Del 11: Latcher og vipper

YNGVAR BERG

I. Innhold

KONVENSJONELLE latcher og vipper i CMOS blir gjennomgått. Latcher som styres av to klokkefaser og klokkepulser blir diskutert. Lacher og vipper med reset, set og enable blir presentert. Latcher med logikk introduseres og differensielle vipper presenteres. Til slutt blir ekte en-fase latcher og vipper introdusert. Alle henvisninger til figurer er relevant for Weste & Harris [1].

1. *Innhold.*
2. *Konvensjonelle CMOS latcher.* Kapittel 7.3.1 side 402 - 405.
3. *Konvensjonelle CMOS vipper.* Kapittel 7.3.2 side 405 - 407.
4. *Latcher som styres av klokkepulser.* Kapittel 7.3.3 side 407 - 408.
5. *Latcher og vipper som kan resettes.* Kapittel 7.3.4 side 408 - 409.
6. *Latcher og vipper som kan enables.* Kapittel 7.3.5 side 410.
7. *Latcher med logikk.* Kapittel 7.3.6 side 410 - 411.
8. *Klass semidynamisk vippe (SDFF).* Kapittel 7.3.6 side 411 - 412.
9. *Differensielle vipper.* Kapittel 7.3.8 side 412 - 413.
10. *Ekte en-fase (TSPC) latcher og vipper.* Kapittel 7.3.9 side 414.

II. Konvensjonelle CMOS latcher

(Kapittel 7.3.1 side 402 - 405)

* *Terskelfall [2]. Pass transistor DC karakteristikk, Kapittel 2.5.6 side 101 - 102.*

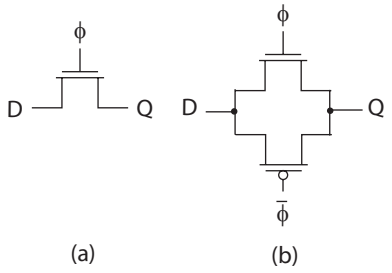

Fig. 1. *Dynamisk en-transistor- og transmisjonsport latch. (FIG7.17a og b)*

I prinsippet kan man lage en *latch* ved hjelp av en transistor som vist i Fig. 1 (a). Ideelt vil utgangen Q følge inngangen når ϕ er høy og holde verdien når ϕ er lav. I utgangspunktet er ofte få transistorer å forterekke framfor flere transistorer når man implementere en krets. For pass transistoren som latch vil det imidlertid være fire viktige begrensinger:

- Utgangen vil ikke svinge mellom GND og V_{DD}. Spesielt vil ikke en nMOS transistor kunne brukes til å drive en logisk høy verdi (V_{DD}) på grunn av terskelfall.

- Utgangen er *dynamisk*, dvs. utgangen vil være udrevet og flyte når kontrollsignalet (ϕ) er lavt. Dette kan medføre at utgangen endres slik at latchen ikke holder riktig verdi.

- Inngangen D driver direkte til en source/drain terminal på en transistor og ikke en gate terminal. Dette kan resultere i merkbar støy og gjør det vanskelig å prediktere forsinkelse i en krets.

- Tilstanden på den lagrede noden (Q) kan påvirkes av støy på utgangen (også Q), slik at lagringsnoden er utsatt for endringer fra utgangen.

I Fig. 1 (b) er det vist en transmisjonsport som en latch. Den vil fungere bedre enn pass transistoren, men vil også ha en begrensninger som latch. Ved å utvide fra pass transistor til transmisjonsport har vi bare fjernet den første begrensingen. Utvidelsen medfører også et behov for t invertert kontrollsignal $\overline{\phi}$.

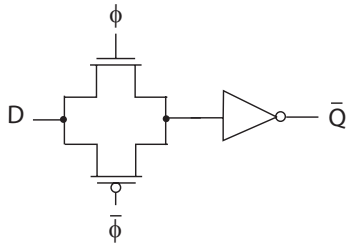

Fig. 2. *Dynamisk latch med transmisjonsport og inverter. (FIG7.17c)*

I Fig. 2 har vi lagt til en inverter på utgangen slik at utgangen blir invertert $\overline{Q}$ og dermed isolert latchens lagrede verdi fra utgangen. Latchen er da ikke utsatt for støy fra utgangen, men vil forsatt være dynamisk og inngang koblet til source/drain terminal på transistor.

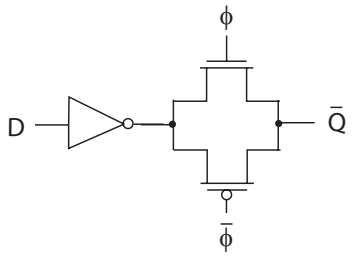

Fig. 3. *Dynamisk latch med inverter og transmisjonsport. (FIG7.17d)*

I Fig. 3 er det vist en latch med inverter og transmisjonsport. Vi har nå inngangen koblet til gate, men utgangen vil være utsatt for støy. En logisk ekvivalent krets som vist i Fig. 3, men med mindre arealbehov er en *klokket CMOS inverter* (C^2MOS) som er vist i Fig. 4. C^2MOS er noe tregere enn enn en inverter og en transmisjonsport fordi transistorene som styres av klokkesignalene aldri vil bidra i parallell. Det er derfor ikke vanlig å bruke klokket CMOS på inngangen til latchen.

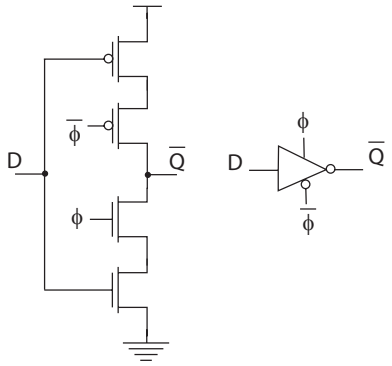

Fig. 4. *Klokket CMOS latch. (FIG7.18)*

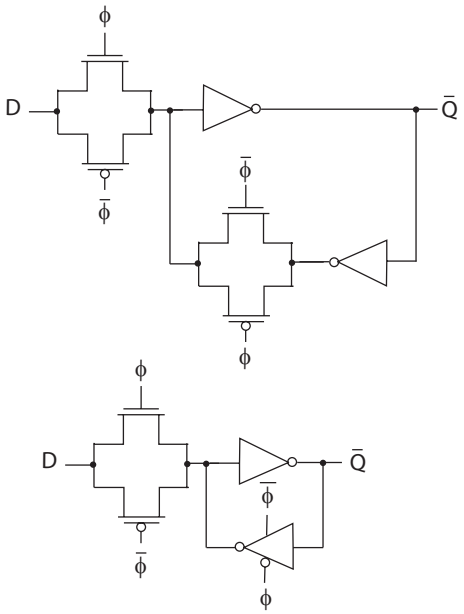

Fig. 5. *Statisk latch med transmisjonsport og inverter og tilbakekobling i motfase øverst. Tilbakekobling med C^2MOS nederst. (FIG7.17e)*

Ved å kombinere latchene i Fig. 2 og 3 og klokke transmisjonsportene i motfase får vi en *statisk latch* som vist i Fig. 5. Det som nå mangler er gate terminal innngang.

I Fig. 6 har latchen fått en inverter på inngangen og utgangen blir dermed Q på grunn av to inverteringer. Utgangen lastes av C^2MOS inverteren i tillegg til eksterne kretser. En raskere latch der lasten på utgangen er redusert er vist i Fig. 7. Dette er en latch som ikke har noen av de begrensinger som ble beskrevet for pass transistor latchen. Vi ser imidlertid at latchen har blitt relativt kompleks, som medfører økt tidsforsinkelse, effektforbruk og økt areal (utlegg). Enklere latcher med gode elektriske egenskaper baseres på latchen i Fig. 7 med forenklinger som øker latchens ytelse og reduserer arealbehovet.

En enkel forenkling av latchen i Fig. 7 er vist i Fig. 8, der C^2MOS inverteren er erstattet med en svak[1] inverter. Når latchen sampler (latcher inn) vil inngangsignalet via inngangsinverteren og transmisjonsporten overstyre tilbakekoblingen slik at utgangen Q får ny verdi. Når transmisjonsporten er skrudd AV vil tilbakekoblingen være tilstrekkelig sterk til å holde ver-

[1]Med svak inverter menes en inverter som leverer lite strøm på grunn av lite W/L forhold for transistorene.

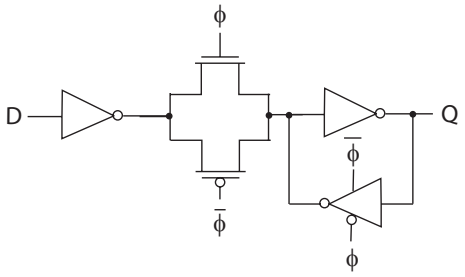

Fig. 6. *Statisk latch med inverter inmngang og utgang Q. (FIG7.17f)*

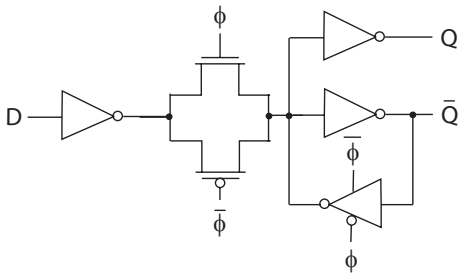

Fig. 7. *Statisk latch med utgang Q. (FIG7.17g)*

dien på Q. Denne latchen er derfor statisk.

A. Varianter av transparente latcher

Latchen som er vist i Fig. 9 brukes typisk i registre eller FGPA (Field Programmable Gate Array) kretser. Latchen styres av kontrollsignalene WR (WRite) og RD (ReaD). Legg merke til at inngangen er koblet til en pass transistor. Dette betyr at noden X i utgangspunktet ved latching ikke kan trekkes helt opp til logisk 1 (V_{DD}), men X vil eventuelt bli trukket helt opp til logisk 1 ved hjelp av inverteren i tilbakekoblingen. Latchene har vanligvis sammenkoblet utgang som forutsetter at bare en av mange latcher med felles utgang kan selekteres ved et gitt tidspunkt.

En annen variant av statisk latch er vist i Fig. 10. Inverteren til venstre brukes for å generere klokke invertert lokalt. Inngangen er koblet til source/drain terminaler på en transmisjonsport slik at man må være oppmerksom på at latchen kan være "tung" å drive for inngangen. Latchen kan utvides med en inverter før transmisjonsporten og dermed vil utgangen bli Q. Tilbakekoblingen er nå delvis klokket, dvs. X blir precharget til logisk 1 dersom $\overline{X} = 0$ (som betyr at $X = 1$) og trukket ned til 0 når $\overline{\phi} = 1$ og $\overline{X} = 1$ (som betyr at $X = 0$). Når latchen skal sample, dvs. når $\phi = 1$ må transmisjonsporten overstyre precharge i tilbakekoblingen, og derfor er pMOS transistoren i tilbakekoblingen svak.

B. Mål

Forstå hvordan konvensjonelle latcher i CMOS kan implementeres.

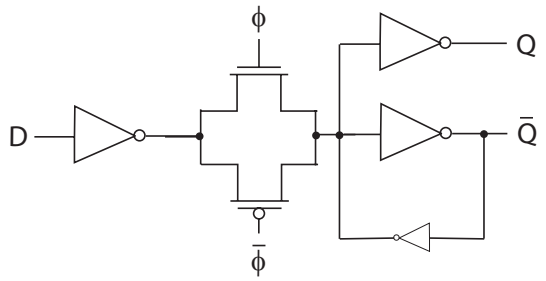

Fig. 8. *Statisk latch med utgang Q og svak uklokket tilbakekobling. (FIG7.17i)*

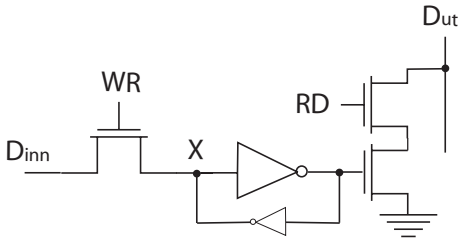

Fig. 9. *Statisk latch for FPGA (Field Programmable Gate Array). (FIG7.17j)*

C. Notater

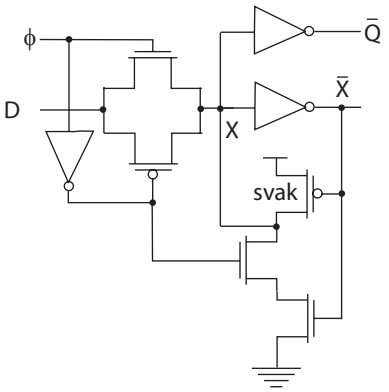

Fig. 10. *Statisk latch variant. (FIG7.17k)*

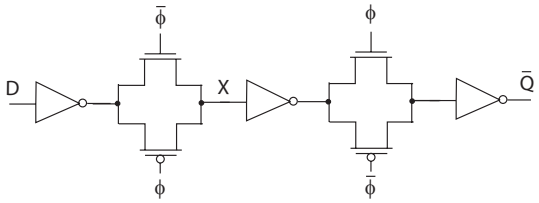

Fig. 11. *Dynamisk vippe. (FIG7.19a)*

En dynamisk vippe er vist i figur 11. Denne vippen er satt sammen av to dynamiske latcher som klokkes i motfase.

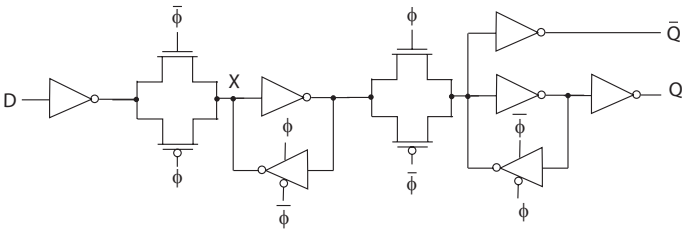

Fig. 12. *Statisk vippe. (FIG7.19b)*

En statisk vippe med to statiske latcher som er klokket i motfase er vist i Fig. 12. Denne vippen har både Q og $\overline{Q}$ utganger. Det er vanlig at vipper bare har en klokkeinngang ϕ og genererer invertert klokkesignal lokalt.

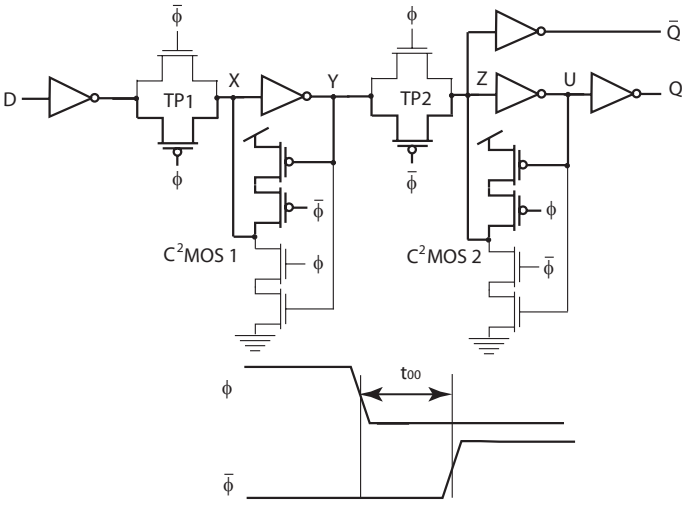

Fig. 13. *Statisk vippe ved negativ klokkeflanke og lokal generering av invertert klokke. (FIG7.19b)*

Ved lokal generering av invertert klokkesignal kan man få en liten forsinkelse for det inverterte klokkesignalet som vist i Fig. 13. Ved negativ klokkeflanke, dvs. ϕ skifter fra 1 til 0 vil det ta en viss tid t_{00} der begge klokkesignalene er lave. Signalveier som er markert med tykke linjer er da PÅ. Vi ser at den første transmisjonsporten $TP1$ er PÅ slik at inngangslatchen sampler inngangen. Tilbakekoblingen i inngangslatchen burde vært skrudd av, men vil i perioden t_{00} ha ett opptrekk som er PÅ. Dette opptrekket er egentlig bare PÅ når $Y = 0$ som betyr at

$X = 1$. I en situasjon der inngangen $D = 1$ får vi en konflikt i noden X fordi inngangen via en inverter og TP1 vil drive X til 0 mens tilbakekoblingen vil drive X til 1. Dette er bare et temporært problem fordi vi må forutsette at $\overline{\phi}$ endres til 1 før positiv klokkeflanke kommer. Etter t_{00} vil tilbakekoblingen skrus AV og nodene X og Y vil få riktig verdi drevet fra inngangen D. Problemet er mer betydelig enn man kan få inntrykk av ved bare å studere inngangslatchen i perioden t_{00}. Husk at utgangslatchen har samme klokkesignaler slik at i periden t_{00} vil transmisjonsporten for utgangslatchen $TP2$ også være feilaktig PÅ. Dette medfører at noden Z vil påvirkes av Y (direkte fra D) og via tilbakekoblingen i utgangslatchen C^2MOS 2. Vi ser at bare opptrekket i tilbakekoblingen er PÅ slik at tilbakekoblingen vil forsøke å precharge Z til 1. Den korrekte funksjonen til utgangslatchen er at $TP2$ er AV og tilbakekoblingen er PÅ. Vi ser at i perioden t_{00} er hele vippen transparent slik at D kan påvirke Q og $\overline{Q}$ direkte. En kritisk situasjon er når $Z = 0$ og $U = 1$ rett før t_{00} og $D = 1$ som betyr at Z drives mot 1 via Y og X fra D. I denne situasjonen er ikke tilbakekoblingen i utgangslatchen aktiv og Z kan derfor drives til 1, som igjen endrer U til 0 og bidrar til å holde $Z = 1$ feilaktig. Når perioden t_{00} er over vil $TP2$ stenge, men dette er for sent til å unngå en feilaktig endring av utgangene.

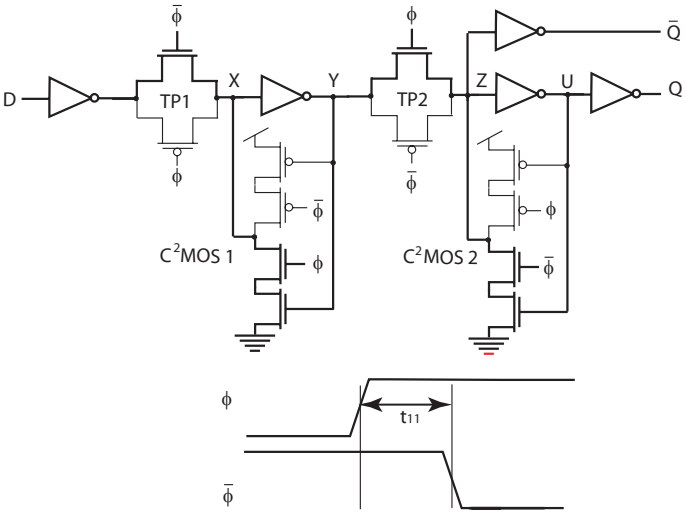

Fig. 14. *Statisk vippe ved positiv klokkeflanke og lokal generering av invertert klokke. (FIG7.19b)*

Vi får et tilsvarende problem ved positiv klokkeflanke som vist i Fig. 14. I perioden t_{11} vil begge klokkesignalene være høye slik at vippen blir temporært transparent. Riktig vippe funksjon er at inngangslatchen ikke sampler inngangen, men har en aktiv tilbakekobling. Dette betyr at $TP1$ skal være AV og C^2MOS 1 skal være PÅ. For utgangslatchen skal $TP2$ være PÅ og tilbakekoblingen C^2MOS 2 være av. I perioden t_{11} kan vi få en alvorlig situasjon dersom nodene X og Y endres på grunn av D og tilbakekoblingen C^2MOS 1 i inngangslatchen ikke kan overstyre $TP1$. I denne situasjonen blir vippen transparent.

En vanlig løsning på problemet med delvis transparente vipper er å bruke tofase ikke-overlappende klokker som vist i Fig. 15.

A. Mål

Forstå hvordan konvensjonelle vipper i CMOS kan implementeres.

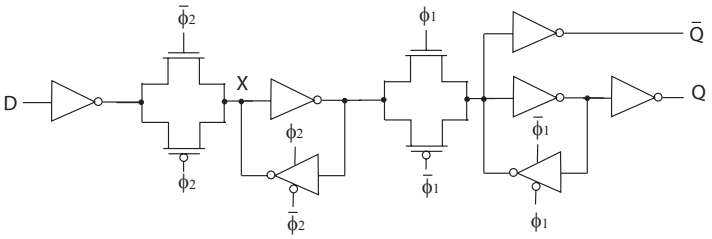

Fig. 15. *Statisk vippe med tofase ikke-overlappende klokker. (FIG7.21)*

B. Notater

IV. Latcher som styres av klokkepulser

(Kapittel 7.3.3 side 407 - 408)

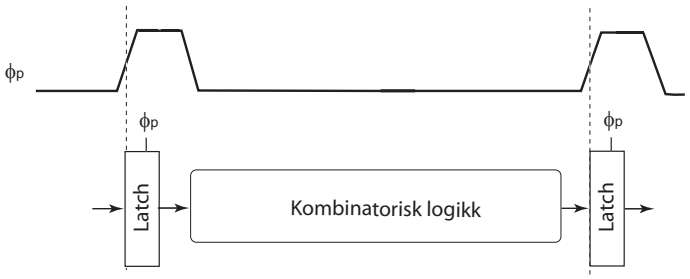

Fig. 16. *Latch som styres av klokkepuls.*

En latch som styres av klokkepulser minner om en konvensjonell transparent latch. Et slikt system er avhengig av forholdsvis stor tidsforsinkelse i kombinatorisk logikk mellom latchene som vist i Fig. 16. For at to latcher med kombinatorisk logikk mellom latchene ikke skal være transparent må det settes krav til tidsforsinkelse i den kombinatoriske logikken:

$$t_{cd} \quad \geq \quad t_{hold} - t_{ccq} + t_{pw}, \tag{1}$$

der t_{cd} er contamination (minimum) forsinkelse i kombinatorisk logikk, t_{hold} er hold tid for inngang fra negativ klokkeflanke, t_{ccq} er klokke til utgang contamination forsinkelse for latchen og t_{pw} er pulsbredden på klokkesignalet.

A. Puls generatorer

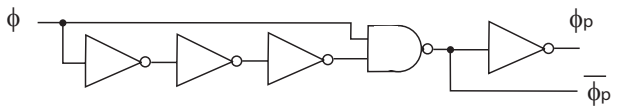

Fig. 17. *Enkel puls generator. (FIG7.22a)*

Med utgangspunkt i et klokke signal ϕ med dutycycle lik 50% kan vi generere en klokke med pulser $\leq 50\%$ som vist i Fig. 17. I dette tilfellet blir invertert klokkesignal også generert.

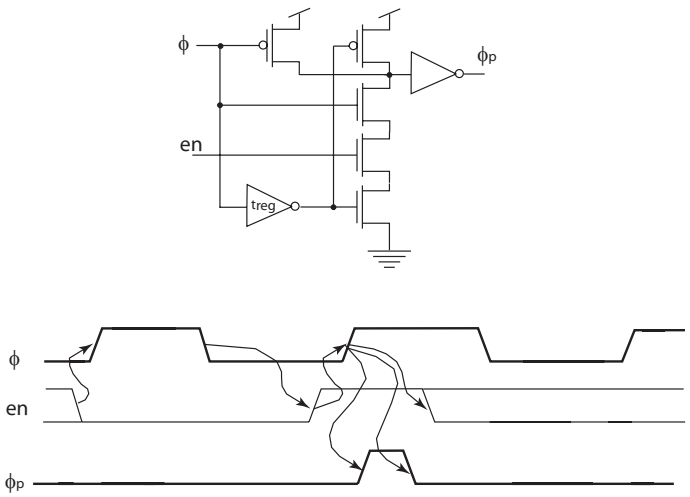

Fig. 18. *Puls generator. (FIG7.22b)*

En annen puls generator er vist i Fig. 18. Denne puls generatoren genererer pulser med meget kort bredde.

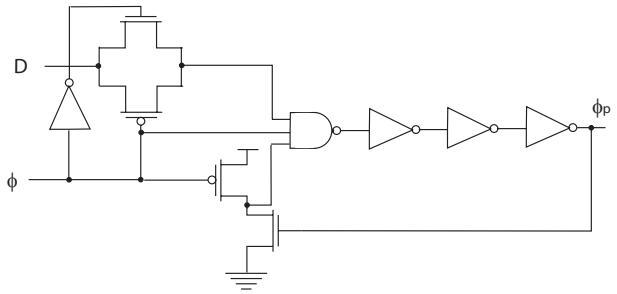

Fig. 19. *Puls generator. (FIG7.22d)*

En tredje pulsgenerator med betydelig bredere pulser er vist i Fig. 19.

Ulike pulsgeneratorer med forskjellig pulsbredder passer til ulike spesielle latcher.

B. Puls latch

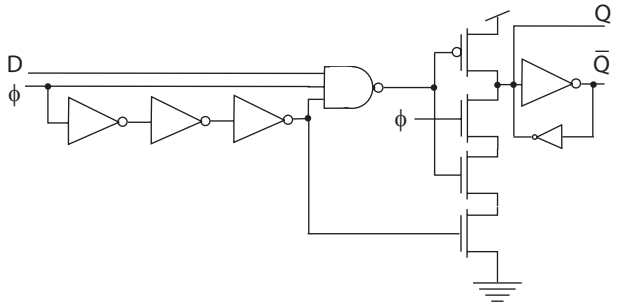

Fig. 20. *Partovi puls latch. (FIG7.23)*

Et eksempel på en latch som er styrt av klokkepulser er vist i Fig. 20. Dette er en såkalt Partovi puls latch som har pulsgeneratoren innebygd i selve latchen.

C. Mål

Forstå hvordan latcher som styres av klokkepulser kan implementeres.

D. Notater

V. Latcher og vipper som kan resettes
(Kapittel 7.3.4 side 408 - 409)

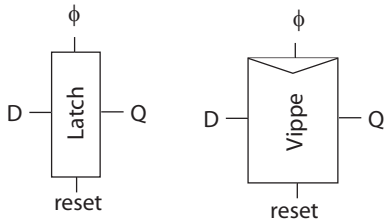

Fig. 21. *Symboler for latch og vippe med reset signal. (FIG7.24)*

Det er praktisk å kunne benytte et reset signal slik at tilstanden til et sekvenseringselement er kjent ved oppstart. Symboler for latch og vippe med reset signal er vist i Fig. 21. Det er to typer av reset:

- *Synkron reset.* Synkrone reset signaler må være stabile for setup- og hold tid ved klokkeflanker.

- *Asynkron reset.* Asynkrone reset signaler resetter et element uavhengig av klokkesignaler.

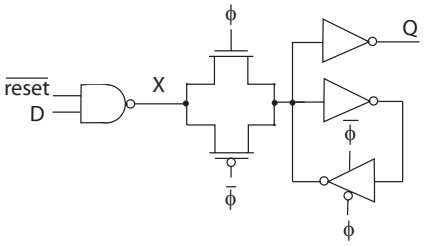

Fig. 22. *Synkron latch med reset signal. (FIG7.24)*

En latch med synkron reset er vist i Fig. 22. Som kjent er ikke latchen følsom for inngangen når $\phi = 0$. NAND porten på inngangen av latchen vil slippe gjennom D når $\overline{reset}$ er 1, vi har for en 2inngangs NAND port (NAND2) $X = \overline{D \cdot \overline{reset}}$ som gitt at $\overline{reset} = 1$ kan forenkles til $X = \overline{D}$. Når $\overline{reset} = 0$ kan uttrykket for NAND porten forenkles til $X = 1$. Når transmisjonsporten åpner for $\phi = 1$ vil latchen sample inn enten $\overline{D}$ eller 1. I det siste tilfellet skal latchen resettes slik at utgangen $Q = 0$ uavhengig av D. Vi legger merke til at latchen ikke resettes før $\phi = 1$.

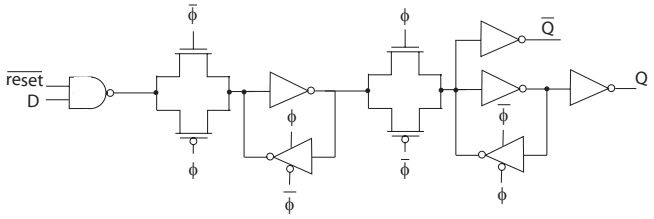

Fig. 23. *Synkron vippe med reset signal. (FIG7.24)*

En vippe med synkron reset er vist i Fig. 23. For inngangslatchen i vippen gjelder samme argumentasjon som for synkron reset av latch, men der inngangslatchen er klokket i motfase[2].

[2]Inngangslatchen kan resettes når $\phi = 0$.

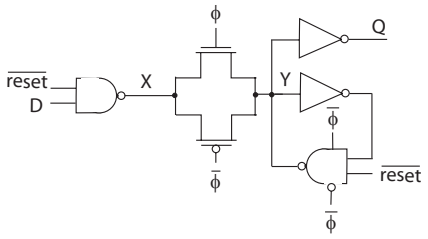

Fig. 24. *Asynkron latch med reset signal. (FIG7.24)*

En latch med asynkron reset er vist i Fig. 24. NAND porten på inngangen fungerer som beskrevet for latch med synkron reset, dvs. Q via Y og X settes til 0 når $reset = 1$ og $\phi = 1$. Latchen blir da resatt via transmisjonsporten på inngangen. Det er i tillegg plassert en dynamisk NAND port i tilbakekoblingen slik at noden Y kan settes til 1, og dermed utgangen Q settes til 0 når $reset = 1$ og $\phi = 0$. Dette betyr at utgangen Q settes til 0 når $reset = 1$ uavhengig av D og ϕ.

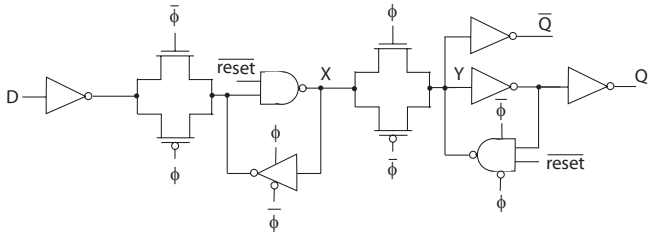

Fig. 25. *Aynkron vippe med reset signal. (FIG7.24)*

En vippe med asynkron reset er vist i FIG. 25. Inngangslatchen vil presse noden X til 1 når $reset = 1$ uavhengig av D, ϕ og (tidligere) verdi på X. Når $\phi = 0$ vil noden Y få verdien 1. Vi har da en situasjon der noden Y blir satt til 1 fra X via transmisjonsporten når $\phi = 1$ eller fra den dynamiske NAND porten når $\phi = 0$. Dette betyr at Y blir resatt til 1 uavhengig av ϕ og Q blir resatt til 1. Legg merke til at denne vippen resettes til 1 når $reset = 1$.

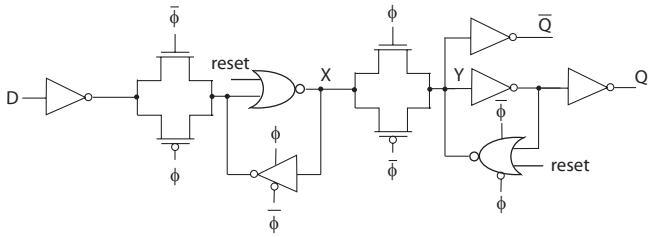

Fig. 26. *Aynkron vippe med reset signal.*

Vi kunne ha byttet ut NAND portene med NOR porter og $\overline{reset}$ med $reset$, som vist i Fig. 26, slik at noden Y ble resatt til 0 for å få resatt utgangen Q til 0 når $reset = 1$.

A. Vippe med asynkron set og reset

I Fig. 27 er en vippe med asynkron set og reset vist. Kretsen benytter to signaler set og $reset$ til å sette vippen i to ulike tilstander. Inngangslatchen har $\overline{set}$ signal som styrer NAND porten som setter noden X til 1 når $set = 1$. For utgangslatchen vil $\overline{set}$ signalet sette noden Y lik 1. $\overline{set}$ signalet setter utgangen på C^2MOS NAND porten i tilbakekoblingen i inngangslatchen

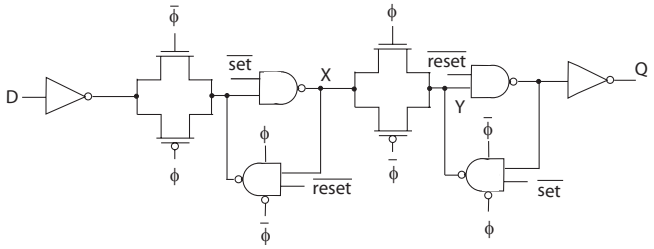

Fig. 27. *Vippe med asynkron reset og set signal.*

til 1 når $reset = 1$ samtidig som NAND porten i utgangslatchen sette inngangen til inverteren til 1 og dermed utgangen Q til 0.

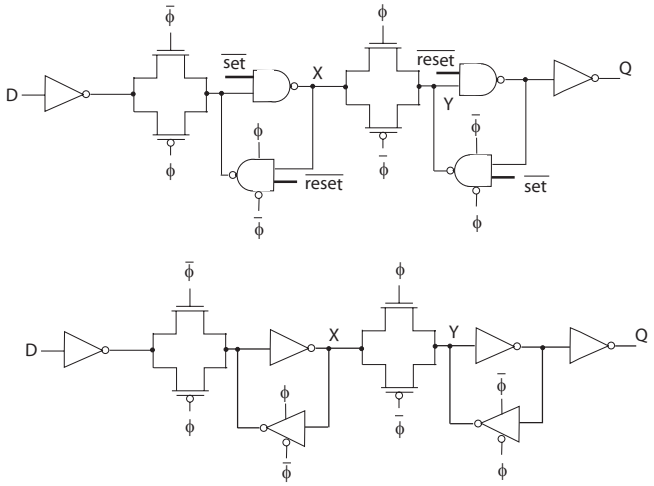

Fig. 28. *Vippe med asynkron reset og set signal.* $Set = reset = 0$

Vippen med $set = reset = 0$ er vist øverst i figur 28. For alle NAND portene vil det være en av inngangene som er 1, dvs. $\overline{set} = \overline{reset} = 1$. Forenklet port og logisk ekvivalent, men ikke elektrisk, ekvivalent, er en inverter som vist i den nederste kretsen i Fig. 28.

Vippen i reset funksjon er vist i Fig. 29 øverst. I dette tilfellet forutsetter vi at det andre kontrollsignalet $set = 0$. For inngangslatchen vil da utgangen på C^2MOS NAND porten i tilbakekoblingen bli satt til 1 slik at den andre NAND porten i inngangslatchen vil sette X til 0. Dette betyr at inngangslatchen vil bli resatt til 0 som er tilsvarende som om vi samplet inn 0 fra inngangen. For utgangslatchen vil NAND porten med $\overline{reset}$ som inngang sette inngangen til inverteren til 1 og dermed utgangen Q til 0. C^2MOS NAND porten i tilbakekoblingen i utgangslatchen vil sørge for at Y blir lik 0 (som er samme verdi som X). Dersom kontrollsignalet $reset$ settes til 0 etter at kretsen er korrekt resatt vil vippen være i tilstanden vist i Fig. 29 nederst[3] inntil vippen eventuelt sampler inn en ny verdi $D = 1$ når $\phi = 0$, eller vippen settes til 1 ved hjelp av kontrollsignalet set.

Vippen i set funksjon er vist i Fig. 30 øverst. I dette tilfellet forutsetter vi at det andre kontrolsignalet $reset = 0$. Inngangslatchen vil sette noden X til 1 som igjen vil sette utgangen til C^2MOS NAND porten i tilbakekoblingen til 0. Denne verdien vil holde seg lik 0 gjennom forenklet logisk ekvivalent vist for inngangslatchen i Fig. 30 nederst. For utgangslatchen vil

[3]Kretsen nederst er logisk ekvivalent når $set = reset = 0$, men ikke elektrisk ekvivalent.

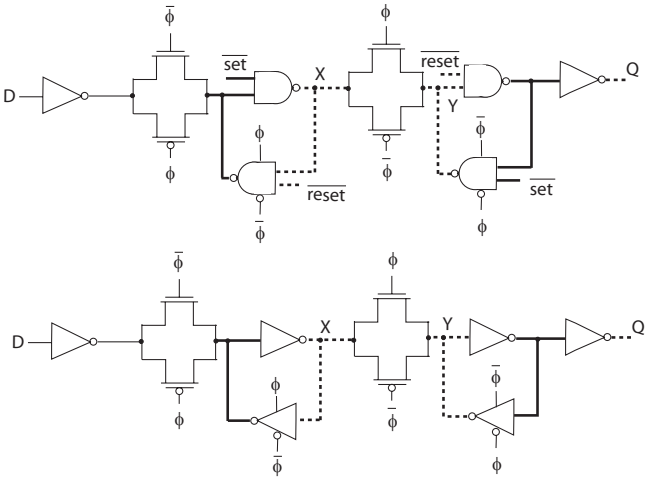

Fig. 29. *Vippe med asynkron reset og set signal.* $Reset = 1$

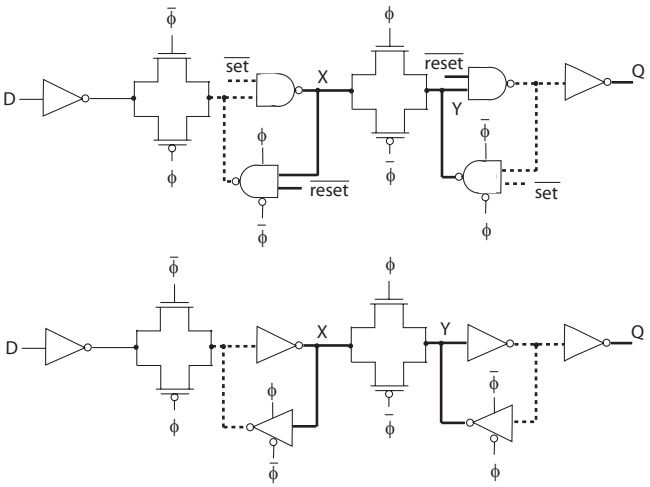

Fig. 30. *Vippe med asynkron reset og set signal.* $Set = 1$

C^2MOS NAND porten i tilbakekoblingen sørge for at $Y = 1$ som vil sette inngangen til utgangsinverteren til 0 og dermed blir utgangen Q lik 1. Vippen nederst i figuren er logisk ekvivalent inntil inngangslatchen sampler inn ny verdi $D = 0$ når $\phi = 0$, eller vippen resettes ved hjelp av kontrolsignalet $reset$.

B. Detaljer

Vi har nå forutsatt at vippen kan ha tre ulike modi:

- **Vippe.** $set = reset = 0$. Vippen fungerer som en vanlig vippe som vist i Fig. 28 nederst.

- **Resett til 0.** $reset = 1$ og $set = 0$. Vippen resettes til 0, dvs. både utgangen og noden X resettes til 0. Når reset signalet endres til 0 vil kretsen operere som kretsenekvivalenten vist nederst i Fig. 29.

- **Sett til 1.** $set = 1$ og $reset = 0$. Vippen resettes til 1, dvs. både utgangen og noden X resettes til 1. Når reset signalet endres til 0 vil kretsen operere som kretsekvivalenten vist nederst i Fig. 30.

Det er en kombinasjon av kontrollsignalene som vi ikke har vurdert. Dersom vi antar at vippen har kontrollsignalene

$reset = set = 1$ har vi en situasjon som ikke kan tillates. Vippen skal i denne situasjonen både settes til 1 og 0 som er selvmotsigende og meningsløst. For ordens skyld kan det være fornuftig å analysere vippen for å se hva som skjer dersom vi ved en feil påtrykker denne ulovelige kombinasjonen av kontrollsignaler.

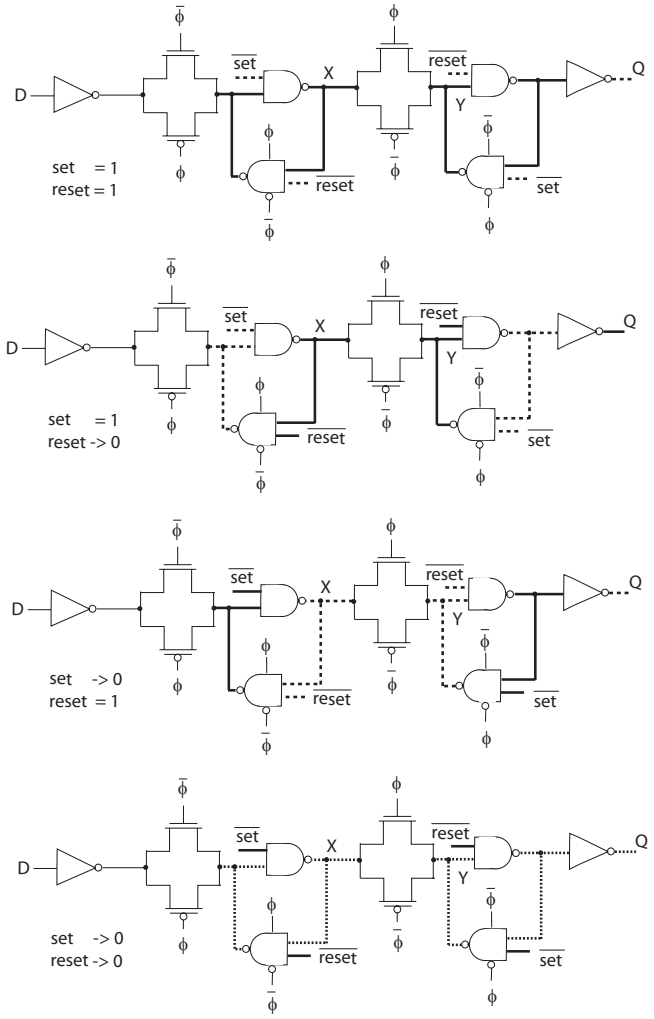

Fig. 31. *Vippe med asynkron reset og set signal. Set = Reset = 1*

I Fig. 31 øverst er det vist hvordan vippen virker når $set = reset = 1$. Vi ser at utgangen Q blir resatt til 0 som i utgangspunktet ligner en vanlig reset. Legg merke til at nodene X og Y blir satt til 1 samtidig. Dette samsvarer ikke med en vanlig reset. Vippens tilstand før eventuell ny sampling av inngangen er avhengig av hvilke av de to kontrollsignalene som skrus av først. Dersom $reset$ blir satt til 0 mens $set = 1$, som vist nest øverst i figuren, vil kretsen oppføre seg som om den ble satt til 1 slik at utgangen Q blir satt til 1. Dersom set blir satt til 0 mens $reset = 1$, som vist nest nederst i figuren, vil kretsen oppføre seg som om den ble satt til 0 slik at utgangen Q forblir 0. I den nederste vippen er det antatt at set og $reset$ endres fra 1 til 0 samtidig. Situasjonen vil da være ukjent, dvs. vi kan ikke forutsi om vippen blir satt til 0 eller 1.

C. Mål

Kunne implementere latcher og vipper med synkron eller asynkron reset.

VI. Latcher og vipper som kan enables

(Kapittel 7.3.5 side 410)

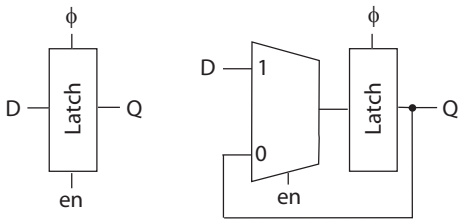

Fig. 32. *Latch med enable realsiert med multiplekser.(FIG7.26)*

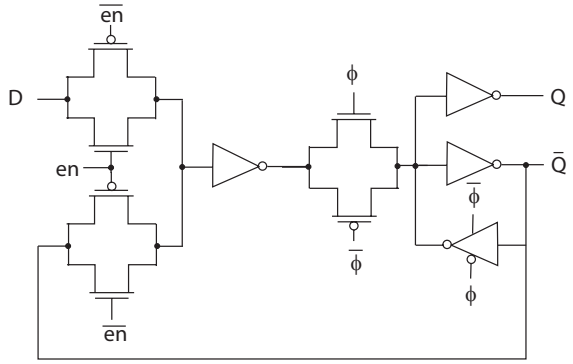

Fig. 33. *Latch med enable realisert med multiplekser.*

I mange tilfeller kan det være hensiktsmessig å kombinere en enable funksjon i latcher og vipper. En latch med *enable* signal er vist i Fig. 32 der en latch er kombinert med en multiplekser. Kretsen har en forholdsvis lang signalvei i tilbakekoblingen via multiplekser som vist i Fig. 33 når $en = 0$ og $\phi = 1$. I tillegg til å bidra med forsinkelse vil multiplekseren bidra med en betydelig arealøkning.

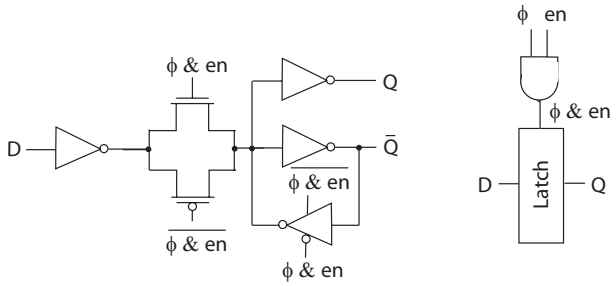

Fig. 34. *Latch med enable funksjon realisert med "clock gating design".(FIG7.26)*

I Fig. 34 er det vist en logisk ekvivalent latch med enable der vi har beholdt den opprinnelige latchen og endret de lokale styresignalene. Dette kalles *clock gating design*. Ved å ANDE en og ϕ til $en\&\phi$ vil kretsen bare sample når både ϕ og en er logisk 1, ellers vil den lokale tilbakekoblingen i latchen sørge for å holde den lagrede verdien.

En vippe med enable signal er vist i Fig. 35 der en vippe er kombinert med en multiplekser.

En tisvarende forenkling som for latchen med enable i Fig. 34 er vist for vippen i Fig. 36.

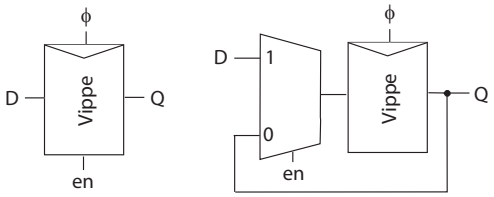

Fig. 35. Vippe med enable realisert med multiplekser.*(FIG7.26)*

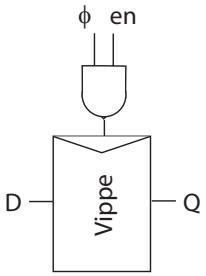

Fig. 36. *Vippe med enable funksjon realisert med "clock gating design". (FIG7.26)*

AND porten som er benyttet for å endre styresignaler til latchen og vippen kan deles av mange sekvenseringselementer og vil derfor ikke bidra med betydelig areal.

A. Mål

Forstå hvordan man kan implementere latcher og vipper med enable signal.

B. Notater

VII. Latcher med logikk
(Kapittel 7.3.6 side 410 - 411)

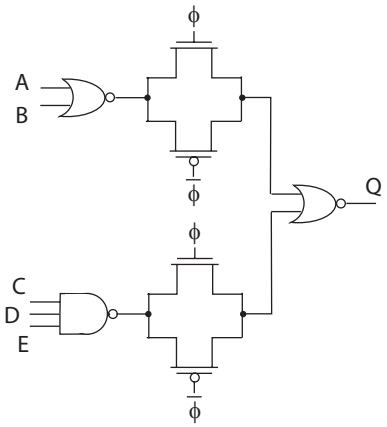

Fig. 37. *Latch med logikk.(FIG7.27)*

Latchene kan lett bygges ut til å prosessere signaler. Et eksempel på en latch med logikk er vist i Fig. 37 der $Q = (A + B) \cdot C \cdot D \cdot E$. Latchen mangler tilbakekobling og er derfor dynamisk.

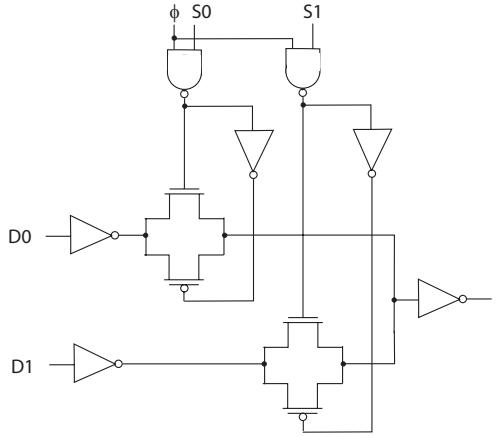

Fig. 38. *Latch med logikk og clock gating. (FIG7.27)*

En multiplekser latch er vist i Fig. 38 der lokale kontrollsignaler blir styrt av $S0$ og $S1$. Denne latchen er også dynamisk.

Statiske latcher og vipper kan også utvides med logikk på samme måte.

A. Mål

Forstå hvordan logikk kan inkluderes i statiske- og dynamiske latcher og vipper.

B. Notater

VIII. Klass semidynamisk vippe (SDFF)

(Kapittel 7.3.7 side 411 - 412)

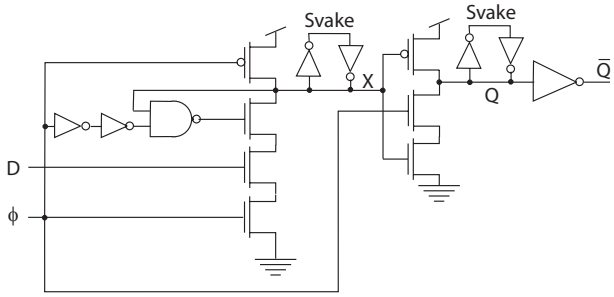

Fig. 39. *Klass semidynamisk vippe. (FIG7.28)*

En *Klass semidynamisk vippe* er vist i Fig. 39. Latchen er en krysning mellom latch styrt av klokkepuls og en vippe.

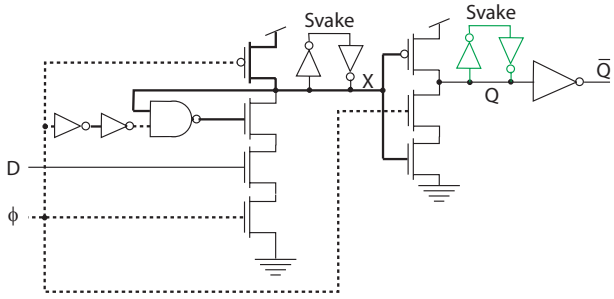

Fig. 40. *Klass semidynamisk vippe der* $\phi = 0$. *(FIG7.28)*

I Fig. 41 er vippen vist når $\phi = 0$. Noden X vil da precharges til 1 og ikke kunne påvirke utgangene Q og $\overline{Q}$. Ved noden Q er det koblet to svake invertere som sørger for å holde verdien på Q og dermed motvirke lekkasje og ladningsdeling.

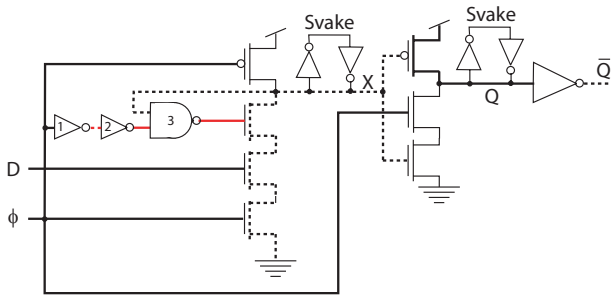

Fig. 41. *Klass semidynamisk vippe der* $\phi = 0$. *(FIG7.28)*

I Fig. 41 er vippen vist når $\phi = 1$. Vi antar nå at kretsen skal latche (sample) inn en ny verdi fra inngangen D. Vi ser at precharge transistoren koblet til X er slik at noden X enten vil holde sin verdi eller trekkes ned til 0. I eksemplet vist i Fig. 41 er $D = 1$. Vi antar nå at utgangen på NAND porten er 1 som vist i Fig. 41 rett før ϕ skifter fra 0 til 1. Det som skjer rett etter denne transisjonen er at noden X trekkes ned til 0 **før** utgangen på NAND porten rekker å reagere på endring av ϕ. Når X er trukket ned til 0 vil tilbakekoblingen via NAND porten sørge for å holde utgangen på NAND porten til 1. Den andre inngangen til NAND porten kommer fra ϕ via to invertere, 1 og 2. Vi må forutsette at ikke dette signalet rekker å skifte fra 0 til før X er trukket ned til 0. Noden Q vil nå bli trukket opp til 1 via en precharge transistor koblet til noden. Utgangen $\overline{Q}$ blir da satt til 0. Vi har nå en situasjon der $\overline{Q} = \overline{D}$ som jo er vippens korrekte funksjon.

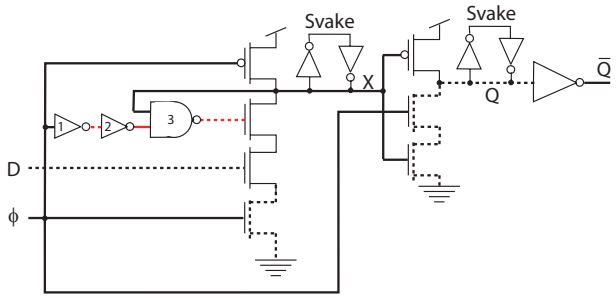

Fig. 42. *Klass semidynamisk vippe. (FIG7.28)*

I Fig. 42 er vippen vist når $\phi = 1$ og $D = 0$. I denne situasjonen er nedtrekkskjeden koblet til X skrudd av slik at X forblir 1. Noden Q vil da trekkes ned via to nMOS transisorer styrt av X og ϕ slik at utgangen $\overline{Q} = 1$. Dette medfører også en situasjon der $\overline{Q} = \overline{D}$ som jo er vippens korrekte funksjon.

A. Mål

Kunne implementere en Klass semidynamisk vippe.

B. Notater

IX. Differensielle vipper

(Kapittel 7.3.8 side 412 - 413)

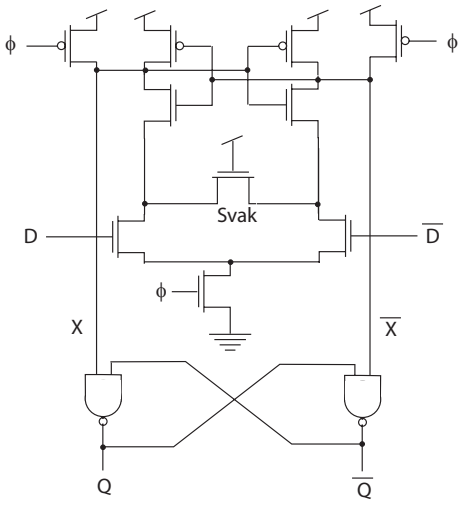

Fig. 43. *Differensiell sense-amplifier vippe. (FIG7.29a)*

En *differensiell* vippe er vist i Fig. 43. Vippen er basert på en såkalt *sense amplifier* som består at et inngangstrinn med D og $\overline{D}$ med en felles transistor med ϕ inngang ned mot GND. De to NAND portene brukes til å holde utgangene stabile.

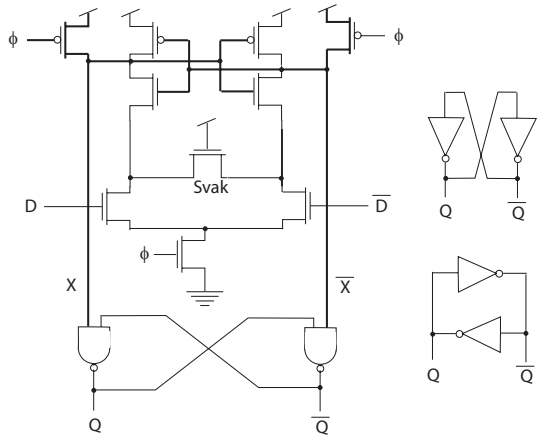

Fig. 44. *Differensiell sense-amplifier vippe med $\phi = 0$. (FIG7.29a)*

Den differensielle vippen med $\phi = 0$ er vist i Fig. 44. Når$\phi = 0$ vil nodene X og $\overline{X}$ precharges til 1 slik at de to NAND portene kan forenkles logisk som vist til høyre for vippen. De to kretsene med krysskoblete invertere er identiske og tilsvarer utgangslatchen på en vanlig vippe.

Den differensielle vippen med $\phi = 1$ er vist i Fig. 45. Vippen skal nå sample inn ny verdi. Som vi ser er vippen fullstendig symmetrisk, vi ser derfor på eksemplet der $D = 1$ som vist i figuren. Noden X blir trukket ned til 0 og dermed $\overline{X}$ opp til 1. Utgangen $Q = D$ som vil holdes til sampling ved neste positive klokkeklanke. Det er en viss tidsforsinkelse gjennom de to NAND portene på utgangene.

Den differenensielle vippen i Fig. 46 har en raskere respons enn vippen med krysskoblete NAND porter. Utgangen blir trukket opp til via en pMOS transistor direkte fra X. De to krysskoblete portene bidrar til å holde verdien i vippen.

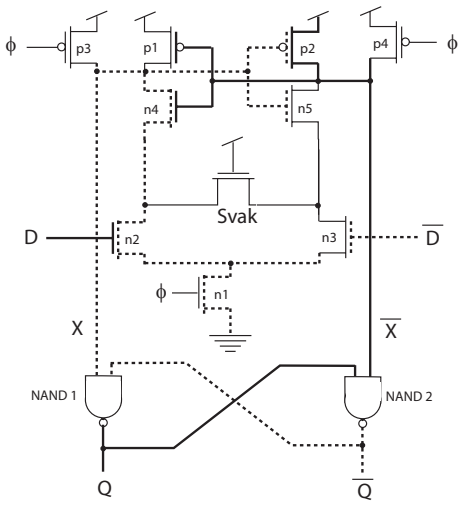

Fig. 45. *Differensiell sense-amplifier vippe $\phi = 1$. (FIG7.29a)*

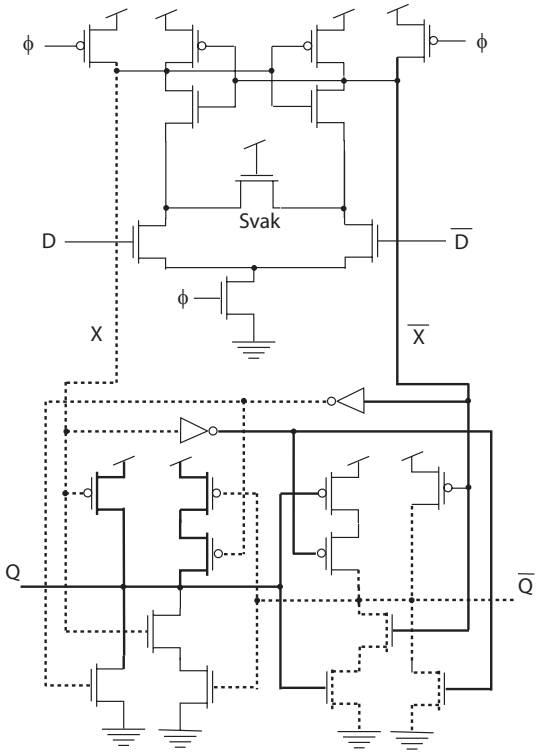

Fig. 46. *Differensiell sense-amplifier vippe. (FIG7.29b)*

A. Mål

Kunne implementere differensielle vipper.

B. Notater

X. Ekte en-fase (TSPC) latcher og vipper
(Kapittel 7.3.9 side 414)

Vanlige latcher og vipper benytter i tillegg til klokkesignal også invertert klokkesignal. I moderne CMOS blir typisk invertert klokkesignal generert lokalt ved latchene eller vippene.

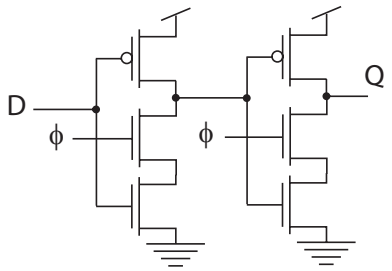

Fig. 47. *Ekte en-fase latch. (FIG7.30a)*

En latch som kun benytter ett klokkesignal er vist i Fig. 50. Vi kaller dette for en *ekte en-fase latch.*

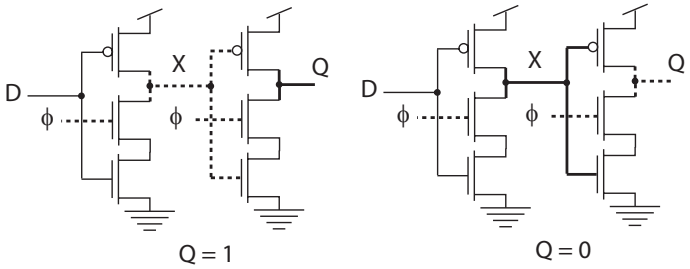

Fig. 48. *Ekte en-fase latch med $\phi = 0$. (FIG7.30a)*

En ekte en-fase latch med $\phi = 0$ er vist i Fig. 49. Latchen skal holde utgangen stabil så lenge $\phi = 0$. Vi ser at nedtrekkene er skrudd AV ved hjelp av ϕ. I utgangspunktet har vi to mulige tilstander; Q var 1 før ϕ skiftet fra 1 til 0 (som vist på venstre side) og Q var 0 opprinnelig (som vist på høyre side). En forutsetnig for at $Q = 1$ er at $X = 0$ som vist til venstre. Når nedtrekket koblet til utgangen ikke kan trekkes ned pga. ϕ vil ikke kretsen kunne endre utgangen så lenge $\phi = 0$. Legg merke til at latchen er dynamisk slik at lekkasje kan påvirke utgangssignalet etter en viss tid. Utgangen holdes høy ved hjelp av en pMOS transistor som er skrudd på forsi $X = 0$. Noden X er ikke drevet og kan endres som følge av lekkasje og dermed påvirge utgangen Q.

Til høyre er en tilstand der $Q = 0$ og $X = 1$. I dette tilfellet er hverken Q eller$X$ drevet og derfor utsatt for lekkasje.

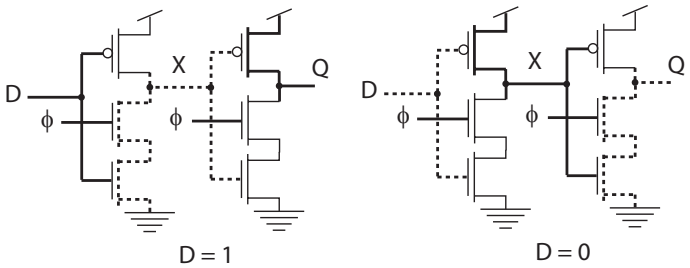

Fig. 49. *Ekte en-fase latch med $\phi = 1$. (FIG7.30a)*

Ved latching av ny verdi (sampling) er er $\phi = 1$ som vist i Fig. 49. I denne situasjonen kan vi (logisk) se bort i fra transistorene styrt av klokke signalet ϕ. Kretsen vil da logisk være to invertere i serie slik at vi alltid får $Q = D$.

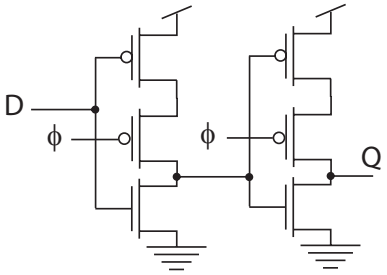

Fig. 50. *Ekte en-fase latch. (FIG7.30b)*

En latch som er følsom for motsatt klokkenivå er vist i Fig. 50. Istedet for å bruke samme latch med invertert klokke signal erstatter vi de klokkestyrte nMOS transistorene med pMOS transistorene og flytter utgangene mellom pMOS og nMOS transistorer.

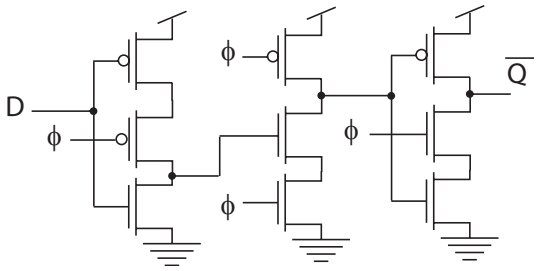

Fig. 51. *Ekte en-fase vippe. (FIG7.30c)*

En ekte en-fase vippe er vist i Fig. 51. Denne vippen er enda enklere en to en-fase latcher klokket i motfase.

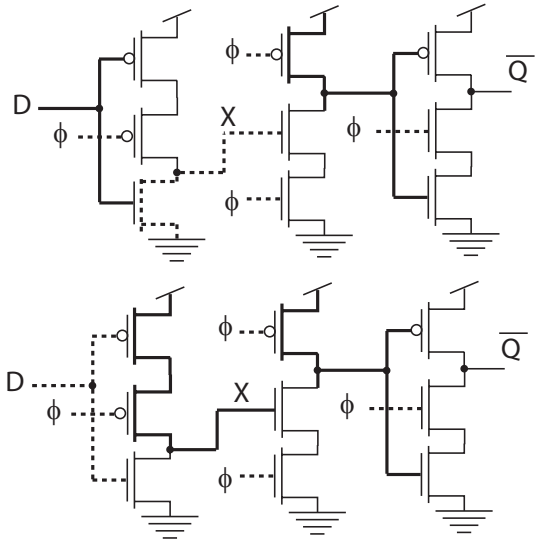

Fig. 52. *Ekte en-fase vippe når $\phi = 0$. (FIG7.30c)*

En-fase vippen nåt $\phi = 0$ og $\phi = 1$ er vist i henholdsvis Fig. 52 og 53.

En-fase latcher og vippe som er beskrevet i dette avsnittet er dynamiske.

A. Mål

Kunne implementere ekte en-fase latcher og vippe.

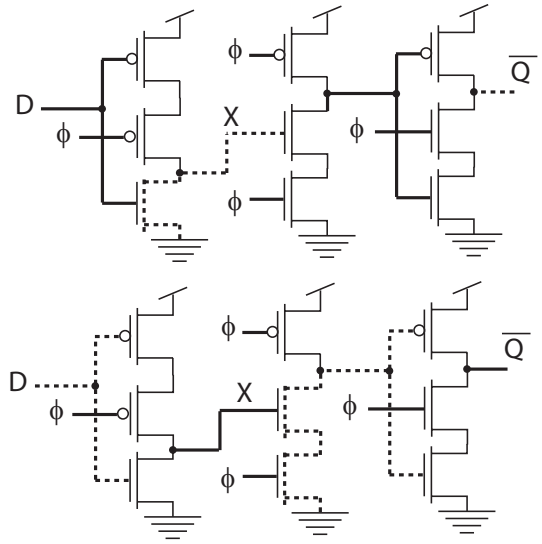

Fig. 53. *Ekte en-fase vippe når $\phi = 1$. (FIG7.30c)*

XI. Indeks

References

[1] Neil H.E. Harris og David Harris"CMOS VLSI DESIGN, A circuit and system perspective" tredje utgave 2005, ISBN: 0-321-26977-2, *Addison Wesley*,

[2] Yngvar Berg, *INF3400 Del 3: Utvidet transistormodell og DC karakteristikk for inverter og pass transistor.*