

1 Teknologi-mapping

- a) Siden funksjonen T er på SOP-form, kan vi lage kretsen basert på NOR-porter hvis vi tar utgangspunkt i den inverterte til \bar{T} . Siste nivå i funksjonen T kan da realiseres som en 3-inngangs NOR-port (som kan splittes opp i to 2-inngangs porter):

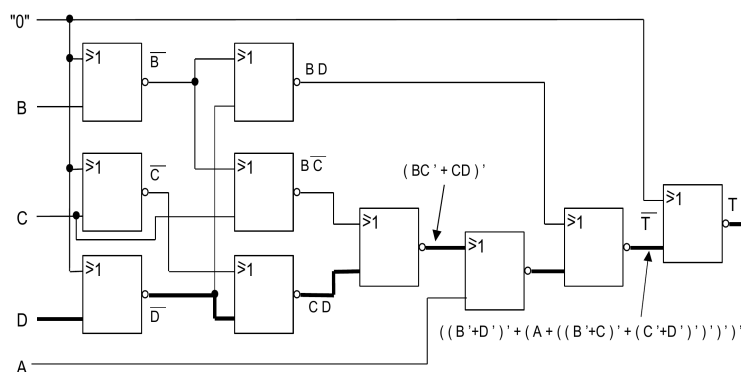
$$\bar{T} = \overline{BD + \bar{A}\bar{B}\bar{C} + \bar{A}CD}$$

Deretter fortsetter vi å jobbe med underliggende nivåer som er produkter av inngangsvariabler. Ved å bruke DeMorgans teorem kan disse realiseres som NOR-porter. Legg merke til at to siste leddene er faktorisert i uttrykket nedenfor for lettere å kunne realisere funksjonen med 2-inngangs porter:

$$\bar{T} = \overline{BD} + \overline{\bar{A}(\bar{B}\bar{C} + CD)} = \overline{\bar{B} + \bar{D}} + \overline{A + (\bar{B}\bar{C} + CD)}$$

Dette utføres rekursivt med alle nivåer inntil vi får en krets realisert fullstendig med 2-inngangs NOR-porter (legg merke til at dobbeltinvertering ikke er nødvendig, men er tatt med her for å vise overgangen mellom f.eks. BD og $\overline{\bar{B} + \bar{D}}$, som er identiske uttrykk):

$$\bar{T} = \overline{\bar{B} + \bar{D}} + \overline{A + (\bar{B}\bar{C} + CD)} = \overline{\bar{B} + \bar{D}} + \overline{A + \overline{\bar{B} + \bar{C}} + \overline{\bar{C} + \bar{D}}}$$



Figur 1: Realisering av funksjonen T med NOR porter

Realiseringen av T er vist i Figur 1. Legg merke til hvordan NOR-porter er brukt som inverterte. Et alternativ er å forbinde begge inngangene til samme signal for invertering, da ville vi ha spart det eksplisitte «0»-signalet. **Kommentar:** I denne figuren ble europeiske IEC-symboler brukt.

- b) Vi har benyttet ti NOR-porter. Hver av disse krever fire transistorer. Løsningens kostnad er følgelig 40 transistorer.
- c) Kritisk sti gjennom kretsen er markert med tykk strek i Figur 1. Denne går gjennom seks NOR-kretser. Hver av disse har en portforsinkelse på 1,4 ns. Total forsinkelsen er følgelig 8,4 ns.

2 Design og analyse av kombinatoriske kretser

a) Utvikling av logisk funksjon

Hver linje i spesifikasjonen gir et produktledd som deretter må summeres sammen. Dette gir oss:

$$Lys = A\bar{B}C + AB + B\bar{C}$$

Denne kan vi forenkle ved manipulasjon. Vi starter med å utvide de to siste leddene slik at vi får funksjonen på kanonisk form:

$$Lys = A\bar{B}C + AB\bar{C} + ABC + \bar{A}B\bar{C} + AB\bar{C}$$

Andre og fjerde ledd er like, så en av disse kan vi fjerne.

$$Lys = A\bar{B}C + ABC + \bar{A}B\bar{C} + AB\bar{C}$$

Nå kan vi trekke ut AC fra de to første leddene og $B\bar{C}$ fra de to siste:

$$Lys = AC(\bar{B} + B) + (\bar{A} + A)B\bar{C}$$

Parentesene kan nå fjernes og vi står igjen med:

$$Lys = AC + B\bar{C}$$

Dette er funksjonen til en multiplekser (selector) med C som valgsignal (se figur 5.13 i Gajski).

b) Adderer

I en CLA-adderer benyttes ekstra porter til å forhåndsregne menteforplantningen. Den har derfor høyere kompleksitet enn en ripple-carry adderer. Den er imidlertid vesentlig raskere.

c) Dekoder

I en 2-4 dekode er utgang C0 høy når inngangene tar verdien 00 og lav ellers. Tilsvarende er C1 høy når inngangene tar verdien 01 og lav ellers, osv. For en komplett løsning må vi sette opp sannhetstabell for hver enkelt funksjon i hver enkelt krets, og se hvilken krets som tilsvarer en 2-4 dekode. For løsning b1 ser vi imidlertid at C3 ikke er høy når inngangene tar verdien 11, følgelig kan dette ikke være en 2-4 dekode. Av de to gjenstående er det bare realiseringen av C1 som er forskjellig. Denne skal være høy når $A1=0$ og $A0=1$. Dette er tilfelle for b3 men ikke for b2.

d) PLA

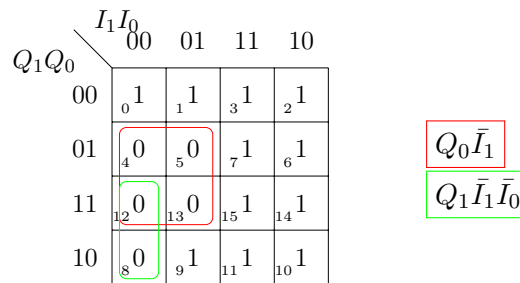
PLA-realiserings av følgende likninger:

$$U_1 = \bar{Q}_1\bar{Q}_0 + Q_0I_1 + Q_1\bar{Q}_0I_0 + Q_1\bar{Q}_0I_1$$

$$U_0 = \bar{Q}_1\bar{Q}_0\bar{I}_1 + \bar{Q}_1Q_0I_1 + \bar{Q}_1\bar{Q}_0I_0 + Q_1\bar{Q}_0\bar{I}_1\bar{I}_0$$

Vi lager Karnaughdiagram for de to likningene. Ettersom målet er å benytte færrest mulig OG- termer, må vi her vurdere hvilken forenkling som gir dette. Vanligvis grupperer vi 1-mintermer (enere i diagrammet), men siden vi har tilgjengelig invertering på utgangen, så kan vi også gruppere 0-mintermer (nullere i diagrammet). Invertering av en utgangsfunksjon oppnås ved å koble den andre inngangen på tilhørende XOR-port til 1 (se sannhetstabell for XOR-port).

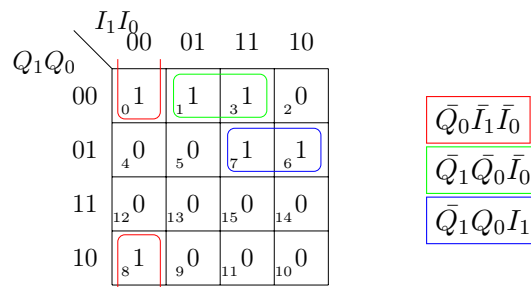
U_1 :



Her ser vi at implementering med 0-mintermer (makstermer) gir to OG-termer. Den alternative løsningen med 1-mintermer ville gitt tre OG-termer. Vi får altså:

$$U_1 = \overline{(Q_0\bar{I}_1 + Q_1\bar{I}_1\bar{I}_0)}$$

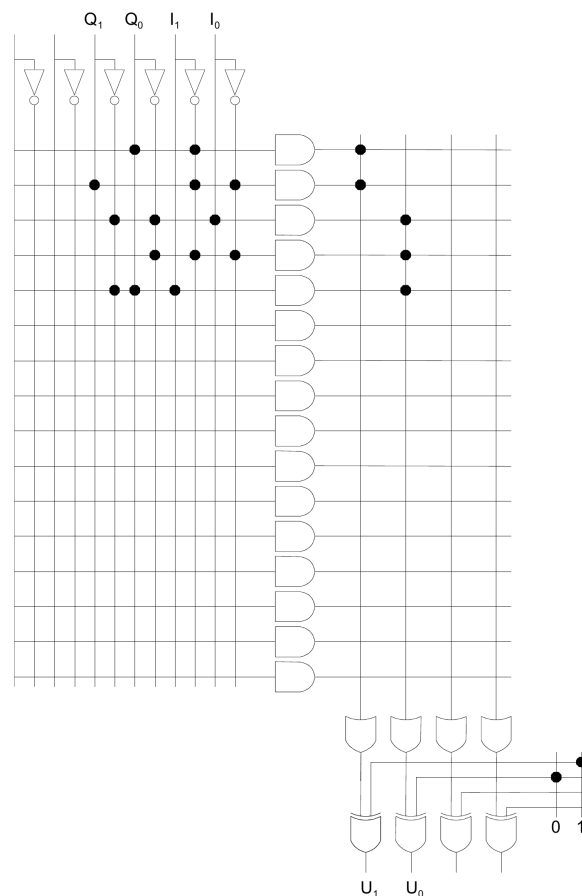
U_0 :



Her ser vi at implementering med 1-mintermer gir tre OG-termer. Den alternative løsningen med 0-mintermer (makstermer) ville gitt fire OG-termer. Vi får altså:

$$U_0 = \bar{Q}_0\bar{I}_1\bar{I}_0 + \bar{Q}_1\bar{Q}_0\bar{I}_0 + \bar{Q}_1Q_0I_1$$

I Figur 2 er de ulike OG-termene merket av i den øvre venstre delen av figuren. ELLER-termene som inngår i den enkelte funksjon er avmerket i den øvre høyre delen av figuren. Legg også merke til at vi inverterer U_1 men ikke U_0 ved å koble henholdsvis en ener og en nuller inn på XOR- portene nederst til høyre i figuren.



Figur 2: PLA med realisering av kombinatorikk

e) Kombinatorisk kretsdesign på bloknivå

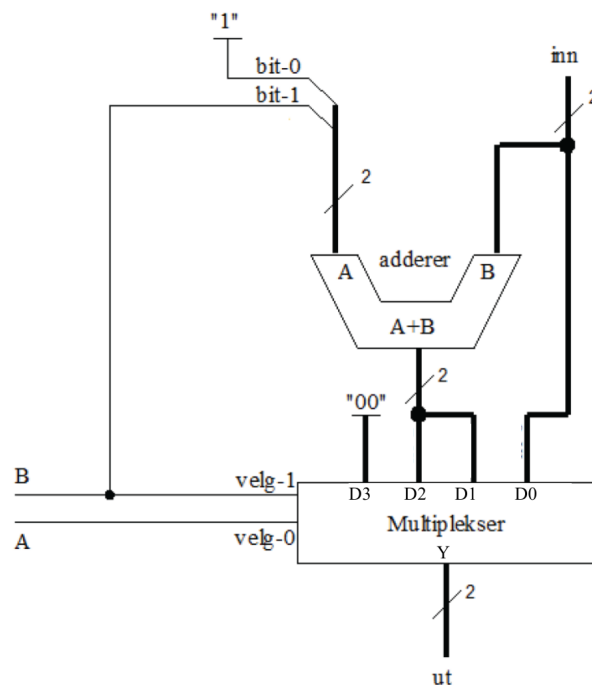
Vi kan starte designprosessen med å dele oppgaven i mindre deloppgaver. Kretsen vår skal kunne

- Trekke 1 fra inn ved hjelp av en $2s$ -komplement addisjonskrets.
- Legge 1 til inn ved hjelp av en $2s$ -komplement addisjonskrets.
- Velge en av fire mulige svar ved hjelp av en multiplekser

Det å trekke fra 1 er det samme som å legge til -1 . Siden vi opererer med 2 bit brede $2s$ -komplement tall så svarer det til å legge til 11. Det vil altså si at vi enten skal legge til 01 (inn + 1) eller legge til 11 (inn - 1). Det er med andre ord bare det mest signifikante av de to bittene som er forskjellige i de to addisjonene. Samtidig har vi at styresignalet $B=0$ når vi skal legge til '01' og $B=1$ når vi skal legge til '11'. Følgelig kan vi la B være verdien av det mest signifikante bittet som skal legges til, og koble det minst signifikante bittet direkte til 1.

Videre kan vi la styresignalene A og B styre multiplekseren slik at den slipper signalet inn rett igjennom, slipper resultatet av addisjonen igjennom, eller slipper verdien '00' igjennom. Resultatet av addisjonen må vi rute inn på to av inngangene på multiplekseren siden det er to kombinasjoner av styresignalene som benytter seg av dette resultatet.

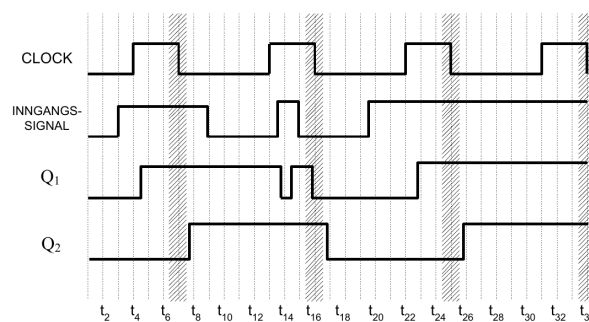
Figur 3 viser en mulig kretsløsning på bloknivå.



Figur 3: Kombinatorisk krets på blokknivå

3 Flanke- og nivåstyrte vipper

- a) Vi ser fra figuren at pulstog Q_1 skifter fra høy til lav, og tilbake til høy, mens klokken er høy. Dette pulstøget tilhører derfor en D-latch. Pulstog Q_2 skifter nivå kun i tilknytning til den fallende klokke-flanken, og hører derfor til en D-vippe (flankestyrt). Plasseringen av invertereren gjør at slave-låsen leder når CLOCK er lav. Følgelig vil utgangen Q endre seg i det CLOCK skifter fra høy til lav, og vi får en vippe som er styrt av den fallende (eller negative) flanken.

Figur 4: De skraverte områdene er tidsintervallet $T_{\text{oppsett}}-T_{\text{holde}}$

- b) $T_{\text{oppsett}}-T_{\text{holde}}$ er et tidsintervall som omslutter den aktive flanken til klokken. Hvis man endrer inngangssignalet i dette tidsrommet har man ikke kontroll på hvordan det vil innvirke på utgangsverdien til kretsen. Dette er vist i Figur 4.
- c) Tabell 1: Vi ser at nestetilstanden til D-vippen er lik inngangssignalet til vippen. Tabell 2: Hvis inngangssignalet T er 1, vil vippen skifte tilstand ved neste aktive klokkeflanke (toggle-vippe).

Q	Q _(next)	D
0	0	0
0	1	1
1	0	0
1	1	1

Tabell 1: Eksitasjonstabell for D-vippe

Q	Q _(next)	T
0	0	0
0	1	1
1	0	1
1	1	0

Tabell 2: Eksitasjonstabell for T-vippe

Tabell 3: Dersom vippen er i tilstanden 0, er det S som bestemmer om vippen skal settes til 1 eller forbli 0. S kalles derfor set-inngangen. Dersom vippen er i tilstand 1, er det R som bestemmer om vippen skal resettes (settes til 0). R kalles derfor reset-inngangen til vippen. Merk at det ikke finnes noen rad i tabellen der både S og R er 1, siden dette er ulovlig for SR-vippen.

Tabell 4 Dersom vippen er i tilstanden 0, er det J som bestemmer om vippen skal settes til 1 eller forbli 0. J kalles derfor set-inngangen. Dersom vippen er i tilstand 1, er det K som bestemmer om vippen skal resettes (settes til 0). K kalles derfor reset-inngangen til vippen. Merk at dersom både J og K er 1, så vil vippen skifte tilstand ved neste aktive klokkeflanke (toggle).

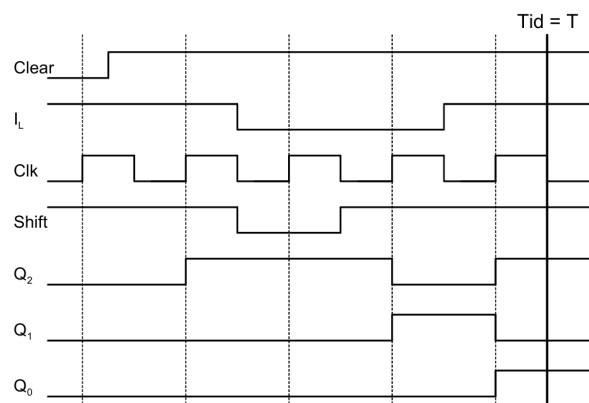
- d) Ved tiden T er $Q_2=1$, $Q_1=0$ og $Q_0=1$. Skiftregisteret svarer til figur 7.4 i Gajski. Se Figur 5 for oppførsel.

Q	Q _(next)	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

Tabell 3: Eksitasjonstabell for SR-vippe

Q	Q _(next)	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Tabell 4: Eksitasjonstabell for JK-vippe



Figur 5: Skiftregister med inngangsverdier