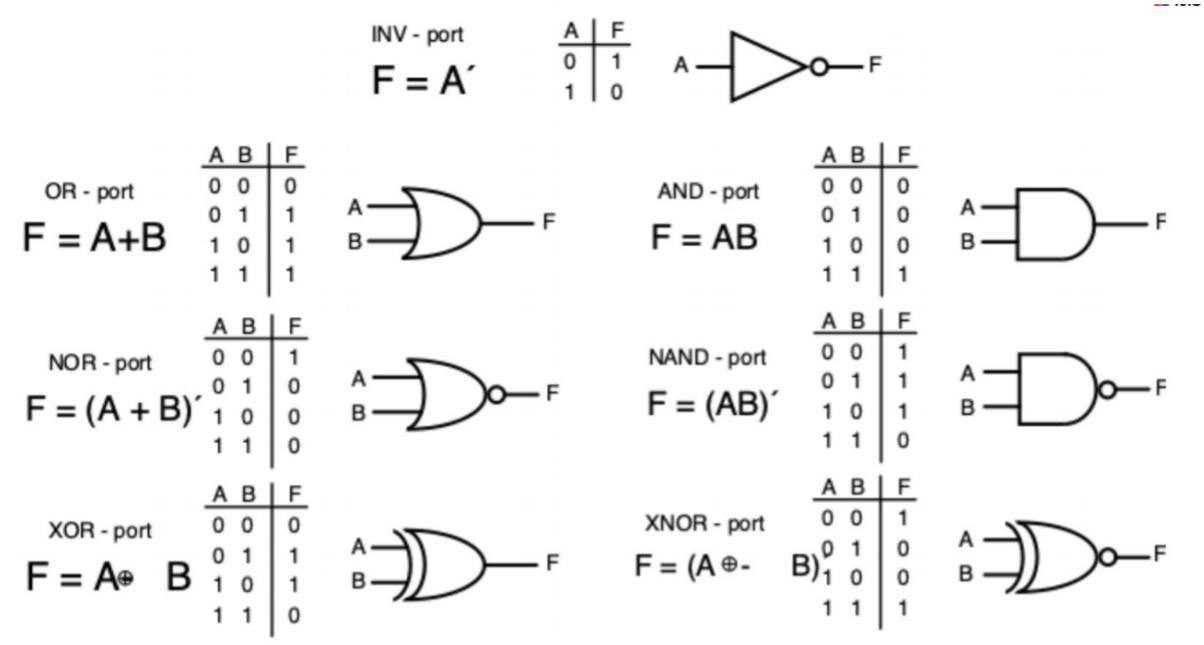
Seminar 7 (porter, synkronisering)

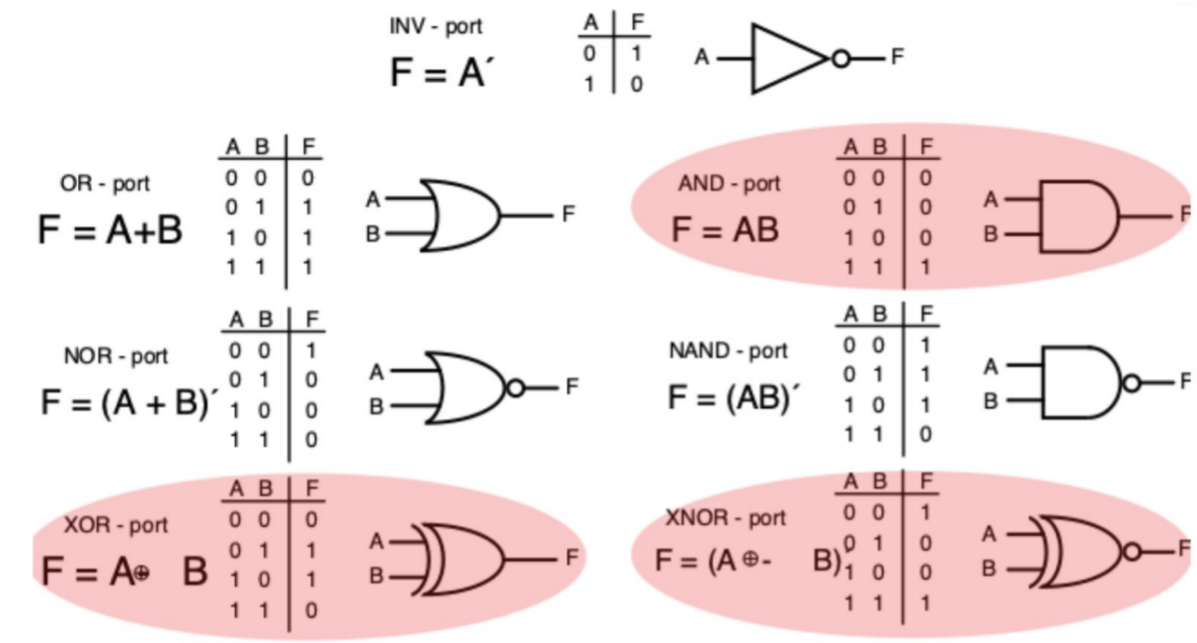
Pensum

Porter og funksjoner

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/7e41c60e-1ec8-454c-9b3c-78b51dbd857e/in1020---forelesning-3---2021-full.pdf

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/dae8bcf1-21cc-44b7-90d8-69c4e6081ae6/03---studentnotater-2021.pdf





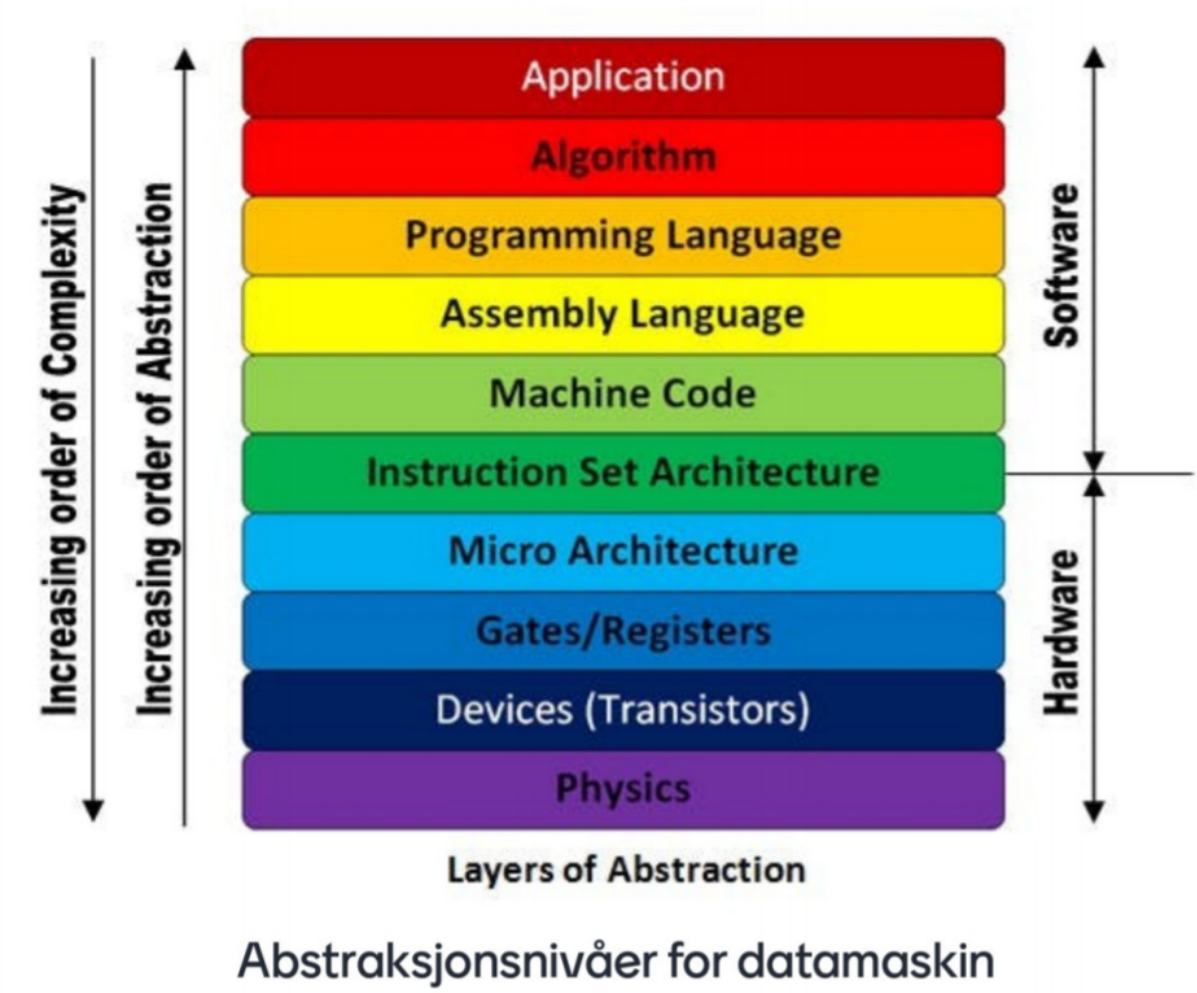
Fysiske begrensninger og synkronisering

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/a30f72a6-af94-49be-b783-e4281e452e66/in1020---forelesning-4---2021-web.pdf

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/edc86750-8573-4874-9f61-8041cb740837/04---studentnotater-2021.pdf

NAND game er en spill som hjelper deg til å forstå hvordan logiske porter fungerer. Du bygger alt fra bunnen - starter med NAND-porter og bygger din vei videre.

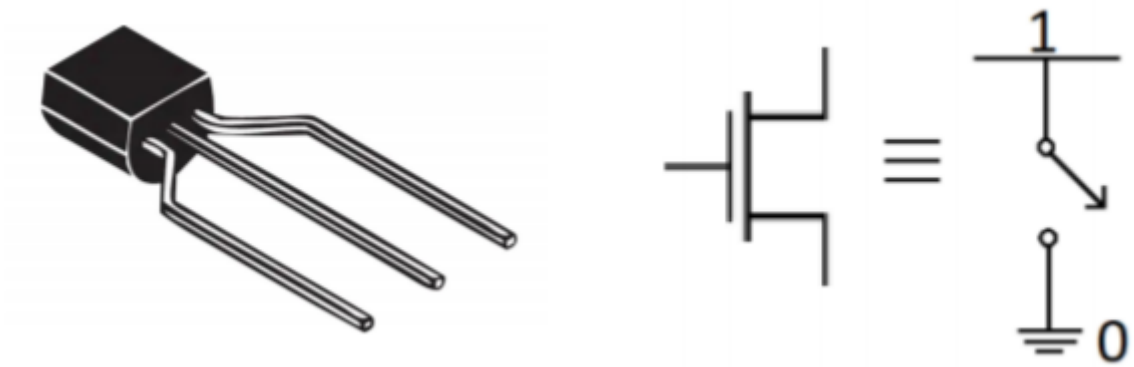
Transistorer



Abstraksjonsnivåer for datamaskin

Vi er nå på den andre nivået (devices (transistors)).

Transistor:



Tre stikkekontakter:

- Gate

- Drain
- Source

Transistor er en slags lås. Hvis det er spenning på **Gate**, så går strømmen gjennom **Drain** og **Source**, og låsen er "stengt" - det er strøm i hele kretsen og den går videre (output blir 1). Hvis det er ingen spenning på Gate, så blir det ingen strøm fra **Source** til **Drain**, og låsen er "åpen" - det er ingen strøm som går videre og output er 0.

Ideelle transistorer funker uten noe forsinkelse, men de ekte har den. Dette er en grunn vi trenger synkronisering.

Synkronisering

Formelen til å regne ut frekvens:

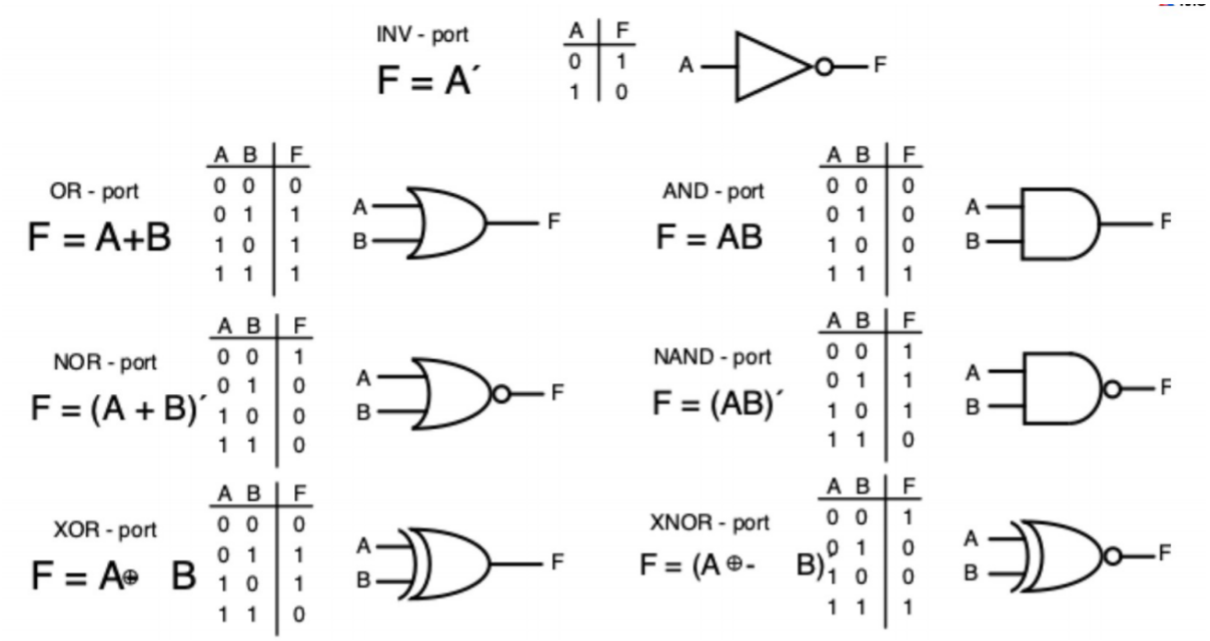
$$frekvens = \frac{1}{klokkeperioden}$$

Ukeoppgaver

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/61bac318-320e-4672-aee7-3af00294912f/in1020---uke-2.pdf

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/d98ff28b-c7c2-4be4-abf4-9236045d2b4f/in1020---uke-2---fasit.pdf

Den generelle tilnærmelsen til å løse kretsoppgaver



Det vi gjør er logiske operasjoner som er en del av boolsk algebra. Dere vil lære mer om dette i IN1150 kurset.

En av tingene dere trenger å kan kretser er å forstå bindesterkhet til ulike operasjoner.

Bindesterkhet rekkefølge

- A' - sterkest
- A*B
- A+B - svakest
- () - forsterker alt som er inni

To måter å løse kretsoppgaver

- **Venstre til høyre**. Brukes når vi bygger en krets fra en gitt formel.
  - Går fra de sterkeste til de svakeste (↓, topp til bunn), siden vi går fra enkelte uttrykk (A, B) til de komplekse (F).
- **Høyre til venstre**. Brukes når vi har en ferdig implementert krets og trenger å skrive en formel.
  - Går fra de svakeste til de sterkeste (↑, bunn til topp), siden vi går fra de komplekse uttrykk (F), til de enkelte (A, B).

Noe regler i boolsk algebra

- **Inverse law for addition (OR-port):** A + A' = 1
- I**nverse law for multiplication (AND-port):** A * A' = 0
- **Distributive law:** A + BC = (A + B)(A + C)
- **"Decyphering" (simplifying) XOR port:** F = A xor B = A'B + AB'