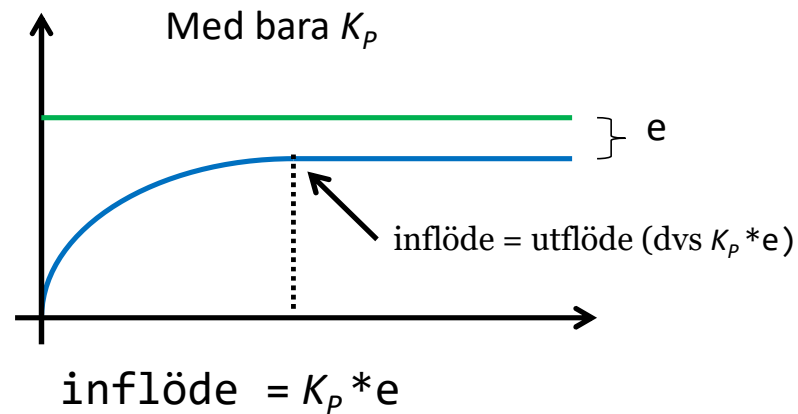
T ex när något befinner sig i fritt fall, som ska behålla konstant höjd.

Lite reglerteknik

44



Vid $t=0$ är vattentanken tom och reglersystemet startar

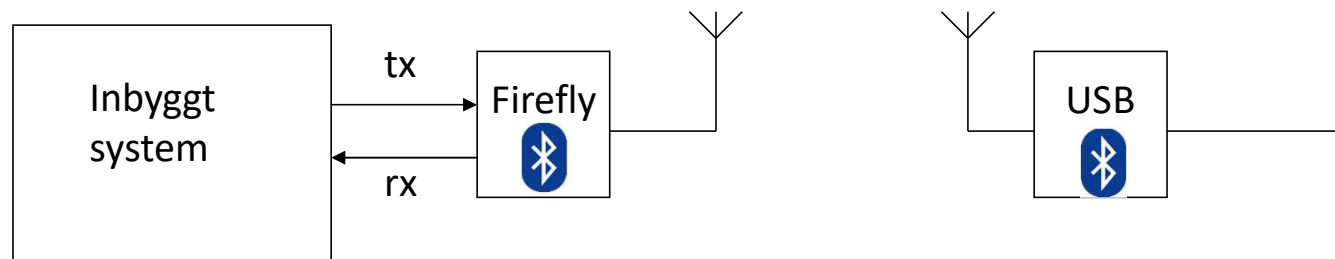


Telemetri

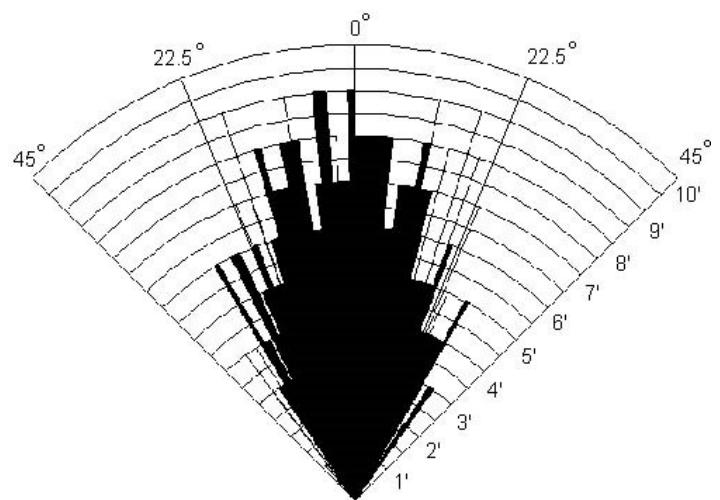
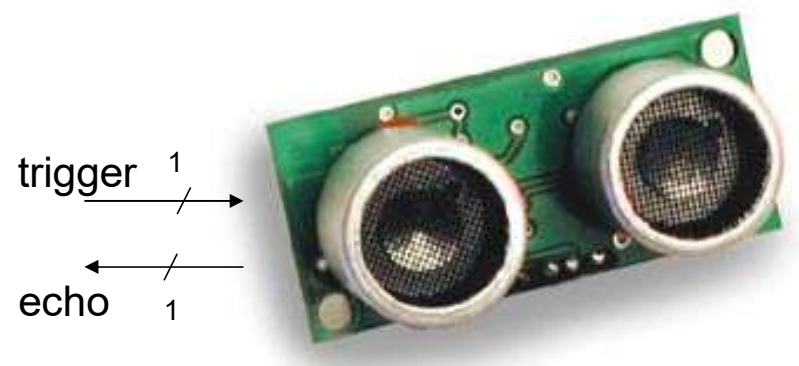
45

Det är bra! Men varför då?

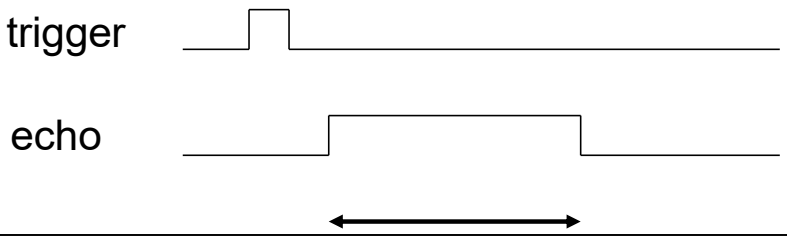
- Lätt att se inre tillståndet i systemet, när det är i drift och rörelse!
- Lätt att ställa in reglerparametrar, i realtid!



Sensor 4: Ultraljud



Voltage	5v
Frequency	40KHz
Max Range	3 m
Min Range	3 cm
Sensitivity	Detect a 3cm diameter stick at > 2 m
Input Trigger	10uS Min. TTL level pulse
Echo Pulse	Positive TTL level signal, width proportional to range.



Timer/Counter

AVR har 3 timer/counters (ATMega 16)

Detta är **timer/counter1**.

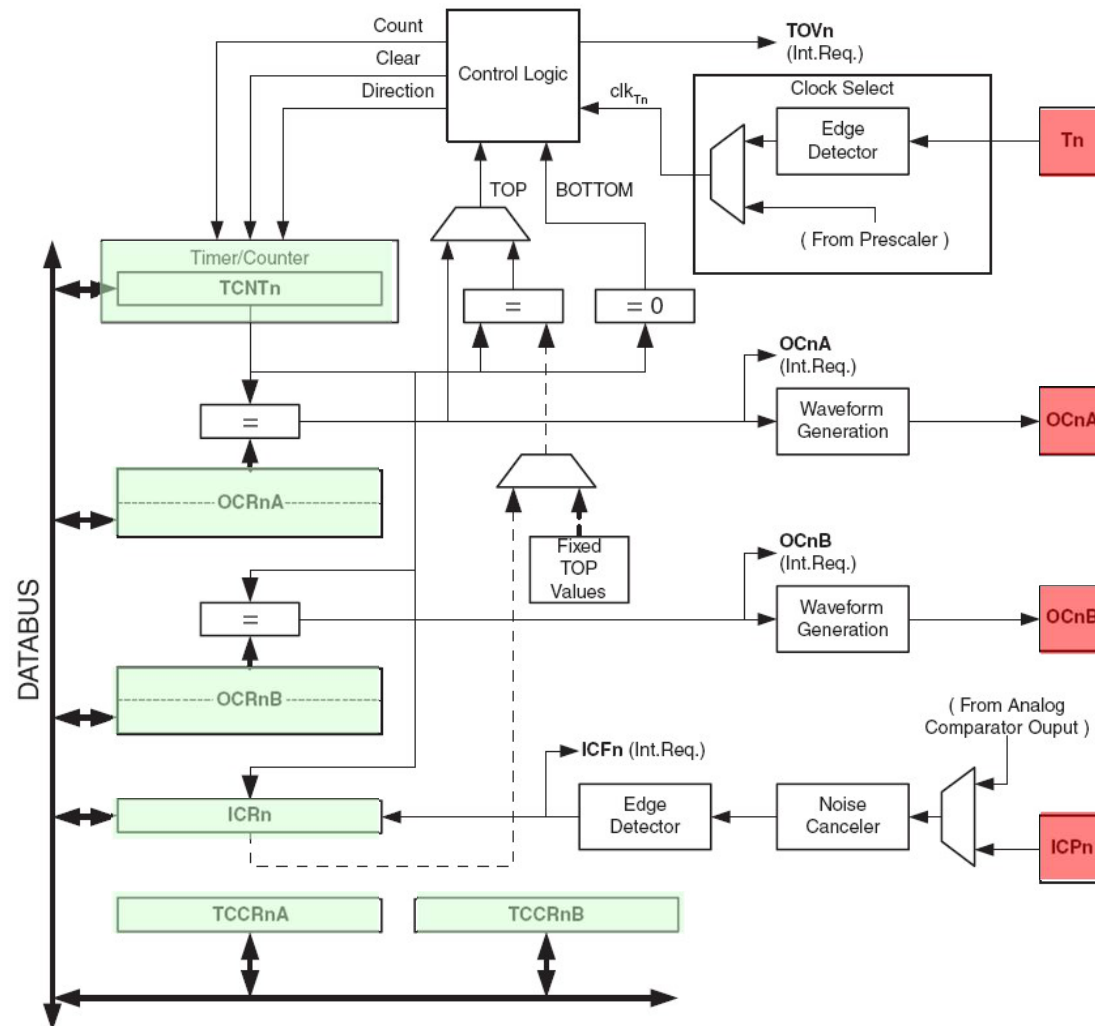
"Simplified block diagram"!

16-bits räknare

jämförelserregister

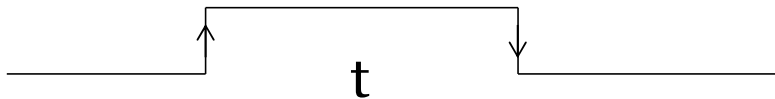
tidregister

kontrollregister



Att mäta tidpulser

48



Polling

- 1) **Vänta på att pinnen går hög**
- 2) Läs av räknaren TCNT1
- 3) **Vänta på att pinnen går låg**
- 4) Läs av räknaren TCNT1
- 5) Subtrahera

Avbrott

- 1) Sätt upp ICR1 för att ta en timestamp på positiv flank på pinnen

I avbrottsrutinen

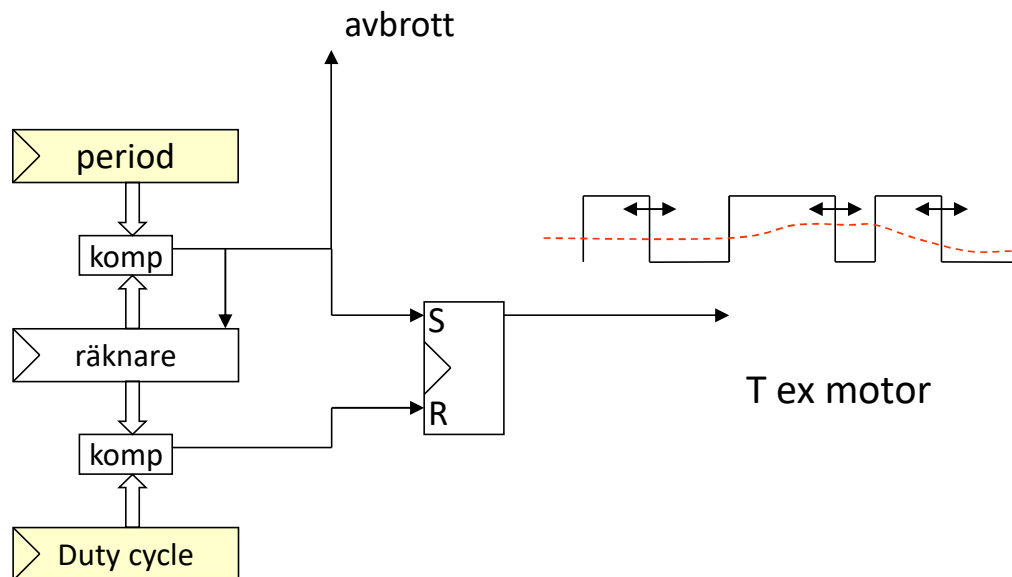
Läs av TCNT1, lägg i en variabel och ställ om polariteten på flanken

Variabeln innehåller "avståndet"!

Att generera tidpulser Pulsbreddsmodulering (PWM)

49

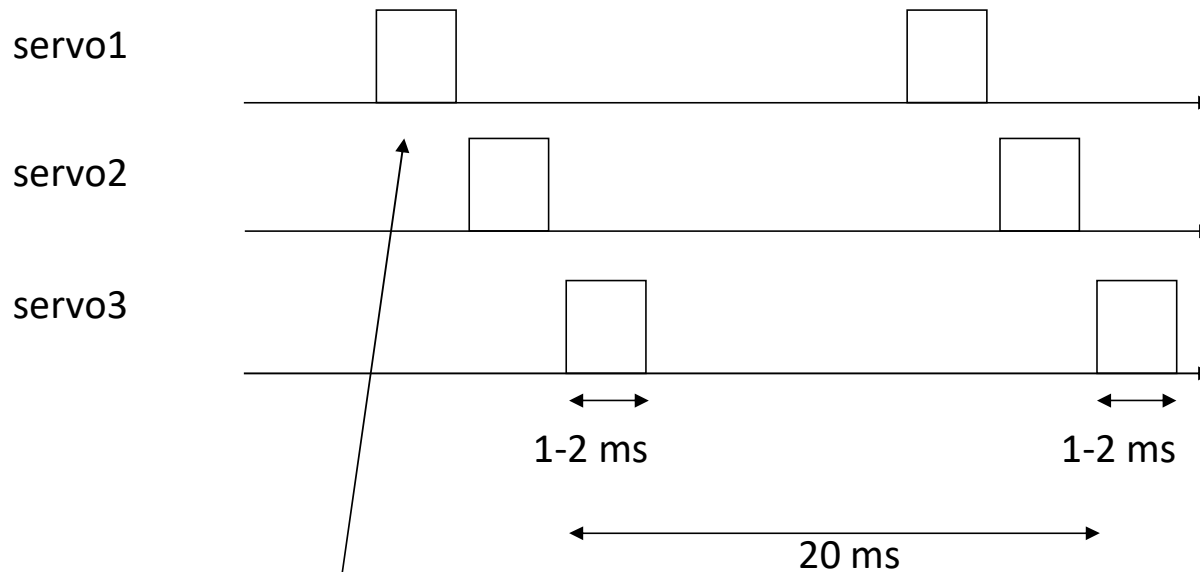
”D/A-omvandling”



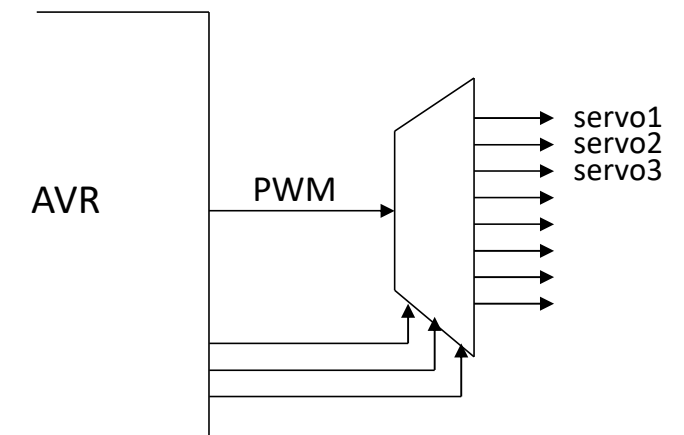
Kontrollera om motorn/servot reagerar på pulskvot eller pulsbredd!

De flesta servon reagerar på pulsbredd

51



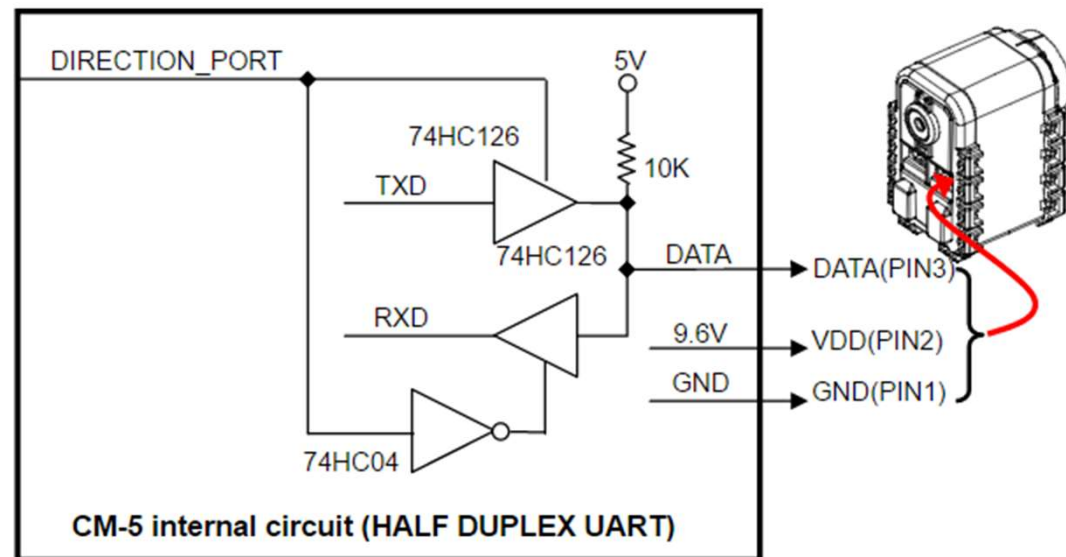
Pulsens bredd avgör servots läge.
1.0 ms => max vridning ena hållet
1.5 ms => mittläge
2.0 ms => max vridning andra hållet
Pulsbredden behöver skickas var 20:e ms för att upprätthålla läget.



Dynamixel-servo (gående robot + robotarm)

53

- Kommunikerar via UART, 1MBaud!
- Kan rotera med olika hastigheter
- Vinkelområdet kan begränsas
- Drivkraft kan begränsas
- Belastning, spänning, temperatur, position m m kan läsas av



Dynamixel-servo (gående robot + robotarm)

Styrs via många
olika parametrar
i servots interna minne →

3-4. Control
Table

EEPROM
Area

RAM
Area

Address	Item	Access	Initial Value
0(0X00)	Model Number(L)	RD	12(0x0C)
1(0X01)	Model Number(H)	RD	0(0x00)
2(0X02)	Version of Firmware	RD	?
3(0X03)	ID	RD,WR	1(0x01)
4(0X04)	Baud Rate	RD,WR	1(0x01)
5(0X05)	Return Delay Time	RD,WR	250(0xFA)
6(0X06)	CW Angle Limit(L)	RD,WR	0(0x00)
7(0X07)	CW Angle Limit(H)	RD,WR	0(0x00)
8(0X08)	CCW Angle Limit(L)	RD,WR	255(0xFF)
9(0X09)	CCW Angle Limit(H)	RD,WR	3(0x03)
10(0X0A)	(Reserved)	-	0(0x00)
11(0X0B)	the Highest Limit Temperature	RD,WR	85(0x55)
12(0X0C)	the Lowest Limit Voltage	RD,WR	60(0X3C)
13(0X0D)	the Highest Limit Voltage	RD,WR	190(0xBE)
14(0X0E)	Max Torque(L)	RD,WR	255(0xFF)
15(0X0F)	Max Torque(H)	RD,WR	3(0x03)
16(0X10)	Status Return Level	RD,WR	2(0x02)
17(0X11)	Alarm LED	RD,WR	4(0x04)
18(0X12)	Alarm Shutdown	RD,WR	4(0x04)
19(0X13)	(Reserved)	RD,WR	0(0x00)
20(0X14)	Down Calibration(L)	RD	?
21(0X15)	Down Calibration(H)	RD	?
22(0X16)	Up Calibration(L)	RD	?
23(0X17)	Up Calibration(H)	RD	?
24(0X18)	Torque Enable	RD,WR	0(0x00)
25(0X19)	LED	RD,WR	0(0x00)
26(0X1A)	CW Compliance Margin	RD,WR	0(0x00)
27(0X1B)	CCW Compliance Margin	RD,WR	0(0x00)
28(0X1C)	CW Compliance Slope	RD,WR	32(0x20)
29(0X1D)	CCW Compliance Slope	RD,WR	32(0x20)
30(0X1E)	Goal Position(L)	RD,WR	[Addr36]value
31(0X1F)	Goal Position(H)	RD,WR	[Addr37]value
32(0X20)	Moving Speed(L)	RD,WR	0
33(0X21)	Moving Speed(H)	RD,WR	0
34(0X22)	Torque Limit(L)	RD,WR	[Addr14] value
35(0X23)	Torque Limit(H)	RD,WR	[Addr15] value
36(0X24)	Present Position(L)	RD	?
37(0X25)	Present Position(H)	RD	?
38(0X26)	Present Speed(L)	RD	?
39(0X27)	Present Speed(H)	RD	?
40(0X28)	Present Load(L)	RD	?
41(0X29)	Present Load(H)	RD	?
42(0X2A)	Present Voltage	RD	?
43(0X2B)	Present Temperature	RD	?
44(0X2C)	Registered Instruction	RD,WR	0(0x00)
45(0X2D)	(Reserved)	-	0(0x00)
46(0x2E)	Moving	RD	0(0x00)
47(0x2F)	Lock	RD,WR	0(0x00)
48(0x30)	Punch(L)	RD,WR	32(0x20)
49(0x31)	Punch(H)	RD,WR	0(0x00)

Avbrottskällor i ATmega16

55

Vector No.	Program Address ⁽²⁾	Source	Interrupt Definition
1	\$000 ⁽¹⁾	RESET	External Pin, Power-on Reset, Brown-out Reset, Watchdog Reset, and JTAG AVR Reset
2	\$002	INT0	External Interrupt Request 0
3	\$004	INT1	External Interrupt Request 1
4	\$006	TIMER2 COMP	Timer/Counter2 Compare Match
5	\$008	TIMER2 OVF	Timer/Counter2 Overflow
6	\$00A	TIMER1 CAPT	Timer/Counter1 Capture Event
7	\$00C	TIMER1 COMPA	Timer/Counter1 Compare Match A
8	\$00E	TIMER1 COMPB	Timer/Counter1 Compare Match B
9	\$010	TIMER1 OVF	Timer/Counter1 Overflow
10	\$012	TIMER0 OVF	Timer/Counter0 Overflow
11	\$014	SPI, STC	Serial Transfer Complete
12	\$016	USART, RXC	USART, Rx Complete
13	\$018	USART, UDRE	USART Data Register Empty
14	\$01A	USART, TXC	USART, Tx Complete
15	\$01C	ADC	ADC Conversion Complete
16	\$01E	EE_RDY	EEPROM Ready
17	\$020	ANA_COMP	Analog Comparator
18	\$022	TWI	Two-wire Serial Interface
19	\$024	INT2	External Interrupt Request 2
20	\$026	TIMER0 COMP	Timer/Counter0 Compare Match
21	\$028	SPM_RDY	Store Program Memory Ready

Vektornamn i C-program

INT0_vect
 INT1_vect
 TIMER2_COMP_vect
 TIMER2_OVF_vect
 TIMER1_CAPT_vect
 TIMER1_COMPA_vect
 TIMER1_COMPB_vect
 TIMER1_OVF_vect
 TIMER0_COMP_vect
 SPI_STC_vect
 USART_RXC_vect
 USART_UDRE_vect
 USART_TXC_vect
 ADC_vect
 EE_RDY_vect
 ANA_COMP_vect
 TWI_vect
 INT2_vect
 TIMER0_COMP_vect
 SPM_RDY_vect

Avbrott i AVR mega 16 (assembler)

56

```
.org 0x0      ; hopptabell
jmp RESET    ;
jmp EXT_INT0 ; yttre avbrott
...
.org 0x1c
jmp ADC      ; A/D omvandlaren klar
```

```
RESET: ...      ; här börjar
...           ; mitt huvudprogram
jmp RESET
```

```
EXT_INT0:      ; yttre avbrott
reti
```

```
ADC:           ; A/D-omvandlaren
reti          ; klar
```

- 1) En händelse sätter en 
- 2) Om avbrottet är aktiverat så fås hopp till avbrottsrutin
- 3) Avbrottskoden måste:
Spara undan SREG
Ta ner 
Betjäna (köra avbrottskoden)
Ta tillbaka SREG
reti

Avbrott i AVR mega 16 (C-program)

57

```
#include <avr/interrupt.h>
```

```
ISR(INT0_vect)
{
    ...    // kod vid externt avbrott INT0
}
```

```
ISR(ADC_vect)
{
    ...    // kod vid A/D-omvandling klar
}
```

```
void main()
{
    ...    // kod för konfigurering av avbrott
    sei(); // aktivera avbrott globalt
    while(1)
    {
        ... // huvudloop, vänta på avbrott
    }
}
```

1) En händelse sätter en flagga 

2) Om avbrottet är aktiverat
så fås hopp till avbrottsrutin

3) Avbrottet kommer
automatiskt att:
-Spara undan SREG
-Ta ner 
-Betjäna (köra avbrottskoden)
-Ta tillbaka SREG
-Återgå

Sensor 5: RFID

58



RFID Tags

RFID Card Reader
RS232 (UART)
2400 BAUD

Sensor 6: LIDAR-lite v2/v3

59



Laseravståndssensor

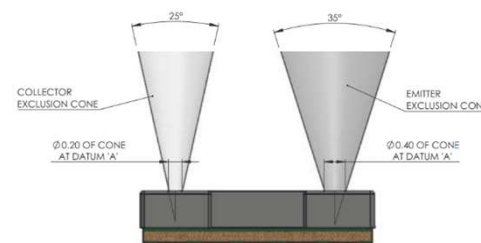
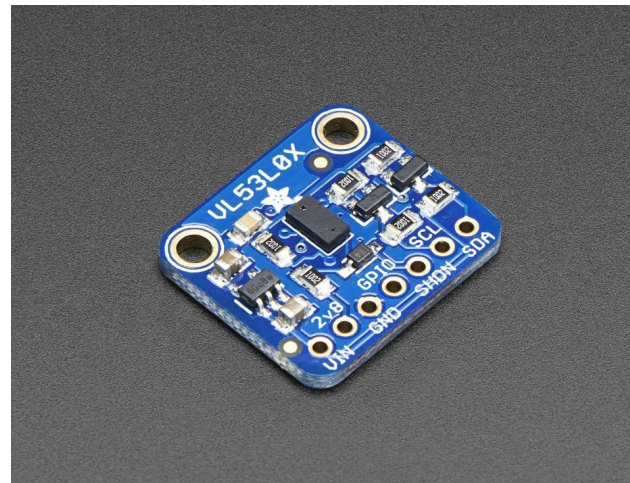
- 15 cm upp till 40 m (1 cm upplösning)
- I2C-interface

Sensor 7 : VL53L0X

Time Of Flight, distance sensor

61

- 50-1200 mm
- 3-5V matning
- 3-5V logik
- I2C-interface
(adressbyte via I2C)
- 25-35° sensor-kon



Sensor 8: Angular rate sensor ("Gyro")

63



MLX90609 Angular Rate Sensor (Standard version)

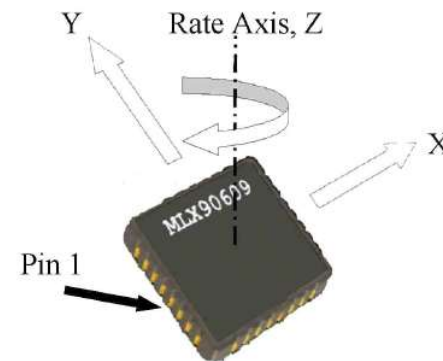
SPI och
analog utgång →

Standard Features and Benefits

- ☐ High resolution and dynamic range
- ☐ Both digital (SPI) and analog outputs
- ☐ Low acceleration and angular rate cross sensitivity
- ☐ Low zero rate output drift
- ☐ Cost effective and compact solution
- ☐ High-performance MEMS sensor in mono crystalline Si yielding a superior long term behavior reliability and dynamic range
- ☐ Programmable bandwidth
- ☐ Factory set full scale range
- ☐ On chip EEPROM calibration
- ☐ Small footprint (SMD CLCC32) with horizontal mounting
- ☐ Operating temperature range: -40 °C to 85 °C

Application

- ☐ Navigation (dead reckoning)
- ☐ Vehicle stability
- ☐ Robots

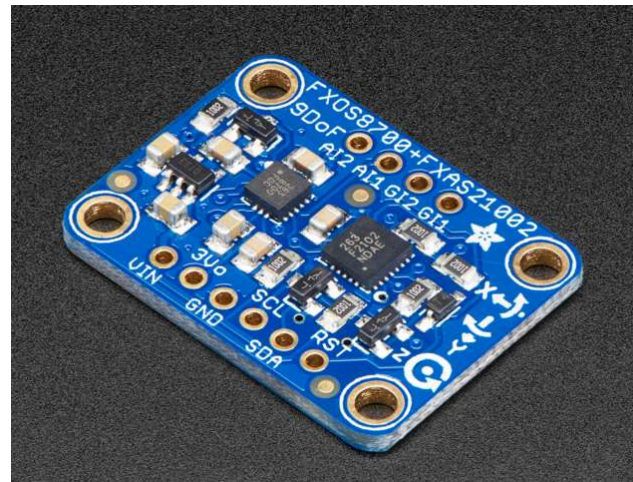


Sensor 9 : Adafruit Precision NXP

67

9-DOF Breakout Board - FXOS8700 + FXAS21002

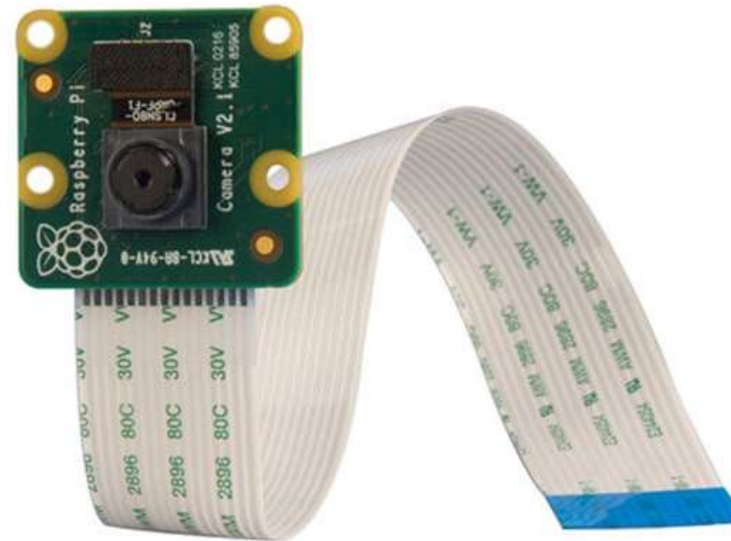
- Gyro
- Accelerometer
- Magnetometer (kompass)
- 5V-kompatibel
- I2C-interface



Sensor 11 : Kamera

Till Raspberry Pi

- 8 megapixel
- 80° resp 160° synfält
- Fixed focus
- Klarar 1080p, 720p60, VGA90
- Avsedd för Raspberry Pi



Sensor 12 : RP-Lidar

Light Detection And Ranging

70

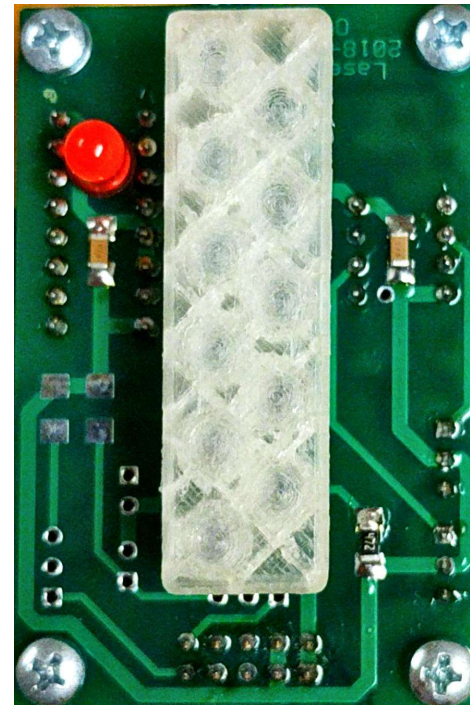
- 360° Laser Range Scanner
- 4000 samples/s
- Rotation speed: 600RPM
- Resolution: 0.9°
- Range: 6m
- Price: \$450!!



Sensor 13 : Laser-IR-detektor

71

- SR-vippa som aktiveras när laserstråle träffar "vit yta"
- SR-vippa återställs med yttre signal
- Kan bestyckas med IR-mottagare



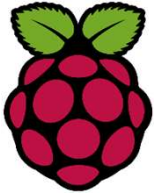
Raspberry Pi

73

- 
- A Raspberry Pi 4 Model B V1.4 single-board computer. It features a green PCB with a black Broadcom BCM2711 SoC. Key components visible include a USB Type-C port, a 3.5mm audio jack, an Ethernet port (labeled EDAC), and four USB-A ports. The board has a 40-pin GPIO header on the top edge and a micro-HDMI port on the left. The Raspberry Pi logo and various certification marks are printed on the board.

Raspberry Pi (raspberrypi.org)

74



"The good ... and the bad"

PLUS

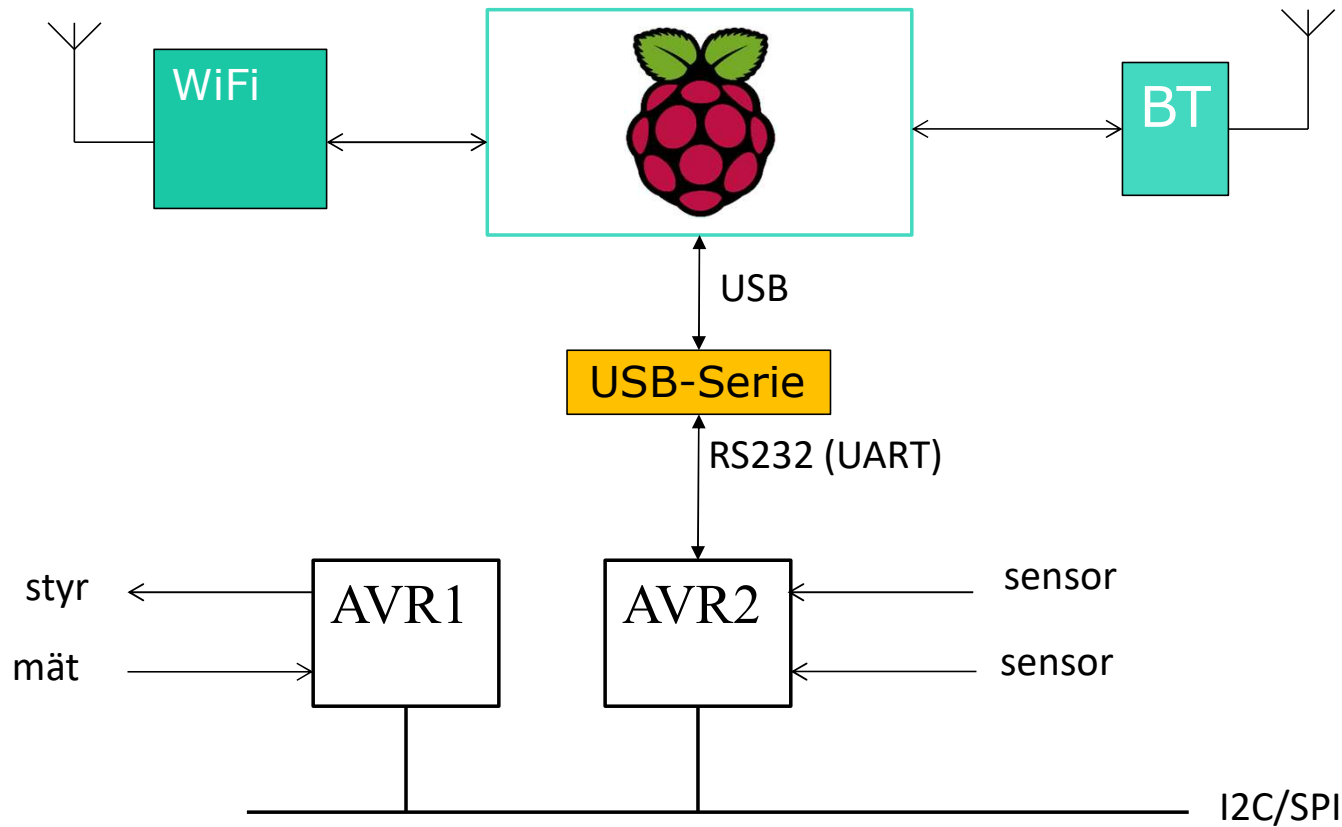
- Ett operativsystem
 - Linux (Raspian, Ubuntu)
- Stort arbetsminne, 1GB
- Kraftfull processor
- Många olika portar
 - USB, HDMI, ETH
 - Audio
- I/O-portar (3.3V)

MINUS

- Ej 5V-kompatibel
- Skrymmande storlek (relativt AVR)
- Krävande att använda

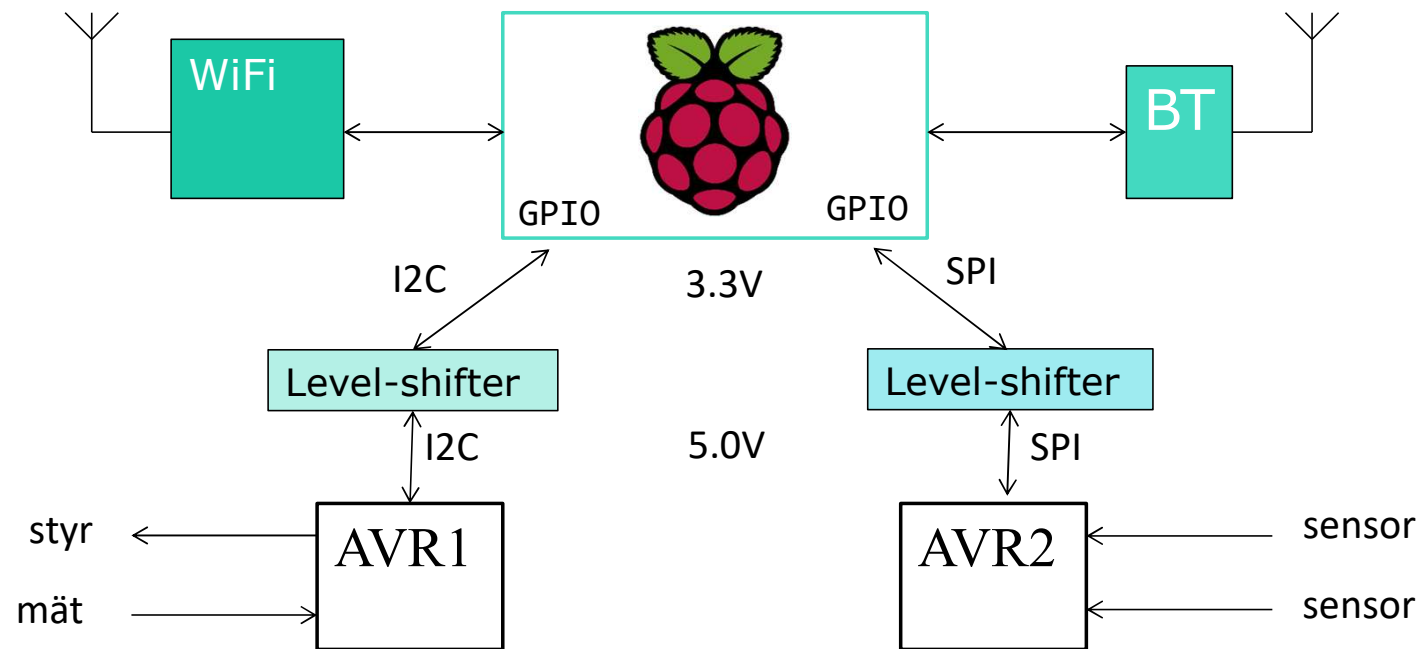
Kopplingsförslag 1

75



Kopplingsförslag 2

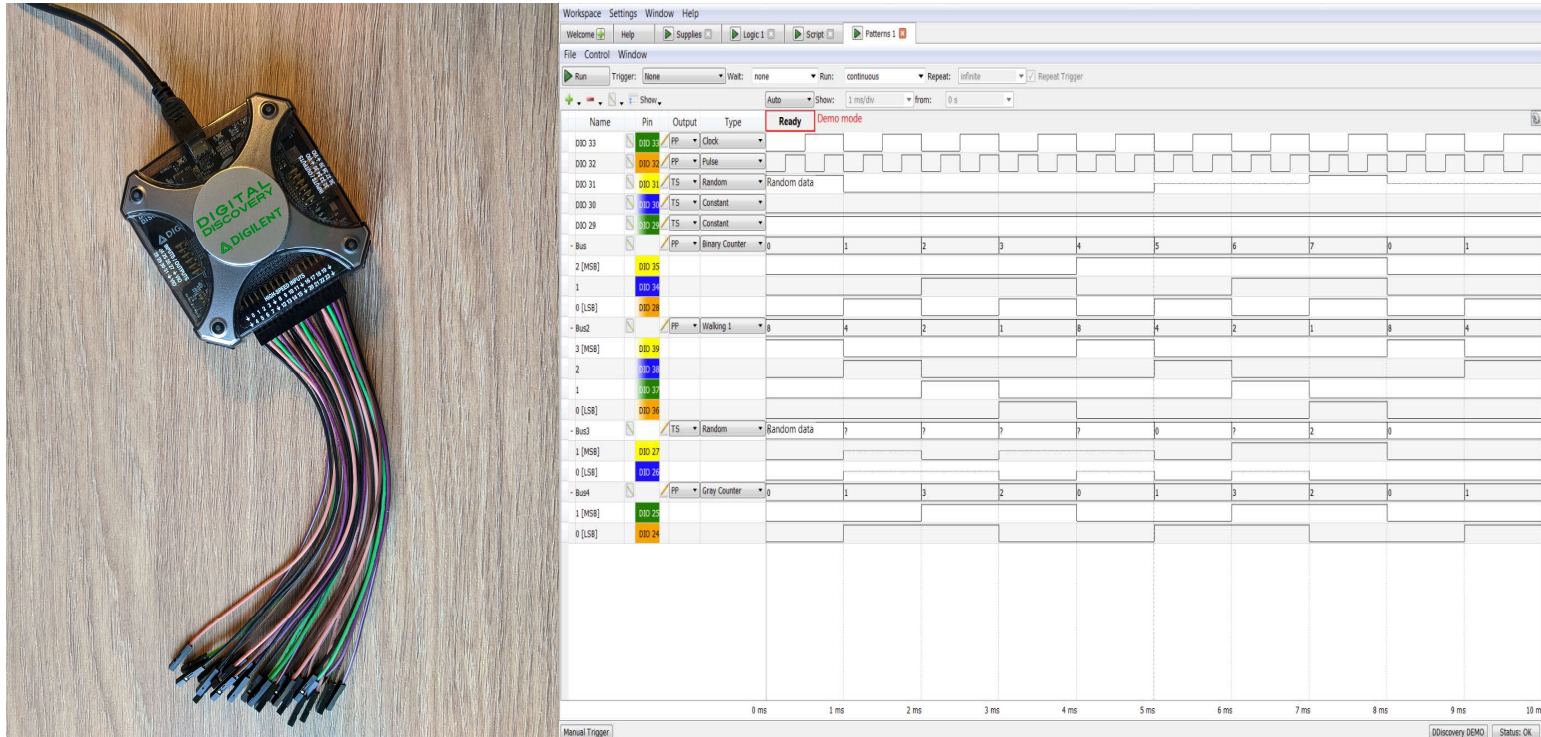
76



Lab1

Lab1 : Mät-lab Introduktion till Digital Discovery

80



"Zu messen ist zu wissen"



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

Labbar : Frågor?

85

Använd kanalen LabbChat i Teams.

Syftet är att ställa frågor där flera kan se och svara och ge vägledning.

Skickar ni mail är det bara en som svarar och bara den som frågat som får svaret. Flera kan ha samma fundering/fråga.

The End

86

Vad göra nu?

- Fundera och planera

- Kolla datablad

vanheden.isy.liu.se

- Tag kontakt och diskutera
eventuella funderingar
med handledare
eller beställare.

27/9 : Designföreläsning

Anders Nilsson

www.liu.se