11. mars 2014

Digitale (binære) kretser og systemer – en kort introduksjon

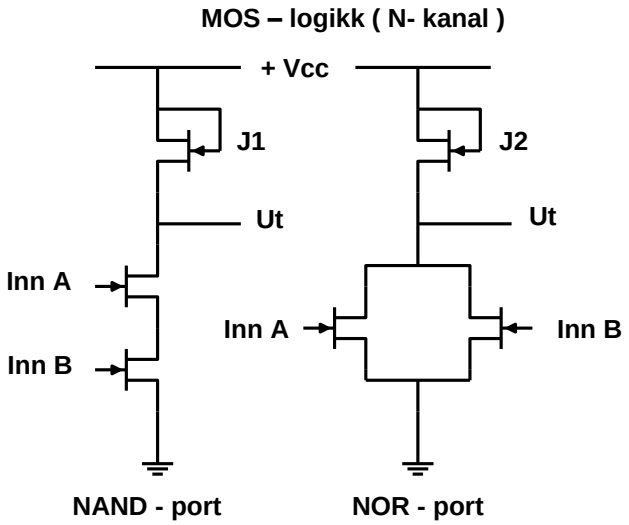
©Torfinn Lindem
Fysisk institutt UiO

Læreboka i kurset FYS1210 beskriver hvordan halvlederkomponenter som dioder, bipolare transistorer (BJT) og felteffekttransistorer (FET) kan brukes til å lage analoge forsterkere for strøm og spenning.
Vi skal nå bruke de samme komponentene til å konstruere ”binære systemer”.
Binære systemer er kretser som arbeider med to mulige tilstander - “0” og ”1” .
Det teoretiske grunnlaget for hvordan slike kretser kan bygges sammen til komplekse datamaskiner ble gjort av George Boole, engelsk matematiker (1815 -1864).
Boolsk algebra er et system for matematisk analyse av binære systemer.
En Boolsk variabel kan ha en av to mulige tilstander, ”0” og ”1” ev. ”false” og ”true”
Binære systemer kan brukes til både logiske og aritmetiske operasjoner.

Logiske porter – ”logic gates”

Betegnelsen ”logic gates” (logiske porter) brukes om/på digitale (binære) kretser som anvendes når vi skal realisere Boolske likninger.

Grunnlaget for boolsk algebra bygger på 3 typer logiske porter: NOT, AND og OR
Disse 3 elementene er byggesteinene for alle digitale systemer. Ved en *negasjon* (invertering) av uttrykkene for AND og OR fremkommer portene NAND og NOR.
Dette er kretser som lettere lar seg realiser kretsteknisk - samtidig som de kan realiser de samme Boolske likningene.

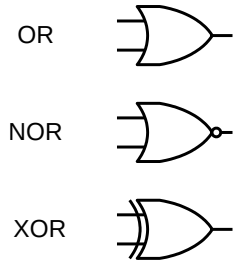


Vi ser her en NAND-funksjon realisert ved en seriekopling av transistorer.
(FET J1 er koplet som motstand)

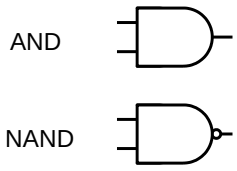
En NOR-funksjon realiseres ved parallellkopling av transistorer.
(FET J2 er koplet som motstand)

Sannhetstabeller

Sannhetstabeller gir sammenhengen mellom logiske nivåer på inngangene og utgangen til portene – legg merke til symbolene som brukes:



OR , NOR og XOR				
A	B	OR	NOR	XOR
0	0	0	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	0	0



AND og NAND			
A	B	AND	NAND
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Kopler vi sammen inngangene på en NAND-port får vi en NOT

XOR – er en noe spesiell krets. Den kan realiseres ved sammensetning av AND, OR og NOT. Kretsen blir mye brukt i blant annet aritmetiske systemer – addisjon og subtraksjon. Digitale datamaskiner kan både multiplisere og dividere – men husk at multiplikasjon i utgangspunktet er en gjentatt addisjon – og divisjon er en gjentatt subtraksjon.
Binær subtraksjon (B - A) kan utføres som en addisjon av ener-kompliment til A (A negert). Dvs. (B - A) = $(B + \overline{A})$
Vi ser at all aritmetikk kan bygges opp av kretser for addisjon. Mer om dette i avsnittet om kombinatoriske kretser.

Tallsystemer

Desimalt system med grunntall 10.
Hva representer de enkelte faktorene i tallet 321?

$$321 = 3 \cdot 10^2 + 2 \cdot 10^1 + 1 \cdot 10^0 = 300 + 20 + 1$$

Binært tallsystem med grunntall 2

Eksempel på hvordan vi regner om fra et binært tall. Gjør om 10011 til et desimalt tall

$$10011 = 1 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 16 + 0 + 0 + 2 + 1 = 19$$

Desimalt	Binært	Desimalt	Binært	
0	0000	5	0101	Et binært siffer (0 eller1)
1	0001	6	0110	kalles et ”BIT”
2	0010	7	0111	8 BIT danner en BYTE
3	0011	8	1000	4 BIT danner en ”nible”
4	0100	9	1001	

Kretsfamilier

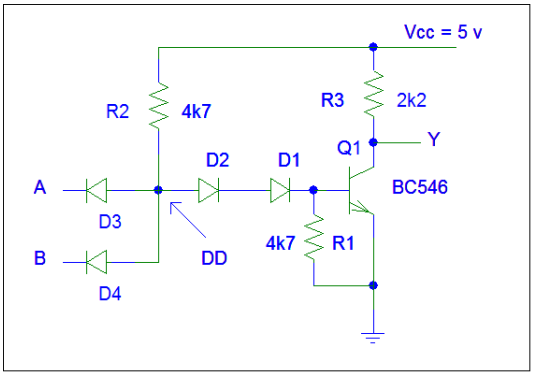
Det finnes flere ”familier” logiske kretser –
det som skiller disse er teknologien som benyttes

Bipolare komponenter (BJT) brukes i to typer logiske kretser: DTL, TTL og ECL.
DTL - Diode Transistor Logic (- historisk – første generasjon 1952-1963)
TTL – Transistor Transistor Logic (- brukes fortsatt i mange systemer –2009)
ECL - Emitter Coupled Logic (- mye brukt i raske datamaskiner 1970 -1990)

Unipolare komponenter (FET) brukes i logiske kretser med NMOS og CMOS teknologi. Teknologien rundt FET utvikles kontinuerlig. Brukes i dagens prosessorer.

Diode Transistor Logikk (DTL)

Første generasjon digitale kretsfamilie. Figur 1 viser et eksempel på en DTL - en 2 input NAND krets. Denne kretsen brukes i Laboratorieoppgave 4 på FYS1210.



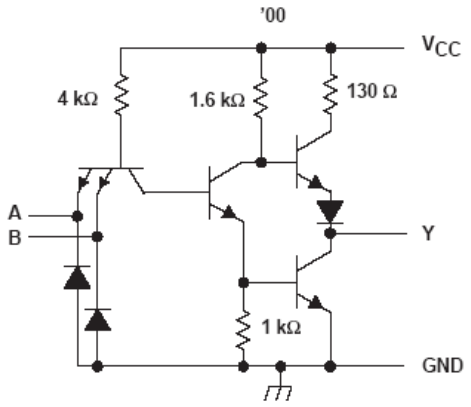
NAND			
A	B		Y
0	0		1
0	1		1
1	0		1
1	1		0

Figur 1

Uten signal på noen av inngangene A eller B vil transistoren BC546 lede.
(Utgangen Y vil være lav – logisk ”0”)
Basis er koplet til +5volt via diodene D1 og D2 som står i lederetning.
Når transistoren leder vil basis-emitter spenningen V_{EB} være ca. 0,7 volt. Vi ser at punktet DD vil ha en spenning på 2,1volt ($V_{D1} + V_{D2} + V_{BE} = 0,7 + 0,7 + 0,7 = 2,1$).
Skal transistoren lede må spenningen i punktet DD minst være 2,1 volt.
Hvis en av inngangene – f.eks B tilkoples ”0” volt (koples til ”jord”) vil dioden D4 lede og spenningen i DD faller til 0,7 volt (= spenningsfallet over dioden D4)
Hvis minst en av inngangene går til logisk ”0” vil utgangen Y gå til logisk ”1”.
Denne type logiske kretser er langsom. Operativ klokkefrekvens max 1 MHz – ofte vesentlig lavere. Dette skyldes at transistoren går i ”saturation” . Base – kollektor dioden begynner å lede. Det tar lang tid å skru ”av” en transistor i ”saturation”. Dette skyldes at det tar tid å fjerne ladningsbærere fra basis og gjenopprette sperresjiktet på BC-dioden.

Transistor Transistor Logikk (TTL)

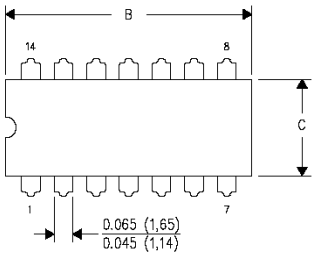
Med ”ny” teknologi forsøker man å redusere antallet ”langsomme” komponenter (1960). Diodene erstattes med en transistor med flere emittere. Man lager en kopling som i størst mulig grad unngår å sette transistorene i dyp ”saturation”- løsninger hvor man raskt kan trekke ladning ut fra basis. Man oppnår derved en vesentlig raskere operasjon. Se eksempel Figur 2. TTL 2 input NAND (¼ av 7400) De to diodene på inngangene beskytter mot negativ overspenning. De to transistorene på utgangen danner en såkalt ”totem pole” som sikrer rask operasjon, lavt strømforbruk og lav utgangsimpedans.



NAND			
A	B		Y
0	0		1
0	1		1
1	0		1
1	1		0

Figur 2 TTL 2-input NAND

Samtidig som denne teknologien ble tatt i bruk ble det introduserte en fysisk og elektrisk standard for integrerte logiske kretser.(1964) De integrerte kretsen pakkes i en kapsel beregnet på hullmontering. Pinnene for tilkopling av signal og forsyningsspenning organiseres i to parallelle rekker - Dual In Line, DIL Se bilde.

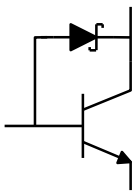


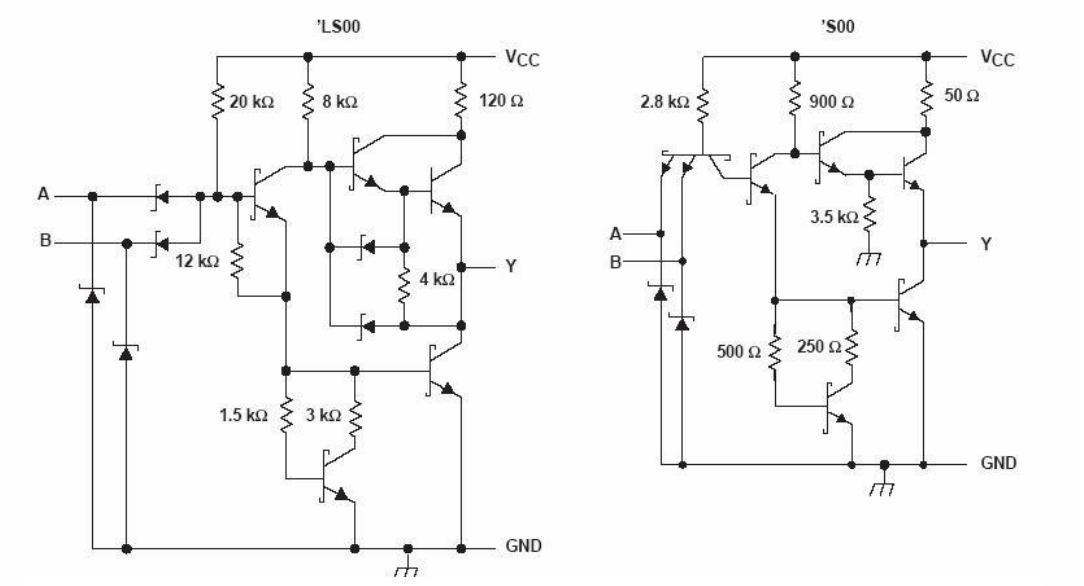
Kretsene har en felles nummerbetegnelse 74xxx.
74 forteller at det er en TTL-krets, de neste tallene forteller hvilke logiske funksjoner som ligger i pakken.
7400 forteller om en TTL-krets som inneholder 4 separate 2-input NAND-porter. Alle produsenter av TTL-logikk bruker samme konfigurering. Det betyr at du kan kjøpe en 7400-krets fra flere forskjellige produsenter – og du vet at de er pin-kompatible. Dvs. du kan bytte ut en 7400 krets fra Philips med en fra Texas Instruments. Det som skiller produsentenes produkter fra hverandre er en bokstavkode foran 74-tallet. En krets fra Texas heter SN7400, - SN er koden for Texas Instruments.
Filosofien bak dette standardiserte regimet er at komplekse digitale systemer ikke skal være avhengig av komponenter fra en enkelt leverandør. Hvis et produksjonsanlegg i for eksempel USA ble utsatt for en ulykke / konkurs – ville en kretsprodusent i Europa eller Japan kunne levere identiske kretser. Alle seriøse produsenter av komplekse elektroniske systemer må tenke på alternative komponentleverandører. Du vil ofte se at det er en bokstavkode inne i selve 74-nummeret f.eks. 74**L**00
Denne/disse bokstavene forteller noe om teknologien som er brukt for å realisere den logiske/ aritmetiske funksjonen. I dette tilfelle vil L bety ”Low power” – Motstandene som er brukt internt i kretsen er store – ofte en faktor 10 større enn på en vanlig 7400

krets. Dette gir lavere effektforbruk (1mW), men kretsen blir langsom. – dvs. 74L00 må arbeide med lavere maksimal klokkefrekvenser enn 7400.

Transistor Transistor Logikk, TTL med Schottky dioder (74S00)

Setter vi en Schottky-diode mellom basis og kollektor (se figuren) hindrer man basis - kollektordioden (BK-dioden) å bli forspent i lederetning.
Schottky-dioden (metall/halvleder-overgang) leder ved ca. 0,3volt. BK-dioden trenger ca 0,7volt før den leder. Det betyr at Schottky-dioden leder først, - og den hindrer derved transistoren i å gå i metning. Det betyr at transistoren fort kan skru seg "av" – da sperresjiktet i BC-dioden er på plass. Figur 3 viser hvorledes dette er realisert i kretsen 74S00 (S- forteller at det er brukt Schottky - teknologi i denne TTL-pakken) På detaljskjema kan du se at dette er markert ved at basiselektroden er formet som en stilisert, stor S.





Figur 3 74LS00 og 74S00 2-input NAND med Schottky-diode

Figur 3 viser hvordan Schottky -teknologi er brukt på 74LS00 og 74S00. 74**LS**00 betyr at kretsen er **L**ow power **S**chottky Det vil si at det i tillegg til schottky dioder er satt inn større motstandsverdier for å redusere strømforbruket (effektforbruket) i kretsen. LS- kretser vil få litt større propagation delay enn rene S –kretser.

Kombinatoriske digitale kretser

Charles Babbage (1791 -1871) var den første som systematisk arbeidet med det teoretiske og praktiske grunnlaget for en «computer». I 1833 bygget han en programmerbar mekanisk «analytical engine» som inneholdt alle de grunnleggende enhetene vi i dag finner i våre datamaskiner.
En kontroll-logikk (logic unit), en aritmetisk enhet (arithmetic unit), en data-hukommelse (memory) – i tillegg til enheter som kunne sende data inn og ut av systemet. - Alle disse enhetene er realisert i dagens elektroniske digitale systemer.

Vi har sett at enkle logiske porter kan realisere spesielle funksjoner. De fleste logiske operasjoner kan utføres ved å sette sammen logiske porter av type – NAND og NOR. (George Boole, 1815-1864)

ADDISJON AV BINÆRE TALL

Vi skal se at enkle logiske funksjoner kan settes sammen /organiseres slik at vi kan utføre aritmetiske operasjoner. Grunnlaget for all aritmetikk ligger i addisjon. Kan vi først addere to binære tall, kan vi like lett subtrahere de samme tallene. Det skal ikke mye fantasi til før vi med utgangspunkt i addisjon og subtraksjon kan utføre både multiplikasjon og divisjon.
La oss begynne med å addere to binære bit.
Hvert bit har to mulige tilstander/verdier - 0 eller 1. Dette gir oss 4 mulige kombinasjoner:

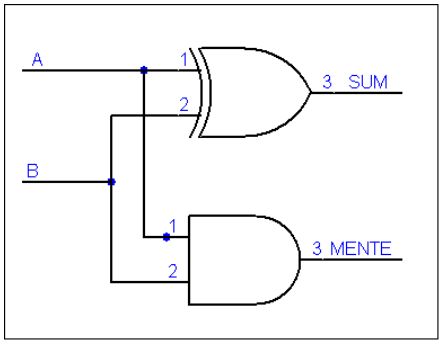
0 + 0 = 0
0 + 1 = 1
1 + 0 = 1
1 + 1 = 0 + mente (1)

Den siste kombinasjonen, 1 + 1 viser at addisjonen kan ha et resultat på 2 bit. En sannhets-tabell for binær addisjon kan derfor se slik ut:

INPUT		OUTPUT	
A	**B**	**MENTE**	**SUM**
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Vi ser her at MENTE er en ren AND-funksjon og SUM er en Eksklusiv OR. (Se side 2)

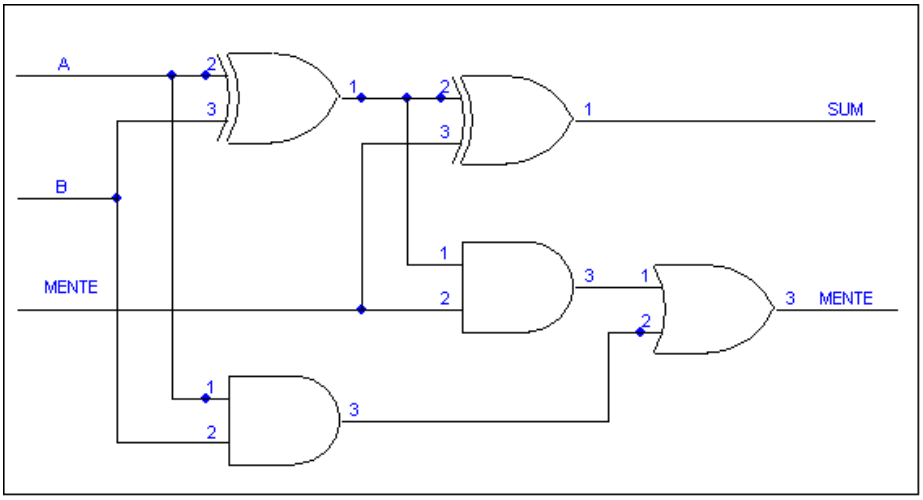
Vi kan derfor realisere denne kretsen lett ved å bruke en AND og en EXOR



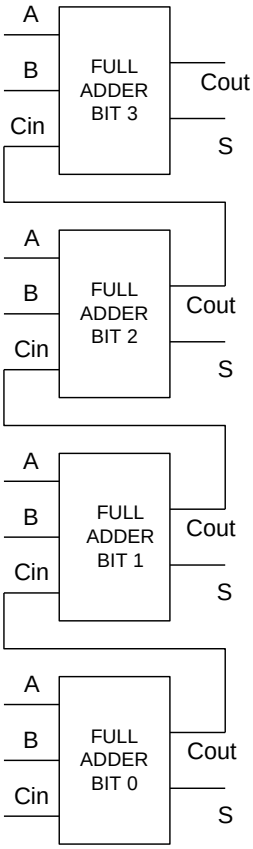
Figuren viser en krets som bare gjør ”halve jobben”. Den mangler ”Mente inn” – og vi kaller derfor en slik krets for en Halv-adder

FULL ADDER

Skal vi realisere en krets som gir oss komplett addisjon, med mente inn og ut, må vi sette sammen to halv-addere. Kretskoplingen sammen med ”sannhetstabellen” vises under :



INPUTS			OUTPUTS	
A	**B**	**Mente inn**	**Mente ut**	**S**
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1



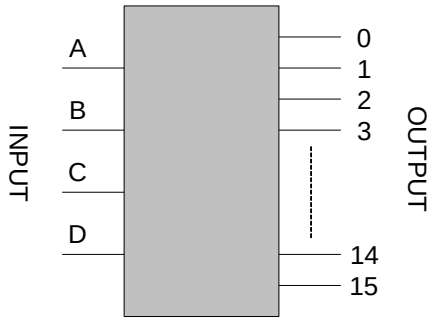
Vi ser at en fullstendig addisjon lett lar seg realisere.
Skal vi addere binære tall med flere bit må vi arbeide med mange full-addere.

Se på figuren til høyre og legg merke til distribusjonen av mente. Minst signifikant bit nederst i rekka.
Vi må vente med avlesning av resultatet til vi er helt sikker på at ”mente” har forplantet seg gjennom alle kretsene..

KOMBINATORISKE KRETSER

DEKODERE

System som mottar en N-bit kode og som etablerer en (og bare en) ”1” på 2^N utgangslinjer.
Ser først på en 4-bit dekoder. 4 bit gir $2^4 = 16$ kombinasjoner .
Se Figur 4 – en 4 bit kode legges inn til dekoderen - for eksempel 1001
Dette er en “ 8 4 2 1 kode” – tallene viser hvilken ”vekt” hvert bit har i koden.
I vår 4-bit kode har Least Signifcant Bit (LSB) vekt ”1” - Most significant bit (MSB) har vekt ”8”. Det betyr at binærkoden koden 1001 har verdi 8 + 0 + 0 + 1 = 9.
Dekoderen vil etablere en ”1” på utgang 9.
Vi bruker ofte en hexadesimal tallrekke når vi beskriver en slik 4-bit kode :



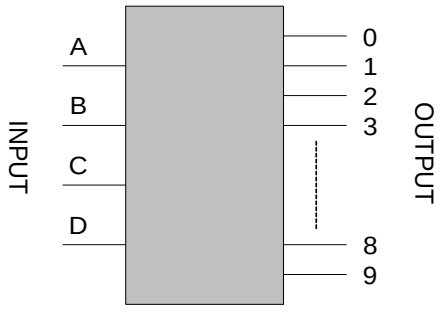
Figur 4. 4 BIT til 16 linjers dekoder

0 1 2 3 ---- 9 A B C D E F (HEX)
hvor 0000 = 0 og 1111 = F

- men i denne dekoderen teller vi
  Desimalt: 0 1 2 3 ----- 14 15
  utgangslinjer

BCD til desimal dekoder

BCD = Binær kodet desimal. En “ 8 4 2 1 kode” hvor alle binærkodene - opp til 1001 aksepteres. Se Table1 under til høyre.



Figur 5. 4 bit BCD til desimal dekoder

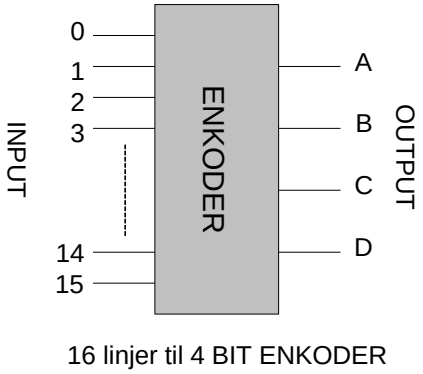
A	**B**	**C**	**D**	Ut
0	0	0	0	**0**
0	0	0	1	**1**
0	0	1	0	**2**
0	0	1	1	**3**
0	1	0	0	**4**
0	1	0	1	**5**
0	1	1	0	**6**
0	1	1	1	**7**
1	0	0	0	**8**
1	0	0	1	**9**

Linjen ”ut” går til ”1”

Table 1

ENKODER

Den inverse prosess til dekoding. For hver "1" på en av N linjer genereres en N-bit kode. Se Table 2.



16 linjer til 4 BIT ENKODER

																A	B	C	D
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
2	0	1	0	0	0	0	0	0								0	0	1	0
3	0	0	1	0	0	0	0	0								0	0	1	1
4	0	0	0	1	0	0	0	0								0	1	0	0
5	0	0	0	0	1	0	0	0								0	1	0	1
6	0	0	0	0	0	1	0	0								0	1	1	0
7	0	0	0	0	0	0	1									0	1	1	1
8	0							1								1	0	0	0
9	0								1							1	0	0	1
A	0									1						1	0	1	0
B	0										1					1	0	1	1
C	0											1				1	1	0	0
D	0												1			1	1	0	1
E	0													1		1	1	1	0
F	0														1	1	1	1	1

Table 2

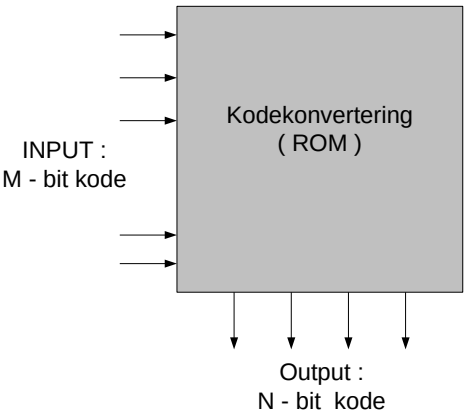
Ofte vil "virkelige" enkodere være organisert etter et litt annet mønster enn den idealiserte som vist i tabell 2.

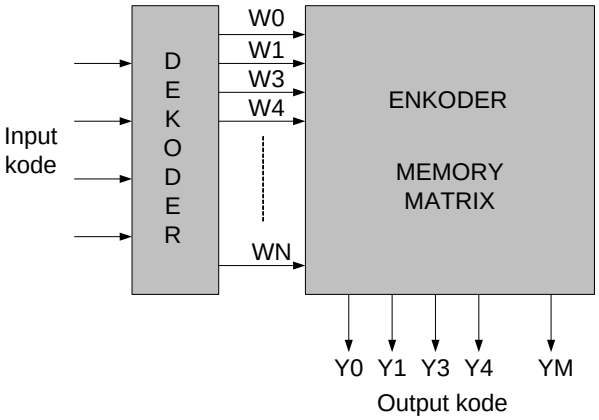
Laboppgave 5 på kurset bruker encoder-kretsen 74LS147 – hvor sammenhengen mellom input og output er vist i tabellen under. (BCD-kode)

H = High L = Low X = irrelevant - det betyr at inngangen kan være enten "1" eller "0" – dvs. den er uten betydning for "Data output".

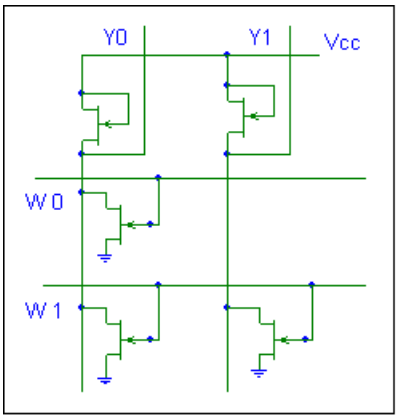
Data input									Address data output			
1	2	3	4	5	6	7	8	9	D	C	B	A
H	H	H	H	H	H	H	H	H	H	H	H	H
X	X	X	X	X	X	X	X	L	L	H	H	L
X	X	X	X	X	X	X	L	H	L	H	H	H
X	X	X	X	X	X	L	H	H	H	L	L	L
X	X	X	X	X	L	H	H	H	H	L	L	H
X	X	X	X	L	H	H	H	H	H	L	H	L
X	X	X	L	H	H	H	H	H	H	L	H	H
X	X	L	H	H	H	H	H	H	H	H	L	L
X	L	H	H	H	H	H	H	H	H	H	L	H
L	H	H	H	H	H	H	H	H	H	H	H	L

READ – ONLY MEMORY (ROM)





Den funksjonelle relasjon mellom "output" og "input" er bygget inn i "hardware". Informasjonen er lagret permanent.
Den interne organiseringen er ofte gjort som vist på figuren over –
med først en DEKODER som så setter opp en entydig forbindelse inn til en ENKODER MEMORY MATRIX.

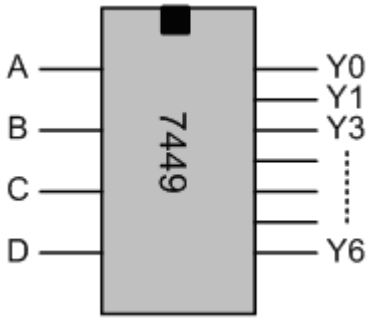


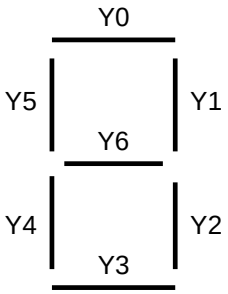
En encoder matrise kan være koplet som en permanent sammenkopling mellom transistorer som vist i figuren til venstre. Tabellen under viser sammenhengen mellom input W og output Y

W0	W1	Y0	Y1
0	0	1	1
0	1	0	0
1	0	0	1

ROM applikasjoner

ROM kan brukes til oppslagstabeller for trigonometriske funksjoner logaritmer etc.
Et typisk eksempel vil være en driver for de 7 lysdiodene i et siffer – display:

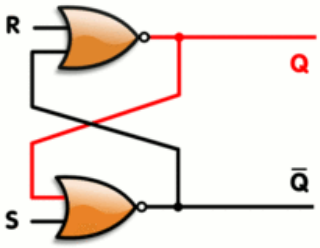




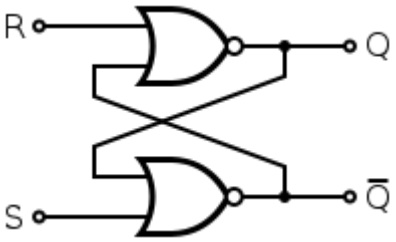
SEKVENSIELLE SYSTEMER

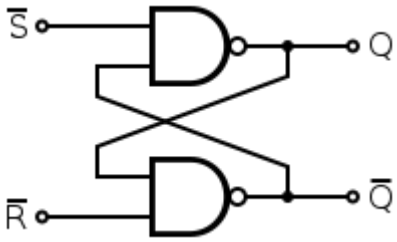
Vi skal behandle kretser av type Flip-Flop og to typer kretser bygget opp av slike elementer, - REGISTERE og TELLERE

1-bit lager (LATCH)



Skal en logisk krets "huske" en tilstand – og opprettholde denne tilstanden - må kretsen inneholde en form for tilbakekopling (feedback).
En latch er en krets som kan eksistere i en av to stabile tilstander, enten med utgangen $Q = 1$ $(\overline{Q} = 0)$ som kalles "1" tilstand, eller med $Q = 0$ $(\overline{Q} = 1)$ som kalles "0" tilstand.
En latch kan realiseres vha. både NAND og NOR-porter. Inngangene er ofte tilordnet bokstavene "S" og "R" som står for henholdsvis "SET" og "RESET". Tabellen under viser sammenhengen mellom kontrollsignaler og tilstanden på utgangene. Realiseres kretsen med NAND vil kontrollsignalene "Set" og "Reset" være "active low"

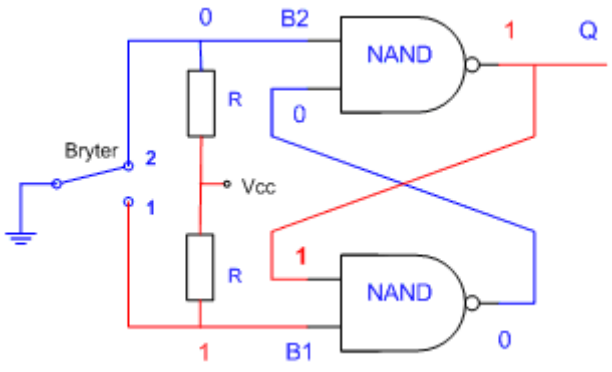


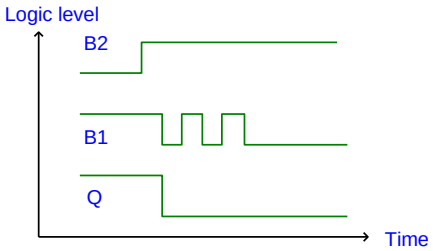


Input		Output	
R	S	Q	$\overline{Q}$
0	0	Latch	
1	0	0	1
0	1	1	0
1	1	Restricted	

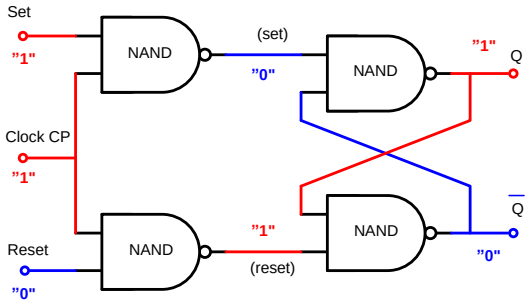
Bruksområde for en slik krets er begrenset. Men når vi skal kople en mekanisk bryter til et digitalt system må vi være sikker på at bryteren gir en – og bare en – puls inn til systemet. Dette kan være problematisk da mekaniske brytere kan "prelle" – dvs. det kan oppstå en rekke små korte pulser før bryteren faller til ro.
Med en latch vil utgangen Q låses til riktig verdi ved første kontakt. Se figuren under . Vi slår bryteren fra posisjon 2 til 1.

Klokkestyrt LATCH





Skal vi bruke LATCH som et aktivt element i digitale "klokkestyrte" systemer må vi tilføre kretsen ekstra styrelogikk. Vi legger på 2 NAND- porter foran latchen.

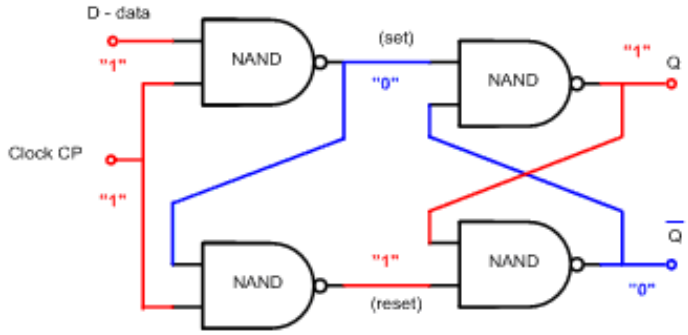


Input		Output	
R	***S***	***Q***	$\overline{Q}$
0	0	***Q***	$\overline{Q}$
1	0	0	1
0	1	1	0
1	1	Ubestemt	

Klokkesignalet (CP) bestemmer om latchen skal settes eller resettes. Det er først når CP går til ”1” at signalet på ”Set” og ”Reset” vil komme igjennom NAND- portene slik at de kan endre tilstand på utgangen Q.
Men vi kan få et problem hvis både ”Set” og ”Reset” blir ”1” samtidig. Dette kan vi løse med en J K LATCH (flip flop)

D-LATCH (Data LATCH)

Utgangen Q vil bli satt til verdien på D-inngangen når klokkesignalet går til ”1”. Vær oppmerksom på at så lenge klokka er høy (”1”) kan D endre verdi – og denne verdien vil legges ut til Q.

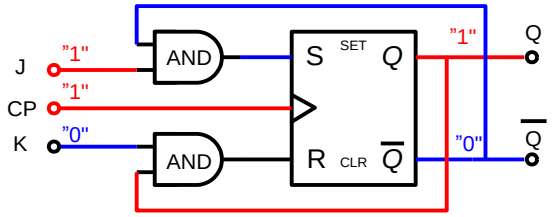


Input		Output	
D	***CP***	***Q***	$\overline{Q}$
0	0	***Q***	$\overline{Q}$
1	0	***Q***	$\overline{Q}$
0	1	0	1
1	1	1	0

Når klokka går fra ”1” til ”0” vil D-verdien bli ”låst” til Q-utgangen.

J K LATCH (flip flop)

I den klokkestyrte ”latchen” har vi et problem hvis ”Set” og ”Reset” begge er ”1”. Vi får da en ubestemt tilstand på Q. Dette kan vi unngå hvis vi lar ”S” og ”R” legges inn til en AND- port som styres av ”latchens” Q-utganger. Se figuren under.



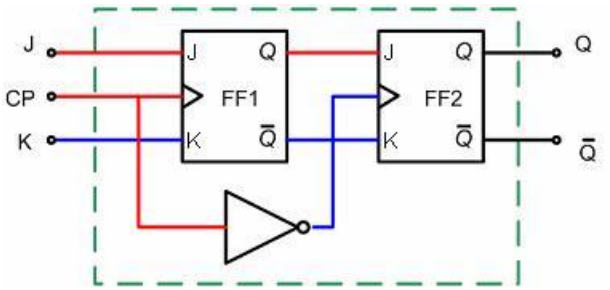
J	K	Q
0	0	Q
1	0	1
0	1	0
1	1	Q’

Transisjonstabell når CP=Høy

Ulempen med denne kretsen er at - så lenge CP er høy kan utgangene variere.

Master – Slave Flip- Flop

Dette er en krets som fjerner ulempene ved J K latchen.
Nå bruker vi 2stk. J K - latcher – hvor den ene styres av den andre. Se figuren under..



J	K	Q
0	0	Q
1	0	1
0	1	0
1	1	Q’

Transisjonstabell for master-slave Flip Flop.
Settes på negativ flanke av klokkepulsen

Utgangen på FF1 settes når klokkepulsen (CP) er høy ,”1”. Verdiene bestemmes av nivåene på J og K. Utgangene fra FF1 koples ikke frem til utgangene på FF2. Dette skyldes inverteren som kopler klokkepulsen til FF2. Så lenge CP på FF1 er høy ”1” – vil klokkeinngangen på FF2 være lav ,”0” - og følgelig koples ikke nivåene på J og K til utgangen.
I det øyeblikk klokkepulsen (CP) på FF1 skifter fra høy til lav vil inverteren sikre at FF2 får ”1” på sin klokkeinngang. Nivåene som i dette øyeblikk befinner seg på J og K låses nå til utgangen.
Vi har fått en krets hvor alle overganger (transisjoner) skjer på neg. flanke til klokkesignalet. Vi har en ”flankestyrt” krets. De fleste digitale systemer er bygget opp av slike kretser.

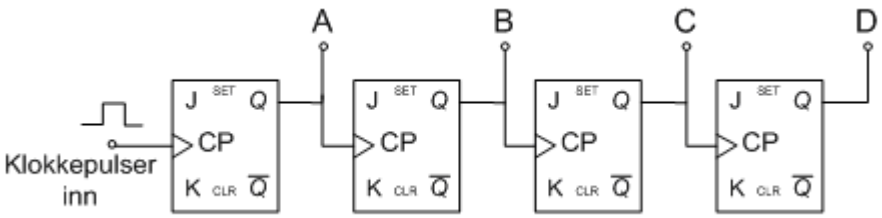
SEKVENSIELLE KRETSER - TELLERE OG REGISTRE

Med flip-flop-kretser av master-slave-type kan vi nå konstruere tellere og registre som alle skifter på negativ flanke til klokkesignalet. Dette gjør sammenstillingen av digitale systemer enklere og tryggere.
Vi må imidlertid skille mellom to forskjellige operasjonsmodi; - synkron- og asynkron operasjon..
I en asynkron operasjon vil klokkesignalet til et kretselement være avhengig av tilstanden til foregående elementer i kjeden – ofte er det foregående krets som leverer klokkesignalet til kretsen.
I en synkron operasjon skifter alle elementene i takt med samme klokkesignal.

Et felles krav for alle digitale systemer er at de kan *telle* – både forlengs og baklengs. Digitale klokker og ”timere” finner vi over alt – i mobiltelefoner, DVD-spillere og i biler. Tellere finnes i mange utgaver – men de er i sin grunnkonstruksjon svært like. Alle er bygget opp rundt flere ”flip- flop” kretser av ”master slave” type. Dette kan vi demonstrere med den enkleste av alle tellere – Binærtelleren.

Binærteller

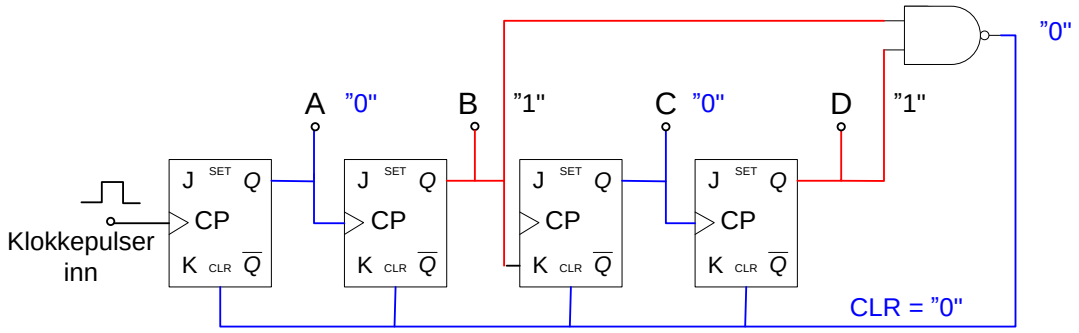
I denne 4-bit telleren bruker vi 4 stk. master-slave flip-flop (FF).
Alle J og K er satt til ”1” – det betyr at Q-utgangen endrer tilstand for hver fallende flanke på klokkeinngangen - (klokkesignalets transisjon fra ”1” til ”0”). Utgangen fra FF-A legges som klokkesignal inn på FF-B osv. Se figur under.
Tallet som leses ut fra denne telleren er organisert i ”reversert” rekkefølge – dvs. D holder ”høyeste bit” – mens A holder ”laveste bit”.
Tallene som denne binærtallet kan håndtere går fra 0000 (desimal 0) til 1111 (desimal 15). Den 16. klokkepulsen (puls #16) vil forsøke å øke tallet til 00001 (desimal 16) – imidlertid har vi ikke en ekstra flip-flop som kan holde denne siste ”1” – og vi sitter igjen med 0000 – som er tilstanden hvor vi startet.
La oss i detalj se hva som skjer i denne telleren når den skifter fra 1111 til 0000. Vi ser av koplingen at utgangen fra en flip-flop brukes som klokkesignal for neste. A klokker B – som klokker C osv. Vi ser at tilstandsendringene ”ripler” nedover rekken av flip-flop - fra A til D. Derfor kalles dette en rippelteller. - En asynkron krets.



Alle J og K forutsettes satt til ”1”

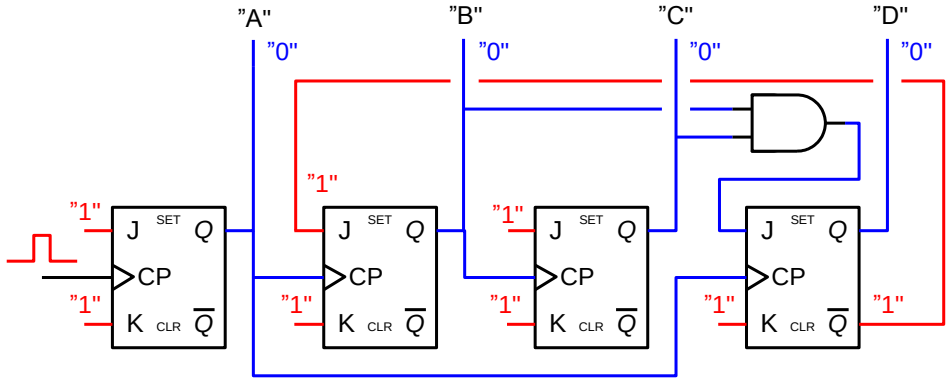
Dekadeteller

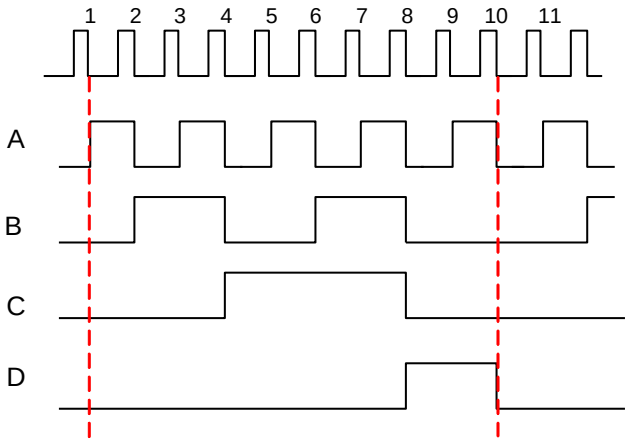
Vi vil lage en krets som teller til 10 - etter puls #10 vil vi at kretsen returnerer tilbake til 0.
Desimaltallet 10 vil på binært format leses som 0101 på vår teller. (reversert format)
(Normalt vil tallet se slik ut 1010 – med LSB – minst sign. bit først)
.
Vi må lage en krets som returnerer til ”0” når denne bitkombinasjonen oppstår. Dette lar seg lett realisere med en kopling av fire flip-flop som vist under:



NAND-porten får tilført signalene fra FF-B og FF-D. Når både B og D er ”1” sendes et CLR-signal til alle FF- elementene.
Denne koplingen er ikke trygg –. Vi ser at når B eller D går fra ”1” til ”0” vil CLR-signalet forsvinne – men vi kan ikke vite om alle FF er satt til ”0”.

Vi bruker noen ekstra forbindelser og får en ”trygg” dekadeteller (BCD)

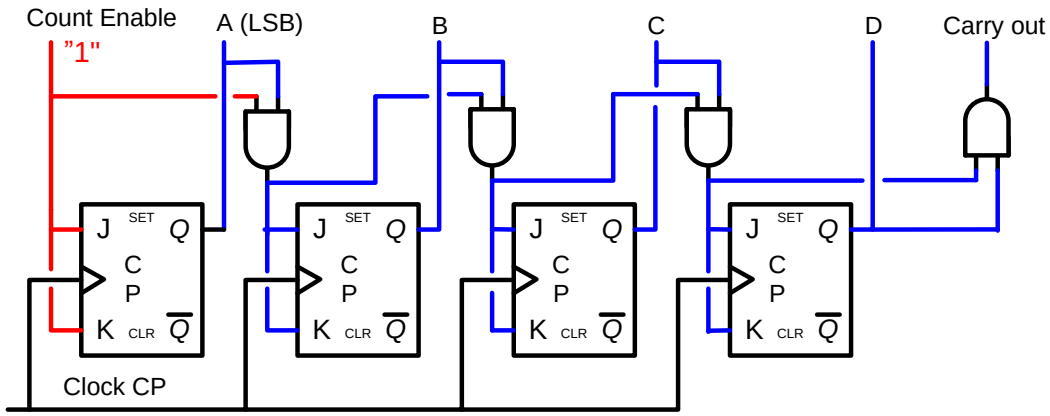




Denne koplingen sikrer en trygg overgang etter 10 pulser. J og K som ikke er i bruk legges til “1”. Puls # 10 (neg. flanke) setter telleren tilbake til 0000.
Se transisjonsdiagrammet til venstre.

Synkronteller

På figuren under har vi tegnet opp en 4 bit binær synkronteller. Dvs. En teller hvor alle FF klokkes samtidig – vi har felles klokkesignal.



Kretsen kan kort forklares slik :
A forandrer tilstand ved hver klokkepuls – fordi J = K = 1
B forandrer tilstand bare når A = 1 (J = K = A)
C forandrer tilstand bare når A = B = 1
D forandrer tilstand bare når A = B = C = 1

J	K	Q
0	0	Q
1	0	1
0	1	0
1	1	Q'

Tellere brukes til –

Måling av frekvens - frekvenstellere

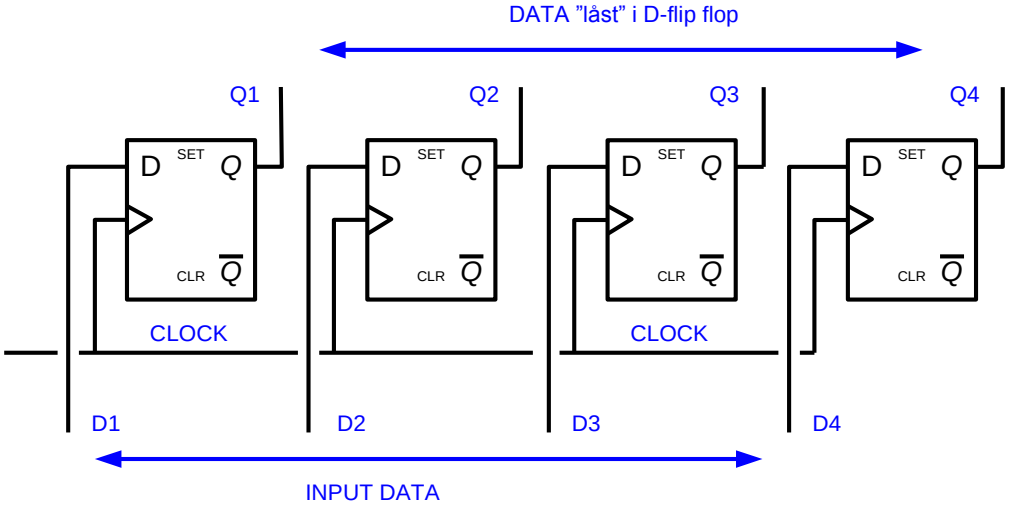
Måling av tid - klokkekretser

- som gir grunnlag for måling av :
- avstand
- hastighet
- osv

Registre

Memory register

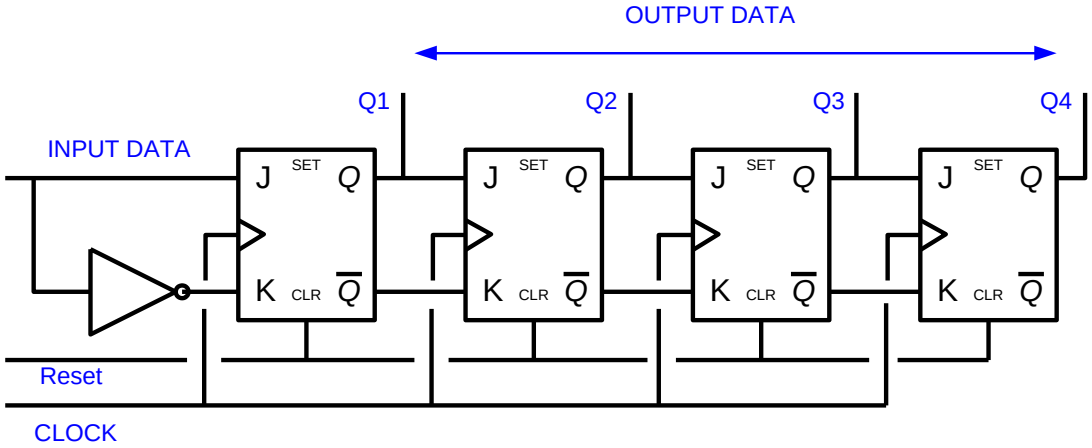
Skal et datasystemet huske binære tallverdier må vi ha en form for datahukommelse. Ved hjelp av ”flip flop” av D-type (master slave) kan vi konstruere en slik hukommelse. De databit som ligger på D-inngangene når klokkesignalet går fra ”1” til ”0”. (negativ flanke) vil bli låst til Q-utgangene.
Så lenge klokkesignalet ligger statisk høy eller lav vil endringer på D-inngangene ikke forandre tilstanden på Q.
Tegningen under viser sammenstillingen.



Logisk ”0” eller ”1” på ”Dn” flyttes til utgangen ”Qn” når klokka skifter fra ”1” til ”0” (negativ flanke)

Skiftregister (shift register)

Med en sammenstilling av 4 stk. JK- ”flip flop” som vist på figuren kan vi klokke en seriell datastrøm (på 4 bit) inn i et register - hvoretter vi kan ta dataene ut i parallell form etter 4 klokkepulser.



Dette avslutter introduksjonen til digitale systemer. For ytterligere informasjon om henvises til biblioteket, www og kurs ved IFI.

Historien bak IKT (Informasjons- og Kommunikasjons Teknologien)
- noen ”milepæler” de første 20 år

1958 Første integrerte krets konstruert av Jack Kilby (Texas Instruments)
- 12 komponenter på samme brikke. (Kilby får Nobel-prisen i 2000)

1958 Programmeringsspråket ALGOL (Algebraic Language) introdusert
1958 Seymour Cray bygger den første transistoriserte ”super computer”, CDC1604
1959 IBM kommer med sine første transistoriserte datamaskiner; 1620 og 1790
1960 Den første ”mini computer” utviklet av DEC. – PDP-1
1961 Første TTL produsert av Fairchild og Texas Instruments ,
1963 Første CMOS IC produsert av RCA
1964 Texas Instruments introduserer 74xxx serien for standardisert TTL
1966 Første bipolare Emitter Coupled Logic (ECL) produsert av Motorola
1966 Første ”Single transistor DRAM” produsert av IBM
1967 Norsk Data stiftes. Produserer sin første NORD-1 datamaskin basert på TTL.
1965-67 SIMULA utvikles av Ole Johan Dahl og Kristen Nygaard.
1970 Første kommersielle Dynamiske RAM (DRAM) – 1KBits.
1971 Verdens første mikroprosessor - 4 bit - Intel 4004 - 2300 transistorer -100kHz
1972 Utvider 4004 prosessoren til 8 bit - Intel 8008, klokke på 200kHz
(1972 Første hovedfagsoppgave i elektronikk på fysisk institutt, UiO som benytter en mikroprosessor – Intel 4004)
1972 Norsk Data leverer NORD-5. Verdens første 32 bit ”Super mini” datamaskin. Operativsystemet NORD-1 TSS, Multilingual Time Sharing System, blir utviklet - oppfattes den gang som verdens mest avanserte operativsystem
1974 Første kommersielle mikroprosessor - Intel 8080, 6000 transistorer - 2MHz NMOS-teknologi . Brukt i Altair computer (CP/M operativsystem, G.Kildall)
1975 Cray-1 supercomputer introdusert (Seymour Cray har opprettet eget firma)
1975 Microsoft etablert av Bill Gates og Paul Allen. Skriver software til MITS/Altair
1978 Intel 8088/8086 med 29000 transistorer, 5 -10 MHz klokke ble valgt av IBM som prosessor i IBM PC. (Microsoft skriver MS-DOS - operativsystemet for IBM-PC)
1978 Halvlederindustrien har etter 20 år en omsetning på 10 milliarder dollar.

I årene etter har utviklingen i omsetning og transistortetthet fulgt ”Moors lov” - dvs. tettheten av komponenter på en mikrochip fordobles hver 18. mnd.