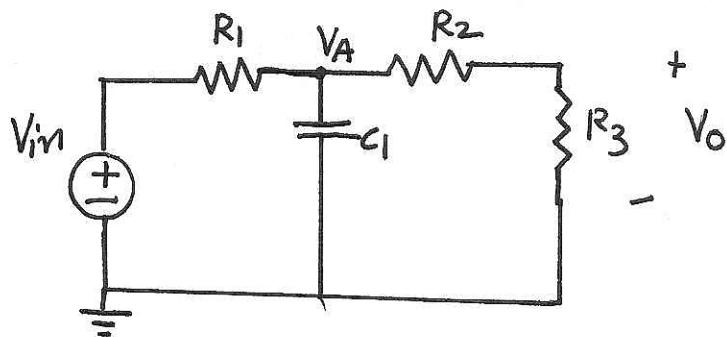


# FRA KREDSLOB TIL DIFF. LIGNING

①



$$\text{Knodepunktsligning: } \frac{V_A - V_{in}}{R_1} + \frac{V_A - V_o}{R_2} + \frac{V_A}{Z_{C1}} = 0$$

$$\frac{V_o - V_A}{R_2} + \frac{V_o}{R_3} = 0$$

$$\text{Benyt kapacitors impedans: } Z_{C1} = \frac{1}{j\omega C_1}$$

$$\text{sys: } \left\{ \frac{V_A - V_{in}}{R_1} + \frac{V_A - V_o}{R_2} + V_A \cdot j\omega C_1 = 0, \quad \frac{V_o - V_A}{R_2} + \frac{V_o}{R_3} = 0 \right\}$$

$$\text{Solve(sys, } \{V_A, V_o\}) \Rightarrow V_o = V_{in} \frac{\frac{R_3}{(C_1 R_1 R_2 + C_1 R_1 R_3) j\omega + R_1 + R_2 + R_3}}$$

$$V_o \left( [C_1 R_1 R_2 + C_1 R_1 R_3] j\omega + R_1 + R_2 + R_3 \right) = V_{in} \cdot R_3 \Leftrightarrow (V_o j\omega \equiv V_o)$$

$$V_o (C_1 R_1 R_2 + C_1 R_1 R_3) + V_o (R_1 + R_2 + R_3) = V_{in} \cdot R_3 \Leftrightarrow \text{standard form}$$

$$V_o + \frac{R_1 + R_2 + R_3}{(C_1 R_1 R_2 + C_1 R_1 R_3)} V_o = V_{in} \cdot \frac{R_3}{(C_1 R_1 R_2 + C_1 R_1 R_3)}$$

Differentialligningen for kredsløbet er fundet på standardform.

# ANDENORDENS DIFF. LIGNINGER - LØSNING

(2)

Generel form:  $\ddot{y}(t) + 2\zeta\omega_0\dot{y}(t) + \omega_0^2 y(t) = x(t)$

$\zeta$ : Dæmpningsfaktoren

$\omega_0$ : Udæmpet resonansfrekvens

3 slags løsninger

$\zeta > 1 \Rightarrow$  overdæmpet system (reelle og forskellige rødder)

$\zeta = 1 \Rightarrow$  kritisk dæmpet system (dobbeltrod)

$\zeta < 1 \Rightarrow$  underdæmpet system (kompleks konjugerede rødder)

$$\text{Overdæmpet: } y(t) = c_1 e^{+\lambda_1 t} + c_2 e^{+\lambda_2 t}$$

$$\text{Kritisk dæmpet: } y(t) = (c_1 + c_2 t)^{+\lambda t}$$

$$\text{Underdæmpet: } y(t) = (c_1 \cos(\omega t) + c_2 \sin(\omega t)) e^{\alpha t} \quad (\text{oscillationer})$$

Eksempel

$$\ddot{v}_o(t) + 4\dot{v}_o(t) + 25v_o(t) = 0, \quad v_o(0) = 1, \quad \dot{v}_o(0) = 1$$

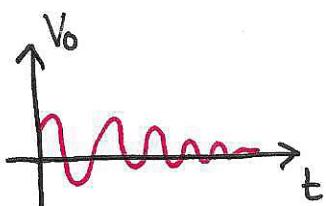
$$\omega_0 = \sqrt{25} = 5 \quad , \quad 2\zeta\omega_0 = 4 \iff$$

$$\zeta = \frac{4}{2\omega_0} = \frac{4}{10}$$

Systemet er underdæmpet.

$$\text{dsolve}\left(\left\{ \ddot{v}_o(t) + 4\dot{v}_o(t) + 25v_o(t) = 0, v_o(0) = 1, \dot{v}_o(0) = 1 \right\}\right)$$

$$v_o(t) = \frac{e^{-2t}}{7} \cdot (7 \cos(\sqrt{21}t) + \sqrt{21} \sin(\sqrt{21}t))$$



# IMPEDANS OG OVERFØRINGSFUNKTION

(3)

Kapacitor:  $Z_C = \frac{1}{j\omega C}$

Spole:  $Z_L = j\omega L$

Hvis en sinusbølge  $e^{j\omega t}$  er input i et lineært system, vil output være en skaleret og faseforskydnede sinusbølge.

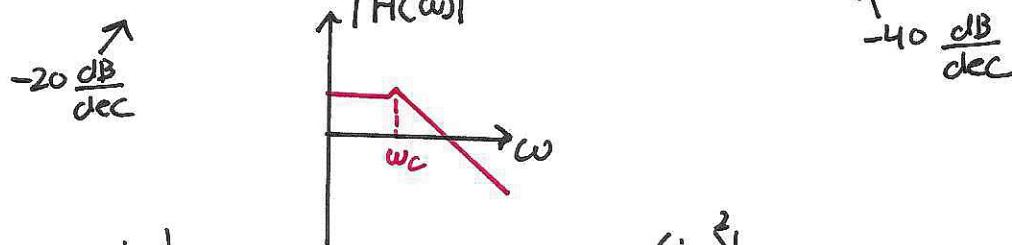
Overføringsfunktionen  $H(\omega)$  fortæller alt.

$|H(\omega)|$  angiver skalingen

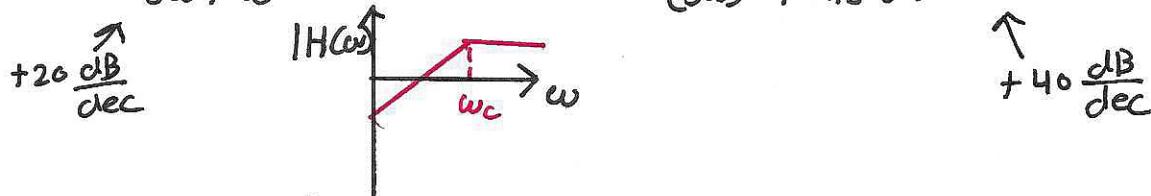
$\angle H(\omega)$  angiver faseforskydningen

$$H(\omega) = \frac{\text{output}}{\text{Input}}$$

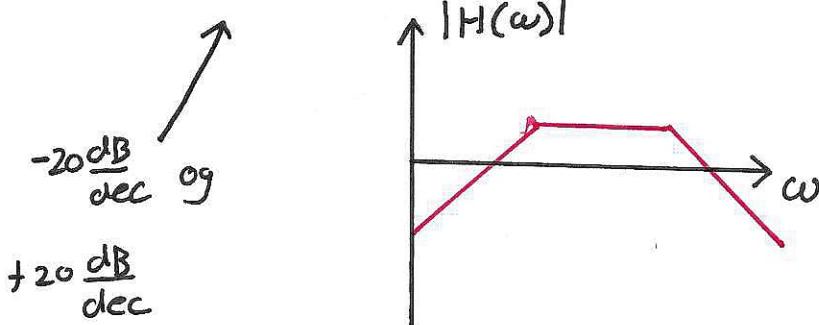
Lavpasfilter:  $H(\omega) = \frac{b_0}{j\omega + a_0}$  eller  $H(\omega) = \frac{b_0}{(j\omega)^2 + j\omega a_1 + a_0}$



Højpasfilter:  $H(\omega) = \frac{j\omega b_1}{j\omega + a_0}$  eller  $H(\omega) = \frac{(j\omega)^2 b_2}{(j\omega)^2 + a_1 j\omega + a_0}$



Båndpasfilter:  $H(\omega) = \frac{j\omega b_1}{(j\omega)^2 + j\omega a_1 + a_0}$

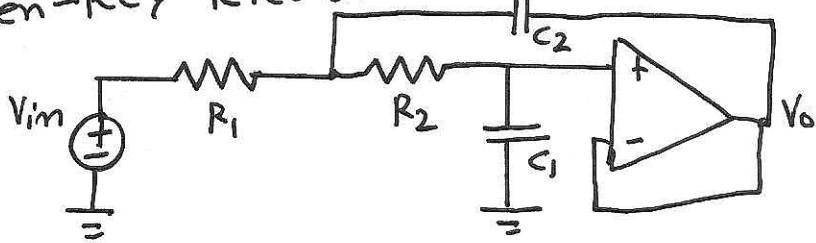


# FILTERDESIGN OG SKALERING

(4)

2. ordens Butterworth lavpassfilter:  $\frac{1}{(\zeta\omega)^2 + \sqrt{2}\zeta\omega + 1}$

Har knækfrekvens i  $\frac{1}{\text{rad}} = \frac{1}{\sqrt{2}}$  og implementeres med et Sallen-key kredsløb.



$$R_1 = R_2 = 0.22 \Omega$$

$$C_1 = 3.3 \text{ F}, C_2 = 6.8 \text{ F}$$

Frekvensskalerer kun kondensatorer for at flytte knækfrekvensen.

$$k_f = \frac{2\pi f_c, \text{new}}{\omega_c, \text{old}} = \frac{2\pi \cdot 1000 \text{ Hz}}{1 \frac{\text{rad}}{\text{s}}} = 6283$$

$$C'_1 = \frac{C_1}{k_f} = \frac{3.3 \text{ F}}{6283} = 0.525 \text{ mF}$$

$$C'_2 = \frac{C_2}{k_f} = \frac{6.8 \text{ F}}{6283} = 1.08 \text{ mF}$$

Impedansskalér alle komponenter.

$$k_2 = \frac{c'_1}{c_1, \text{ønsket}} = \frac{0.525 \text{ mF}}{33 \text{ nF}} = 15909$$

↑  
Tildeligt

$$c''_1 = 33 \text{ nF}$$

$$c''_2 = \frac{c'_2}{k_2} = \frac{1.08 \text{ mF}}{15909} = 68 \text{ nF}$$

$$R'_1 = R_1 k_2 = 0.22 \cdot 15909 = 22 \text{ k}\Omega$$

$$R'_2 = R_2 k_2 = 0.22 \cdot 15909 = 22 \text{ k}\Omega$$

# SENSITIVITET

(5)

Sensitivitetsfunktionen:

$$S_x^y = \frac{x}{y} \cdot \frac{\partial y}{\partial x}$$

$S_x^y = 5$  betyder, at hvis  $x$  ændrer sig + 1% ændrer  $y$  sig + 5%.

For Sallen-key filtre:

Unity gain ( $K = DC$ -forstørrelse = 1) for insensitive filter

Lavpasfilter:  $R_1 = R_2$  for insensitive filter

Højpasfilter:  $C_1 = C_2$  for insensitive filter.

$$\begin{aligned} \bullet \frac{C_2}{C_1} &= \frac{4a_0}{a_1} \\ \bullet R_1 = R_2 &= \frac{a_1}{2a_0 C_1} \end{aligned} \quad \left. \begin{array}{l} \\ \end{array} \right\} \text{Lavpas}$$

$$\begin{aligned} \bullet R_1 C &= \frac{2}{a_1} \\ \bullet R_2 &= \frac{a_1}{2a_0 C} \end{aligned} \quad \left. \begin{array}{l} \\ \end{array} \right\} \text{Højpas}$$

Man kan transformere et lavpasfilter til højpasfilter med variabelsubstitution:

$$j\omega \leftarrow \frac{1}{j\omega}$$

dvs. de steder i den oprindelige overføringsfunktion står  $\frac{1}{j\omega}$  står der nu  $j\omega$ .

## EKG-SIGNALS - STØJKILDER

(6)

- 50 Hz common mode støj: Stammer fra lysnettet og kobles til kroppen via strømkapaciteter.  
Findes overalt i kroppen.
- Halvcelle-potentieler: Et DC (0 Hz) signal der stammer fra elektrodens interface med huden.
- EMG-signaler: Muskelsignaler (50-200 Hz)
- Baseline drift: Et lavfrekvent signal der opstår på grund af bevægelsesartefakter.
- EKG-signal: Differentielt signal (0.05-150 Hz) med  $-25 \text{ mV} - 25 \text{ mV}$  amplituder.  
Findes som projektionen af hjertevektoren ned på leadvektoren.

Patient eller EKG-forstærker må ikke være forbundet bygningsjord.  
Common-mode signal undertrykkes af differential/instrumenteringsforstærker.  
Halvcelle-potentieler undertrykkes af passivt højpasfilter inden indgangen på instrumenteringsforstærkeren.  
Tilstedeværelsen af udæmpede halvcelle-potentieler begrænser instrumenteringsforstærkerens differentielle gain.

Instrumenteringsforstærker har høj indgangsmodstand.

# GAIN OG OFFSET & CMRR

7

COMMON MODE REJECTION RATIO (CMRR):

Et udtryk for hvor meget en differensforstørker kan undertrykke common mode signaler.

$$CMRR = \frac{|A_d|}{|A_{cm}|} = A_{d,dB} - A_{cm,dB}$$

## Eksempel

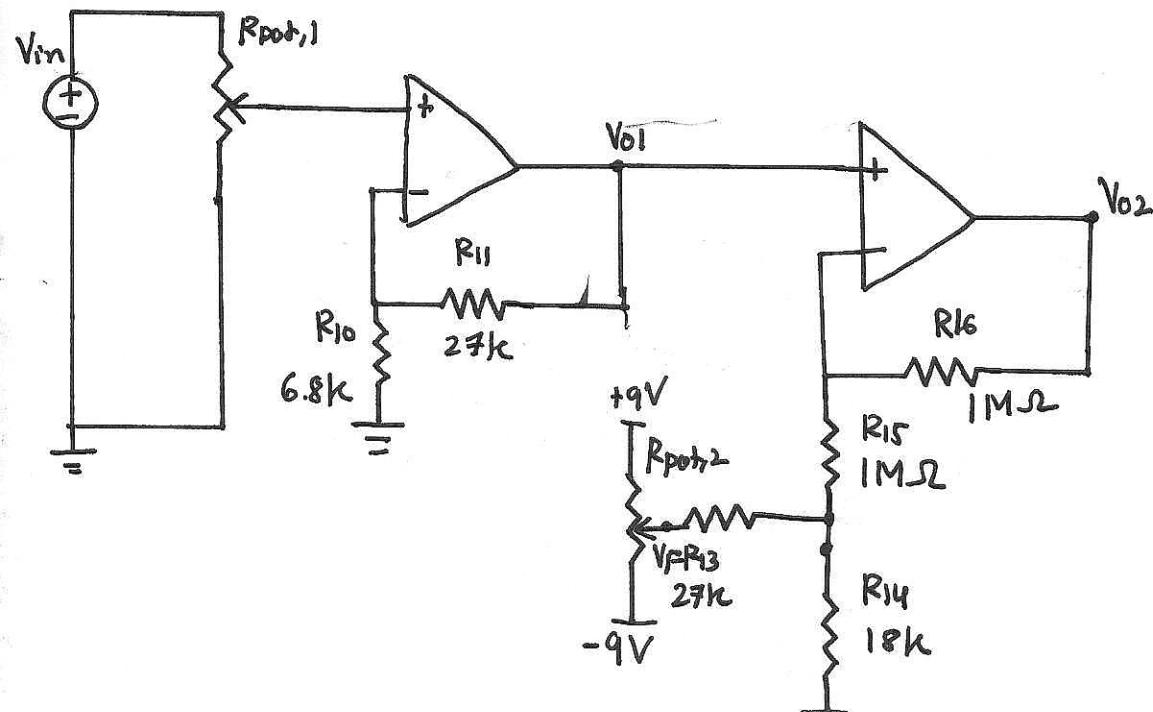
Input:  $V_{diff} = 0 - 5 \text{ mV}$     $V_{cm} = 2 \text{ V}$

Output:  $V_{diff} = 0 - 5 \text{ V}$     $V_{cm} = 2 \text{ mV}$

Vi ønsker en forstørkning  $|A_d| = 1000 = 60 \text{ dB}$  og en cm-dæmpning  $|A_{cm}| = \frac{1}{1000} = -60 \text{ dB}$

Hvor meget CMRR skal vores differentialforstørker mindst have?

$$CMRR = A_{d,dB} - A_{cm,dB} = 60 \text{ dB} - (-60 \text{ dB}) = \underline{120 \text{ dB}}$$



Kredsløbsligning:  $V_{o2} = 2 \times \left(1 + \frac{R_{11}}{R_{10}}\right) V_{in} - \frac{R_{14}}{R_{13} + R_{14}} \cdot V_F$

Gain: 0 til 9.94V  
Offset: -3.6V til 3.6V

# HARDWARE OG SOFTWARE INTERRUPT

(8)

Interrupt: Anmodning til processoren om at afbryde dens nuværende opgave og foretage en anden (kort) opgave.

Interrupt service routine (ISR): Det kode som udføres når interruptet sker.

Status register: SREG 

I	T	H	S	V	N	Z	C
---	---	---	---	---	---	---	---

  
↑  
Global interrupt enable

## Hardware interrupts:

Kan ske på Arduinosens pin 2 eller pin 3.

Interrupt trigger event: LOW RISING CHANGE FALLING

EIMSK: 

-	-	-	-	-	-	INT1	INT0
---	---	---	---	---	---	------	------

  
↑  
PIN3  
interrupt enable  
↑  
PIN2  
interrupt enable

(External interrupt control register A)

swap names

EICRA: 

-	-	-	-	ISC11	ISC10	ISC01	ISC00
---	---	---	---	-------	-------	-------	-------

  
(External interrupt mask register)

Styrer trigger event for PIN3 og PIN2.

EIFR: 

-	-	-	-	-	-	INTF1	INTFO
---	---	---	---	---	---	-------	-------

 (External interrupt flag register)

Register med interrupt flag. Et flag heises når et interrupt trigger event sker. Flaget sænkes når interrupt service routinen er færdig.

ISR udføres kun hvis det tilsvarende bit i EIMSK er 1.

Fx: ISR(INT0\_vect) kaldes når INTFO heises og hvis INT0 i EIMSK er sat til 1.

# SOFTWARE INTERRUPT

(9)

PIN change interrupt: En vilkårlig ændring af spænding på en PIN tilhørende en bank med pin change interrupt enabled forårsager interrupt.

PCICR: 

-	--	--	--	--	PCIE2	PCIE1	PCIE0
---	----	----	----	----	-------	-------	-------

 (Pin change interrupt control register)

Styrer hvilke pins der kan ske et pin change interrupt.

PCIE0: D8-D13 + 2 crystal pins

PCIE1: A0-A5 + reset

PCIE2: D0-D7

PCIFR: 

-	--	--	--	--	PCIF2	PCIF1	PCIFO
---	----	----	----	----	-------	-------	-------

 (Pin change interrupt flag register)

Register med interrupt flag. Et flag hejses når et pin change interrupt sker, på en pin i en enabled bank, og pinnen selv er enabled.

Man kan kun se hvilken bank pinnen tilhører når der sker et interrupt.

PCMSk2 : 

PCINT23	PCINT22	PCINT21	PCINT20	PCINT19	PCINT18	PCINT17	PCINT16
---------	---------	---------	---------	---------	---------	---------	---------

Pin change mask register 2, for pins D0-D7.

Fx. hvis PCINT23 er sat til 1, og PCIE2 er sat til 1, vil interrupt service rutinen ISR(PCINT2-vect) kaldes, når flaget PCIF2 hejses.

PCIF2 hejses når der sker en vilkårlig ændring i spænding (pin change) på ~~en~~ pin i bank 2 (D0-D7), svarende til PCINT23 (D7).

Flaget sænkes når ISR er fuldført.

Flaget kan også sænkes ved at sætte bittet højt (dvs.

$PCIFR |= (1 << PCIF2);$

# TIMER TRIGGERED INTERRUPT

(10)

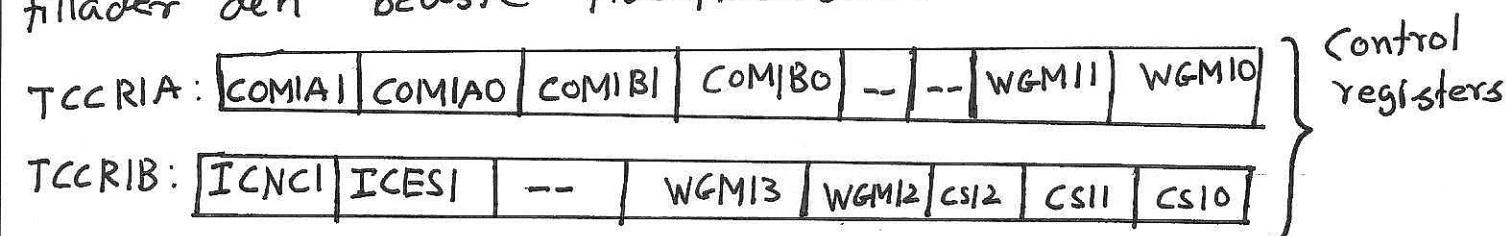
Timer triggered interrupt: Interrupt forårsaget af en counter når en specificeret værdi.

Timer0: 8-bit

Timer1: 16-bit

Timer2: 8-bit

Normalt benyttes timer1, på grund af mangden af bits tillader den bedste tidsopløsning.



WGM13 - WGM10 styrer timerens mode.

CS12 - CS10 styrer prescaleren, (mindre = bedre tidsopløsning)

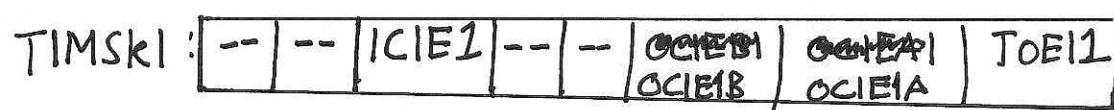
## Overflow mode

Counteren TCNT1 gives en startværdi og teller op til  $2^{16} - 1 = 65535$ . Når TCNT1 når 65535 sker et interrupt, og TCNT1 sættes til startværdi. Ønskes interrupt med periodetiden Tdur bruges formlen

$$TCNT1 = 2^{16} - \frac{16 \text{ MHz} \cdot T_{\text{dur}}}{\text{Nprescaler}}$$

## CTC mode

Counteren TCNT1 starter i 0 og teller op til en værdi i OCRIA. Når TCNT1 teller til OCRIA sker et interrupt og TCNT1 sættes til 0 igen.  $OCRIA = \frac{16 \text{ MHz} \cdot T_{\text{dur}}}{\text{Nprescaler}} - 1$



Timer1 mask register



Timer1 interrupt register

OCF1A hejses når TCNT1 = OCRIA. Hvis OCIE1A er sat `HIGH` udføres ISR(Timer1-COMPBA-vect), og OCF1A sættes lavt når rutinen er fuldført. Alternativt kan OCF1A sænkes med  $\text{TIFR1.IOCF1A} = (\text{ICF1A})$ ;

# SAMPLING, KVANTISERING & ALIASING

11

Sampling  $\approx$  opsamle eller gemme en værdi.

Kvantisering  $\approx$  Afrundning af værdi til et niveau.

Sampling sker før kvantisering.

En ADC (analog-digital konverter) både sampler og kvantiserer.

Arduino: 10-bit ADC.

Tommelfingerregel: Dynamic range per bit  $\rightarrow 6 \frac{\text{dB}}{\text{bit}}$

10-bit ADC har DR = 60 dB.

Med en referencespænding på 5 V er kvantiseringsniveauerne:

$$\Delta V_{\text{LSB}} = \frac{5 \text{ V}}{2^{10}} = \frac{5 \text{ V}}{1024} = 4.88 \text{ mV}$$

$$0000 \ 0000 \ 01 \approx 4.88 \text{ mV}$$

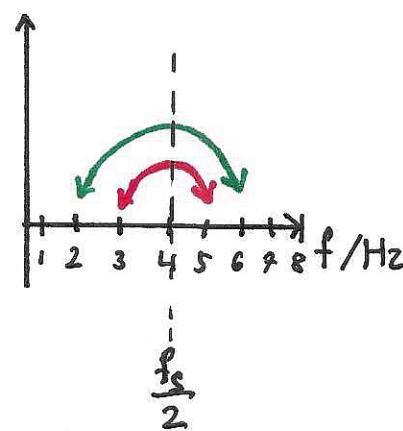
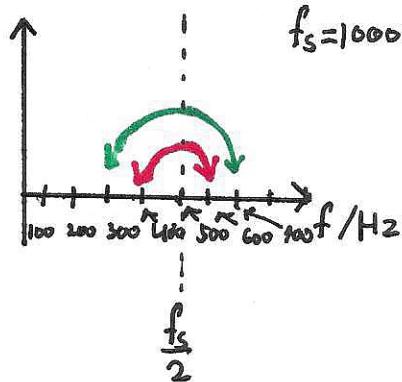
$$0000 \ 0000 \ 10 \approx 9.76 \text{ mV}$$

:

Ved sampling med en given samplingfrekvens, vil høje frekvenser have aliaser i lavere frekvenser.

$f_s = 1000 \text{ Hz} \rightarrow 300 \text{ Hz}$  har alias i 700 Hz

$f_s = 8 \text{ Hz} \rightarrow 2 \text{ Hz}$  har alias i 6 Hz



Alias kan ikke undgås, men effekten kan gøres ubetydelig hvis signalerne ved  $\frac{f_s}{2}$  har en total dæmpning på DR.

For en 10-bit ADC kræver det  $|H(2\pi \cdot \frac{f}{f_s})| = -60 \text{ dB}$

## ARDUINOENS ADC

(12)

Arduino kan programmeres til at foretage en analog-digital konvertering med analogRead eller en trigger source (som timer)

ADMUX: [REFSI | REFSO | APLAR | -- | MUX3 | MUX2 | MUX1 | MUX0]

REFSI og REFSO styrer reference spændingen ( $AV_{cc}$ ) er 5V.  
APLAR styrer om resultatet af konverteringen right/left justeres.

MUX3-MUX0 styrer hvilken PIN konverteringen sker.

ADCSRA: [ADEN | ADSC | ADATE | ADIF | ADIE | ADPS2 | ADPS1 | ADPS0]

ADEN: Enable ADC

ADSC: ADC start konvertering

ADATE: ADC auto trigger enable, så ADC kan starte konvertering på et trigger signal.

ADIF: ADC interrupt flag der hejses, når en konvertering er komplet.

ADIE: ADC interrupt enable. Hvis dette bit er sat højt vil

ISR(ADC-vect) kaldes, når ADIF hejses. Når service-rutinen er færdig sænkes ADIF.

ADPS2-ADPS0: ADC prescaler, sættes efter hvilken samplingfrekvens der ønskes. <sup>Høj</sup> prescaler  $\Rightarrow$  mest præcis konvertering.

ADCSR B: [-- | ACME | -- | -- | -- | ADTS2 | ADTS1 | ADTS0]

ACME: Analog comparator multiplexer enable.

ADTS2-ADTS0: ADC trigger source. Timer1 kan trigger ADC'en med overflow interrupt og timer compare match B interrupt.

ADTS2	ADTS1	ADTS0	
0	0	0	Free running
0	0	1	Analog comparator
0	1	0	Ext. interrupt
0	1	1	Timer0 compare match A
1	0	1	Timer0 overflow
1	0	0	Timer1 compare match B
1	1	0	Timer1 overflow
1	1	1	Timer1 capture event

## SPI OG SD-KORT

(13)

SPI-kontrollen er en fuld-duplex kommunikationsprotokol der bruger 4 signaler:

SCLK: System clock (synkronisering af slave & master)

MOSI & MISO: Datalinjer der tillader samtidig kommunikation af master og slave.

SS/CS: Chip-select pin, til at vælge hvilken slave der er aktiv.

Både slave og master har et 8-bit data-register.

Når 8 bits (1 byte) er blevet sendt/modtaget hejses SPIF.

~~SPCR~~

SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPRI	SRRO	SPCR
------	-----	------	------	------	------	------	------	------

SPIE: SPI enable interrupt

SPE: SPI enable: Tillader brug af SPI-operationer

DORD: Data order: Bestemmer om LSB (DORD=1) sendes først.

MSTR: Master/slave select: Bestemmer om enheden er master (MSTR=1)

CPOL: Clock polarity: Bestemmer om klokkenens ledende kant er rising (CPOL=0) eller falling (CPOL=1).

CPHA: Clock phase: Bestemmer om data opsamples på klokkenens ledende kant (CPHA=0) eller bagkanten

SPRI-SRRO: Bestemmer prescaling af klokken.

SPSR: 

SPIF	WCOL	--	--	--	--	SPI2X
------	------	----	----	----	----	-------

SPIF: Interrupt flag der hejses når 8 bits er sendt/modtaget fra slaven/masteren. Hvis SPIE er sat high vil ISR(SPI\_STC\_vect) kaldes. Flaget sænkes når rutinen er fuldført.

WCOL: Write collision flag:

SPI2X: Double SPI speed bit: Hvis dette bit er sat bliver prescaler'en halvt så stor.

# SD-KORT OG BUFFERING

14

SD-kortet benytter SPI-protokollen.

Arduino-kommandoer:

SD.begin(csPin);

File myFile;

myFile = SD.open("fileName", FILE\_WRITE); skrivetilladelse

Både læsning og skrivning til SD-kortet er Little Endian, dvs. den mindst signifikante byte sendes først.

0011 1100 1100 0101  
MSB      LSB (sendes først i little endian)

I EKG-forstærker loggingkoden benyttes teknikken "cirkulær buffering".

Cirkulær Buffering:

- Definér to arrays: adcBlock0, adcBlock1
- Start med at fyldе adcBlock0 op med samples.
- Når adcBlock0 er fuld fyldes de næste samples ind i adcBlock1, samtidig med at adcBlock0 overføres til SD-kort.
- Når adcBlock1 er fuld overføres indholdet til SD-kortet og de gamle samples i adcBlock0 overskrives med nye samples.

Skal man skrive tekst til en fil på SD-kortet: SD.print();

Skal man skrive tal (hurtigt): SD.write();

Tekstfiler kan åbnes og forstås.

Binære filer .bin kan ikke fortolkkes af mennesker.

## POINTERS OG MOVING AVERAGE

(15)

Pointer: En variabel der indeholder adresse i hukommelsen for en anden variabel.

int a = 10;

int \*p = &a;

↑  
ampersand (&)  
memory address

int \*p = &a;

Moving average filter: En metode til at finde (tage gennemsnit) af et antal samples ad gangen.

$$MA(n+1) = MA(n) + \frac{x(n+1)}{M} - \frac{x(n-M)}{M}$$

Rekursivt filter.

M = antallet af punkter der middles af gangen.

Kumulativ moving average bruges de første M samples:

$$CMA(n+1) = \frac{x(n+1) + n \cdot CMA(n)}{n+1}$$

Glidende gennemsnit på dansk.

God til at fjerne deterministisk støj, som fx common mode støj med 50 Hz fra lysnettet.

QED.