

IN1080 Mekanikk

Introduksjon 2022

Oversikt Introduksjon

- Hvem er vi?
- Oppbygning av kurset
- Pensum og lærestoff
- Studieteknikk
- Læringsplattform-Canvas
- Lab og adgang
- Hva er mekatronikk?
- Fagdisipliner
- Forkunnskaper

Hvem er vi?

▪ Forelesere:



Sigbjørn Næss
Førsteamanuensis II



Yngve Hafting
Universitetslektor

Gruppelærere/Labveiledere:

- Afsah Asif Rashid
- Christoffer Lingjærde
- Geir Yngve Paulsen
- Sondre Røvik Kippenes

Andre

- Vegard Søyseth (Senioringeniør) Lab, tekniske fasiliteter
- Mats Høvin (Førsteamanuensis) Lab, eksamen

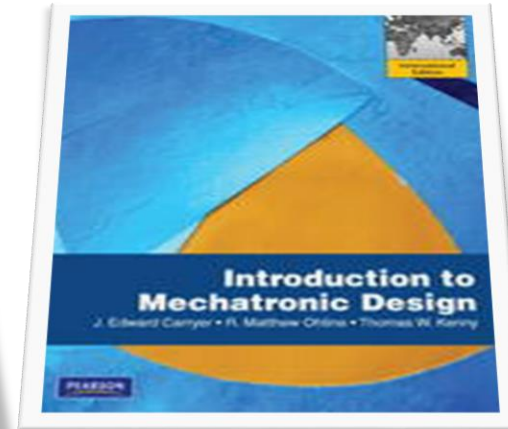
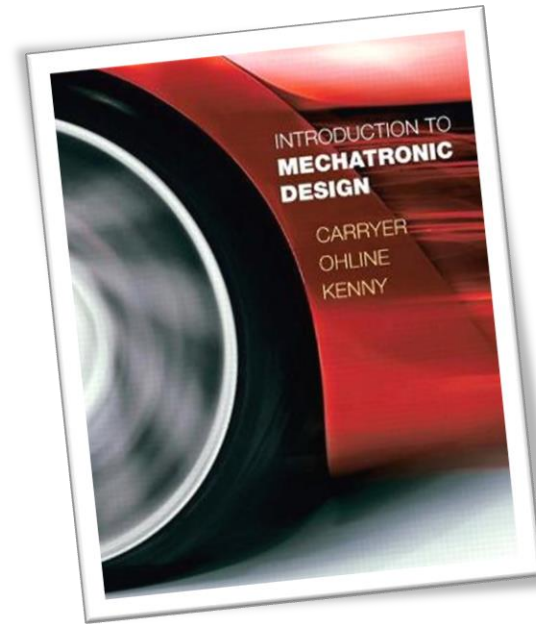
Oppbygning av kurset

- Forelesninger (Zoom, Caml,C)
 - Tirsdag
 - Onsdag
 - (Regneøvelse = forelesning)
 - Onsdager fom. 26.1
 - Virtuelt i januar (Zoom)
 - Fysisk Februar
 - Caml på tirsdager
 - Seminarrom C på onsdager
 - (Mulig endring av lokale)
- Lab
 - Grupper starter opp fom. 31.1.

- Kommunikasjonskanaler
 - Vortex (kurssiden)
 - Beskjeder til alle
 - Canvas
 - Innlevering og tilbakemeldinger (obliger)
 - Husk å gi beskjed ved avvik!
 - «astro-discourse»
<https://astro-discourse.uio.no/c/in1080/>
 - Kursrelaterte spørsmål
 - Ikke personlige spørsmål
 - (utsettelse osv)
 - Matternost?
 - Epost
 - Personlige spørsmål
 - Kontakt din labveileder først

Læremidler

- Lærebok:
 - **Introduction to mechatronic design**
 - Carrier, Ohline, Kenny (COK)
 - ISBN: 9780131433564; 0-13-143356-3
 - Prentice Hall/Pearson
 - Ulike cover, samme innhold
 - Bokhandel vs bibliotek
 - forskjellige sidetall
- **Arduino kit**
 - Kan kjøpes i resepsjonen 1.etg på ifi.
 - Noen ekstra deler vil bli distribuert (via resepsjonen)



Lab

- Starter fom. 31.1.
- Nå i egne rom (se timeplan)
- Jobb med obliger og ukesoppgaver
- Arduino
 - hjemmefra = lov
 - Valg mellom å lage video og å demonstrere for labveileder på laben
 - Merk: Biblioteker kan gjøre «for mye» noen ganger.

Utgangspunkt?

- Poll: www.menti.com <1564 6391>
 - Avstemmingen foregår anonymt

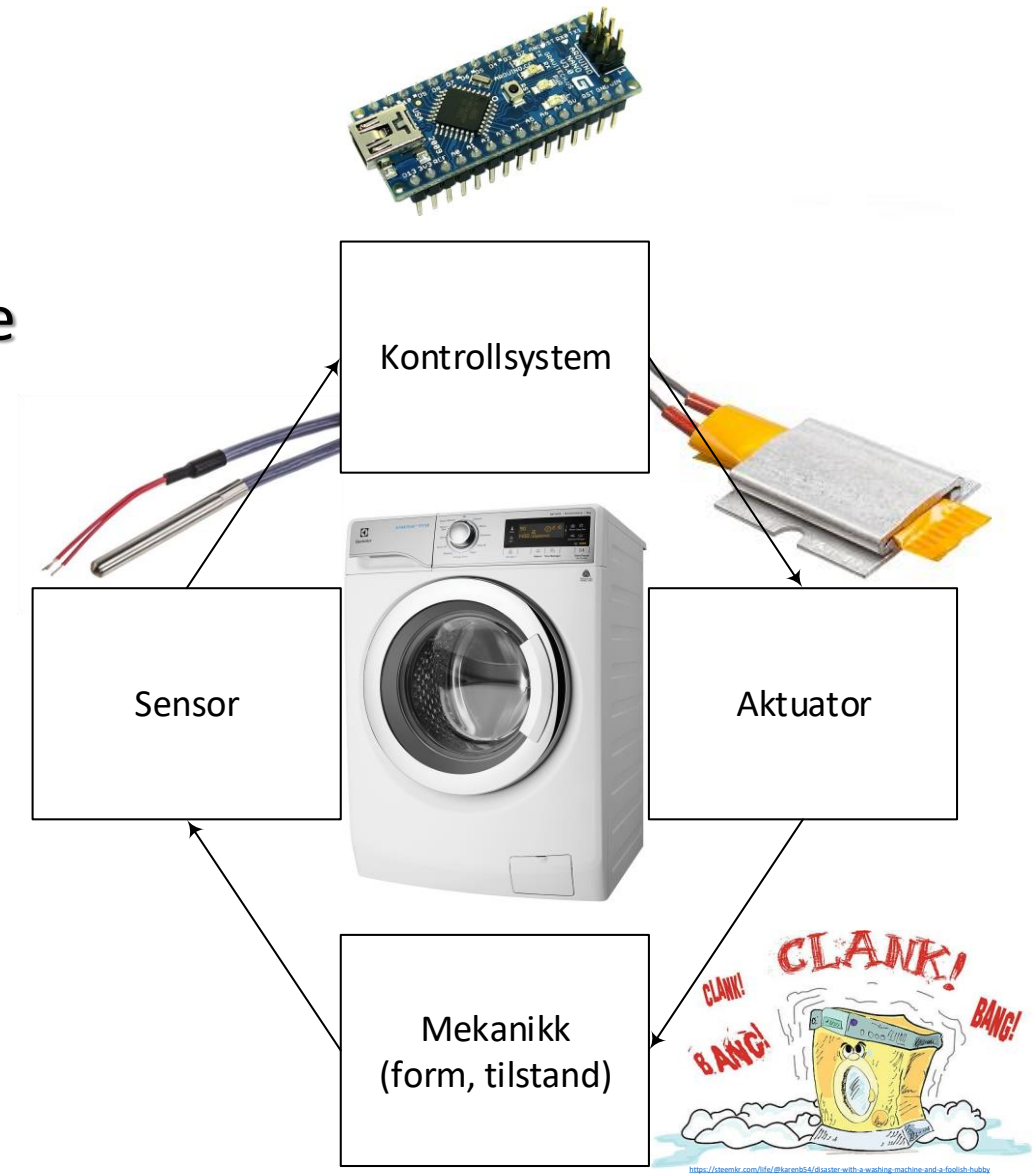
www.mentimeter.com

Forkunnskaper

- R1+R2 fra VGS, samt fordypning i minst ett realfagsemne til
 - *Fysikk er en fordel*
 - Ikke Fysikk? => vurder FYSIKK 1 som støttelitteratur
 - Relevante tema fra Fysikk 1 og 2:
 - Rettlinjet bevegelse, Newtons lover, kraft, arbeid, energi, strøm, spenning, resistans, frekvens, halvledere og transistorer, induksjon, sirkelbevegelse, bevegelsesmengde.
- Anbefalte emner:
 - IN1000 (o.l.)
 - Generell forståelse av hvordan program er bygd opp
 - Noe trening i å programmere
 - IN1020
 - Sekvensiell logikk, flippfloppe
 - Noe trening i å forstå oppbygning av digitale systemer

Hva er Mekatronikk?

- Læren om systemer som inneholder mekaniske komponenter i samspill med elektronikk og programvare.
- Et mekatronisk system består typisk av bevegelige deler i samspill med sensorer som leses elektronisk og et kontrollsystem som styrer aktuatorer som beveger delene.
- For å beherske mekatronikk bør man derfor ha kjennskap til mange fagdisipliner (neste side).



Disipliner

- **Mekanikk**

- Kraft
- Dreiemoment
- Bevegelseslover
- Energi og arbeid

- **Elektronikk**

- Elektrisitet
- Magnetisme
- Halvledere
- Kretsteori

- **Måleteknikk**

- Sensorer
- AD-konvertering
- Forsterkere

- **Informatikk**

- Lavnivå programmering
- Busser

- **Kontrollteori**

- Kontrollsystemer

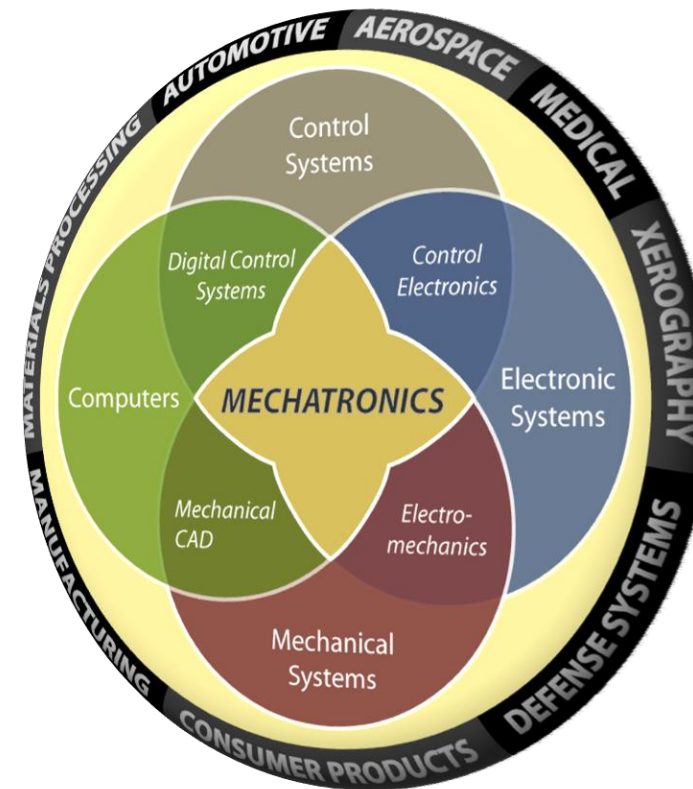


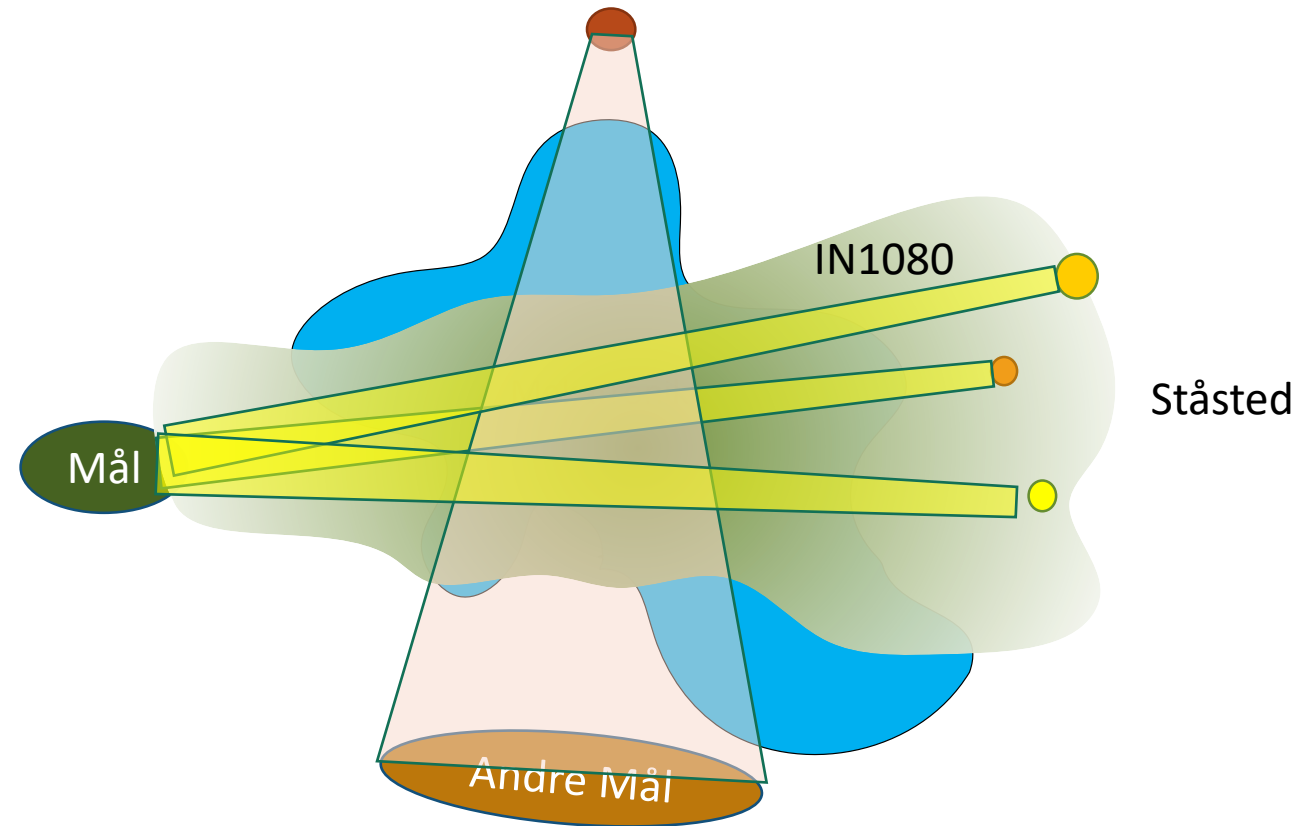
Image by Ahm2307 at English Wikipedia, CC BY 2.5,
<https://commons.wikimedia.org/w/index.php?curid=3267847>

Mål

- *Studieprogram*
 - «Studiet lærer deg hvordan maskiner kan sanse og handle, og hvordan du kan bygge systemer som løser oppgaver.»
- *Kursbeskrivelse*
 - Etter å ha tatt IN1080 kan du:
 - forstå virkemåten til analoge kretser. Aktuelle begreper er: strøm, spenning, motstand, effekt, impedans, likestrøm, vekselstrøm, RCL, MOS, FET, OPamp
 - bruke klassiske analysemetoder basert på Kirchhoff, Thevenin og Nortons teoremer
 - forstå og anvende sensorer, signalkondisjonering og konvertering, samt noen komponent-komponent busser
 - bygge og programmere enkle mekatroniske systemer med mikrokontroller, aktuatorer og sensorer
 - forstå grunnleggende kontrollteori og virkemåte for PIDkontrollere
- Egne mål?

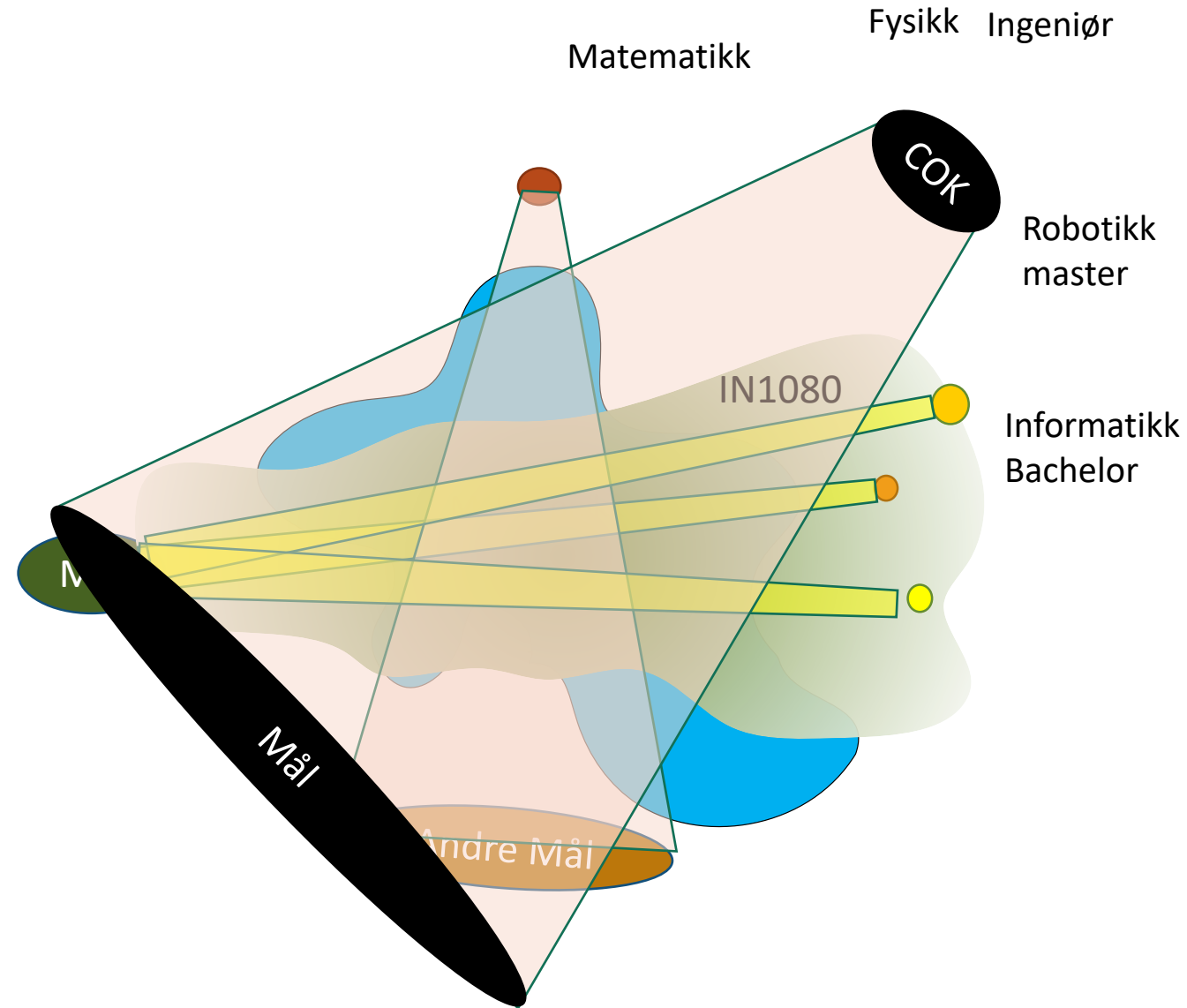
Ulike perspektiv på mekatronikk

- Hvordan kombinere Mekatronikkens egenart med Informatikk?
- Utfordringer
 - Fysisk forståelse vs
 - Systemforståelse vs
 - Anvendelse
 - Kvantitativt vs kvalitativt
- Har vi samme ståsted og mål som andre?



Merk om Læreboka

- Bredere målgruppe enn oss..
- Vi bruker ikke hele
 - Kan være fin å ha hvis man vil gjøre noe praktisk senere... (master?)



IN 1080

Innebygde systemer og mikrokontrollere

Yngve Hafting, 2022

Hvor står vi og hvor går vi...

Kort om emnet

- Grunnleggende analog elektronikk, sensorer og **sensor grensesnitt**, aktuatorer. **Programmering av mekatroniske systemer.**

Hva lærer du?

Etter å ha tatt IN1080 kan du:

- forstå virkemåten til analoge kretser. Aktuelle begreper er: strøm, spenning, motstand, effekt, impedans, likestrøm, vekselstrøm, RCL, MOS, FET, OPamp
- bruke klassiske analysemetoder basert på Kirchhoff, Thevenin og Nortons teoremer
- forstå og anvende sensorer, signalkondisjonering og konvertering, samt noen komponent-komponent busser
- **bygge og programmere enkle mekatroniske systemer med mikrokontroller**, aktuatorer og sensorer
- forstå grunnleggende kontrollteori og virkemåte for PIDkontrollere

Lab

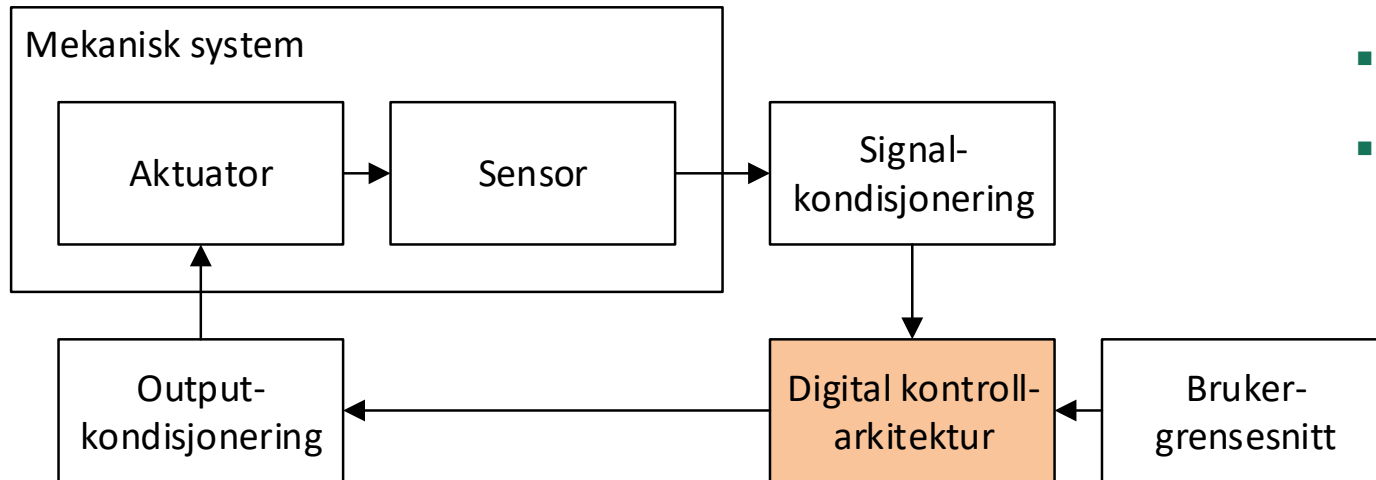
- ...

Forelesning

- Kjenne til hva vi mener med innebygde systemer, mikrokontrollere og mikroprosessorer
 - Fordeler og ulemper med ulike arkitekturer
 - Velge arkitektur ut i fra behov

-

Systemperspektiv og oversikt



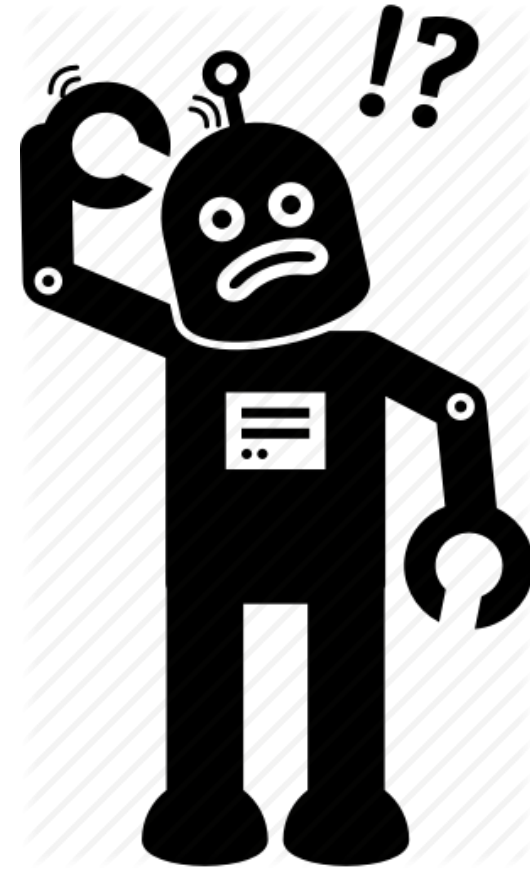
- Innebygde systemer
- Mikrokontrollere
 - ARDUINO
 - Komme igang..

Hvordan styrer vi roboter eller mekatroniske systemer?

- Ulike formål
 - Motorkontroll
 - Sansing
 - Styresystemer
 - Brukergrensesnitt
- Ulike krav
 - Regnekraft
 - Beregninger i sanntid
 - I/O (Input/output)
 - Analog
 - Digital
 - Effektforbruk

I dag...

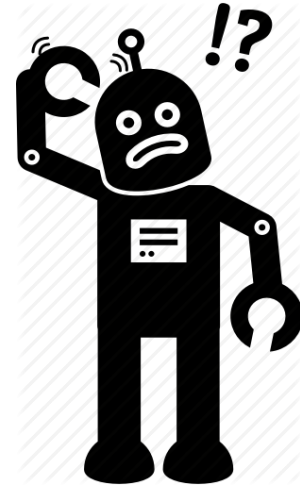
- Rask gjennomgang
 - Datamaskiner (PC/MAC ol)
 - Mobiltelefoner
 - Innebygde systemer
- + Litt mer om...
 - Innebygde systemer
 - Prosessorarkitekturer



<https://icon-library.com/icon/confusion-icon-13.html>>Confusion Icon # 101165

Hvilke valg har vi?

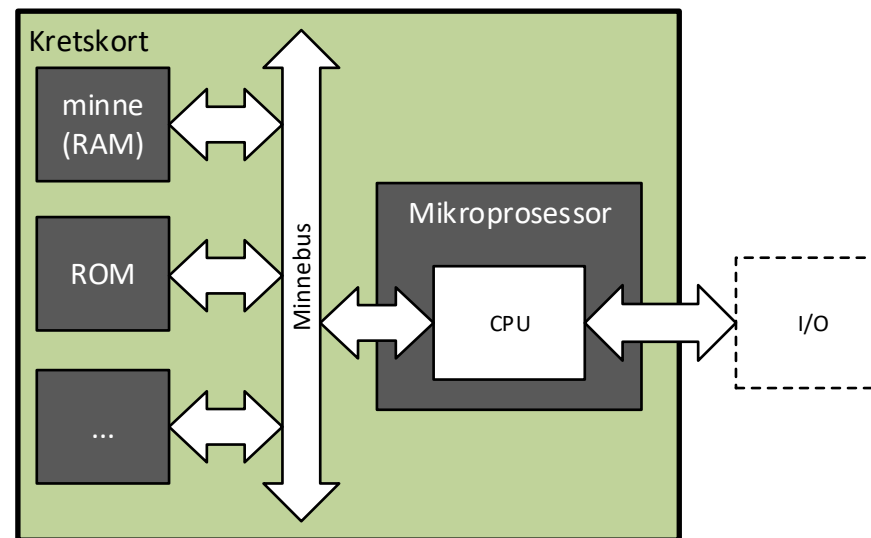
	Datamaskiner, servere	Mobiltelefoner	Innebygde systemer (Embedded systems)
Organisering	Mange separate kretskort • hovedkort • minnekort • skjermkort • SSD, HD	Alt på ett kretskort • Maksimal plassutnyttelse	Tilpasset • Ofte enkeltstående kretskort
Plassbruk	Stor	Svært liten	ofte liten
IO	Standardiserte grensesnitt • USB, Ethernet, displayport, wifi, .. • Krever periferienheter med egne kontrollere.	<i>som datamaskiner, men noe mer begrenset</i>	Avhengig av formålet • Fullt tilpasset • <i>Standarder for chip-chip kommunikasjon:</i> • SPI, I2C, UART
Operativsystem	Ja	Ja	Kun ved behov (<i>helst ikke</i>)
Strømforbruk	Stor	Liten	Tilpasset
Proseszor-arkitektur	Mikroprosessorer • CISC (for det meste) • Intel, AMD, etc	Mikroprosessorer • ARM, RISC • Qualcomm, Apple, Samsung, ...	Mikrokontrollere (<i>Mikroprosessorer</i>) FPGA DSPer ASICs
sanntidsbruk?	Nei • Ytelse muliggjør mye, men kan ikke garanteres	Nei	<i>Ja!</i>



<https://icon-library.com/icon/confusion-icon-13.html> Confusion Icon # 101165

Mikroprosessorer

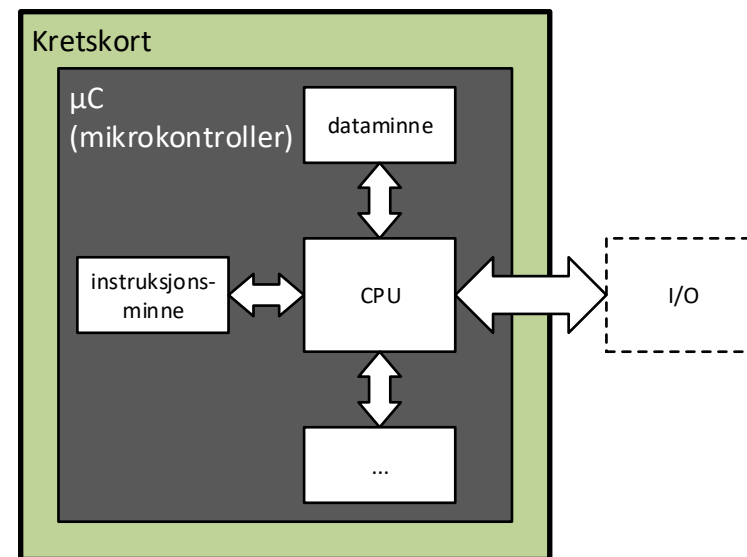
- Prosessor «CPU» (*Central processing unit*) alene
 - Minne og prosessor i separate brikker eller kretskort
 - Prosessor er enkel å lage/ spesialisere
 - Mer minne => mer avansert programvare
- Von Neumann arkitektur
 - Alle periferienheter koblet til den samme minnebussen
 - Ie «Program og data i samme minne»
 - Vanskelig å bruke, men enkel arkitektur gir god hastighet
- Høynivå programmeringspråk vanlig
 - *Python, Java, C++, etc*
- Bruker som regel et OS.
 - Krevende å utvikle for sanntidsoperasjon
- *Pipelinet arkitektur*
 - *En operasjon er ikke ferdig før den neste startes*
- *Ikke førstevalg for mekatronikk*



- Arkitekturer
 - RISC, CISC, ARM
- Produsenter
 - AMD, Apple, Intel, Qualcomm, Samsung, ...
- Bruksområder
 - PCer
 - Mobiltelefoner
 - Nettbrett
 - Biler
 - Ettkortsmaskiner
«Single board computers»
 - Raspberry Pi, Beagle Bone , ...
- *Mer om dette i IN2060...*

Mikrokontrollere ($\mu C = \mu c$)

- *Ett chips datamaskin*
- Minne og prosessor på samme chip
 - Typisk flashminne for program.
 - Data lagres i RAM
 - Periferienheter (timere ol.) i samme chip
- Harvard-arkitektur (oftest)
 - Separat instruksjons- og databuss
 - Lettere å bruke, men mer komplisert arkitektur
- Kan brukes til sanntidsoperasjon
 - Ikke OS
 - Single-cycle operasjon (ikke pipeline)
- Dedikerte periferienheter /ressurser
 - timere, tellere, pwm moduler
 - IO pinner
 - AD'er
 - Busser til periferienheter
 - SPI, I2C, UART, ...
- Programmering på lavnivå
 - C, assembler
- *Velegnet for mekatronikk formål*



- Arkitekturnavn
 - ARM cortex-M
 - Atmel AVR
 - intel 8051
 - PIC
- Kjente aktører
 - Arduino
 - Microchip
 - Atmel
 - ST Microelectronics

FPGAer, DSP, ASIC

- FPGA (Field programmable gate array)
 - Ikke en prosessorarkitektur
 - Programmering bestemmer funksjonen til en digital krets.
 - Man velger selv prosessering og parallellitet
 - *Kan* ha prosessor(er)
 - System on Chip «SOC» / «PSOC»
 - Full kontroll på samtidighet og timing
 - Svært gode muligheter til å styre IO.
 - Designspråk:
 - VHDL, Verilog, *system C*
 - Ikke programmering i normal forstand
 - *Velegnet til mekatronikk og robotikk, men krever kjennskap til digital design.*
 - IN3160 går i dybden på digitale design med FPGA
- DSP (Digital Signal Processing unit)
 - Spesiallaget mikrokontroller eller mikroprosessor
 - For regneoperasjoner / signalbehandling
 - MAC
 - Lyd- og bildebehandling
 - FIR-filtre
 - Osv
 - Betegnelsen brukes på noen motorkontrollere
 - *Ikke eget tema*
- ASIC (Application Specific Integrated Circuit)
 - = Det å lage en brikke (chip) med et bestemt formål
 - Brikker med integrerte kretser er alle ASICs, men vi bruker ikke betegnelsen når de har andre innarbeidede navn
 - Å lage ASICs krever inngående kjennskap til digital design og CMOS teknologi
 - Startkostnad i millionklasse
 - *Ikke vanlig i mekatronikkdesign*

FPGA

- Xilinx,
- Altera/Intel, ...
- ...

DSP'er

- Texas Instruments
- Analog Devices
- ...

ASIC

- TSMC
- ?

«Periferienheter»

- «Enheter» i «periferien» til prosessoren. DVS
 - *Kan* være hva som helst
 - Eks:
 - USB kontroller
 - håndterer signaler på USB bussen
 - AD-konverter
 - Oversetter analoge signaler til tall
 - Tellere
 - Teller antall pulser....
 - Timere
 - Kan brukes til å lage interrupter, venting osv.
 - PWM- moduler
 - Lager *pulsbreddemodulerte* signaler
 - For å drive motorer, lysdioder osv.
 - ...
 - *Vi kommer tilbake til mye av dette senere...*

IN1080- Lab:

- Styring av motorer
- Sansing
- Kontroll
- *(ikke alt på en gang fra start)*

- $\mu\text{C} := \text{Arduino}$
 - Programmering:
 - [C med påbygg](#)
 - Krever ikke forkunnskap i C
 - *Gir ikke inngående kunnskap om C*
 - *Mye template kode tilgjengelig*
- *Dedikert HW og ferdigskrevne moduler gjør det lett å få til mye*
 - Noen ganger gjør vi ting «mer tungvint» for å lære
 - *Ikke hopp bukk over læringsmålene...*
 - (det kommer en eksamen...)

Arduino kode

```
void setup() {  
    // sett inn alt oppsett..  
    // input pinner  
    // output pinner  
    // serielle grensesnitt  
    // globale variabler  
    // etc.  
}
```

```
void loop() {  
    // kjøres i loop  
    // helt til vi skrur av  
}
```

- Arduino-rammeverket kjører alltid
 - Setup først, dernest loop til evig tid.
- Andre prosedyrer kan deklarereres først men brukes bare når de blir kalt fra setup() og loop()

Noen Arduino-kommandoer (prosedyrer)

- `digitalRead()`
 - Leser digitalverdien til en pinne (0 / 1)
 - *Krever at pinnen er satt opp som input*
- `digitalWrite()`
 - Skriver til en pinne
 - '0' => 0V, '1' => 3,3V
 - *Krever at pinnen er satt opp som output*
- `analogRead()`
 - Leser spenningen til en pinne
 - 0V – 3,3V => 0 - 4096
 - Bruker en intern ADC koblet til noen spesifikke pinner (*ADC tema senere i kurset*)
- `delayMicroseconds()`
 - Venter en bestemt tid gitt i mikrosekunder
 - Bruker en intern timer/ interrupthåndtering
- `delay()`
 - Venter et antall millisekunder

Kommandoer for serieporten (*Krever at serieporten er satt opp*)

- `Serial.read()`
 - Leser **én byte** fra serieporten
 - Kan brukes til å snakke med PC (bruker USB som virtuell serieport)
- `Serial.write()`
 - Skriver **én byte** til serieporten
 - Brukes typisk til å snakke med PC (eller andre mikrokontrollere)
- `Serial.println()`, `Serial.print()`,
 - Skriver ASCII strenger til serieporten
 - Med eller uten linjeskift
- `Serial.readString()`
 - Leser en hel streng

Lese

- Læreboka (COK):
 - Introduksjon til boka og Mekatronikk:
 - 1.0, 1.1, 1.4, 1.5
 - Mikro- kontrollere og prosessorer
 - 2.1-2.3, 2.8, 2.9, 2.10, 2.12
- Oppgaver
 - 2.1, 2.2, 2.3